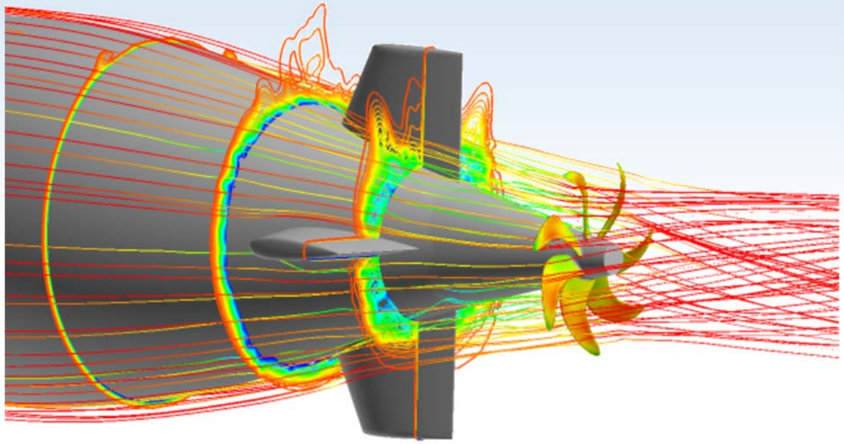
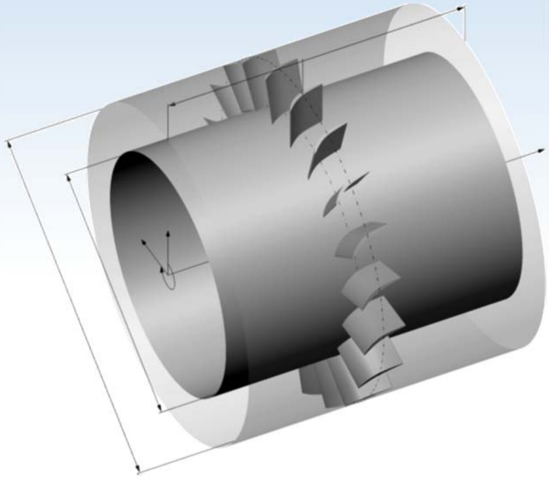
Validation of open source code BEM++ for simulation of acoustic problems

