Engineering Analysis with Boundary Elements 161 (2024) 103–112

Contents lists available at ScienceDirect

Engineering Analysis with Boundary Elements

journal homepage: www.elsevier.com/locate/enganabound

# A time-domain boundary element method using a kernel-function library for 3D acoustic problems

Zhenyu Gao, Zonglin Li, Yijun Liu [*]

*Department of Mechanics and Aerospace Engineering, Southern University of Science and Technology, Shenzhen, China*

A R T I C L E   I N F O

*Keywords:*
Time-domain boundary element method
Kernel-function library
3D acoustic problems
Transient response
Tuning fork

A B S T R A C T

Time-domain boundary element method (TDBEM) can be applied in studying transient acoustic wave problems, especially in solving exterior problems. However, in order to avoid calculations of coefficients at different time steps, TDBEM usually requires storing or recomputing the coefficients at each time step. This can lead to significant amount of memory usage or long computing time. In this paper, an acoustic TDBEM based on a kernel-function library (KFL-BEM) is proposed, in order to reduce the memory consumption of the time-domain conventional BEM (CBEM) and speedup the computation. In this approach, the storage requirement is reduced from $O(N^2N_{t\min})$ to $O(N^2)$, where $N$ represents the number of degrees of freedom of the model, and $N_{t\min}$ is the minimum number of time steps for which coefficients need to be computed and stored. To demonstrate the effectiveness of the KFL-BEM, two verification examples are presented using the problems of a pulsating sphere and sound propagating in a channel. Compared with the CBEM, the KFL-BEM can save significantly the memory storage as it does not require storing coefficients for any previous time steps. This method can also be applied to solve vibro-acoustic problems in the time domain. As an example, the acoustic radiation responses of a tuning fork under different striking loads are studied using the finite element method and the proposed KFL-BEM, which clearly shows the potentials of the KFL-BEM in solving time-domain acoustic problems.

## 1. Introduction

The boundary element method (BEM) [1] is one commonly used method for solving acoustic wave problems. Compared to the finite element method (FEM) [2], the BEM does not require partitioning the entire computational domain into elements. It only discretizes the surface of a 3D domain into boundary elements. Moreover, the radiation boundary conditions are automatically satisfied at infinity [3] for exterior problems. Therefore, the BEM has been widely used in solving acoustic wave problems for several decades [4–7].

However, the coefficient matrix of the BEM is usually a dense and asymmetric matrix, which makes the BEM expensive to use. In order to improve the computational efficiency of the BEM, Rokhlin and Greengard [8] in 1980's proposed the fast multipole method (FMM), which has been widely applied to the solution of boundary element equations over the years [9–13]. In addition, based on the concept of H-matrices [14], Bebendorf proposed the adaptive cross approximation (ACA) method [15], which is easier to implement to solve the BEM equations than using the FMM although it requires more memory usage. With the advance of computing hardware, the fast direct solver methods have received widespread attention recently [16–18]. Compared to algorithms using iterative solvers such as the FMM and ACA, fast direct solvers do not suffer from non-convergence issues. All these fast solution methods have been widely applied to accelerate the acoustic BEM solutions, mainly for frequency-domain problems.

For acoustic problems, the frequency domain analysis is a common and major approach, allowing one to observe a series of acoustic behaviors related to the frequency responses of structures, material models, and so on. On the other hand, in many cases, one is also interested in the behavior of sound waves due to sources of a transient nature, such as a few impulse sources in target search of objects. In this case, using the time-domain BEM (TDBEM) to solve acoustic problems is a more natural and effective approach [19].

In contrast, the acceleration of TDBEM has not been extensively studied. Ergin et al. [20] has conducted the research on acoustic problems in time domain [21,22]. They proposed a plane wave time-domain (PWTD) algorithm for accelerating the TDBEM, which is considered to be the time-domain version of the FMM. In addition, Takahashi et al.

* Corresponding author.
*E-mail address:* liuyj3@sustech.edu.cn (Y. Liu).

https://doi.org/10.1016/j.enganabound.2024.01.001
Received 27 October 2023; Received in revised form 2 January 2024; Accepted 2 January 2024
Available online 27 January 2024

[23,24] proposed an FMM algorithm based on interpolation for time-domain acoustic BEM and also extended the method to the field of electromagnetism [25]. Aimi et al. applied the ACA method to the acoustic and elastic waves exterior problems in TDBEM [26].

Regardless of the utilization of the algorithms for acceleration, the large consumption of memory with the TDBEM is still a crucial aspect that need to be addressed. In order to avoid computing identical coefficients during different time steps, it is customary to store these coefficients with the TDBEM. Therefore, the memory requirement is $O(N^2 N_{t\min})$, where $N$ and $N_{t\min}$ represent the number of degrees of freedom in the BEM model and the minimum number of time steps that need to be computed, respectively. When $N_{t\min}$ is small, the coefficient matrix is sparse, so in many cases, the exponent of $N$ is actually less than 2. The upper bound for $N_{t\min}$ is denoted by $N_T$, which corresponds to the total number of time steps. To reduce the memory consumption, Thirard's non-uniform grid time domain method (NGTD) [27] can reduce the value of $O(N^2 N_{t\min})$to $O(N^\alpha N_{t\min})$, where α ranges from 1 to 2. Additionally, the implementation of memory-saving algorithms proposed in Ref. [28] can reduce the upper bound $N_T$ to $N_T\ /2$ without introducing new errors. Current methods to reduce memory usage mainly focus on reducing the α, however, the demand of high memory shortages can still arise in large-scale computations when the $N_{t\min}$ is large. If it is not possible to store all values of the coefficients, a partial storage of the coefficients and recalculation of non-stored coefficients can be performed. Unfortunately, this recalculation can significantly increase the computation time.

