DOI: 10.14529/jsfi250403

Supercomputer Methods of Ultrasound Tomographic Imaging in NDT Based on Lamb Waves

Alexander S. Belyaev[1], ***Alexander V. Goncharsky***[1], ***Sergey Y. Romanov***[1], ***Vadim V. Voevodin***[1]

This article concerns the developing of supercomputer methods for solving inverse problems of ultrasonic tomography in application to nondestructive testing of thin plates using Lamb waves. Such problems are computationally expensive, as longitudinal, shear, and other waves propagate in solids, requiring the use of vector elastic models of wave propagation. Iterative methods for solving the inverse problem have been developed. The methods are based on gradient descent methods of minimizing the residual functional. Efficiency of the proposed algorithms is illustrated on model problems. The field of wave tomography, which is currently under development, requires powerful computing resources. Parallel computations in this study have been performed on general-purpose processors of the Lomonosov supercomputer complex. Solving the Helmholtz equation is the core element of the developed algorithms for solving inverse problems of wave tomography. The most demanding computations involve solving linear equations with large-scale sparse matrices using LU-decomposition method. The algorithms were implemented using linear algebra libraries with serial and parallel code. Effectiveness, scalability and performance of the method has been evaluated on CPU computing platforms.

Keywords: supercomputer, mathematical modeling, guided wave tomography, inverse problems, Lamb waves.

Introduction

This paper concerns the developing of developing ultrasonic tomographic imaging methods for non-destructive testing of solids. Immediately after the advent of the first medical X-ray tomographs, the idea of ultrasonic tomographic imaging devices arose. Solving this problem would eliminate the use of ionizing radiation in medical imaging. Ultrasonic tomography opens up fundamentally new possibilities for nondestructive imaging of solids. However, developing wave tomography devices has proven much more complex than X-ray ones, primarily due to the mathematical challenges involved. While the inverse problems in X-ray tomography can be solved using linear mathematical models, wave tomography imaging requires solving nonlinear inverse problems.

Over the past 20 years, breakthrough results in ultrasound tomography have been achieved in soft tissue diagnostics. One of the key challenges in modern medicine is the diagnosis of breast cancer in the early stages of the disease. This problem can be successfully solved using ultrasound tomography [1–5]. A unique feature of breast cancer diagnostics is the ability to interpret tomography data in a scalar wave model, since soft tissues are more than 90% water. Mostly the pressure waves propagate through soft tissues. However, even in this simple model, the inverse problem is nonlinear, and its solution requires the use of supercomputers [6]. Efficient iterative algorithms for interpreting experimental data have been developed using the explicit formulation of the gradient of the residual functional between the calculated and measured wave fields at the detectors [7, 8]. Effectiveness of the developed methods has been evaluated on supercomputers using various computing platforms [9, 10].

Inverse problems of ultrasound tomography in NDT are much more complex, since the mathematical model must describe the propagation of both longitudinal and shear waves. The

[1]Lomonosov Moscow State University, Moscow, Russian Federation

displacement of a point is described by a vector, and a system of second-order differential equations is constructed for the components of the displacement vectors [11–14]. The inverse problem is reduced to minimizing a functional dependent on three unknowns: density and two elasticity coefficients as functions of spatial coordinates. As in the scalar case, a representation for the gradient of the residual functional with respect to these three parameters is obtained in the vector elastic model [13, 15, 16]. Thus, from a mathematical standpoint, in both the scalar and vector models, the objective is to solve a coefficient inverse problem for differential equations. Iterative methods for finding an approximate solution employ the explicit representation for the gradient of the residual functional.

However, in this formulation, the inverse problem is computationally very expensive. For a number of practically important non-destructive testing (NDT) problems, the use of simpler mathematical models appears promising [17]. One such problem is ultrasonic diagnostics of defects in thin plates [18–21]. This study evaluates the supercomputer implementation of the ultrasonic tomographic imaging method for NDT using Lamb waves. Capabilities of the approximate solution method for the inverse problem of detecting surface defects in metal plates are examined in detail. Solving the Helmholtz equation is the core element of the proposed imaging algorithms. A specialized scalar model for thin-plate defect diagnostics allows for determining not only shapes, but also the depth map of defects. Even in this simplified model, a supercomputer is required to implement the developed algorithms in practice. The effectiveness of tomographic image reconstruction algorithms on a CPU platform is discussed in detail.

The article is organized as follows. Sections 1, 4.1, 4.2 are devoted to the formulation and model computations of the direct problem of ultrasound tomography. Sections 2, 3, 4.3, 4.4 are devoted to the formulation and model computations of the inverse problem. Section 5 contains performance analysis of a supercomputer implementation. Conclusion summarizes the study.

1. Statement of the Direct Problem of Ultrasound Tomography in a Vector Elastic Model

The object of tomographic imaging in this study is a thin, homogeneous flat plate, the thickness of which is on the order of the central wavelength of the sounding pulse (Fig. 1). Defect detection in thin plates and pipes is a pressing practical issue [18, 19]. The specificity of this problem is that Lamb waves of different modes propagate in thin plates and pipes. Defects in the imaged sample can be internal, associated with a local change in the parameters of the medium, or surface defects, associated with a local change in the plate thickness. First, we present the formulations of the direct and inverse problems for the case of internal defects.

Figure 1 shows a diagram of a tomographic imaging experiment, including sources and detectors of ultrasonic radiation. The sources sequentially emit ultrasonic sounding pulses, which are recorded by the detectors.

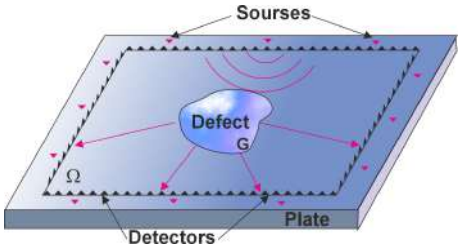


Figure 1. Scheme of the tomographic experiment

We assume that the sounding waves propagate in isotropic, linear, perfectly elastic media described by the Lame parameters and the density of the material. To simulate the wave propagation process in elastic media, the problem is often formulated in terms of particle velocity and stress. The dynamic equations in $\boldsymbol{R}^3$ describing wave propagation in an elastic volumetric plate in the velocity-stress formulation have the form of a first-order hyperbolic system [22]

$$\rho\frac{\partial u_1}{\partial t}=\frac{\partial \sigma_{11}}{\partial x_1}+\frac{\partial \sigma_{12}}{\partial x_2}+\frac{\partial \sigma_{13}}{\partial x_3},\quad \rho\frac{\partial u_2}{\partial t}=\frac{\partial \sigma_{12}}{\partial x_1}+\frac{\partial \sigma_{22}}{\partial x_2}+\frac{\partial \sigma_{23}}{\partial x_3},\quad \rho\frac{\partial u_3}{\partial t}=\frac{\partial \sigma_{13}}{\partial x_1}+\frac{\partial \sigma_{23}}{\partial x_2}+\frac{\partial \sigma_{33}}{\partial x_3},$$

$$\frac{\partial}{\partial t}\begin{pmatrix}\sigma_{11}\\ \sigma_{22}\\ \sigma_{33}\\ \sigma_{23}\\ \sigma_{13}\\ \sigma_{12}\end{pmatrix}=\begin{pmatrix}\lambda+2\mu & \lambda & \lambda & 0 & 0 & 0\\ \lambda & \lambda+2\mu & \lambda & 0 & 0 & 0\\ \lambda & \lambda & \lambda+2\mu & 0 & 0 & 0\\ 0 & 0 & 0 & \mu & 0 & 0\\ 0 & 0 & 0 & 0 & \mu & 0\\ 0 & 0 & 0 & 0 & 0 & \mu\end{pmatrix}\times\begin{pmatrix}\frac{\partial u_1}{\partial x_1}\\ \frac{\partial u_2}{\partial x_2}\\ \frac{\partial u_3}{\partial x_3}\\ \frac{\partial u_2}{\partial x_3}+\frac{\partial u_3}{\partial x_2}\\ \frac{\partial u_3}{\partial x_1}+\frac{\partial u_1}{\partial x_3}\\ \frac{\partial u_1}{\partial x_2}+\frac{\partial u_2}{\partial x_1}\end{pmatrix}+\begin{pmatrix}f_{11}\\ f_{22}\\ f_{33}\\ f_{23}\\ f_{13}\\ f_{12}\end{pmatrix}. \tag{1}$$

Here $\boldsymbol{u}(t,\boldsymbol{r})=(u_1,u_2,u_3)$ is the shear velocity vector, $\boldsymbol{\sigma}(t,\boldsymbol{r})=\sigma_{ij}$ is the stress tensor of size 3×3; $\mu(\boldsymbol{r})$, $\lambda(\boldsymbol{r})$ are the Lame coefficients; $\rho(\boldsymbol{r})$ is the bulk density; $\boldsymbol{f}(t,\boldsymbol{r})=f_{ij}$ is the tensor defining the external force. For these equations, we should also specify the initial conditions of rest at the time $t=0$

$$\boldsymbol{u}(0,x)=0,\ \boldsymbol{\sigma}(0,x)=0. \tag{2}$$

When solving the direct problem of elastic wave propagation, the functions $\mu(\boldsymbol{r})$, $\lambda(\boldsymbol{r})$, $\rho(\boldsymbol{r})$ are known, and the components of the wave field for each point of the region Ω are to be determined. Boundary conditions are applied on the boundary $\partial\Omega$ of the plate Ω. The lateral sides of the boundary $\Gamma\subset\partial\Omega$ are assumed stationary with the following boundary conditions

$$\boldsymbol{u}(t,\boldsymbol{r})\mid_{\Gamma}=0. \tag{3}$$

The rest of the boundary (the upper and lower sides of the plate) $\partial\Omega\setminus\Gamma$ is assumed a free boundary, i.e., no external forces act on $\partial\Omega\setminus\Gamma$, and we apply the conditions

$$\sigma(t,\boldsymbol{r})\,\boldsymbol{n}\mid_{\partial\Omega\setminus\Gamma}=0, \tag{4}$$

where $\boldsymbol{n}$ is the outer normal vector to the boundary at point $\boldsymbol{r}$.