P. Lukashin (ISP RAS, Russia), S. Strijhak, (ISP RAS, Russia), G. Shcheglov (BMSTU, Russia)

Aeroacoustic sources

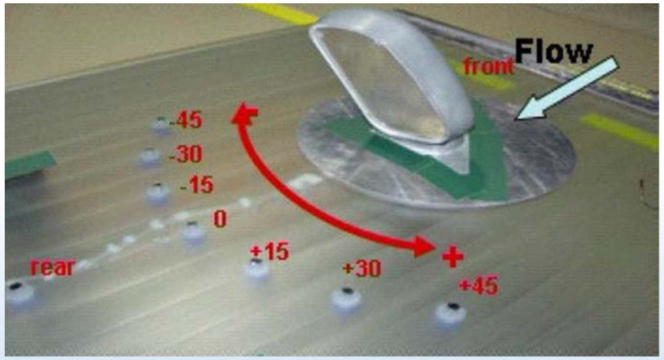


A rotating 7-bladed propeller on submarine



Compressor stator test rig

Jet's engines of Launch abort vehicle



Car's outside rear-view mirror

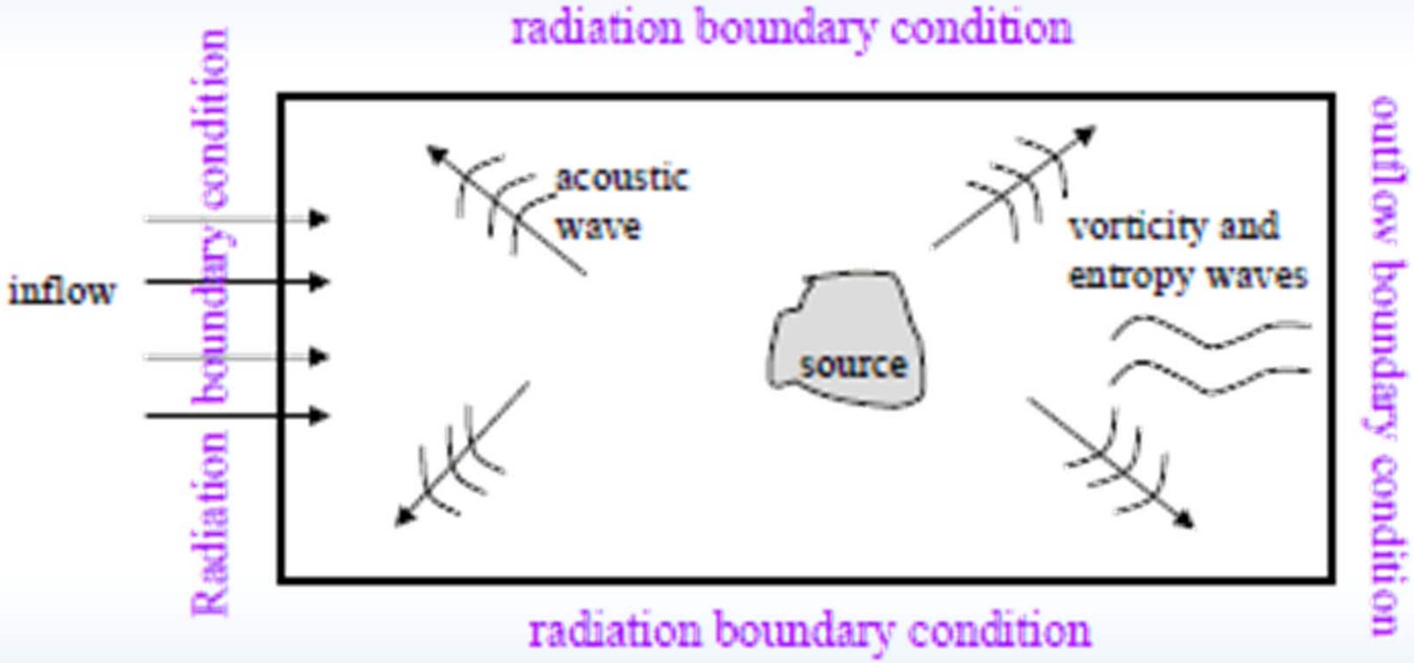
Sound-pressure levels

Table 1.1. *Sound-pressure levels for common sounds*

(dB(A))	Common sounds
30	whisper
50	rainfall, quiet office, refrigerator
60	dishwasher, normal conversation
70	traffic, vacuum cleaner, restaurant
80	alarm clock, subway, factory noise
90	electric razor, lawnmower, heavy truck or road drill at 7 m
100	garbage truck, chain saw, stereo system set above halfway mark
110	rock concert, power saw
120	jet takeoff, nightclub, thunder
130	jack hammer
140	shotgun, air raid system
180	rocket-launching pad

C. Wagner, T. Hüttl, and P. Sagaut, eds. Large-Eddy Simulation for Acoustics. Cambridge Univ. Press, New York, 2007.

Definition of acoustic's problem and boundary conditions



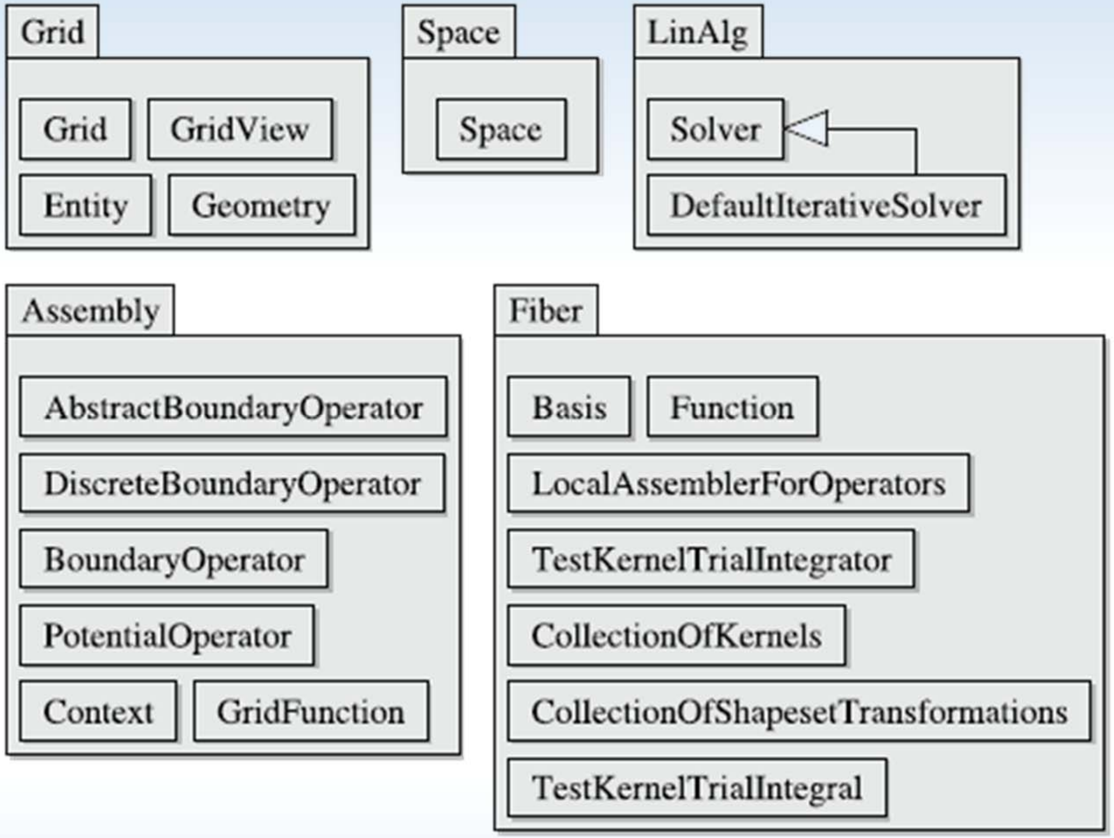
The main approaches for numerical modelling in aeroacoustics