In 1998, Pan, Adams and Rizzo proposed the idea of using Green's function library in the BEM for modeling composite materials [29]. The approach can help users to quickly conduct the analysis of composite material models without much knowledge of the BEM by pre-storing the matrix or discretized Green's functions. Borrowing from this idea of the Green's function library, a method based on a kernel function library that uses the characteristics of the TDBEM kernel functions is proposed in this paper. The construction of the library is only dependent on the distances between elements and normal derivatives of elements. The library is established once and can be called directly during computation at all time steps, instead of computing and storing the coefficients at many time steps. The memory occupation with this library approach is reduced from $O(N^2 N_{t\min})$ to $O(N^2)$, which can remedy the problem of massive memory consumption in the TDBEM for solving large-scale problems.

An interesting application of the TDBEM is to facilitate the direct numerical computation of sound fields due to a vibrating structure over time. This enables us to audibly perceive the sound generated by the vibrating structure such as tapping a tuning fork, through structural acoustic analysis. In order to achieve this, numerous time steps and small-time intervals are typically required. The TDBEM using the kernel function library is capable of effectively addressing memory limitations in such applications as demonstrated in this paper.

The paper is organized as follows: In Section 2, we review the boundary integral formulations of the TDBEM and the implementation details of TDBEM based on the kernel function library (KFL-BEM). In Section 3, we verify the accuracy and effectiveness of the algorithm through numerical examples of two time-domain acoustic problems. In Section 4, we present the FEM-BEM model and computed results (including the sound playback file) of the transient vibro-acoustic analysis of a tuning fork under different striking loads. In Section 5, we provide some discussions and conclusions on this work.

## 2. TDBEM based on a kernel-function library

This section summarizes the formulations of the conventional acoustic time-domain BEM. Since the acoustic time-domain BEM requires the use of results from previous time steps to compute the boundary values at a specific time step, the computation process is time-consuming and requires a large amount of memory consumption. Therefore, the KFL-BEM is proposed to accelerate the computation process and reduce the memory consumption.

### 2.1. Time domain boundary integral equation

In a three-dimensional (3D) domain, the acoustic wave equation can be written as:

$$\nabla^2\phi(\mathbf{x},t) - \frac{1}{c^2}\frac{\partial\phi^2(\mathbf{x},t)}{\partial t^2} + \gamma(\boldsymbol{\xi},t) = 0,\ \forall\mathbf{x}\in E, \tag{1}$$

in which $\nabla^2$ is the Laplacian operator, ϕ is the sound velocity potential at point **x** and time *t*, *c* is sound speed, γ(ξ, *t*) is a possible sound source located at ξ and at time *t*. The acoustic domain *E*, which can be either a finite domain inside a closed surface *S* or an infinite domain outside the closed surface *S* (Fig. 1), is considered to be isotropic and homogenous.

If there is a unit point source, that is, γ(ξ, *t*) = δ(ξ − **y**, *t* − τ) (here γ represents a unit point and impulse sound source located at **y** at time τ), with δ being the Dirac delta function, then the solution (sound field located at **x** at time *t*) satisfies the above governing equation is called the fundamental solution *G*(**x**, *t*; **y**, τ). For 3D time-domain acoustic problems, *G*(**x**, *t*; **y**, τ) is given by the following expression [19]:

$$G(\mathbf{x},t;\mathbf{y},\tau) = \frac{1}{4\pi r}\delta\left(t-\tau-\frac{r}{c}\right), \tag{2}$$

where *r* is the distance between the field point **x** and source point **y**.

To derive the boundary integral equation (BIE) corresponding to the acoustic wave Eq. (1), we utilize the second Green's identity in conjunction with the above fundamental solution. As a result, we can obtain the following BIE:

$$c(\mathbf{x})\phi(\mathbf{x},t) = \int_0^t\int_S[G(\mathbf{x},t;\mathbf{y},\tau)q(\mathbf{y},\tau) - F(\mathbf{x},t;\mathbf{y},\tau)\phi(\mathbf{y},\tau)]d\tau dS(\mathbf{y}), \tag{3}$$

in which $c(\mathbf{x}) = 1/2$ for **x** on the boundary *S* (assume to be smooth), **y** is the source point on the boundary, $q(\mathbf{y},\tau) = \partial\phi/\partial n$ is the normal derivative of the sound velocity potential ϕ at point **y** and at time τ, $F(\mathbf{x},t;\mathbf{y},\tau) = \partial G(\mathbf{x},t;\mathbf{y},\tau)/\partial n(\mathbf{y})$ is the normal derivative of the fundamental solution *G*(**x**, *t*; **y**, τ).