In all the computations presented, the following sounding waveform was used as the external disturbance

$$f_{33}(x_1,x_2,x_3=0,t)=\begin{cases}C\sin\left(\frac{2}{3}\pi\nu_0 t\right)\sin\left(2\pi\nu_0 t\right), & 0<t<3/w_0,\\ 0,\ \text{otherwise},\end{cases} \tag{5}$$

and the remaining components of the force tensor $f_{ij}=0$. Here w_0 is the central frequency of the sounding pulse. The constant C determines the amplitude of the external disturbance. For the simulations presented below, we assume $C=1$ GPa/$\mu s=10^{-6}$ kg/(mm $\cdot$ μs^3). The time dependence of the pulse is a smooth function with a zero time derivative at the boundaries and a zero integral over the time of action. Under these conditions, the pulse spectrum does not contain very low and very high frequencies. The external disturbance is applied at a certain point on the upper surface of the plate.

2. Statement of the Inverse Problem of Ultrasonic Tomographic Imaging of Internal Defects in a Plate in Elastic Vector Models

Let us consider the inverse problem of ultrasonic tomography for the velocity-stress equations for detecting internal defects in a plate. Figure 1 shows a diagram of ultrasonic tomography of a plate. A defect with three unknown parameters of the medium $(\mu(\boldsymbol{r}), \lambda(\boldsymbol{r}), \rho(\boldsymbol{r}))$ occupies a region G. A plate Ω has a known constant thickness d_0 and parameters $(\mu_0, \lambda_0, \rho_0)$ outside the defect G. Ultrasound emitters and detectors are placed around the defect G on the surface of the plate. The emitters are located at points $\boldsymbol{q_j}$ with a total of M emitter positions $j = 1, \ldots, M$. Wave field $\boldsymbol{v}(t, \boldsymbol{r}; \boldsymbol{q_j})$ is measured at the boundary S surrounding the region G. Let $\boldsymbol{V}(t, \boldsymbol{s}; \boldsymbol{q_j})$ denote the experimental data collected for detector position $s \in S$, source position $\boldsymbol{q_j}$, $j = 1, \ldots, M$, over the time interval $[0, T]$. A known function $\boldsymbol{f}(t, \boldsymbol{r})$ describes the sounding pulse. In the inverse problem, the goal is to determine the unknown parameters of the medium $(\mu(\boldsymbol{r}), \lambda(\boldsymbol{r}), \rho(\boldsymbol{r}))$ for $\boldsymbol{r} \in G$, knowing the measured experimental data $\boldsymbol{V}(t, \boldsymbol{s}; \boldsymbol{q_j})$. The displacement function $\boldsymbol{v}(t, \boldsymbol{r}; \boldsymbol{q_j}; \mu, \lambda, \rho)$ is obtained by solving the main problem (1)–(4) for velocities and stresses, followed by integrating the velocity function using the formula $\boldsymbol{v}(t, \boldsymbol{r}) = \int_0^t \boldsymbol{u}(t, \boldsymbol{r}) dt$. The wave field $\boldsymbol{v}(t, \boldsymbol{s}; \boldsymbol{q_j}; \mu, \lambda, \rho)$ computed for the parameters $(\mu(\boldsymbol{r}), \lambda(\boldsymbol{r}), \rho(\boldsymbol{r}))$ must satisfy the equation $\boldsymbol{v}\left(t, \boldsymbol{s}; \boldsymbol{q_j}; \overline{\mu}, \overline{\lambda}, \overline{\rho}\right) = \boldsymbol{V}(t, \boldsymbol{s}; \boldsymbol{q_j})$, where $\boldsymbol{s} \in S$, for all source positions $\boldsymbol{q_j}$ and for the exact (unknown) values of the parameters $(\overline{\mu}(\boldsymbol{r}), \overline{\lambda}(\boldsymbol{r}), \overline{\rho}(\boldsymbol{r}))$.

Let us introduce the residual functional $\Phi(\mu, \lambda, \rho)$ of the arguments $(\mu(\boldsymbol{r}), \lambda(\boldsymbol{r}), \rho(\boldsymbol{r}))$, which is the difference between the experimental data and the computed data

$$\Phi(\mu, \lambda, \rho) = \sum\nolimits_{j=1}^{M} \frac{1}{2} \int_0^T \int_S (\boldsymbol{v}(t, \boldsymbol{s}, \boldsymbol{q_j}; \mu, \lambda, \rho) - \boldsymbol{V}(t, \boldsymbol{s}; \boldsymbol{q_j}))^2 ds dt. \tag{6}$$

For multiple wave sources, the residual functional is the sum of the values $j = 1, \ldots, M$ obtained for each source. For each fixed source j, the integrals over the time interval $[0, T]$ and over the boundary S are calculated for all the detectors in S. Mathematically, the inverse problem is posed as the problem of finding functions $(\overline{\mu}(\boldsymbol{r}), \overline{\lambda}(\boldsymbol{r}), \overline{\rho}(\boldsymbol{r}))$ that minimize the residual functional (6) $(\overline{\mu}(\boldsymbol{r}), \overline{\lambda}(\boldsymbol{r}), \overline{\rho}(\boldsymbol{r})) : \min\limits_{\mu, \lambda, \rho} \Phi(\mu, \lambda, \rho) = \Phi(\overline{\mu}, \overline{\lambda}, \overline{\rho})$. The functions $(\overline{\mu}(\boldsymbol{r}), \overline{\lambda}(\boldsymbol{r}), \overline{\rho}(\boldsymbol{r}))$ constitute an approximate solution to the inverse problem.

To minimize the residual functional, we use gradient descent methods. A rigorous mathematical formulation for the gradient of the residual functional $\Phi(\mu, \lambda, \rho)$ with respect to parameters (μ, λ, ρ) was obtained in [15] and has the following form:

$$\begin{aligned}
\Phi_\mu(\boldsymbol{r}; \mu, \lambda, \rho) &= \sum\nolimits_{j=1}^{M} \int_0^T \boldsymbol{\varepsilon}(\boldsymbol{v}(t, \boldsymbol{r}; \boldsymbol{q_j}; \mu, \lambda, \rho)) : \boldsymbol{\varepsilon}(\boldsymbol{h}(t, \boldsymbol{r}; \boldsymbol{q_j}; \mu, \lambda, \rho)) dt, \\
\Phi_\lambda(\boldsymbol{r}; \mu, \lambda, \rho) &= \sum\nolimits_{j=1}^{M} \int_0^T \operatorname{div}(\boldsymbol{v}(t, \boldsymbol{r}; \boldsymbol{q_j}; \mu, \lambda, \rho)) \cdot \operatorname{div}(\boldsymbol{h}(t, \boldsymbol{r}; \boldsymbol{q_j}; \mu, \lambda, \rho)) dt, \\
\Phi_\rho(\boldsymbol{r}; \mu, \lambda, \rho) &= \sum\nolimits_{j=1}^{M} \int_0^T (\boldsymbol{v_t}(t, \boldsymbol{r}; \boldsymbol{q}_j; \mu, \lambda, \rho), \boldsymbol{h_t}(t, \boldsymbol{r}; \boldsymbol{q_j}; \mu, \lambda, \rho)) dt.
\end{aligned} \tag{7}$$