Approach	Numerical method	Accuracy	Required resources / Applicability
Solution of «full» Navier – Stokes equations	Direct Numerical Simulation (DNS)	Fine	Very much / Impossible
Large Eddy Simulation (LES)	Good	Very much / Impossible	
Unsteady Reynolds-Averaged Navier – Stokes (URANS)	Low	Few / Possible but not reliable	
CFD-CAA CFD-ALE CFD-APE	Linearized Euler Equation (LEE) or Acoustic Perturbation Equation (APE)	Good	Needs many resources due to grid resolution for LES and acoustics. Applicable for small-scale cases.
CFD Surface Integral Analytic Methods (Lighthill, Kirchhoff, FW-H)	Hybrid methods (LES/RANS) and analytical methods for far-field modelling	Sufficient	Can not used in cases with reflection of acoustic waves somewhere in considered region
CFD-BEM	Connection of the CFD solution (LES/RANS) in the near-field and the solution of Helmholtz equation by BEM (Boundary Element Method) in the far-field	Good	Good for large cases to avoid big grids

The BEM++ Library

- Core library in C++, complete interface via Python
- Support for Laplace, Helmholtz, Maxwell
- Shared-Memory parallelization
- H-matrix assembly
- Open source license
- Mac and Linux supported

Introduction to BEM++



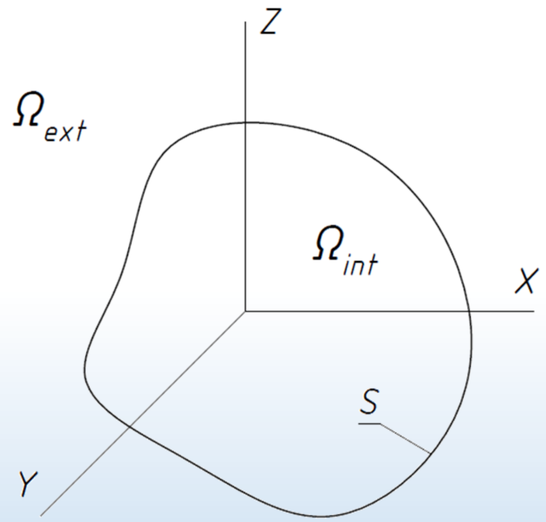
BEM++ project: www.bempp.org
University College London

Classification of acoustic problems

Acoustic wave equation: $\nabla^2 u - \frac{1}{c^2} \cdot \frac{\partial^2 u}{\partial t^2} = 0, \qquad u = f(x,t)$

$$u(x,t) = U(x) \cdot e^{-i\omega t} \quad \Rightarrow$$

Helmholtz equation: $\nabla^2 U + k^2 \cdot U = 0, \qquad k = \omega / c \; - \; wavenumber$



Boundary conditions on S:

$$\frac{\partial U}{\partial n} - i\beta U = h$$

$\beta = 0$ (Neumann BCs)

$|\beta| >> 1$ (Dirichlet BCs)

1) Radiation problem: $h \neq 0$

2) Scattering problem: $h = 0$

Formulation of the test problem

Scattering from the unit sphere of plane wave:

$$\begin{cases} \nabla^2 U + k^2 \cdot U = 0 \\ \dfrac{\partial U}{\partial n} - i\beta U = 0 \\ U = U^S + U^I \\ U^I = U_0 \cdot e^{ikx\vec{a}} \\ \lim_{R\to\infty}[R\left|\dfrac{\partial U^S}{\partial R} - ikU^S\right|] = 0 \end{cases}$$

$U|_S = 0$ (Dirichlet problem)

$\dfrac{\partial U}{\partial n}|_S = 0$ (Neumann problem)

Source data: $k, U_0, R, \vec{a}$

BEM problem

Boundary Integral equation (BIE) for

Dirichlet problem:

$$(\frac{1}{2}I + T - i\eta S)U_n^* = (\frac{\partial U^I}{\partial n} - i\eta U^I)|_S$$

Neumann problem:

$$(H + i\eta(\frac{1}{2}I - K))U^* = -(\frac{\partial U^I}{\partial n} - i\eta U^I)|_S$$

U_n^* *and* U^* are surface potentials, which need to be calculated.

$$T[U](x) = \int_S \frac{\partial G(x,y)}{\partial n(x)} \cdot U(y)dS(y) \qquad H[U](x) = \frac{\partial}{\partial n(x)} \int_S \frac{\partial G(x,y)}{\partial n(y)} \cdot U(y)dS(y)$$

$$S[U](x) = \int_S G(x,y) \cdot U(y)dS(y) \qquad K[U](x) = \int_S \frac{\partial G(x,y)}{\partial n(y)} \cdot U(y)dS(y)$$

$$I[U](x) = \int_S U(x)dS(x)$$

BEM solution

Integral equation to derive the solution for

Dirichlet problem:

$$U = U^I - S'[U_n^*]$$

Neumann problem:

$$U = U^I + K'[U^*]$$

Single-layer potential operator: $S'[U](x) := \int_S G(x,y) \cdot U(y) dS(y)$

Double-layer potential operator: $K'[U](x) := \int_S \frac{\partial G(x,y)}{\partial n(y)} U(y) dS(y)$