Next, the expression of the time-domain fundamental solution (2) is substituted into the time-domain BIE (3). Using the filter property of the Dirac δ function distribution in time and taking the time derivative, we obtain the following form of the 3D time-domain BIE [19] :

$$\begin{aligned}\frac{1}{2}\phi(\mathbf{x},t) = &\int_S q\left(\mathbf{y},t-\frac{r}{c}\right)\ \frac{1}{4\pi r}dS(\mathbf{y}) + \int_S\phi\left(\mathbf{y},t-\frac{r}{c}\right)\ \frac{1}{4\pi r^2}\frac{\partial r}{\partial n}dS(\mathbf{y})\\ &+\int_S\frac{\partial\phi}{\partial t}\left(\mathbf{y},t-\frac{r}{c}\right)\ \frac{1}{4\pi rc}\frac{\partial r}{\partial n}dS(\mathbf{y}).\end{aligned} \tag{4}$$

Eq. (4) is used to solve the null initial condition (i.e., ϕ = *q* = 0 for *t* < 0), the boundary conditions ϕ or $q = \partial\phi/\partial n$ are given, solve for the unknown quantity on the boundary, and one can obtain the sound pressure

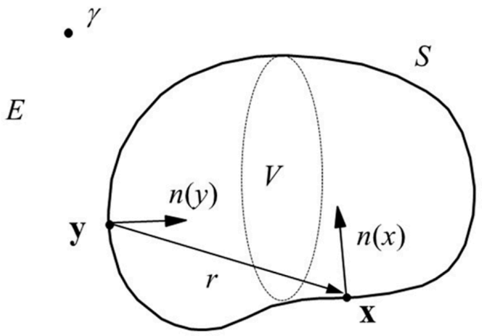


**Fig. 1.** Structure *V* and the acoustic domain *E*.

at any field point in the domain. For exterior problems, the Sommerfeld condition at infinity is satisfied by the solution of BIE (3) or BIE (4).

For solving time domain acoustic problems, the boundary can be discretized into constant, linear, quadratic or other higher-order elements. To discretize BIE (4) in this study, we use constant boundary elements on the boundary and the temporal axis is divided by a constant time step of size $\Delta t$, with the number of total time steps equal to $N_T$. With the piecewise-constant and piecewise-linear bases for space and time, respectively, the boundary variables $\phi$ and $q$ are approximated by $\phi_j$ and $q_j$ on the $j$th element $E_j$ and for $t > 0$, and are expressed as follows:

$$\phi_j(t) = \sum_{k=0}^{N_T} N_t^k(t)\phi_j^k,\ q_j(t) = \sum_{k=0}^{N_T} N_t^k(t) q_j^k, \tag{5}$$

respectively, where $\phi_j^k$ and $q_j^k$ designate the values of $\phi_j$ and $q_j$ on element $E_j$ and at the $k$th time step $t_k$, respectively. Functions $N_t^k$ are given as follows:

$$N_t^k(t) = \begin{cases} \dfrac{(t-k\Delta t)}{\Delta t}+1, & (k-1)\Delta t \le t \le k\Delta t, \\ -\dfrac{(t-k\Delta t)}{\Delta t}+1, & k\Delta t < t \le (k+1)\Delta t, \\ 0, & \text{elsewhere.} \end{cases} \tag{6}$$

$N_t^k$ is the piecewise-linear interpolation function of time as shown in Fig. 2.

The boundary variables $\phi$ and $q$ are represented by evenly spaced sub-intervals with a linear approximation for the time dependency (Fig. 3). This results in constant functions being used to approximate $\frac{\partial \phi}{\partial t}$. As a result, the time derivative of the time interpolation function $\frac{\partial}{\partial t}N_t^k(t)$ is used to approximate the time derivative of $\phi$.

Inserting these approximations into BIE (4), we can obtain the following discretized BIE at time step $t_m$:

$$\frac{1}{2}\phi_i^m = \sum_{j=1}^{N}\left[\sum_{k=1}^{m}\int_{E_j} N_t^k(t_m')\frac{1}{4\pi r}dS_j\cdot q_j^k\right] + \sum_{j=1}^{N}\left[\sum_{k=1}^{m}\int_{E_j} N_t^k(t_m')\frac{1}{4\pi r^2}\frac{\partial r}{\partial n}dS_j\cdot \phi_j^k\right] - \sum_{j=1}^{N}\left[\sum_{k=1}^{m}\int_{E_j}\frac{\partial}{\partial t}N_t^k(t_m')\frac{1}{4\pi r c}\frac{\partial r}{\partial n}dS_j\cdot\phi_j^k\right]. \tag{7}$$

where $t_m'$ is the retarded time $t_m' = t_m - \frac{r}{c}$.

To solve Eq. (7) at time step $t_m$, we rearrange the equation in a standard form for a linear system of equations:

$$\mathbf{Ax} = \mathbf{b}, \tag{8}$$

in which $\mathbf{A}$ is the coefficient matrix with a dimension of $N$ by $N$, $\mathbf{x}$ is the vector with unknowns $\phi_i^m$ or $q_i^m$ on each element, and $\mathbf{b}$ is the known right-hand-side vector computed from the known boundary values.

Finally, we note a property that can be used to effectively save the memory usage. The time loop in Eq. (7) needs to start from 1, but due to the truncated attributes of the fundamental solution, many of these

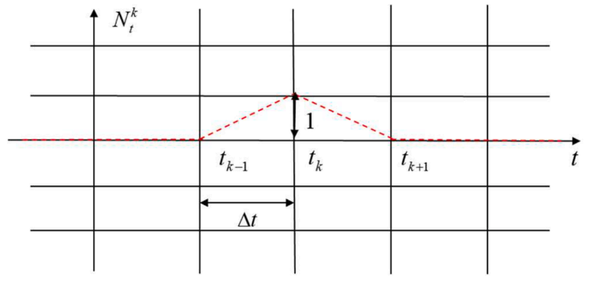


Fig. 2. Piecewise-linear interpolation function of time for $t = t_k$.