Here, $\boldsymbol{\varepsilon}(\boldsymbol{v}(t, \boldsymbol{r})) = (D\boldsymbol{v}(t, \boldsymbol{r}) + (D\boldsymbol{v}(t, \boldsymbol{r}))^T)/2$; $D\boldsymbol{v}$ is the Jacobian of $\boldsymbol{v}(t, \boldsymbol{r})$ with respect to $\boldsymbol{r}$; and similarly for $\boldsymbol{h}(t, \boldsymbol{r})$. In the first line, the operator $(A : B) = \sum_{ij} A_{ij} B_{ij}$ denotes the element-wise scalar product of two matrices, and in the third line $(\cdot, \cdot)$ is the scalar product of vectors from $\boldsymbol{R}^3$. Equation (7) uses the displacement functions $\boldsymbol{v}(t, \boldsymbol{r}; \boldsymbol{q_j}; \mu, \lambda, \rho)$ and $\boldsymbol{h}(t, \boldsymbol{r}; \boldsymbol{q_j}; \mu, \lambda, \rho)$. The displacement function $\boldsymbol{v}(t, \boldsymbol{r}; \boldsymbol{q_j}; \mu, \lambda, \rho)$ is obtained by solving the main problem (1)–(4)

for velocities and stresses, followed by integrating the velocity function using the formula

$$\boldsymbol{v}(t,\boldsymbol{r}) = \int_0^t \boldsymbol{u}(t,\boldsymbol{r})dt, \tag{8}$$

where the initial conditions of rest are used for $\boldsymbol{u}(t,\boldsymbol{r})$. Function $\boldsymbol{h}(t,\boldsymbol{r};\boldsymbol{q_j};\mu,\lambda,\rho)$ is obtained from the solution of the following adjoint velocity-stress problem (9)–(10) for the given values (μ,λ,ρ) followed by integration of the velocity function using formula (11).

$$\begin{cases} \rho(\boldsymbol{r})\frac{\partial \boldsymbol{\omega}(t,\boldsymbol{r})}{\partial t} - \text{div } \boldsymbol{\sigma}(t,\boldsymbol{r}) = \boldsymbol{u}(t,\boldsymbol{s};\boldsymbol{q_j};\mu,\lambda,\rho) - \boldsymbol{U}(t,\boldsymbol{s};\boldsymbol{q_j}), \\ \frac{\partial \boldsymbol{\sigma}(t,\boldsymbol{r})}{\partial t} = 2\mu(\boldsymbol{r})\frac{\partial \boldsymbol{\varepsilon}(t,\boldsymbol{r})}{\partial t} + \lambda(\boldsymbol{r})\text{div } \boldsymbol{\omega}(t,\boldsymbol{r})\ I. \end{cases} \tag{9}$$

$$\omega(t=T,\boldsymbol{r},\boldsymbol{q_j};\mu,\lambda,\rho)=0, \quad \sigma(t=T,\boldsymbol{r},\boldsymbol{q_j};\mu,\lambda,\rho)=0, \quad \omega(t,\boldsymbol{r})\mid_\Gamma = 0, \quad \sigma(t,\boldsymbol{r})\boldsymbol{n}\mid_{\partial\Omega\setminus\Gamma} = 0. \tag{10}$$

$$\boldsymbol{h}(t,\boldsymbol{r},\boldsymbol{q_j};\mu,\lambda,\rho) = -\int_t^T \boldsymbol{\omega}(t,\boldsymbol{r},\boldsymbol{q_j};\mu,\lambda,\rho)dt. \tag{11}$$

Here, $\boldsymbol{\varepsilon}(t,\boldsymbol{r})=(D\boldsymbol{\omega}(t,\boldsymbol{r})+(D\boldsymbol{\omega}(t,\boldsymbol{r}))^T)/2$; $D\boldsymbol{\omega}$ is the Jacobian of $\boldsymbol{\omega}(t,\boldsymbol{r})$; I is the identity matrix.

For numerical integration over time in formulas (8) in order to obtain $\boldsymbol{v}(t,\boldsymbol{r},\boldsymbol{q_j};\mu,\lambda,\rho)$, it is convenient to use the formula for integration in reverse time $\boldsymbol{v}(t,\boldsymbol{r}) = \int_0^t \boldsymbol{u}(t,\boldsymbol{r})dt = \int_0^T \boldsymbol{v}(t,\boldsymbol{r})dt - \int_t^T \boldsymbol{v}(t,\boldsymbol{r})dt$, where the value $\int_0^T \boldsymbol{v}(t,\boldsymbol{r})dt$ is computed in advance during solving the main problem (1)–(4). In order to compute the gradient using formulas (7), the functions $\boldsymbol{v}_t(t,\boldsymbol{r},\boldsymbol{q_j};\mu,\lambda,\rho)$ and $\boldsymbol{h}_t(t,\boldsymbol{r},\boldsymbol{q_j};\mu,\lambda,\rho)$ are replaced by $\boldsymbol{u}(t,\boldsymbol{s},\boldsymbol{q_j};\mu,\lambda,\rho)$ and $\boldsymbol{\omega}(t,\boldsymbol{r},\boldsymbol{q_j};\mu,\lambda,\rho)$, respectively, which are obtained explicitly from the solution of the main and adjoint problems.

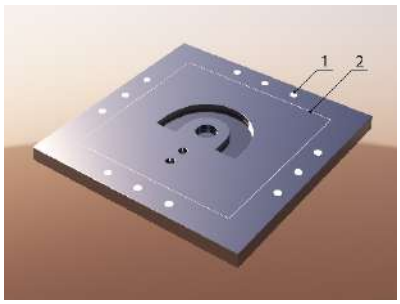
If the lateral boundaries of the plate Ω are located sufficiently far away, it is convenient to replace boundary condition (3) $\boldsymbol{u}(t,\boldsymbol{r})\mid_\Gamma = 0$ with some boundary transparency condition on Γ [23] in the computations of the main and adjoint problems. Iterative gradient-descent methods have proven highly effective in solving inverse coefficient problems. It is expected that the inverse problem in the presented formulation can be solved using high-performance computers.

3. Statement of the Inverse Problem of Ultrasonic Tomography for Detecting Surface Defects in a Plate Using a Scalar Model

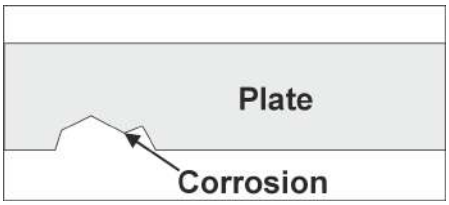
To solve the inverse problem of detecting defects inside a plate rigorously, it is necessary to formulate the inverse problem in a three-dimensional vector model, as described in Section 2. However, such a formulation is associated with a huge amount of computations, which is a demanding task even for HPC systems.

Figure 2a shows a tomographic scheme for examining surface defects on a plate. Let the defect represent a localized change in plate thickness $d(\boldsymbol{r})$; outside the defect, the plate thickness is constant and equal to d_0 (Fig. 2b). Ultrasonic radiation sources 1 are located around the defect. The wave field is recorded on line 2. The case where the defect is located on the side opposite to the sources and detectors (lower side on Fig. 2a) is of greatest interest.

Based on a number of considerations, among the various Lamb wave modes, the A_0 mode [24] is suitable for tomographic imaging. Let us consider a simplified formulation of the inverse problem for detecting surface defects. For a plate thickness that smoothly varies in the region



(a) location of sources 1 and detectors 2



(b) mathematical model of a 2D inverse problem of plate thickness reconstruction $d(\boldsymbol{r})$

Figure 2. Scheme of tomographic diagnostics of surface defects

of defects, the propagation of Lamb waves of A_0 mode is approximately described by a two-dimensional scalar wave model with varying wave velocity in the defect region. We introduce a two-dimensional coordinate system XY on the lower surface of the plate, where the sources and detectors are located. A point $\boldsymbol{r}$ on the surface thus has the coordinates $\boldsymbol{r} = (x, y) \in \boldsymbol{R}^2$. The propagation of vertical oscillations of A_0 mode Lamb waves on the lower surface of the plate is approximately described by a two-dimensional scalar model. Since the wave velocity of A_0 mode depends on the frequency, we use the Helmholtz equations for a certain set of frequencies from the spectrum of the sounding pulse

$$\Delta_r \boldsymbol{u}(w_i, \boldsymbol{r}, \boldsymbol{q_j}) + (w_i / c(\boldsymbol{r}, w_i, d(\boldsymbol{r})))^2 \boldsymbol{u}(w_i, \boldsymbol{r}, \boldsymbol{q_j}) = F\ \delta(\boldsymbol{r} - \boldsymbol{q_j}), \tag{12}$$

where $\boldsymbol{u}(w_i, \boldsymbol{r}, \boldsymbol{q_j})$ is a scalar wave at a point $\boldsymbol{r} \in \boldsymbol{R}^2$ on the lower surface of the plate; w_i is the sounding frequency $(i = 1, \ldots, N)$; $\boldsymbol{q_j} \in \boldsymbol{R}^2$ $(j = 1, \ldots, M)$ is the position of the source on the lower surface of the plate; $c(\boldsymbol{r}, w_i, d(\boldsymbol{r}))$ is the velocity of A_0 mode waves, which depends on the frequency w_i and plate thickness $d(\boldsymbol{r})$ at a point $\boldsymbol{r}$ in accordance with the dispersion relation [24]. F is the amplitude of the sounding wave. Non-reflecting boundary condition is applied at the boundary of the computational domain.

