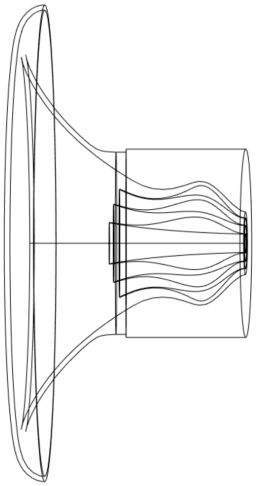
Application Note 1

# Spherical Wave Forming



Version 1, December 2021

## Introduction

Whereas a flat exit wavefront has been historically a design goal in majority of compression drivers and their phasing plugs, acoustically a spherical wave of a puslating cap is a fundamentally better suited for driving constant directivity waveguides. As the size of a waveguide throat is increased, inevitably the high-frequency beaming, inherent to all axially vibrating sources (either a rigid piston or a flat exit wavefront of a compression driver) shifts to lower frequencies - for a 1" throat it's practical to reach a true constant directivity of 90° up to around 16 kHz, for a 2" throat this limit is around 8 kHz.

If a suitable waveguide was driven with an ideal spherical wavefront instead of flat, no high-frequency beaming would occur, irrespective of the throat diameter. One solution would be to design a whole new compression driver and especially its phase plug for that very purpose, with a possible downside that the driver would be actually optimized for only one coverage angle. Perhaps a more practical approach is to utilize an existing compression driver but convert the exit wavefront into spherical by some external means, before the sound wave is let to propagate freely in the waveguide - that's where the design of an e*xternal shaping plug (ESP)* comes in. It's a device for converting a flat wavefront into spherical by dividing the incoming sound wave into several discrete channels and combining the outputs again in a form of a spherical wave.

In this application note a design of an external shaping plug depicted on Fig 1 is described. ESP is implemented in Ath[1] version 4.8.0 an above.

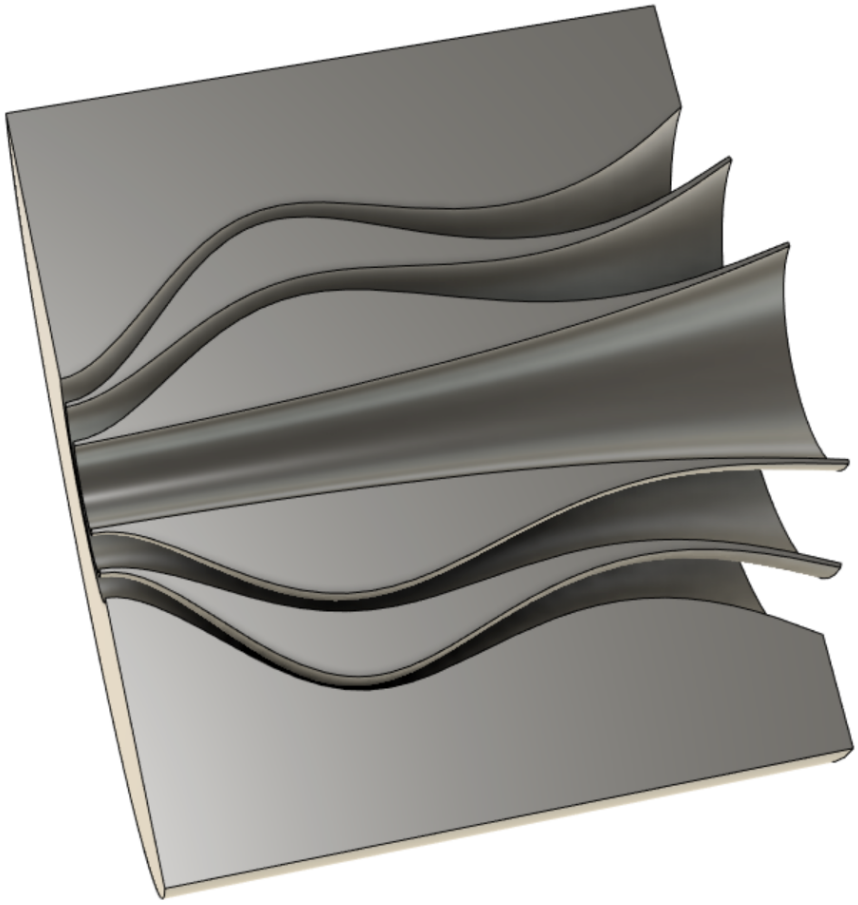


*Fig 1: ESP example (CAD visualization)*

1 https://www.at-horns.eu

## ESP design example

First, a whole ATH script code for a waveguide connected to an ESP (esp-demo.txt) is quoted below for clarity and further reference. To include an ESP in your ATH project, specify **Source.Contours = ::esp <esp_section>** in the script, where <esp_section> is the name of a script section with the ESP definition, defined earlier somewhere in the script ("plug25" in the following example).

