NTNU – Trondheim
Norwegian University of
Science and Technology

Modal sound propagation in curved horns of rectangular cross-section

Bjørn Kolbrek

Master of Science in Electronics
Submission date: June 2013
Supervisor: Peter Svensson, IET

Norwegian University of Science and Technology
Department of Electronics and Telecommunications

PROJECT TEXT

Sound propagation in horn loudspeakers and other ducts can be simulated using a modal decomposition approach. Most previous investigations with this method have studied straight ducts and often with a circular cross-section. This thesis project should expand the modal decomposition approach to ducts of rectangular cross-section, both straight and curved. Central to the study of such rectangular ducts with curves is the transition between modal descriptions in two connected domains: a straight rectangular duct connected to a single curved element. The project should investigate which approaches can be used for this, and if possible, implement some method.

ABSTRACT

Wave propagation in horns can be simulated by propagating both the plane wave component and the higher order modes down the horn, as described by Kemp [1]. While this is most common for axisymmetric ducts, the method can quite easily be extended to rectangular ducts. The mode functions in this case are simple cosine functions. If the duct is quarter-symmetric, all the asymmetric modes will be zero, and the modal description is simplified. In this thesis, this simpler description is extended to the asymmetric case. The case where the horn expands in one direction and contracts in the other is also treated.

If the rectangular duct is curved with a constant radius, the sound field is most easily described in cylindrical coordinates, with propagation in the angular direction. The radial mode functions will be a combination of Bessel functions. The angular wavenumber depends on the cutoff wavenumbers of the modes in both the z (transverse) direction and the radial direction.

In this thesis, a thorough study of the literature concerning curved rectangular ducts is conducted. The study reveals the many complications in developing and implementing a modal model for duct bends. The two main complications are finding the roots of the so-called dispersion relation, including the difficulty of computing Bessel functions of imaginary order, and in coupling the modal description in the bend to the modal description in the straight sections.

The modal radiation impedance at the mouth of the horn must also be computed. Both for the symmetric and for the asymmetric case, the expression for this impedance contains an oscillatory double integral. For low frequencies (low ka values), the standard Matlab numerical integration routines can be used. For high frequencies, the Numerical Method of Steepest Descent is efficient, with a computation time that is independent of frequency.

The Modal Propagation Method (MPM) has been implemented for both symmetric and asymmetric rectangular ducts, and compared to the Boundary Element Rayleigh Integral Method (BERIM). The MPM shows accuracy comparable to BERIM, but with considerably shorter computation time. The computation time of MPM increases as N^4, where N is the number of modes in each direction. The method has not been implemented for curved ducts or horns.

SAMANDRAG

Bølgjeforplantning i horn kan simulerast ved å la både planbølgjekomponenten og dei høgare ordens modene forplante seg gjennom hornet, slik som Kemp [1] beskriv. Sjølv om dette er mest vanleg for aksesymmetriske horn, kan metoden lett utvidast til rektangulære rør. Modefunksjonane er i dette tilfellet enkle cosinus-funksjonar. Dersom kanalen er kvartsymmetrisk, så vil alle symmetriske modar vere null, og den modale beskrivinga blir enklare. I denne oppgåva blir denne forenkla beskrivinga utvida til asymmetriske kanalar. Tilfellet der hornet utvidar seg i eit plan og trekker seg saman i det andre er også tatt med.

Dersom den rektangulære kanalen er krumma med konstant radius, kan ein enklast beskrive lydfeltet i sylindriske koordinatar, med bølgjeforplantning i vinkelretninga. Dei radielle modefunksjonane vil vere kombinasjonar av Besselfunksjonar. Bølgjetalet i vinkelretning er avhengig av "cutoff"-bølgjetalet til modene både i z-retning (på tvers av kanalen) og i radiell retning.

I denne oppgåva har det blitt gjort eit grundig studium av litteraturen på krumme rektangulære kanalar. Dette studiet avdekker komplikasjonane med å utvikle ein modal modell for krumme kanalar. Hovedkomplikasjonane er å finne røtene til kombinasjonen av deriverte av Besselfunksjonar, inkludert det å rekne ut Besselfunksjonar av imaginær orden, og å kople den modale beskrivinga i den krumme delen til den modale beskrivinga i dei rette delane.

Den modale strålingsimpedansen ved hornmunninga må også reknast ut. Både for det symmetriske og det asymmetriske tilfellet vil uttrykket for denne impedansen innehalde eit oscillatorisk dobbeltintegral. For låge frekvensar (låge ka-verdiar) kan standard Matlabrutiner for numerisk integrasjon brukast. For høge frekvensar, er metoden Numerical Method of Steepest Descent effektiv, med ei reknetid som er uavhengig av frekvens.

Metoden for Modal Bølgjeforplantning (eng: Modal Propagation Method (MPM)) har blitt implementert for både symmetriske og asymmetriske rektangulære kanalar, og sammenligna med Overflate-Element-Rayleigh-Integral-Metoden (eng: Boundary Element Rayleigh Integral Method (BERIM)). MPM har nøyaktighet som kan sammenlignast med BERIM, men har ein god del kortare reknetid. Reknetida for MPM aukar som N^4, der N er talet på modar i kvar retning. Metoden har ikkje blitt implementert for krumme kanalar eller horn.

PREFACE

This is the final work on my Master's degree, and also signifies the end of the first part of the Integrated PhD program at NTNU.

The idea for this work came mainly from my deep interest in horn loudspeakers, and the desire to model more complicated geometries than simple axisymmetric horns. It was also a desire to be able to simulate the performance of the large fullrange horns used in early cinema sound systems like the Vitaphone. These horns are so large, and cover such a large frequency range, that using ordinary element based methods is not possible. While this task is not yet completed, this work is a step in the right direction.

First of all I want to thank my advisor, Peter Svensson, who supervised this work during his sabbatical year. Thanks to Jonathan Kemp for discussions on the Modal Propagation Method. Thanks to Andreas Asheim for providing Matlab code for the Numerical Method of Steepest Descent, which probably saved me one or more days of numerical computations. Finally, thanks to David J. McBean for discussions on horn simulation models, and for correcting all my small typos and mistakes in the manuscript.

Thanks also to André Miede for providing this beautiful LaTeX style package, classicthesis, and to Nicholas Mariette and Ivo Pletikosić for making the LyX port.

Trondheim, June 2013
Bjørn Kolbrek

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LISTINGS

ACRONYMS

MPM Modal Propagation Method

BEM Boundary Elements Method

BERIM Boundary Elements Rayleigh Integral Method

MSD Method of Steepest Descent

1 INTRODUCTION

Both in musical instruments and in electroacoustics, the horn plays an important role. A horn is a duct with changing cross section, usually progressing from a small area (the mouth piece of the instrument, or the loudspeaker unit) to a large area. Over a certain bandwidth, the horn will function as an impedance transformer. This property can be used to increase the efficiency of loudspeakers by greatly increasing the radiation resistance seen by the diaphragm. Another important aspect of a horn is its ability to control directivity. This is a very important and useful property, and is used to a large degree in sound reinforcement applications.

At low frequencies, horns can often be analyzed by the so-called horn equation, see for instance Webster [4]. The equation is applicable provided there are only axial variations of pressure along the horn. However, when the horn cross-section becomes large compared to the wavelength, the waves in the horn will also exhibit variations in the other two dimensions. This is due to the introduction of higher modes [5]. Some of these modes will propagate through the horn and will be radiated into air, others will decay exponentially. But both kinds of modes will take their energy from the fundamental mode.

The presence of higher modes will alter the acoustical impedance seen from the throat (input) end of the horn, the internal pressure distribution, and the radiation pattern from the mouth (output) end. Accurate calculation of the loading and radiating properties of horns can only be performed if these higher order modes are taken into account.

Several methods for including higher order modes in horns have been proposed. In one method, the duct is approximated by a number of steps, using a series of cylindrical sections. This approach was probably first implemented by Alfredson [6], who used an iterative technique. Another method, which is similar to the one described by Pagneux [7], is described by Kemp [1]. The method has mainly been used for horn musical instrument simulations, but a few researchers have also used it for loudspeaker simulations, see Shindo et. al [8] and Schuhmacher and Rasmussen [9]. This method includes both means for propagating modes along a uniform cylinder, and across a discontinuity.

This method, named the Modal Propagation Method (MPM), was implemented and compared to the Boundary Elements Method (BEM) and the Boundary Elements Rayleigh Integral Method (BERIM) [10] in

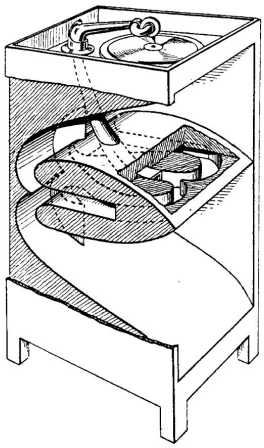


Figure 1: Victor Orthophonic horn (from Maxfield [2])

a preliminary study, Fall 2011 [11]. In a project, Spring 2012[12], the efficiency and accuracy of this method was studied. It was shown that good accuracy could be obtained even with few modes, and that the required number of modes for a given accuracy depended on the frequency of operation and physical dimensions of the horn.

These two studies were, however, only concerned with round, axisymmetric horns. While round horns are in use in many of the domestic horn loudspeakers available, and also in a few public address speakers, they are in general outnumbered by horns of rectangular shape, or horns with shapes that do not conform to any simple coordinate system.

For low frequency horns, still another factor comes into play; horns for low frequency use tend to be very large, and are consequently folded into a suitable shape. The folding can be done in many different ways, from gentle curving to sharp 180 degree bends. These horns almost invariably have rectangular shape.

This thesis will deal with the following issues in the simulation of horns:

- Rectangular, quarter-symmetric straight horns. This has already been covered by Kemp, but the method will be implemented and compared against BERIM.
- Rectangular, asymmetric straight horns.
- Rectangular horns curved in one dimension.

The reason for including the study of asymmetric horns, is that the asymmetric modes are also required in curved horns, due to the asymmetry introduced by the bend.

The sound field in a curved horn is significantly more complex than that of a straight horn. Therefore, it is not surprising that the literature on curved horns is not as extensive as that for straight horns.

Figure 2: Western Electric 15B cinema horn (Courtesy of AT&T Archives and History Center)

And most of the work is to be found outside electroacoustics. Some work has been done on curved rectangular and circular waveguides both in acoustics and electromagnetics, and more recently, also in quantum mechanics.

In the early history of electroacoustics, curved horns were used for the full frequency range, for instance in gramophones, like the Victor Orthophonic (see Figure 1), which used a bifurcated folded horn path, or in the early full-range cinema horns, like the Western Electric 15A and B (see Figure 2), which used simple "curling". This horn was a further development of an earlier hor which had been curled both ways, something that was found to impair the speech quality significantly.

The computational possibilities at the time were not sufficiently advanced to analyze these horns in detail, so simple rules-of-thumb, or experiments, were used. Wilson [13, 14] and Voigt [15] give methods for curving or folding horns to minimize the impact of the folding. Chief engineer at Victor Talking Machine Company, S. T. Williams, published a study of the effect of curving, leaks and other perturbations of horns, in 1926 [16].

As soon as multi-way speaker systems became the norm in the mid-1930s, folded horns were reserved for low frequency use, commonly below 2-400Hz, and straight horns were used for the higher frequencies. This reduced the need for a detailed mathematical analysis of these horns, but re-entrant folded horns were still in use for paging service. Tanaka [17] describes such horns, and Carlisle [18] offers a method to improve the response. No mathematical analysis of the folding is given.

Bruce Edgar noticed, during the design of a folded bass horn [19], that certain types of bends or folds significantly altered the frequency response. He analyzed the bends as acoustical masses inserted in between transmission line segments, with a reference to work by Cummings [20] on 180° folds in rectangular ducts.

A folded horn was analyzed by Bright et al. [21], comparing BEM and a method using a series of one-dimensional exponential segments [22]. Backman [23] attempts to construct one-dimensional models of bends based on a comparison with FEM solutions, but the results seem to be hard to generalize to arbitrary bends.

In horn musical instrument acoustics, the effect of bends on the resonance frequencies of brass instruments has been studied at length. An early reference on the effective length and diameter of a toroidal bend is Nederveen [24], more recent ones are Keefe and Benade [25] and Ting and Miksis [26]. None of these deal with multimodal propagation. This is however done by Felix and Pagneux, who consider both two-dimensional [27] and three-dimensional toroidal [28] bends. The case of toroidal bends is extended further by Braden [29]. We will take a closer look at this method later.

In general acoustics, much of the work on bends and folding has been done in connection with ventilation ducts. A typical case found in these studies is a curved rectangular section of constant width, height and radius of curvature, connected between two straight ducts of the same width and height, as shown in Figure 6. One of the early studies of modes in curved ducts of this type was done by Grigor'yan [30], who also cites previous work on electromagnetic waveguides. Osborne [31, 32, 33] and Cummings [3, 20, 34] cover rectangular bends, and the coupling to straight ducts using a modal description.

Furnell and Bies [35, 36] describe an approximation to the acoustic field in a curved, elliptic waveguide. Firth and Fahy investigate bends in round tubes [37], as do Felix and Pagneux, and Braden, as mentioned above. Sarigül [38] investigates right-angle bends of circular cross-section by BEM.

Furnell and Bies also present a scattering matrix approach to the problem of a ducting system consisting of several bends and straight sections [35]. A similar method is described by Kim and Ih [39]. Felix et al. [27] also present a analytical representation of the reflection and transmission matrices valid for bends of constant curvature and width.

While many of the authors use a modal description of the sound field, Tam [40] (with later comments and corrections by Furnell [41]) uses the Galerkin method. Cabelli [42] and Cabelli and Shepherd [43] use a finite difference solution to the two-dimensional Helmholtz equation, and also investigate the effect of turning vanes.

The problem of curved ducts lined with absorbing material has also been studied, for instance by Ko [44] and Felix and Pagneux [45].

The problem of curved rectangular waveguides has also received much attention in calculation of electromagnetic waveguides, see for instance Sorolla [46], and also in the analysis of quantum waveguides [47]. Many of the numerical methods required in the analysis of the dispersion relation (see next chapter) can be found in papers in these subjects.

2

THEORY

As has been pointed out both in previous investigations [11, 12], and by other authors [48, 49], the classical, one-dimensional horn theory, based on what is known as Webster's horn equation[1], is not able to predict the sound field radiated by a general horn. Putland [49] has shown that the horn equation is accurate for three horn types; straight tubes, parabolic horns (when two sides are parallel, the other two expanding in a conical fashion) and axisymmetric conical horns[2]. The horn equation here corresponds to the fundamental mode of propagation in Cartesian, cylindrical and spherical coordinates, respectively. Still, diffraction at the mouth of the horn can not be properly modeled, and the radiated field is still not possible to predict.

By instead using the full wave equation, and letting the horn walls follow the coordinate surfaces of a coordinate system where the wave equation is separable, analytical solutions, including higher order modes of propagation, can be found. There are, however, only eleven coordinate systems where the wave equation is separable, and very few of them have surfaces that give useful horn contours [48].

A solution to this problem is to divide the horn into small sections that each have simple mode functions, and then to couple the higher order modes between the sections. A common choice is to use cylindrical sections of varying radius. This method, developed by Pagneux et al. [7] and Kemp [1], has been investigated in [11, 12].

Not all horns are circular, however, and Kemp has shown how the method can be extended to symmetrical rectangular horns. A summary of the theory, and an extension to asymmetric horns, will be given in this chapter.

2.1 MODAL PROPAGATION IN RECTANGULAR HORNS

The pressure and velocity in a uniform duct can be expressed as a weighted sum of allowable modes [52, 53]. In a rectangular duct, there will be n_x by n_y nodal lines, and $(n_x, n_y) = (0, 0)$ represents the plane wave mode.

1 Although the equation in question was derived and discussed by Bernoulli, Lagrange and Euler [50].

2 The parabolic horn is sometimes called a cylindrical waveguide, since it can be described exactly in cylindrical coordinates when the shape is as described above. For the same reason, an axisymmetric conical horn is sometimes called a spherical waveguide[51].

Pair no.	Modes	Pair no.	Modes
1	(0,0)	9	(2,2)
2	(0,1)	10	(0,3)
3	(1,0)	11	(3,0)
4	(1,1)	12	(1,3)
5	(0,2)	13	(3,1)
6	(2,0)	14	(2,3)
7	(1,2)	15	(3,2)
8	(2,1)	16	(3,3)

Table 1: Mode pairs sorted in increasing order

Along the duct, assuming propagation in the z direction, the pressure can be expressed as a sum of all the modes:

$$p(x,y,z) = \sum_{n=0}^{\infty} P_n(z)\psi_n(x,y) \tag{1}$$

where P_n is the pressure profile along the tube, and ψ_n is the pressure profile in the (x,y) plane. It is advantageous to separate ψ_n into two parts, one dependent on x, and the other on y:

$$\psi_n = \phi_{n_x}\sigma_{n_y} \tag{2}$$

Assuming hard walls in the horn, the boundary conditions in the x and y directions are

$$\frac{\partial \psi_n}{\partial x} = 0, \quad x = a_-, a_+ \tag{3}$$

$$\frac{\partial \psi_n}{\partial y} = 0, \quad y = b_-, b_+ \tag{4}$$

where a_- and a_+ are the locations of the walls in the x direction, and correspondingly in the y direction for b_- and b_+.

Since there are modes in two directions in the case of rectangular horns (and also circular horns with non-axisymmetric sound fields), these modes have to be sorted so that the matrices still are two-dimensional. This means that a vector of mode pairs needs to be generated, preferably with the modes sorted in increasing order. An example is given in Figure 1. This example also shows that for four modes in each direction, a total of 16 mode pairs are generated. The size of the matrices are therefore much larger than in the case of axisymmetric horns, for the same accuracy.

2.2 SYMMETRIC HORNS

In symmetric horns, we assume that the horn is symmetric both horizontally and vertically across the central axis, i.e. $a_- = -a$, $a_+ = a$, $b_- = -b$ and $b_+ = b$. The wave functions in Eq. (2) then become

$$\phi_{n_x} = \begin{cases} 1 & : n_x = 0 \\ \sqrt{2}\cos\left(\frac{n_x \pi x}{a}\right) & : n_x > 0 \end{cases} \tag{5}$$

$$\sigma_{n_x} = \begin{cases} 1 & : n_y = 0 \\ \sqrt{2}\cos\left(\frac{n_y \pi y}{b}\right) & : n_y > 0 \end{cases} \tag{6}$$

The corresponding eigenvalues are

$$\alpha_n = \sqrt{\left(\frac{n_x \pi}{a}\right)^2 + \left(\frac{n_y \pi}{b}\right)^2} \tag{7}$$

In the z-direction, the pressure can be expressed as

$$P_n(z) = A_n e^{-ik_n z} + B_n e^{ik_n z} \tag{8}$$

where

$$k_n = \pm\sqrt{k^2 - \alpha_n^2} \tag{9}$$

is the wavenumber of the nth mode in the axial direction and k is the free space wavenumber. We can see that the axial wavenumber will for certain values of α_n^2 be imaginary, and the propagation in z direction will be evanescent (exponentially damped). The frequency where k_n becomes real is called the cut-off (or sometimes cut-on) frequency of the corresponding mode, $k_c = \alpha_n$.

From Eq. (9) we have that the wavenumber of the nth mode is

$$k_n = \pm\sqrt{k^2 - \left(\frac{n_x \pi}{a}\right)^2 - \left(\frac{n_y \pi}{b}\right)^2} \tag{10}$$

We must choose the right signs of the square root to get the correct behavior of the waves. When the mode is propagating, we must have the normal equation of propagation, so we must take the positive root. When the mode is in cutoff, the modal wavenumber is purely imaginary, and the wave should be exponentially damped with distance. We must therefore take the negative root. In summary

$$k_n = \begin{cases} -\sqrt{k^2 - \left(\frac{n_x \pi}{a}\right)^2 - \left(\frac{n_y \pi}{b}\right)^2} & : k^2 < (n_x \pi/a)^2 + (n_y \pi/b)^2 \\ \sqrt{k^2 - \left(\frac{n_x \pi}{a}\right)^2 - \left(\frac{n_y \pi}{b}\right)^2} & : k > (n_x \pi/a)^2 + (n_y \pi/b)^2 \end{cases} \tag{11}$$

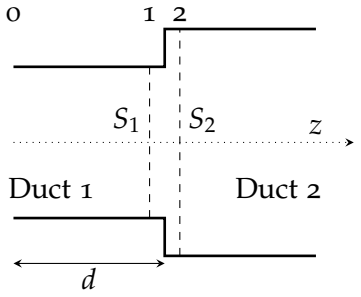


Figure 3: Two ducts joined by a discontinuity

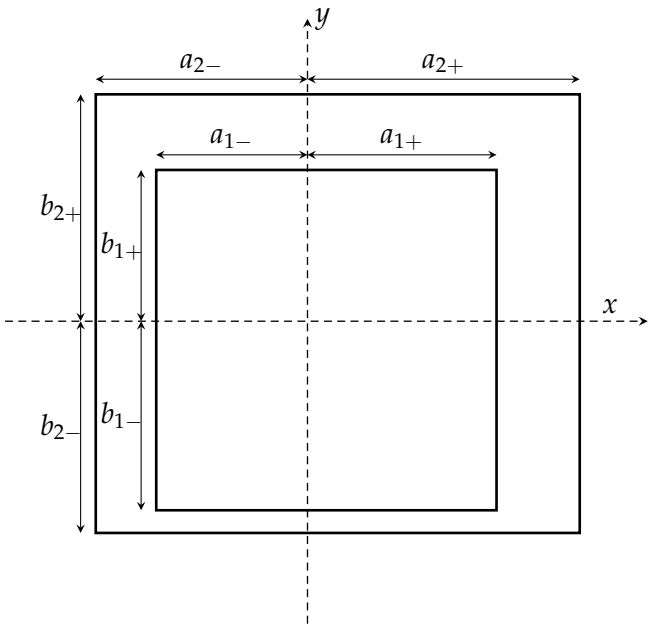


Figure 4: General discontinuity between two rectangular ducts

2.2.1 *Propagation across a discontinuity*

The geometry is illustrated in Figure 3 and Figure 4. For two symmetric ducts, $a_- = -a$, $a_+ = a$, $b_- = -b$ and $b_+ = b$. When the wave propagates across the discontinuity, there must be continuity of pressure and velocity right before (at point 1) and right after the discontinuity (at point 2). But the nth mode in duct 1 will not match the nth mode in duct 2, so the pressure field in duct 2 must be made up of a new sum of modes. Every mode in duct 1 excites a series of modes in duct 2, this is known as modal coupling or mode conversion. Kemp [1] derives expressions for this coupling.

If $\vec{P}^{(1)}$ is the vector of modal pressure amplitudes at point 1, and $\vec{P}^{(2)}$ is the vector of modal pressure amplitudes at point 2, the vectors are related by a matrix F so that

$$\vec{P}^{(1)} = F\vec{P}^{(2)} \tag{12}$$

where the matrix elements are defined as

$$F_{nm} = \frac{1}{S_1} \int_{S_1} \psi_n^{(1)} \psi_m^{(2)} dS \tag{13}$$

$\psi_n^{(1)}$ gives the mode shapes in duct 1, and $\psi_m^{(2)}$ gives the mode shapes in duct 2. This relation holds if $a_1 < a_2$, where a_1 and a_2 are the widths of the two ducts (and correspondingly for the heights $b_1$and b_2).

If $a_1 > a_2$, and $b_1 > b_2$, the relation is

$$\vec{P}^{(2)} = V\vec{P}^{(1)} \tag{14}$$

where the matrix elements are defined as

$$V_{nm} = \frac{1}{S_2} \int_{S_2} \psi_n^{(2)} \psi_m^{(1)} dS \tag{15}$$

For a rectangular duct, it is most convenient to express Eq. (13) as an *element-wise* multiplication of two terms:

$$\begin{aligned} F_{nm}(\beta_x, \beta_y) &= \frac{1}{S_1} \int_{S_1} \psi_n^{(1)} \psi_m^{(2)} dS \\ &= \frac{1}{2a_1} \int_{-a_1}^{a_1} \phi_{n_x}^{(1)} \phi_{m_x}^{(2)} dx \frac{1}{2b_1} \int_{-b_1}^{b_1} \sigma_{n_y}^{(1)} \sigma_{m_y}^{(2)} dy \\ &= X_{n_x m_x} Y_{n_y m_y} \end{aligned} \tag{16}$$

where $\beta_x = a_1/a_2$ and $\beta_y = b_1/b_2$, and

$$X_{n_x m_x} = \begin{cases} 1 & : \quad n_x = m_x = 0, \\ \sqrt{2}\text{sinc}\left(m_x \pi \beta_x\right) & : \quad n_x = 0,\ m_x > 0, \\ 2\text{sinc}\left(\pi(m_x\beta_x - n_x)\right) \frac{m_x\beta_x}{m_x\beta_x + n_x} & : \quad n_x > 0 \end{cases} \tag{17}$$

$$Y_{n_y m_y} = \begin{cases} 1 & : \quad n_y = m_y = 0, \\ \sqrt{2}\text{sinc}\left(m_y \pi \beta_y\right) & : \quad n_y = 0,\ m_y > 0, \\ 2\text{sinc}\left(\pi(m_y\beta_y - n_y)\right) \frac{m_y\beta_y}{m_y\beta_y + n_y} & : \quad n_y > 0 \end{cases} \tag{18}$$

The V matrix then becomes

$$V_{nm} = F_{nm}(1/\beta_x,\ 1/\beta_y) \tag{19}$$

For values of β close to one, there is a simpler and faster method, also given by Kemp. Here, $X_{n_x m_x}$ and $Y_{n_y m_y}$ are approximated by

$$X_{n_x m_x} = \begin{cases} 1 & : \quad n_x = m_x = 0, \\ 1 - \frac{1}{2}\epsilon_x & : \quad n_x = m_x \neq 0, \\ \sqrt{2}\,(-1)^{m_x}\,(-\epsilon_x) & : \quad n_x \neq m_x,\ n_x = 0, \\ 2\,(-1)^{m_x+n_x}\,(-\epsilon_x)\,\frac{m_x}{m_x^2+n_x^2} & : \quad n_x \neq m_x,\ n_x > 0, \end{cases} \tag{20}$$

$$Y_{n_y m_y} = \begin{cases} 1 & : \quad n_y = m_y = 0, \\ 1 - \frac{1}{2}\epsilon_y & : \quad n_y = m_y \neq 0, \\ \sqrt{2}\,(-1)^{m_y}\,(-\epsilon_y) & : \quad n_y \neq m_y,\ n_y = 0, \\ 2\,(-1)^{m_y+n_y}\,(-\epsilon_y)\,\frac{m_y}{m_y^2+n_y^2} & : \quad n_y \neq m_y,\ n_y > 0, \end{cases} \tag{21}$$

where $\epsilon_x = (a_2 - a_1)/a_1$ and $\epsilon_y = (b_2 - b_1)/b_1$.

Volume velocity can be propagated across the discontinuity as

$$\vec{U}^{(2)} = F^T \vec{U}^{(1)},\ S_1 < S_2 \tag{22}$$

$$\vec{U}^{(1)} = V^T \vec{U}^{(2)},\ S_1 > S_2 \tag{23}$$

The generation of higher modes through a horn will depend on the rate of change of slope of the walls. In the stepped horn model used in this thesis, one can understand this by looking at what happens at a discontinuity. Between each discontinuity, the modes are uncoupled. But at the steps, the modes are coupled through the F matrix, Eq. (13), since the off-diagonal terms are non-zero. This coupling depends on both β_x, β_y , and on the eigenvalues in the two tubes.

2.3 ASYMMETRIC HORNS

In symmetric horns, only symmetric modes propagate. If asymmetry is introduced, for instance if the horn expands more upwards than downwards, referred to the horn axis, asymmetric modes will be introduced. In all implementations of the MPM found in the literature, symmetry has been assumed. This is in many cases allowable, as the large majority of straight, rectangular horns are quarter-symmetric. If the horn is curved, however, asymmetric modes must be taken into account. This section gives the equations for the asymmetric case.

The wave functions in Eq. (2) for asymmetric ducts become

$$\phi_{n_x} = \begin{cases} 1 & : n_x = 0 \\ \sqrt{2}\cos\left(\frac{n_x \pi (x - a_-)}{a_+ - a_-}\right) & : n_x > 0 \end{cases} \tag{24}$$

$$\sigma_{n_x} = \begin{cases} 1 & : n_y = 0 \\ \sqrt{2}\cos\left(\frac{n_y\pi(y-b_-)}{b_+-b_-}\right) & : n_y > 0 \end{cases} \tag{25}$$

The corresponding eigenvalues are

$$\alpha_n = \sqrt{\left(\frac{n_x\pi}{a_+-a_-}\right)^2 + \left(\frac{n_y\pi}{b_+-b_-}\right)^2} \tag{26}$$

Note that the mode numbers are not the same as for a symmetric duct; in the symmetric case all odd modes are zero, and can consequently be removed. Therefore n_x and n_y in Eq. (7) will correspond to $2n_x$ and $2n_y$ in Eq. (26).

The equations for propagation along the duct are the same as given in Eq. (8) and Eq. (11), apart from the change in eigenvalues.

2.3.1 *Propagation across a discontinuity*

The geometry is illustrated in Figure 4. It is again most convenient to express the F matrix as an element-wise product of two terms:

$$\begin{aligned} F_{nm} &= \frac{1}{S_1}\int_{S_1}\psi_n^{(1)}\psi_m^{(2)}dS \\ &= \frac{1}{a_{1+}-a_{1-}}\int_{a_{1-}}^{a_{1+}}\phi_{n_x}^{(1)}\phi_{m_x}^{(2)}dx\,\frac{1}{b_{1+}-b_{1-}}\int_{b_{1-}}^{b_{1+}}\sigma_{n_y}^{(1)}\sigma_{m_y}^{(2)}dy \\ &= X_{n_xm_x}Y_{n_ym_y} \end{aligned} \tag{27}$$

where $X_{n_xm_x}$ and $Y_{n_xm_x}$ now are

$$X_{n_xm_x} = \begin{cases} 1 & : \ n_x = m_x = 0, \\ \sqrt{2}\mathrm{sinc}\left(\frac{m_x\pi}{2}\beta_{x,t}\right)\cos\left(\frac{m_x\pi}{2}\beta_{x,a}\right) & : \ n_x = 0,\ m_x > 0, \\ \mathrm{sinc}\left(\frac{\pi}{2}\left(n_x - m_x\beta_{x,t}\right)\right)\cos\left(\frac{\pi}{2}\left(n_x - m_x\beta_{x,a}\right)\right) & \\ +\mathrm{sinc}\left(\frac{\pi}{2}\left(n_x + m_x\beta_{x,t}\right)\right)\cos\left(\frac{\pi}{2}\left(n_x + m_x\beta_{x,a}\right)\right) & : \ n_x > 0 \end{cases} \tag{28}$$

$$Y_{n_ym_y} = \begin{cases} 1 & : \ n_y = m_y = 0, \\ \sqrt{2}\mathrm{sinc}\left(\frac{m_y\pi}{2}\beta_{y,t}\right)\cos\left(\frac{m_y\pi}{2}\beta_{y,a}\right) & : \ n_y = 0,\ m_y > 0, \\ \mathrm{sinc}\left(\frac{\pi}{2}\left(n_y - m_y\beta_{y,t}\right)\right)\cos\left(\frac{\pi}{2}\left(n_y - m_y\beta_{y,a}\right)\right) & \\ +\mathrm{sinc}\left(\frac{\pi}{2}\left(n_y + m_y\beta_{y,t}\right)\right)\cos\left(\frac{\pi}{2}\left(n_y + m_y\beta_{y,a}\right)\right) & : \ n_y > 0 \end{cases} \tag{29}$$

Here the ratio

$$\beta_{x,t} = \frac{a_{1+}-a_{1-}}{a_{2+}-a_{2-}} \tag{30}$$

describes the symmetrical part of the field at the junction, and corresponds to β_x in the symmetrical case.

$$\beta_{x,a} = \frac{a_{1+} + a_{1-} - 2a_{2-}}{a_{2+} - a_{2-}} \tag{31}$$

describes the asymmetry.

A full derivation of the equations is given in Appendix A.

2.4 CURVED HORNS

The curved part of the horn can be approximated by sections of bent rectangular ducts with wave propagation in the angular direction. The curved sections replace the straight sections through the curved part of the horn, and are coupled with corresponding F-matrices. An illustration is shown in Figure 5.

2.4.1 *Propagation through rectangular bends*

The wave equation in cylindrical coordinates is given as

$$\left(\frac{\partial^2}{dr^2} + \frac{1}{r}\frac{\partial}{\partial r} + \frac{1}{r^2}\frac{\partial^2}{\partial \phi^2} + \frac{\partial^2}{\partial z^2}\right) p + k^2 p = 0 \tag{32}$$

If we assume a solution of the form [3]

$$p(r, z, \phi, t) = PR(z)Z(z)e^{-ik_\varphi \phi} \tag{33}$$

we can solve Eq. (32) by the separation of variables to get

$$Z(z) = \cos(k_z z) + B \sin(k_z z) \tag{34}$$

$$R(r) = J_{k_\varphi}(k_r r) + AY_{k_\varphi}(k_r r) \tag{35}$$

where J_{k_φ} and Y_{k_φ} are Bessel functions. See Appendix B for more details. The wavenumbers in r and z directions are coupled as

$$k_r^2 = k^2 - k_z^2 \tag{36}$$

and the angular wavenumber k_φ is dependent on the two other wavenumbers, and fixed by the boundary conditions at the duct walls. The case of the angular wavenumber is a bit special, though, as it is dimensionless, unlike k_r and k_z.

For a circular duct, with propagation in the z-direction, k_φ is limited to integer values. Here, propagation is the angular direction, and k_φ can take any value that is determined by the boundary conditions at the walls.