The inverse 2D problem is to determine the thickness $\overline{d}(\boldsymbol{r})$ that minimizes the residual functional $\Phi(d(\boldsymbol{r}))$, which depends on the thickness $d(\boldsymbol{r})$

$$\Phi(d(\boldsymbol{r})) = \sum\nolimits_{j=1}^{M} \sum\nolimits_{i=1}^{N} \int_S (\boldsymbol{u}(w_i, \boldsymbol{s}, \boldsymbol{q_j}, d(\boldsymbol{r})) - \boldsymbol{U}(w_i, \boldsymbol{s}, \boldsymbol{q_j}))^2 ds. \tag{13}$$

Here, $\boldsymbol{U}(w_i, \boldsymbol{s}, \boldsymbol{q_j})$ are the experimental data, which represent vertical oscillations of the A_0 mode Lamb wave at the detector locations $\boldsymbol{s}$. The gradient of the residual functional can be computed explicitly

$$\frac{\delta \Phi(d(\boldsymbol{r}))}{\delta d(\boldsymbol{r})} = \sum_{i=1}^{N} \frac{2c_0^2(w_i)}{c^3(w_i, d(\boldsymbol{r}))} \frac{\delta c(w_i, d(\boldsymbol{r}))}{\delta d(\boldsymbol{r})} \sum_{j=1}^{M} \left(\frac{w_i}{c_0(w_i)} \right)^2 \mathrm{Re}(\boldsymbol{u}(w_i, \boldsymbol{r}, \boldsymbol{q_j}, d(\boldsymbol{r})) \boldsymbol{z}(w_i, \boldsymbol{r}, \boldsymbol{q_j}, d(\boldsymbol{r}))), \tag{14}$$

where $\boldsymbol{z}(w_i, \boldsymbol{r}, \boldsymbol{q}, d(\boldsymbol{r}))$ is the solution to the adjoint problem [25]; $c(w_i, \boldsymbol{d}(\boldsymbol{r}))$ and its derivative is determined from the dispersion relation [24]. Knowing the gradient, we can use iterative gradient-descent methods for minimizing the residual functional in order to solve the inverse problem.

4. Model Computations

4.1. Numerical Method for Solving the Direct Problem of Elastic Wave Propagation in a Solid Body

For the numerical solution of the system of equations of the dynamic elasticity theory (1), a standard explicit finite-difference scheme on staggered grids is used [26–28]. The grid consists of integer and half-integer nodes. We will use the following notation

$$(x_1)_i = ih_1, \quad (x_1)_{i+1/2} = (i+1/2)h_1, \quad (x_2)_j = jh_2, \quad (x_2)_{j+1/2} = (j+1/2)h_2,$$
$$(x_3)_k = kh_3, \quad (x_3)_{k+1/2} = (k+1/2)h_3, \quad t^n = n\tau, \qquad t^{n+1/2} = (n+1/2)\tau,$$

where h_m is the grid step along the coordinate x_m, and τ is the time step. Discretized functions at integer and half-integer grid points are introduced

$$(u_1)^n_{i+1/2,j,k} = u_1((x_1)_{i+1/2}, (x_2)_j, (x_3)_k, t^n), \quad (u_2)^n_{i,j+1/2,k} = u_2((x_1)_i, (x_2)_{j+1/2}, (x_3)_k, t^n),$$
$$(u_3)^n_{i,j,k+1/2} = u_3((x_1)_i, (x_2)_j, (x_3)_{k+1/2}, t^n), \quad (\sigma_{11})^{n+1/2}_{i,j,k} = \sigma_{11}((x_1)_i, (x_2)_j, (x_3)_k, t^{n+1/2}),$$
$$(\sigma_{22})^{n+1/2}_{i,j,k} = \sigma_{22}((x_1)_i, (x_2)_j, (x_3)_k, t^{n+1/2}), \quad (\sigma_{33})^{n+1/2}_{i,j,k} = \sigma_{33}((x_1)_i, (x_2)_j, (x_3)_k, t^{n+1/2}),$$
$$(\sigma_{23})^{n+1/2}_{i,j+1/2,k+1/2} = \sigma_{23}((x_1)_i, (x_2)_{j+1/2}, (x_3)_{k+1/2}, t^{n+1/2}),$$
$$(\sigma_{13})^{n+1/2}_{i+1/2,j,k+1/2} = \sigma_{13}((x_1)_{i+1/2}, (x_2)_j, (x_3)_{k+1/2}, t^{n+1/2}),$$
$$(\sigma_{12})^{n+1/2}_{i+1/2,j+1/2,k} = \sigma_{12}((x_1)_{i+1/2}, (x_2)_{j+1/2}, (x_3)_k, t^{n+1/2}).$$

To construct an explicit finite-difference scheme, the following approximations for time derivatives are used $D_t\left[f\right]^N_{I,J,K} = \left(f^{N+1/2}_{I,J,K} - f^{N-1/2}_{I,J,K}\right)/\tau$, where f is the differentiated function. Lowercase subscripts denote integer values, while uppercase subscripts are used for both integer and half-integer values. Spatial derivative of f with respect to x_1 is approximated as $D_1\left[f\right]^N_{I,J,K} = \left(f^N_{I+1/2,J,K} - f^N_{I-1/2,J,K}\right)/\tau$.

Approximations of other spatial derivatives are constructed similarly. The numerical approximation of the system of equations (1) takes the following form:

$$\begin{aligned}
&\rho_{i+1/2,j,k} D_t[u_1]^{n-1/2}_{i+1/2,j,k} = D_1[\sigma_{11}]^{n-1/2}_{i+1/2,j,k} + D_2[\sigma_{12}]^{n-1/2}_{i+1/2,j,k} + D_3[\sigma_{13}]^{n-1/2}_{i+1/2,j,k},\\
&\rho_{i,j+1/2,k} D_t[u_2]^{n-1/2}_{i+1/2,j,k} = D_1[\sigma_{12}]^{n-1/2}_{i,j+1/2,k} + D_2[\sigma_{22}]^{n-1/2}_{i,j+1/2,k} + D_3[\sigma_{23}]^{n-1/2}_{i,j+1/2,k},\\
&\rho_{i,j,k+1/2} D_t[u_3]^{n-1/2}_{i,j,k+1/2} = D_1[\sigma_{13}]^{n-1/2}_{i,j,k+1/2} + D_2[\sigma_{23}]^{n-1/2}_{i,j,k+1/2} + D_3[\sigma_{33}]^{n-1/2}_{i,j,k+1/2},\\
&D_t[\sigma_{11}]^n_{i,j,k} = (\lambda_{i,j,k} + 2\mu_{i,j,k}) D_1[u_1]^n_{i,j,k} + \lambda_{i,j,k} D_2[u_2]^n_{i,j,k} + \lambda_{i,j,k} D_3[u_3]^n_{i,j,k} + [f_{11}]^n_{i,j,k},\\
&D_t[\sigma_{22}]^n_{i,j,k} = \lambda_{i,j,k} D_1[u_1]^n_{i,j,k} + (\lambda_{i,j,k} + 2\mu_{i,j,k}) D_2[u_2]^n_{i,j,k} + \lambda_{i,j,k} D_3[u_3]^n_{i,j,k} + [f_{22}]^n_{i,j,k},\\
&D_t[\sigma_{33}]^n_{i,j,k} = \lambda_{i,j,k} D_1[u_1]^n_{i,j,k} + \lambda_{i,j,k} D_2[u_2]^n_{i,j,k} + (\lambda_{i,j,k} + 2\mu_{i,j,k}) D_3[u_3]^n_{i,j,k} + [f_{33}]^n_{i,j,k},\\
&D_t[\sigma_{23}]^n_{i,j+1/2,k+1/2} = \mu_{i,j+1/2,k+1/2}(D_2[u_3]^n_{i,j+1/2,k+1/2} + D_3[u_2]^n_{i,j+1/2,k+1/2}) + [f_{23}]^n_{i,j+1/2,k+1/2},\\
&D_t[\sigma_{13}]^n_{i+1/2,j,k+1/2} = \mu_{i+1/2,j,k+1/2}(D_3[u_1]^n_{i+1/2,j,k+1/2} + D_1[u_3]^n_{i+1/2,j,k+1/2}) + [f_{13}]^n_{i+1/2,j,k+1/2},\\
&D_t[\sigma_{12}]^n_{i+1/2,j+1/2,k} = \mu_{i+1/2,j+1/2,k}(D_1[u_2]^n_{i+1/2,j+1/2,k} + D_2[u_1]^n_{i+1/2,j+1/2,k}) + [f_{12}]^n_{i+1/2,j+1/2,k}.
\end{aligned} \tag{15}$$

The stability condition for scheme (15) has the form of $c_p\sqrt{\left(\frac{1}{h_1^2} + \frac{1}{h_2^2} + \frac{1}{h_3^2}\right)} \leq 1$, where c_p is the longitudinal wave velocity in the material.

