Finite Elements in Acoustics

Simulation Methods in Acoustics

The Helmholtz problem

- Problem: we seek the acoustic pressure $p(\mathbf{x})$ in a domain Ω
- The Helmholtz BVP in linear acoustics

$$
\begin{array}{llll}
\text{PDE:} & \nabla^2 p(\mathbf{x}) + k^2 p(\mathbf{x}) = 0 & \mathbf{x} \in \Omega \subset \mathbb{R}^d & \\
\text{BCs:} & p(\mathbf{x}) = \bar{p}(\mathbf{x}) & \mathbf{x} \in \Gamma_p & \text{(Dirichlet)} \\
 & v_n = \bar{v}(\mathbf{x}) & \mathbf{x} \in \Gamma_v & \text{(Neumann)} \\
 & z = \bar{z}(\mathbf{x}) & \mathbf{x} \in \Gamma_z & \text{(Robin)}
\end{array}
$$

- If the BCs are given such that $\Gamma_p \cup \Gamma_v \cup \Gamma_z = \Gamma$, we have a well-posed problem.
- Thus, if we know p, v, or z on the whole boundary Γ, the pressure $p(\mathbf{x})$ for the whole domain Ω can be computed! The solution is unique.[1]

[1] except when there are undamped modes

The weak form in multiple dimensions

- Test function $\psi(\mathbf{x})$ and integration over the domain Ω

$$\int_\Omega \psi \nabla^2 p \, \mathrm{d}\mathbf{x} + k^2 \int_\Omega \psi p \, \mathrm{d}\mathbf{x} = 0$$

- Partial integration
  - Use $\nabla \cdot (f\mathbf{g}) = (\nabla f) \cdot \mathbf{g} + f \nabla \cdot \mathbf{g}$ with $f = \psi$ and $\mathbf{g} = \nabla p$
  - We have $\psi \nabla^2 p = \nabla \cdot (\psi \nabla p) - \nabla \psi \cdot \nabla p$

$$\int_\Omega \nabla \psi \cdot \nabla p \, \mathrm{d}\mathbf{x} - k^2 \int_\Omega \psi p \, \mathrm{d}\mathbf{x} = \int_\Omega \nabla \cdot (\psi \nabla p) \, \mathrm{d}\mathbf{x}$$

- Use Gauss (divergence) theorem on the right hand side
  - $\int_\Omega \nabla \cdot \mathbf{f} \, \mathrm{d}\mathbf{x} = \int_\Gamma \mathbf{f} \cdot \mathbf{n} \, \mathrm{d}\mathbf{x}$ ($\mathbf{n}$ is outward normal)
  - Here, $\mathbf{f} = \psi \nabla p$

$$\int_\Omega \nabla \psi \cdot \nabla p \, \mathrm{d}\mathbf{x} - k^2 \int_\Omega \psi p \, \mathrm{d}\mathbf{x} = \int_\Gamma \psi \underbrace{\nabla p \cdot \mathbf{n}}_{\frac{\partial p}{\partial n}} \, \mathrm{d}\mathbf{x}$$

Apply boundary conditions

- Make use of the Euler equation for the Neumann BC

$$\nabla p + \mathrm{j}\omega\rho_0\mathbf{v} = \mathbf{0} \quad \text{multiply by } \mathbf{n}$$
$$\frac{\partial p}{\partial n} + \mathrm{j}\omega\rho_0 v_n = 0$$

- We get the final shape of the weak form

$$\rho_0 c^2 \int_\Omega \nabla\psi \cdot \nabla p \,\mathrm{d}\mathbf{x} - \omega^2\rho_0 \int_\Omega \psi p \,\mathrm{d}\mathbf{x} = -\mathrm{j}\omega\rho_0^2 c^2 \int_\Gamma \psi v_n \,\mathrm{d}\mathbf{x}$$

- Notice that
  1. Instead of simple evaluation of v_n we have a boundary integral
  2. Everything else is the same as in 1D
- The similarity is greater between 2D–3D than between 1D–2D