The boundary conditions in r- and z-directions are

$$\frac{\partial p}{\partial r} = 0 \text{ at } R = R_1 \text{ and } R = R_2 \tag{37}$$

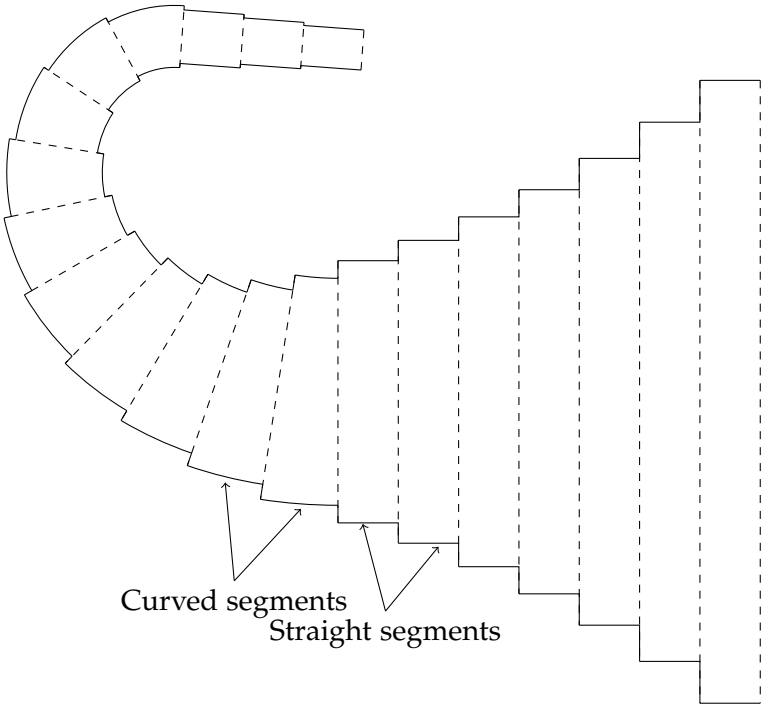


Figure 5: A curved horn made up of straight and curved sections

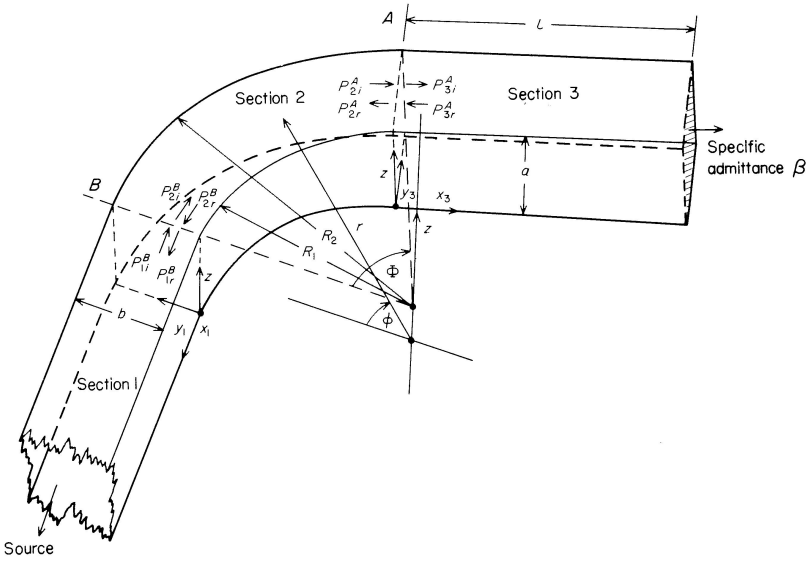


Figure 6: Geometry of a bend connected to two straight ducts (after Cummings [3])

$$\frac{\partial p}{\partial z} = 0 \text{ at } z = 0 \text{ and } z = a \tag{38}$$

where a is the height of the duct, and R_1 and R_2 are the inner and outer radii, respectively, see Figure 6. This gives for the z-direction

$$Z(z) = \cos(k_z z) \tag{39}$$

where

$$k_z = \frac{n_z \pi}{a} \tag{40}$$

For the radial and angular directions, things are a bit more complicated. The constant A in Eq. (35) can be found to be

$$A = -\frac{J'_{k_\varphi}(k_r R_1)}{Y'_{k_\varphi}(k_r R_1)} \tag{41}$$

and the angular wave number is found from solving for k_φ in the eigenequation (also called the dispersion relation)

$$J'_{k_\varphi}(k_r R_1) Y'_{k_\varphi}(k_r R_2) - J'_{k_\varphi}(k_r R_2) Y'_{k_\varphi}(k_r R_1) = 0 \tag{42}$$

k_r can be both real and imaginary, and k_φ can also be both real and imaginary, since the radial modes can also be in cutoff.

If s corresponds to the $(s+1)$th root of Eq. (42), $s = 0$ indicating the first root[3] etc., we can introduce the mode function Ψ defined as

$$\Psi_{n_z s}(r, z) = R_s(r) Z_{n_z}(z) \tag{43}$$

where

$$R_s(r) = J_{k_\varphi^{n_z s}}(k_r r) + -\frac{J'_{k_\varphi^{n_z s}}(k_r R_1)}{Y'_{k_\varphi^{n_z s}}(k_r R_1)} Y_{k_\varphi^{n_z s}}(k_r r) \tag{44}$$

where k_φ^s is the $(s+1)$th root of Eq. (42) and k_r is the radial wavenumber given by Eq. (36) and Eq. (40). We now have that

$$Z_{n_z}(z) = \cos\left(\frac{n_z \pi z}{a}\right) \tag{45}$$

and the total expression for the sound field is (time dependence implied):

$$p(r, z, \phi) = \sum_{n_z=0}^{\infty} \sum_{s=0}^{\infty} P_{n_z s} \Psi_{n_z s}(r, z) \left(e^{-i k_\varphi^{n_z s} \phi} + C_{n_z s} e^{i k_\varphi^{n_z s} \phi} \right) \tag{46}$$

where C is the relative amplitude of the reflected wave, and P is the modal amplitude. P and C can be combined to yield the modal amplitudes in each direction, $A^+_{n_z s}$ and $A^-_{n_z s}$:

$$p(r, z, \phi) = \sum_{n_z=0}^{\infty} \sum_{s=0}^{\infty} \Psi_{n_z s}(r, z) \left(A^-_{n_z s} e^{-i k_\varphi^{n_z s} \phi} + A^+_{n_z s} e^{i k_\varphi^{n_z s} \phi} \right) \tag{47}$$

3 The nomenclature differs somewhat between the papers, and it is not always clear which index refers to which mode.

2.4.2 *The dispersion relation*

The angular wavenumber is found from solving for k_φ in the dispersion relation, Eq. (42). This is an implicit equation, where the angular wavenumber is found as the order of the Bessel functions satisfying the equation for a fixed argument. Eq. (42) can also be expressed as

$$J'_\nu(x)Y'_\nu(\rho x) - J'_\nu(\rho x)Y'_\nu(x) = 0 \tag{48}$$

which is a form that is found in several references.

We know that

- k_r can be both real and imaginary
- k_φ can be both real and imaginary
- Assuming no losses (other than radiation), none of the wavenumbers can be complex.

Cochran discusses the properties of Eq. (48) in references [54, 55, 56], and Chapman also discusses the properties of this equation and its approximations in [57]. Horvat and Prosen discuss a similar equation, $J_\nu(kr)Y_\nu(k) - J_\nu(k)Y_\nu(kr) = 0$, in [47], in connection with bends in quantum waveguides. This equation would correspond to Dirichlet boundary conditions at the walls of the bend [46].

In [54], the following properties of Eq. (48) are given:

- For real and positive x, Eq. (48) has a finite number of ν-roots, the rest are purely imaginary.
- There are infinitely many imaginary ν-roots for real and positive x.
- There are no complex x-roots for real ν.
- There are no complex ν-roots for real x.
- Eq. (48) is even in both x and ν.

Cochran also shows that the roots of Eq. (48) are identical to those of

$$g_\nu^{(\prime)} = J'_{i\mu}(x)J'_{-i\mu}(\rho x) - J'_{i\mu}(\rho x)J'_{-i\mu}(x) \tag{49}$$

This equation can, for higher orders, be approximated by

$$g_{\nu\to\infty}^{(\prime)} = -\left(\frac{i\mu}{\rho\pi x^2}\right) e^{\mu\pi}\left[\sin(\mu\log\rho) + O(\mu^{-1})\right] \tag{50}$$

The imaginary zeros are given asymptotically by

$$\mu = \frac{s\pi}{\log\rho} + O(s^{-1}) \tag{51}$$

For imaginary x, there do not, from numerical experiments, seem to be any real ν-roots.

2.4.3 *Coupling to straight rectangular ducts*

An important note here: the cylindrical coordinate system will make z the direction of the thickness of the duct, the dimension perpendicular to the curving. In the theory for straight rectangular horns, z is the direction of wave propagation. This may lead to some confusion when coupling the two domains. In the following theory, we will therefore assume the propagation to be in the y-direction, the horn will be curved in the x-direction (vertical), and there will be no curving in the horizontal (z) direction. The indices for the straight sections will be re-labeled n_x and n_z. For the modal eigenfunctions in the straight duct, we will use Eq. (2), where the index n contains what is now n_x and n_z. To not make the integrals in this section too complex, we will set the dimensions of the ducts to be a and b, so that $a_- = 0$, $a_+ = a$ and so on.

The discontinuities between straight and bent ducts are labeled A and B in Figure 6. Across these discontinuities, the pressure and velocity must be continuous.

Cummings [3] simplifies the problem somewhat, by considering only plane waves incident at B, and that only the plane wave, or (0,0) mode, propagates in the straight duct. This will not be the case for high frequencies, and especially not if there are discontinuities in the straight parts of the horn preceding the bend, where mode coupling takes place.

The full equations for modal coupling are given by Osborne [33]. We will use his assumption that $\phi = 0$ at one end of the duct, and $\phi = \Phi$ at the other end.

At the entrance of the duct, we have for the continuity in pressure:

$$\begin{aligned}&\sum_{n_x=0}^{\infty}\sum_{n_z=0}^{\infty}\int_0^a\int_0^b A^-_{n_xn_z}\psi_n(x,z)dxdz + \sum_{n_x=0}^{\infty}\sum_{n_z=0}^{\infty}\int_0^a\int_0^b A^+_{n_xn_z}\psi_n(x,z)dxdz \\ &= \sum_{n_z=0}^{\infty}\sum_{s=0}^{\infty}\int_0^a\int_{R_1}^{R_2} A^-_{n_zs}\Psi_{n_zs}(r,z)dzdr + \sum_{n_z=0}^{\infty}\sum_{s=0}^{\infty}\int_0^a\int_{R_1}^{R_2} A^+_{n_zs}\Psi_{n_zs}(r,z)dzdr\end{aligned} \tag{52}$$

And for the continuity in velocity:

$$\begin{aligned}&\sum_{n_x=0}^{\infty}\sum_{n_z=0}^{\infty}\int_0^a\int_0^b \frac{k_{n_xn_z}}{k\rho c}A^-_{n_xn_z}\psi_n(x,z)dxdz + \sum_{n_x=0}^{\infty}\sum_{n_z=0}^{\infty}\int_0^a\int_0^b \frac{k_{n_xn_z}}{k\rho c}A^+_{n_xn_z}\psi_n(x,z)dxdz \\ &= \sum_{n_z=0}^{\infty}\sum_{s=0}^{\infty}\int_0^a\int_{R_1}^{R_2} \frac{k_\varphi^{n_zs}}{k\rho cr}A^-_{n_zs}\Psi_{n_zs}(r,z)dzdr + \sum_{n_z=0}^{\infty}\sum_{s=0}^{\infty}\int_0^a\int_{R_1}^{R_2} \frac{k_\varphi^{n_zs}}{k\rho cr}A^+_{n_zs}\Psi_{n_zs}(r,z)dzdr\end{aligned} \tag{53}$$

At the exit of the duct, we have for the continuity in pressure:

$$\sum_{n_z=0}^{\infty}\sum_{s=0}^{\infty}\int_0^a\int_{R_1}^{R_2} A^-_{n_zs}\Psi_{n_zs}(r,z)e^{-ik_\varphi^{n_zs}\Phi}dzdr + \sum_{n_z=0}^{\infty}\sum_{s=0}^{\infty}\int_0^a\int_{R_1}^{R_2} A^+_{n_zs}\Psi_{n_zs}(r,z)e^{ik_\varphi^{n_zs}\Phi}dzdr = \sum_{n_x=0}^{\infty}\sum_{n_z=0}^{\infty}\int_0^a\int_0^b B^-_{n_xn_z}\psi_n(x,z)dxdz + \sum_{n_x=0}^{\infty}\sum_{n_z=0}^{\infty}\int_0^a\int_0^b B^+_{n_xn_z}\psi_n(x,z)dxdz \quad (54)$$

And for the continuity in velocity:

$$\sum_{n_z=0}^{\infty}\sum_{s=0}^{\infty}\int_0^a\int_{R_1}^{R_2} \frac{k_\varphi^{n_zs}}{k\rho cr}A^-_{n_zs}\Psi_{n_zs}(r,z)e^{-ik_\varphi^{n_zs}\Phi}dzdr + \sum_{n_z=0}^{\infty}\sum_{s=0}^{\infty}\int_0^a\int_{R_1}^{R_2} \frac{k_\varphi^{n_zs}}{k\rho cr}A^+_{n_zs}\Psi_{n_zs}(r,z)e^{ik_\varphi^{n_zs}\Phi}dzdr = \sum_{n_x=0}^{\infty}\sum_{n_z=0}^{\infty}\int_0^a\int_0^b \frac{k_{n_xn_z}}{k\rho c}B^-_{n_xn_z}\psi_n(x,z)dxdz + \sum_{n_x=0}^{\infty}\sum_{n_z=0}^{\infty}\int_0^a\int_0^b \frac{k_{n_xn_z}}{k\rho c}B^+_{n_xn_z}\psi_n(x,z)dxdz \quad (55)$$

These equations have to be solved for the modal amplitudes, and it may be possible to construct an F-matrix similar to that used in straight rectangular ducts. But it seems from the above equations, that this matrix will not be frequency independent, since the order of the Bessel functions is a function of the angular wavenumber. And this wavenumber is given by the dispersion relation in Eq. (42). This is a complication not found in straight ducts, and it makes it extremely difficult to find the F-matrix for connecting duct bends. Osborne [33] makes some simplified calculations using these equations, but considers only propagating modes.

2.4.4 *Discontinuities in curved ducts*

The discontinuity between two curved segments of different width and/or radius of curvature, may be handled in the same way as in sections 2.2.1 and 2.3.1. We may express the eigenfunctions as a product of eigenfunctions in the r and z directions, as in Eq. (43). $Z_{n_z}(z)$

from this equation corresponds to σ_{n_z} from Eq. (2). It should then be possible to develop an F-matrix for the discontinuities between two curved ducts in the same manner as for two straight ducts.

$$\begin{aligned} F_{nm} &= \frac{1}{S_1}\int_{S_1}\Psi^{(1)}_{n_z s}\Psi^{(2)}_{m_z t}dS \\ &= \frac{1}{a}\int_0^a \sigma^{(1)}_{n_z}\sigma^{(2)}_{m_z}dx\frac{1}{R_2-R_1}\int_{R_1}^{R_2} R^{(1)}_s R^{(2)}_t dr \\ &= Z_{n_z m_z}R_{st} \end{aligned} \tag{56}$$

So the R_{st}-matrix for this discontinuity will become

$$R_{st} = \frac{1}{R_2-R_1}\int_{R_1}^{R_2} R^{(1)}_s R^{(2)}_t dr \tag{57}$$

where

$$R_s(r) = J_{k_\varphi^{n_z s}}(k_r r) + -\frac{J'_{k_\varphi^{n_z s}}(k_r R_1)}{Y'_{k_\varphi^{n_z s}}(k_r R_1)}Y_{k_\varphi^{n_z s}}(k_r r) \tag{58}$$

Again comes the complication that the resulting matrix is not frequency independent. The integration of the mode functions in Eq. (57) is also very complicated. Inserting it into the mathematical software Maple [58] and performing the integration, produces an equation extending over about 200 lines. It may be possible to simplify the results by the proper orthogonality relationships, but considering the complications mentioned in the previous section, this was not considered worth the effort at this point.

2.4.5 *An alternative method*

The complexities of the modal description of duct bends have inspired researchers to find alternative methods. A method is described by Felix and Pagneux [27] for two-dimensional curved ducts of both constant and varying width. This method is later extended to toroidal bends [28] and also lined bends [45]. The method will be briefly outlined here. For more details, please consult the references.

The pressure and axial velocity through the (two-dimensional) bend are expressed as infinite series:

$$p(r,\phi) = \sum_n \psi_n(r)P_n(\phi) \tag{59}$$

$$v_\phi(r,\phi) = \sum_n \psi_n(r)U_n(\phi) \tag{60}$$

The functions ψ_n are

$$\psi_n(r) = A_n\cos\left(\frac{n\pi}{h}(r-R_1)\right) \tag{61}$$

where

$$A_n = \sqrt{\frac{2 - \delta_{n0}}{b}} \tag{62}$$

These functions are not the radial eigenfunctions of a curved duct, but satisfy the Neumann boundary condition at the walls. They are orthogonal, and can be used as a modal representation of the sound field in the bend. An advantage of this approach is that there is no need to couple two different modal descriptions where the bend connects to the straight duct.

Felix and Pagneux use a normalization of pressure and velocity to the reference pressure $\rho_0 c_0^2$ and reference velocity c_0. They further derive expressions for the differential pressure and velocity:

$$U' = \frac{1}{jk}\left(C + KB\right)P \tag{63}$$

$$P' = -jkBU \tag{64}$$

where the matrices K, B, and C are given as

$$K_{mn} = \left(k^2 - \left(\frac{n\pi}{b}\right)^2\right)\delta_{mn} \tag{65}$$

$$B_{mn} = \int_{R_1}^{R_2} r\psi_n(r)\psi_m(r)dr \tag{66}$$

which to a first order approximation becomes

$$B_{mn} = \begin{cases} R_1 + \frac{b}{2} & : \quad n = m, \\ A_m A_n \left(\frac{b}{\pi}\right)^2 ((-1)^{m+n} - 1)\frac{m^2+n^2}{(m^2-n^2)^2} & : \quad n \neq m \end{cases} \tag{67}$$

and

$$C_{mn} = \int_{R_1}^{R_2} \psi_n(r)\psi_m'(r)dr \tag{68}$$

which to a first order approximation becomes

$$C_{mn} = \begin{cases} 0 & : \quad n = m, \\ A_m A_n((-1)^{m+n} - 1)\frac{m^2}{m^2-n^2} & : \quad n \neq m \end{cases} \tag{69}$$

Eq. (63) and Eq. (64) cannot be integrated directly because of the evanescent modes, which would cause numerical instability. Therefore, an impedance matrix is formed, so that

$$P = ZU \tag{70}$$

By using $P' = Z'U + ZU'$, a Riccati[4] (nonlinear) differential equation is found for the normalized[5] impedance matrix:

$$Z' = -jkB - \frac{1}{jk}Z(C + KB)Z \tag{71}$$

By integrating this differential equation, the modal input impedance to the bend can be found from the modal impedance at the outlet.

Felix and Pagneux also describe an extension of this model to two-dimensional ducts of varying cross section, where R_1 and R_2 are functions of angle. In this case,

$$\psi_n(r,\theta) = \sqrt{\frac{2-\delta_{n0}}{R_2(\phi) - R_1(\phi)}} \cos\left(n\pi \frac{r - R_1(\phi)}{R_2(\phi) - R_1(\phi)}\right) \tag{72}$$

The Riccati impedance equation becomes

$$Z' = -jkB - \frac{1}{jk}Z(C + KB)Z + ZD - DZ + EZ \tag{73}$$

where the extra matrices are

$$C_{mn} = \begin{cases} \frac{R_2' - R_1'}{R_2 - R_1}\left(1 - \frac{\delta_{m0}}{2}\right) & : \quad n = m, \\ A_m A_n((-1)^{m+n}R_2' - R_1')\frac{m^2}{m^2 - n^2} & : \quad n \neq m \end{cases} \tag{74}$$

$$E_{mn} = A_m A_n((-1)^{m+n}R_2' - R_1') \tag{75}$$

and are functions of the variation of cross section.

2.5 MODAL RADIATION IMPEDANCE

As for circular horns, a modal radiation impedance is needed. Again, this is found by integrating over the opening of the duct, and we have for the symmetrical case [1]:

$$Z_{nm} = \frac{j\omega\rho}{2\pi S^2} \int_{-a}^{a} dx \int_{-b}^{b} dy \int_{-a}^{a} dx_0 \int_{-b}^{b} dy_0 \psi_m(x_0, y_0)\psi_n(x,y)\frac{e^{-jkh}}{h} \tag{76}$$

where ψ_n is given in Eq. (2) and following equations, and

$$h = \sqrt{(x - x_0)^2 + (y - y_0)^2} \tag{77}$$

Kemp [1] shows how this quadruple integral can be reduced to a sum of a double integral, a single integral, and an analytical function.

4 The Riccati equation is a nonlinear differential equation of the general form $Y' = A(x)Y^2 + B(x)Y + C(x)$.

5 This equation is normalized by the characteristic impedance z_c, which in the case of a three-dimensional duct would be $\rho_0 c_0 / S$. The non-normalized version of Eq. (71) would be $Z' = -jkz_c B - (jkz_c)^{-1} Z(C + KB)Z$ [29].

The procedure is given in the reference, and only the result is given here:

$$
\begin{aligned}
Z_{nm} = & \\
& \frac{j\rho c}{2\pi S} \int_0^{2ka} du \int_0^{2kb} dv \frac{\left(1-\frac{u}{2ka}\right)\left(1-\frac{v}{2kb}\right)}{\sqrt{u^2+v^2}} \times \\
& \left[e^{-\sqrt{u^2+v^2}} \frac{G(n_x, m_x, \frac{u}{2ka}, \frac{1}{2})}{1-\frac{u}{2ka}} \frac{G(n_y, m_y, \frac{v}{2kb}, \frac{1}{2})}{1-\frac{v}{2kb}} - f(n,m) \right] \\
& + \frac{j\rho c}{2\pi S} \int_0^{2ka} du \left(1-\frac{u}{2ka}\right) \times \\
& \left[\ln\left(2kb + \sqrt{u^2+(2kb)^2}\right) + \frac{u}{2kb} - \frac{1}{2kb}\sqrt{u^2+(2kb)^2} \right] f(n,m) \\
& + \frac{j\rho c}{2\pi S} \left[-ka\ln(2ka) + \frac{3}{2}ka \right] f(n,m)
\end{aligned} \tag{78}
$$

where

$$
\begin{aligned}
G(n_x, m_x, \xi, a) = & N_{n_x} N_{m_x} \times \\
& \left[\operatorname{sinc}\left((n_x+m_x)\,\pi \left(1-\frac{\xi}{2a}\right)\right) \cos\left(\frac{(n_x-m_x)\,\pi\xi}{2a}\right) + \right. \\
& \left. \operatorname{sinc}\left((n_x-m_x)\,\pi \left(1-\frac{\xi}{2a}\right)\right) \cos\left(\frac{(n_x+m_x)\,\pi\xi}{2a}\right) \right]
\end{aligned} \tag{79}
$$

with

$$
N_{n_x} = \begin{cases} 1 & : n_x = 0 \\ \sqrt{2} & : n_x > 0 \end{cases} \tag{80}
$$

and

$$
\begin{aligned}
f(n,m) = & N_{n_x} N_{m_x} N_{n_y} N_{m_y} \times \\
& \left[\operatorname{sinc}\left(n_x+m_x\right)\pi + \operatorname{sinc}\left(n_x-m_x\right)\pi \right] \times \\
& \left[\operatorname{sinc}\left(n_y+m_y\right)\pi + \operatorname{sinc}\left(n_y-m_y\right)\pi \right]
\end{aligned} \tag{81}
$$

Similarly, the expressions for asymmetric ducts, or rather, ducts where asymmetric modes are included, are given by Kemp [59] as:

$$
\begin{aligned}
Z_{mn} = & \\
& \frac{j\rho c}{2\pi S} \int_0^{ka} du \int_0^{kb} dv \frac{\left(1-\frac{u}{2ka}\right)\left(1-\frac{v}{2kb}\right)}{\sqrt{u^2+v^2}} \times \\
& \left[e^{-\sqrt{u^2+v^2}} \frac{G(n_x, m_x, \frac{u}{ka}, 1)}{1-\frac{u}{2ka}} \frac{G(n_y, m_y, \frac{v}{kb}, 1)}{1-\frac{v}{2kb}} - f(n,m) \right] \\
& + \frac{j\rho c}{2\pi S} \int_0^{ka} du \left(1-\frac{u}{2ka}\right) \times \\
& \left[\ln\left(kb + \sqrt{u^2+(kb)^2} \right) + \frac{u}{2kb} - \frac{1}{2kb}\sqrt{u^2+(kb)^2} \right] f(n,m) \\
& + \frac{j\rho c}{2\pi S} \left[-\frac{3}{4} ka \ln(ka) + \frac{7}{8} ka \right] f(n,m)
\end{aligned} \tag{82}
$$

where $f(n,m)$ is given as before, but

$$
\begin{aligned}
G(n_x, m_x, \xi, a) = & N_{n_x} N_{m_x} \times \\
& \left\{ \cos\left((n_x - m_x)\, \pi \frac{\xi}{2a} \right) \times \right. \\
& \frac{1}{2} \left[(2a-\xi)\, \mathrm{sinc}\left((n_x + m_x)\, \pi \left(1 - \frac{\xi}{2a} \right) \right) \right. \\
& \left. -\xi \mathrm{sinc}\left((n_x + m_x)\, \pi \frac{\xi}{2a} \right) \right] \\
& + \cos\left((n_x + m_x)\, \pi \frac{\xi}{2a} \right) \\
& \frac{1}{2} \left[(2a-\xi)\, \mathrm{sinc}\left((n_x - m_x)\, \pi \left(1 - \frac{\xi}{2a} \right) \right) \right. \\
& \left. \left. -\xi \mathrm{sinc}\left((n_x - m_x)\, \pi \frac{\xi}{2a} \right) \right] \right\}
\end{aligned} \tag{83}
$$

When $n_x = m_x = 0$, then

$$
G\left(n_x, m_x, \xi, 1\right) = 2\left(1-\xi\right) \tag{84}
$$

These expressions can be integrated numerically to give the values for the computations. Details about the implementation can be found in the next chapter.

2.6 PROJECTION OF IMPEDANCE AND VOLUME VELOCITY

The pressure and volume velocity are related through the modal impedance. We know the impedance at the end of the duct. To calculate the pressure field inside the duct, we first need to know the impedances throughout the duct. Then we can apply, for instance, a

plane wave of constant velocity at one end, and propagate this velocity through the duct using the equations given in sections 2.1 and 2.3. The pressure is then found by multiplying the velocity at the point in question by the impedance at that point.

Kemp gives the equations for propagating the impedance from duct 2 to duct 1 as

$$Z^{(1)} = FZ^{(2)}F^T,\ S_1 < S_2 \tag{85}$$

$$Z^{(1)} = V^{-1}Z^{(2)}(V^T)^{-1},\ S_1 > S_2 \tag{86}$$

For propagation along a uniform duct, where $Z^{(0)}$ is the impedance at the input of the duct, due to an impedance $Z^{(1)}$ at the output, the relation is

$$Z^{(0)} = \left(D_1 Z^{(1)} + D_2 Z_c\right)\left(D_2 Z_c^{-1} Z^{(1)} + D_1\right)^{-1} \tag{87}$$

If we simplify this by multiplying top and bottom by D^{-1}, we get

$$Z^{(0)} = \left(Z^{(1)} + jD_3 Z_c\right)\left(jD_3 Z_c^{-1} Z^{(1)} + I\right)^{-1} \tag{88}$$

But this form assumes that the impedance matrices are diagonal, so it is better to use [1, corrected version]

$$Z^{(0)} = (jD_3)^{-1} Z_c - D_2^{-1} Z_c \left(Z^{(1)} + (jD_3)^{-1} Z_c\right)^{-1} D_2^{-1} Z_c \tag{89}$$

The extra matrices are defined as follows:

$$D_1(n,m) = \begin{cases} \cos(k_n d) & :\ n = m \\ 0 & :\ n \neq m \end{cases} \tag{90}$$

$$D_2(n,m) = \begin{cases} j\sin(k_n d) & :\ n = m \\ 0 & :\ n \neq m \end{cases} \tag{91}$$

$$D_3(n,m) = \begin{cases} \tan(k_n d) & :\ n = m \\ 0 & :\ n \neq m \end{cases} \tag{92}$$

$$Z_c(n,m) = \begin{cases} k\rho c / k_n S & :\ n = m \\ 0 & :\ n \neq m \end{cases} \tag{93}$$

where d is the length of the duct as in Figure 3, and S is the cross-sectional area.

Volume velocity is propagated along the duct as

$$\vec{U}^{(1)} = \left(-D_2 Z_c^{-1}\left(Z^{(0)} - Z_c\right) + E\right)\vec{U}^{(0)} \tag{94}$$

where the extra matrix is:

$$E(n,m) = \begin{cases} e^{-jk_n d} & : \ n = m \\ 0 & : \ n \neq m \end{cases} \tag{95}$$

For discontinuities, the velocity is propagated according to equations (22) and (23).

2.6.1 *Expansion in one plane, contraction in the other*

Some horns can expand in one direction and contract in the other. This leads to complications in deriving the F matrix, but can be solved in a much simpler way if one is prepared to compute two matrices instead of one. Since the F matrices are computed only once for each horn profile, the extra computational cost is small. The idea is to split the discontinuity into two successive discontinuities, the first expanding in only one direction, with no change in the other, and a second only contracting in the other direction, with no change in the first. See Figure 7. For instance, $\beta_x < 1$, $\beta_y = 1$ for the first, and $\beta_x = 1$, $\beta_y > 1$ for the second.

For velocity propagation across these two discontinuities, which we will call a and b, we have that

$$\vec{U}^{(2)} = F_a^T \vec{U}^{(1)} \tag{96}$$

$$\vec{U}^{(3)} = F_b^T \vec{U}^{(2)} \tag{97}$$

so that

$$\vec{U}^{(3)} = F_b^T F_a^T \vec{U}^{(1)} \tag{98}$$

Since $(AB)^T = B^T A^T$, we can form a composite matrix $F_C = F_a F_b$ that replaces the two F matrices. Actually, one of the matrices will be a V matrix, since there is a contraction in one plane. If this is the second matrix, $F_C = F_a V_b^{-1}$.

That this also works for impedances, can be seen from the following:

$$Z^{(1)} = F_a Z^{(2)} F_a^T \tag{99}$$

$$Z^{(2)} = F_b Z^{(3)} F_b^T \tag{100}$$

$$Z^{(1)} = F_a F_b Z^{(3)} F_b^T F_a^T \tag{101}$$

where we again can form the composite matric $F_C = F_a F_b$.

The case of a skewed duct where, for instance, $a_{1+} < a_{2+}$ while at the same time $a_{1-} < a_{2-}$, cannot be solved using this technique, and will need special treatment. In practice, however, it is very rare for this to happen in straight horns, and in curved horns the problem could be solved by using a series of bend elements.

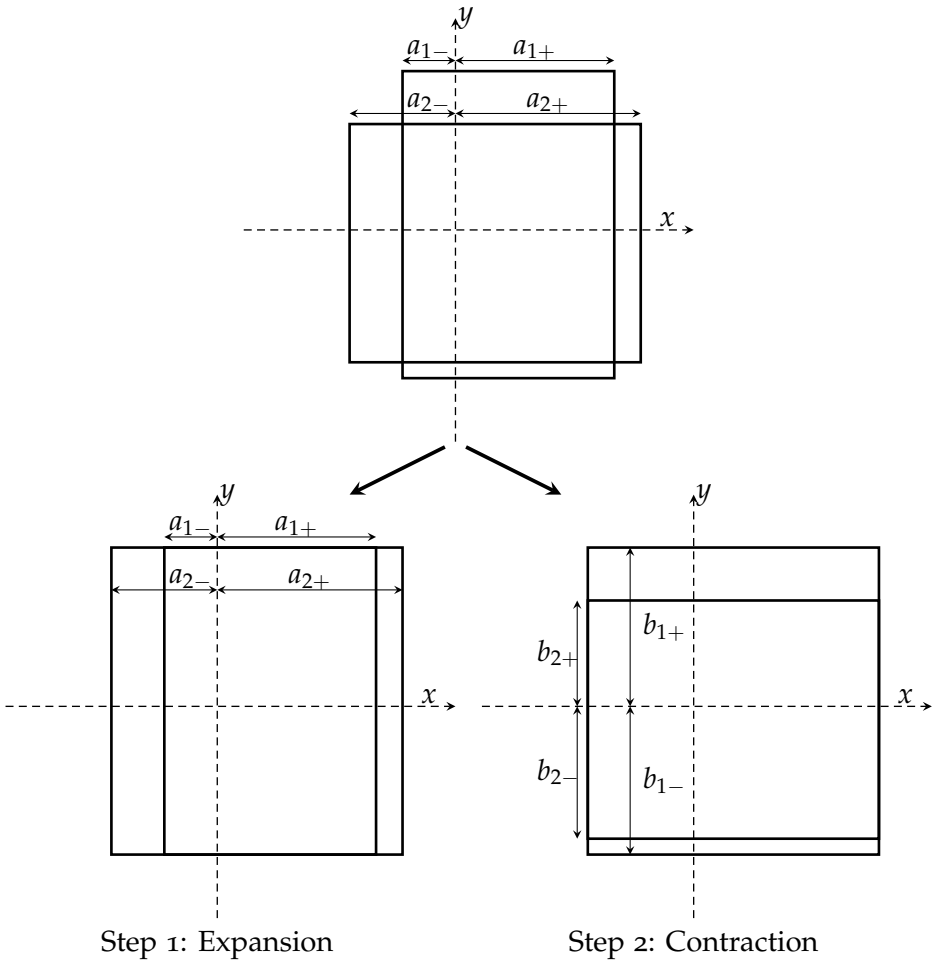


Figure 7: Partitioning of a discontinuity

2.7 CALCULATION OF THE PRESSURE RESPONSE

When the horn is mounted in an infinite baffle, the radiated pressure can be found from the Rayleigh integral [60, 61], as in the case of circular horns. The Rayleigh integral maps the velocity profile in the mouth of the horn to far field pressure response, and is given as

$$p(r) = \frac{jk\rho c}{2\pi} \int_S u(S) \frac{e^{-jkr}}{r} dS \tag{102}$$

where $u(S)$ is the velocity in the mouth plane. This is found from the modal volume velocity amplitudes, the mode functions, and the mouth area.

Another method, given by Geddes [51], is to use the modal amplitudes directly. If the velocity profile is given as

$$F(x,y) = A_n \cos\left(n\pi\frac{x}{a}\right) \cdot B_m \cos\left(m\pi\frac{y}{b}\right) \tag{103}$$

the pressure can be found as

$$\begin{aligned} p(r,\theta_x,\theta_y) &= j\rho ck\frac{e^{jkr}}{2\pi r} \sum_{n,m} A_n B_m \int_{-b}^{b}\int_{-a}^{a} \cos\left(n\pi\frac{x}{a}\right)\cos\left(m\pi\frac{y}{b}\right) e^{jk_x x} e^{jk_y y} dxdy \\ &= j\rho ckS\frac{e^{jkr}}{2\pi r} \sum_{n,m} A_n B_m G_n(ka\sin\theta_x) \cdot G_m(kb\sin\theta_y) \end{aligned} \tag{104}$$

where

$$G_n(x) = (-1)^n \frac{x\sin(x)}{x^2 - (n\pi)^2}$$

A_n and B_m can be found from the velocity profile as

$$A_n = \int_{-a}^{a} u(x)\cos\left(\frac{n\pi x}{a}\right) dx \tag{105}$$

$$B_m = \int_{-b}^{b} u(y)\cos\left(\frac{m\pi y}{a}\right) dy \tag{106}$$

But when the modal amplitudes already are known from the modal propagation through the horn, we have

$$A_n B_m = a_n b_m u(n,m) \tag{107}$$

where

$$a_n = \begin{cases} 1 & n = 0 \\ \sqrt{2} & n > 0 \end{cases} \tag{108}$$

$$b_m = \begin{cases} 1 & m = 0 \\ \sqrt{2} & m > 0 \end{cases} \tag{109}$$

and $u(n, m)$ are the modal velocity amplitudes.