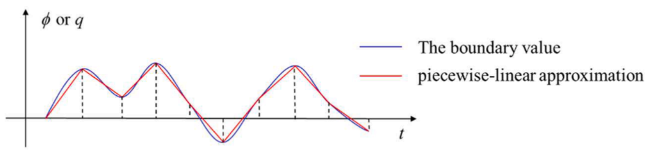


Fig. 3. Functions $\phi$ and $q$ are approximated with the piecewise-linear bases in time.

terms have zero values, where the non-zero values depend on the maximum distance $r_{\max}$ between element $E_i$ and $E_j$, specified as:

$$N_{t\min} = \frac{r_{\max}}{c\Delta t} + 1, \tag{9}$$

where $N_{t\min}$ is the minimum number of time steps that need to be computed for the pair of elements $E_i$ and $E_j$. Thus, the number of time loops $m$ in Eq. (7) can be reduced to $N_{t\min}$, when $m > N_{t\min}$.

### 2.2. A kernel-function library based TDBEM

One of the problems for the TDBEM is the consumption of large amount of memory. In order to avoid calculating the kernel functions for the same time difference at different time steps, the kernel functions for each time difference are usually stored, which brings the amount of storage to $O(N^2 N_{t\min})$. However, it is not possible to store all the data for larger scale models. Only a portion of the coefficients can be stored, and the remaining coefficients will need be recomputed, which increases the computation time. Therefore, it is necessary to develop a new method for storing coefficients to save memory and not to increase the computational effort at the same time.

In Eq. (8), computing the matrix $\mathbf{A}$ is only related to the current time step, but computing the right-hand-side vector $\mathbf{b}$ requires $N_{t\min}$ time loops when the $m - N_{t\min} > 0$, i.e., all boundary values need to be considered. Computing the vector $\mathbf{b}$ requires a large amount of memory usage when $N_{t\min}$ is large. Due to the truncation of the kernel function, the coefficient matrix is usually sparse. It is not difficult to solve the system of linear equations. Therefore, often the computation of the vector $\mathbf{b}$ containing the time loops occupy a larger proportion of the overall computation.

The pseudo codes of the main program and the subroutine for the vector $\mathbf{b}$ generation when the $m - N_{t\min} > 0$ for the TDBEM is given below.

```
Code 1 Main
Input the element and node information and boundary conditions
1: for time step $m$ =1,2, …, $N_t$ do
2:   Compute the vector b for current time step $m$ according to Code 2.
3:   Compute the coefficient matrix A for current time step $m$.
4:   Solve the $Ax = b$ to obtain the unknown vector x.
5:   Compute $\phi$ at any field point in the space if needed.
6: end for
```

```
Code 2 Compute the vector b
1: for element $j = 1, 2, …, N$ do
2:   for node $i$ =1, 2, …, $N$ do
3:     if $i \neq j$ then
4:       for $l = m - N_{t\min}$ to $m$ do
5:         Numerical Gaussian quadrature of the kernel function computed
           according to the Eq. (7)
6:       end for
7:     else
8:       for $l = m - N_{t\min}$ to $m$ do
9:         Analytical integration of the kernel function computed according to
           the Eq. (7)
10:      end for
11:    end if
12:  end for
13: end for
```

Through the Code 2 it can be learned that in the process of calculating the vector **b**, a loop of $N_{t\min}$ times is needed for computing the kernel with the effect of retarded time. If coefficients at all time steps are stored, it will need a huge amount of storage space. If coefficients at only a part of time steps are stored, it will increase the computation time.

However, computing kernel function is only related to the $r$, $\partial r/\partial n$ (Fig. 4) and the time step size $\Delta t$. Therefore, we can generate a library in which values of those terms are stored. The library can be directly used in calculating the kernel functions at all time steps.

In this paper, the minimum value of $r_{\min}$ is set to be $c\Delta t$, the maximum value of $r$ is related to the overall size of the model ($r_{\max}$) and $\partial r/\partial n = \cos\beta$ is between -1 and 1. The kernel function library can be used for the far-away elements, whose distance from the element C is greater than the minimum value, as shown in Fig. 5. The elements, for which the distances from element C are less than the minimum value, are called the near elements, and the kernel functions on the near elements are computed using the direct method. The influence of the selection of near elements on the computing results is discussed in the numerical example in Section 3.1.

We define the $N_r$ and $N_{\partial r/\partial n}$ as the numbers of $r$ and $\partial r/\partial n$ that need to be stored in the library, respectively. The pseudo code for the library generation is given below.

**Code 3 Generate the library**

```
1: Input the $r_{\min}$, $r_{\max}$
2:   for $i$ =1, 2, ..., $N_r$ do
3:     for $j$ =1, 2, ..., $N_{\partial r/\partial n}$ do
4:       for $l = m - N_{t\min}$ to $m$ do
5:         Compute the kernel function $N_t^l(t'_m)\frac{1}{4\pi r}$, $N_t^l(t'_m)\frac{1}{4\pi r^2}\frac{\partial r}{\partial n}$, $\frac{\partial}{\partial t}N_t^l(t'_m)\frac{1}{4\pi rc}\frac{\partial r}{\partial n}$
   in Eq. (7), and store the nonzero kernel function and the corresponding $l$.
6:     end for
7:   end for
8: end for
```

According to Code 3, we compute and store the kernels in the library for the different $r$ and $\partial r/\partial n$, which are selected at equal intervals $\Delta r$ and $\Delta(\partial r/\partial n)$ to facilitate program calls, the influence of the choice of $\Delta r$ and $\Delta(\partial r/\partial n)$ on the computing results is also discussed in the numerical example in Section 3.1.

From the Eq. (6), we can see that for every $r$ and $\partial r/\partial n$, there will be two sets of non-zero kernel functions and $l$. So the memory usage of the kernel function library is $O(2 \times 4N_rN_{\partial r/\partial n})$.