Green's function: $G(x,y) = \frac{e^{ik|x-y|}}{4\pi |x-y|}$

Discretization of the BIE and solving

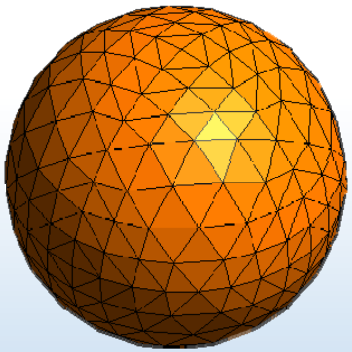
Continuous piecewise linear functions for Dirichlet data:

$$\psi_k(x) = \begin{cases} 1, & x \in \tau_k \\ 0, & x \notin \tau_k \end{cases}$$

Piecewise constant functions for Neumann data:

$$\phi_j(x_i) = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}$$

Triangulated surface with elements τ_k and nodes x_i



Solving the system by GMRES or CG

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N1} & a_{N2} & \cdots & a_{NN} \end{bmatrix} \begin{Bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_N \end{Bmatrix} = \begin{Bmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{Bmatrix}$$

Interface basics of BEM++

	Implementation in BEM++
Geometry	grid = bempp.api.shapes.regular_sphere (n)
Space function	space = bempp.api.function_space(grid, "P", 1)
BCs	gr_fun = bempp.api.GridFunction(space, fun=fun)
Identity operator	I = bempp.api.operators.boundary.sparse.identity(space, space, space)
Boundary Integral Operators	K = bempp.api.operators.boundary.helmholtz.double_layer (space, space, space, k) H = bempp.api.operators.boundary.helmholtz.hypersingular (space, space, space, k)
Potential Operators	S = bempp.api.operators.potential.helmholtz.single_layer(space, points, k) K = bempp.api.operators.potential.helmholtz.double_layer(space, points, k)
Solving the System	func,info=bempp.api.linalg.gmres (-H+1j *k*(0.5*I-K))=gr_fun)
Deriving the results	res = np.absolute(u_inc + K.evaluate(func)+1j*k*S.evaluate(b*func))

Analytical Solution

$$U = U^I + U^S$$

$$U^S = \sum_{l=0}^{n} A_l h_l(kr) P_l(\cos\theta)$$

$$U^I = U_0 \cdot e^{ikx\vec{a}} = \sum_{l=0}^{n} (2l+1) i^l j_l(kr) P_l(\cos\theta)$$

Dirichlet problem:

$$U^I + U^S = 0, \quad at \;\; r = a$$

$$A_l = -U_0 (2l+1) i^l \frac{j_l(ka)}{h_l(ka)}$$

Neumann problem:

$$\frac{\partial U^I}{\partial r} + \frac{\partial U^S}{\partial r} = 0, \quad at \;\; r = a$$

$$A_l = -U_0 (2l+1) i^l \frac{l j_{l-1}(ka) - (l+1) j_{l+1}(ka)}{l h_{l-1}(ka) - (l+1) h_{l+1}(ka)}$$

Spherical Bessel function j_l

Legendre function P_l

Spherical Hankel function h_l

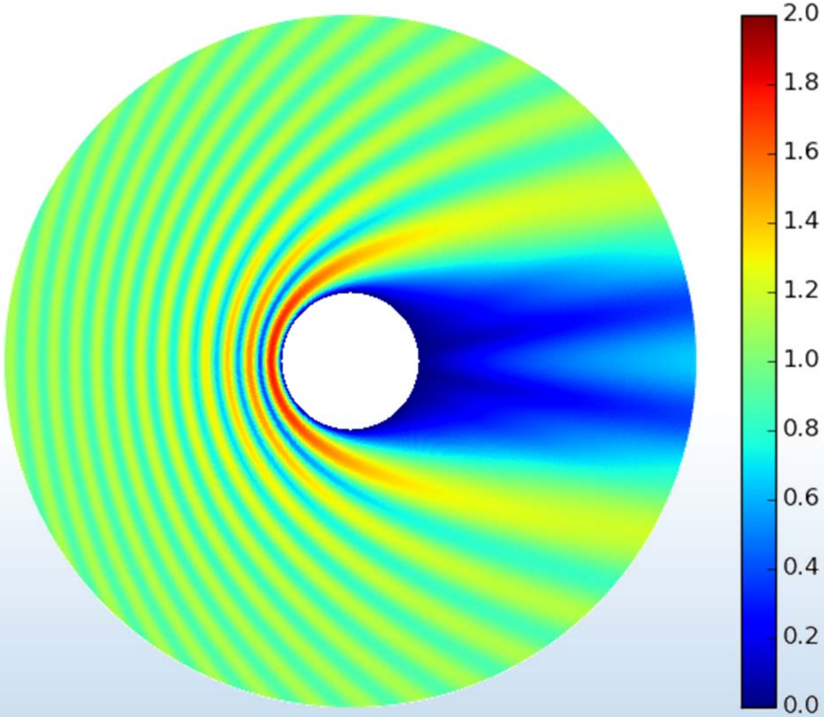
Compare the results for Dirichlet problem