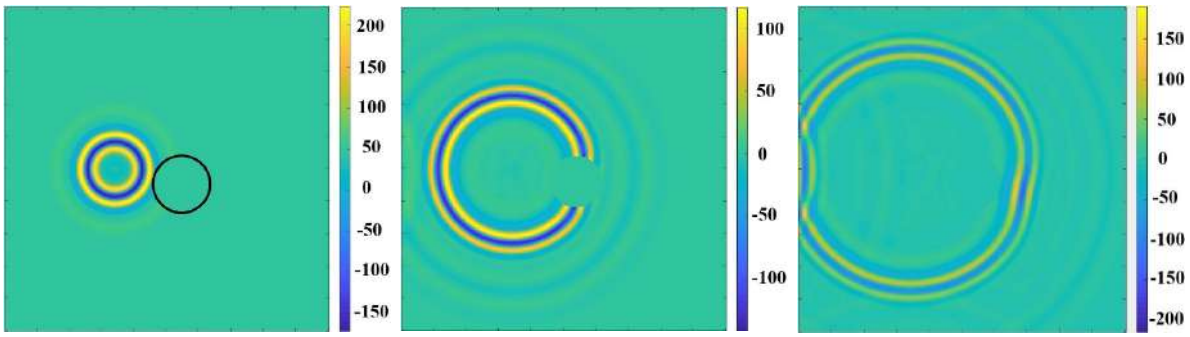
4.2. Model Computations of the Direct Problem of Wave Propagation in Thin Plates

The problem of computing the wave field in thin plates is of particular interest in this work. Due to the complex effects of interference in thin plates, the resulting waves are so-called normal waves or Lamb waves. We consider Lamb waves in plates with a thickness comparable to the wavelength. Lamb waves have different modes, each traveling at a different speed. A distinction is made between symmetric modes (denoted as Si) and asymmetric modes (denoted as Ai) of Lamb waves. In symmetric modes, the upper and lower surfaces of the plate move in opposite directions along the Z-axis perpendicular to the surfaces, while in asymmetric modes the surfaces move in the same direction.

An important feature of Lamb waves is that for a given plate thickness only the zero-order A_0 and S_0 modes are excited for frequencies below a certain threshold frequency. As the frequency increases, additional higher-order modes appear. In tomographic experiments, it is advisable to use the frequencies below this threshold frequency to avoid the appearance of additional modes. The fewer modes are present, the simpler and more reliable are the experimental data obtained. The A_0 mode is excited much more easily than the S_0 mode by striking the plate from above. Furthermore, with this type of pulse excitation, as shown by model computations, the amplitude of the A_0 mode is significantly higher than the amplitude of the S_0 mode. The propagation velocities of the A_0 and S_0 modes in the frequency range under consideration differ by approximately a factor of 2, which allows for reliable separation of these modes in processing the experimental data for solving inverse problems of thin plate diagnostics.

As an example, we present a computation of the wave field in a plate (Fig. 3) with a notch-shaped defect, with a depth smoothly increasing from zero toward the defect center as a cosine function. The parameter values used in the computation effectively excite only the A_0 Lamb wave mode:

- $\omega = 75$ kHz is the central frequency of the pulse;
- $R = 5$ mm is the radius of the circle of force application;
- L= 62.5 cm is the plate length;
- d= 1.0 cm is the plate thickness;
- $a_1 = 106.3$ mm, $b_1 = 156.3$ mm are the coordinates of the center of the force application;
- $L = 8.5$ cm; $h = 0.5$ cm are the diameter and maximum depth of the defect.



(a) 25 μs (b) 50 μs (c) 75 μs

Figure 3. Velocity u_3 on the plate surface at different moments of time

As can be seen from this figure, the defect significantly distorts the wavefront (the defect is highlighted by the black line in Fig. 3a. This distortion appears due to the fact that the residual plate thickness under the defect is 70% of the initial plate thickness, which leads to a decrease in the propagation velocity of the A_0 Lamb waves, since their velocity decreases with decreasing plate thickness.

4.3. Numerical Method for Solving the Inverse Problem of Plate Thickness Reconstruction

The inverse problem of plate thickness reconstruction was solved using a two-dimensional scalar model approximation, described by the Helmholtz equation (12) for a certain set of frequencies. To numerically solve the inverse problem of minimizing the residual functional $\Phi(d(r))$ (13), an iterative process was constructed consisting of the following steps:

1. The initial approximation at the first iteration is $d_1(r) = d_0 = const$, where d_0 is the thickness of the plate without defects.
2. For a given $d_n(r)$, the direct problem (12) of computing the wave field is solved using the finite difference approximation (16).
3. Using the simulated measured values of the wave field on the detectors $U(w_i, s, q_j)$ and computed $u(w_i, s, q_j, d(r))$ for a given $d(r)$, the adjoint problem is solved in finite difference approximation.
4. Using the obtained values $z(w_i, r, q_j, d(r))$ and $u(w_i, r, q_j, d(r))$, the gradient of the residual functional $\Phi_d'(d(r))$ is computed by formula (14).
5. Knowing the gradient, we compute the next iterative approximation using the formula $d_{n+1}(r) = d_n(r) - \gamma(n)\,\Phi_d'(d(r))$. The step size at the first iteration is determined based on prior considerations. If the value of the residual functional increases at the next iteration, $\gamma(n)$ is reduced by a factor of 2. The process returns to step 2.

The Helmholtz equation is discretized on a uniform rectangular grid, and the corresponding derivatives are approximated by finite differences using standard formulas. The discretized Helmholtz equation takes the following form:

$$\frac{u_{i+1j} - 2u_{ij} + u_{i-1j}}{\Delta x^2} + \frac{u_{ij+1} - 2u_{ij} + u_{ij-1}}{\Delta y^2} + \frac{\omega}{c^2(\omega,\ d(r_{ij}))} u_{ij} = F_{i,j}. \qquad (16)$$

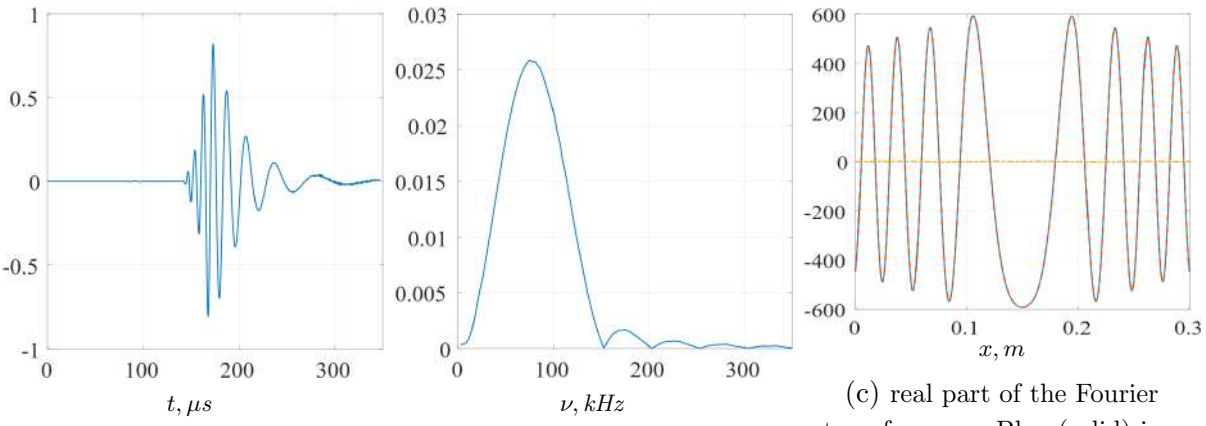
The system of algebraic equations is constructed as the indices $i,\ j$ run through all the values within the workspace. The matrix of this system is sparse and stored in CSR format. The Eigen library is used for sparse matrix manipulation and solving the problem. We will discuss the implementation details in Section 5.

4.4. Model Computations of the Inverse Problem of Plate Thickness Reconstruction

A series of computational experiments were conducted on solving the inverse problem of surface defect imaging. The experiment involved solving the forward problem using a three-dimensional vector wave model and recording the detector data. These data were then used as simulated experimental data to solve the inverse problem for a two-dimensional scalar wave equation.

The direct problem was solved for a three-dimensional thin steel plate (Fig. 2) with a thickness d and a size of $L \times L$. Free boundary conditions were applied on the upper and lower boundaries. The size of the computational domain and the locations of the sources and detectors were chosen such that the wave reflected from the boundary of the computational domain arrived at the detector sufficiently later than the wave reflected from the defect. A square workspace is located in the center of the computational domain. Defects, which are cylindrical depressions on the underside of the plate, are located within the workspace. Sources and detectors are located at the edges of the workspace on the upper side of the plate.

The sources generate sounding pulses defined by formula (5). The detectors are uniformly distributed around the perimeter of the workspace and record the vertical component of the displacement velocity $u_3(t, r)$ on the upper side of the plate. Data is recorded with the same time step as in the forward problem calculation. For each source-detector pair, the data recorded is a function of time. For each detector, the data undergoes a Fourier transform, and the result is used as $U(w, s; q_j)$ to solve the inverse problem for the Helmholtz equation. An example of a recorded signal is shown in (Fig. 4a), and a plot of the Fourier transform modulus for one detector is shown in (Fig. 4b).