In the library, when the $r = c\Delta t + i\Delta r$, $\partial r/\partial n = -1 + j\Delta(\partial r/\partial n)$, we store the nonzero kernel functions as follows (Table 1):

For Code 2, when the $r$ and $\partial r/\partial n$ are given, the results in the library can be called directly, reducing the scale of memory consumption from $O(N^2N_{t\min})$ to $O(N^2 + 8N_rN_{\partial r/\partial n})$, in which $N_r$ is generally equal to $N$, and $N_{\partial r/\partial n}$ is generally much smaller than $N$, when $N$ is larger, leading to $O(N^2 + 8N_rN_{\partial r/\partial n}) \sim O(N^2)$.

The following is the pseudo code for calculating the vector **b** using the kernel function library when the $m - N_{t\min} > 0$.

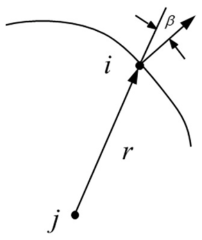


**Fig. 4.** The distance $r$ and normal derivative $\partial r/\partial n = \cos\beta$ between elements $i$ and $j$.

**Code 4 Compute the vector b using the kernel function library**

```
1: for element $j$ = 1, 2, ..., $N$ do
2:   for node $i$ =1, 2, ..., $N$ do
3:     if $i \neq j$ then
4:       if $r < c\Delta t$ then
5:         for $l = m - N_{t\min}$ to $m$ do
6:           Numerical Gaussian quadrature of the kernel function computed
   according to the Eq. (7)
7:       end for
8:       else
9:           Numerical Gaussian quadrature of the kernel function called directly
   or interpolated from the library
10:      end if
11:    else
12:      for $l = m - N_{t\min}$ to $m$ do
13:      Analytical integration of the kernel function computed according to
   the Eq. (7)
14:    end for
15:    end if
16:  end for
17: end for
```

Because the $r$ and $\partial r/\partial n$ in the library are selected by equal intervals, there will be slight differences in values between those from the library and those computed directly. Therefore, we need to perform linear interpolation on the value of kernels in the library according to the relationship between the value computed directly and the value from the library. Then the results from the interpolation can be used for subsequent computing.

## 3. Verification

In order to verify the accuracy and show advantages of the KFL-BEM, two examples with analytical solutions are presented in this section first. That is, a problem of a pulsating sphere and a problem of sound propagation in a channel. Through the comparison of the computed results of the sound pressure at the boundary points and/or field points with those from the analytical solutions, the accuracy and efficiency of the KFL-BEM can be verified and studied. Results for all examples are computed on a PC workstation with an Intel(R) Xeon(R) CPU E5-2650 v4 at 2.2GHz and RAM size of 64 GB. The program is written in Fortran language and Intel Math Kernel Library is used to solve the linear system of equations.

### *3.1. Radiation problem of a pulsating sphere*

A pulsating sphere at a single frequency is considered as the first example testing the KFL-BEM, for which the analytical solution is readily available for checking the BEM solution. The radius of the pulsing sphere is 0.5 m and its center is located at the point (0, 0, 0) m. The surface of the sphere is divided into 10800 and 19200 triangular constant elements, respectively (Fig. 6). The time interval is 0.0001s, and 1000 time steps are computed. The normal vibration velocity of the sphere surface

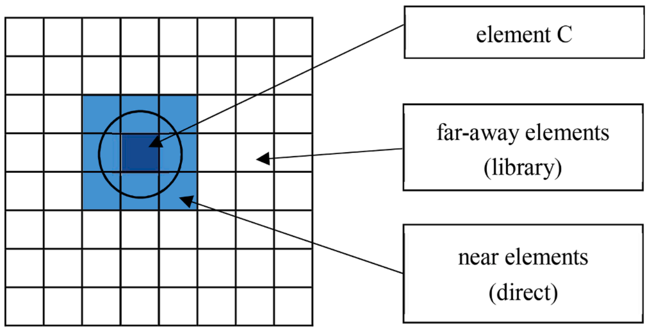


**Fig. 5.** Group of elements for element C.

is given as $q = v_0\sin(\omega t)$, where $v_0 = 1\text{m}/\text{s}$, and ω is the angular frequency. The field point where the sound pressure is evaluated is located at (0, 0, 1) m.

The computed results of sound pressure at the boundary points and the field point of the pulsating sphere are compared with the analytical values of the pulsating sphere, where different numbers of elements are used. The expression for the analytical value is given by:

$$\phi(\mathbf{x}, t) = \frac{\rho c v_0 k a^2}{r\sqrt{1+(ka)^2}}\sin[\omega t - k(r-a) + \theta], \tag{10}$$

where $\rho = 1.29\text{kg}\cdot\text{m}^{-3}$ is the mass density, $c = 343\text{m}\cdot\text{s}^{-1}$ is the speed of sound, $k$ is the wave number, $a$ is the radius of the pulsating sphere, $r$ is the distance from the center of the sphere to the field point, $\omega = 2\pi f$ is the angular frequency, $f$ is the frequency, and $\theta = \arctan(1/ka)$.