Items of an ESP definition block will be further referred to as "ESP::<item>".

```
Throat.Diameter = 64                ; [mm]
Throat.Profile = 1                  ; 1 = OSSE
Throat.Angle = 35                   ; [deg]
Coverage.Angle = 45                 ; [deg]
OS.k = 4
Length = 67.8                       ; [mm]
Term.s = 1.0
Term.n = 3.0
Term.q = 0.996

; -------------------------- ESP related code ---------------------------

plug25 = {                          ; ESP definition
  Dt = 25.4                         ; input diameter [mm]
  At = 0                            ; input angle [deg]
  Ae = 33                           ; exit angle [deg]
  L = 75                            ; length [mm]
  Pos0 = 0.4,0.76                   ; input positions of the vanes
  Sk = 0.65                         ; meander skew
  CP1 = 0,0.4,0.4                   ; control point positions
  CP2 = 0,0.4,0.4                   ; control point positions
  EndAngle = 1                      ; minimum tip angle [deg]
  WT = 0.5                          ; minimum wall thickness [mm]
}

Source.Contours = ::esp plug25      ; source definition
; -----------------------------------------------------------------------

Rollback = 1
Rollback.StartAt = 0.4
Rollback.Angle = 180                ; [deg]
Rollback.Exp = 1.5

Mesh.AngularSegments = 8
Mesh.LengthSegments = 60
Mesh.SubdomainSlices =
Mesh.WallThickness = 4              ; [mm]

Mesh.ZMapPoints = 0.5,0.4,0.5,0.98

ABEC.SimType = 2
ABEC.SimProfile = 0
ABEC.f1 = 200                       ; [Hz]
ABEC.f2 = 20000                     ; [Hz]
ABEC.NumFrequencies = 100
ABEC.MeshFrequency = 42000          ; [Hz]

ABEC.Polars:SPL = {
  MapAngleRange = 0,180,37
}

Report = {
  Title = "ESP Demo"
  NormAngle = 10
  DrvImp_Range = 40
  MaxRadius = 150
  Width = 840
  Height = 800
  GnuplotCode = 3x2n.gpl
}

Output.STL = 0
Output.ABECProject = 1
```

## Basic design parameters

Note that the **Throat.Diameter** is set to 64 mm - this is, at the same time, the exit diameter of the ESP. This is also the only parameter that is automatically shared by the both parts of the design (WG and ESP). Opening throat angle of the waveguide (**Throat.Angle**) and the exit angle of the ESP (**ESP::Ae**) are set independently.

The quoted ESP definition example results in the following geometry, see Fig 2.