(a) example of a recorded signal (b) spectrum of a recorded signal

(c) real part of the Fourier transform. u_3. Blue (solid) is a vector model, red (dashed) is a scalar model, yellow is difference

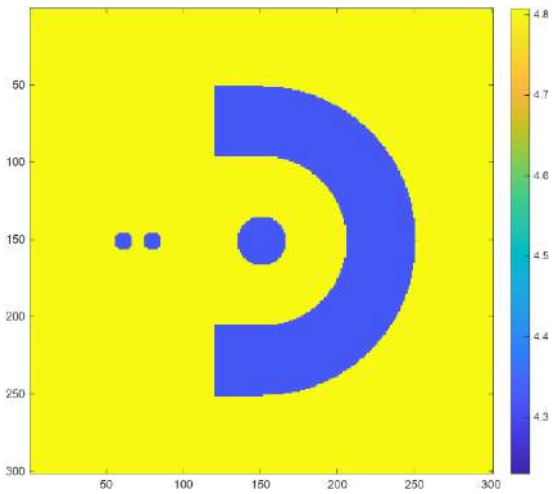
Figure 4. Signal and its spectrum and comparison u_3 in vector model with u_3 in scalar model

A key aspect in solving the inverse problem in a scalar two-dimensional wave model described by the Helmholtz equation is the correct selection of the amplitude and phase of the source perturbation and the wave propagation velocity for each harmonic. These parameters cannot be obtained explicitly with the required accuracy, even if the source and medium parameters in the forward three-dimensional problem for an elastic vector medium are known. In a real physical experiment, these parameters are unknown or approximate, making the case even worse.

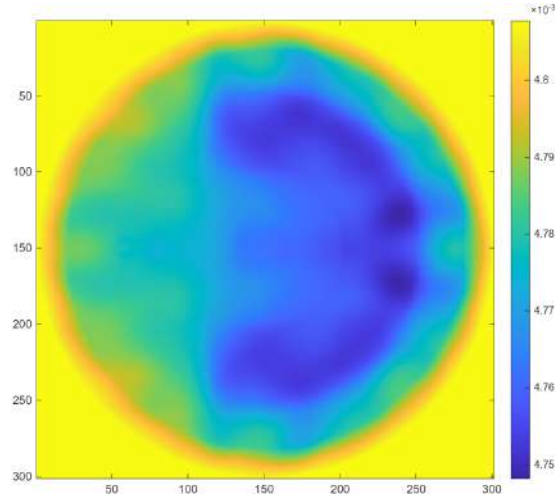
In order to determine the missing parameters, data fitting was performed on a single-source problem with standardly positioned detectors on a defect-free plate. The vector model was used to compute the wave field at the detectors, perform a Fourier transform, and obtain complex-valued data at the detectors for the selected frequency. The phase, velocity, and amplitude of the harmonics in the scalar model were determined by minimizing the norm of the difference between the complex-valued data at the detectors for a given frequency in the scalar and vector problems.

Figure 4c shows the data fitting error between the two models. Blue line represents the data from the vector model, red line represents the data from the scalar model, and yellow line represents their difference. The root-mean-square fitting error is approximately 0.02%. The procedure for determining these parameters was performed for each frequency used in solving the inverse problem.

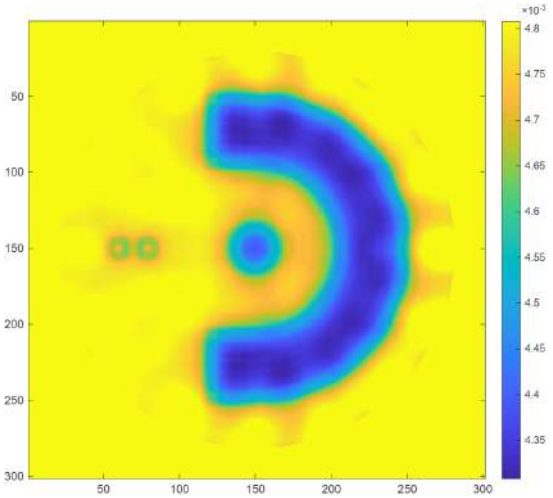
To demonstrate the algorithm for solving the inverse problem of plate thickness imaging, a computational experiment was conducted with 12 sources located 10 cm inwards from the midpoints of the sides of a 30×30 cm square computational domain. 1200 detectors were placed equally spaced around the perimeter of the square. The computations used 41 frequencies in the range of 23–147 kHz. The simulated sample object contains a collection of defects in the form of cylindrical depressions.



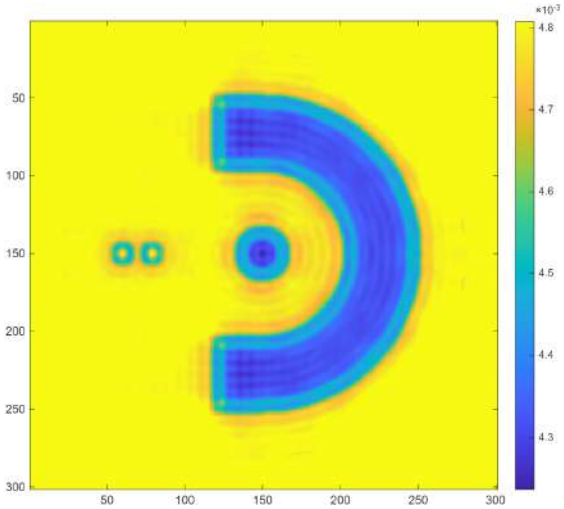
(a) sample



(b) gradient at the first iteration



(c) reconstructed image after 50 iterations



(d) reconstructed image after 140 iterations

Figure 5. Simulation experiment

Figure 5 shows the sample, the gradient at the first iteration, and the reconstructed image after 50 and 140 iterations. The results demonstrate that defects are localized with high accuracy, and their shape is precisely reconstructed. A large number of model simulations were performed with defects of varying shapes and depths. These simulations demonstrated that the developed

algorithm reliably detects both point-like and extended defects, precisely reconstructs their shape and depth, and has high spatial resolution.

5. Performance Analysis of a Supercomputer Implementation for Solving Inverse Problems of Ultrasound Tomography Using the Lamb Wave

Solving the inverse problems under consideration requires huge computational resources. During the iterative solution process, the Helmholtz differential equation (12) must be solved for both the direct and the adjoint problem at each iteration, for each source position, and for each frequency used. Typical numbers of sources and frequencies can vary from a few to several dozen. For the model computations presented in Section 4.4, 12 sources and 41 frequencies were used, in which case the Helmholtz equation must be solved $12 \times 41 \times 2 = 984$ times for every iteration.

A finite-difference approximation was used for the Helmholtz equation, resulting in a system of linear algebraic equations (SLAE), the solution of which is the most computationally expensive and time-consuming task. Third-party linear algebra libraries were used for SLAE solution. The SLAE matrix has a size equal to the number of grid points and is highly sparse, with a fill-factor of approximately 0.0008%. The matrix is stored in CSR format. For the model computations from Section 4.4, the matrix size was approximately $6.4 \cdot 10^5 \times 6.4 \cdot 10^5$ with the number of nonzero elements approximately 3.2×10^6. Note that the matrix in this problem is not symmetric.

An inverse problem solution algorithm was implemented using C++ language plus MPI for parallelization, and Eigen library [29] to fill in the matrix, the right-hand side, and solving the SLAE. In this Section, we analyze the performance of the developed application for solving inverse problems. The computations were performed in the "compute" section of the Lomonosov-2 supercomputer [30]. In this section, each node contains a single 14-core Intel Xeon E5-2697 v3 CPU with 64 GB of RAM. The application was run on 164 processes. Intel MPI 2017 was used to compile and run the distributed MPI version. For this performance analysis, the program was configured to run for ∼5 minutes, this was achieved by reducing the number of iterations. The iterations are essentially identical, so the overall picture in this case matches the program's behavior when using the number of iterations used for real-life calculations.

First, it was necessary to determine the optimal number of processes per node. This program is quite memory-intensive, and in such cases, it is sometimes beneficial to occupy only a part of processor cores in order to reduce memory contention. Experiments with 8 to 14 processes per node were performed in the "compute" section, with 10 launches of every experiment performed in order to obtain enough statistics. A smaller number of processes per node was not considered in this case because it involves too many compute nodes, which was considered inexpedient.

Table 1 shows the minimum execution times obtained. We consider the minimum value, not average or mean, because supercomputers often experience different external factors that can slow down a user program (such as influence of other users' applications running concurrently, the overload of distributed file system, OS noise, etc.). This leads to periodic anomalous or simply slower executions, which have notably longer execution times and are of no interest in our case. But we note that the average execution time behaves generally similarly to the minimum time. Considering the minimum time allows us to estimate how quickly a program can potentially execute under the chosen parameters, without the influence of external factors.

Table 1. Program execution time with different numbers of processes per node

Processes per node	8	9	10	11	12	13	14
Minimum execution time, s	360	360	294	222	222	228	228

It can be seen that the program executes the slowest at 8–9 processes per node, then the execution time begins to decrease, reaching a minimum at 11–12 processes per node. This is presumably due to the fact that fewer compute nodes are required as the number of processes per node increases, and, as a result, the amount of data transferred between nodes decreases (i.e., the communication decreases). The execution time then increases slightly, likely due to the aforementioned effect of memory contention between processes on the node. Thus, the minimum execution time is achieved at 11–12 processes per node. However, it should be taken into account that when running user programs on a supercomputer, a queuing system is used, and the more nodes a program requires, the longer it will usually wait in the queue for the required number of nodes to become available. Given that the difference in execution time between the versions with 11–12 and 14 processes per node is small (a slowdown of ∼3%), but fewer nodes are used (difference in two nodes, or 16.7%), it is recommended to run this program on 14 processes per node, thereby utilizing all available processor cores. This will both speed up the job's progress through the queue and reduce the overall resource consumption in terms of node-hours, thereby saving resources for other users' jobs.

