Parallel Time-Domain Boundary Element Method for 3-Dimensional Wave Equation

M. Merta[1,2] A. Veit[3] J. Zapletal[1,2] D. Lukáš[1,2]

[1]Dept. of Applied Mathematics,
[2]IT4Innovations National Supercomputing Center, VŠB-TU Ostrava, Czech Republic
[3]The University of Chicago

michal.merta@vsb.cz

December 22, 2016

Boundary element method in the BEM4I library

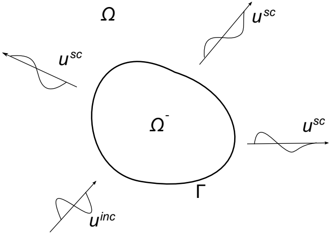
- Boundary element method
  - Reduces problem to the boundary of computational domain
  - Suitable for exterior problems or shape optimization
- BEM4I
  - Developed at IT4Innovations National Supercomputing Center, Ostrava, Czech Republic
  - C++, OpenMP & MPI, SIMD vectorization
  - Acceleration using the Intel Xeon Phi coprocessors
  - Adaptive Cross Approximation
  - BETI by interfacing the ESPRESO DDM library
  - 3D Laplace, Helmholtz, Lamé, and **wave equation**

Wave equation

Sound-hard scattering problem

$$\begin{cases} \frac{1}{c^2}\frac{\partial^2 u^{\mathrm{sc}}}{\partial^2 t}(\boldsymbol{x},t) - \Delta u^{\mathrm{sc}}(\boldsymbol{x},t) &= 0 & \text{in } \Omega \times \mathbb{R}, \\ u^{\mathrm{sc}}(\boldsymbol{x},0) &= 0 & \text{in } \Omega, \\ \frac{\partial u^{\mathrm{sc}}}{\partial t}(\boldsymbol{x},0) &= 0 & \text{in } \Omega, \\ \frac{\partial u^{\mathrm{sc}}}{\partial \boldsymbol{n}}(\boldsymbol{x},t) &= -\frac{\partial u^{\mathrm{inc}}}{\partial \boldsymbol{n}}(\boldsymbol{x},t) & \text{on } \partial\Omega \times \mathbb{R}_+. \end{cases}$$

- Space-time integral equations
  - using the fundamental solution of the wave equation
  - global in time - large system matrix
  - special quadrature method needed.



Boundary integral formulation

Boundary integral ansatz [Bamberger, Ha-Duong 86]

We search for u^{sc} in the form of the retarded double-layer potential

$$u^{\mathrm{sc}} = -\frac{1}{4\pi} \int_{\Gamma} \frac{\boldsymbol{n}_{\boldsymbol{y}}(\boldsymbol{x}-\boldsymbol{y})}{\|\boldsymbol{x}-\boldsymbol{y}\|} \left(\frac{\phi(\boldsymbol{y}, t-\|\boldsymbol{x}-\boldsymbol{y}\|)}{\|\boldsymbol{x}-\boldsymbol{y}\|^2} + \frac{\dot{\phi}(\boldsymbol{y}, t-\|\boldsymbol{x}-\boldsymbol{y}\|)}{\|\boldsymbol{x}-\boldsymbol{y}\|} \right) dS_{\boldsymbol{y}},$$

which satisfies the wave equation and the initial conditions. It remains to fulfill the Neumann boundary condition

$$\underbrace{\lim_{\Omega \ni \widetilde{\boldsymbol{x}} \to \boldsymbol{x} \in \Gamma} n_{\boldsymbol{x}} \cdot \nabla_{\widetilde{\boldsymbol{x}}} u^{\mathrm{sc}}(\widetilde{\boldsymbol{x}}, t)}_{=:(W\phi)(\boldsymbol{x},t)} = g(\boldsymbol{x}, t) \text{ on } \Gamma \times [0, T],$$

where $g := -\frac{\partial u^{\mathrm{inc}}}{\partial \boldsymbol{n}}$.