The sound pressure on the boundary and at the field point at two frequencies, 100 Hz and 500 Hz, are computed. To obtain the sound pressure values, it is necessary to use TDBEM to compute the acoustic velocity potential $\phi(\mathbf{x}, t)$ and then use the following formula to compute the sound pressure:

$$p(\mathbf{x}, t) = -\rho\frac{\partial}{\partial t}\phi(\mathbf{x}, t). \tag{11}$$

We use the pulsating sphere divided into 10,800 elements and at the frequency of 100 Hz as an example to compare the effects of near-field and far-field selections and the selection of $N_r$ and $N_{\partial r \partial n}$ in the library on the results. Since the value of $\partial r/\partial n$ ranges from $-1$ to 1 and does not vary with the degrees of freedom of the model, the number of $N_{\partial r \partial n}$ in all examples in this paper is set to 1000 and the size of $\Delta(\partial r/\partial n)$ is 0.002. The influence of $N_r$ and the near-elements selection $r_{\min}$ on the results is mainly discussed below.

We set the library, in which $r_{\min}$ is 0.01m, $r_{\max}$ is 1m and the value of $N_r$ is 100, 500, 1000, and 10000, respectively. The computed results of CBEM and KFL-BEM with different numbers of $N_r$ are compared with the analytical solutions, and the influence of $N_r$ on computational accuracy and stability is discussed.

The relative errors of the CBEM and the KFL-BEM at the boundary points are studied. Starting from 0.01s, relative errors are computed at selected time intervals every 20 steps, as shown in Fig. 7.

We can see from the Fig. 7 that when the $N_r$ is equal to 100, the computing results diverge due to the large error, part of the results after divergence are not shown in the Fig. 7; when the $N_r$ is equal to 1000, the error is basically less than 3%; when the $N_r$ is equal to 10,000, the number of $N_r$ in the library is basically the same as the degree of freedom of the model, and the error of the KFL-BEM is consistent with the CBEM on the whole model.

The influence of the selection of near elements on the result is discussed below. The value of $r_{\min}$ is selected as 0.001m, 0.01m, 0.03m to determine the near element. For the case when 0.03 m is approximately equal to $c\Delta t$, the value of $N_r$ is equal to 1,000, the accuracy of the computed results is compared in Fig. 8.

As can be seen from Fig. 8, with the increase of $r_{\min}$, the computational accuracy gradually increases. However, compared with the effect of $N_r$, $r_{\min}$ has a smaller effect on the accuracy.

It is worth noting that for the first few initial time steps, due to the presence of the delay time $t_r = t - r/c$, there exists a significant deviation between the results of the BEM and the steady-state analytical solution. Because the full acoustic wave has not propagated to all elements and the field point in these initial steps, there is always a small segment that deviates slightly from the analytic solution at the beginning of time scale in Figs. 7 and 8.

From the above discussion, the computational error gradually decreases with the increase of $N_r$ and $r_{\min}$. When $N_r$ and the number of elements are similar, and the near field is determined by $r_{\min} = c\Delta t$, the error of KFL-BEM is basically the same as that of CBEM.

Setting $N_r$ as 1000 and $r_{\min}$ as 0.01 m in the library, we compute the pulsating sphere model when the number of elements is 10800 and 19200, respectively. Then we compare the computing time and memory usage of CBEM and KFL-BEM as shown in Table 2.

For this case, $N_{t\min}$ is equal to 30, and the estimated memory usage of CBEM is $N_{t\min}$ times the size of the single step memory usage. In Table. 2, it is observed that the memory usage of the KFL-BEM consumes slightly more memory than the single step memory usage of the CBEM, due to the need for additional kernel function library storage. However, the estimated memory usage of the CBEM is proportional to $N_{t\min}$. Because the estimated memory usage is too large to store, the CBEM only stores coefficients at one time step and recompute other coefficients.

In the case of the same memory usage for the CBEM and KFL-BEM, the CBEM need to recompute other coefficients, resulting in significantly longer computing time compared to the KFL-BEM. When the numbers of elements are 10,800 and 19,200, the wall clock time for all time steps with the CBEM is 2.65 and 2.57 times of those with the KFL-BEM, respectively. For the CBEM, the memory usage increases linearly with each additional time step of the coefficients stored, while the computing time decreases accordingly. On the other hand, the KFL-BEM only requires the storage of the kernel function library, and the memory usage is only dependent on the model and kernel function library sizes rather than the number of time steps.

Finally, the time retarded phenomenon of the field points at different distances from the center of the sphere is discussed. Field points were set at (1, 0, 0), (2, 0, 0), (3, 0, 0), and (4, 0, 0) m respectively, using the same library as the example in Table 2 to compute the sound pressure with time. Fig. 9

The sound pressure results of field points at different distances from the sphere center all satisfy the time retarded equation $t_r = t - r/c$.

## 3.2. Problem of sound propagation in A channel

To further verify the accuracy of the program, a plane wave (or sine wave) propagating in a channel is studied. The length of the channel is 2 m, with a square cross-sectional area measured by 0.5 m X 0.5 m. The $z$ coordinate of the left end of the channel is 0 (Fig. 10). The surfaces of the channel are divided into 14400 constant triangular elements, and the computation is performed for 1000 time steps with a time interval $\Delta t$ of 0.0001 s. The sound velocity potential at the boundary point (-0.0083, 0.0083, 0) m is computed and compared with the analytical solution. Fig. 11

In this case, sound propagates along the $+z$-axis. The boundary condition on the $z = 0$ m end is specified as a known function of $q = H(t)$ m/s, where $H(t)$ is the Heaviside function. The boundary condition on the $z = 2$ m is specified as a known $\phi = 0$. The side surfaces have $q = 0$ boundary conditions. Hence, it is essentially a 1D problem for which the analytical solution exists [19].