The mouth velocity profile given in Eq. (103) corresponds to the mode functions for a symmetric duct. For an asymmetric duct, the integration limits have to be changed.

3 IMPLEMENTATION

The MPM for rectangular horns is implemented in Matlab [62] in much the same way as for axisymmetric horns, described in [11, 12], and is basically a straightforward implementation of the equations in the previous chapter.

3.1 MODAL RADIATION IMPEDANCE

The calculation of the modal radiation impedance requires the calculation of an oscillatory double integral that becomes more and more expensive to evaluate as the mode number increases, and as frequency increases beyond $ka = 1$. Another challenge with both the radiation impedance and the rest of the MPM calculations, is that the total number of modes increases as 4th power of the number of modes in each direction. Thus 8 modes in each direction results in a 64×64 matrix, and 4096 modal radiation impedance values to compute for each frequency. Thankfully, the matrix is symmetric, but that still leaves 2080 values to be computed. With 16 modes in each direction, 32896 values must be computed.

Due to these issues, it was decided to precompute the radiation impedance for a set of modes and a set of frequencies, for a square duct. For the ordinary computations, the impedance values will be interpolated between the frequencies used in the calculations. All horns investigated in this thesis will therefore have a square mouth.

Several methods were tried to increase the speed of the computations. The Matlab Parallel Computing toolbox was used, and Andreas Asheim provided a fast numerical integration algorithm based on the Method of Steepest Descent (MSD) [63], giving high speed and high accuracy for high ka values. Since the accuracy increases with ka, other methods have to be used at low frequencies. The integration is performed with the Matlab function `dblquad`, where the `quadgk` function is supplied as the integration algorithm. As frequency and mode number rise, the integrand becomes increasingly oscillatory, and the computation time lengthens. When the computation time starts increasing rapidly, results are compared to the results of the MSD algorithm, and if the result is sufficiently accurate, this method is used for the subsequent computations. The algorithms used are listed in the appendix.

Figure 8 shows how the two methods compare. The case is a symmetrical rectangular duct with $a = 1$, $b = 2$, $n = (1, 2)$ and $m = (2, 1)$. The transfer to the MSD algorithm happens at $k = 14.2$, showing a

markedly reduced, and constant, computation time compared to the standard method.

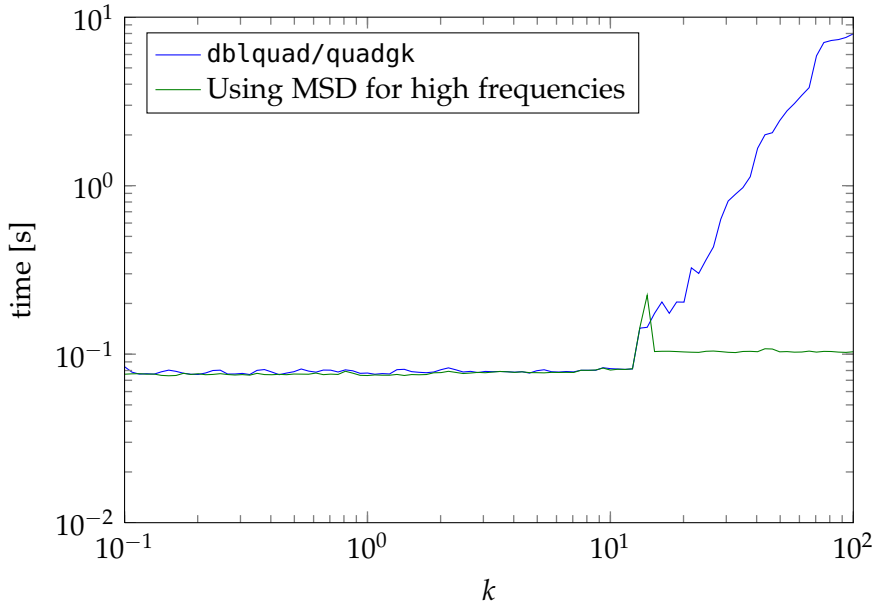


Figure 8: Calculation times for the two methods used computing the radiation impedance

Even with the MSD, the computation of 16×16 modes for the symmetrical case took 7 days 11 hours 17 minutes on a 8-core computer. 16×16 modes for the asymmetric case took around 36 hours, since the actual highest mode number is only half of that for the symmetric case (16×16 asymmetric modes corresponds to 8×8 symmetric modes, since half of the symmetric modes are zero).

A more efficient way to calculate the radiation impedance, for instance by doing the calculations in the time domain, would be required if the MPM is to be feasible for the general rectangular horn, since the radiation impedance for a large number of different aspect ratios would be needed.

A couple of alternative methods were tried, before settling on the MSD algorithm. One method consisted of transforming Eq. (78) to polar coordinates. The other method used a Fourier transform of the near-field pressure distribution for each velocity mode. These methods gave either no gain in computation time, or had excessive memory requirements and scaling problems. They are considered outside the scope of this work, and will not be described further.

A few examples of modal radiation impedances are given in Figure 9 and Figure 10, for a square baffle with symmetrical modes. Figure 9 shows the "inter-modal" coupling, i.e. the coupling between velocity modes and pressure modes with the same mode numbers. The higher the mode numbers, the higher the peak, and the higher

in frequency the peak occurs. Also, the curve is smoother, with the ripple magnitude decreasing for the higher modes.

Figure 10 shows a few examples of cross-modal impedances, that couple one velocity mode to a different pressure mode. These impedances are significant only over a limited frequency range, and are smaller in magnitude than the inter-modal impedances. As the difference between the velocity pattern and the pressure pattern increases, i.e. the mode numbers differ more than in the examples given, the coupling is even weaker. Typically the peaks are at least three orders of magnitude lower than the asymptotic impedance for inter-modal coupling.

Since the largest number of entries in the radiation impedance matrix is made up from cross-modal impedances, these constitute a large part of the computational cost. When these, in addition, are very small in magnitude except over a small frequency range, it would be logical to find ways to avoid computing these impedances, or to make approximations. This has, however, been considered beyond the scope of this work.

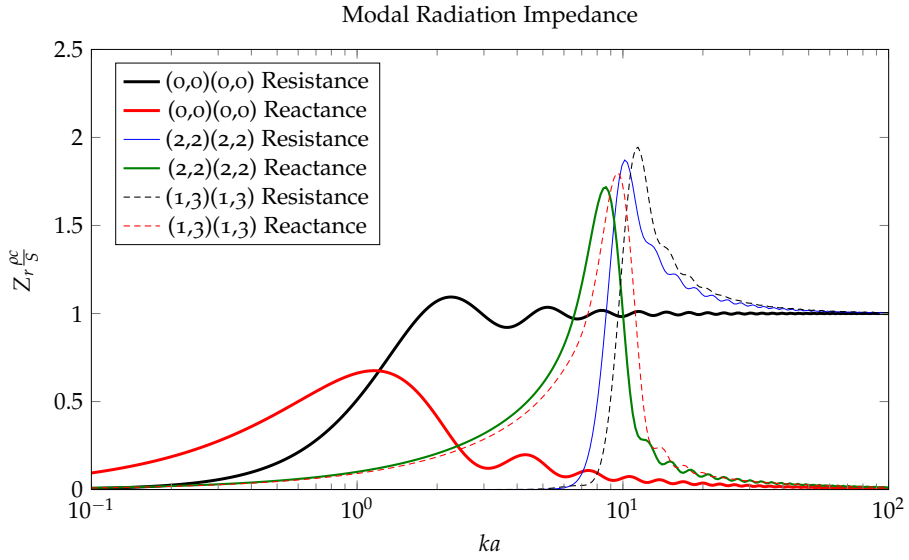


Figure 9: Modal radiation impedance, velocity modes and pressure modes have the same mode numbers

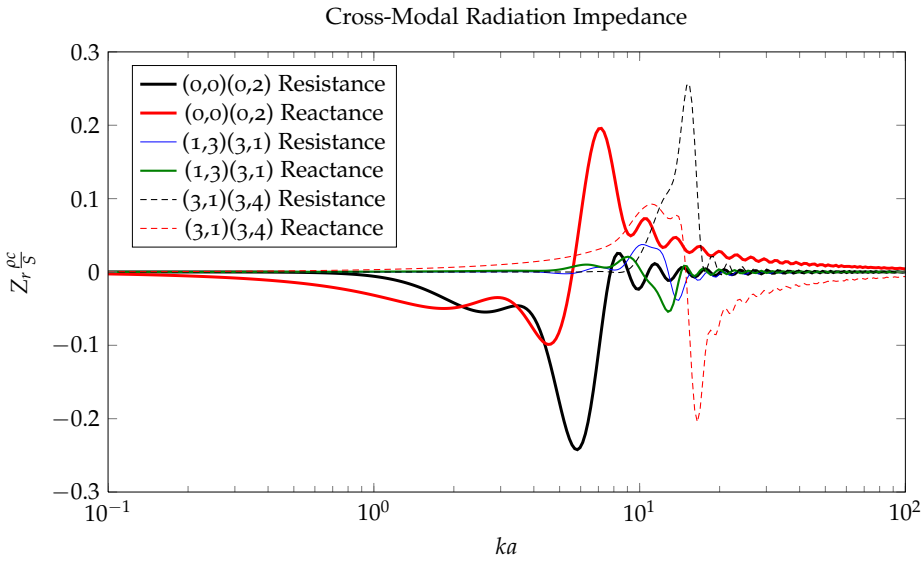


Figure 10: Cross-modal impedances. Shows the coupling between modes where the mode numbers of the pressure and velocity modes differ

3.2 BEND MODES

As presented in section 2.4.1, the angular wavenumber is found from solving for k_φ in the dispersion relation:

$$g' = J'_{k_\varphi}(k_r R_1) Y'_{k_\varphi}(k_r R_2) - J'_{k_\varphi}(k_r R_2) Y'_{k_\varphi}(k_r R_1) = 0 \tag{110}$$

The derivatives of Bessel functions are given as

$$Z'_\nu(z) = -Z_{\nu+1}(z) + \frac{\nu}{z} Z_\nu(z) \tag{111}$$

where Z_ν is a general Bessel function of order ν.

The dispersion relation is plotted in Figure 11 as a function of real order $\nu = k_\varphi$ for real arguments k_r, R_1 and R_2. The ordinate range is limited to [-1,1] in order to show the zero-crossings more clearly. The value of Eq. (110) increases exponentially, and for the case shown, reaches $2 \cdot 10^{19}$ for $\nu = 50$. As can be seen, there is a finite number of real zeros.

An algorithm for finding the roots of Eq. (110) was described by Osborne [31]. The idea was to compute Eq. (110) for a number of trial values, and then search for the roots. The algorithm outlined was tried, but did not produce successful results.

Another algorithm by Sorolla et al. [46] was also tried, but this algorithm did not produce successful results either.

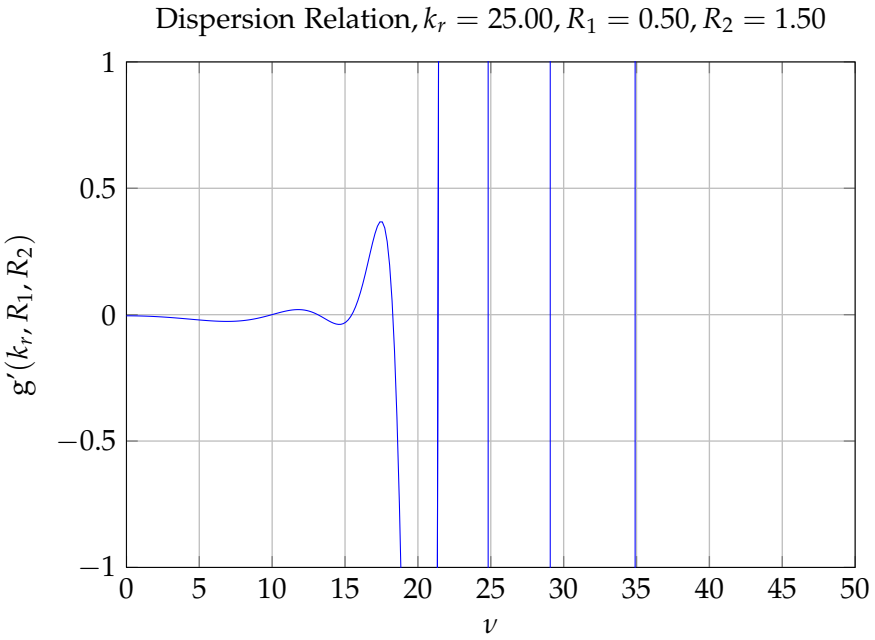


Figure 11: The dispersion relation as function of ν

k_φ
9.9825
13.1324
15.4665
18.2794
21.3817
24.8541
28.9510
34.8383

Table 2: The $\nu-$zeros for the case plotted in Figure 11

Instead, an algorithm was written that first brackets the roots by the method given in [64], followed by the Matlab function `fzero` for each of the subranges containing a root. By supplying enough bracketing intervals, this method succeeded in finding all the real roots.

An approximation for high and imaginary wavenumbers is given by Cochran [54], as presented in section 2.4.2. However, this approximation is only valid for fairly high wavenumbers, and cannot be used universally.

Figure 12 shows how Cochran's approximation approaches the true value of Eq. (110) for high values of ν. The functions are plotted as $y = \operatorname{sgn}(g') \cdot \log_{10}(|\Im(g')|)$, since the magnitude of g' increases exponentially. The approximation has the same general shape as the true function, and becomes better for lower values of k_φ.

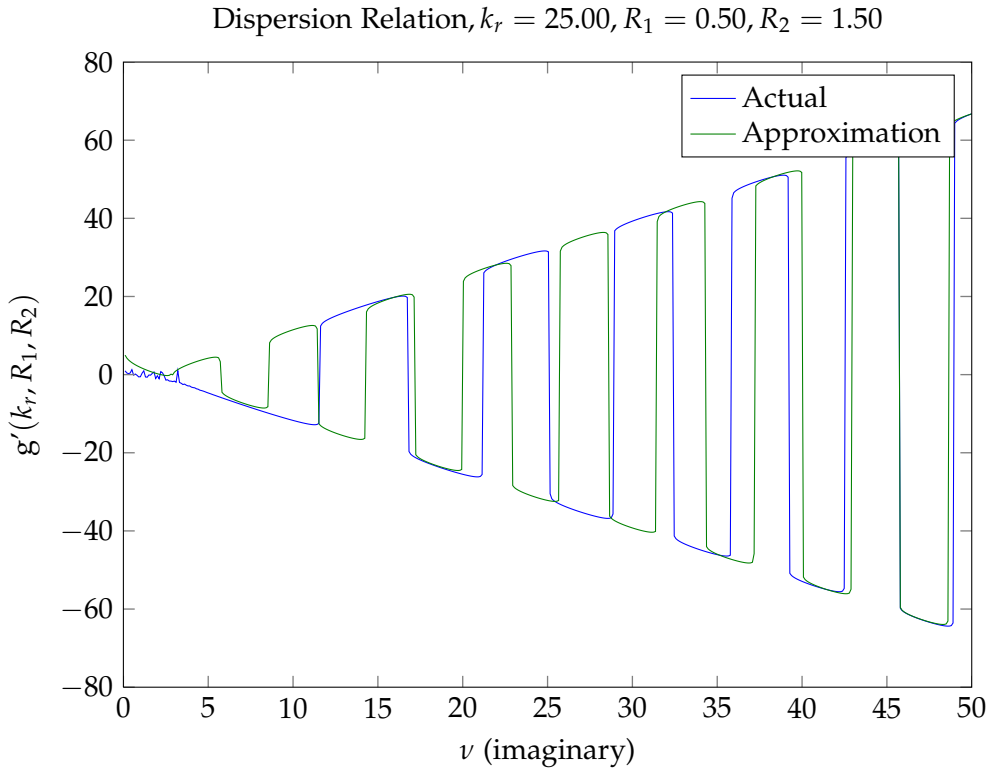


Figure 12: Comparison of the true value of Eq. (110) and Cochran's approximation for imaginary ν

There seems to be some numerical noise for $\nu < 5$, perhaps due to the way Eq. (110) is implemented for imaginary ν.

To find the general imaginary roots, algorithms for calculating Bessel functions of imaginary order are necessary. However, these are not generally available. Matlab can give Bessel functions of real order with complex arguments using an algorithm by Amos [65], and corresponding algorithms are given by Zhang [66] and Press et al. [64]. Morgan [67] tabulates Bessel functions of imaginary order, computed by direct numerical integration of the differential equation, but tables are not very practical for the problem at hand.

Various relations and approximations are given by Dunster [68]. Matyshev and Fohtung [69] present methods for calculating the functions, but these were not implemented.

The Bessel function $J_\nu(z)$ is given by the series expansion

$$J_\nu(z) = \sum_{m=0}^{\infty} (-1)^m \frac{\left(\frac{z}{2}\right)^{2m+\nu}}{m!\Gamma(m+\nu+1)} \tag{112}$$

which is valid for all ν and z. Since none of the other algorithms found could be implemented satisfactorily, this equation was directly implemented. It was necessary to use a Gamma function from the

Matlab Central [70], since the Gamma function in Matlab does not take complex arguments. The sum was continued until the factor $\left(\frac{z}{2}\right)^{2m+\nu} / m!\Gamma(m+\nu+1)$ was less than 10^{-12}. The factorial in the denominator was calculated successively by keeping the previous $1/m!$ term, and dividing by m.

The results were compared to Matlab's built-in Bessel functions for real ν, and to the values tabulated by Morgan, and the numerical evaluations of Maple for imaginary and complex ν, and were found to be sufficiently accurate.

3.3 HORN CONTOURS AND MESHING

To compare MPM and BERIM, the same horn contour must be used in both cases. It is necessary to generate identical horn contours for both methods, in a simple an efficient way. To study rectangular horns in BERIM, a full 3D triangular mesh of the horn is needed. This presents a rather bigger challenge than the mesh for an axisymmetric horn, which consists of only straight lines.

Most of the framework for generating contours for rectangular horns had already been implemented in a software project named HornCAD that the author has been working on for many years. This Windows program can generate and simulate a wide variety of horn contours, in addition to other loudspeaker configurations. The dimensions of the horns can be exported as a text file, and this text file subsequently imported into Matlab for use with the MPM code.

Generating the 3D triangular mesh for the horn required some more work. While there are several free and commercial software packages for this task, like GiD [71], they can be hard to integrate with the rest of the system, and may have expensive licences if meshes of usable sizes are to be generated. To achieve a tight integration with the software used in this project, custom code was used. The author has not been able to find open code for 3D surface meshing, but there are several algorithms available for 2D meshing. One package able to generate very high quality 2D meshes is Triangle [72], which the author has used on several occasions. It is also available as a DLL file [73], which was incorporated into HornCAD and used to generate 3D surface meshes. An example of such a mesh is shown in Figure 13. A structured mesh is generated for the horn walls, and here Triangle is only used to connect the vertices into triangles. For the horn mouth and throat, irregular meshes are generated, using the Delaunay capability of Triangle. A mesh size giving at least 6 elements per wavelength is used.

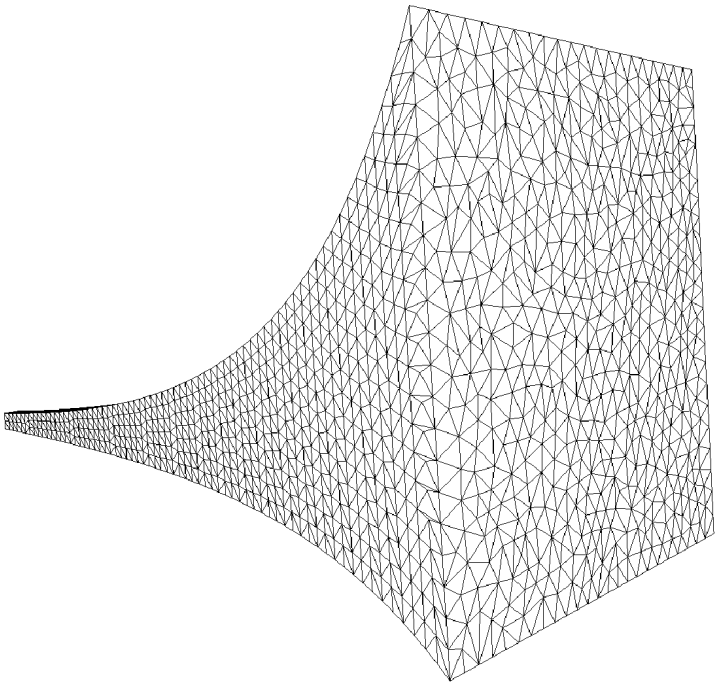


Figure 13: Example of surface mesh generated by HornCAD

3.4 BERIM

Even for low resolution meshes (2kHz mesh bandwidth), the number of elements can be quite large. The horn mesh shown in Figure 13 has 3193 elements. Since Boundary Element based methods generate full matrices, and since these matrices in the case of BERIM [10, 74] are $(n + 2m) \times (n + 2m)$, where n and m are the number of elements in the horn walls and the mouth opening, respectively, the solution time can be excessive. In the axisymmetric case, the main part of the computation time is used for the numerical integration, while for the 3D case, most of the time is used for solving the linear set of equations. The large matrices are also challenging in terms of computer memory.

The code was therefore modified to allow symmetry. In addition to integrating over each element with respect to each other element, the mirror images of these other elements are also taken into account. In the case of a quarter-symmetric horn, this reduces the required number of elements in the mesh to one fourth, and since the solution time of a set of N linear equation is of the order $\mathcal{O}(N^3)$, the required time is reduced by a factor of 64.

4 RESULTS

In the study of the MPM for axisymmetric horns[11, 12], the method was compared to the BERIM, a method that couples the BEM in the interior to a Rayleigh integral solution in the exterior[74]. The BERIM is ideally suited to simulate horns that are terminated in an infinite baffle, and as such, is the perfect reference for the MPM.

In the following tests, unless otherwise stated, only one processor core has been used. This is because the BEM/BERIM program that was used has not been adapted to multiple cores, and it was desired to make a realistic comparison of computation times. A more detailed study of the computation times is given in the next chapter.

The horns studied are fairly large. At first it may be thought that a study of smaller horns would be beneficial for reducing the mesh size, but this would also move the frequency range of interest to a higher frequency. This would again require a finer mesh. The size of the horn in terms of the wavelength at the highest frequency of interest, is what determines the mesh size.

In all the test cases, a reference solution from a BERIM simulation is compared to the results from MPM using different numbers of modes.

4.1 SYMMETRICAL CASE

The first test case was a quarter-symmetric exponential horn with the following parameters:

S_{th} Throat area, 10cm², 3.16 × 3.16 cm.

S_m Mouth area, 1600cm², 40 × 40 cm.

L_h Horn length, 60cm.

The horn is shown in Figure 14.

The throat was given a normal velocity of 1m/s.

As a reference, the horn was simulated in 3D BERIM, at 75 frequencies log-spaced in the range 200-2000Hz, with a mesh of 1203 nodes and 2247 triangular elements. The mesh bandwidth was set to 2kHz to avoid excessive memory requirements or computation times. Computation time was 2 hours 43 minutes. The following figures will compare the MPM to the reference BERIM simulation.

It was found that the normalized asymptotic throat impedance level in the BERIM simulations was a little below one. So in all the comparisons, the throat impedance and field point pressure of the BERIM simulations have been multiplied by 1.03. The reason for also

adjusting the radiated pressure is as follows: The throat impedance is calculated from the pressure at the throat surface generated by the throat velocity. Since this pressure is a little low, it was reasoned that pressures at other locations would be offset by the same amount.

As can be seen from the results, the MPM compares favorably with BERIM. At least for a simple horn geometry such as this, not very many modes are needed below 2kHz.

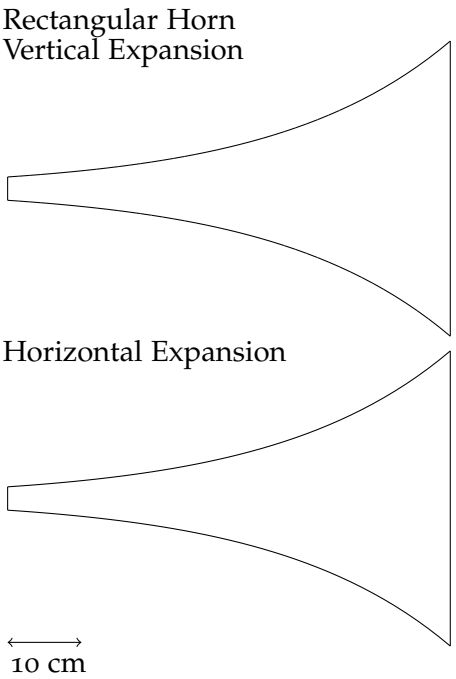
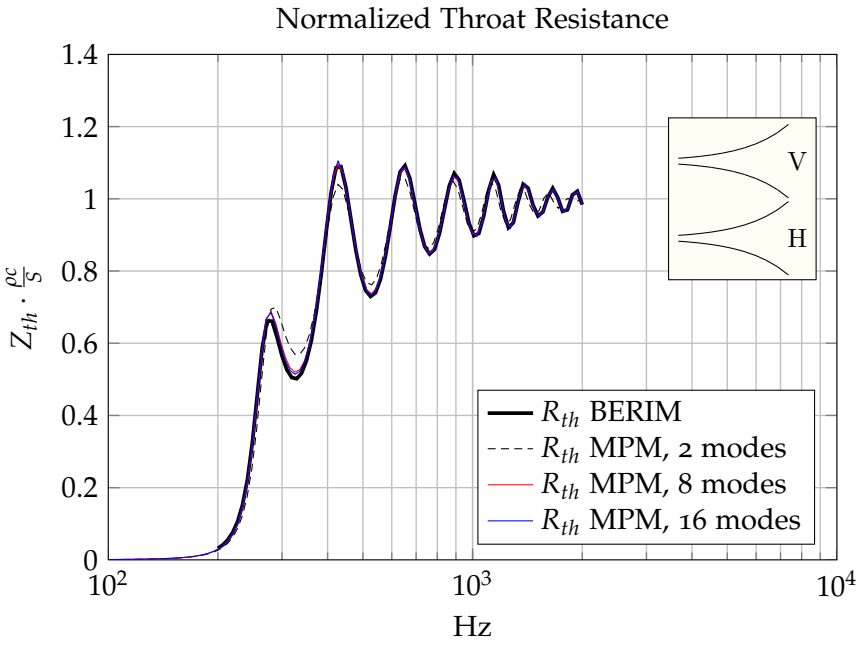


Figure 14: Profiles of symmetrical test horn

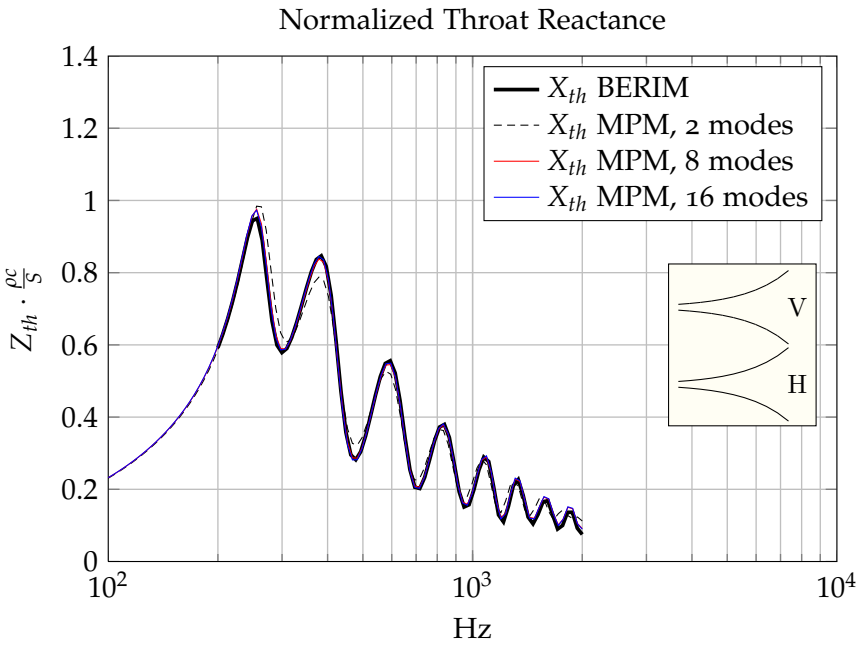
4.1.1 *Throat impedance*

The throat impedance of this horn is shown in Figure 15, for the BERIM reference case, and for 2, 8 and 16 modes in each direction. The three MPM simulations took 6.8 seconds, 5 minutes 21 seconds, and approximately 1 hour 35 minutes respectively.

The results with 8 and 16 modes are almost identical, and both differs just a little from the BERIM reference. For 2 modes, the first impedance peak and the ripple amplitude, are both somewhat off, and the higher frequency impedance peaks are also slightly misplaced.



(a) Throat resistance



(b) Throat reactance

Figure 15: Throat impedance of the quarter symmetric test horn as a function of the number of modes

4.1.2 *Pressure response*

Figures 16, 17 and 18 show the pressure response with 2, 8 and 16 modes, respectively, for the test horn. The response is calculated at 0, 50 and 90 degrees off-axis at a 3 meters distance from the horn

mouth, and is calculated by the two methods outlined in Section 2.7. This is indicated in the legend by the letters RI for Rayleigh integral and M for modal. The Rayleigh integral used simple midpoint rule integration with 25 integration points in each direction.

The response is fairly accurate even for only two modes, up to 1kHz, except that the first resonance peak is slightly misplaced, and the errors increase as we move off-axis.

For both 8 and 16 modes, the response is nearly indistinguishable from the BERIM results.

The modal pressure calculation method has somewhat larger errors than the Rayleigh integral, but this is because this method is a far-field approximation. For larger distances, like 30m, the results are practically identical with the Rayleigh integral results.

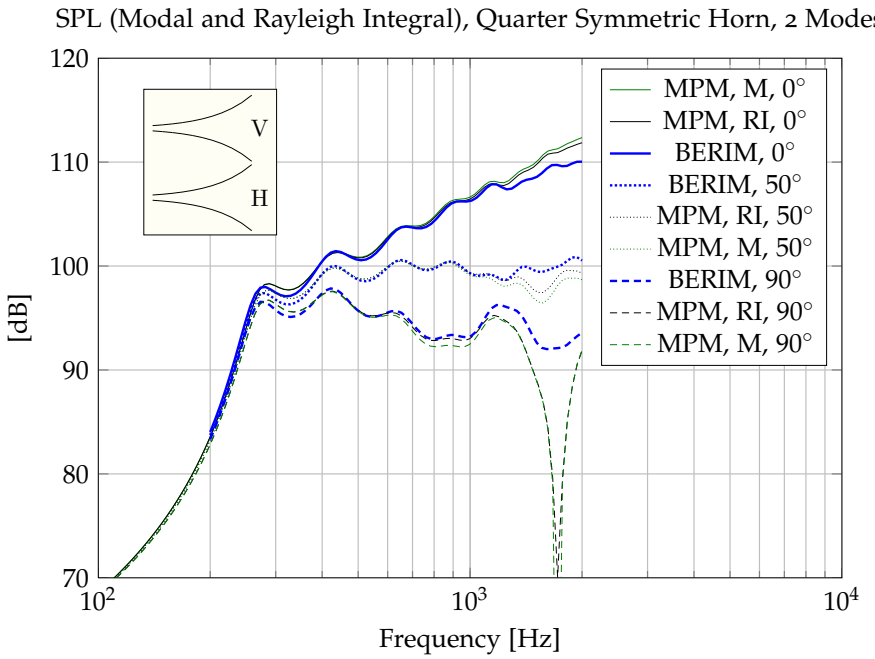


Figure 16: Pressure response of the quarter symmetric test horn, 2 modes. RI indicates that the Rayleigh integral is used in calculating the pressure response (Eq. (102)), M indicates the modal method of Eq. (104)

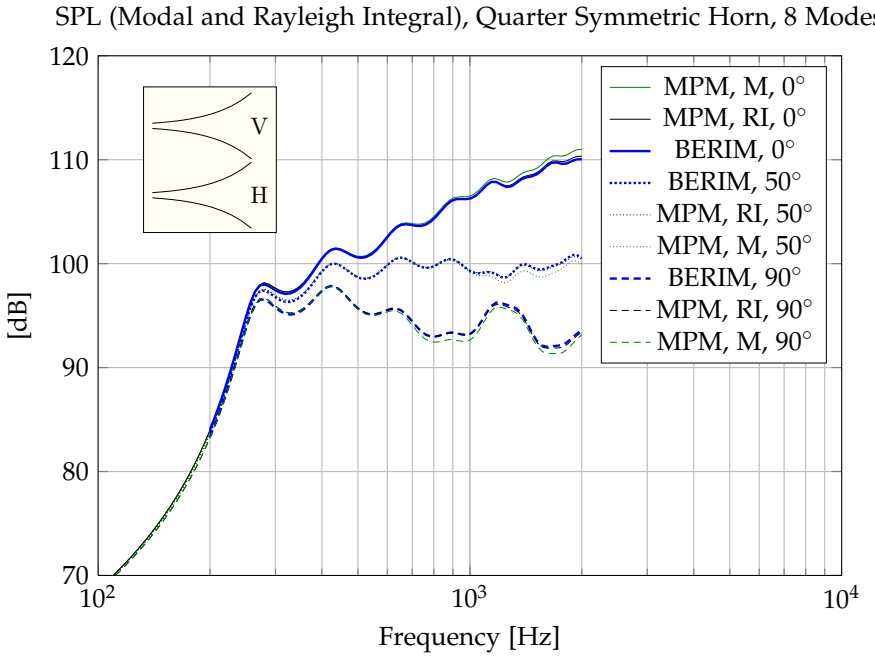


Figure 17: Pressure response for 8 modes, otherwise as in Figure 16

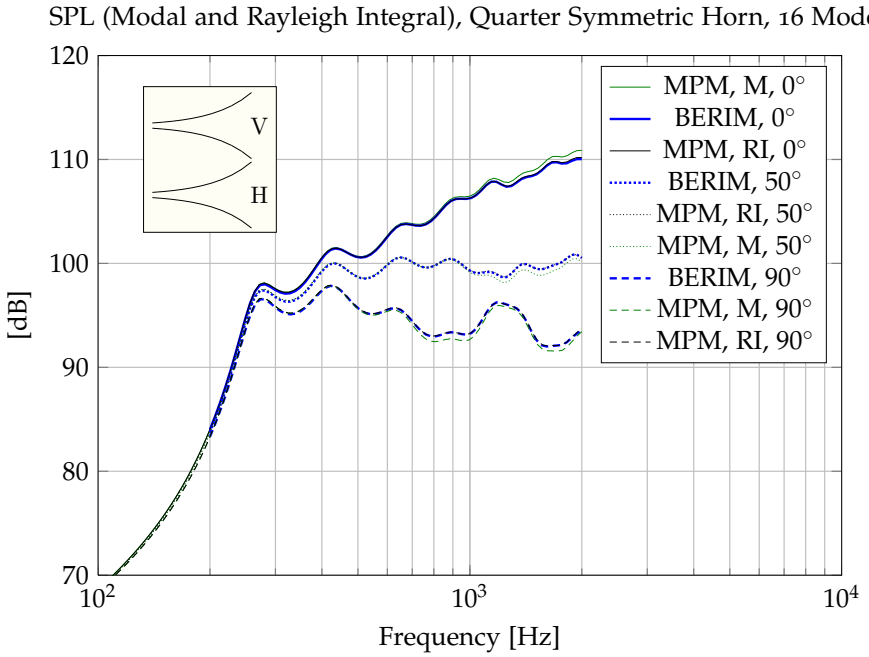


Figure 18: Pressure response for 16 modes, otherwise as in Figure 16

4.2 ASYMMETRICAL CASE

The horn is shown in Figure 19. It is identical to the symmetrical horn, except for the asymmetry, which is 50% vertically at the mouth in this case. 50% asymmetry means that of the increase in the total height from the throat to the mouth, $H_m - H_{th}$, there is 50% more increase upwards than downwards. 100% asymmetry would place all the increase in the upward direction, and 0% symmetry would place the increase equally upwards and downwards.