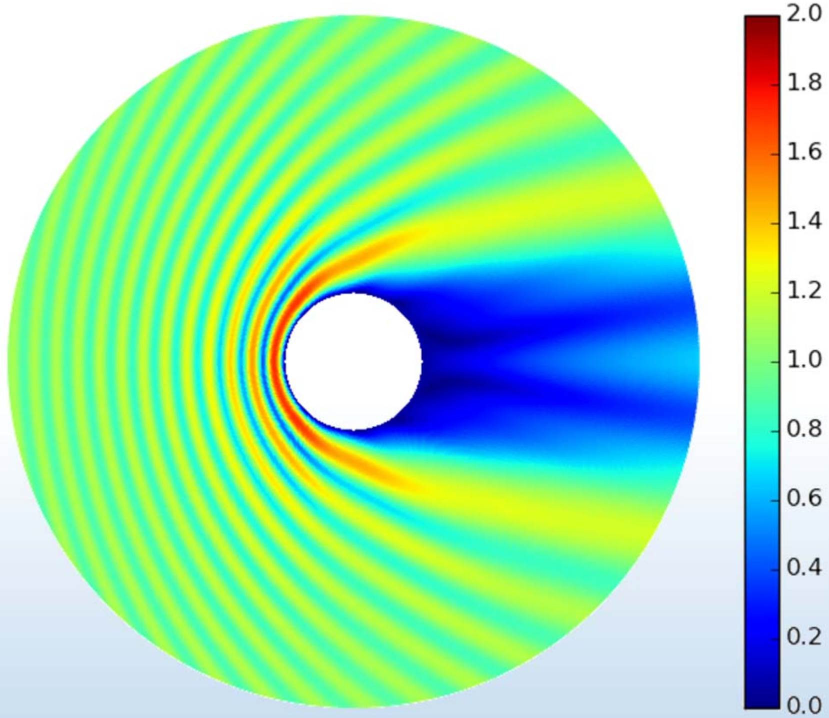
$$k = 10, U_0 = 1, R = 1, \vec{a} = \{1;0;0\}$$

BEM solution

Analytical solution

OXY plane





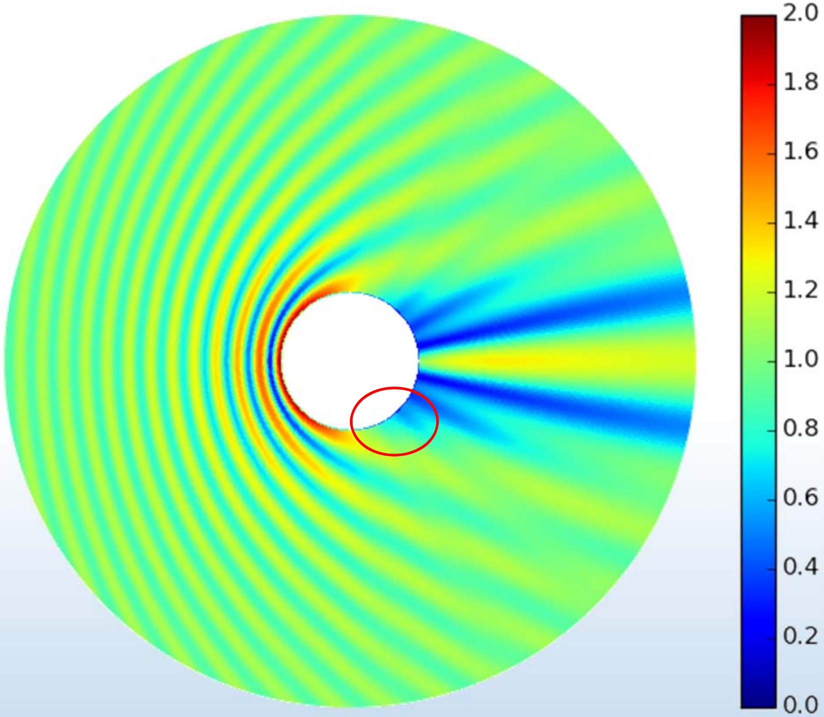
Compare the results for Neumann problem

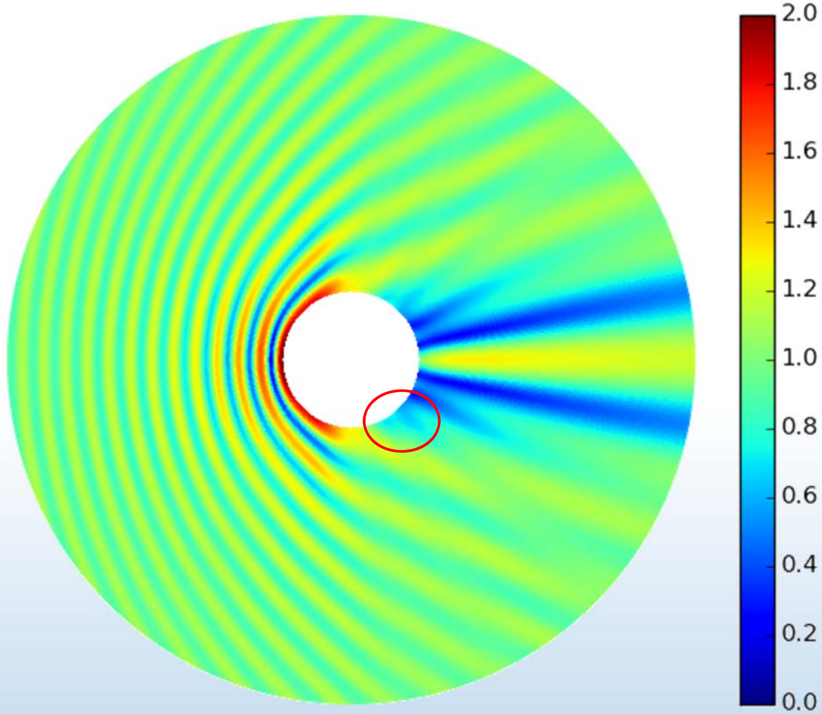
$$k = 10, U_0 = 1, R = 1, \vec{a} = \{1;0;0\}$$