Comparing the KFL-BEM and CBEM solutions with the analytical solution, it is observed that at different time steps, all the computed results of the KFL-BEM exhibit a good agreement with those of the

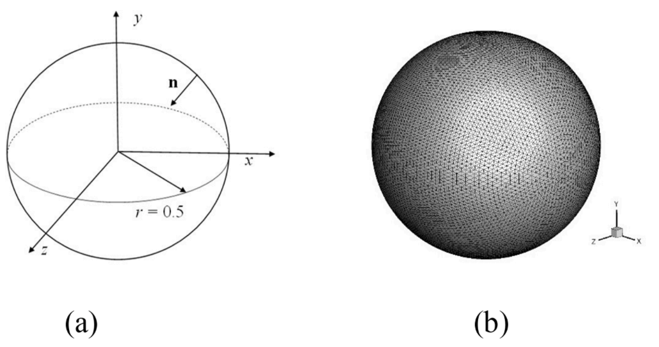
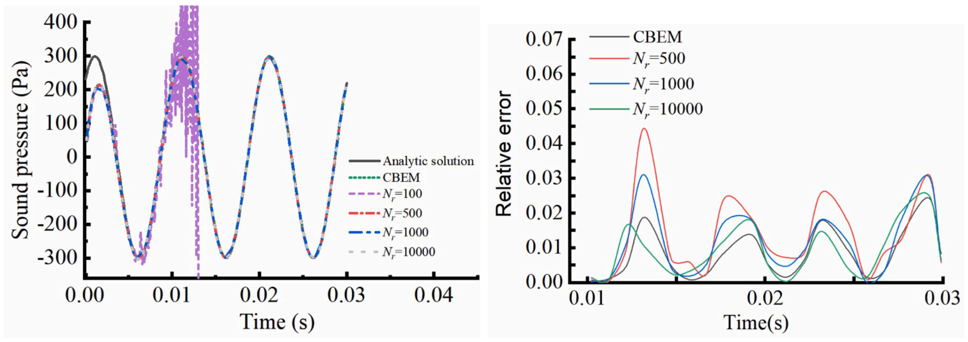
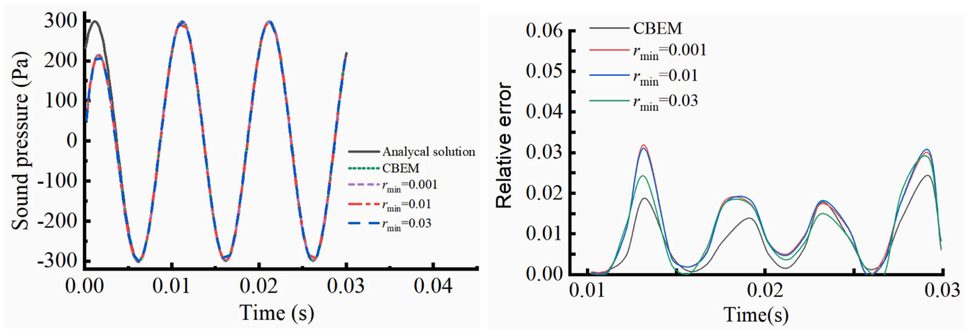


**Fig. 6.** Diagram of the pulsating sphere (a) and the mesh using boundary elements for $N = 10800$ (b).



(a) (b)

**Fig. 7.** The computed results on boundary point (0.5, 0, 0) m using the CBEM and the KFL-BEM with different numbers of $N_r$ (a) and the errors (b).



(a) (b)

**Fig. 8.** The computed results on boundary point (0.5, 0, 0) m using the CBEM and the KFL-BEM with the different values of $r_{\min}$ (a) and the errors (b).

**Table 1**
Kernel functions stored in library.

| $l_1$ | $l_2$ |
|---|---|
| $N_t^{l_1}(t'_m)\frac{1}{4\pi(c\Delta t+i\Delta r)}$ | $N_t^{l_2}(t'_m)\frac{1}{4\pi(c\Delta t+i\Delta r)}$ |
| $N_t^{l_1}(t'_m)\frac{1}{4\pi(c\Delta t+i\Delta r)^2}(-1+j\Delta(\partial r/\partial n))$ | $N_t^{l_2}(t'_m)\frac{1}{4\pi(c\Delta t+i\Delta r)^2}(-1+j\Delta(\partial r/\partial n))$ |
| $\frac{\partial}{\partial t}N_t^{l_1}(t'_m)\frac{1}{4\pi(c\Delta t+i\Delta r)c}(-1+j\Delta(\partial r/\partial n))$ | $\frac{\partial}{\partial t}N_t^{l_2}(t'_m)\frac{1}{4\pi(c\Delta t+i\Delta r)c}(-1+j\Delta(\partial r/\partial n))$ |

**Table 2**
Comparison of the computing time and memory usage by CBEM and KFL-BEM.

| | No. of elements | Wall clock time for all time steps (s) | Single step memory usage (Mb) | Estimated memory usage (Mb) |
|---|---|---|---|---|
| CBEM | 10,800 | 206,460 | 1,193 | 35,790 |
| KFL-BEM | 10,800 | 77,940 | 1,277 | 1,277 |
| CBEM | 19,200 | 732,600 | 3,336 | 100,080 |
| KFL-BEM | 19,200 | 285,060 | 3,427 | 3,427 |

analytical solution.