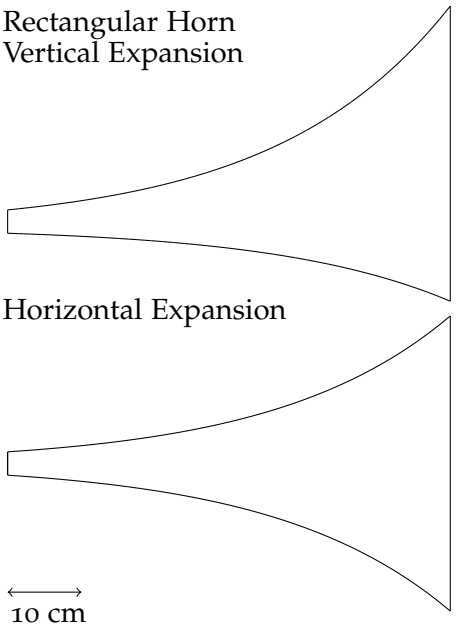


Figure 19: Profiles of asymmetrical test horn

Other conditions are as for the symmetrical case.

Again as a reference, the horn was also simulated in 3D BERIM, at 75 frequencies log-spaced in the range 200-2000Hz, with a mesh of 2435 nodes and 4702 triangular elements. Computation time was 14 hours 38 minutes. The following figures will compare the MPM to the reference BERIM simulation.

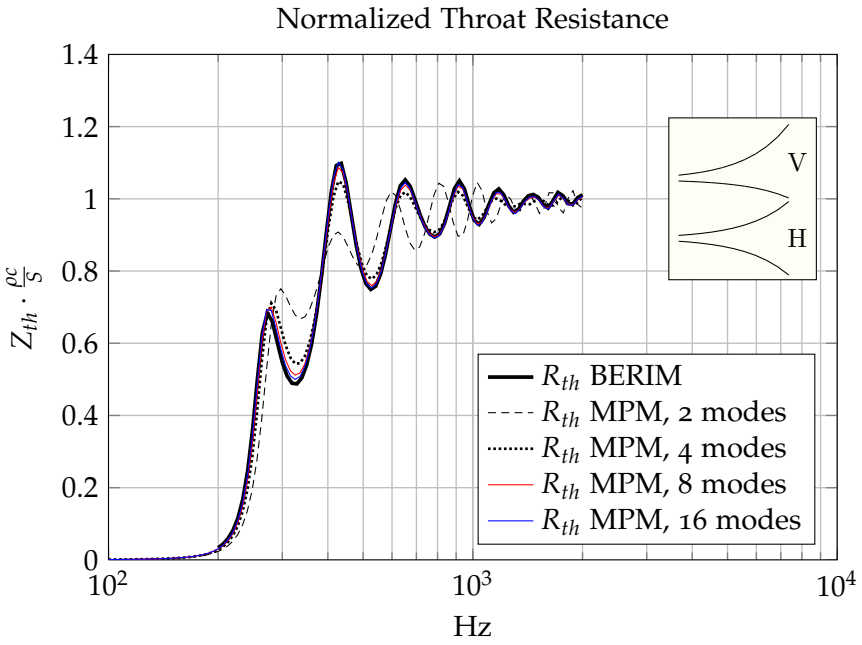
The four MPM simulations for 2, 4, 8 and 16 modes took 8.5 seconds, 30 seconds, 5 minutes 42 seconds and 1 hour 51 minutes 21 seconds, respectively.

4.2.1 *Throat impedance*

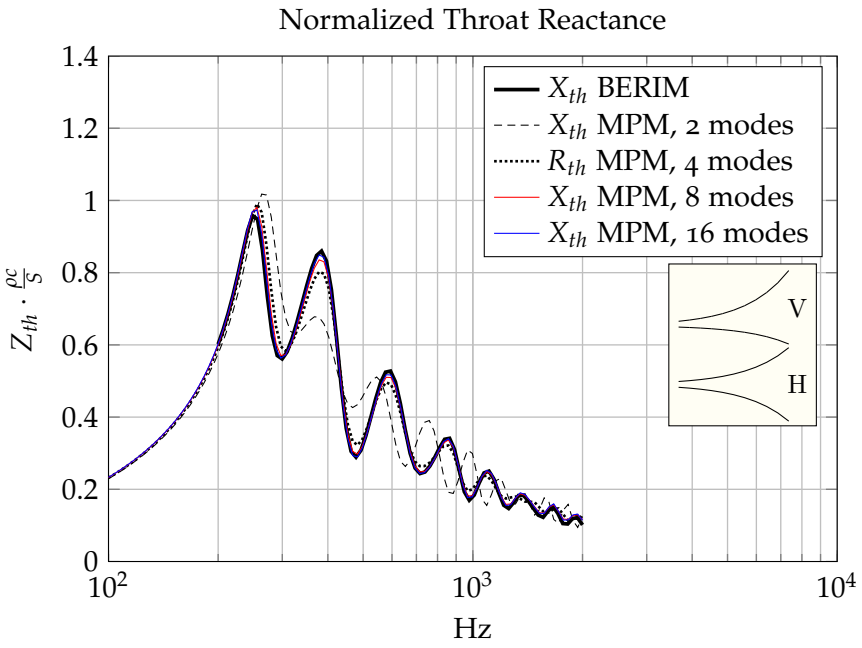
The throat impedance of this horn is shown in Figure 20, for the BERIM reference case, and for 2, 4, 8 and 16 modes in each direction.

For 2 modes, there is a clear difference, most of the impedance peaks are misplaced, and the ripple amplitude is different. The difference is much larger than in the symmetrical case, and the reason is most likely that no symmetrical modes are taken into account with

a total of 2 modes. 4 modes is a clear improvement, it corresponds to 2 symmetrical modes, and the error can be seen to be in the same range as for 2 modes in the symmetric case. The results with 8 and 16 modes are almost identical, and both differ just a little from the BERIM reference, except at the lowest frequencies. For all plots, it can be seen that the resonance frequencies move closer to their real values as the number of modes is increased.



(a) Throat resistance



(b) Throat reactance

Figure 20: Throat impedance of the asymmetric test horn as a function of the number of modes

4.2.2 *Pressure response*

Figures 21, 22 and 23 show the pressure response with 2, 4, 8 and 16 modes, respectively, for the test horn. The response is calculated at vertical angles -70, 0 and +70 degrees off-axis at a 3 meters distance

from the horn mouth. The Rayleigh integral used simple midpoint rule integration with 100 integration points in each direction.

For 4 modes, the response is fairly close to the BERIM results, especially for on-axis response, and for positive angles (upward, where the horn flares the most), up to about 1kHz. At negative vertical angles, the differences are larger.

For both 8 and 16 modes, the response is very close BERIM results, but not quite as close as for the quarter-symmetric case. Part of the explanation is that 16 asymmetric modes correspond to only 8 symmetric modes, and a calculation including 32 modes would have to be performed for an accurate comparison with the 16 mode quarter-symmetric case.

It was also found that more integration points were required for the evaluation of the Rayleigh integral in the asymmetric case. In the examples 100 integration points were used, as 25 turned out to be too few, see Figure 24. A better integration method, using for instance Gauss-Legendre quadrature, would be beneficial.

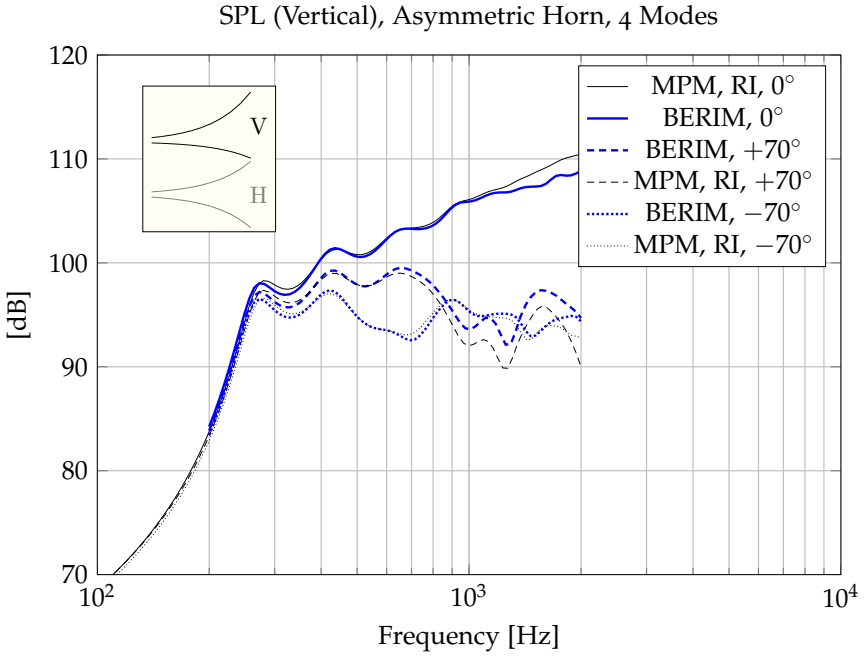


Figure 21: Pressure response in the vertical plane of the asymmetric test horn, using the Rayleigh integral (Eq. (102))

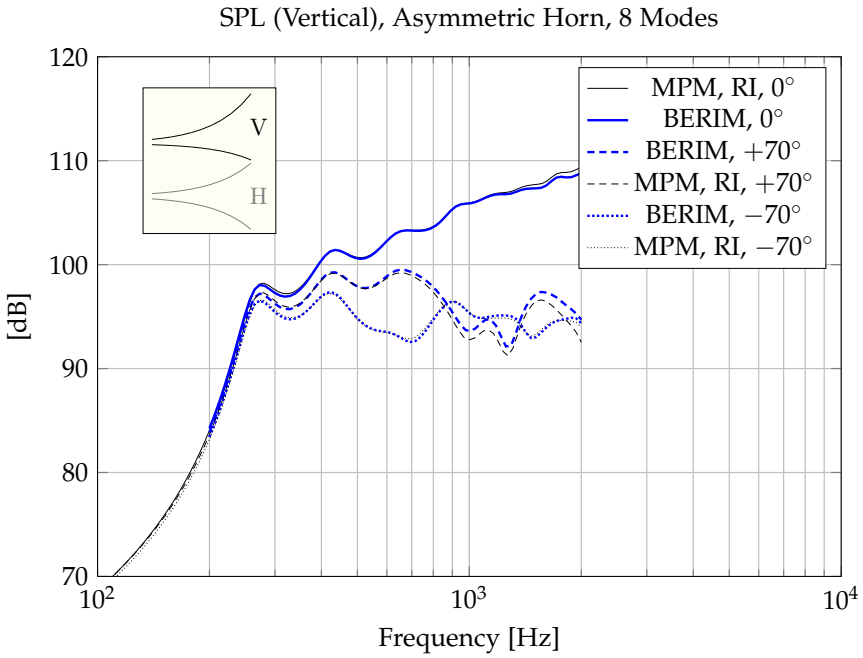


Figure 22: Pressure response for 8 modes, otherwise as in Figure 21

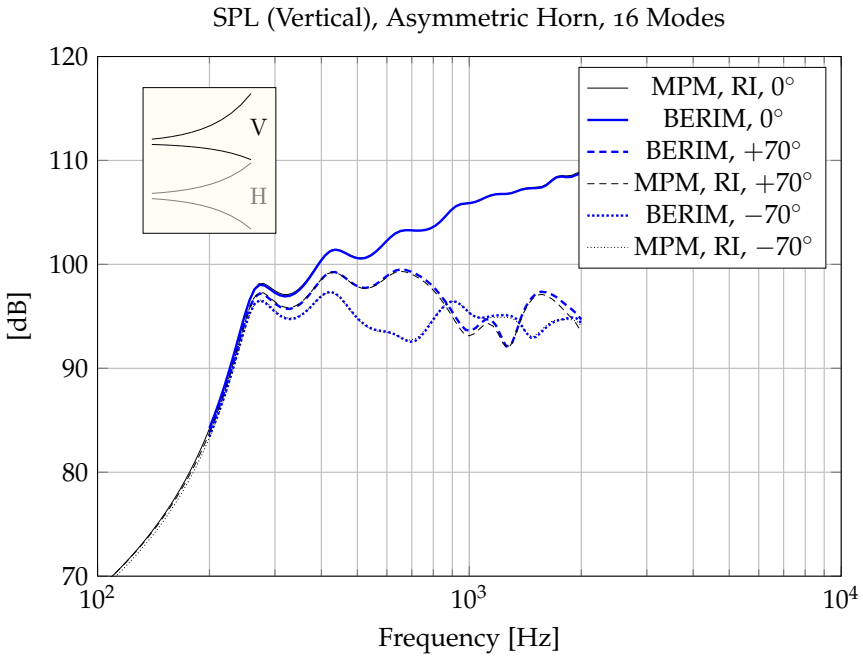


Figure 23: Pressure response for 16 modes, otherwise as in Figure 21

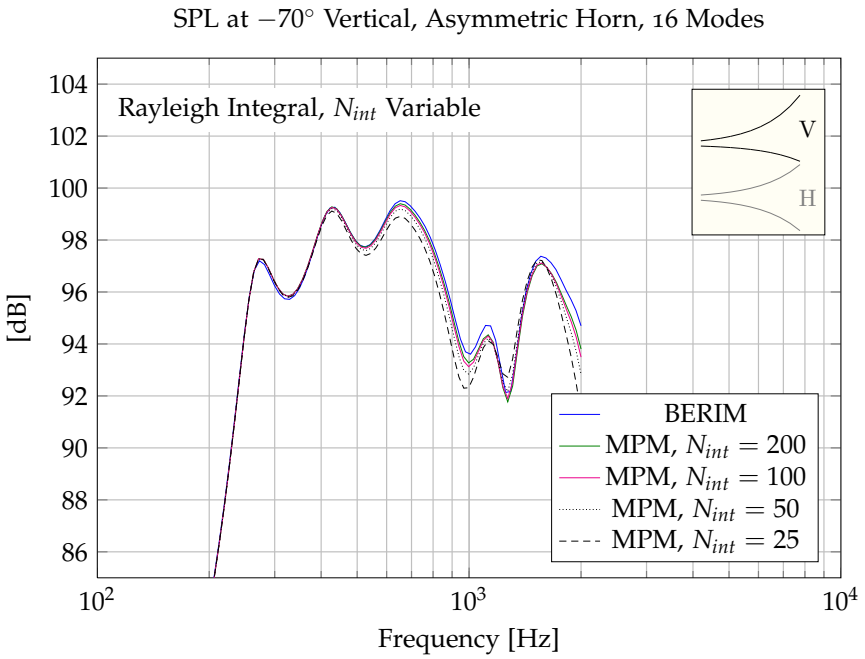


Figure 24: Variation in computed pressure response with varying number of integration points in the Rayleigh integral. Please note the change of vertical scale

4.3 "PINCHED" HORN

This horn is designed to test the accuracy of the case where the horn expands in one plane and contracts in the other. The profile is shown in Figure 25. The vertical expansion is conical, forcing the horizontal expansion to start with a contraction, a "pinched" part, to keep the required area expansion. The horn is a Hypex horn [75] of the same length and terminal dimensions as the previous cases. The area expansion is given by

$$S(x) = S_{th} \left(\cosh k_c x + T \sinh k_c x\right)^2 \tag{113}$$

where k_c is the cutoff wavenumber. In this example $T = 0$. This produces a horn that expands slowly near the throat (the slope of an axisymmetric horn would be zero at the throat), and slowly approaches the exponential horn profile. This horn type is also known as a Catenoidal horn.

As a reference, the horn was also simulated in BERIM 3D, at 75 frequencies log-spaced in the range 100-2000Hz, with a mesh of 1259 nodes and 2353 triangular elements. Computation time was 2 hours 49 minutes. The following figures compare the MPM to the reference BERIM simulation.

The four MPM simulations for 2, 4, 8 and 16 modes took 5.5 seconds, 21 seconds, 4 minutes 9 seconds and 1 hour 7 minutes 16 seconds, respectively. This time, 75 frequencies were used, and the frequency range was 100Hz to 2kHz as before.

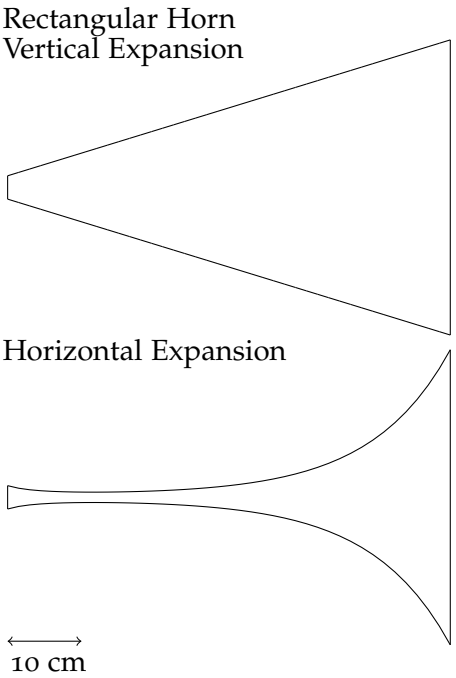


Figure 25: Profiles of a "pinched" horn

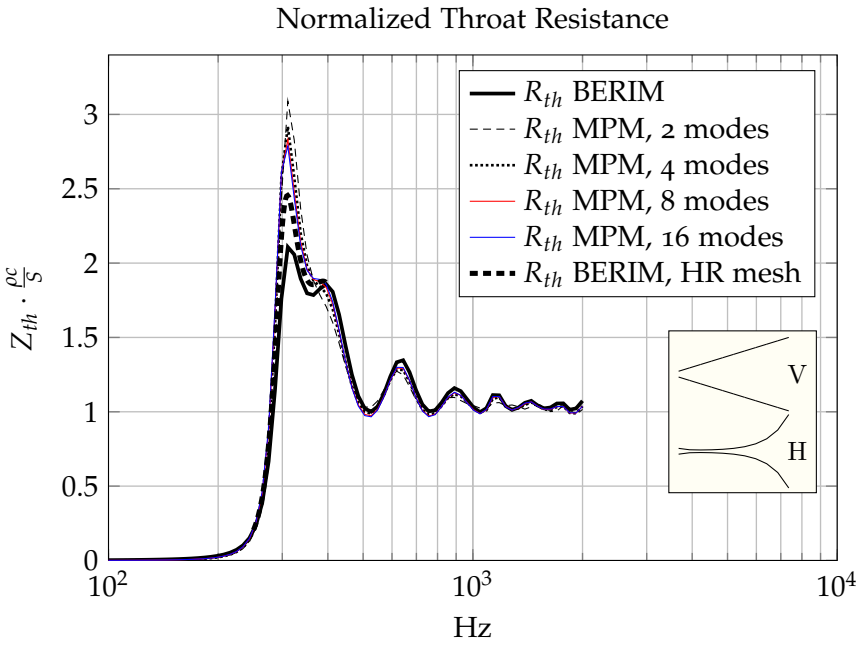
4.3.1 *Throat impedance*

The throat impedance of this horn is shown in Figure 26, for the BERIM reference case, and for 2, 4, 8 and 16 modes in each direction. As can be seen, the results are not nearly as good as in the case where the horn expands in both planes. This was at first surprising, and the theory and implementation were checked to see if any errors had been made. Investigations of the mathematics of the F-matrices for this case, given in section 2.6.1 did not turn up any mistakes, neither could there be found any problems in the implementation. It was thought that the way the V-matrices are handled was suspect, so a test case of an axisymmetric horn with both expansion and contraction was compared to BERIM. With 16 modes, the impedance curves were overlapping up to 4kHz.

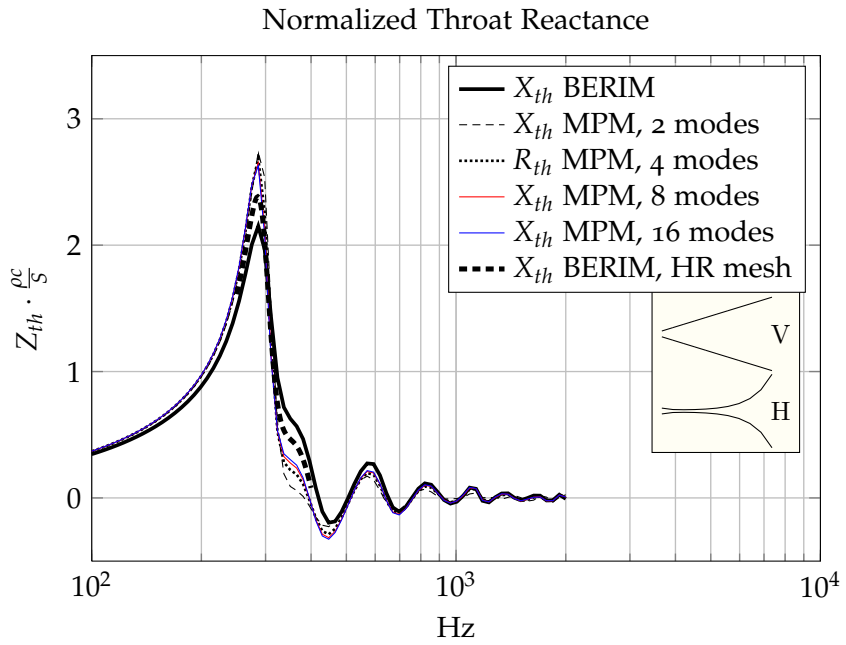
A new test was done in which the discretization of the horn into straight segments was changed slightly, so that a segment where the horn had both expansion and contraction was split into two segments, one only expanding and one only contracting. This did not change the outcome of the simulation.

It was not practical to test the MPM with more modes, as this would require another time-consuming round of radiation impedance computations. Since the computation time increases with mode number, the extra modes would most likely take even longer than the first set. But it can be seen that as the number of modes increase, the results seem to converge to a value very close to the BERIM results above 400Hz. Below 400Hz, the results do not converge to the BERIM results, but they do seem to converge. One reason could be that the contraction needs very many modes to converge to the BERIM results. It was also thought possible that the BERIM results were inaccurate in this frequency range, and that smaller elements were needed.

In order to test this, a simulation was run with a finer mesh, 2353 nodes and 4482 elements, for 75 frequencies between 250 and 400Hz. This simulation took 16 hours 48 minutes, and the results are shown as 'BERIM HR mesh' in Figure 26. These results show that the narrow part of the horn actually need more elements than dictated by the 6 elements per wavelength rule, even for low frequencies. What was most surprising, was that the errors are largest in the low frequency range. A view of the meshes near the throat is shown in Figure 27, and it can be seen that the medium resolution mesh has a longer part where there is only one triangle spanning the width of the horn. It was unfortunately not possible, due to memory requirements (the BEM/BERIM application was not able to handle more than 2GB of memory), to run the simulation with a finer mesh. A different problem could have been defined, but there was not enough time to do that.

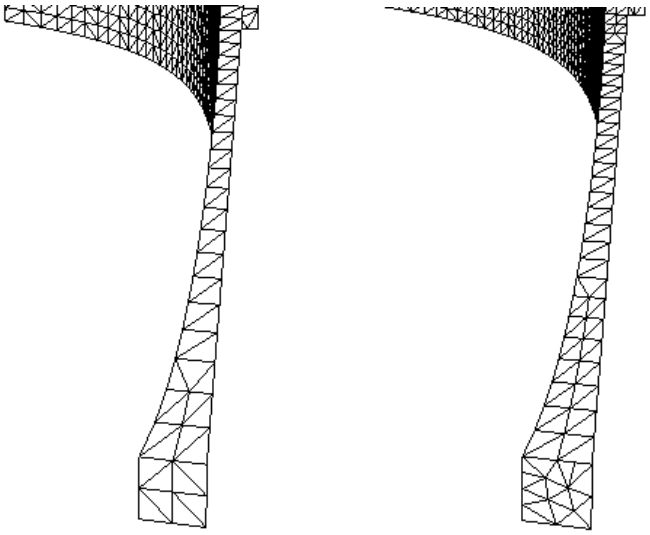


(a) Throat resistance



(b) Throat reactance

Figure 26: Throat impedance of the 'pinched' test horn as a function of the number of modes. 'BERIM, HR mesh" indicates the results from using the higher resolution mesh in Figure 27b



(a) Medium resolution BERIM mesh

(b) Higher resolution BERIM mesh

Figure 27: BERIM meshes

4.3.2 *Pressure response*

Figures 28, 29 and 30 show the pressure response with 4, 8 and 16 modes, respectively, for the test horn, for the horizontal plane. Figures 31, 32 and 33 show the response for the vertical plane. The response is calculated at angles of 0, 50 and 90 degrees off-axis in both planes, at a 3 meters distance from the horn mouth. The Rayleigh integral is used, with a simple midpoint rule integration using 50 integration points in each direction.

For 4 modes, the response is fairly close to the BERIM results in the range 400-1000Hz, especially for on-axis response.

For both 8 and 16 modes, the response is very close BERIM results above 400Hz. Clearly, the MPM converges to the BERIM results in this frequency range. The deviations below 400Hz come to a large degree from the difference in throat impedance. It is higher in the MPM case, and consequently this results in slightly higher radiated power at these frequencies.

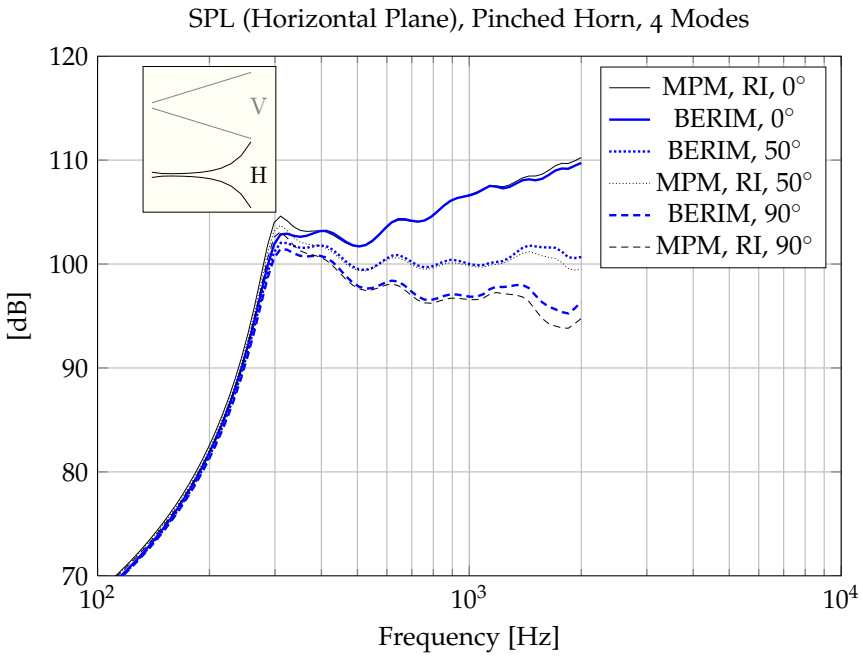


Figure 28: Pressure response (horizontal) of the 'pinched' test horn. Rayleigh integral computation of the radiated pressure