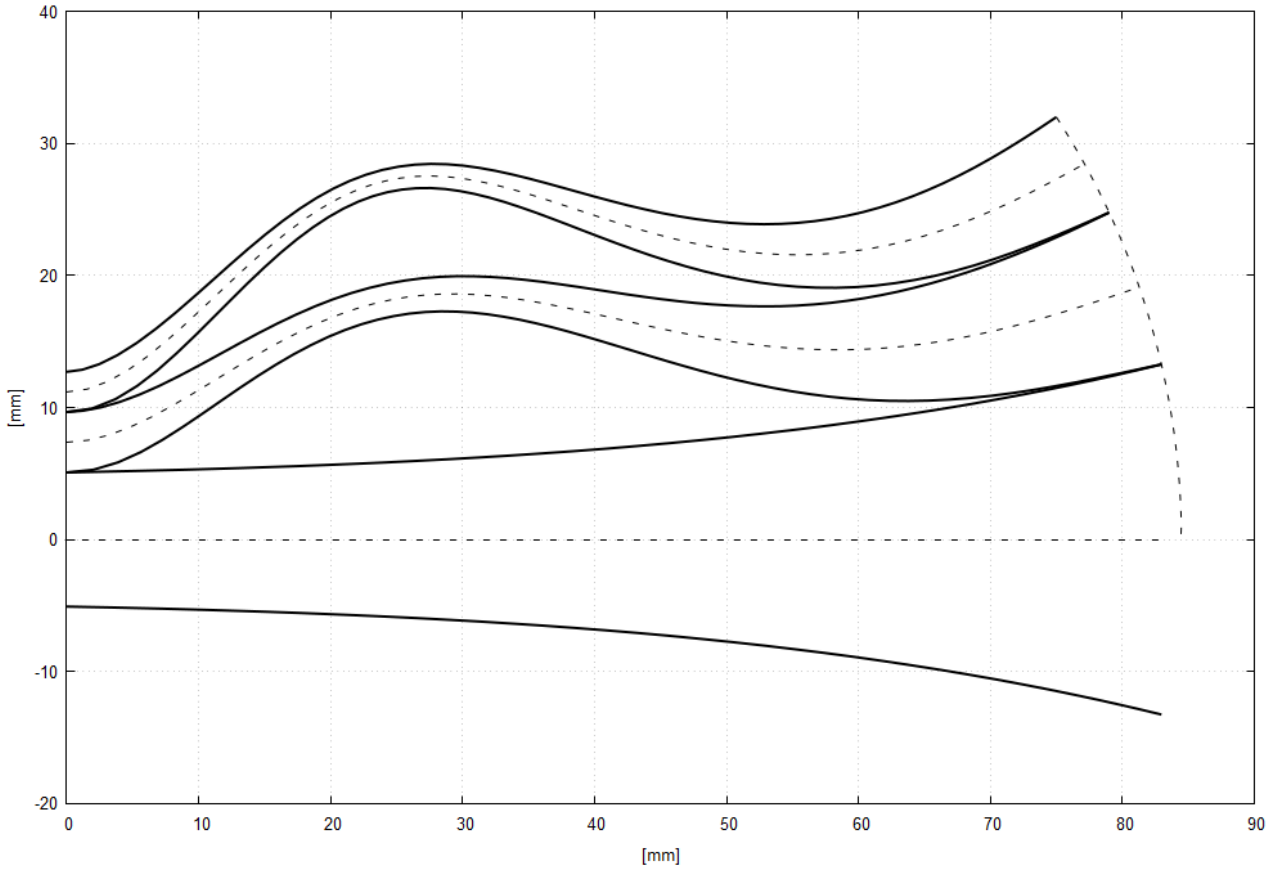



*Fig 2: ESP geometry*

There are two vanes defined, making the plug composed of three channels for sound propagation.

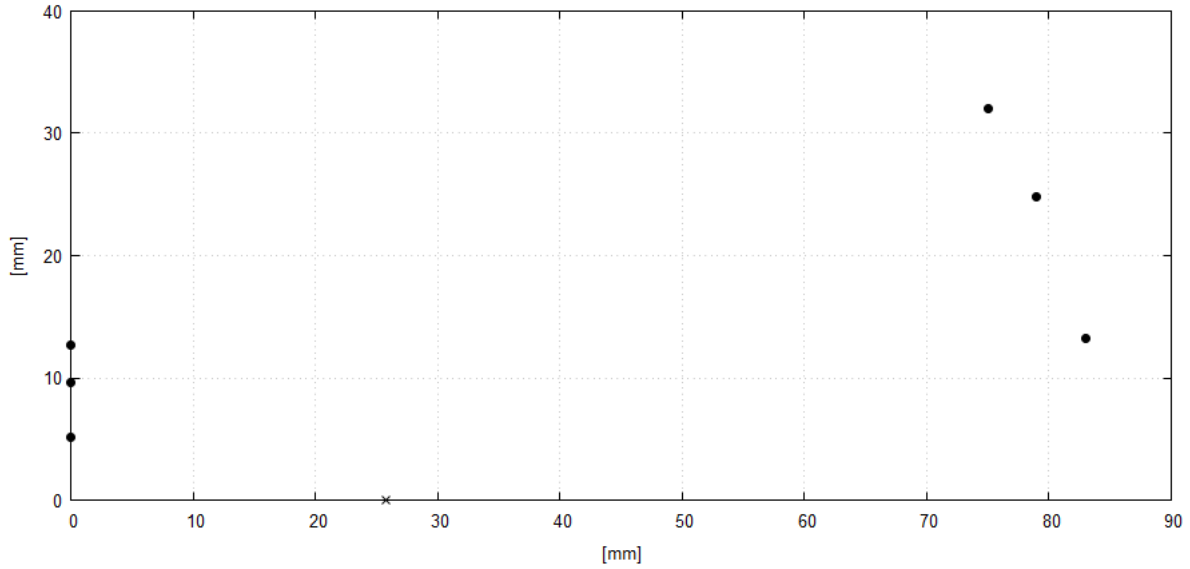
The input diameter of the ESP is specified As **ESP::Dt**, the input angle as **ESP::At.**

Length defined as **ESP::L** is the axial distance of the outmost exit point (75 mm in this case) - it's the point where the ESP connects to the horn.