Function space discretization

- Choose discrete function spaces for representing
  1. Pressure: $p(\mathbf{x}) \approx \sum N_i^{(p)}(\mathbf{x}) p_i$
  2. Normal velocity: $v_n(\mathbf{x}) \approx \sum N_j^{(v)}(\mathbf{x}) v_j$
  3. Test function: $\psi(\mathbf{x}) \approx \sum w_k \psi_k(\mathbf{x})$
     Galerkin method: $\psi_i(\mathbf{x}) = N_i^{(p)}(\mathbf{x})$
- Shape functions $N_i^{(p)}(\mathbf{x})$ and $N_j^{(v)}(\mathbf{x})$ can / must be different
- The finite dimensional approximation of the velocity function $v_n(\mathbf{x})$ is needed in the basis $\mathcal{V} = \left\{N_1^{(v)}, N_2^{(v)}, \ldots, N_m^{(v)}\right\}$
- In the weak form, on the r.h.s. we have

$$-\mathrm{j}\omega\rho_0^2 c^2 \int_\Gamma \psi v_n \,\mathrm{d}\mathbf{x} = -\mathrm{j}\omega \mathbf{w}^{\mathrm{T}} \underbrace{\rho_0^2 c^2 \int_\Gamma \boldsymbol{\psi}^{\mathrm{T}} \mathbf{N}^{(v)} \,\mathrm{d}\mathbf{x}}_{\text{Boundary mass mat.: } \mathbf{A}} \mathbf{v}$$

- Thus, the weak matrix form is written as

$$\left(\mathbf{K} - \omega^2 \mathbf{M}\right)\mathbf{p} = -\mathrm{j}\omega \mathbf{A}\mathbf{v}$$

Geometrical discretization and the FE model

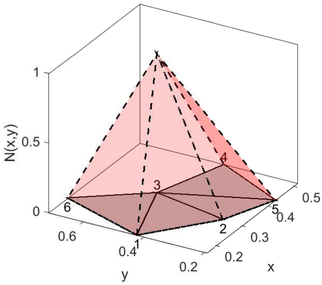
- The same approach as in 1D to assemble the matrices:
  1. Local interpolation functions (non-zero over a few elements)
  2. Integrate over elements to get small matrices ($\mathbf{K}_e$, $\mathbf{M}_e$, $\mathbf{A}_e$)
  3. Fill and add element matrices in the system matrices
- Practical storage: **node** locations + **element** connections

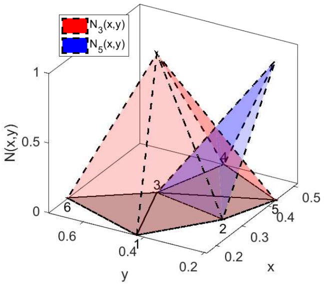
$$\texttt{nodes} = \begin{bmatrix} 1 & x_1 & y_1 & z_1 \\ 2 & x_2 & y_2 & z_2 \\ \vdots & \vdots & \vdots & \vdots \\ n & x_n & y_n & z_n \end{bmatrix} \quad \texttt{elems} = \begin{bmatrix} 1 & 3 & e_{11} & e_{12} & e_{13} & 0 \\ 2 & 3 & e_{21} & e_{22} & e_{23} & 0 \\ 3 & 4 & e_{31} & e_{32} & e_{33} & e_{34} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ m & 4 & e_{m1} & e_{m2} & e_{m3} & e_{m4} \end{bmatrix}$$

- The finite element model consists of the geometry, **material** descriptions and other **properties**. The latter two are most often associated to elements.

Elements in 2D

- In a 2D configuration we usually have the following elements
  1. Triangular elements in Ω ("tria")
  2. Quadrangular elements in Ω ("quad")
  3. Linear elements in Γ ("line")
- Other elements (such as hexagons, e.g.) are possible, but not used in practice (integration over the parent element ...)
- Example linear shape functions





Integration over elements

- Recall that we define the local coordinate systems for each element $\boldsymbol{\xi} = [\xi, \eta, \zeta]$
- Shape functions are defined and integration is performed in these local systems