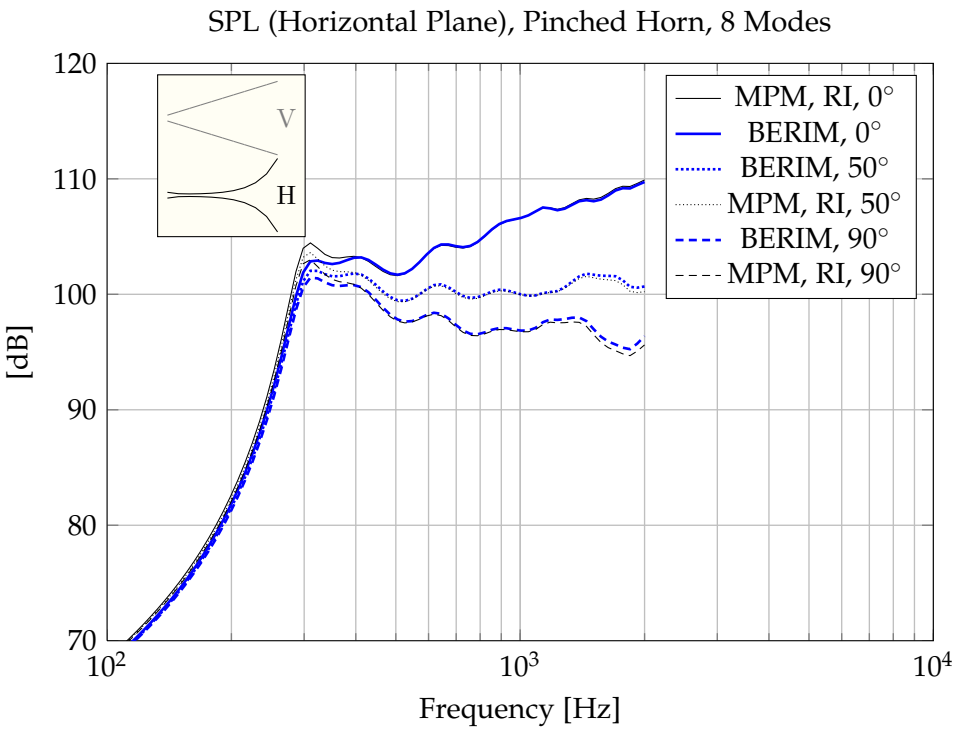


Figure 29: Pressure response, 8 modes, horizontal, otherwise as Figure 28

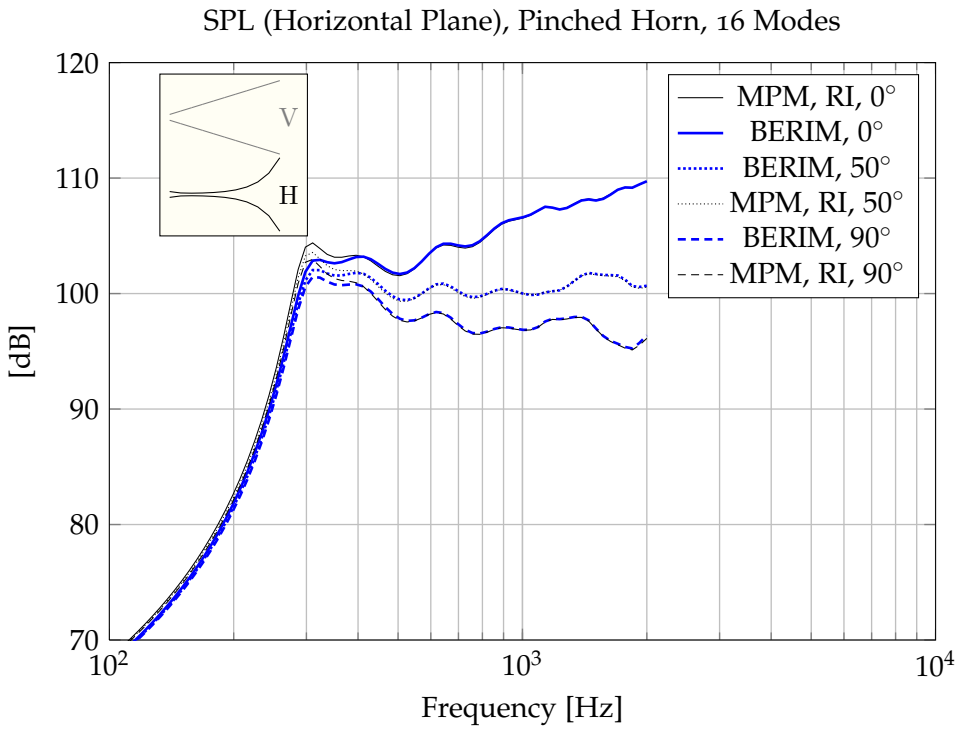


Figure 30: Pressure response, 16 modes, horizontal, otherwise as Figure 28

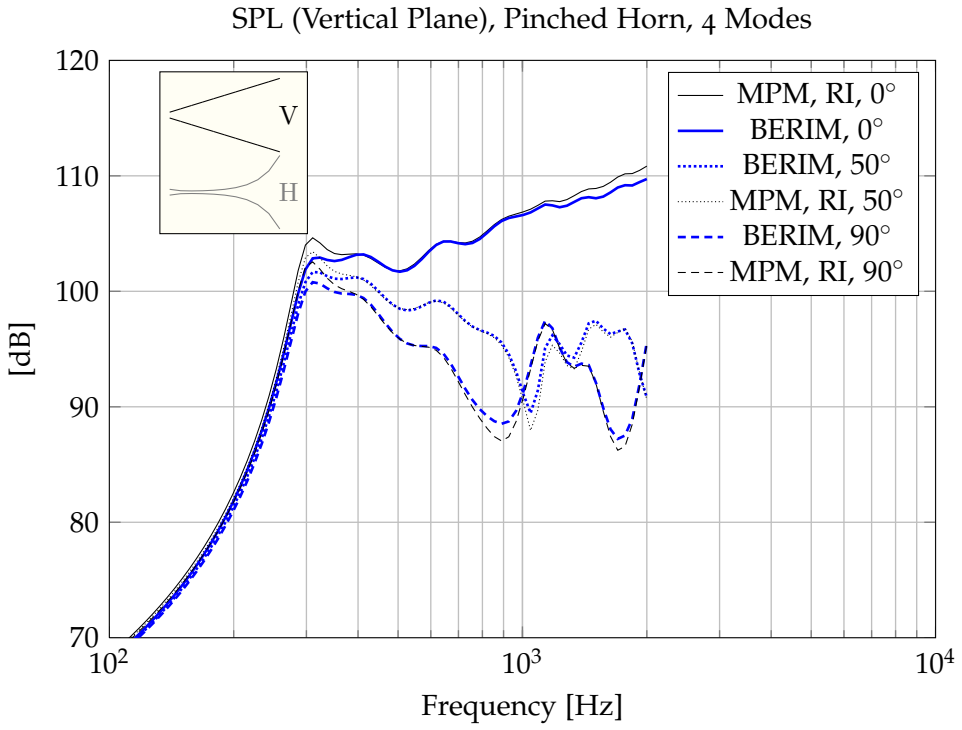


Figure 31: Pressure response, 4 modes, vertical, otherwise as Figure 28

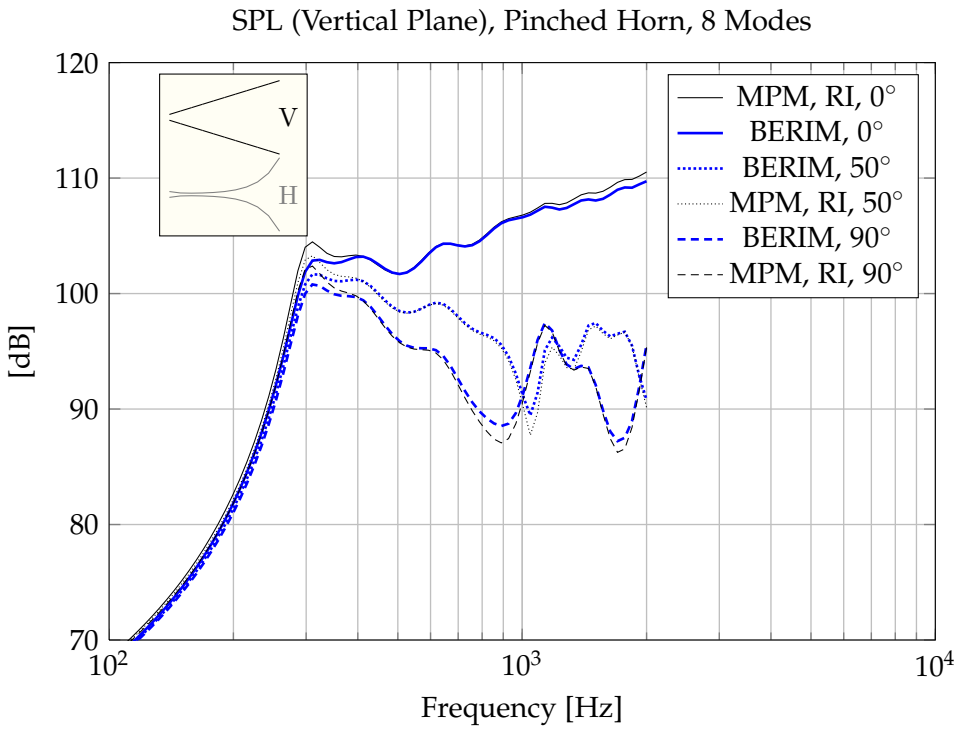


Figure 32: Pressure response, 8 modes, vertical, otherwise as Figure 28

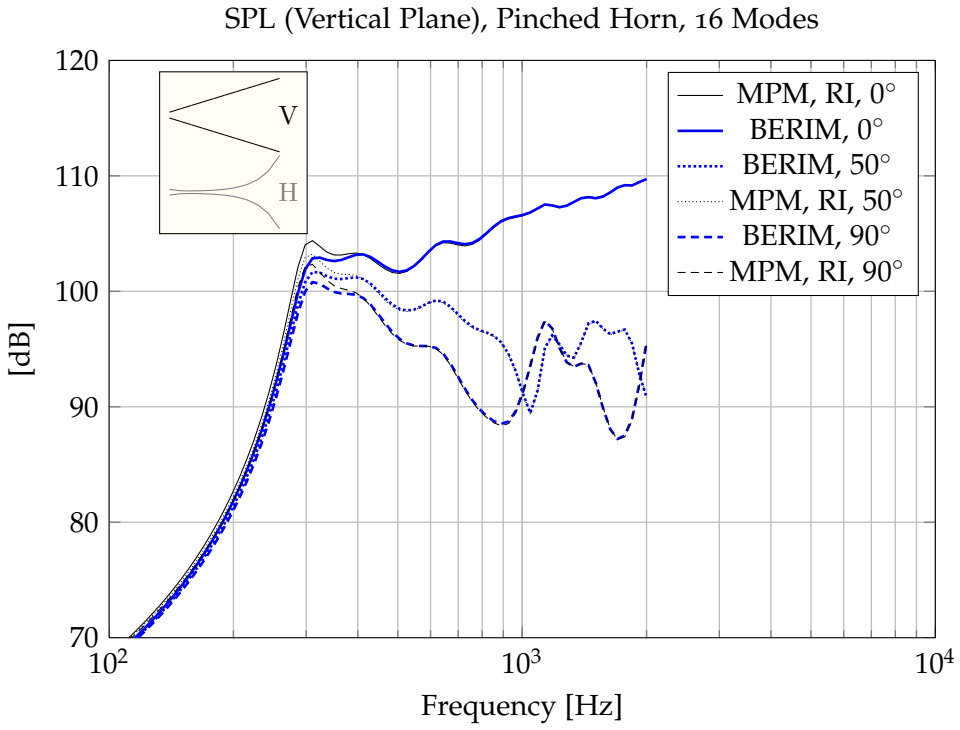


Figure 33: Pressure response, 16 modes, vertical, otherwise as Figure 28

4.4 CURVED DUCTS

The MPM has not been implemented for curved horns and ducts. But as an example of how complex the sound field in a curved duct can be, Figure 34 shows the real part of the pressure in a 180° bend excited with a plane pressure wave (Dirichlet boundary condition) at the right end, and no reflected wave entering the far end. Inner radius is 0.2 m, outer radius 0.5 m, and $k = 15$. It is assumed that there is no pressure variation in the z-direction. Under these conditions, there are two propagating modes in the bend. Only the propagating modes are included in the sound field.

A comparison was done, comparing this simulation with 2D BEM. The input end of the bend was driven with a pressure release surface, while the output end was terminated in a 1.0m long duct with absorbing walls, to simulate an anechoic termination.

Since the visualizer in the BEM SW only shows the magnitude values at a dB scale, display of results from the modal method was changed to logarithmic z-values, as shown in Figure 35. The results from BEM are shown in Figure 36. The same magnitude range has been used. The results are similar, but far from identical. Some of the features of the sound field can be recognized, but not all. Since the modal method does not include the evanescent waves, this may be one reason for the discrepancy.

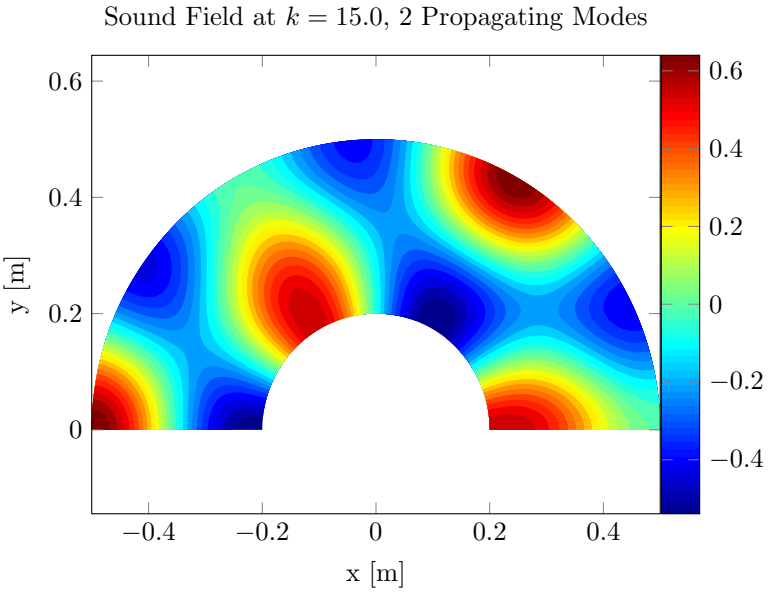


Figure 34: Sound field in a bend (real part)

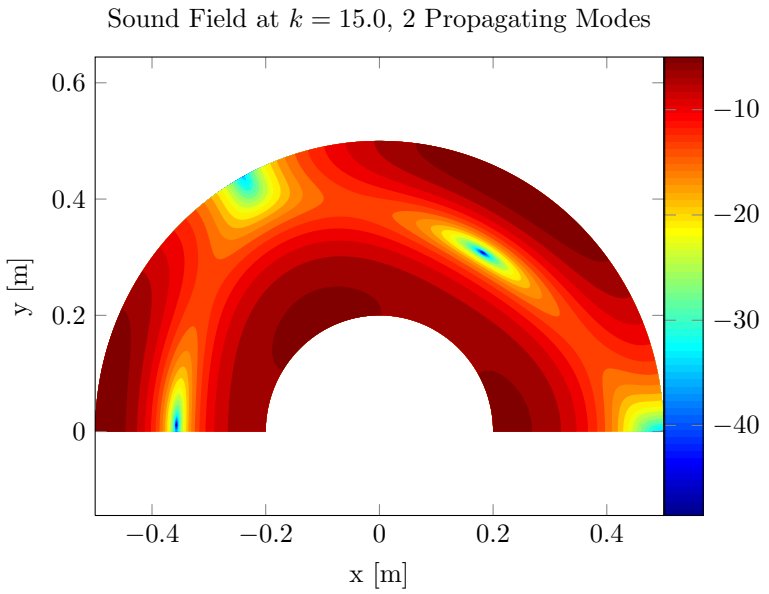


Figure 35: Magnitude plot (in dB) of the sound field in Figure 34

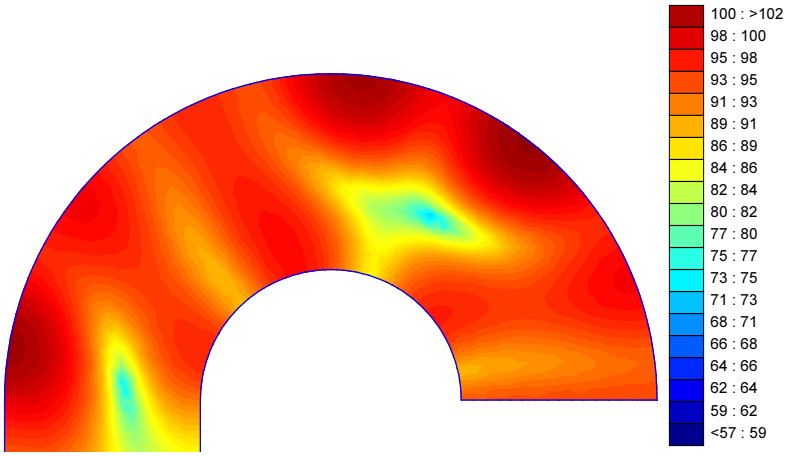


Figure 36: Magnitude plot (in dB) of the sound field in the same bend as above, simulated in 2D BEM

5 DISCUSSION

5.1 COMPUTATION TIMES

The computation times for the tests are shown in Table 3. Results for the "pinched" horn, scaled to 100 frequencies, are also included.

By scaling the computation times to 75 frequencies (since computation time increases linearly with the number of frequencies), the results in Table 4 are produced. This table shows that the MPM with 16 modes is around 2 times faster than BERIM in the symmetrical case, but nearly 8 times faster in the asymmetrical case. If the horn had not had any symmetry plane, the gain would have been even larger. This does not take into account the fact that for the same accuracy as in the symmetrical case, the asymmetrical case with the MPM would need 32 modes, increasing the computation cost considerably. As mentioned, by doubling the number of modes in each direction, the matrices become four times as large in each dimension. With the most basic methods of matrix multiplication and inversion, both $\mathcal{O}\left(N^3\right)$ operations, the theoretical increase in computational cost would be 64 times.

It can be seen that the increase is not as large as this for the examples given, when the number of modes is doubled. The main reason for this, is the overhead in computing the F-matrices and interpolating the radiation impedance.

There is no way to simply exploit half-symmetry, as is possible in general mesh-based methods like BERIM.

A series of calculations, using the symmetric horn test case, 75 frequencies and the frequency range 200-2000Hz for easy comparison with BERIM, was made. Throat impedance and on-axis response were calculated for different numbers of modes. The computation times as a function of N_{modes} for the symmetric case are shown on a *log-log* scale in Figure 37. In addition to the times for the various parts of

	Symmetric H.	Asymmetric H.	Pinched H.	P.H., scaled
BERIM	02:43:00	14:38:00	2:49:29	—
2 modes	00:00:06.8	00:00:08.5	00:00:05.5	00:00:07.3
4 modes	—	00:00:30	00:00:21	00:00:28
8 modes	00:05:21	00:05:42	00:04:09	00:05:32
16 modes	01:35:00	01:51:21	01:07:16	01:29:42

Table 3: Comparison of computation times

	Symmetric H.	Asymmetric H.	Pinched H.	P.H., high res.
BERIM	02:43:00	14:38:00	02:49:29	16:47:59
2 modes	00:00:05.2	00:00:06.4	00:00:05.5	
4 modes	—	00:00:22	00:00:21	
8 modes	00:04:01	00:04:17	00:04:09	
16 modes	01:11:15	01:23:31	01:07:16	
Gain, 16 m.	2.29	7.89	2.03	15.0

Table 4: Comparison of computation times, scaled to 75 frequencies

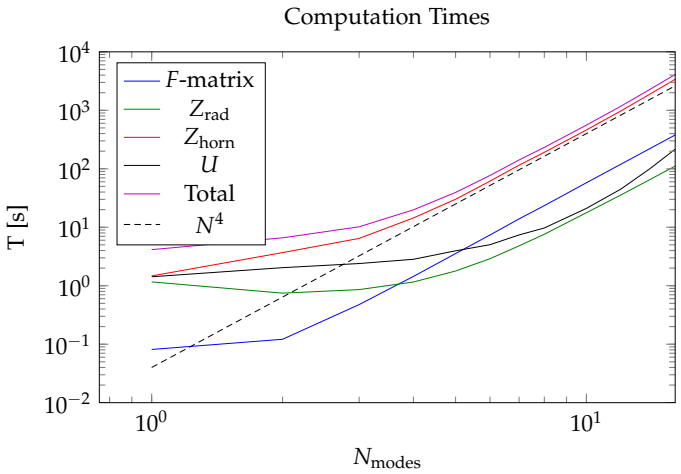


Figure 37: Computation times as a function of N_{modes}

the algorithm, the total time, and a line corresponding to a constant times N^4 are plotted. It can be seen that in the range $N = 5 \ldots 16$, the slopes of most of the curves correspond to this line.

The main part of the computation time is the propagation of the radiation impedance back to the throat. For few modes, the computation of the radiation impedance and propagation of the volume velocity is a large part of the cost. With increasing number of modes the computation of the F-matrices becomes an increasingly important component, but this part is dependent on how many frequencies are computed. Since these matrices are computed only once, the component will still be relatively small in the case when many frequencies are used.

It can be seen that there is a slight increase of slope for the computation time for the impedance near the end of the line, compared to the straight N^4 line. This may indicate that for higher orders, the computation time increases even faster.

5.2 ACCURACY

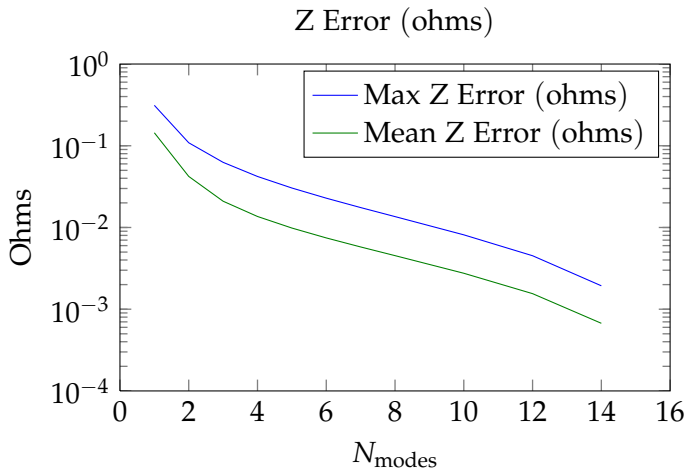
No detailed study of the accuracy of the method has been made, but a few comparisons can be made on the basis of data gathered in the timing experiment mentioned above. Two comparisons are made, and for both cases both the mean and maximum errors in response and throat impedance over the 200-2000Hz range are calculated.

In the first case, the results are compared to the results for 16 modes. There is a steady decline, with no indication of leveling out. The decrease is largest for the first 3–4 modes, for more modes the improvement for each added mode is less. Figure 38a and Figure 39a show the results for impedance and on-axis response.

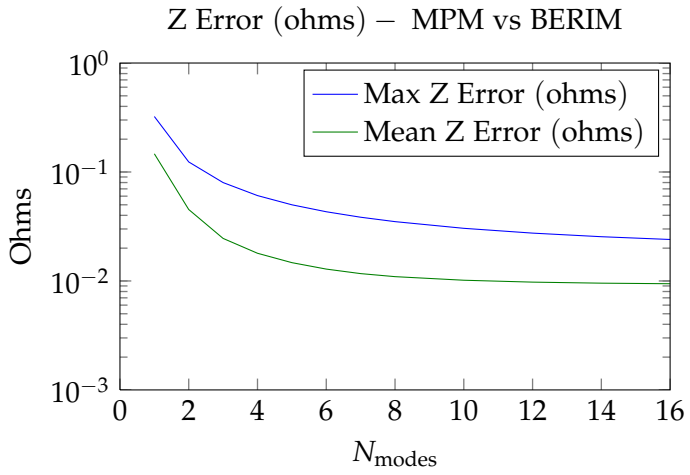
When the results are compared to BERIM, the results show a convergence, although not complete, towards the BERIM results. Results indicate that for the frequency range considered, 8 modes are enough for a 0.3dB maximum error in response and 3% maximum error in throat impedance.

That the errors level out instead of decreasing to zero, may indicate that the BERIM mesh is too coarse also in this case, as was the case with the "pinched" horn in Section 4.3.

While MPM may require many modes for high accuracy, BERIM also requires a fine mesh for the same accuracy. The advantage of MPM is that the accuracy is also acceptable for relatively few modes, like 8, in which case the solution time is merely a few minutes. The N^4 increase in computation time makes it hard to obtain very high accuracy without high cost, but it is easy to obtain fair accuracy in a short time.

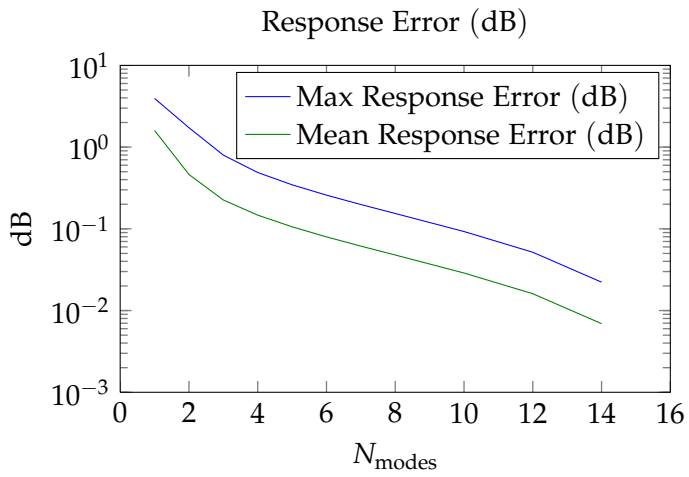


(a) Impedance error (in normalized acoustical ohms), results from 1–14 modes, compared to the result for 16 modes

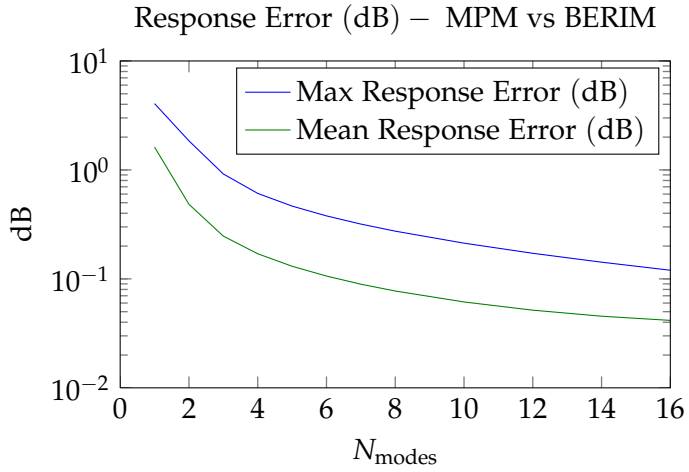


(b) Impedance error (in normalized acoustical ohms), compared to BERIM results

Figure 38: Impedance error as function of the number of modes, mean and maximum



(a) Frequency response error, results from 1–14 modes, compared to the result for 16 modes



(b) Frequency response error, compared to BERIM results

Figure 39: Error in on-axis pressure response as function of the number of modes, mean and maximum

5.3 MEMORY REQUIREMENTS

The MPM is an easily scalable model with good accuracy. However, with the sizes of the matrices, the number of elements also increasing as N^4, most of them also of the complex double precision data type, memory requirements can be taxing for MPM. For 16×16 modes, this amounts to 1GB of storage for each matrix. In practice, several matrices have to be kept in memory at the same time. The practical limit on a computer with 8GB RAM turned out to be 24×24 modes.

Similar memory requirements do of course also apply to BERIM, but the memory requirements do not grow quite as fast.

5.4 SUGGESTIONS FOR FURTHER WORK

The problem of modal propagation in curved ducts poses some numerical challenges. This section describes further work that must be completed before curved horns can be simulated by this method.

5.4.1 *Riccati equation*

The method used by Felix and Pagneux, outlined in section 2.4.5, requires integration of a Riccati impedance equation. This equation, reproduced here as Eq. (114), is in matrix form.

$$Z' = -jkB - \frac{1}{jk} Z(C + KB)Z \tag{114}$$

Matlab has two built-in functions for integrating ordinary differential equations (ODEs), `ode23` and `ode45`, which both use the Runge-Kutta method to different orders. These methods can, however, not be used, because it is necessary to implement the integration in matrix form. The matrices cannot be integrated term by term, due to the modal coupling throughout the duct. The coupling has to be taken into account at each step of the integration, and the resulting impedance matrix stored for future propagation of velocity. Felix and Pagneux advocate the use of a Runge-Kutta method with adaptive step size [27].

In addition, a fixed step size integration of the velocity differential equation through the bend must be performed. This requires another, different implementation of the Runge-Kutta method in matrix form.

5.4.2 *Reference solution*

A reference solution for a curved horn is necessary to test the method. While a method to create curved horns has been included in HornCAD, the meshing of this geometry has not yet been implemented.

An alternative is to do the simulation in Comsol. But to make the same geometry in Comsol as would be used for the modal method, some way of importing the geometry into Comsol will be required.

5.4.3 *Other suggestions*

The accuracy of the method, as applied to rectangular horns, should be studied in detail. This has been done for axisymmetric horns, but in the case of rectangular horns there is an additional degree of freedom that may reduce the gain in speed over BERIM found for the axisymmetric case.

There is currently no way to directly calculate the far-field radiated pressure from the mouth velocity mode amplitudes for asymmetric horns. This is yet another suggestion for future work.

Better integration methods for the Rayleigh integral (like Gauss-Legendre quadrature) also need to be implemented. The Rayleigh integral is important when calculating the near field pressure.

Apart from this, the same suggestions apply as for the axisymmetric case, and will be repeated here:

A reformulation of the method could in some cases be useful, and would perhaps also speed up the calculations. If the matrices for each element and discontinuity could be combined in a way that related pressure and volume velocity at the throat to pressure and volume velocity at the mouth, the throat impedance could be quickly computed (by a single matrix multiplication) for any radiation impedance, without recalculating the horn. In the same way, the mouth volume velocity could be computed for any throat volume velocity, without having to propagate it through each element of the horn. This would also significantly reduce storage requirements, as the present method requires the impedance matrix at all points in the horn to be stored, for all frequencies. For detailed simulations, it is quite possible to run out of memory. This is especially true for rectangular horns, as the number of modes is the square of the number of modes required in the axisymmetric case. In the axisymmetric case, the impedance matrix was first calculated for the entire horn for all frequencies, then the volume velocity was propagated from throat to mouth. Due to memory limitations, the implementation of the rectangular case had to consider one frequency at a time, first computing the impedance matrix for one frequency, then the volume velocity. However, Felix and Pagneux [27] point out that this reformulation is not numerically stable due to the evanescent modes. The presence of the evanescent modes is the main reason for computing the impedance matrix through the horn first.

Further work on the method should also look into more reliable ways to compute the maximum size of the elements for a given con-

tour, and the maximum number of modes that can be used for a given element size.

6 CONCLUSION

In this project, the Modal Propagation Method (MPM) has been investigated for straight and curved rectangular horns.

The basic method had previously been implemented for axisymmetric geometries, and it was desired to extend the technique to rectangular and curved geometries. In the process of extending the method to curved geometries, asymmetry of rectangular horns had to be taken into account. This work is new. Also new is the extension to horns expanding in one plane and contracting in the other.

Comparisons with BERIM show good agreement, both in terms of throat impedance and radiated pressure, except near cutoff for certain cases. Particularly horns with an expansion in one plane and a contraction in the other, and with a very thin part, had some deviation that appears to be caused by a too coarse mesh in the BERIM simulation. This observation is interesting, in that it indicates that for certain horn geometries, BERIM is not accurate enough even at low frequencies. The very fine mesh that is required to overcome this problem, with the resulting increase in computation time, gives the MPM a definite advantage over BERIM.

In general, the MPM shows a markedly higher speed than full 3D BERIM. The gain is large for asymmetric horns, where BERIM requires a much larger mesh than for symmetric horns. Still, MPM would require more modes for certain geometries, which reduces the advantage again, as the computation time increases approximately as N^4, where N is the number of modes in each direction.

Computation of the modal radiation impedance of a rectangular duct has been performed for the simplified case of a square duct. The resources required to perform the computation were rather larger than expected, and the author is indebted to Andreas Asheim for the Matlab code for performing double integrals by the Numerical Method of Steepest Descent. This probably saved many hours (or even days) of computation time.

As has been shown, the mathematics describing modal sound propagation in duct bends is complicated, and there are considerable challenges in the implementation. These come partly from the mathematical difficulties, and partly from the lack of available numerical routines, for instance for Bessel functions of imaginary order.

So while the MPM has not been fully implemented for the curved horn case, the following is work considered to be of value:

- Extension of the MPM to asymmetric rectangular horns

- Extension of the MPM to quarter symmetric horns expanding in one direction and contracting in the other (this is easily extended to asymmetric horns)
- Application of the method of steepest descent to compute an oscillatory double integral, to the radiation impedance matrix
- A literature search on modal descriptions of bends in rectangular ducts, which also revealed that work is currently being done in this field in both quantum mechanics and electromagnetics.
- A literature search on the computation of Bessel functions of imaginary order, and on the computation of roots of the dispersion relation

It is hoped that the literature study will be of value for further research on this topic. It is also hoped that the work done on symmetric and asymmetric rectangular horns can be useful for fast simulation of these types of horns.

BIBLIOGRAPHY

[1] J. A. Kemp, "Theoretical and experimental study of wave propagation in brass musical instruments," Ph.D. dissertation, University of Edinburgh, 2002. [Online]. Available: http://www.kempacoustics.com/thesis2/ (Cited on pages iii, v, 1, 7, 10, 22, and 25.)

[2] J. P. Maxfield and H. C. Harrison, "Methods of High Quality Recoding and Reproduction of Music and Speach based on Telephone Research," *Bell System Technical Journal*, vol. 5, pp. 493–523, Jul 1926. (Cited on pages xi and 2.)

[3] A. Cummings, "Sound transmission in curved duct bends," *J. Sound Vibr.*, vol. 35, no. 4, pp. 451–477, 1974. (Cited on pages xi, 4, 14, 15, and 18.)

[4] A. G. Webster, "Acoustical Impedance and the Theory of Horns and of the Phonograph," *Proc. Nat. Ac. Sci.*, vol. 5, no. 7, pp. 275–282, Jul 1919. (Cited on page 1.)

[5] A. H. Benade and E. V. Jansson, "On Plane and Spherical Waves in Horns With Non-Uniform Flare Part 1," *Acoustica*, vol. 31, pp. 79–98, 1974. (Cited on page 1.)

[6] R. J. Alfredson, "The Propagation of Sound in a Circular Duct of Continuously Varying Cross-Sectional Area," *J. Sound Vibr.*, vol. 23, no. 4, pp. 433–442, 1972. (Cited on page 1.)

[7] V. Pagneux, N. Amir, and J. Kergomard, "A study of wave propagation in varying cross-section waveguides by modal decomposition. Part I. Theory and validation," *J. Acoust. Soc. Am.*, vol. 100, no. 4, pp. 2034–2048, Oct 1996. (Cited on pages 1 and 7.)

[8] T. Shindo, T. Yoshioka, and K. Fukuyama, "Calculation of Sound Radiation from an Unbaffled, Rectangular-Cross-Section Horn Loudspeaker Using Combined Analytical and Boundary-Element Methods," *J. Audio Eng. Soc.*, vol. 38, no. 5, pp. 340–349, May 1990. (Cited on page 1.)

[9] A. Schuhmacher and K. B. Rasmussen, "Modelling of horn-type loudspeakers for outdoor sound reinforcement systems," *Applied Acoustics*, vol. 56, pp. 25–37, 1999. (Cited on page 1.)

[10] S. Kirkup and A. Thompson, "Computing the Acoustic Field of a Radiating Cavity by the Boundary Element - Rayleigh Integral Method (BERIM)," *Proceedings of the*

World Congress on Engineering, vol. 2, 2007. [Online]. Available: http://www.kirkup.info/papers/SKAT07.pdf (Cited on pages 1 and 38.)

[11] B. Kolbrek, "Simulation of Sound Propagation in an Axisymmetric Duct," Norwegian University of Science and Technology, (Unpublished), 2011. [Online]. Available: http://kolbrek.hoyttalerdesign.no/images/misc/kolbrek_simulation_of_sound_propagation_in_an_axisymmetric_duct.pdf (Cited on pages 2, 7, 31, 39, and 88.)

[12] ——, "Modal Propagation in Acoustic Horns," Norwegian University of Science and Technology, (Unpublished), 2012. [Online]. Available: http://kolbrek.hoyttalerdesign.no/images/misc/evaluation_mpm_2012_bk.pdf (Cited on pages 2, 7, 31, 39, and 88.)

[13] P. Wilson and G. W. Webb, *Modern Gramophones and Electrical Reproducers*. Cassell and Co., London, 1929. (Cited on page 3.)

[14] P. Wilson, "Tractrix Horns," *The Gramophone*, pp. 119–120, Aug 1935. (Cited on page 3.)

[15] P. G. A. H. Voigt, "Improvements in Horns for Acoustic Instruments," British Patent 278 098, Jul. 5, 1927. (Cited on page 3.)

[16] S. Williams, "Recent developments in the recording and reproduction of sound," *Journal of the Franklin Institute*, vol. 202, no. 4, pp. 413 – 448, 1926. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0016003226906534 (Cited on page 3.)

[17] S. Tanaka, "On the Acoustic Folded Horns," *Science Reports of the Research Institutes, Tohoku Unviversity, Ser. A. Physics, chemistry and metallurgy*, no. 1, pp. 243–248, 1949. (Cited on page 3.)

[18] R. W. Carlisle, "Method of Improving Acoustic Transmission in Folded Horns," *J. Acoust. Soc. Am.*, vol. 31, no. 8, pp. 1135–1137, Aug 1959. (Cited on page 3.)

[19] B. C. Edgar, "The Monolith Horn," *Speaker Builder*, no. 6, pp. 12–14, 16, 18, 24–25, 1993. (Cited on page 4.)

[20] A. Cummings, "Sound Transmission in 180 Degree Duct Bends of Rectangular Section," *J. Sound Vibr.*, vol. 41, no. 3, pp. 321–334, 1975. (Cited on page 4.)

[21] K. F. F. J. Bright, Andrew; Holland, "Analysis of a Folded Acoustic Horn," *J. Audio Eng. Soc.*, vol. 52, no. 10, pp. 1029–1042, 2004. [Online]. Available: http://www.aes.org/e-lib/browse.cfm?elib=13023 (Cited on page 4.)

[22] K. R. Holland, F. J. Fahy, and C. L. Morfey, "Prediction and Measurement of the One-Parameter Behavior of Horns," *J. Audio Eng. Soc.*, vol. 37, no. 5, pp. 315–337, May 1991. (Cited on page 4.)

[23] J. Bakcman, "Improvements of one-dimensional loudspeaker models," *123rd Convention of the Audio Engineering Society*, Oct 2007, convention Paper 7253. (Cited on page 4.)

[24] C. J. Nederveen, *Acoustical aspects of woodwind instruments*. Fritz Knuf, Amsterdam, 1969. (Cited on page 4.)

[25] D. H. Keefe and A. H. Benade, "Wave Propagation in Strongly Curved Ducts," *J. Acoust. Soc. Am.*, vol. 74, no. 1, pp. 320–332, Jul 1983. (Cited on page 4.)

[26] L. Ting and M. J. Miksis, "Wave propagation through a slender curved tube," *J. Acoust. Soc. Am.*, vol. 74, no. 2, pp. 631–639, Aug 1983. (Cited on page 4.)

[27] S. Félix and V. Pagneux, "Sound propagation in rigid bends: A multimodal approach," *J. Acoust. Soc. Am.*, vol. 110, no. 3, pp. 1329–1337, Sep 2001. (Cited on pages 4, 20, 66, and 67.)

[28] S. Felix and V. Pagneux, "Multimodal analysis of acoustic propagation in three-dimensional bends," *Wave Motion*, vol. 36, pp. 157–168, 2002. (Cited on pages 4 and 20.)

[29] Alistair C.P. Braden, "Bore Optimisation and Impedance Modelling of Brass Musical Instruments," Ph.D. dissertation, University of Edinburgh, 2006. (Cited on pages 4 and 22.)

[30] F. E. Grigor'yan, "Theory of sound wave propagation in curvilinear waveguides," *Soviet Physics – Acustics*, vol. 14, pp. 315–321, 1969. (Cited on page 4.)

[31] W. Osborne, "Calculation of the angular propagation constant for a bend," *J. Sound Vibr.*, vol. 37, no. 1, pp. 65–77, 1974. (Cited on pages 4 and 34.)

[32] ——, "Errata–Calculation of the angular propagation constant for a bend Journal of Sound and Vibration," *J. Sound Vibr.*, vol. 38, no. 4, p. 497, 1975. (Cited on page 4.)

[33] ——, "Higher mode propagation of sound in short curved bends of rectangular cross-section," *J. Sound Vibr.*, vol. 45, no. 1, pp. 39–52, 1976. (Cited on pages 4, 18, and 19.)

[34] A. Cummings, "Sound Transmission in a Folded Annular Duct," *J. Sound Vibr.*, vol. 41, no. 3, pp. 375–379, 1975. (Cited on page 4.)

[35] G. D. Furnell and D. A. Bies, "Matrix analysis of acoustic wave propagation within curved ducting systems," *Journal of Sound and Vibration*, vol. 132, pp. 245–263, 1986. (Cited on page 4.)