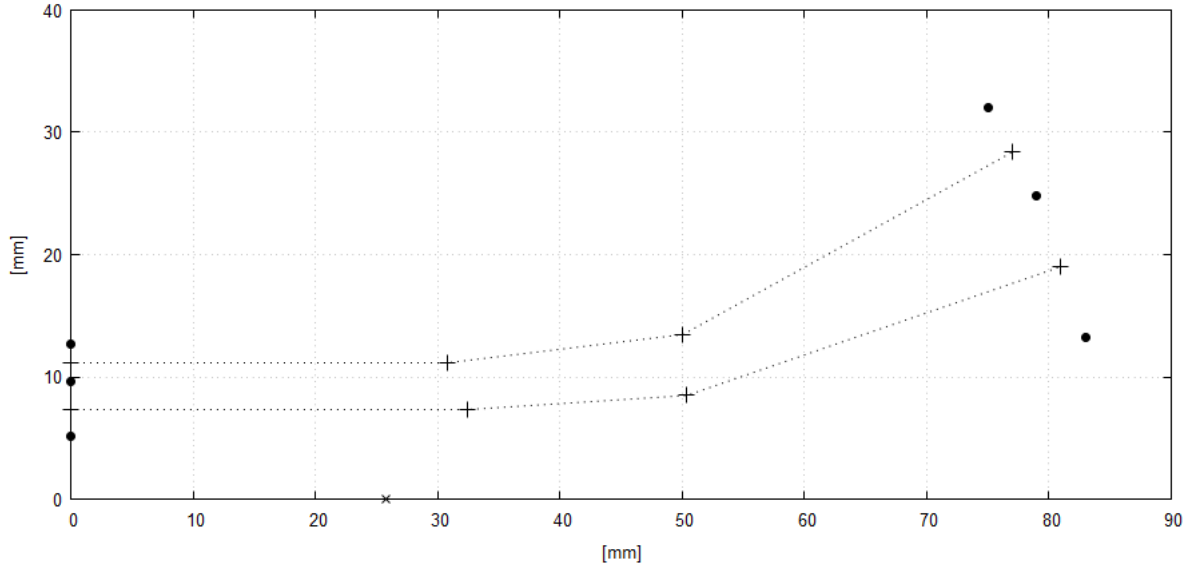
The number of vanes is defined via the array **ESP::Pos0** which defines the normalized positions of the vanes at the input (0=on-axis, 1=full throat radius). The number of values in this array sets directly the number of vanes created. By default the output positions of the vanes are calculated so that the ratios of the corresponding channel exit areas are made the same as on the input. This can be overwritten by defining an item **ESP::Pos1** with the same number of values.

## Detailed ESP construction

The starting point of a construction are the basic parameters described on the previous page. The following text gives a better understanding of the whole process of the ESP design in more detail. With the basic parameters set we have a situation from the following picture - the input/output wavefront areas and the positions of the vanes. The small "x" symbol on the horizontal axis denotes the calculated center of the output spherical wavefront.

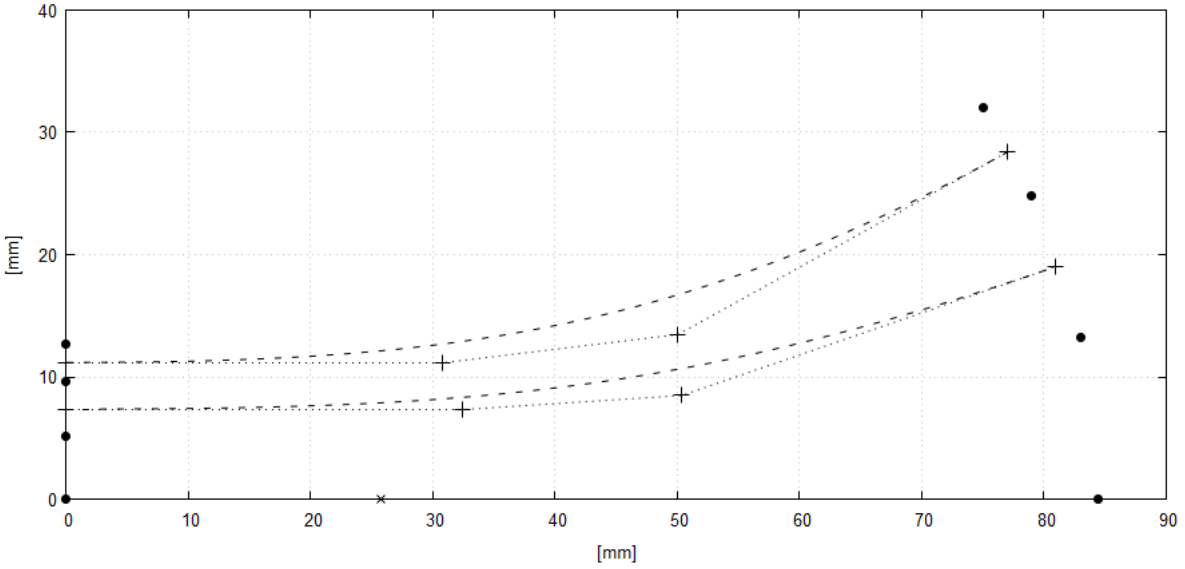



The next step is to construct channel centerlines. This is done by using cubic Bezier curves[2], each connecting corresponding mid-points between the (future) vane tips. Each curve has four control points, P0 - P3, starting at the input. The two middle control points of each curve (P1, P2) are positioned in the direction of a wave propagation at the input (P1) and at the output (P2) boundary. How much are these points shifted from the end points towards the center of the curve define the parameters **ESP::CP1** (P1) and **ESP::CP2** (P2), both having a value 0.4 in this case.