Boundary integral formulation

Weak boundary integral formulation [Bamberger, Ha-Duong 86]

Find ϕ such that

$$a(\xi, \phi) = b(\xi) \quad \forall \xi \in V,$$

where

$$a(\xi, \phi) := \int_0^T \int_\Gamma \int_\Gamma \left\{ \frac{\boldsymbol{n}_x \cdot \boldsymbol{n}_y}{4\pi \|\boldsymbol{x} - \boldsymbol{y}\|} \, \dot{\xi}(\boldsymbol{x}, t) \, \ddot{\phi}(\boldsymbol{y}, t - \|\boldsymbol{x} - \boldsymbol{y}\|) + \frac{\mathrm{curl}_\Gamma \dot{\xi}(\boldsymbol{x}, t) \cdot \mathrm{curl}_\Gamma \phi(\boldsymbol{y}, t - \|\boldsymbol{x} - \boldsymbol{y}\|)}{4\pi \|\boldsymbol{x} - \boldsymbol{y}\|} \right\} dS_y \, dS_x \, dt,$$

$$b(\xi) := \int_0^T \int_\Gamma g(\boldsymbol{x}, t) \, \dot{\xi}(\boldsymbol{x}, t) \, dS_x \, dt.$$

Discretization

Discrete ansatz

Replace V by a finite-dimensional subspace $V^{h,\Delta t}$ spanned by the tensor-product of N temporal and M spatial basis functions:

$$\phi^{h,\Delta t}(x,t) := \sum_{l=1}^{N} \sum_{j=1}^{M} \alpha_l^j \, \varphi_j(x) \, b_l(t).$$

We arrive at the $(N\,M) \times (N\,M)$ block linear system

$$\begin{pmatrix} \mathbf{A}_{1,1} & \dots & \mathbf{A}_{1,N} \\ \vdots & \ddots & \vdots \\ \mathbf{A}_{N,1} & \dots & \mathbf{A}_{N,N} \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha}_1 \\ \vdots \\ \boldsymbol{\alpha}_N \end{pmatrix} = \begin{pmatrix} \mathbf{b}_1 \\ \vdots \\ \mathbf{b}_N \end{pmatrix},$$

where

$$(\mathbf{A}_{k,l})_{i,j} := a\left(\varphi_i(x)\, b_k(t), \varphi_j(y)\, b_l(t)\right), \quad (\mathbf{b}_k)_i := b\left(\varphi_i(x)\, b_k(t)\right), \quad (\boldsymbol{\alpha}_l)_j := \alpha_l^j.$$

System matrix

$$(\boldsymbol{A}_{k,l})_{i,j} = \int\limits_{\operatorname{supp}\varphi_i} \int\limits_{\operatorname{supp}\varphi_j} \frac{\boldsymbol{n_x}\cdot\boldsymbol{n_y}}{4\pi\|\boldsymbol{x}-\boldsymbol{y}\|}\varphi_i(\boldsymbol{x})\,\varphi_j(\boldsymbol{y}) \overbrace{\int_0^T \dot{b}_k(t)\,\ddot{b}_l(t-\|\boldsymbol{x}-\boldsymbol{y}\|)\,dt}^{=:\Psi_{k,l}(\|\boldsymbol{x}-\boldsymbol{y}\|)}\,dS_{\boldsymbol{y}}\,dS_{\boldsymbol{x}}$$

$$+\int\limits_{\operatorname{supp}\varphi_i} \int\limits_{\operatorname{supp}\varphi_j} \frac{\operatorname{curl}_\Gamma\varphi_i(\boldsymbol{x})\cdot\operatorname{curl}_\Gamma\varphi_j(\boldsymbol{y})}{4\pi\|\boldsymbol{x}-\boldsymbol{y}\|} \underbrace{\int_0^T \dot{b}_k(t)\,b_l(t-\|\boldsymbol{x}-\boldsymbol{y}\|)\,dt}_{=:\widetilde{\Psi}_{k,l}(\|\boldsymbol{x}-\boldsymbol{y}\|)}\,dS_{\boldsymbol{y}}\,dS_{\boldsymbol{x}},$$

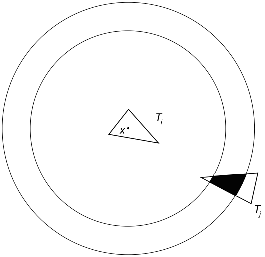
Integration problem

Piecewise polynomial time-ansatz $\rightsquigarrow$ expensive quadrature due to nontrivial intersection of the light cone

$$\operatorname{supp}\Psi_{k,l},\quad \operatorname{supp}\widetilde{\Psi}_{k,l}$$

with

$$\operatorname{supp}\varphi_i\times\operatorname{supp}\varphi_j.$$



C^∞ temporal basis functions

- In cooperation with A. Veit (Uni. of Zurich/Chicago)
- C^∞ temporal basis functions based on the partition of unity method [Sauter, Veit, 2012]
- Temporal basis functions in the form