[36] ——, "Characteristics of modal wave propagation within longitudinally curved acoustic waveguides," *Journal of Sound and Vibration*, vol. 130, pp. 405–423, 1989. (Cited on page 4.)

[37] D. Firth and F. J. Fahy, "Acoustic characteristics of circular Bends in Pipes," *Journal of Sound and Vibration*, vol. 97, pp. 287–303, 1984. (Cited on page 4.)

[38] A. S. Sarigül, "Sound attenuation characteristics of right-angle pipe bends," *Journal of Sound and Vibration*, vol. 228, pp. 837–844, 1999. (Cited on page 4.)

[39] Jun-Tai Kim and Jeong-Guon Ih, "Transfer matrix of curved duct bends and sound attenuation in curved expansion chambers," *Applied Acoustics*, vol. 56, pp. 297–309, 1999. (Cited on page 4.)

[40] C. K. W. Tam, "A study of sound transmission in curved duct bends by the Galerkin method," *Journal of Sound and Vibration*, vol. 45, pp. 91–104, 1976. (Cited on page 4.)

[41] G. D. Furnell, "Corrections to "A study of sound transmission in curved duct bends by the Galerkin method"," *Journal of Sound and Vibration*, vol. 130, pp. 329–332, 1989. (Cited on page 4.)

[42] A. Cabelli, "The acoustic characteristics of duct bends," *J. Sound Vibr.*, vol. 68, no. 3, pp. 369–388, 1980. (Cited on page 4.)

[43] A. Cabelli and I. C. Shepherd, "Duct acoustics - A numerical technique for the higher order mode solution of three-dimensional problems with rigid walls and no flow," *Journal of Sound and Vibration*, vol. 92, pp. 419–426, 1984. (Cited on page 4.)

[44] S.-H. Ko, "Three-dimensional acoustic waves propagating in acoustically lined cylindrically curved ducts without fluid flow," *Journal of Sound and Vibration*, vol. 66, pp. 165–179, 1979. (Cited on page 5.)

[45] S. Felix and V. Pagneux, "Sound attenuation in lined bends," *J. Acoust. Soc. Am.*, vol. 116, pp. 1921–1931, 2004. (Cited on pages 5 and 20.)

[46] Edén Sorolla, Juan R. Mosig, and Michael Mattes, "Algorithm to calculate a large number of roots of the cross-product of Bessel functions," *Journal of LaTeX class files*, vol. 6, pp. 1–10, 2012. (Cited on pages 5, 17, and 34.)

[47] Martin Horvat and Tomaz Prosen, "The bends on a quantum waveguide and cross-products of Bessel functions," *Journal of Physics A: Mathematical and Theoretical*, vol. 40, pp. 6349–6379, 2007. (Cited on pages 5 and 17.)

[48] E. R. Geddes, "Acoustic Waveguide Theory," *J. Audio Eng. Soc.*, vol. 37, no. 7/8, pp. 554–569, Jul/Aug 1989. (Cited on page 7.)

[49] G. R. Putland, "Every One-Parameter Acoustic Field Obeys Webster's Horn Equation," *J. Audio Eng. Soc.*, vol. 41, no. 6, pp. 435–451, Jun 1993. (Cited on page 7.)

[50] E. Eisner, "Complete Solutions of the Webster Horn Equation," *J. Acoust. Soc. Am.*, vol. 41, no. 4 (2), pp. 1126–1146, 1966. (Cited on page 7.)

[51] E. Geddes and L. Lee, *Audio Transducers*. Hong Kong, 2002, iSBN 0-9722085-0-X. (Cited on pages 7 and 28.)

[52] P. M. Morse and U. Ingard, *Theoretical Acoustics*. McGraw-Hill, 1986. (Cited on page 7.)

[53] A. D. Pierce, *Acoustics*. Acoustical Society of America, 1994. (Cited on page 7.)

[54] James Alan Cochran, "Remarks on the Zeros of Cross-product Bessel Functions," *J. Soc. Indust. Appl. Math.*, vol. 12, no. 3, pp. 580–578, Sept 1964. (Cited on pages 17 and 35.)

[55] ——, "The Asymptotic Nature of Zeros of Cross-product Bessel Functions," *Quart. Journ. Mech. and Applied Math.*, vol. 19, pp. 511–522, 1966. (Cited on page 17.)

[56] ——, "The analyticity of cross-product Bessel function zeros," *Proc. Camb. Phil. Soc.*, vol. 62, pp. 215–226, 1966. (Cited on page 17.)

[57] C. J. Chapman, "The asymptotic theory of dispersion relations containing Bessel functions of imaginary order," *Proc. R. Soc. A*, vol. 468, pp. 4008–4023, 2012. (Cited on page 17.)

[58] Maple, *version 15.00*. Maplesoft, 2011. [Online]. Available: http://www.maplesoft.com/ (Cited on page 20.)

[59] J. A. Kemp, "Multimodal radiation impedance of a rectangular duct terminated in an infinite baffle," *Acta Acustica united with Acustica*, vol. 87, no. 1, pp. 11–15, Jan/Feb 2001. (Cited on page 24.)

[60] L. E. Kinsler, A. R. Frey, Alan B. Coppens, and James V. Sanders, *Fundamentals of Acoustics*, 4th ed. John Wiley & Sons, Inc., 2004. (Cited on page 28.)

[61] E. R. Geddes, "Sound Radiation from Acoustic Apertures," *J. Audio Eng. Soc.*, vol. 41, no. 4, pp. 214–230, Apr 1993. (Cited on page 28.)

[62] MATLAB, *version 7.13.0 (R2011b)*. Natick, Massachusetts: The MathWorks Inc., 2011. [Online]. Available: http://www.mathworks.se/ (Cited on page 31.)

[63] Andreas Asheim, "Applying the numerical method of steepest descent on multivariate oscillatory integrals in scattering theory," 2013. [Online]. Available: http://arxiv.org/abs/1302.1019v1 (Cited on page 31.)

[64] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery, *Numerical Recipes in C*, 2nd ed. Cambridge University Press, 1992. (Cited on pages 35 and 36.)

[65] D. E. Amos, "Algorithm 644: A Portable Package for Bessel Functions of a Complex Argument and Nonnegative Order," *ACM Transactions on Mathematical Software*, vol. 12, no. 3, pp. 265–273, Sept 1986. (Cited on page 36.)

[66] Shanjie Zhang and Jianming Jin, *Computation of Special Functions*. John Wiley & Sons, Inc., 1996. (Cited on pages 36 and 85.)

[67] Samuel P. Morgan, *Tables of Bessel functions of imaginary order and imaginary argument*. California Institute of Technology, Pasadena, 1947. (Cited on pages 36 and 86.)

[68] T. M. Dunster, "Bessel functions of purely imaginary order, with an application ot second-order linear differential equations having a large parameter," *SIAM J. Math. Anal.*, vol. 21, pp. 995–1018, 1990. (Cited on page 36.)

[69] A. A. Matyshev and E. Fohtung, "On the Computation and Applications of Bessel Functions with Pure Imaginary Indices," *(To be Submitted to Journal of Computational and Applied Mathematics)*, 2009. [Online]. Available: http://arxiv.org/abs/0910.0365v1 (Cited on page 36.)

[70] Paul Godfrey. (2011) gamma: Gamma function valid in the entire complex plane. (Renamed cgamma in to not interfere with the built-in gamma function). [Online]. Available: http://www.mathworks.se/matlabcentral/fileexchange/3572-gamma (Cited on page 37.)

[71] GiD - The Personal Pre and Post Processor. Accessed 14.12.12. [Online]. Available: http://gid.cimne.upc.es/ (Cited on page 37.)

[72] Jonathan Richard Shewchuk. Triangle, A Two-Dimensional Quality Mesh Generator and Delaunay Triangulator. Accessed 14.12.12. [Online]. Available: https://www.cs.cmu.edu/~quake/triangle.html (Cited on page 37.)

[73] A. Weidauer. Triangulation API for Delphi under Windows NT. Accessed 14.12.12. [Online]. Available: http://www.triplexware.huckfinn.de/triapi_eng.html (Cited on page 37.)

[74] Stephen M. Kirkup, Ambrose Thompson, Bjørn Kolbrek, and J. Yazdani, "Simulation of the Acoustic Field of a Horn Loudspeaker By the Boundary Element-Rayleigh Integral Method," *Journal of Computational Acoustics*, vol. 21, 2013. [Online]. Available: http://www.worldscientific.com/doi/abs/10.1142/S0218396X12500208 (Cited on pages 38 and 39.)

[75] V. Salmon, "A New Family of Horns," *J. Acoust. Soc. Am.*, vol. 17, no. 3, pp. 212–218, Jan 1946. (Cited on page 50.)

[76] G. N. Watson, *A treatise on the theory of Bessel functions*. Cambridge University Press, 1962. (Cited on page 85.)

APPENDIX

A

F-MATRIX FOR AN ASYMMETRIC RECTANGULAR DUCT

The derivation of the F-matrix for asymmetric rectangular ducts will be given here.

A.1 THE CASE $n_x = 0,\ m_x > 0$

For a general duct discontinuity, we have the width of the first duct going from a_{1-} to a_{1+}, the second duct goes from a_{2-} to a_{2+}. The eigenfunctions will be

$$\phi_{n_x} = \begin{cases} 1 & : n_x = 0 \\ \sqrt{2}\cos\left(\frac{n_x\pi(x-a_-)}{a_+-a_-}\right) & : n_x > 0 \end{cases} \tag{115}$$

We get for the X matrix, $n_x = 0,\ m_x > 0$:

$$X_{n_x m_x} = \frac{1}{a_{1+}-a_{1-}} \int_{a_{1-}}^{a_{1+}} \sqrt{2}\cos\left(\frac{m_x\pi(x-a_{2-})}{a_{2+}-a_{2-}}\right) dx \tag{116}$$

$$= \frac{\sqrt{2}}{a_{1+}-a_{1-}} \left[\frac{a_{2+}-a_{2-}}{m_x\pi}\sin\left(\frac{m_x\pi(x-a_{2-})}{a_{2+}-a_{2-}}\right)\right]_{a_{1-}}^{a_{1+}} \tag{117}$$

$$= \frac{a_{2+}-a_{2-}}{a_{1+}-a_{1-}} \times \frac{\sqrt{2}}{m_x\pi}\left[\sin\left(m_x\pi\frac{a_{1+}-a_{2-}}{a_{2+}-a_{2-}}\right) - \sin\left(m_x\pi\frac{a_{1-}-a_{2-}}{a_{2+}-a_{2-}}\right)\right] \tag{118}$$

Using the identity $\sin A - \sin B = 2\sin\frac{1}{2}(A-B)\cos\frac{1}{2}(A+B)$ we get

$$\frac{1}{2}(A-B) = \frac{\pi m_x}{2}\cdot\frac{a_{1+}-a_{1-}}{a_{2+}-a_{2-}}$$

$$\frac{1}{2}(A+B) = \frac{\pi m_x}{2}\cdot\left(\frac{a_{1+}+a_{1-}-2a_{2-}}{a_{2+}-a_{2-}}\right)$$

We then define

$$\beta_{x,t} = \frac{\alpha_1}{\alpha_2} = \frac{a_{1+}-a_{1-}}{a_{2+}-a_{2-}} \tag{119}$$

$$\beta_{x,a} = \frac{a_{1+}+a_{1-}-2a_{2-}}{\alpha_2} \tag{120}$$

so we can write $X_{n_x m_x}$ as

$$X_{n_x m_x} = \sqrt{2}\,\mathrm{sinc}\left(\frac{m_x\pi}{2}\beta_{x,t}\right)\cos\left(\frac{m_x\pi}{2}\beta_{x,a}\right) \tag{121}$$

A.2 THE CASE $n_x > 0,\ m_x > 0$

Now the integral becomes

$$X_{n_x m_x} = \frac{1}{a_{1+} - a_{1-}} \int_{a_{1-}}^{a_{1+}} 2\cos\left(\frac{n_x \pi (x - a_{1-})}{a_{1+} - a_{1-}}\right) \cos\left(\frac{m_x \pi (x - a_{2-})}{a_{2+} - a_{2-}}\right) dx \tag{122}$$

which can be transformed to

$$\begin{aligned} X_{n_x m_x} \quad = \quad & \frac{1}{a_{1+} - a_{1-}} \int_{a_{1-}}^{a_{1+}} \left\{ \cos\left(\frac{n_x \pi (x - a_{1-})}{a_{1+} - a_{1-}} + \frac{m_x \pi (x - a_{2-})}{a_{2+} - a_{2-}}\right) \right. \\ & \left. + \cos\left(\frac{n_x \pi (x - a_{1-})}{a_{1+} - a_{1-}} - \frac{m_x \pi (x - a_{2-})}{a_{2+} - a_{2-}}\right) \right\} dx \end{aligned} \tag{123}$$

In the following, let

$$\alpha_1 = a_{1+} - a_{1-}$$

$$\alpha_2 = a_{2+} - a_{2-}$$

$$\begin{aligned} X_{n_x m_x} \quad = \quad & \frac{\alpha_2}{\pi} \left[(n_x \alpha_2 - m_x \alpha_1)^{-1} \right. \\ & \times \sin\left(\frac{\pi}{\alpha_1 \alpha_2} \left((n_x \alpha_2 - m_x \alpha_1) x - (n_x a_{1-} \alpha_2 - m_x a_{2-} \alpha_1)\right)\right) \\ & + (n_x \alpha_2 + m_x \alpha_1)^{-1} \\ & \left. \times \sin\left(\frac{\pi}{\alpha_1 \alpha_2} \left((n_x \alpha_2 - m_x \alpha_1) x - (n_x a_{1-} \alpha_2 + m_x a_{2-} \alpha_1)\right)\right) \right]_{a_{1-}}^{a_{1+}} \end{aligned} \tag{124}$$

Inserting the limits and ordering by denominator:

$$\begin{aligned} X_{n_x m_x} = {} & \frac{\alpha_2}{\pi (n_x \alpha_2 - m_x \alpha_1)} \left[\sin\left(\frac{\pi}{\alpha_1 \alpha_2} \left((n_x \alpha_2 - m_x \alpha_1) a_{1+} - (n_x a_{1-} \alpha_2 - m_x a_{2-} \alpha_1)\right)\right) \right. \\ & \left. - \sin\left(\frac{\pi}{\alpha_1 \alpha_2} \left((n_x \alpha_2 - m_x \alpha_1) a_{1-} - (n_x a_{1-} \alpha_2 - m_x a_{2-} \alpha_1)\right)\right) \right] \\ & + \frac{\alpha_2}{\pi (n_x \alpha_2 + m_x \alpha_1)} \left[\sin\left(\frac{\pi}{\alpha_1 \alpha_2} \left((n_x \alpha_2 - m_x \alpha_1) a_{1+} - (n_x a_{1-} \alpha_2 + m_x a_{2-} \alpha_1)\right)\right) \right. \\ & \left. - \sin\left(\frac{\pi}{\alpha_1 \alpha_2} \left((n_x \alpha_2 - m_x \alpha_1) a_{1-} - (n_x a_{1-} \alpha_2 + m_x a_{2-} \alpha_1)\right)\right) \right] \end{aligned} \tag{125}$$

Again using $\sin A - \sin B = 2 \sin \frac{1}{2}(A - B) \cos \frac{1}{2}(A + B)$ with

$$A = \frac{1}{\alpha_1 \alpha_2} \left((n_x \alpha_2 - m_x \alpha_1) a_{1+} - (n_x a_{1-} \alpha_2 - m_x a_{2-} \alpha_1)\right)$$

$$B = \frac{1}{\alpha_1 \alpha_2} \left((n_x \alpha_2 - m_x \alpha_1) a_{1-} - (n_x a_{1-} \alpha_2 - m_x a_{2-} \alpha_1)\right)$$

for the first pair gives

$$A - B = n - m\frac{\alpha_1}{\alpha_2}$$

$$A + B = n - m\frac{a_{1+} + a_{1-} - 2a_{2-}}{\alpha_2}$$

For the second pair,

$$A = \frac{1}{\alpha_1\alpha_2}\left((n_x\alpha_2 - m_x\alpha_1)a_{1+} - (n_x a_{1-}\alpha_2 + m_x a_{2-}\alpha_1)\right.$$

$$B = \frac{1}{\alpha_1\alpha_2}\left((n_x\alpha_2 - m_x\alpha_1)a_{1-} - (n_x a_{1-}\alpha_2 + m_x a_{2-}\alpha_1)\right.$$

gives

$$A - B = n + m\frac{\alpha_1}{\alpha_2}$$

$$A + B = n + m\frac{a_{1+} + a_{1-} - 2a_{2-}}{\alpha_2}$$

As before,

$$\beta_{x,t} = \frac{\alpha_1}{\alpha_2} = \frac{a_{1+} - a_{1-}}{a_{2+} - a_{2-}} \tag{126}$$

$$\beta_{x,a} = \frac{a_{1+} + a_{1-} - 2a_{2-}}{\alpha_2} \tag{127}$$

Combining these factors, gives

$$\begin{aligned} X_{n_x m_x} = &\operatorname{sinc}\left(\frac{\pi}{2}(n - m\beta_{x,t})\right)\cos\left(\frac{\pi}{2}(n - m\beta_{x,a})\right) \\ &+ \operatorname{sinc}\left(\frac{\pi}{2}(n + m\beta_{x,t})\right)\cos\left(\frac{\pi}{2}(n + m\beta_{x,a})\right) \end{aligned} \tag{128}$$

This expression has also been checked numerically against direct numerical integration of (122).

B

BESSEL FUNCTIONS

When the wave equation is solved in cylindrical coordinates by separation of variables, the solution in the radial and angular directions is described by a differential equation that can be written in the form of Bessel's differential equation [76, 66]

$$z^2\frac{d^2W}{dz^2} + z\frac{dW}{dz} + \left(z^2 + \nu^2\right)W = 0 \tag{129}$$

where z is complex, and ν is a constant. If ν is not an integer, this equation has the two linearly independent solutions

$$J_\nu = \sum_{m=0}^{\infty}(-1)^m \frac{\left(\frac{z}{2}\right)^{2m+\nu}}{m!\Gamma(m+\nu+1)} \tag{130}$$

and

$$J_{-\nu} = \sum_{m=0}^{\infty}(-1)^m \frac{\left(\frac{z}{2}\right)^{2m-\nu}}{m!\Gamma(m-\nu+1)} \tag{131}$$

Another linearly independent solution, which must be used if $\nu = n$ is an integer (in which case $J_{-n}(z) = (-1)^n J_n(z)$), is

$$Y_\nu = \frac{J_\nu(z)\cos\nu\pi - J_{-\nu}(z)}{\sin\nu\pi} \tag{132}$$

$J_\nu(z)$ is called the Bessel function of the first kind, while $Y_\nu(z)$ is called the Bessel function of the second kind, sometimes also referred to as the Neumann function $N_\nu(z)$. The solution to Eq. (129) can be either a linear combination of $J_\nu(z)$ and $J_{-\nu}(z)$, or of $J_\nu(z)$ and $Y_\nu(z)$.

A set of combinations of J_ν and Y_ν that are useful for wave functions, are the Hankel functions of the first and second kind:

$$H_\nu^{(1)}(z) = J_\nu(z) + iY_\nu(z) \tag{133}$$

$$H_\nu^{(2)}(z) = J_\nu(z) - iY_\nu(z) \tag{134}$$

These functions must be used in the case where the boundary conditions in the duct are not purely Neumann or Dirichlet.

If we let $z \to iz$, Eq. (129) becomes

$$z^2\frac{d^2W}{dz^2} + z\frac{dW}{dz} - \left(z^2 + \nu^2\right)W = 0 \tag{135}$$

which is known as the modified Bessel's equation, and its solutions are the modified Bessel functions $I_\nu(z)$, $I_{-\nu}(z)$ and $K_\nu(z)$, and a complete solution is formed as in the previous case as a linear combination of either $I_\nu(z)$ and $I_{-\nu}(z)$, or $I_\nu(z)$ and $K_\nu(z)$.

In the case when both z and ν are purely imaginary, Eq. (129) becomes [67]

$$z^2 \frac{d^2W}{dz^2} + z\frac{dW}{dz} - \left(z^2 - \nu^2\right) W = 0 \tag{136}$$

Regardless of order and argument, Eq. (130) is valid, and can be used as a reference for testing other approximations.

C

MPM TOOLBOX MATLAB CODE

Note: the functions `PlaceFigures` and `besselzero` can be downloaded from the Matlab Central.

Only the first listing has the full file header. For subsequent listings, the licence text and the function name at the start of the header has been hidden, to save space.

Functions that are common to the axisymmetric code are not listed, but can be found in [11, 12].

Functions specific to the rectangular horn code start with MPM_REC (symmetric) or MPM_RECA (asymmetric).

C.1 MAIN FUNCTIONS

Complete code for simulating a rectangular horn. Includes comparison with BEM results.

Listing 1: RectHornCalc.m

```
%RectHornCalc
%
%    This file contains sample code showing how the MPM toolbox can be used
%  to calculate the performance of a rectangular horn exited by a plane wave
%  at the throat.
%    The horn contour is imported from a text file containing the dimensions,
%  and cast into the form of a stepped duct. Next, the modal radiation
%  impedance at the mouth is interpolated from precalculated values.
%  Following this, the throat impedance, and the modal impedances throughout
%  the horn are calculated. This data is then used in the propagation of
%  velocity from throat to mouth. The mouth velocity is used to compute the
%  resulting pressure in the free field, using the Rayleigh integral.
%
%  See Hints.m for hints on usage.
%
% ------------------------------------------------------------ ...
    ------------------------------------------------
%    This file is part of the Modal Propagation Method (MPM) Toolbox by Bjørn ...
    Kolbrek.
%
%    The MPM Toolbox is free software: you can redistribute it and/or modify
%    it under the terms of the GNU General Public License as published by the ...
    Free Software
%    Foundation, either version 3 of the License, or (at your option) any ...
    later version.
%
%    The MPM Toolbox is distributed in the hope that it will be useful,
%    but WITHOUT ANY WARRANTY; without even the implied warranty of ...
    MERCHANTABILITY or FITNESS
%    FOR A PARTICULAR PURPOSE. See the GNU General Public License for more ...
    details.
%
```

```
27 %    You should have received a copy of the GNU General Public License along ...
       with the
28 %    MPM Toolbox. If not, see <http://www.gnu.org/licenses/>.
29 %  ------------------------------------------------------------ ...
       ---------------------------------------------
30 %
31 %  see also Hints, MPM_Horndemo2, MPM_AShorncoord, MPM_refinecoords, ...
       MPM_makesteps,
32 %  MPM_ASbaffledradzmatrix, MPM_getimpedances, MPM_makebigfmat,
33 %  MPM_getmouthvolvel, MPM_getfieldpointpressures
34 %  % Test of rectangular horn simulation
35
36 testno = 1;
37
38 N = 2; %maximum number of modes in each direction
39
40 if testno == 2
41     % text file containing horn dimensions
42     filename = 'horns\RectAsymTesthorn1.txt';
43     t = 'Asymmetric';
44 else
45     % text file containing horn dimensions
46     filename = 'horns\QsymHorn1.txt';
47     t = 'Quarter symmetric';
48     testno = 1;
49 end;
50
51 % simulation parameters
52 fmin = 200; %lower frequency
53 fmax = 10000; % upper frequency
54 Nf = 150; % number of frequencies
55 %N = 2; %maximum number of modes in each direction
56 c = 344; %sound speed
57 rho = 1.205;% air density
58
59 % read coordinates from text file
60 [horncoords, Nsteps, geomtype] = MPM_readHornDims(filename);
61 % make steps
62 horncoords = MPM_makesteps(horncoords);
63 % plot profile
64 figure(1);
65 if (geomtype==1)
66     plot(horncoords(:,1),horncoords(:,2),horncoords(:,1),horncoords(:,3));
67     ylim([0,1.1*max(max(horncoords(:,2)), max(horncoords(:,3)))]);
68     legend('Width','Height','location','northwest');
69 elseif geomtype==2
70     plot(horncoords(:,1),horncoords(:,2),'b',horncoords(:,1),horncoords(:,4),...
71         'r',horncoords(:,1),horncoords(:,3),'b',horncoords(:,1),horncoords(:,5),'r');
72     %ylim([0,1.1*max(max(horncoords(:,2)), max(horncoords(:,3)))]);
73     legend('Width', 'Height','location','northwest');
74 end;
```

```
xlabel('z axis [m]');
ylabel('Height or width [m]');
title([t ' Horn profile']);
axis equal;
%%
freq = logspace(log10(fmin), log10(fmax), Nf);
k = 2*pi*freq/c;

t00 = tic;
% calculate radiation impedance matrix
disp(['Calculating radiation impedance']);
%load Zmat.mat;
%Zmat = Zmat(1:N^2,1:N^2,:);
if geomtype==1
    am = horncoords(end,2);
    bm = horncoords(end,3);
    Sm = am*bm*4;
    St = 4*horncoords(1,2)*horncoords(1,3);
    Zmat = MPM_RECbaffledradzmatrixIntp(k, rho, c, am, bm, N);
elseif geomtype==2
    am = horncoords(end,3)-horncoords(end,2);
    bm = horncoords(end,5)-horncoords(end,4);
    Sm = am*bm;
    St = (horncoords(1,3)-horncoords(1,2))*(horncoords(1,5)-horncoords(1,4));
    Zmat = MPM_RECAbaffledradzmatrixIntp(k, rho, c, am, bm, N);
end;
% save('Zmat.mat','Zmat')


toc(t00);
disp(['Calculating F-matrices']);
BigF = MPM_makebigfmat(N, horncoords, 0,geomtype);
toc(t00);
%%
disp(['Calculating horn impedance']);
Z00mat = [];
Ztmat = zeros(N^2,N^2,Nf);
U0 = zeros(N^2,1); %plane wave at throat
U0(1) = 1*St; %unit velocity at throat (not volume velocity)
Umouth = zeros(N^2,Nf);
for ii = 1:Nf
    disp(['Calculating k = ' num2str(k(ii)) ' (' num2str(ii/length(k)*100) ...
        '%)']);
    BigZ = MPM_getimpedances(k(ii), horncoords, 0, squeeze(Zmat(:,:,ii)), ...
        BigF, rho, c, geomtype, false);
    Umouth(:,ii) = MPM_getmouthvolvel(k(ii), horncoords, U0, 0, BigZ, BigF, ...
        rho, c, geomtype);
    Ztmat(:,:,ii) = squeeze(BigZ(:,:,1,:));
end;
ttot = toc(t00);
toc(t00);
```

```
123
124 Z00mat = squeeze(Ztmat(1,1,:));
125 Z00mat = St/(rho*c)*Z00mat;
126
127 clear BigZ; % free memory
128
129
130 %% Field point response
131
132 % Uncomment code below to read results from BEM simulation
133 % usecoords = [17 19];
134 % groupno = 2;
135 % fpdata = readFPdata('horns\RectAsymTesthorn1FPData.txt');
136 % fpcoords = fpdata(groupno).coords(usecoords,:);
137 % fpcoords(:,3) = fpcoords(:,3)-0.6;
138 % pbem = 1.03*fpdata(groupno).data.';
139
140 % create field points
141 angH = linspace(0,90,10)';
142 angV = angH;
143 rFP = 3;
144
145 x = rFP*sin(angH*pi/180);
146 z = rFP*cos(angH*pi/180);
147 y = 0*x;
148
149 fpH = [x y z];
150
151 y = rFP*sin(angV*pi/180);
152 z = rFP*cos(angV*pi/180);
153 x = 0*y;
154
155 fpV = [x y z];
156
157 vm = Umouth/Sm;
158 % pextm = MPM_RECmodalradiatedpressure(k, horncoords(end,2), ...
      horncoords(end,3), vm, fpcoords, rho, c);
159 pextH = MPM_getfieldpointpressures(k, Umouth, fpH, horncoords(end,:), 50, N, ...
      rho, c, geomtype);
160 pextV = MPM_getfieldpointpressures(k, Umouth, fpV, horncoords(end,:), 50, N, ...
      rho, c, geomtype);
161 figure(2);
162 semilogx(freq,  94+20*log10(abs(pextH)), 'k-');%, freq, ...
      94+20*log10(abs(pextm)), 'm-');
163 %semilogx(fpdata(groupno).fvec, 94+20*log10(abs(pbem(usecoords,:))), 'b', ...
      freq,  94+20*log10(abs(pext)), 'k-');%, freq, 94+20*log10(abs(pextm)), 'm-');
164 xlim([fmin fmax]);
165 title('Horizontal Directivity');
166 figure(3);
167 semilogx(freq,  94+20*log10(abs(pextV)), 'b-');%, freq, ...
      94+20*log10(abs(pextm)), 'm-');
```

```
%semilogx(fpdata(groupno).fvec, 94+20*log10(abs(pbem(usecoords,:))), 'b', ...
    freq,  94+20*log10(abs(pext)), 'k-');%, freq, 94+20*log10(abs(pextm)), 'm-');
xlim([fmin fmax]);
title('Vertical Directivity');
%title(fpdata(groupno).title);
% legend('-70','0','70');

pext0 = pextH(1,:);
pext0 = pext0(ones(length(fpH),1),:);
pextn = pextH ./ pext0;

figure(4);
contourf(freq, angH, 94+20*log10(abs(pextn)), 15);
set(gca, 'xscale', 'log');
ylabel('Degrees');
xlabel('Hz');
title('Polar map, Horizontal');
caxis([55 95]);
colorbar;

pext0 = pextV(1,:);
pext0 = pext0(ones(length(fpH),1),:);
pextn = pextV ./ pext0;

figure(5);
contourf(freq, angH, 94+20*log10(abs(pextn)), 15);
set(gca, 'xscale', 'log');
ylabel('Degrees');
xlabel('Hz');
title('Polar map, Vertical');
caxis([55 95]);
colorbar;

%%
figure(6);
semilogx(freq, real(Z00mat), 'k-', freq, imag(Z00mat), 'r-');%, freq, ...
    real(Zsum), freq, imag(Zsum));
grid;
title([t ' Horn throat impedance']);

ylabel(['Normalized acoustic Z(' num2str(0) ',' num2str(0) ')']);
xlabel('Hz');
legend('R_{th}','X_{th}');%,'R_{thsum} (50%)','X_{thsum} (50%)');

%% Comparison with BEM results

% Uncomment code below to read impedance results from BEM simulation
% zdata = readZdata('horns\RectAsymTesthorn1_Zdata.txt');
% zdata.Z = zdata.Z * 1.03;
% figure(7);
```

```
% semilogx(freq, real(Z00mat), 'k-', freq, imag(Z00mat), 'r-', zdata.fvec, ...
    real(zdata.Z), zdata.fvec, imag(zdata.Z));
% grid;
% title('Horn throat impedance - Symmetric horn');
% ylabel(['Normalized acoustic Z(' num2str(0) ',' num2str(0) ')']);
% xlabel('Hz');
% legend('R_{th} MPM','X_{th} MPM','R_{th} BEM','X_{th} BEM');



%% Save the results

% create new file name and save
iiV = find((filename=='\')|(filename=='.'));
matfn = filename(iiV(1)+1:iiV(2)-1);

matfn = [matfn '_' num2str(N) 'M.mat'];

save(matfn);
```

C.2 READ HORN DIMENSIONS

Function for reading the horn profile from a text file of dimensions. This was included to be able to generate horns in HornCAD, instead of implementing all the required horn geometry functions in Matlab.

Listing 2: MPM_readHornDims.m

```
% [horndims, n, geomtype] = MPM_readHornDims(fname)
%
% Reads horn dimensions exported from Horn Simulator.
% The file contains a header that determines the geometry,
% and a list of coordinates.
% The cordinate list is output in horndims.
% The columns are arranged as follows:
% For axisymmetrid horns:
%   col 1 = length along horn (z)
%   col 2 = radius (r)
% For rectangular, quarter symmetric horns:
%   col 1 = length along horn (z)
%   col 2 = width of horn, left (a+)
%   col 3 = width of horn, right (b+)
% For rectangular horns:
%   col 1 = length along horn (z)
%   col 2 = width of horn, left (a-)
%   col 3 = width of horn, right (a+)
%   col 4 = height of horn, down (b-)
```

```
%   col 5 = height of horn, up (b+)
%   (directions refer to the horn viewed from the mouth)
%
% Input parameters:
% fname : file name of the textfile
%
% Output parameters:
% horndims : the matrix containing the horn dimensions
% n : the number of elements
% geomtype : determines the geometry of the horn:
%    0 : axisymmetric (default)
%    1 : quarter symmetric rectangular
%    2 : asymmetric rectangular
%    (no other geometries supported yet)

function [horndims, n, geomtype] = MPM_readHornDims(fname);
% reads horn dimensions exported from HornCAD
fid=fopen(fname, 'rt');   %open the file
if fid==-1
    error('File does not exist');
    return;
end;
horndims = [];
s = fgetl(fid);  %read the first line
s = fgetl(fid);  %read the second line
a = findstr(s, 'Radius');
r = findstr(s, 'Height1');
if length(a)>0
    % circular/axisymmetric horn
    s = fgetl(fid);
    s = fgetl(fid);
    [dims, count] = fscanf(fid, '%e');
    horndims = reshape(dims, 2, count/2)';
    n = count/2;
    geomtype = 0;
elseif length(r)>0
    % rectangular horn
    s = fgetl(fid);
    s = fgetl(fid);
    s = fgetl(fid);
    s = fgetl(fid);
    s = fgetl(fid);
    [dims, count] = fscanf(fid, '%e');
    n = count/5;
    dims = reshape(dims, 5, count/5)';
    dims = dims*1e-2; % file contains dimensions in cm
    % rearrange into a-, a+, b-, b+
    z = dims(:,1);
    bp = dims(:,2);
    bn = -dims(:,3);
    an = -dims(:,4);
```

```
    ap = dims(:,5);
    if (max(abs(an+ap)) < 1e-6) & (max(abs(bn+bp)) < 1e-6)
        geomtype = 1;
        horndims = [z ap bp];
    else
        geomtype = 2;
        horndims = [z an ap bn bp];
    end;
end;
fclose(fid);   %close the file
```

C.3 RADIATION IMPEDANCE

The modal radiation impedance is calculated by numerical integration. For high *ka*-values, the MSD-method is used. The code for this method, reffered to by the function call `quad_rad` is not included in the appendix, in agreement with the author, Andreas Asheim.

Below: function to calculate the modal radiation impedance for rectangular duct, symmetrical modes only.