2 https://en.wikipedia.org/wiki/B%C3%A9zier_curve

The calculated channel centerlines, i.e. the Bezier curves given by the control points are shown of the following picture. Let's note that the centerline of the central (on-axis) channel is just a straight horizontal line (not shown).




The next step is to make all the channels the same length (right now the outmost channel would be the shortest, preventing a coherent acoustic sum). This is accomplished by superposing a so called *meander function*[3] *M* (Fig 3) on the channel centerline, at each point in a direction of its normal vector, effectively deviating it from the initial shape and making it longer. The exact meander amplitude for a required prolongation in each channel is calculated automatically via simple iterative methods. Parameters for the skew a sharpness can be set manually with the parameters **ESP::Sk** and **ESP::Sh**.

$$M(x) = A_M \sin(\pi x^{Sk})^{Sh}, 0 \le x \le 1$$

| | |
|---|---|
| $x$ | Normalized distance along the centerline curve |
| $A_M$ | Meander amplitude [mm] |
| $Sk$ | Meander skew (typically 0.5 - 1) |
| $Sh$ | Meander sharpness (typically 2 - 4) |

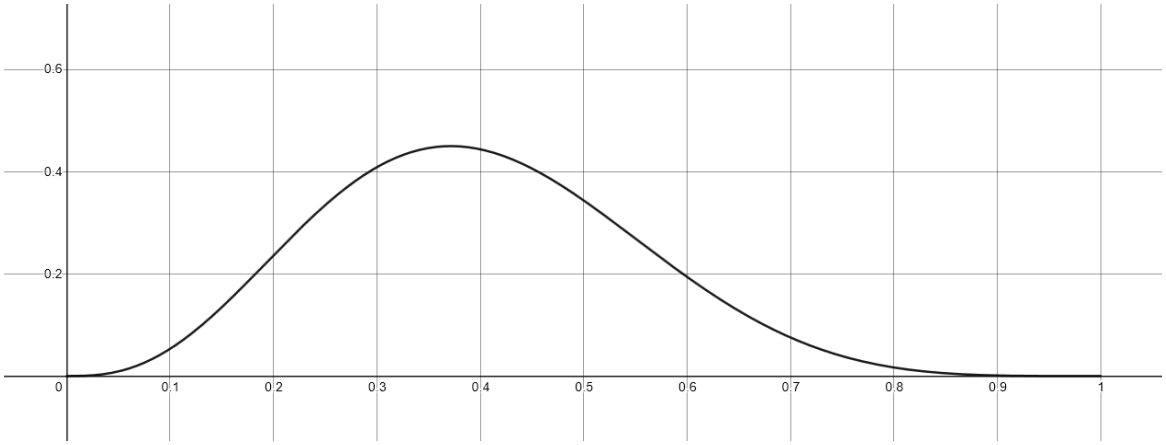
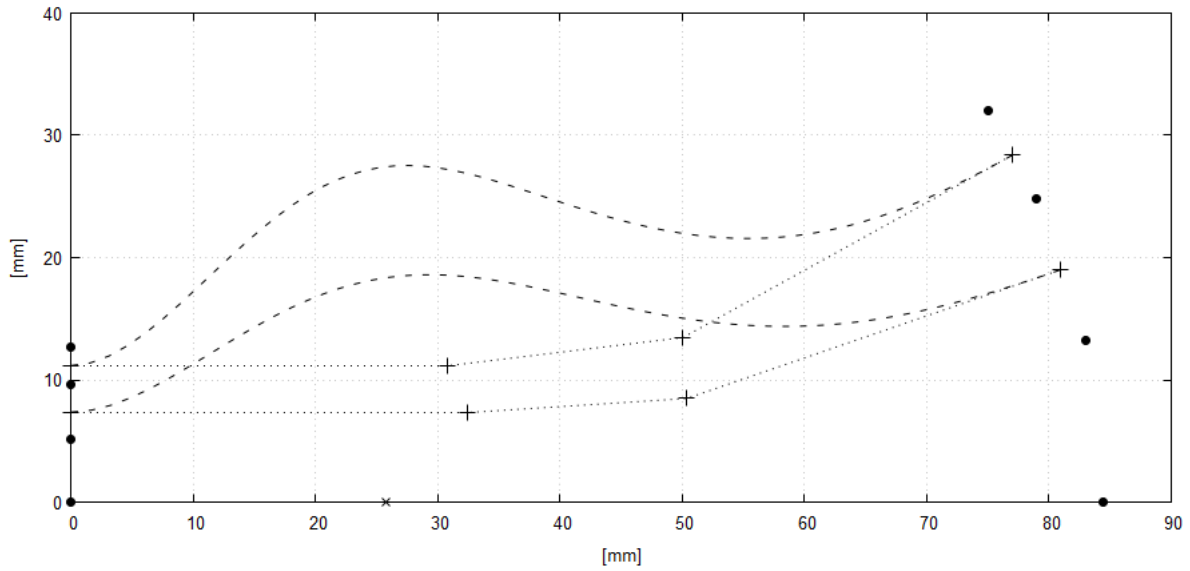