Next, we analyzed the program's performance, using 14 processes per node. First, we examined the MPI usage in the program. mpiP 3.5 profiling tool [31] was used for this purpose. Analysis of its results showed that program processes spend 3–10% of their execution time on MPI, which can be considered as a good result for a program running on 164 processes and 12 compute nodes. It should be noted that MPI usage characteristics do not differ significantly for individual processes, i.e., there is no significant imbalance in the program.

Approximately 70–75% of the execution time spent on MPI is accounted for by just two `MPI_Allreduce` calls, with almost all of the remaining time accounted for by two `MPI_Bcast` calls. It is interesting to note that both `MPI_Allreduce` calls are responsible for transferring a tiny amount of data – less than 0.1% of the total amount of transferred data (while `MPI_Bcast` calls are responsible for transferring the rest of the data). Therefore, if there is a need to optimize MPI usage, these `MPI_Allreduce` calls can clearly be singled out as the main candidates, since they account for the majority of the time, but they transfer very little data. One could assume that the reason for their long execution is related to imbalance caused by some processes reaching this MPI operation earlier and simply waiting for the other processes to start, but this is not always the case. In some cases, all involved processes spent significant time performing these operations. The exact reasons for this behavior are currently unknown; however, the total time spent on MPI in the current program configuration is quite small, so further detailed analysis and optimization in this direction is not of great interest at the moment (this may change in the future if a larger number of processes are involved, likely leading to the increase of MPI share in execution time).

After studying MPI usage, an analysis of the execution efficiency of one individual process was conducted. The program is generally well-balanced, so the overall results for one process presented below hold true for others as well. The analysis was performed using Intel VTune 2019.5 and Intel Advisor 2019.5.

Initially, the most general performance characteristics of parallel programs were studied:

- The average number of cycles per instruction (CPI) is ∼0.5, i.e., two instructions are executed per clock cycle on average, which is high for real-world HPC applications.
- The utilization of processor cores is also quite high, at 85–90%.
- The Retiring metric, which roughly shows the percentage of time the CPU was fully utilized by performing useful operations, is of interest as well. This metric is calculated using a Top-down approach proposed by Intel [32], the goal of which is to identify the dominant bottlenecks in an application performance. For this program, the value of this metric is 45–55%, which indicates a fairly efficient HPC program (according to Intel, an expected range of Retiring value for well-tuned HPC programs is 30–70% [32]).

Intel VTune allows collecting various useful information using the Top-down approach. However, Hyperthreading is enabled on compute nodes in the "compute" partition, making some Top-down metrics unavailable in VTune. Therefore, to analyze these metrics, we conducted special launches of the program in the "pascal" partition of the Lomonosov-2 supercomputer. The nodes in this partition are equipped with a different processor – a 12-core Intel Xeon Gold 6126. However, the comparison showed that the overall picture provided by Top-down approach almost does not change across runs in different partitions, so here we show a more complete list of metrics after running in the "pascal" partition.

Figure 6 shows the Top-down metrics for this program provided by Intel VTune. It can be seen that there are no issues with data prefetch or instruction preparation (Bad Speculation and Front-End Bound metrics, respectively). In this case, the application is classified as Back-End Bound (50%), meaning that processor slots were frequently only partially occupied, with the main reason for this being memory management (Memory Bound = 29%). It should be noted here that Core Bound in the screenshot is just slightly lower than Memory Bound, and this is the only noticeable difference from what is observed in the "compute" section (being of primary interest for us), where Core Bound is 10%, being a small value. Therefore, we do not consider this issue, since it does not occur when running in "compute".

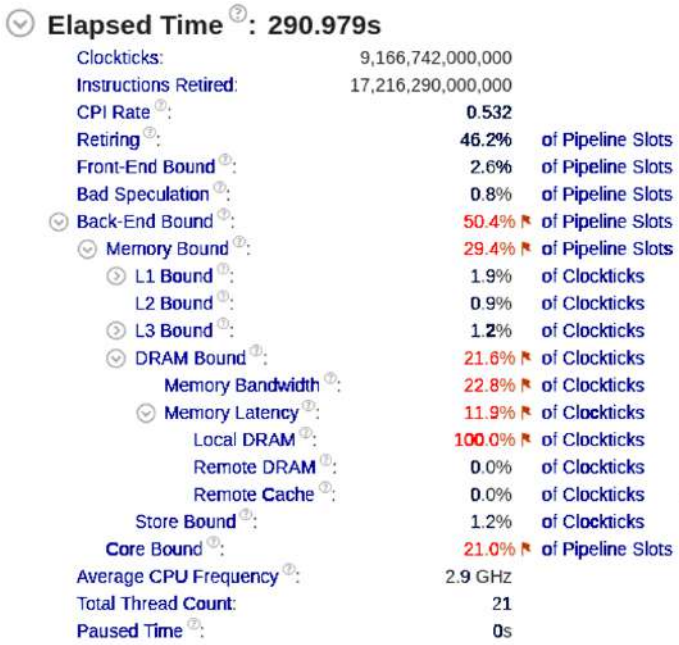


Figure 6. Top-down metrics obtained by Intel VTune

In terms of memory usage, this program exhibits no issues with any cache levels (L1/L2/L3 Bound values are very small). However, some problems with RAM can be noted. These include mainly memory bandwidth issues (the program reaches maximum bandwidth to RAM, likely causing a slowdown) as well as some memory latency issues (the program has to wait for data from RAM, as the retrieval process takes notable time). The values for these metrics are not very high, meaning the issues are apparently not that significant, but they are still present.

The presence of performance issues with RAM is as well confirmed by studying the values of performance monitoring counters also provided by VTune. They show that when accessing L3 cache, there are 2.5 misses for every hit on average (and L3 miss necessitates an access to RAM), which is a very poor indicator. Furthermore, one L3 cache miss occurs for every 200 memory accesses, meaning the L3 is used quite heavily. This suggests that memory access is organized not in the most efficient manner, leading to fairly frequent RAM accesses and, consequently, waiting for data from memory.

The vectorization level using Intel Advisor was also studied. It revealed that the program is not vectorized at all: only 0.1% of the execution time is spent in vectorized code fragments. Furthermore, vectorization does not use the newer AVX instructions, only SSE and SSE2. This, however, is currently not particularly significant given that almost all the code is scalar.

A separate study of the most computationally intensive fragments (those that consume the most time during program execution) was conducted. A general analysis on the level of modules revealed that the program spends >90% of its execution time within its own code, with another ∼5% executing MPI functions, 1.5% interacting with the OS kernel (vmlinux), and ∼1% calling the libm math library. Within the program itself, almost all of the time is spent executing the code from the aforementioned Eigen library.

Thus, the following conclusions can be drawn from the analysis performed. The program demonstrates relatively high performance, with MPI operations taking up only a small portion of the program's execution time – ∼5% of the total runtime, which is quite low for a program with 164 processes. Certain performance issues with RAM usage were noted; they do not significantly slow down the program but do represent a potential area for optimization. Vectorization is essentially absent, which is also a clear candidate for optimization (if this area is further to be explored, a detailed analysis of the non-vectorized code fragments will be required to determine the feasibility of vectorization). However, everything discussed regarding memory and vectorization applies to the usage of external Eigen library, as almost all execution time not spent on MPI is spent working with it, which significantly complicates the possibility of making changes and optimizations. One of the possible solutions in this case is to switch to using other libraries for solving SLAE like MKL.

Conclusion

This study examines the challenges of supercomputer implementation of ultrasonic tomographic imaging of internal and surface defects for nondestructive testing of solids. It has been shown that the imaging problem of ultrasonic imaging of thin plates using Lamb waves is a coefficient inverse problem, i.e., it involves determining unknown coefficients in differential equations describing wave propagation in scalar and vector models.

It has been shown that a scalar two-dimensional model is quite suitable for detecting surface defects. A simplified formulation of the inverse problem assumes that the propagation of Lamb waves of mode A_0 is approximately described by the Helmholtz equation. The experimental data

were obtained from a numerical solution of the direct problem in a 3D elastic model in time domain. An explicit expression for the residual functional gradient with respect to the plate thickness was formulated for a range of frequencies. Iterative methods have been developed to solve the inverse problem. The methods are based on gradient descent methods to minimize the residual functional. Efficiency of the proposed algorithms is illustrated on model problems.