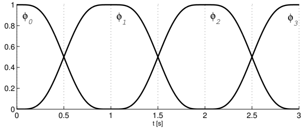
$$b_{i,m} := \phi_i(t) P_m(t)$$

  - $\phi_i(t)$ are C^∞ partition of unity function, $\{P_m(t)\}_{m=0}^p$ are Legendre polynomial

$$f(t) := \begin{cases} \frac{1}{2}\operatorname{erf}(2\operatorname{artanh} t) + \frac{1}{2} & |t| < 1, \\ 0 & t \le -1, \\ 1 & t \ge 1 \end{cases} \qquad f_i(t) := f\left(2\frac{t - t_i}{\Delta t} - 1\right), \quad \text{where } f_i(t) = \begin{cases} 0 & t \le t_i, \\ 1 & t \ge t_{i+1}. \end{cases}$$

$$\rho_i(t) := \begin{cases} f_{i-1}(t) & t_{i-1} \le t \le t_i, \\ 1 - f_i(t) & t_i \le t \le t_{i+1}, \\ 0 & \text{otherwise.} \end{cases} \qquad \Phi_1 := 1 - f_0, \quad \Phi_N := f_{N-2}, \quad \forall i \in \{2, \ldots, N-1\} : \Phi_i := \rho_{i-1}.$$

C^∞ temporal basis functions

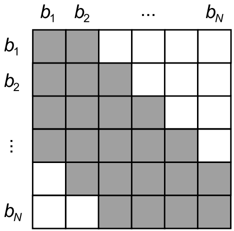


$$(\boldsymbol{A}_{k,l})_{i,j} = \int\limits_{\operatorname{supp}\varphi_i} \int\limits_{\operatorname{supp}\varphi_j} \frac{\boldsymbol{n_x}\cdot\boldsymbol{n_y}}{4\pi\|\boldsymbol{x}-\boldsymbol{y}\|}\varphi_i(x)\,\varphi_j(y)\,\overbrace{\int_0^T \dot{b}_k(t)\,\ddot{b}_l(t-\|\boldsymbol{x}-\boldsymbol{y}\|)\,dt}^{=:\Psi_{k,l}(\|\boldsymbol{x}-\boldsymbol{y}\|)\in C^\infty(\mathbb{R})}\,dS_{\boldsymbol{y}}\,dS_{\boldsymbol{x}}$$
$$+\int\limits_{\operatorname{supp}\varphi_i} \int\limits_{\operatorname{supp}\varphi_j} \frac{\operatorname{curl}_\Gamma\varphi_i(\boldsymbol{x})\cdot\operatorname{curl}_\Gamma\varphi_j(\boldsymbol{y})}{4\pi\|\boldsymbol{x}-\boldsymbol{y}\|}\underbrace{\int_0^T \dot{b}_k(t)\,b_l(t-\|\boldsymbol{x}-\boldsymbol{y}\|)\,dt}_{=:\widetilde{\Psi}_{k,l}(\|\boldsymbol{x}-\boldsymbol{y}\|)\in C^\infty(\mathbb{R})}\,dS_v y\,dS_{\boldsymbol{x}},$$

- Allows for Sauter-Schwab quadrature over $\operatorname{supp}\varphi_i \times \operatorname{supp}\varphi_j$
- The method allows variable order of temporal basis functions
- Computationally demanding $\Rightarrow$ hybrid parallelization by OpenMP/MPI
- To accelerate the assembly, Ψ and $\widetilde{\Psi}$ are replaced by piecewise Chebyshev interpolants

C^∞ temporal basis functions

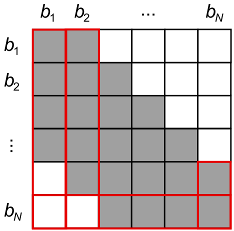
- System matrix with block-Hessenberg structure



- $N \times N$ sparse blocks where N is number of time-steps
- Each block has dimension $(p+1)M \times (p+1)M$
- Hybrid parallelization by OpenMPI and MPI
- System solved by GMRES

C^∞ temporal basis functions

- System matrix with block-Hessenberg structure



- $N \times N$ sparse blocks where N is number of time-steps
- Each block has dimension $(p+1)M \times (p+1)M$
- Hybrid parallelization by OpenMPI and MPI
- System solved by GMRES

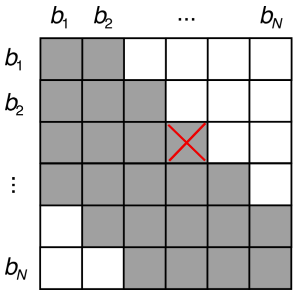
Parallelization of the system matrix assembly