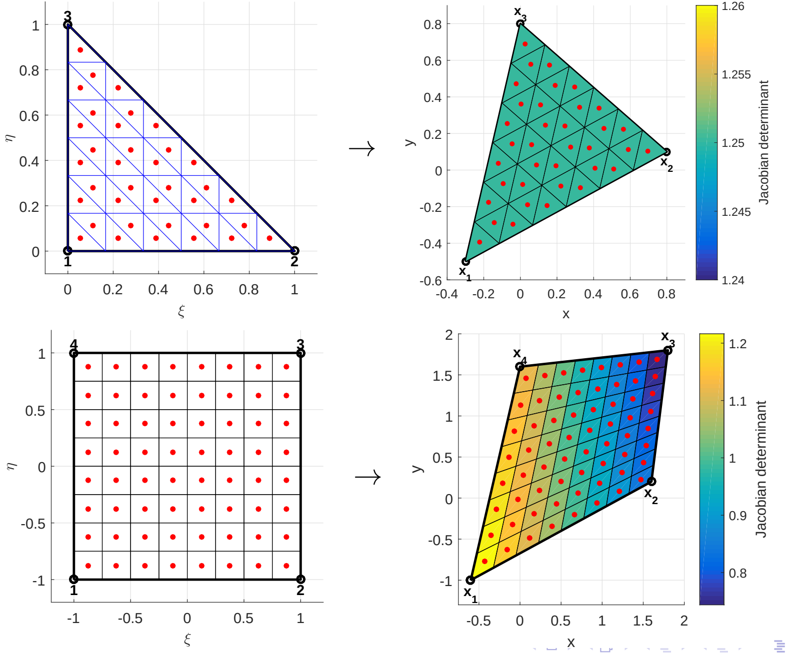
$$\int_{\Omega_e} \dots \,\mathrm{d}\mathbf{x} = \int_{\Omega_e} \dots \,\det\left[\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}\boldsymbol{\xi}}\right] \mathrm{d}\boldsymbol{\xi} \qquad \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}\boldsymbol{\xi}} = \mathbf{J} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} \end{bmatrix}$$

- Recall that $\mathbf{x}(\boldsymbol{\xi})$ is defined by means of the geometrical shape functions (a.k.a. mapping functions) $\mathbf{L}(\boldsymbol{\xi})$

$$\mathbf{x}(\boldsymbol{\xi}) = \sum_{i=1}^{n} L_i(\boldsymbol{\xi})\mathbf{x}_i = \mathbf{L}(\boldsymbol{\xi})\mathbf{X} \qquad \mathbf{X} = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \vdots & \vdots \\ x_n & y_n \end{bmatrix}$$

- n is the number of corner nodes of the element
- $\det \mathbf{J}$ is the ratio of surface areas that can change over the element, numerical integration is necessary most of the time

Jacobian determinant over the element



The element stiffness matrix

- In $\mathbf{K}_e$ we need

$$\int_{\Omega_e} \left(\frac{\mathrm{d}\mathbf{N}(\mathbf{x}(\boldsymbol{\xi}))}{\mathrm{d}\mathbf{x}} \right)^{\mathrm{T}} \frac{\mathrm{d}\mathbf{N}(\mathbf{x}(\boldsymbol{\xi}))}{\mathrm{d}\mathbf{x}} \, |\mathbf{J}| \, \mathrm{d}\boldsymbol{\xi}$$

- Notation:

$$\nabla_{\mathbf{x}}\mathbf{N} = \frac{\mathrm{d}\mathbf{N}}{\mathrm{d}\mathbf{x}} = \begin{bmatrix} \frac{\partial N_1}{\partial x} & \frac{\partial N_2}{\partial x} & \frac{\partial N_3}{\partial x} \\ \frac{\partial N_1}{\partial y} & \frac{\partial N_2}{\partial y} & \frac{\partial N_3}{\partial y} \end{bmatrix} \quad \nabla_{\boldsymbol{\xi}}\mathbf{N} = \frac{\mathrm{d}\mathbf{N}}{\mathrm{d}\boldsymbol{\xi}} = \begin{bmatrix} \frac{\partial N_1}{\partial \xi} & \frac{\partial N_2}{\partial \xi} & \frac{\partial N_3}{\partial \xi} \\ \frac{\partial N_1}{\partial \eta} & \frac{\partial N_2}{\partial \eta} & \frac{\partial N_3}{\partial \eta} \end{bmatrix}$$

- Use the chain rule $\frac{\partial N_1}{\partial \xi} = \frac{\partial N_1}{\partial x}\frac{\partial x}{\partial \xi} + \frac{\partial N_1}{\partial y}\frac{\partial y}{\partial \xi}$ etc. to get

$$\nabla_{\boldsymbol{\xi}}\mathbf{N} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix} \nabla_{\mathbf{x}}\mathbf{N} = \mathbf{J}^{\mathrm{T}}\nabla_{\mathbf{x}}\mathbf{N} \quad \rightarrow \quad \nabla_{\mathbf{x}}\mathbf{N} = \mathbf{J}^{-\mathrm{T}}\nabla_{\boldsymbol{\xi}}\mathbf{N}$$