Within this simplified model, tomographic imaging technique makes it possible to reconstruct the shape, size, and depth of defects. Solving the Helmholtz equation is the core element of the developed algorithms for solving inverse problems of wave tomography. The most demanding computations involve solving linear equations with large-scale sparse matrices using LU-decomposition method. The algorithms were implemented using linear algebra Eigen library. The performance analysis of the solution of the inverse problem in the scalar formulation of the Helmholtz equation (with Eigen library used for solving SLAE) was performed on the Lomonosov-2 supercomputer.

Acknowledgements

This research was supported by Russian Science Foundation grant No. 25-11-00089, https://rscf.ru/en/project/25-11-00089/. This research was carried out at the Lomonosov Moscow State University. The results in Section 2 were obtained within the framework of the state assignment of Lomonosov Moscow State University "Development of Methods for Solving Inverse Problems of Diagnostics and Synthesis in Wave Models". The research is carried out using the equipment of the shared research facilities of HPC computing resources at Lomonosov Moscow State University.

References

1. Ruiter, N.V., Zapf, M., Hopp, T., *et al.*: USCT data challenge. Proc. SPIE. Medical Imaging: Ultrasonic Imaging and Tomography 10139, 101391N (2017).

2. Duric, N., Littrup, P., Sak, M., *et al.*: A novel marker, based on ultrasound tomography, for monitoring early response to neoadjuvant chemotherapy. J. Breast Imaging 2(6), 569–576 (2020). https://doi.org/10.1093/jbi/wbaa084

3. Wiskin, J., Borup, D., Andre, M., *et al.*: Three-dimensional nonlinear inverse scattering: Quantitative transmission algorithms, refraction corrected reflection, scanner design, and clinical results. J. Acoust. Soc. Am. 133(5), 3229–3229 (2013). https://doi.org/10.1121/1.4805138

4. Lucka, F., Pérez-Liva, M., Treeby, B.E., Cox, B.T.: High resolution 3D ultrasonic breast imaging by time-domain full waveform inversion. Inverse Problems 38(2), 025008 (2022). https://10.1088/1361-6420/ac3b64

5. Goncharsky, A.V., Romanov, S.Y., Seryozhnikov, S.Y.: Low-frequency ultrasonic tomography: Mathematical methods and experimental results. Moscow University Physics Bulletin 74(1), 43–51 (2019). https://doi.org/10.3103/S0027134919010090

6. Goncharsky, A.V., Romanov, S.Y., Seryozhnikov, S.Y.: Supercomputer technologies in tomographic imaging applications. Supercomputing Frontiers and Innovations 3(1), 41–66 (2016). https://doi.org/10.14529/jsfi160103

7. Goncharsky, A.V., Kubyshkin, V.A., Makan, I.I., *et al.*: Supercomputer simulations in designing medical ultrasound tomographic imaging devices. Lobachevskii Journal of Mathematics 45(7), 3038–3050 (2024). https://doi.org/10.1134/S199508022460393X

8. Goncharsky, A.V., Romanov, S.Y., Seryozhnikov, S.Y.: Multistage Iterative Method to Tackle Inverse Problems of WaveTomography. Supercomputing Frontiers and Innovations 9(1), 87–107 (2022). https://doi.org/10.14529/jsfi220106

9. Goncharsky, A.V., Romanov, S.Y., Seryozhnikov, S.Y.: Computational Efficiency of Iterative Methods for Solving Inverse Problems. In: Voevodin, V., Sobolev, S., Yakobovskiy, M., Shagaliev, R. (eds.) Supercomputing. RuSCDays 2023. Lecture Notes in Computer Science, vol. 14388, pp. 35–46. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-49432-1_3

10. Bazulin, E.G., Goncharsky, A.V., Romanov, S.Y., Seryozhnikov, S.Y.: Parallel CPU- and GPU-algorithms for inverse problems in nondestructive testing. Lobachevskii Journal of Mathematics 39(4), 486–493 (2018). https://doi.org/10.1134/S1995080218040030

11. Yang, P., Brossier, R., Metivier, L., Virieux, J.: A review on the systematic formulation of 3-D multiparameter full waveform inversion in viscoelastic medium. Geophys. J. Int. 207, 129–149 (2016). https://doi.org/10.1093/gji/ggw262

12. Virieux, J., Asnaashari, A., Brossier, R., *et al.*: An introduction to full waveform inversion. Society of Exploration Geophysicists 6, R1-1–R1-40 (2014). https://doi.org/10.1190/1.9781560803027.entry6

13. Pan, W., Innanen, K.A.: Elastic full-waveform inversion: density effects, cycle-skipping, and inter-parameter mapping. CREWES Research Report 28, 62.1–62.18 (2016).

14. Li, Y., Gu, H.: Full waveform inversion for velocity and density with rock physical relationship constraints. Journal of Applied Geophysics 167, 106–117 (2019). https://doi.org/10.1016/j.jappgeo.2019.04.005

15. Lechleiter, A., Schlasche, J.W.: Identifying Lamé parameters from time-dependent elastic wave measurements. Inverse Problems in Science and Engineering 25(1), 2–26 (2017). https://doi.org/10.1080/17415977.2015.1132713

16. Plessix, R.E.: A review of the adjoint-state method for computing the gradient of a functional with geophysical applications. Geophys. J. Int. 167, 495–503 (2006). https://doi.org/10.1111/j.1365-246X.2006.02978.x

17. Bazulin, E.G., Goncharsky, A.V., Romanov, S.Y., Seryozhnikov, S.Y.: Inverse problems of ultrasonic tomography in nondestructive testing: Mathematical methods and experiment. Russian Journal of Nondestructive Testing 55(6), 453–462 (2019). https://doi.org/10.1134/S1061830919060020

18. Huthwaite, P., Simonetti, F.: High-resolution guided wave tomography. Wave Motion 50(5), 979–993 (2013). https://doi.org/10.1016/j.wavemoti.2013.04.004

19. Rao, J., Ratassepp, M., Fan, Z.: Guided Wave Tomography Based on Full Waveform Inversion. IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control 63(5), 737–745 (2016). https://doi.org/10.1109/TUFFC.2016.2536144

20. Zhao, X., Rose, J.L.: Ultrasonic guided wave tomography for ice detection. Ultrasonics 67, 212–219 (2016). https://doi.org/10.1016/j.ultras.2015.12.005

21. Tong, J., Lin, M., Wang, X., *et al.*: Deep learning inversion with supervision: A rapid and cascaded imaging technique. Ultrasonics 122, 106686 (2022). https://doi.org/10.1016/j.ultras.2022.106686

22. Landau, L.D., Lifshitz, E.M.: Course of Theoretical Physics, Vol. 7: Theory of Elasticity. Pergamon, Oxford (1995).

23. Engquist, B., Majda, A.: Absorbing boundary conditions for the numerical simulation of waves. Math. Comput. 31, 629–629 (1977). https://doi.org/10.1090/s0025-5718-1977-0436612-4

24. Viktorov, I.A.: Lamb's Ultrasonic Waves. Soviet Physics. Acoustics 11(1), 1–18 (1965).

25. Natterer, F., Sielschott, H., Dorn, O., *et al.*: Fréchet derivatives for some bilinear inverse problems. SIAM J. Appl. Math. 62(6), 2092–2113 (2002). https://doi.org/10.1137/S0036139901386375

26. Virieux, J.: P-SV Wave Propagation in Heterogeneous Media: Velocity-Stress Finite-Difference method. Geophysics 51(4), 889–901 (1986). https://doi.org/10.1190/1.1442147

27. Lisitsa, V.V.: Numerical Methods and Algorithms for Calculating Wave Seismic Fields in Media with Local Complicating Factors. Doctoral Thesis in Physics and Mathematics (Trofimuk Institute of Petroleum Geology and Geophysics SB RAS, Novosibirsk, 2017). https://www.dissercat.com/content/chislennye-metody-i-algoritmy-rascheta-volnovykh-seismicheskikh-polei-v-sredakh-s-lokalnymi, accessed: 2023-07-10

28. Belyaev, A.S., Goncharsky, A.V., Romanov, S.Y.: Development of Numerical Algorithms for Solving the Direct Problem of Propagation of Ultrasonic Waves in Thin Plates. Numerical Methods and Programming 24(3), 275–290 (2023). https://doi.org/10.26089/NumMet.v24r320 (in Russian)

29. Eigen as a C++ template library for linear algebra: matrices, vectors, numerical solvers, and related algorithms. https://eigen.tuxfamily.org/, accessed: 2025-11-14

30. Voevodin, Vl., Antonov, A., Nikitenko, D., *et al.*: Supercomputer Lomonosov-2: Large Scale, Deep Monitoring and Fine Analytics for the User Community. Supercomputing Frontiers and Innovations 6(2), 4–11 (2019). https://doi.org/10.14529/jsfi190201

31. Vetter, J., Chambreau, C.: mpiP: Lightweight, Scalable MPI Profiling (2005). http://gec.di.uminho.pt/Discip/MInf/cpd1415/PCP/MPI/mpiP_%20Lightweight,%20Scalable%20MPI%20Profiling.pdf

32. Description of Top-Down approach. https://www.intel.com/content/www/us/en/docs/vtune-profiler/cookbook/2023-0/top-down-microarchitecture-analysis-method.html, accessed: 2025-10-15