- Assembly of individual blocks
  1. Precompute the non-zero pattern of the block
  2. Distribute pairs of elements contributing to non-zero matrix entries among computational nodes using MPI
  3. On each computational node assemble its contribution to the block in a shared memory using OpenMP
  4. Gather the data on an MPI rank(s) owning the given block
- Distribution of blocks among MPI processes

$$\begin{bmatrix}
\tilde{\mathbf{A}}_{1,1} & \tilde{\mathbf{A}}_{1,2} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\tilde{\mathbf{A}}_{2,1} & \tilde{\mathbf{A}}_{2,2} & \tilde{\mathbf{A}}_{2,3} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\tilde{\mathbf{A}}_{3,1} & \tilde{\mathbf{A}}_{3,2} & \tilde{\mathbf{A}}_{3,3} & \tilde{\mathbf{A}}_{3,4} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \tilde{\mathbf{A}}_{4,2} & \tilde{\mathbf{A}}_{4,3} & \tilde{\mathbf{A}}_{4,4} & \tilde{\mathbf{A}}_{4,5} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \tilde{\mathbf{A}}_{5,3} & \tilde{\mathbf{A}}_{5,4} & \tilde{\mathbf{A}}_{5,5} & \tilde{\mathbf{A}}_{5,6} & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \tilde{\mathbf{A}}_{6,4} & \tilde{\mathbf{A}}_{6,5} & \tilde{\mathbf{A}}_{6,6} & \tilde{\mathbf{A}}_{6,7} & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \tilde{\mathbf{A}}_{7,5} & \tilde{\mathbf{A}}_{7,6} & \tilde{\mathbf{A}}_{7,7} & \tilde{\mathbf{A}}_{7,8} \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \tilde{\mathbf{A}}_{8,6} & \tilde{\mathbf{A}}_{8,7} & \tilde{\mathbf{A}}_{8,8}
\end{bmatrix} \xrightarrow{R}$$

	1	0					
	2	1	0				
	4	3	1	0			
		4	3	2	0		
			4	3	2	1	
				4	3	2	

Preconditioning



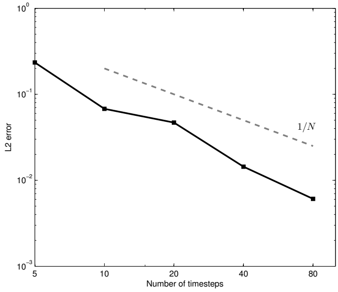
- We approximate the upper Hessenberg matrix by an inexact lower triangular preconditioner:

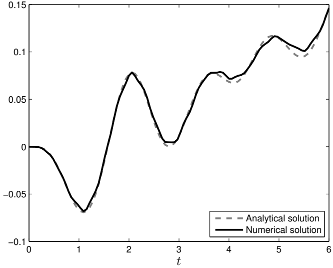
$$\boldsymbol{A} := \begin{pmatrix} \boldsymbol{A}_{I,I} & \boldsymbol{A}_{I,II} \\ \boldsymbol{A}_{II,I} & \boldsymbol{A}_{II,II} \end{pmatrix} \approx \begin{pmatrix} \widehat{\boldsymbol{A}}_{I,I} & \boldsymbol{0} \\ \boldsymbol{A}_{II,I} & \widehat{\boldsymbol{A}}_{II,II} \end{pmatrix} =: \widehat{\boldsymbol{A}}$$

  in the recursive fashion so that $\widehat{\boldsymbol{A}}_{I,I}$ and $\widehat{\boldsymbol{A}}_{II,II}$ are again the inexact lower triangular preconditioners to the upper Hessenberg matrices $\boldsymbol{A}_{I,I}$ and $\boldsymbol{A}_{II,II}$, respectively.
- Only a few iterations of the inner solvers are applied
- Since the inner systems are solved inexactly we use the FGMRES algorithm [Saad 93].