- Finally, substituting into the original integral

$$\begin{aligned} \int_{\Omega_e} \nabla_{\mathbf{x}}\mathbf{N}^{\mathrm{T}}\nabla_{\mathbf{x}}\mathbf{N} \, |\mathbf{J}| \, \mathrm{d}\boldsymbol{\xi} &= \int_{\Omega_e} \left(\mathbf{J}^{-\mathrm{T}}\nabla_{\boldsymbol{\xi}}\mathbf{N}\right)^{\mathrm{T}} \mathbf{J}^{-\mathrm{T}}\nabla_{\boldsymbol{\xi}}\mathbf{N} \, |\mathbf{J}| \, \mathrm{d}\boldsymbol{\xi} \\ &= \int_{\Omega_e} \nabla_{\boldsymbol{\xi}}\mathbf{N}^{\mathrm{T}}\mathbf{J}^{-1}\mathbf{J}^{-\mathrm{T}}\nabla_{\boldsymbol{\xi}}\mathbf{N} \, |\mathbf{J}| \, \mathrm{d}\boldsymbol{\xi} \end{aligned}$$

Surface elements

- On the boundary Γ we map from a local coordinate system of $d-1$ dimensions (1D→2D or 2D→3D)
- Example: line element on a 2D boundary

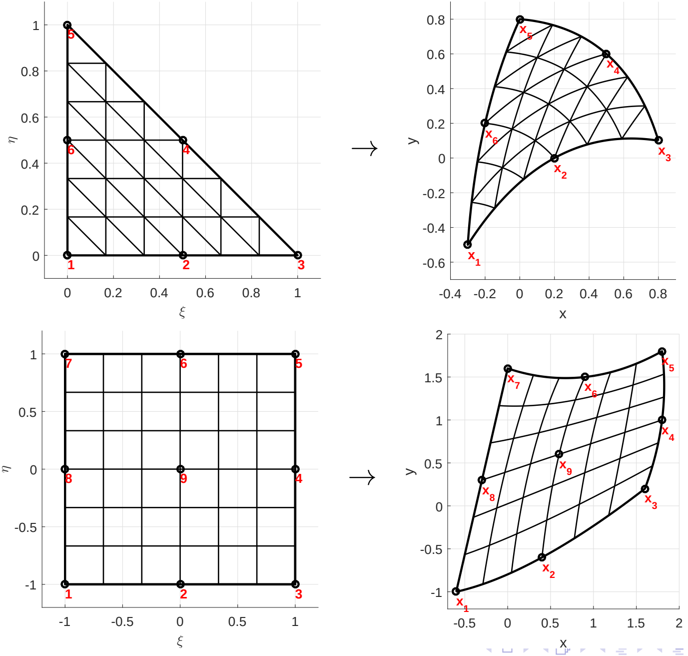
$$\mathbf{x}(\xi) = \begin{bmatrix} \frac{1+\xi}{2} & \frac{1-\xi}{2} \end{bmatrix} \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \end{bmatrix} \qquad \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}\xi} = \frac{1}{2} \begin{bmatrix} x_2 - x_1 \\ y_2 - y_1 \end{bmatrix}$$

- Generalized Jacobian (with unit vectors **i**, **j**, **k**)

$$\text{2D: } \mathbf{J} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \mathbf{i} \\ \frac{\partial y}{\partial \xi} & \mathbf{j} \end{bmatrix} \qquad \text{3D: } \mathbf{J} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} & \mathbf{i} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} & \mathbf{j} \\ \frac{\partial z}{\partial \xi} & \frac{\partial z}{\partial \eta} & \mathbf{k} \end{bmatrix}$$

- The Jacobian determinant is a vector in this case
  1. Points toward the surface normal
  2. Its length $|\mathbf{J}|$ is the ratio of the areas

Elements with higher polynomial order



Modal solution

- Same as in the case of concentrated parameter systems

$$(\mathbf{K} - \omega^2 \mathbf{M})\phi = \mathbf{0}$$

- We do not need and cannot compute all the modes
  - Choose the ones with the lowest eigenfrequencies (use Rubin)
- Write the system in a modal basis

$$\mathbf{\Phi}^{\mathrm{T}} \left(\mathbf{K} - \omega^2 \mathbf{M}\right) \mathbf{\Phi q} = -\mathrm{j}\omega \mathbf{\Phi}^{\mathrm{T}} \mathbf{A v} \qquad \text{with} \quad \mathbf{p} = \mathbf{\Phi q}$$

$$\left(\mathbf{\Lambda} - \omega^2 \mathbf{I}\right) \mathbf{q} = -\mathrm{j}\omega \mathbf{\Phi}^{\mathrm{T}} \mathbf{A v}$$