BEM solution

Analytical solution

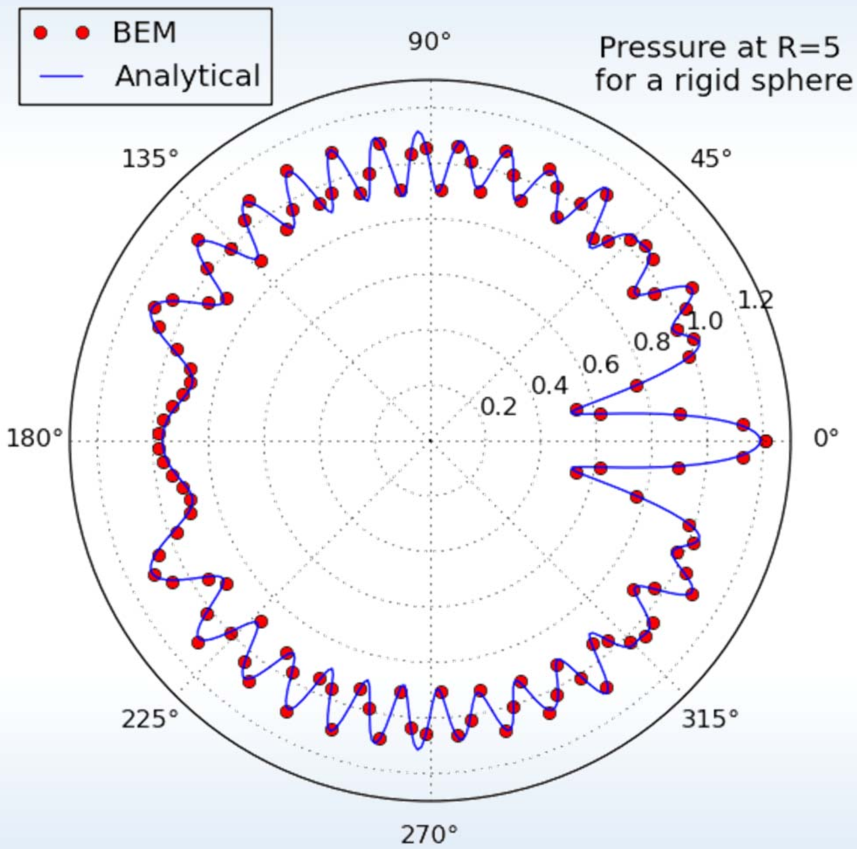
OXY plane

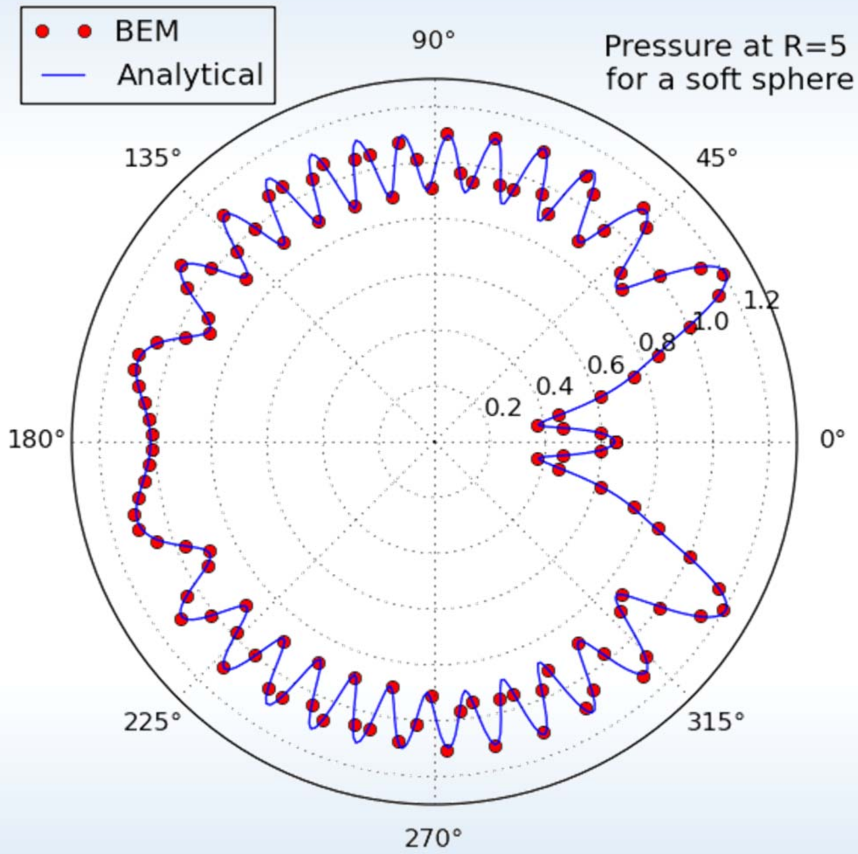




Polar plots

$$k = 10, U_0 = 1, R = 1, \vec{a} = \{1;0;0\}$$

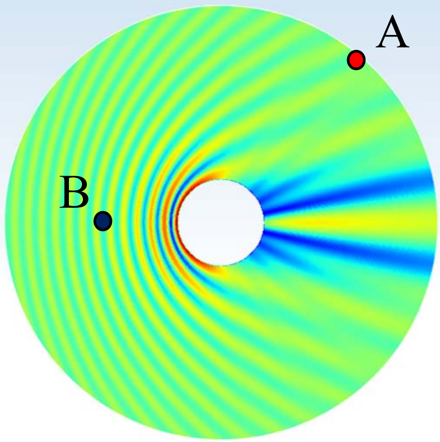




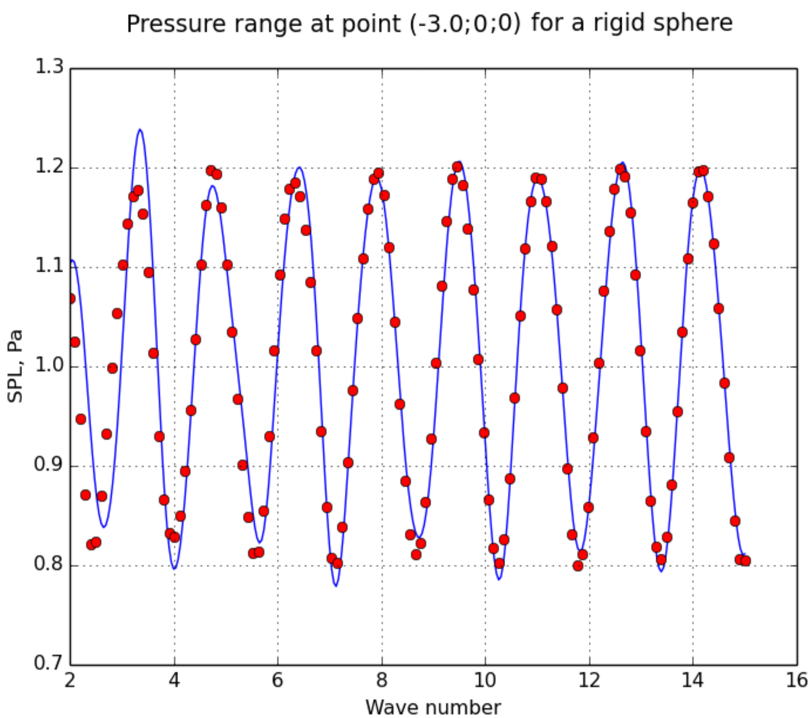
OXY plane