Convergence

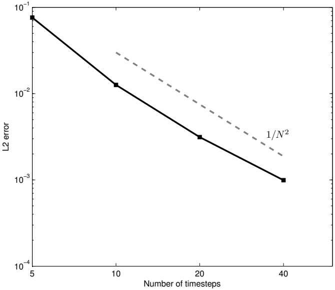
- $\|\phi - \phi_{\mathrm{ex}}\|_{L^2(\Gamma,[0,T])}$ for $T = 6$
- exact solution for special RHS $g(x,t)$ given in [Veit: *Numerical Methods for Time-Domain Boundary Integral Equations*, 2011]
- Legendre polynomial order $p = 1$

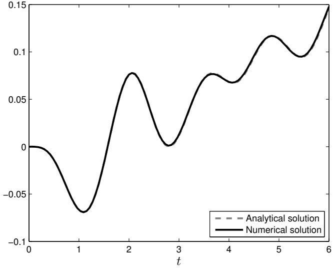




Convergence

- $\|\phi - \phi_{\text{ex}}\|_{L^2(\Gamma,[0,T])}$ for $T = 6$
- exact solution for special RHS $g(x,t)$ given in [Veit: *Numerical Methods for Time-Domain Boundary Integral Equations*, 2011]
- Legendre polynomial order $p = 2$





Convergence of iterative solver

	GMRES(50)	
N	# iterations	time [s]
5	595	1.6
10	2121	14.3
15	4021	44.5
20	5448	99.0

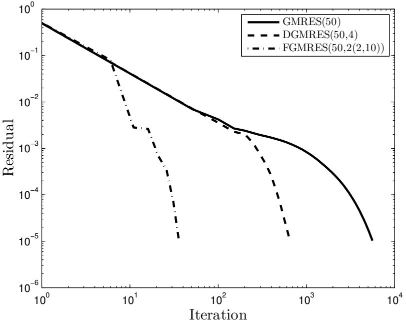
Table: Convergence of GMRES for $p = 1$.

	FGMRES(50, 1(10))		FGMRES(50, 1(5))		FGMRES(50, 2(2, 10))	
N	# iterations	time [s]	# iterations	time [s]	# iterations	time [s]
5	24	0.7	45	0.9	23	0.8
10	43	3.1	126	6.8	26	3.3
15	51	7.3	205	20.0	28	5.9
20	48	9.7	341	51.2	34	10.6

Table: Convergence of FGMRES with recursive preconditioner for $p = 1$.

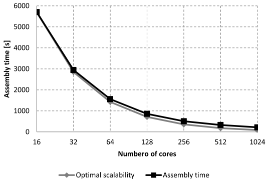
Convergence of iterative solver

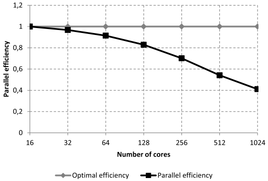
- Comparison of solution by GMRES, DGMRES, and FGMRES with recursive preconditioner



Parallel assembly and solve

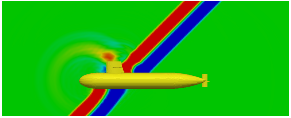
Matrix assembly scalability





GMRES(500)		FGMRES(500, 1(30))		FGMRES(500, 1(40))	
# iterations	time [s]	# iterations	time [s]	# iterations	time [s]
9656	1607	307	1076	243	962

- 5604 surface elements, 40 time-steps, $p = 1$, up to 64 compute nodes, 16 OpenMP threads per node
- computed at the Anselm cluster (209 comp. nodes, 2x 8-core Intel Xeon E5-2665, Sandy Bridge, 64 GB RAM, InfiniBand)



Conclusion & outlook

- Parallel implementation of BEM for the wave equation based on an approach using smooth temporal basis functions to overcome problems with numerical integration
- For Dirichlet and Neumann problem
- Implemented in the in-house boundary element library BEM4I
- Outlook
  - Improvement of a preconditioner
  - Better load balancing for parallel computation
  - Problems on half-spaces

References

A. Veit, M. Merta, J. Zapletal, and D. Lukáš. *Numerical solution of time-domain boundary integral equations arising in sound-hard scattering*. Int. J. Numer. Meth. Engng. 2016. In press.

S. Sauter, A. Veit. *A Galerkin method for retarded boundary integral equations with smooth and compactly supported temporal basis functions*. Numerische Mathematik. 2013.

Thank you for your attention!

Conclusion & outlook

- Parallel implementation of BEM for the wave equation based on an approach using smooth temporal basis functions to overcome problems with numerical integration
- For Dirichlet and Neumann problem
- Implemented in the in-house boundary element library BEM4I
- Outlook
  - Improvement of a preconditioner
  - Better load balancing for parallel computation
  - Problems on half-spaces

References

A. Veit, M. Merta, J. Zapletal, and D. Lukáš. *Numerical solution of time-domain boundary integral equations arising in sound-hard scattering*. Int. J. Numer. Meth. Engng. 2016. In press.

S. Sauter, A. Veit. *A Galerkin method for retarded boundary integral equations with smooth and compactly supported temporal basis functions*. Numerische Mathematik. 2013.

Thank you for your attention!