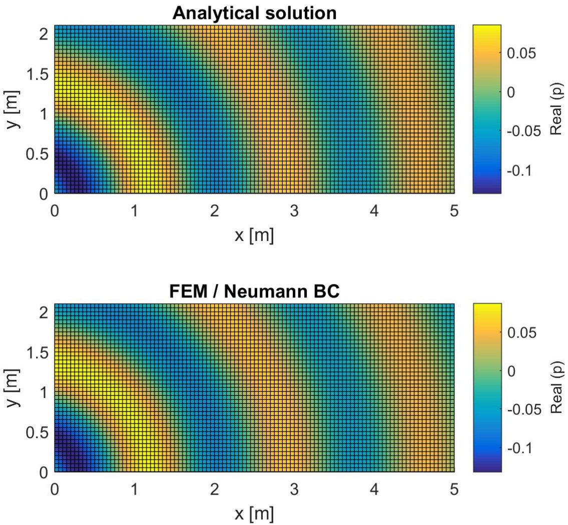
- Diagonalization to get the modal coordinates q_i easily

$$q_i = \frac{-\mathrm{j}\omega \phi_i^{\mathrm{T}} \mathbf{A v}}{\omega_i^2 - \omega^2}$$

A test problem

- *Transparent problems* are good for testing numerical techniques. In a transparent problem, we apply the field of a simple source (such as point source) on the boundary of the domain as BCs (Dirichlet or Neumann), and compute the field inside the domain using the given BCs.
- The resulting field should match the known field of the simple source in all points of the domain. This is true independent of the shape and elements of the mesh, thus, the problem is called *transparent*.
- In the following example, the field of a line source is forced on the boundary of a rectangular domain, prescribing normal velocity on the boundaries. (Neumann BC.) The line source is located outside the domain.

A test problem – Results



- We compute the relative error of the FEM solution compared to the analytical one. (The relative error is $\approx 2\%$ here.)

Solution in the time domain I.

- The matrix equation is transformed into the time domain

$$\mathbf{K}\mathbf{p} + \mathbf{M}\ddot{\mathbf{p}} = -\mathbf{A}\dot{\mathbf{v}}$$

- We already know time stepping methods to solve this! (Forward & backward Euler, Runge–Kutta etc.)
- Initial conditions are required: $\mathbf{p}(t=0) = \mathbf{p}_0$, $\dot{\mathbf{p}}(t=0) = \mathbf{g}_0$
- The Newmark scheme is often used on the form:

$$\mathbf{M}\ddot{\mathbf{p}}_{n+1} + \mathbf{K}\mathbf{p}_{n+1} = \mathbf{f}_{n+1}$$

- Introduce the semi-implicit approximations

$$\mathbf{p}_{n+1} = \mathbf{p}_n + \Delta t\dot{\mathbf{p}}_n + \frac{\Delta t^2}{2}\left[(1-2\beta)\ddot{\mathbf{p}}_n + 2\beta\ddot{\mathbf{p}}_{n+1}\right]$$

$$\dot{\mathbf{p}}_{n+1} = \dot{\mathbf{p}}_n + \Delta t\left[(1-\gamma)\ddot{\mathbf{p}}_n + \gamma\ddot{\mathbf{p}}_{n+1}\right]$$

Solution in the time domain II.

- Solve using a predictor–corrector procedure
  1. Predictor step

$$\tilde{\mathbf{p}} = \mathbf{p}_n + \Delta t \dot{\mathbf{p}}_n + (1 - 2\beta)\frac{\Delta t^2}{2}\ddot{\mathbf{p}}_n$$

$$\tilde{\dot{\mathbf{p}}} = \dot{\mathbf{p}}_n + \Delta t (1 - \gamma)\ddot{\mathbf{p}}_n$$

  2. Solution step (solve algebraic system)

$$\left(\mathbf{M} + \beta \Delta t^2 \mathbf{K}\right) \ddot{\mathbf{p}}_{n+1} = \mathbf{f}_{n+1} - \mathbf{K}\tilde{\mathbf{p}}$$

  3. Corrector step

$$\mathbf{p}_{n+1} = \tilde{\mathbf{p}} + \beta \Delta t^2 \ddot{\mathbf{p}}_{n+1}$$

$$\dot{\mathbf{p}}_{n+1} = \tilde{\dot{\mathbf{p}}} + \gamma \Delta t \ddot{\mathbf{p}}_{n+1}$$

- Common choice of parameters: $\beta = 1/4$, $\gamma = 1/2$

Time domain example – propagation of a Gaussian pulse

- Spreading of a sound pulse in a rectangular domain with rigid walls, simulated using Newmark scheme