Listing 3: MPM_RECbaffledradz.m

```
% [Z, t] = MPM_RECbaffledradz(k, rho, c, a, b, n, m, usequadrad)
%
% Calculates the modal radiation impedance for the
% end of a rectangular duct terminated in an infinite baffle
% by numerical interation.
% Calculates ONLY symmetrical modes.
%
% Input parameters:
% k : wavenumber
% rho : density of medium
% c : sound speed in medium
% a, b : half width and half height of the duct
% n, m : mode numbers n(1) = n_x, n(2) = n_y etc.
% usequadrad : (optional) use quad_rad integration algorithm by A. Asheim
%
% Output parameters:
% Z : modal radiation impedance
% t : calculation time
function [Z, t] = MPM_RECbaffledradz(k, rho, c, a, b, n, m, usequadrad)
%Zmat = zeros(maxmodes, maxmodes, length(k));
if nargin < 8
    usequadrad = true;
end;
S = 4*a*b;
nx = n(1);
ny = n(2);
```

```
mx = m(1);
my = m(2);

Z = k;
t = 0*k;
f = ffunc(nx,ny,mx,my);
Z3 = (-k.*a.*log(2*k*a)+1.5*k*a).*f;

mode = 0;
testcnt = 0;

n1 = 10;
n2 = 15;

if usequadrad
    t01 = tic;
    Z = conj(-1i*1*k(end)/(2*pi*S).*quad_rad(k(end), ...
        @(x,y)ffunction(x,y,n,m,a,b),2*a,2*b,n1,n2));
    tref = toc(t01);
end;

for ii=1:length(k)
    t01 = tic;
    kk = k(ii);
    if mode==0
        Z(ii) = 1i*rho*c/(2*pi*S)*(Z3(ii) + BaffleradzI1I2(kk, a, b, ...
            nx,ny,mx,my, f));
        t(ii) = toc(t01);
        if (ii>2)&&(kk*sqrt(a*b)>2)&&usequadrad
            if t(ii) > 1.5*tref%t(ii-2) % is the calculation time increasing ...
                fast?
                mode = 1;
                testcnt = 0;
            end
        end;
    elseif mode==1
        if testcnt == 0 %time for testing?
            % checking the accuracy of quad_rad
            Z(ii) = 1i*rho*c/(2*pi*S)*(Z3(ii) + BaffleradzI1I2(kk, a, b, ...
                nx,ny,mx,my, f));
            Zt = conj(-1i*kk*c*rho/(2*pi*S^2).*quad_rad(kk, ...
                @(x,y)ffunction(x,y,n,m,a,b),2*a,2*b,n1,n2));
            error = abs((Zt-Z(ii))/Z(ii));
            if error < 0.1
                n1 = 10;
                n2 = 25;
            end;
            %disp(sprintf('checking... k = %.2f, e = %e', kk,error));
            if error < 1e-3
                mode = 2; % acceptable accuracy, continue with quadrad
```

```
72                  disp(sprintf('transfering to quad_rad at k = %.2f, mode ...
                        (%d,%d),(%d,%d)', kk,nx,ny,mx,my));
73              else
74                  testcnt = 0;%floor(max(min(log10(error),0),5));% not accurate ...
                        enough, wait a few frequencies and try again0
75              end;
76          else
77              testcnt = testcnt-1; %wait...
78              Z(ii) = 1i*rho*c/(2*pi*S)*(Z3(ii) + BaffleradzI1I2(kk, a, b, ...
                    nx,ny,mx,my, f));
79          end;
80
81      elseif mode==2
82  %         Zt = 1i*rho*c/(2*pi*S)*(Z3(ii) + BaffleradzI1I2(kk, a, b, ...
        nx,ny,mx,my, f));
83          Z(ii) = conj(-1i*kk*c*rho/(2*pi*S^2).*quad_rad(kk, ...
                @(x,y)ffunction(x,y,n,m,a,b),2*a,2*b,n1,n2));
84  %         error = abs((Zt-Z(ii))/Z(ii));
85  %         disp(sprintf('checking mode 2... k = %.2f, e = %e', kk,error));
86      end;
87      t(ii) = toc(t01);
88  end;
89
90
91  function Z = BaffleradzI1I2(k, a, b, nx,ny,mx,my, f)
92      eps = 10.^(-log(2.5*k*min(a,b))-1);
93      eps = max(min(0.01,eps),1e-6);
94      k2a = 2*k*a;
95      k2b = 2*k*b;
96      %disp(sprintf('Symmetric radiation Z for mode (%d,%d),(%d,%d) at ...
            k=%f\n',nx,ny,mx,my,kk));
97      Z1 = dblquad(@(x,y) Integrand1(x,y,k2a,k2b,nx,ny,mx,my,f), 0,k2a,0,k2b, ...
            eps, @myquad);%
98      Z2 = quadl(@(x) Integrand2(x,k2a,k2b,nx,ny,mx,my,f),0,k2a,eps);
99      Z = Z1+Z2;
100
101 function f = ffunc(nx, ny, mx, my)
102 % sinc(x) = sin(pi*x)/(pi*x), so skip multiplying by pi.
103 f = Nnx(nx)*Nnx(mx)*Nnx(ny)*Nnx(my)*(sinc((nx+mx)) + sinc((nx-mx))) * ...
        (sinc((ny+my)) + sinc((ny-my)));
104
105 function N = Nnx(nx)
106 if (nx==0)
107     N = 1;
108 else
109     N = sqrt(2);
110 end;
111
112 function G = Gfunc(nx, mx, ksi, a, divksi)
113 if nargin < 5
114     divksi = true;
```

```
end;
if (ksi==1)&(divksi)
    ksi = 0.999999999;
end;
if (nx==0)&(mx==0)
    G = 2*(2*a-ksi);
    if divksi
        G = G./(1-ksi);
    end;
else
    % sinc(x) = sin(pi*x)/(pi*x), so skip multiplying by pi.
    sa1 = (nx+mx)*(1-ksi/(2*a));
    ca1 = (nx-mx)*pi*ksi/(2*a);
    sa2 = (nx-mx)*(1-ksi/(2*a));
    ca2 = (nx+mx)*pi*ksi/(2*a);
    G = (2*a-ksi).*( sinc(sa1).*cos(ca1) + sinc(sa2).*cos(ca2));
    if divksi
        G = G ./ (1-ksi);
    end;
    G = Nnx(nx)*Nnx(mx).*G;
end;

function I1 = Integrand1(x,y,ka,kb,nx,ny,mx,my,f)
onem1 = (1-x/ka);
onem2 = (1-y/kb);
rt = sqrt(x.^2+y.^2);
Gf =Gfunc(nx,mx,x/ka,0.5).*Gfunc(ny,my,y/kb,0.5);
%Gf = Gf./((1-x./(ka)).*(1-y./(kb)));
I1 = onem1.*onem2./rt.*(exp(-1i*rt).*Gf-f);

function I2 = Integrand2(x,ka,kb,nx,ny,mx,my,f)
rt = sqrt(x.^2+(kb).^2);
I2 = (1-x./ka).*(log(kb+rt)+ (x-rt)/kb).*f;


function q = myquad(f,a,b,tol,trace,varargin)
q = quadgk(@(x)f(x,varargin{:}),a,b,'RelTol',tol);

function f = ffunction(x,y,n,m,a,b)
nx = n(1);
ny = n(2);
mx = m(1);
my = m(2);
f =Gfunc(nx,mx,x,a, false).*Gfunc(ny,my,y,b, false);
```

Function to calculate the modal radiation impedance for rectangular duct, symmetrical and asymmetrical modes.

Listing 4: MPM_RECAbaffledradz.m

```
% [Z, t] = MPM_RECAbaffledradz(k, rho, c, a, b, n, m, usequadrad)
%
% Calculates the modal radiation impedance for the
% end of a rectangular duct terminated in an infinite baffle
% by numerical interation.
% Calculates both symmetrical and asymmetrical modes.
%
% Input parameters:
% k : wavenumber
% rho : density of medium
% c : sound speed in medium
% a, b : full width and full height of the duct
% n, m : mode numbers n(1) = n_x, n(2) = n_y etc.
% usequadrad : (optional) use quad_rad integration algorithm by A. Asheim
%
% Output parameters:
% Z : modal radiation impedance

function [Z, t] = MPM_RECAbaffledradz(k, rho, c, a, b, n, m, usequadrad)
%Zmat = zeros(maxmodes, maxmodes, length(k));
if nargin < 8
    usequadrad = true;
end;
ka = k*a;
kb = k*b;
S = a*b;
nx = n(1);
ny = n(2);
mx = m(1);
my = m(2);
Z = k;
t = 0*k;

mode = 0;
testcnt = 0;

f = ffunc(nx,ny,mx,my);
Z3 = (-0.75*k.*a.*log(k*a)+0.875*k*a).*f;

n1 = 10;
n2 = 15;

if usequadrad
    t01 = tic;
    Z = conj(-1i*1*k(end)*c*rho/(2*pi*S^2).*quad_rad(k(end), ...
        @(x,y)ffunction(x,y,n,m,a,b),a,b,n1,n2));
    tref = toc(t01);
end;

for ii=1:length(k)
```

```
50      t01 = tic;
51      kk = k(ii);
52      %adaptive accuracy
53      if mode<2
54          eps = 10.^(-log(2.5*kk*min(a,b))-1);
55          eps = max(min(0.01,eps),1e-6);
56          %eps = 1e-6;
57          ka = kk*a;
58          kb = kk*b;
59      end;
60      if mode==0
61          %disp(sprintf('Asymmetric radiation Z for mode (%d,%d),(%d,%d) at ...
                k=%f\n',nx,ny,mx,my,kk));
62  %         Z1 = dblquad(@(x,y) Integrand1(x,y,ka,kb,nx,ny,mx,my,f), 0,ka,0,kb, ...
        eps, @myquad);%
63  %         Z2 = quadl(@(x) Integrand2(x,ka,kb,nx,ny,mx,my,f),0,ka,eps);
64      %     Z2 = quadv(@(x) Integrand2(x,ka,kb,nx,ny,mx,my),0,ka,eps);
65          Z(ii) = 1i*rho*c/(2*pi*S)*(Z3(ii) + AbaffleradzI1I2(kk, a, b, ...
                nx,ny,mx,my, f));
66          t(ii) = toc(t01);
67          if (ii>2)&(kk*sqrt(a*b)>2)&usequadrad
68              if t(ii) > 1.5*tref%t(ii-2) % is the calculation time increasing ...
                    fast?
69                  mode = 1;
70                  testcnt = 0;
71              end
72          end;
73      elseif mode==1
74          if testcnt == 0 %time for testing?
75              % checking the accuracy of quad_rad
76              Z(ii) = 1i*rho*c/(2*pi*S)*(Z3(ii) + AbaffleradzI1I2(kk, a, b, ...
                    nx,ny,mx,my, f));
77              Zt = conj(-1i*1*kk*c*rho/(2*pi*S^2).*quad_rad(kk, ...
                    @(x,y)ffunction(x,y,n,m,a,b),a,b,n1,n2));
78              error = abs((Zt-Z(ii))/Z(ii));
79              if error < 0.1
80                  n1 = 10;
81                  n2 = 25;
82              end;
83              disp(sprintf('checking... k = %.2f, e = %e', kk,error));
84              if error < 1e-3
85                  mode = 2; % acceptable accuracy, continue with quadrad
86                  disp(sprintf('transfering to quad_rad at k = %.2f, mode ...
                        (%d,%d),(%d,%d)', kk,nx,ny,mx,my));
87              else
88                  testcnt = 0;%floor(max(min(log10(error),0),5));% not accurate ...
                        enough, wait a few frequencies and try again0
89              end;
90          else
91              testcnt = testcnt-1; %wait...
```

```
92              Z(ii) = 1i*rho*c/(2*pi*S)*(Z3(ii) + AbaffleradzI1I2(kk, a, b, ...
                    nx,ny,mx,my, f));
93          end;
94
95      elseif mode==2
96          Z(ii) = conj(-1i*1*kk*c*rho/(2*pi*S^2).*quad_rad(kk, ...
                @(x,y)ffunction(x,y,n,m,a,b),a,b,n1,n2));
97      end;
98      t(ii) = toc(t01);
99  end;
100
101 function Z = AbaffleradzI1I2(k, a, b, nx,ny,mx,my, f)
102     eps = 10.^(-log(2.5*k*min(a,b))-1);
103     eps = max(min(0.01,eps),1e-6);
104  %    eps = 1e-6;
105     ka = k*a;
106     kb = k*b;
107     Z1 = dblquad(@(x,y) Integrand1(x,y,ka,kb,nx,ny,mx,my,f), 0,ka,0,kb, eps, ...
            @myquad);%
108     Z2 = quadl(@(x) Integrand2(x,ka,kb,nx,ny,mx,my,f),0,ka,eps);
109     Z = Z1+Z2;
110
111
112 function I1 = Integrand1(x,y,ka,kb,nx,ny,mx,my,f)
113 onem1 = (1-0.5*x/ka);
114 onem2 = (1-0.5*y/kb);
115 rt = sqrt(x.^2+y.^2);
116 Gf =Gfunc(nx,mx,x/ka,1).*Gfunc(ny,my,y/kb,1);
117 Gf = Gf./((1-x./(2*ka)).*(1-y./(2*kb)));
118 I1 = onem1.*onem2./rt.*(exp(-1i*rt).*Gf-f);
119
120 function I2 = Integrand2(x,ka,kb,nx,ny,mx,my,f)
121 rt = sqrt(x.^2+kb.^2);
122 I2 = (1-0.5*x./ka).*(log(kb+rt)+ (x-rt)*0.5/kb).*f;
123
124 function G = Gfunc(nx, mx, ksi, a)
125 if (nx==0)&(mx==0)
126     G = 2*(a-ksi);
127 else
128 % sinc(x) = sin(pi*x)/(pi*x), so skip multiplying by pi.
129     sa1 = (nx+mx)*(1-ksi/(2*a));
130     sa1b = (nx-mx)*ksi/(2*a);
131     ca1 = sa1b*pi;
132     sa2 = (nx-mx)*(1-ksi/(2*a));
133     sa2b = (nx+mx)*ksi/(2*a);
134     ca2 = sa2b*pi;
135
136     G = cos(ca1).*0.5.*((2*a-ksi).*sinc(sa1)-ksi.*sinc(sa2b)) ...
137        + cos(ca2).*0.5.*((2*a-ksi).*sinc(sa2)-ksi.*sinc(sa1b));
138
139     G = Nnx(nx)*Nnx(mx).*G; % ./ (1-0.5*ksi);
```

```
140 end;
141
142 function f = ffunc(nx, ny, mx, my)
143 % sinc(x) = sin(pi*x)/(pi*x), so skip multiplying by pi.
144 f = Nnx(nx)*Nnx(mx)*Nnx(ny)*Nnx(my)*(sinc((nx+mx)) + sinc((nx-mx))) * ...
        (sinc((ny+my)) + sinc((ny-my)));
145
146 function N = Nnx(nx)
147 if (nx==0)
148     N = 1;
149 else
150     N = sqrt(2);
151 end;
152
153 function q = myquad(f,a,b,tol,trace,varargin)
154 q = quadgk(@(x)f(x,varargin{:}),a,b,'RelTol',tol);
155
156 function f = ffunction(x,y,n,m,a,b)
157 nx = n(1);
158 ny = n(2);
159 mx = m(1);
160 my = m(2);
161 f =Gfunc(nx,mx,x,a).*Gfunc(ny,my,y,b);
```

Function to interpolate the modal radiation impedance from a precalculated table, symmetrical and asymmetrical modes.

Listing 5: MPM_RECbaffledradzmatrixIntp.m

```
1  % Zmat = MPM_ASbaffledradzmatrixIntp(k, rho, c, S, maxmodes, bz)
2  %
3  % Calcualtes the modal radiation impedance matrix for the
4  % end of a rectangular tube terminated in an infinite baffle
5  % by interpolation of a lookup table.
6  % The fundamental (plane wave) mode impedance is calcualted
7  % by analytical functions.
8  %
9  % The matrix Zmat is a square, symmetrical matrix.
10 %
11 % Input parameters:
12 % k : wavenumber
13 % rho : density of medium
14 % c : sound speed in medium
15 % a : half width of opening
16 % b : half height of opening
17 % Nmodes : number of modes calculated in each direction
18 %
19 % ---------------------------------------------------- ...
     ------------------------------------------
```

```
20
21 % freq = logspace(log10(1), log10(20000), 200);
22 % k = logspace(log10(0.01), log10(30), 200);% 2*pi*freq/c;
23 %
24 % a = 1;
25 % b = 1;
26 % c = 1;
27 % rho = 1;
28 % Nmodes = 2;
29
30
31 kain = k*a;
32 modeindex = MPM_GetRectModeIndexing(Nmodes);
33 N = size(modeindex,1);
34 ZmatOut = zeros(N, N, length(k));
35
36 if abs(a/b-1)<0.1
37     load ZmatRecSq16x16.mat;
38     L = length(Zmat(:,1,1));
39     if L < N
40         error(sprintf('to many modes! Current: %d, Requested: %d', L, N));
41         return;
42     end;
43     L = length(k);
44     for i=1:N%n
45         for j=1:N%m
46             negexpol = find(kain < min(ka));
47             intpol = find((kain ≥ min(ka)) & (kain ≤ max(ka)));
48             posexpol = find(kain > max(ka));
49             Y = squeeze(real(Zmat(i,j,:)));
50             R1 = interp1(ka', Y, kain(negexpol)','pchip','extrap');
51             R2 = interp1(ka', Y, kain(intpol)','spline','extrap');
52             if (i==j)
53                 R3 = 0.25*ones(size(kain(posexpol)'));
54             else
55                 R3 = zeros(size(kain(posexpol)'));
56             end;
57             %R3 = interp1(ka', Y, kain(posexpol)','linear','extrap');
58             R = [R1; R2; R3];
59             %R = ones(size(kain'));
60             Y = squeeze(imag(Zmat(i,j,:)));
61             X1 = interp1(ka', Y, kain(negexpol)','linear','extrap');
62             X2 = interp1(ka', Y, kain(intpol)','spline','extrap');
63             X3 = interp1(ka', Y, kain(posexpol)','linear','extrap');
64             X = [X1; X2; X3];
65             Zmn = R +1j*X;
66             ZmatOut(i,j,:) = Zmn;
67         end;
68     end;
69 else
70     disp(sprintf('Aspect ratio: %.10f',a/b));
```

```
    error('No tabulated values for this aspect ratio!');
end;

ZmatOut = rho*c/(a*b)*ZmatOut; % the 4 in S (S=4*a*b) is already accounted for

clear Zmat;

% figure(1);
% Z = squeeze(ZmatOut(1,1,:));
% Z2 = squeeze(ZmatOut(2,3,:));
% semilogx(kain, real(Z), 'k', kain, imag(Z),'r', kain, real(Z2), 'b', kain, ...
    imag(Z2),'m');
```

Function to interpolate the modal radiation impedance from a precalculated table, symmetrical and asymmetrical modes.

Listing 6: MPM_RECAbaffledradzmatrixIntp.m

```
% Zmat = MPM_RECAbaffledradzmatrixIntp(k, rho, c, S, maxmodes, bz)
%
% Calcualtes the modal radiation impedance matrix for the
% end of a rectangular tube terminated in an infinite baffle
% by interpolation of a lookup table.
% Asymmetric modes are included.
%
% The matrix Zmat is a square, symmetrical matrix.
%
% Input parameters:
% k : wavenumber
% rho : density of medium
% c : sound speed in medium
% a : full width of opening
% b : full height of opening
% Nmodes : number of modes calculated in each direction
%
% ------------------------------------------------------------ ...
    ----------------------------------------------
%    This file is part of the Modal Propagation Method (MPM) Toolbox by Bjørn ...
    Kolbrek.
% freq = logspace(log10(1), log10(20000), 200);
% k = logspace(log10(0.01), log10(30), 200);% 2*pi*freq/c;
%
% a = 1;
% b = 1;
% c = 1;
% rho = 1;
% Nmodes = 2;


```

```
kain = k*a;
modeindex = MPM_GetRectModeIndexing(Nmodes);
N = size(modeindex,1);
ZmatOut = zeros(N, N, length(k));

if abs(a/b-1)<0.1
%     load ZmatRecASq16x16.mat;
    load ZmatRecASq16x16.mat
    if length(Zmat(:,1,1)) < N
        error(sprintf('too many modes! Must be < %d', length(Zmat(:,1,1))));
        return;
    end;
    max(ka)
    L = length(k);
    for i=1:N%n
        for j=1:N%m
            negexpol = find(kain < min(ka));
            intpol = find((kain ≥ min(ka)) & (kain ≤ max(ka)));
            posexpol = find(kain > max(ka));
            Y = squeeze(real(Zmat(i,j,:)));
            R1 = interp1(ka', Y, kain(negexpol)','pchip','extrap');
            R2 = interp1(ka', Y, kain(intpol)','spline','extrap');
            if (i==j)
                R3 = ones(size(kain(posexpol)'));
            else
                R3 = zeros(size(kain(posexpol)'));
            end;
            %R3 = interp1(ka', Y, kain(posexpol)','linear','extrap');
            R = [R1; R2; R3];
            %R = ones(size(kain'));
            Y = squeeze(imag(Zmat(i,j,:)));
            X1 = interp1(ka', Y, kain(negexpol)','linear','extrap');
            X2 = interp1(ka', Y, kain(intpol)','spline','extrap');
            X3 = interp1(ka', Y, kain(posexpol)','linear','extrap');
            X = [X1; X2; X3];
            Zmn = R +1j*X;
            ZmatOut(i,j,:) = Zmn;
        end;
    end;
else
    error('No tabulated values for this aspect ratio!');
end;

ZmatOut = rho*c/(a*b)*ZmatOut; % the 4 in S (S=4*a*b) is already accounted for

clear Zmat;

% figure(1);
% Z = squeeze(ZmatOut(1,1,:));
% Z2 = squeeze(ZmatOut(2,3,:));
```

```
% semilogx(kain, real(Z), 'k', kain, imag(Z),'r', kain, real(Z2), 'b', kain, ...
    imag(Z2),'m');
```

C.4 *F*-MATRIX

Functions for calculating the *F*-matrix: the first loops through the horn, and then calls the second function for each discontinuity.

Listing 7: MPM_makebigfmat.m

```
% bigF = MPM_makebigfmat(N, coords, bz, geomtype)
%
% Calculates the scattering matrices F  for all discontinuities  in the
% horn. bigF(:,:,i) is the matrix F at position i in the horn.
%
% Input parameters:
% N : number of modes (in each direction for rectangular horns)
% coords : horn coordinates
% bz : zeros of Bessel function J1 (axisymmetric horns only)
% geomtype : determines the geometry of the horn:
%    0 : axisymmetric (default)
%    1 : quarter symmetric rectangular
%    2 : asymmetric rectangular
%    (no other geometries supported yet)
%
function bigF = MPM_makebigfmat(N, coords, bz, geomtype)
if nargin<4
    geomtype = 0;
end;

Lc = size(coords,1);
if geomtype == 0 % axisymmetric horn
    bigF = zeros(N,N,Lc);
    for iz = 1:Lc-1% (length(coords)-1):-1:1
        L = coords(iz+1,1) - coords(iz,1);
        if (L==0) %propagate across discontinuety
            R1 = coords(iz,2);
            R2 = coords(iz+1,2);
            F = MPM_ASmakefmat(N,R1,R2,bz);
            bigF(:,:,iz)=F;
        end;
    end;
elseif geomtype == 1 % quarter symmetric rectangular
    bigF = zeros(N^2,N^2,Lc);
    for iz = 1:Lc-1% (length(coords)-1):-1:1
        L = coords(iz+1,1) - coords(iz,1);
        if (L==0) %propagate across discontinuety
```

```
            a1 = coords(iz,2);
            a2 = coords(iz+1,2);
            b1 = coords(iz,3);
            b2 = coords(iz+1,3);
            F = MPM_RECmakefmat(N,a1,a2,b1,b2);
            bigF(:,:,iz)=F;
        end;
    end;
elseif geomtype == 2 %asymmetric rectangular
    bigF = zeros(N^2,N^2,Lc);
    for iz = 1:Lc-1% (length(coords)-1):-1:1
        L = coords(iz+1,1) - coords(iz,1);
        if (L==0) %propagate across discontinuety
            a1 = coords(iz,2:3);
            a2 = coords(iz+1,2:3);
            b1 = coords(iz,4:5);
            b2 = coords(iz+1,4:5);
            F = MPM_RECAmakefmat(N,a1,a2,b1,b2,false);
            bigF(:,:,iz)=F;
        end;
    end;
    %F = MPM_RECAmakefmat(N,a1,a2,b1,b2, forceexact);
else
    error(['Geometry type ' num2str(geomtype) ' not supported!']);
end;
```

Listing 8: MPM_RECmakefmat.m

```
% F = MPM_RECmakefmat(N,R1,R2,bz);
%
% Calculates the scattering matrix F, used to propagate modes across
% a discontinuity (rectangular case).
%
% Input parameters:
% N : number of modes in each direction
% a1 : half x-width of tube 1
% a2 : half x-width of tube 2
% b1 : half y-height of tube 1
% b2 : half y-height of tube 2
% forceexact : set to true to avoid approximation for small a1/a2 (not
% implemented)
%
function F = MPM_RECmakefmat(N,a1,a2,b1,b2, forceexact);
if nargin<6
    forceexact = false;
end;

betax = a1/a2;
betay = b1/b2;
```

```
22 bx=betax;
23 by=betay;
24 unitx = false;
25 unity = false;
26 vx = false;
27 vy = false;
28
29 if (betax > 1)
30     betax = 1/betax;
31     vx = true;
32 elseif (betax==1)
33     X = eye(N^2, N^2);
34     unitx = true;
35 end;
36
37 if (betay > 1)
38     betay = 1/betay;
39     vy = true;
40 elseif (betay==1)
41     Y = eye(N^2, N^2);
42     unity = true;
43 end;
44
45 if (unitx && unity)
46     F = eye(N^2, N^2);
47     return;
48 end;
49
50 if ((bx>1) & (by<1))|((bx<1) & (by>1))
51     warning('The horn is expanding in one dimension and contracting in the ...
           other.');
52     % expand first in x
53     Fx = MPM_RECmakefmat(N, a1, a2, b1, b1, forceexact);
54
55     % then expand in y
56     Fy = MPM_RECmakefmat(N, a2, a2, b1, b2, forceexact);
57 %     if vy
58 %         Fy = inv(Fy);
59 %     end;
60     if vx
61         F = Fx\Fy;
62     elseif vy
63         F = Fx/Fy;
64     else
65         F = Fx*Fy; %should not come here...
66     end;
67     return;
68 end;
69 X = zeros(N^2, N^2);
70 Y = X;
71 rt2 = sqrt(2);
```

```
72
73 modeindex = MPM_GetRectModeIndexing(N);
74 L = size(modeindex,1);
75
76 for n=1:L
77      for m=1:L
78          nx = modeindex(n,1);
79          ny = modeindex(n,2);
80          mx = modeindex(m,1);
81          my = modeindex(m,2);
82          if (nx==0)&(mx==0)
83              X(n,m) = 1;
84          elseif ((nx==0)&(mx>0))
85              X(n,m) = rt2*sinc(mx*betax);
86          else
87              X(n,m) = 2*sinc((mx*betax-nx))*mx*betax/(mx*betax+nx);
88          end;
89          if (ny==0)&(my==0)
90              Y(n,m) = 1;
91          elseif ((ny==0)&(my>0))
92              Y(n,m) = rt2*sinc(my*betay);
93          else
94              Y(n,m) = 2*sinc((my*betay-ny))*my*betay/(my*betay+ny);
95          end;
96      end;
97 end;
98
99 if unitx
100     X = eye(N^2);
101 end;
102 if unity
103     Y = eye(N^2);
104 end;
105
106 F = X.*Y;
```

Listing 9: MPM_RECAmakefmat.m

```
1  % F = MPM_RECAmakefmat(N,a1,a2,b1,b2, forceexact);
2  %
3  % Calculates the scattering matrix F, used to propagate modes across
4  % a discontinuity (rectangular, asymmetric case).
5  %
6  % Input parameters:
7  % N : number of modes in each direction
8  % a1 : x-widths of tube 1
9  % a2 : x-widths of tube 2
10 % b1 : y-heights of tube 1
11 % b2 : y-heights of tube 2
```

```
%  a1(1) is the negative (left) width, a(2) is the positive (right) width
%  etc.
% forceexact : set to true to avoid approximation for small a1/a2 (not
% implemented)
%
function F = MPM_RECAmakefmat(N,a1,a2,b1,b2, forceexact);
if nargin<6
    forceexact = false;
end;

alphax1 = a1(2)-a1(1);
alphax2 = a2(2)-a2(1);
alphay1 = b1(2)-b1(1);
alphay2 = b2(2)-b2(1);

betaxt = alphax1/alphax2;
betayt = alphay1/alphay2;
vx = false;
vy = false;

if (betaxt > 1)
    betaxt = 1/betaxt;
    vx = true;
end;

if (betayt > 1)
    betayt = 1/betayt;
    vy = true;
end;

if (vx & ¬vy)|(¬vx & vy)
    warning('The horn is expanding in one dimension and contracting in the ...
        other. NOT YET IMPLEMENTED!');
    F = eye(N^2);
    return;
end;

if (betaxt==1)&(betayt==1)
    F = eye(N^2);
    return;
end;

modeindex = MPM_GetRectModeIndexing(N);
L = size(modeindex,1);
ip = 2; in = 1;% indexing

A1 = a1(ip)-a1(in);
A2 = a2(ip)-a2(in);
beta_xt = A1/A2;
if beta_xt > 1
    beta_xt = 1/beta_xt;
```

```
    beta_xa = (a2(ip)+a2(in)-2*a1(in))/A1;
else
    beta_xa = (a1(ip)+a1(in)-2*a2(in))/A2;
end;

B1 = b1(ip)-b1(in);
B2 = b2(ip)-b2(in);
beta_yt = B1/B2;
if beta_yt > 1
    beta_yt = 1/beta_yt;
    beta_ya = (b2(ip)+b2(in)-2*b1(in))/B1;
else
    beta_ya = (b1(ip)+b1(in)-2*b2(in))/B2;
end;

rt2 = sqrt(2);
X = ones(L,L); Y = X;
for n=1:L
     for m=1:L
         nx = modeindex(n,1);
         ny = modeindex(n,2);
         mx = modeindex(m,1);
         my = modeindex(m,2);
         if (nx==0)&(mx==0)
             X(n,m) = 1;
         elseif ((nx==0)&(mx>0))
             X(n,m) = rt2*sinc(mx*beta_xt*0.5)*cos(mx*pi*0.5*beta_xa);
         else
             X(n,m) = sinc((nx-mx*beta_xt)/2)*cos(pi/2*(nx-mx*beta_xa))...
                 + sinc((nx+mx*beta_xt)/2)*cos(pi/2*(nx+mx*beta_xa));
         end;
         if (ny==0)&(my==0)
             Y(n,m) = 1;
         elseif ((ny==0)&(my>0))
             Y(n,m) = rt2*sinc(my*beta_yt*0.5)*cos(my*pi*0.5*beta_ya);
         else
             Y(n,m) = sinc((ny-my*beta_yt)/2)*cos(pi/2*(ny-my*beta_ya))...
                 + sinc((ny+my*beta_yt)/2)*cos(pi/2*(ny+my*beta_ya));
         end;
     end;
end;

F = X.*Y;

function FF = OneDimFMatrix(modeindex, a1, a2, xy, forceexact)
% xy is the is the direction, x=1,y=2
L = size(modeindex,1);
ip = 2; in = 1;% indexing
alpha1 = a1(ip)-a1(in);
alpha2 = a2(ip)-a2(in);
beta_xt = alpha1/alpha2;
```

```
if beta_xt > 1
    beta_xt = 1/beta_xt;
    beta_xa = (a2(ip)+a2(in)-2*a1(in))/alpha1;
else
    beta_xa = (a1(ip)+a1(in)-2*a2(in))/alpha2;
end;
rt2 = sqrt(2);
FF = zeros(L,L);
for n=1:L
     for m=1:L
         nx = modeindex(n,xy);
         mx = modeindex(m,xy);
         if (nx==0)&(mx==0)
             FF(n,m) = 1;
         elseif ((nx==0)&(mx>0))
             FF(n,m) = rt2*sinc(mx*beta_xt*0.5)*cos(mx*pi*0.5*beta_xa);
         else
             FF(n,m) = sinc((nx-mx*beta_xt)/2)*cos(pi/2*(nx-mx*beta_xa))...
                 + sinc((nx+mx*beta_xt)/2)*cos(pi/2*(nx+mx*beta_xa));
         end;
     end;
end;
```

C.5 IMPEDANCE CALCULATIONS

Function for calculating the impedances throughout the horn.