*Fig 3: Meander function example*

3 https://www.desmos.com/calculator/nck62lxx4y

Below are the centerlines after "meandering". As a result all channels now have the same length.




We already know the input and output area of each channel. The channel boundaries are now calculated so that the wavefront area inside each channel expands exponentially from the input to the output value. This gives us the resulting ESP device (Fig 4, Fig 5). By default the expansion rate is determined automatically so that the tips of the vanes are fabricable (too low expansion rate would lead to a channel overlap, something not physically possible).

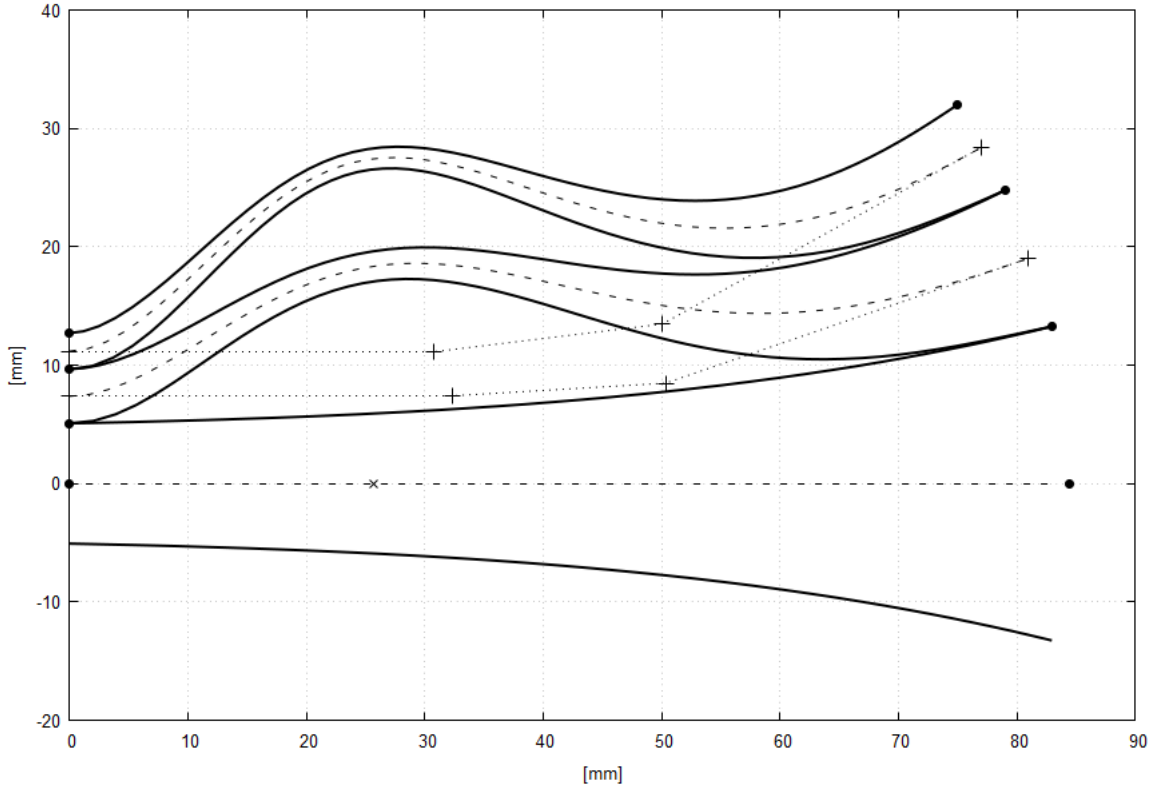



*Fig 4: Resulting ESP geometry*

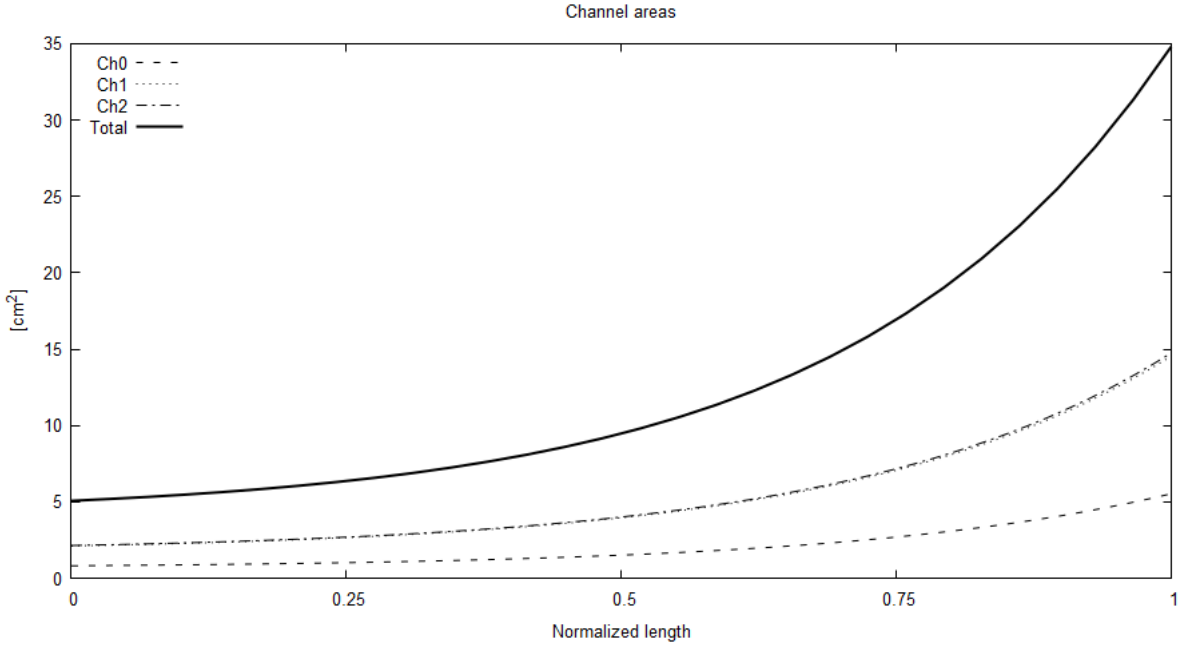



*Fig 5: Resulting ESP channel areas*

Note that in this example the areas of the two outer channels are virtually the same (which is actually more of a coincidence rathen than a careful design decision). Also note that even though the outer channels "narrow" as they bend outward, the wavefront area is still a monotonically increasing function of length, which is a result of taking into account the varying centerline radius of rotation - we always have to think in three dimensions when it comes to wavefront areas.

For a visual inspection during the design process see the files created in a subdirectory "ESP" under the ABEC project directory. The ESP subdirectory is created by default but this can be disabled by setting **ESP::Report = 0**.

## BEM simulation

There is nothing special about simulating an ESP device as it becomes an integral part of a generated ATH/ABEC project of a horn. In fact, the whole plug is internally implemented as a *Source Definition Script*, for long already a standard part of Ath.

BEM analysis results for the example described, including a small free-standing waveguide, is shown on the following pictures using VACS.