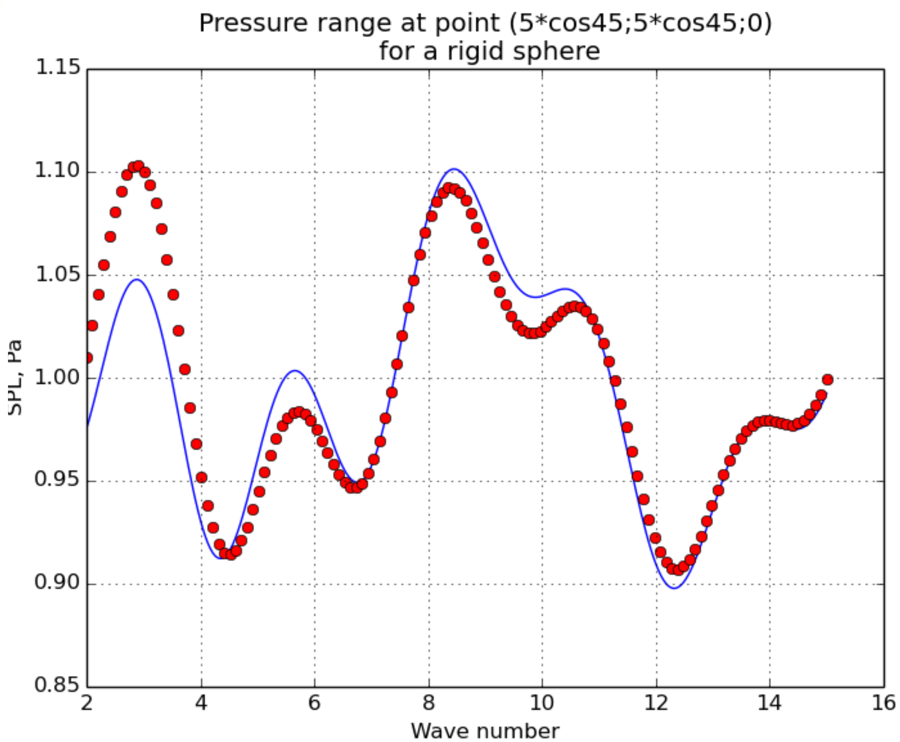
Pressure range for a rigid sphere

Point B



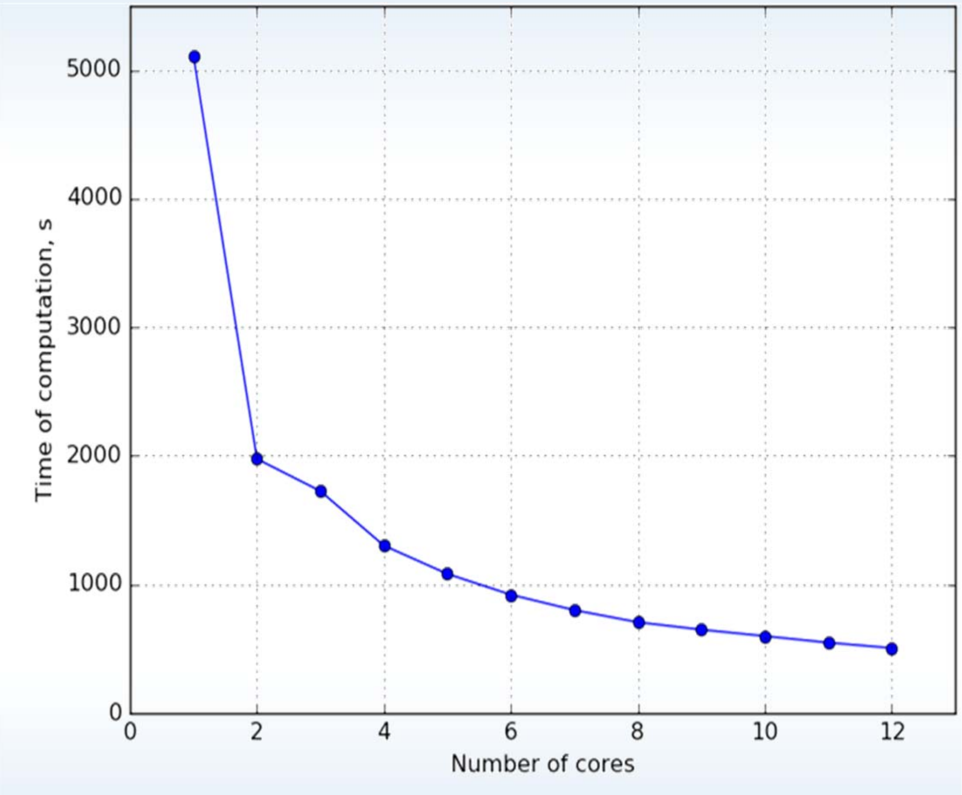
Point A





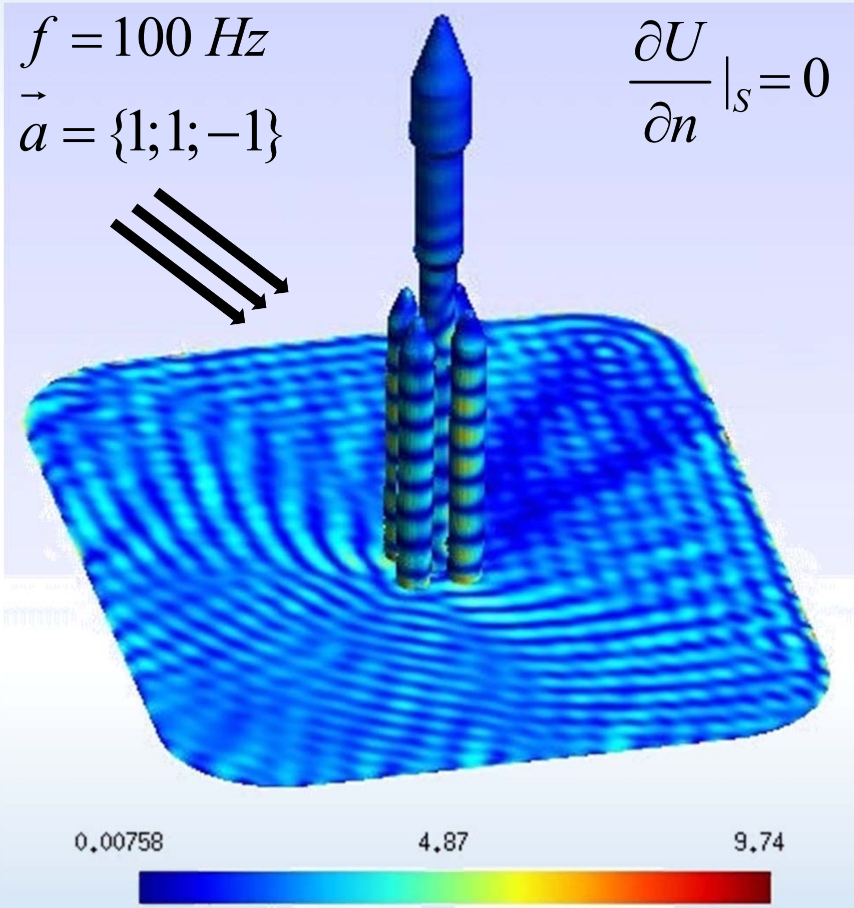
Simulation for prototype of Launch Vehicle

CPU Time



Mesh size: 56500 elements

Pressure plot



Conclusions

- Successful validation of the BEM++ open source library on test case «Sphere»
- The pressure error is less than 5 % compared to analytical solution
- Shared memory technology restricts efficiency of solver: sufficiently time of computation for industrials problems with large meshes is possible for low frequencies (~100 Hz)