Listing 10: MPM_getimpedances.m

```
% BigZ = MPM_getimpedances(k, coords, bz, Zend, BigF, rho, c, geomtype, ...
    progressreport)
%
% Calculates the modal impedances at every duct junction in a horn defined
% by the coordinate list coords.
% The F matrices (BigF) must have been calculated on beforehand.
% The resulting matrix BigZ contains the modal impedances n,m at point iz
% and wavenumber index ik as BigF(n,m,iz,ik)
%
% Input parameters:
% k : wavenumber (vector)
% coords : horn coordinates
% bz : zeros of Bessel function J1 (axisymmetric horns only)
% Zend : (radiation) impedance at the mouth end of the horn
% BigF : scattering matrix F for all junctions
% rho : density of the medium
% c : sound speed in medium
% geomtype : (optional) determines the geometry of the horn:
%    0 : axisymmetric (default)
```

```
%    1 : quarter symmetric rectangular
%    2 : asymmetric rectangular
%    (no other geometries supported yet)
% progressreport : (boolean, optional) prints the current wavenumber and the
%    percentwise progress. Default is off. Calculations are slightly faster
%    with this option turned off.
%
function BigZ = MPM_getimpedances(k, coords, bz, Zend, BigF, rho, c, ...
    geomtype, progressreport)
if nargin<8
    geomtype = 0;
end;
if nargin<9
    progressreport = false;
end;

if geomtype == 0
    S = pi*coords(:,2).^2;
    N = size(BigF);
    N = N(1);
    I = eye(N);
    BigZ = zeros(N,N,length(coords), length(k));
    BigZ(:,:,end,:) = Zend;
    for fi = 1:length(k)
        if progressreport
            disp(['Calculating k = ' num2str(k(fi)) ' (' ...
                num2str(fi/length(k)*100) '%)']);
        end;
        kn = MPM_ASmakekm(k(fi),coords(end,2),bz,N);
        Z = Zend(:,:,fi);
        % propagate back to throat
        for iz = (size(coords,1)-1):-1:1
            R1 = coords(iz,2);
            R2 = coords(iz+1,2);
            %R2 = duct(iz+1,1);
            %beta = R1/R2;
            L = coords(iz+1,1) - coords(iz,1);
            if (L>0) %propagate along straight duct
                kn = MPM_ASmakekm(k(fi),R1,bz,N);
                D1 = diag(sin(L*kn));
                D2 = diag(1i*sin(L*kn));
                D3 = diag(tan(L*kn));
                Zc = diag(k(fi)*rho*c./(S(iz)*kn));
                D2Zc = D2\Zc;
                iD3Zc =(1i*D3)\Zc;
                invZc = diag((S(iz)*kn)./(k(fi)*rho*c));
                %Z = (Z + 1i*D3*Zc)/(1i*D3*invZc*Z+I);
                %Z = (1i*D3)^-1*Zc - D2^-1*Zc*(Z+(1i*D3)\Zc)^-1*D2^-1*Zc;
                Z = iD3Zc - D2Zc/(Z+iD3Zc) * D2Zc;
            else %propagate across discontinuety
                F = BigF(:,:,iz);
```

```
68                  if R1>R2
69                      Z = F\Z/F.';
70                  else
71                      Z = F*Z*F.';
72                  end;
73              end;
74              BigZ(:,:,iz,fi) = Z; %keep the impedance for velocity forward ...
                    propagation
75          end;
76      end;
77  elseif (geomtype == 1)|(geomtype == 2) % quarter symmetric and asymmetric ...
        rectangular
78      N2 = size(BigF,1);
79      N = sqrt(N2);
80      I = eye(N2);
81      if (geomtype == 1)
82          S = 4*coords(:,2).*coords(:,3);
83      else
84          S = (coords(:,3)-coords(:,2)).*(coords(:,5)-coords(:,4));
85      end;
86      modeindex = MPM_GetRectModeIndexing(N);
87      BigZ = zeros(N2,N2,size(coords,1), length(k));
88      for fi = 1:length(k)
89          if progressreport
90              disp(['Calculating k = ' num2str(k(fi)) ' (' ...
                    num2str(fi/length(k)*100) '%)']);
91          end;
92          %kn = MPM_RECmakekm(k(fi),coords(end,2),coords(end,3),N,modeindex);
93                 % Zc = diag(k(fi)*rho*c./(S(end)*kn));
94          Z = Zend(:,:,fi);
95          % propagate back to throat
96          for iz = (size(coords,1)-1):-1:1
97              if (geomtype == 1)
98                  alphax1 = coords(iz,2);
99                  alphax2 = coords(iz+1,2);
100                 alphay1 = coords(iz,3);
101                 alphay2 = coords(iz+1,3);
102             else
103                 a1 = coords(iz,2:3);
104                 a2 = coords(iz+1,2:3);
105                 b1 = coords(iz,4:5);
106                 b2 = coords(iz+1,4:5);
107                 alphax1 = a1(2)-a1(1);
108                 alphax2 = a2(2)-a2(1);
109                 alphay1 = b1(2)-b1(1);
110                 alphay2 = b2(2)-b2(1);
111             end;
112             L = coords(iz+1,1) - coords(iz,1);
113             if (L>0) %propagate along straight duct
114                 if (geomtype == 1)
115                     kn = MPM_RECmakekm(k(fi),alphax1,alphay1,N,modeindex);
```

```
116                 else
117                     kn = MPM_RECAmakekm(k(fi),a1,b1,N,modeindex);
118                 end;
119                 D1 = diag(sin(L*kn));
120                 D2 = diag(1i*sin(L*kn));
121                 D3 = diag(tan(L*kn));
122                 Zc = diag(k(fi)*rho*c./(S(iz)*kn));
123                 D2Zc = D2\Zc;
124                 iD3Zc =(1i*D3)\Zc;
125                 %invZc = diag((S(iz)*kn)./(k(fi)*rho*c));
126                 %Z = (Z + 1i*D3*Zc)/(1i*D3*invZc*Z+I);
127                 %Z = (1i*D3)^-1*Zc - D2^-1*Zc*(Z+(1i*D3)\Zc)^-1*D2^-1*Zc;
128                 Z = iD3Zc - D2Zc/(Z+iD3Zc) * D2Zc;
129
130                 %D3 = diag(tan(L*kn));
131                 %Zc = diag(k(fi)*rho*c./(S(iz)*kn));
132                 %invZc = diag((S(iz)*kn)./(k(fi)*rho*c));
133                 %Z = (Z + 1i*D3*Zc)/(1i*D3*invZc*Z+I);
134                 %%Z = (1i*D3)^-1*Zc - D2^-1*Zc*(Z+(1i*D3)\Zc)^-1*D2^-1*Zc;
135                 %%Z = D3\Zc - D2\(Z+D3\Zc)\D2\I;
136             else %propagate across discontinuety
137                 F = BigF(:,:,iz);
138                 if (alphax1>alphax2) & (alphay1>alphay2)% contraction in both ...
                        directions
139                     Z = F\Z/F.';
140                 else
141                     Z = F*Z*F.';
142                 end;
143             end;
144             BigZ(:,:,iz,fi) = Z; %keep the impedance for velocity forward ...
                    propagation
145         end;
146     end;
147 end;
```

Helper function for MPM_getimpedances.m.

Listing 11: MPM_RECmakekm.m

```
1  % kn = MPM_RECmakekm(k,a,b,M,ModeIndices)
2  %
3  % Calculates the modal wave number matrix km.
4  %
5  % Input parameters:
6  % k : free space wave number
7  % a : half width of the duct
8  % b : half height of the duct
9  % M : number of modes in each direction
10 % ModeIndices : sorted index of modes
11 %
```

```
function kn = MPM_RECmakekm(k,a,b,M,ModeIndices)
alpha2 = (ModeIndices(:,1)*pi/a).^2+(ModeIndices(:,2)*pi/b).^2;
alpha2m = alpha2(:,ones(1,length(k)));
kn = k(ones(M^2,1),:);
co=find(alpha2m>kn.^2);
kn = sqrt(kn.^2-alpha2m);
kn(co)=-kn(co);
```

Listing 12: MPM_RECAmakekm.m

```
% kn = MPM_RECAmakekm(k,a,b,M,ModeIndices)
%
% Calculates the modal wave number matrix km.
%
% Input parameters:
% k : free space wave number
% a : vector of duct widths, [a- a+]
% b : vector of duct heights, [b- b+]
% M : number of modes in each direction
% ModeIndices : sorted index of modes
%
function kn = MPM_RECAmakekm(k,a,b,M,ModeIndices)
an = a(1);
ap = a(2);
bn = b(1);
bp = b(2);
A = ap-an;
B = bp-bn;
alpha2 = (ModeIndices(:,1)*pi/A).^2+(ModeIndices(:,2)*pi/B).^2;
alpha2m = alpha2(:,ones(1,length(k)));
kn = k(ones(M^2,1),:);
co=find(alpha2m>kn.^2);
kn = sqrt(kn.^2-alpha2m);
kn(co)=-kn(co);
```

C.6 VOLUME VELOCITY CALCULATIONS

Function to calculate the mouth volume velocity.

Listing 13: MPM_getmouthvolvel.m

```
% Umouth = MPM_getmouthvolvel(k, coords, U0, bz, BigZ, BigF, rho, c, geomtype)
%
% Calculates the modal volume velocity at the mouth of a horn defined
% by the coordinate list coords, given a modal throat velocity U0.
```

```
% The F and Z matrices (BigF, BigZ) must have been calculated on beforehand.
%
% Input parameters:
% k : wavenumber (vector)
% coords : horn coordinates
% U0 : throat modal velocity
% bz : zeros of Bessel function J1 (for geomtype = 1)
% BigZ : impedance matrices for all points in the horn, and all wavenumbers
% BigF : scattering matrix F for all junctions
% rho : density of the medium
% c : sound speed in medium
% geomtype : (optional) determines the geometry of the horn:
%    0 : axisymmetric (default)
%    1 : quarter symmetric rectangular
%    2 : asymmetric rectangular
%    (no other geometries supported yet)
%
function Umouth = MPM_getmouthvolvel(k, coords, U0, bz, BigZ, BigF, rho, c, ...
    geomtype)
if nargin<8
    geomtype = 0;
end;

Umouth = [];
SS = size(U0);

if geomtype==0
    S = pi*coords(:,2).^2;
    N = size(BigF,1);
    for ki = 1:length(k)
        if SS(2)>1
             U = U0(:,ki);
        else
             U = U0;
        end;
        Sm = S(end);
        for iz = 1:length(coords)-1
            R1 = coords(iz,2);
            R2 = coords(iz+1,2);
            L = coords(iz+1,1) - coords(iz,1);
            if (L>0)
                Z0 = BigZ(:,:,iz,ki);
                kn = MPM_ASmakekm(k(ki),R1,bz,N);
                D2 = diag(1i*sin(L*kn));
                E = diag(exp(-1i*L*kn));
                Zc = diag(k(ki)*rho*c./(S(iz)*kn));
                invZc = diag((S(iz)*kn)./(k(ki)*rho*c));
                U = (-D2*invZc*(Z0-Zc)+E)*U;
            else
                F = BigF(:,:,iz);
                if R1>R2
```

```
55                      U = (F.')\U;
56                  else
57                      U = F.'*U;
58                  end;
59              end;
60          end;
61          Umouth = [Umouth U];
62      end;
63  elseif (geomtype == 1)|(geomtype == 2) % quarter symmetric and asymmetric ...
        rectangular
64      N2 = size(BigF,1);
65      N = sqrt(N2);
66      if (geomtype == 1)
67          S = 4*coords(:,2).*coords(:,3);
68      else
69          S = (coords(:,3)-coords(:,2)).*(coords(:,5)-coords(:,4));
70      end;
71      modeindex = MPM_GetRectModeIndexing(N);
72      for ki = 1:length(k)
73          if SS(2)>1
74               U = U0(:,ki);
75          else
76               U = U0;
77          end;
78          Sm = S(end);
79
80          for iz = 1:length(coords)-1
81              if (geomtype == 1)
82                  alphax1 = coords(iz,2);
83                  alphax2 = coords(iz+1,2);
84                  alphay1 = coords(iz,3);
85                  alphay2 = coords(iz+1,3);
86              else
87                  a1 = coords(iz,2:3);
88                  a2 = coords(iz+1,2:3);
89                  b1 = coords(iz,4:5);
90                  b2 = coords(iz+1,4:5);
91                  alphax1 = a1(2)-a1(1);
92                  alphax2 = a2(2)-a2(1);
93                  alphay1 = b1(2)-b1(1);
94                  alphay2 = b2(2)-b2(1);
95              end;
96              L = coords(iz+1,1) - coords(iz,1);
97              if (L>0)
98                  Z0 = BigZ(:,:,iz,ki);
99                  if (geomtype == 1)
100                     kn = MPM_RECmakekm(k(ki),alphax1,alphay1,N,modeindex);
101                 else
102                     kn = MPM_RECAmakekm(k(ki),a1,b1,N,modeindex);
103                 end;
104 %                 kn = MPM_ASmakekm(k(ki),R1,bz,N);
```

```
                D2 = diag(1i*sin(L*kn));
                E = diag(exp(-1i*L*kn));
                Zc = diag(k(ki)*rho*c./(S(iz)*kn));
                invZc = diag((S(iz)*kn)./(k(ki)*rho*c));
                U = (-D2*invZc*(Z0-Zc)+E)*U;
            else
                F = BigF(:,:,iz);
                if (alphax1>alphax2) & (alphay1>alphay2)
                    U = (F.')\U;
                else
                    U = F.'*U;
                end;
            end;
        end;
        Umouth = [Umouth U];

    end;

end;
```

C.7 FIELD POINT PRESSURE CALCULATIONS

Various functions to calculate the raditated pressure. This function uses the Rayleigh integral.

Listing 14: MPM_getfieldpointpressures.m

```
% prext = MPM_getfieldpointpressures(k, Umouth, pe, DimM, Nr, bz, rho, c, ...
    geomtype)
%
% Calculates the sound pressure of a horn in the free field, given the mouth ...
    velocity
% and the field point coordinates. The mouth is divided into Nr concentric
% rings, and the particle velocity of each ring is determined. Numerical
% integration is performed over the surface using the Rayleigh integral, to
% obtain the pressure in the free field.
%
% Input parameters:
% k : wavenumber (vector)
% Umouth : modal mouth velocity
% pe : field point coordinate list. One point per row. Columns are (z,r) or
%      (z, x, y).
% DimM : mouth dimensions: radius for axisymmetric horns, x,y for
%      rectangular horns.
% Nr : mouth radius resolution: the number of points in the radial direction.
%      Determines partly the resolution of the calculated field point
%      pressure.
```

```
% bzN : zeros of Bessel function J1 (for geomtype 0), number of modes for
%      rectangular geometries.
% rho : density of the medium
% c : sound speed in medium
% geomtype : (optional) determines the geometry of the horn:
%    0 : axisymmetric (default)
%    1 : quarter symmetric rectangular
%    2 : asymmetric rectangular
%    (no other geometries supported yet)

function prext = MPM_getfieldpointpressures(k, Umouth, pe, DimM, Nr, bzN, ...
    rho, c, geomtype)
if nargin<8
    geomtype = 0;
end;

prext = [];

if geomtype==0
    r = (0:DimM/(Nr-1):DimM);
    rp2=r(2:end);
    rp1=r(1:end-1);
    rp=(rp1+rp2)/2;
    a = max(r);
    phi = MPM_ASgeteigenfunctions(a, rp,  bzN, true);
    Ur = phi*Umouth; %volume velocity as function of radius
    uo = Ur / (a^2*pi); % mouth particle velocity

    prext = zeros(length(pe), length(k));
    Np = size(pe);
    for ii = 1:Np(1)
        pext = pe(ii,:);
        pr = MPM_ASrayleighint(k, r, uo, pext, rho, c);
        prext(ii,:) = pr;
    end;
elseif (geomtype==1)|(geomtype==2)
    [phi, coords] = MPM_RECgeteigenfunctions(bzN, DimM, Nr, Nr, geomtype);
    Ur = phi*Umouth;
    uo = Ur/(DimM(2)*DimM(3)*4);
    Np = size(pe);
    prext = zeros(Np(1), length(k));
    % make coordinates 3D, add z-axis
    Lc = size(coords, 1);
    Cz = zeros(Lc, 1);
    coords = [coords Cz];
    dx = DimM(2)*2/Nr;
    dy = DimM(3)*2/Nr;
    dS = dx*dy;
    for ii = 1:Np(1)

        pext = pe(ii,:);
```

```
        pr = RECrayleighint(k, uo, pext, coords, rho, c, dS);
        prext(ii,:) = pr;
    end;
end;

function pr = RECrayleighint(k, uo, pext, coords, rho, c, dS)
pr = zeros(size(k));
pext = pext(ones(size(coords,1),1),:);
% create vector of distances from pext to all coord points on the surface
r = sqrt(sum(((coords-pext).^2)'));
% calculate the Rayleigh integral
for ik = 1:length(k)
    expmat = exp(-1j*k(ik)*r)./r;
    integrand = uo(:,ik).*expmat;
    integral = dS*sum(integrand);
    pr(ik) = 1i*k(ik)*rho*c/(2*pi)*integral;
end;
```

This function uses the modal amplitudes in the mouth directly. For symmetrical horns.

Listing 15: MPM_RECmodalradiatedpressure.m

```
% prext = MPM_RECmodalradiatedpressure(k, a, b, v0, Pe, rho, c)
%
% Calculates the radiated pressure in front of a baffled
% rectangular aperture with a given modal velocity distribution.
% The radiator is assumed to be in the z = 0 plane.
% Total number of modes included depends on the size of v0.
%
% Input parameters:
% k : wavenumber
% a, b : x and y dimensions
% v0 : the modal velocity amplitudes
% Pe : exterior point, (x,y,z).
% rho : density of the medium
% c : sound speed in medium
%
function prext = MPM_RECmodalradiatedpressure(k, a, b, v0, Pe, rho, c)
Nmodes = sqrt(length(v0(:,1)));
modeindex = MPM_GetRectModeIndexing(Nmodes);
Np = size(Pe,1);
prext = zeros(Np, length(k));
%avoid very small numbers, that may give non-zero values for thetax and
%thetay even if the angle really is zero.
iv = find(abs(Pe) < 1e-6);
Pe(iv) = 0;
for ii = 1:Np
    pe = Pe(ii,:);
    R = norm(pe);
    thetax = atan2(pe(1),pe(3));%y,x
```

```
    thetay = atan2(pe(2),pe(3));
    sx = k*a*sin(thetax);
    sy = k*b*sin(thetay);
    ModalSum = 0;
    for nm = 1:Nmodes^2
       AnBm = v0(nm,:);
       n = modeindex(nm,1);
       m = modeindex(nm,2);
       if n>0
           AnBm = AnBm*sqrt(2);
       end;
       if m>0
           AnBm = AnBm*sqrt(2);
       end;
       Gn = fGn(sx,n);
       Gm = fGn(sy,m);
       ModalSum = ModalSum + AnBm.*Gn.*Gm;
    end;
    pf = 1i*rho*c*(4*a*b)/(2*pi*R)*exp(-1i*k*R).*k;
    prext(ii,:) = pf.*ModalSum;
end;

function g = fGn(x,n)
if n==0
    g = sinc(x/pi);
else
    g = (-1)^n*(x.*sin(x))./(x.^2-(n*pi)^2);
end;
```

Listing 16: MPM_RECgeteigenfunctions.m

```
% [psi, coords] = MPM_RECgeteigenfunctions(Nmodes, throatdims, Nx, Ny, geomtype)
%
% Calculates the eigenfunctions for a rectangular duct of dimensions given
% in throatdims at the coordinates calculated based on Nx and Ny.
%
% Input parameters:
% Nmodes: number of modes in each direction
% dims: dimensions of the duct, in the same format as ordinary
%       horn coordinates (column 1 is z)
% Nx, Ny : number of sample points in each direction
% geomtype : determines the geometry of the horn:
%     0 : axisymmetric (not supported in this function, use ...
    MPM_ASgeteigenfunctions)
%     1 : quarter symmetric rectangular
%     2 : asymmetric rectangular
%     (no other geometries supported yet)
%
% Output parameters:
```

```
% psi : the eigenfunction matrix. Each column is a mode combination,
%     each row is a point on the surface, given in coords.
%     The matrix is (Nx * Ny) rows and (Nmodes^2) columns.
% coords : list of the coordinates used
function [psi, coords] = MPM_RECgeteigenfunctions(Nmodes, dims, Nx, Ny, ...
    geomtype, thx, thy)
modeindex = MPM_GetRectModeIndexing(Nmodes);
if geomtype==1
    % get the x and y coordinates at the center of rectangular elements
    if nargin < 6
        thx = ([-Nx/2:Nx/2-1]+0.5)*2*dims(end,2)/Nx;
        thy = ([-Ny/2:Ny/2-1]+0.5)*2*dims(end,3)/Ny;
    end;
%     thx = linspace(-dims(1,2),dims(1,2),Nx);
%     thy = linspace(-dims(1,3),dims(1,3),Ny);

    [X,Y] = meshgrid(thx,thy);
    coords = [reshape(X,Nx*Ny,1),reshape(Y,Nx*Ny,1)];% zeros(nx*ny,1)];
    % eigenfunctions, x
    ax = coords(:,1); % x values
    ax = ax(:,ones(1,Nmodes^2));
    modenx = modeindex(:,1)';
    modenx = modenx(ones(size(ax,1),1),:);
    phinx = sqrt(2)*cos(modenx.*ax*pi/dims(1,2));
    m0 = find(modenx==0);
    phinx(m0) = 1;
    % eigenfunctions, y
    ay = coords(:,2); % y values
    ay = ay(:,ones(1,Nmodes^2));
    modeny = modeindex(:,2)';
    modeny = modeny(ones(size(ay,1),1),:);
    sigmay = sqrt(2)*cos(modeny.*ay*pi/dims(1,3));
    m0 = find(modeny==0);
    sigmay(m0) = 1;
elseif geomtype==2
    %thx = ([0:Nt-1]+0.5)*throatdims(1,2)/Nt;% more to do here!!!!!
    aneg = dims(end,2);
    apos = dims(end,3);
    bneg = dims(end,4);
    bpos = dims(end,5);
    if nargin < 6
        thx = linspace(aneg,apos,Nx);
        thy = linspace(bneg,bpos,Ny);
    end;
    [X,Y] = meshgrid(thx,thy);
    coords = [reshape(X,Nx*Ny,1),reshape(Y,Nx*Ny,1)];% zeros(nx*ny,1)];
    % eigenfunctions, x
    ax = coords(:,1); % x values
    ax = ax(:,ones(1,Nmodes^2));
    modenx = modeindex(:,1)';
    modenx = modenx(ones(size(ax,1),1),:);
```

```
%     phinx = sqrt(2)*cos(modenx.*ax*pi/dims(1,2));
    phinx = sqrt(2)*cos(modenx.*(ax-aneg)*pi/(apos-aneg));
    m0 = find(modenx==0);
    phinx(m0) = 1;
    % eigenfunctions, y
    ay = coords(:,2); % y values
    ay = ay(:,ones(1,Nmodes^2));
    modeny = modeindex(:,2)';
    modeny = modeny(ones(size(ay,1),1),:);
    sigmay = sqrt(2)*cos(modeny.*(ay-bneg)*pi/(bpos-bneg));
%     sigmay = sqrt(2)*cos(modeny.*ay*pi/dims(1,3));
    m0 = find(modeny==0);
    sigmay(m0) = 1;
end;
psi = phinx.*sigmay;
```

C.8 OTHER FUNCTIONS AND FILES

Various helper functions or utility functions used in the toolbox.

Listing 17: MPM_GetRectModeIndexing.m

```
% rmi = MPM_GetRectModeIndexing(Nmodes);
%
% Creates a vector of (m,n) mode pairs for rectangular ducts. The same
% number of modes is assumed in each direction.
%
% input parameter:
% Nmodes : number of modes in each direction
function rmi = MPM_GetRectModeIndexing(Nmodes);

Ntot = Nmodes.^2;
rmi = zeros(Ntot,2);
indx = 1;
for i=0:Nmodes-1
    for j=0:i
        if j==i
            rmi(indx,:) = [i j];
            indx = indx+1;
        else
            rmi(indx,:) = [j i];
            indx = indx+1;
            rmi(indx,:) = [i j];
            indx = indx+1;
        end;
    end;
end;
```

D

TESTING FUNCTIONS

This appendix includes the functions and scripts used to test out various parts of the bend modal method.

D.1 BESSEL RELATED FUNCTIONS

Function to calculate the real roots of the dispersion relation for real arguments. This function may be extended to all roots.

Listing 18: BesselCrossproductRoots.m

```
function BCR = BesselCrossproductRoots(kr, R1, R2)

if abs(imag(kr))  < 1e-6*abs(real(kr))
    % real argument
    kr = real(kr);
    % do the initial bracketing of the real roots
    x1 = 0;
    x2 = kr*R2; % angular wave number should be ≤ kr ???
    n = 1000; % this should be enough???
    xb1 = zeros(1,n);
    xb2 = zeros(1,n);

    nbb=1;
    v = x1;
    dx = (x2-x1)/n;
    fp = RealCrossproduct(v,kr*R1, kr*R2);
    for i=1:n
        v = v+dx;
        fc = RealCrossproduct(v,kr*R1, kr*R2);
        if (fc*fp) < 0
            xb1(nbb) = v-dx;
            xb2(nbb) = v;
            %disp(sprintf('Interval: %f-%f , function values: %e  %e', ...
                xb1(nbb), xb2(nbb), fc, fp));
            nbb = nbb+1;

        end;
        fp = fc;
    end;
    %disp(sprintf('%d roots in the interval %d,%d', nbb-1, x1, x2));
    BCR = zeros(nbb-1,1);

    for zn = 1:nbb-1
        v0 = [xb1(zn) xb2(zn)];
        BCR(zn) = fzero(@(v) RealCrossproduct(v,kr*R1, kr*R2), v0);
    end;
%     for zn=1:nbb-1
%         disp(sprintf('Interval: %f-%f , zero: %f', xb1(zn), xb2(zn), bzs(zn)));
```

```
%     end;
end;
```

```
function rcpv = RealCrossproduct(v,A,B)
rcpv = dbesselj(v,A).*dbessely(v,B) - dbesselj(v,B).*dbessely(v,A);
```

Functions for calculating the derivative of Bessel functions J_ν and Y_ν.

Listing 19: dbesselj.m

```
function dJ = dbesselj(v,x)
%dJ = 0.5*(besselj(v-1,x) - besselj(v+1,x));

if (abs(imag(v)) < abs(1e-8*real(v))) | (imag(v) == 0)
    v = real(v);
    dJ = -besselj(v+1,x)+v./x.*besselj(v,x);
else
    dJ = 0*v;
    for iv = 1:length(v)
        vv = v(iv);
        dJ(iv) = -gbesselj(vv+1,x)+vv./x.*gbesselj(vv,x);
    end;
end;
```

Listing 20: dbessely.m

```
function dY = dbessely(v,x)
%dY = 0.5*(bessely(v-1,x) - bessely(v+1,x));
dY = -bessely(v+1,x)+v./x.*bessely(v,x);
```

Function for calculating the Bessel function J_ν for all orders and arguments by series expansion. Works for real as well as for imaginary and complex arguments, but is somewhat slow.

Listing 21: gbesselj.m

```
function bj = gbesselj(v,z)

% series expansion
bj = zeros(length(v), length(z));
L = length(z);
for jj = 1:length(v)
    n = fix(v(jj));
```

```
    for ii = 1:L
        bjx = 0;
        m = 0;
        factor = 1e6;
        if (abs(n-v(jj)) > 0) | (abs(imag(v)) > 0)
            while abs(factor) > 1e-12
                sgn = (-1)^m;
                if m==0
                    invfac = 1;
                else
                    invfac = invfac / m;
                end;
                factor = (z(ii)/2).^(2*m+v(jj));
                factor = invfac * factor ./ cgamma(m+v(jj)+1);
                bjx = bjx + sgn*factor;
                m = m+1;
            end;
        else
            bjx = besselj(v(jj),z(ii));
        end;

        bj(jj,ii) = bjx;
    end;
end;
```

D.2 DISPERSION RELATION

Script for plotting the dispersion relation.

Listing 22: DispRelationPlotting.m

```
% angular wave number

% equation: g'(x,rho) = J'v(x)*Y'v(rho*x) - J'v(rho*x)*Y'v(x) = 0
% test 1: real roots of g' for real arguments
k = 25;
Ri = 0.5;
Ro = 1.5;

vmax = 50;

x = k*Ri;
rho = Ro/Ri;

v = linspace(0,vmax,300);
lims = [-1 1];

s = 1:vmax;
```

```
asrt = s*pi/log(k);
asrtv = 0*asrt;

y = disprelation(v,x,rho);%dbesselj(v,x).*dbessely(v,rho*x) - ...
    dbesselj(v,rho*x).*dbessely(v,x);
y=RealCrossproduct(v,k*Ri,k*Ro);%
figure(1);
plot(v,real(y));%,v,yy3);%, v, real(yy),asrt,asrtv, 'o' );%,v,ya);%.\ ...
    imag(y));%, v, imag(y));
title(sprintf('Dispersion relation, k=%.2f, Ri=%.2f, Ro=%.2f',k,Ri,Ro));
ylim(lims);
xlim([min(v),max(v)]);
grid;
xlabel('\nu');
ylabel('g''(x,\rho)');
roots = BesselCrossproductRoots(k,Ri,Ro)

%%
%test 2: imaginary roots of g' for real arguments,
% comparison with Cochran's approximation
k = 25;
Ri = 0.5;
Ro = 1.5;

vmax = 5;
% x = 8;
% rho = 0.2;

x = k*Ri;
rho = Ro/Ri;

v = 10;
v = linspace(0,vmax,500);
lims = [-1 1];

s = 1:vmax;
asrt = s*pi/log(k);
asrtv = 0*asrt;

y = disprelation(v*1i,x,rho);%dbesselj(v,x).*dbessely(v,rho*x) - ...
    dbesselj(v,rho*x).*dbessely(v,x);
yd = dbesselj(v,x)./dbessely(v,x);
yy = -(1i*v/(rho*pi*x^2)).*exp(v*pi).*sin(v*log(rho));
%ya = -(v./(rho*pi*x^2)).*exp(v*log(rho));
r = real(y);
yi = y;
y = imag(y);

s = sign(y);
y2 = s.*log10(abs(y));
yy = imag(yy);
```

```
s = sign(yy);
yy3 = s.*log10(abs(yy));

%%
figure(2);
plot(v,y2,v,yy3);%,v,yy3);%, v, real(yy),asrt,asrtv, 'o' );%,v,ya);%.\ ...
    imag(y));%, v, imag(y));
title(sprintf('Dispersion relation, k=%.2f, Ri=%.2f, Ro=%.2f',k,Ri,Ro));
% ylim(lims);
xlim([min(v),max(v)]);
xlabel('\nu');
ylabel('g''(x,\rho)');
legend('Actual','Approximation');
%%
figure(3);
plot(v,real(yi));
```

Function used in the above script

Listing 23: disprelation.m

```
function BZ = disprelation(v,x,rho)
if (abs(imag(v)) < 1e-12*real(v)) | (imag(v) == 0)
    BZ = dbesselj(v,x).*dbessely(v,rho*x) - dbesselj(v,rho*x).*dbessely(v,x);
else
    BZ = dbesselj(v,x).*dbesselj(-v,rho*x) - dbesselj(v,rho*x).*dbesselj(-v,x);
end;
```

Script to plot the sound field in a bend

Listing 24: test_fieldplot.m

```
% test plotting field

R1 = 0.2;
R2 = 0.5;
x0 = [0 0];
k = 15;
% k = 1600*2*pi/344;
m = 10;

Rmr = @(k_phi, k_r, r, R1) besselj(k_phi,k_r*r) - ...
    (dbesselj(k_phi,k_r*R1)./dbessely(k_phi,k_r*R1)).*bessely(k_phi,k_r*r);

r = linspace(R1,R2, 100)';
ang = linspace(0,pi, 200);

broots = BesselCrossproductRoots(k, R1, R2);
```

```
% length(broots)
mmax = min(length(broots), m);
% kphi = broots(m);
% mode function in r direction
M = zeros(length(r),length(ang));
for m=1:min(length(broots), mmax);
    kphi = broots(m);
    Rdir = Rmr(kphi, k, r, R1);
    Angdir = exp(1i*kphi*ang);
    Mm = Rdir*Angdir;
%     Rmat = Rdir(:,ones(1,length(ang)));
%     Angmat = Angdir(ones(length(r),1),:);
%     Rmat .* Angmat - Mm
    M = M + Mm;
end;


% plotting
x = r*cos(ang);
y = r*sin(ang);
z = 20*log10(abs(M));%real(M); %sin(x.^2+log(y+1))+sin(x.^2+x);


figure(1);
contourf(x,y,z,25);
view(2);
axis equal
shading flat
colorbar;
title(sprintf('sound field at k = %.2f, %d propagating modes', k, mmax));
xlabel('x [m]');
ylabel('y [m]');

%%
fpdata = readFPdata('horns\ArcField_FPData.txt');
x2=fpdata.coords(:,1);
y2=fpdata.coords(:,2);
[x3,y3] = meshgrid(x2,y2);
z2=x3;%real(fpdata.data(1,:));
figure(2);
contourf(x3,y3,z2,25);
view(2);
axis equal
% shading flat
colorbar;
```

E

FILES

The files listed above, and the complete MPM toolbox code are included in the attached zip-file. In addition, the geometry files for the three horns investigated, and the text files containing containing the data from the BERIM simulations (including functions to read them into Matlab) are included.

NTNU	Kartlegging av risikofylt aktivitet	Utarbeidet av	Nummer	Dato	
		HMS-avd.	HMSRV2601	22.03.2011	
		Godkjent av	Side	Erstatter	
HMS		Rektor	1 av 1	01.12.2006	

Enhet: Institutt for Elektronikk og Telekommunikasjon (IET) **Dato:** 11.06.13
Deltakere ved kartleggingen (m/ funksjon): Bjørn Kolbrek (student), Peter Svensson (veileder)

Kort beskrivelse av hovedaktivitet/hovedprosess: Programmering

ID nr.	Aktivitet/prosess	Ansvarlig	Eksisterende dokumentasjon	Eksisterende sikringstiltak	Lov, forskrift o.l.	Kommentar
1	Musearm og stive skuldre	BK				

NTNU	Risikovurdering	utarbeidet av	Nummer	Dato	
		HMS-avd.	HMSRV2603	04.02.2011	
		godkjent av	side	Erstatter	
HMS/KS		Rektor	1 av 2	9.2.2010	

Enhet: Institutt for Elektronikk og Telekommunikasjon (IET) **Dato:** 11.06.13
Linjeleder: Ragnar Hergum
Deltakere ved risikovurderingen (m/ funksjon): Bjørn Kolbrek (student), Peter Svensson (veileder)

ID nr	Aktivitet fra kartleggings-skjemaet	Mulig uønsket hendelse/ belastning	Vurdering av sannsyn-lighet (1-5)	Vurdering av konsekvens: Menneske (A-E)	Ytre miljø (A-E)	Øk/ materiell (A-E)	Om-dømme (A-E)	Risiko-verdi	Kommentarer/status Forslag til tiltak
1	Musearm og stive skuldre		3	B	A	A	A	3B	Sitte riktig, pauser

Sannsynlighet
1. Svært liten
2. Liten
3. Middels
4. Stor
5. Svært stor

Konsekvens
A. Svært liten
B. Liten
C. Moderat
D. Alvorlig
E. Svært alvorlig

Risikoverdi (beregnes hver for seg):
Menneske = Sannsynlighet x Konsekvens Menneske
Ytre miljø = Sannsynlighet x Konsekvens Ytre miljø
Økonomi/materiell = Sannsynlighet x Konsekvens Øk/matriell
Omdømme = Sannsynlighet x Konsekvens Omdømme

Sannsynlighet vurderes etter følgende kriterier:

Svært liten 1	Liten 2	Middels 3	Stor 4	Svært stor 5
1 gang pr 50 år eller sjeldnere	1 gang pr 10 år eller sjeldnere	1 gang pr år eller sjeldnere	1 gang pr måned eller sjeldnere	Skjer ukentlig

Konsekvens vurderes etter følgende kriterier:

Gradering	Menneske	Ytre miljø Vann, jord og luft	Øk/materiell	Omdømme
E **Svært Alvorlig**	Død	Svært langvarig og ikke reversibel skade	Drifts- eller aktivitetsstans >1 år.	Troverdighet og respekt betydelig og varig svekket
D **Alvorlig**	Alvorlig personskade. Mulig uførhet.	Langvarig skade. Lang restitusjonstid	Driftsstans > ½ år Aktivitetsstans i opp til 1 år	Troverdighet og respekt betydelig svekket
C **Moderat**	Alvorlig personskade.	Mindre skade og lang restitusjonstid	Drifts- eller aktivitetsstans < 1 mnd	Troverdighet og respekt svekket
B **Liten**	Skade som krever medisinsk behandling	Mindre skade og kort restitusjonstid	Drifts- eller aktivitetsstans < 1uke	Negativ påvirkning på troverdighet og respekt
A **Svært liten**	Skade som krever førstehjelp	Ubetydelig skade og kort restitusjonstid	Drifts- eller aktivitetsstans < 1dag	Liten påvirkning på troverdighet og respekt

Risikoverdi = Sannsynlighet x Konsekvens
Beregn risikoverdi for Menneske. Enheten vurderer selv om de i tillegg vil beregne risikoverdi for Ytre miljø, Økonomi/materiell og Omdømme. I så fall beregnes disse hver for seg.

Til kolonnen "Kommentarer/status, forslag til forebyggende og korrigerende tiltak":
Tiltak kan påvirke både sannsynlighet og konsekvens. Prioriter tiltak som kan forhindre at hendelsen inntreffer, dvs. sannsynlighetsreduserende tiltak foran skjerpet beredskap, dvs. konsekvensreduserende tiltak.