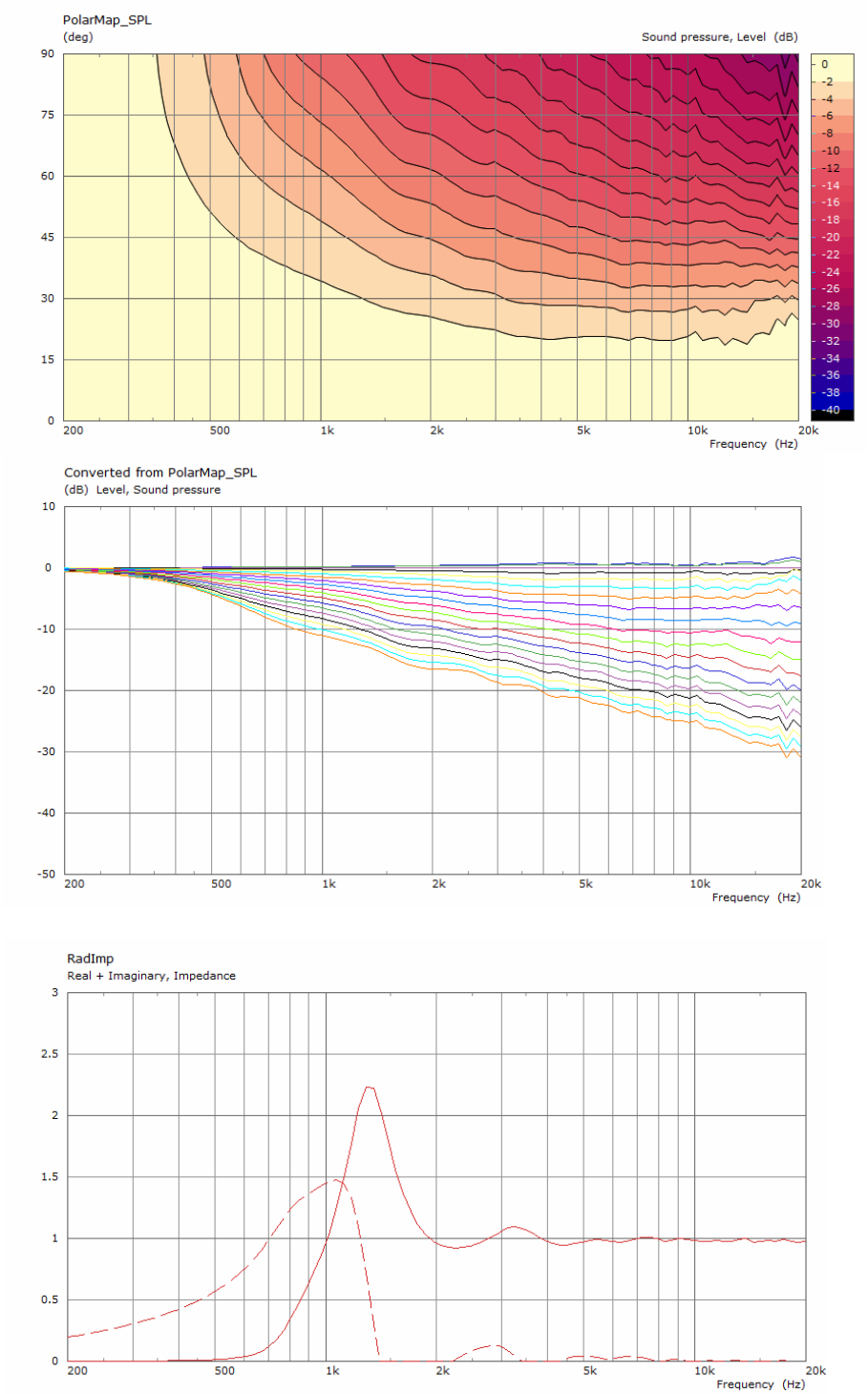
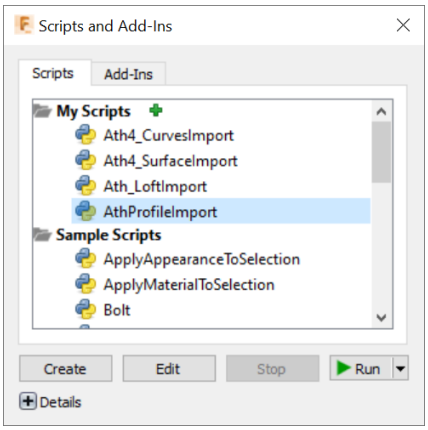



*Fig 6: BEM analysis results*

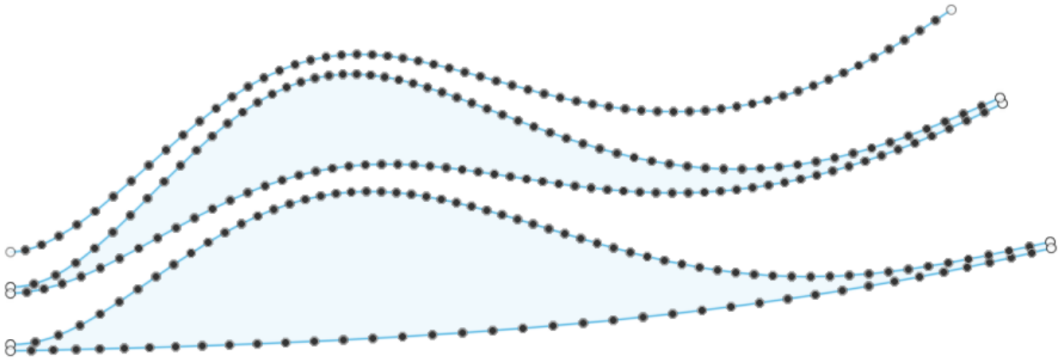
## ESP - exporting the geometry

Ath offers a way of exporting and importing of the defined geometry into Autodesk Fusion 360 via its own "AFP" file format (*.afp). By default an AFP file of the defined plug is written in the project directory as '<*esp_section*>.afp'. This can be disabled by setting **ESP::SaveAFP = 0**.

To import an AFP file into Fusion 360, install and use the supplied user script "AthProfileImport":




The script imports the ESP shape as a single sketch, using smooth splines for the vanes where applicable. Note that in this case the tips of the vanes are not sharp but terminate with a non zero width, which value is set with the parameter **ESP::WT**. This is to allow for an easier fabrication of the device.



A solid shape can be obtained from a finished sketch by revolving it around the X axis: