Lecture n°2 — Solving scattering problems with the Boundary Element Method

Florian Feppon

October 13, 2021

1 Integral formulation of the scattering problem

At the last lecture, we consider the scattering of sound through a set of obstacles.

$$\begin{cases} \Delta u + k_b^2 u = 0 \text{ in } D, \\ \Delta u + k^2 u = 0 \text{ in } \mathbb{R}^d \setminus D, \\ \frac{1}{\rho}\frac{\partial u}{\partial n}\Big|_+ = \frac{1}{\rho_b}\frac{\partial u}{\partial n}\Big|_- \text{ on } \partial D, \\ u|_+ = u|_- \text{ on } \partial D, \\ \left(\frac{\partial}{\partial |x|} - \mathrm{i}k\right)(u - u_{\text{in}}) = O\left(|x|^{-(d+1)/2}\right) \text{ as } |x| \to +\infty. \end{cases} \tag{1}$$

In this lecture, we derive an integral formulation for (1) and we discuss its numerical solution with the Boundary Element Method.

Using the jump relations, we start by deriving an integral equation formulation for the scattering problem (1). Because the function u has a continuous trace accross ∂D, we choose to represent u with a single layer potential representation:

$$u(x) := \begin{cases} \mathcal{S}_D^{k_b}[\phi](x) \text{ if } x \text{ in } D, \\ u_{\text{in}}(x) + \mathcal{S}_D^k[\psi](x) \text{ if } x \in \mathbb{R}^d \setminus D. \end{cases} \tag{2}$$

where we recall the definition of the single layer potential:

$$\mathcal{S}_D^k[\phi](x) := \int_{\partial D} \Gamma^k(x - t')\phi(t')\mathrm{d}\sigma(t'), \qquad x \in \mathbb{R}^d \setminus \partial D, \quad \phi \in L^2(\partial D).$$

By choosing such representation, the function u satisfies automatically

$$\begin{cases} \Delta u + k_b^2 u = 0 \text{ in } D, \\ \Delta u + k^2 u = 0 \text{ in } \mathbb{R}^d \setminus D, \\ \left(\frac{\partial}{\partial |x|} - \mathrm{i}k\right)(u - u_{\text{in}}) = O(|x|^{-(d+1)/2}) \text{ as } |x| \to +\infty. \end{cases}$$

The continuity condition $u|_+ = u|_-$ on ∂D is equivalent to require

$$\mathcal{S}_D^{k_b}[\phi] = \mathcal{S}_D^k[\psi] + u_{\text{in}} \text{ on } \partial D. \tag{3}$$

It remains to impose the continuity of the flux. We recall the jump identity

$$\frac{\partial}{\partial n}\mathcal{S}_D^k\Big|_\pm = \pm\frac{1}{2}\mathrm{I} + \mathcal{K}_D^{k*}, \tag{4}$$

where $\mathcal{K}_D^{k*}$ is the adjoint of the Neumann-Poincaré operator and is given (when D is a smooth domain) by

$$\mathcal{K}_D^{k*}[\phi](t) = \int_{\partial D} \boldsymbol{n}(t) \cdot \nabla_t \Gamma^k (t-t')\phi(t')\mathrm{d}\sigma(t').$$

Applying the identity (4) to (2), we obtain

$$\left.\frac{\partial u}{\partial n}\right|_- = \left(-\frac{1}{2}\mathrm{I} + \mathcal{K}_D^{k_b*}\right)[\phi], \quad \delta \left.\frac{\partial u}{\partial n}\right|_+ = \delta\left(\frac{1}{2}\mathrm{I} + \mathcal{K}_D^{k*}\right)[\psi] + \delta\frac{\partial u_{\mathrm{in}}}{\partial n},$$

therefore, equating the two, we obtain that (ϕ, ψ) solves a system of integral equations:

$$\begin{bmatrix} \mathcal{S}_D^{k_b} & -\mathcal{S}_D^{k} \\ \left(-\frac{1}{2}\mathrm{I} + \mathcal{K}_D^{k_b*}\right) & -\delta\left(\frac{1}{2}\mathrm{I} + \mathcal{K}_D^{k*}\right) \end{bmatrix} \begin{bmatrix} \phi \\ \psi \end{bmatrix} = \begin{bmatrix} u_{\mathrm{in}} \\ \delta\frac{\partial u_{\mathrm{in}}}{\partial n} \end{bmatrix}. \tag{5}$$

Recalling that $k = \omega/v$, $k_b = \omega/v_b$ and the expression of the potentials, it turns out that each of the above operators have an analytic continuation to $\mathbb{C}$ (if $d = 3$) or $\mathbb{C} \setminus \mathrm{i}\mathbb{R}$ (if $d = 2$) (this is Steiner theorem). Therefore, the above system can be formulated as

$$\mathcal{A}(\omega, \delta) \begin{bmatrix} \phi \\ \psi \end{bmatrix} = \begin{bmatrix} u_{\mathrm{in}} \\ \delta\frac{\partial u_{\mathrm{in}}}{\partial n} \end{bmatrix} \tag{6}$$

where the operator $\mathcal{A}(\omega, \delta)$ is a holomorphic Fredholm operator.

2 Boundary element method

Here we shall see how to solve (5) with the Boundary Element Method using the open source library Gypsilab [1].

We consider the variational formulation

find $(\phi, \psi) \in L^2(\partial D) \times L^2(\partial D)$ such that for any $(v, w) \in L^2(\partial D) \times L^2(\partial D)$,

$$\begin{cases} \int_{\partial D} (\mathcal{S}_D^{k_b}[\phi] - \mathcal{S}_D^{k}[\psi])v\mathrm{d}\sigma = \int_{\partial D} u_{\mathrm{in}} v\mathrm{d}\sigma \\ \int_{\partial D} \left(\left(-\frac{1}{2}\mathrm{I} + \mathcal{K}_D^{k_b*}\right)[\phi] - \delta\left(\frac{1}{2}\mathrm{I} + \mathcal{K}_D^{k*}\right)[\psi]\right) w\mathrm{d}\sigma = \int_{\partial D} \delta\frac{\partial u_{\mathrm{in}}}{\partial n} w\mathrm{d}\sigma. \end{cases} \tag{7}$$

These various integrals are discretized with the finite element method: we consider a mesh discretization of ∂D and we approximate the space $L^2(\partial D)$ by the space V_h of piecewise linear functions where h denotes the mesh size. Then we discretize (7) as follows: find $(\phi_h, \psi_h) \in V_h \times V_h$ such that for any $(v_h, w_h) \in V_h \times V_h$,

$$\begin{cases} \int_{\partial D} (\mathcal{S}_D^{k_b}[\phi_h] - \mathcal{S}_D^{k}[\psi_h])v_h\mathrm{d}\sigma = \int_{\partial D} u_{\mathrm{in}} v_h\mathrm{d}\sigma \\ \int_{\partial D} \left(\left(-\frac{1}{2}\mathrm{I} + \mathcal{K}_D^{k_b*}\right)[\phi_h] - \delta\left(\frac{1}{2}\mathrm{I} + \mathcal{K}_D^{k*}\right)[\psi_h]\right) w_h\mathrm{d}\sigma = \int_{\partial D} \delta\frac{\partial u_{\mathrm{in}}}{\partial n} w_h\mathrm{d}\sigma. \end{cases} \tag{8}$$

Then, the system is a linear system with as many degrees of freedom than the dimension of V_h. If $V_h = \mathrm{span}(u_i)_{1\leqslant i\leqslant n}$, and denoting by $\phi_h = \sum_{i=1}^n x_i u_i$ and $\psi = \sum_{i=1}^n y_i u_i$, this system reads

$$A_h \begin{bmatrix} (x_i)_{1\leqslant i\leqslant n} \\ (y_i)_{1\leqslant i\leqslant n} \end{bmatrix} = b$$

with A_h being the matrix

$$A_h = \begin{bmatrix} \int_{\partial D} \mathcal{S}_D^{k_b}[u_j]u_i\mathrm{d}\sigma & -\int_{\partial D} \mathcal{S}_D^{k}[u_j]u_i\mathrm{d}\sigma \\ \int_{\partial D}\left(-\frac{1}{2}\mathrm{I} + \mathcal{K}_D^{k_b*}\right)[u_j]u_i\mathrm{d}\sigma & -\delta\int_{\partial D}\left(\frac{1}{2}\mathrm{I} + \mathcal{K}_D^{k*}\right)[u_j]u_i\mathrm{d}\sigma \end{bmatrix}$$

and b the vector

$$b = \left[\left(\int_{\partial D} u_{\mathrm{in}} u_i\mathrm{d}\sigma\right)_{1\leqslant i\leqslant N} \left(\int_{\partial D} \delta\frac{\partial u_{\mathrm{in}}}{\partial n} u_i\mathrm{d}\sigma\right)_{1\leqslant i\leqslant n}\right]$$

Remark 1. *The numerical analysis of the convergence of such finite-element method for Fredholm problems is much more involved than the one associated with coercive systems where Céa's lemma [6, 2]. See e.g. [5] for an approach to Boundary Element Methods.*

3 BEM with Gypsilab

The use of Gypsilab is quite convenient for solving quickly integral formulations.

3.1 Computing finite element matrices

The syntax of Gypsilab is quite easy to use. One may want to compute several types of integrals:

- finite-element matrices of the form

$$S_{ij} := \int_\Gamma \int_\Sigma u_i(x) G(x-y) v_j(y) \mathrm{d}x \mathrm{d}y$$

  where G is an integral kernel, Γ and Σ some bounded domains (volume or surface) and u_i and v_j are finite element functions. A domain Σ is declared e.g. by

```
mesh = mshCircle(100,0.5);
Sigma = dom(mesh,3);
```

  where we define a mesh of a circle at the first line, and we state that `Sigma` is a domain of integration with an integration rule of order 3. A finite element space is declared by using

```
Vh = fem(mesh,'P1');
```

  Then, for regular kernels G, the matrix S_{ij} is easily computed by the procedure

```
Sij = integral(Gamma, Sigma, Wh, G, Vh);
```

  An additional regularization operation is required for singular kernels described below.

- Interpolation matrices of the form

$$\int_\Gamma G(x_i - y) v_j(y) \mathrm{d}y$$

  where (x_i) are points of a mesh. This integral is computed by

```
pts = mesh.vtx;
interps = integral(Gamma,pts,G,Vh);
```

3.2 Regularization operations for singular operators

Singularities are treated by using the `regularize` function. To compute the finite element matrix associated to the single layer potential in two dimensions, one uses the following code

```
GxySk = @(X,Y) femGreenKernel(X,Y,'[H0(kr)]',k);
Sk = -1i/4*integral(Gamma,Gamma,Vh,GxySk,Vh);
AR = 1/(2*pi)*regularize(Gamma,Gamma,Vh,'[log(r)]',Vh);

Sk = Sk+AR;
```

The first line returns the Green Kernel based on the Hankel function. The second line computes the matrix $\left(\int_{\partial D}\int_{\partial D} \Gamma^k(x-y) u_i(x) u_j(y) \mathrm{d}\sigma(x) \mathrm{d}\sigma(y)\right)_{1\leqslant i,j\leqslant N}$ by using a standard integration rule (neglecting the singularity on the diagonal). The third line returns a matrix of corrections for the diagonal terms based on a semi-analytical integration of the singularity near the diagonal. For the Hankel function, we have

$$-\frac{\mathrm{i}}{4} H_0^{(1)}(k|x-y|) \sim -\frac{1}{2\pi} \log|x-y|$$

as $|x-y| \to 0$, whence the arguments of `regularize`.

3.3 Acceleration methods

The expensive step of the method is the computation of the inverse of convolution matrices of the form

$$G_{ij} := (G(y_i - y_j))_{1 \leqslant i \neq j \leqslant N}$$

or variants of it corresponding to the quadrature rule considered. Several acceleration techniques have been proposed in the literature:

1. The Fast Multipole Method of Rokhlin and Greengard [7], called one of the top 10 algorithms of the century. It allows to compute very efficiently matrix vector products of the form

   $$\sum_{j=1}^{N} G(y_i - y_j) v_j, \qquad 1 \leqslant i \leqslant M$$

   in an optimal complexity in M and N, which allows to solve linear systems in G with iterative methods. However the implementation is non-trivial.

2. Non optimal methods which are easier to implements, or simply available: the efficient Bessel decomposition [4] and the Fast and Free Memory Method [3] (both available in Gypsilab).

3. Compression techniques: H-matrices [8] which can be readily used by adding a tolerance parameter in `integral`.

```
tol=1e-3;
Sk = -1i/4*integral(Gamma,Gamma,Vh,GxySk,Vh,tol);
```

4 Acceleration of boundary element methods

The expensive step of the method is the computation of the inverse of convolution matrices of the form

$$G_{ij} := (G(y_i - y_j))_{1 \leqslant i \neq j \leqslant N}$$

or variants of it corresponding to the quadrature rule considered. Several acceleration techniques have been proposed in the literature:

1. The Fast Multipole Method of Rokhlin and Greengard [7], called one of the top 10 algorithms of the century. It allows to compute very efficiently matrix vector products of the form

   $$\sum_{j=1}^{N} G(y_i - y_j) v_j, \qquad 1 \leqslant i \leqslant M$$

   in an optimal complexity in M and N, which allows to solve linear systems in G with iterative methods. However the implementation is non-trivial.

2. Non optimal methods which are easier to implements, or simply available: the efficient Bessel decomposition [4] and the Fast and Free Memory Method [3] (both available in Gypsilab).

3. Compression techniques: H-matrices [8] which can be readily used by adding a tolerance parameter in `integral`.

```
tol=1e-3;
Sk = -1i/4*integral(Gamma,Gamma,Vh,GxySk,Vh,tol);
```

References

[1] F. Alouges and M. Aussal, *FEM and BEM simulations with the Gypsilab framework*, The SMAI journal of computational mathematics, 4 (2018), pp. 297–318.

[2] K. E. Atkinson and W. Han, *Theoretical numerical analysis. A Functional Analysis Framework.*, vol. 39, Springer, 2005.

[3] M. Aussal and M. Bakry, *The Fast and Free Memory Method for the efficient computation of convolution kernels*, arXiv preprint arXiv:1909.05600, (2019).

[4] M. Averseng, *Fast discrete convolution in $\mathbb{R}2$ with radial kernels using non-uniform fast Fourier transform with nonequispaced frequencies*, Numerical Algorithms, 83 (2020), pp. 33–56.

[5] S. H. Christiansen, *Discrete Fredholm properties and convergence estimates for the electric field integral equation*, Mathematics of Computation, 73 (2003), pp. 143–167.

[6] A. Ern and J.-L. Guermond, *Theory and Practice of Finite Elements*, vol. 159, Springer, 2004.

[7] Z. Gimbutas and L. Greengard, *Simple FMM libraries for electrostatics, slow viscous flow, and frequency-domain wave propagation*, Communications in Comput. Phys, 18 (2015), pp. 516–528.

[8] W. Hackbusch, *Sparse matrix arithmetic based on H-matrices. Part I: Introduction to H-matrices*, Computing (Vienna/New York), 62 (1999), pp. 89–108.