We set $N_r$ as 11000 and $r_{\min}$ as 0.03 in the library, similar to the values in computing the pulsating sphere model. Because the estimated memory usage is too large to store, the CBEM only stores the coefficients for the current time step, while the coefficients for previous time steps needed are recomputed. The number of boundary elements for the channel is 14400. For this example, $N_{t\min}$ is 69, by comparing the memory usage and computation time between the CBEM and the KFL-BEM, it is found that the CBEM consumes 1,973.7 Mb of memory, while the KFL-BEM consumes 2,981.4 Mb of memory. The estimated memory usage of CBEM is 136,185.3 Mb. In this case, a total of 1000 time steps are computed. The wall clock time for all time steps in the CBEM is approximately 640,921 s, while for the KFL-BEM the wall clock time is approximately 224,316 s. The wall clock time using the CBEM is 2.86 times longer than the wall clock time using the KFL-BEM.

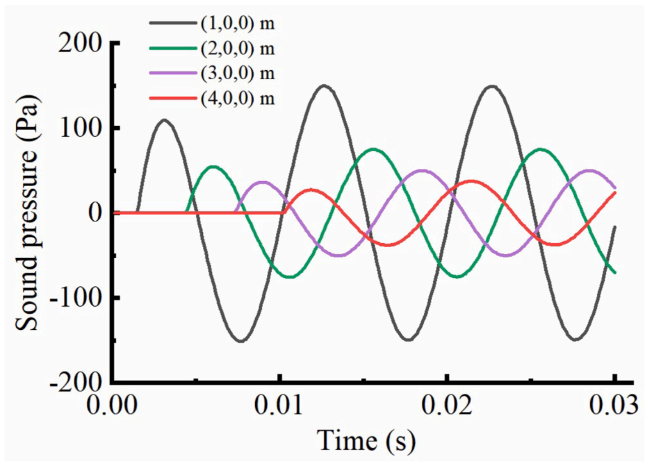


**Fig. 9.** The computed results on field points using the KFL-BEM

By analyzing the radiation problem of a pulsating sphere and the

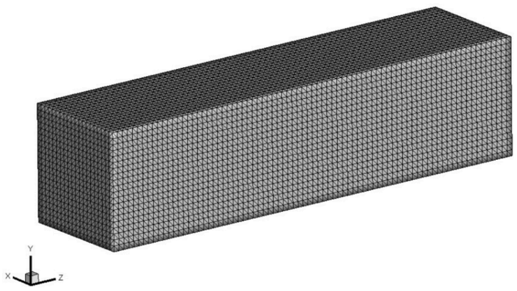


**Fig. 10.** Boundary elements used for the channel model with $N = 14400$.

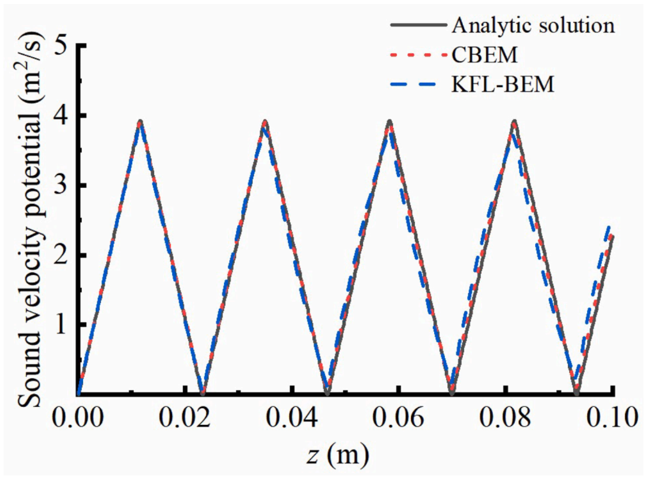


**Fig. 11.** The sound velocity potential at the boundary point (-0.0083, 0.0083, 0) m.

sound propagation problem in a channel, we can conclude that when the coefficients at only a single time step are stored, the KFL-BEM will use a similar size of memory as the CBEM. When considering equal levels of memory usage, the KFL-BEM offers significant advantages in computational efficiency while maintaining an acceptable level of accuracy. As mentioned above, since the size of the memory used by the KFL-BEM can be pre-determined, that is, it is only related to the size of the model and size of the kernel function library, and independent of the number of time steps.

## 4. A tuning fork example

To show the potential applications of the developed time-domain KFL-BEM, we model the vibro-acoustic responses of a tuning fork under various striking excitations. The results are used to generate a '.wav' file based on the sound pressure values as a function of time at a specified field point. This allows us to hear directly the sound produced by the tuning fork through numerical simulations.

A tuning fork is a metallic instrument with two arms that produces a consistent, fixed frequency sound. It serves as an effective tool for instrument tuning and pitch determination. In 2000, Russell [30] investigated the acoustic radiation properties of tuning forks in the frequency domain and compared them to linear quadrupole or dipole sources with respect to various vibration modes. This paper presents a time-domain analysis of the sound radiation generated by the vibration of a tuning fork under different strikes. The commercial FEM software Abaqus is utilized to compute the vibration responses of the fork, and the vibration velocities of the surface nodes are extracted to form the input for the KFL-BEM sound field computation. The contour plots of the sound velocity potential of the field at different time steps are provided for comparison with research findings in Ref. [30].

To investigate the sound radiation characteristics of a tuning fork under different striking loads, four different striking loads are applied (Fig. 12). The commercial tuning fork has an arm length of 95 mm and a distance of 13.3 mm between the arms, and rectangular cross-section arms with dimensions of 5.7 mm X 10 mm. This tuning fork has the standard frequency of 440 Hz. The surface of the tuning fork is divided into 10,772 triangular constant elements (Fig. 13). An annulus field surface with an outer diameter of 1 m and an inner diameter of 0.04 m is used, the center of the annular surface is located on the main axis of the tuning fork. The field surface is further divided into 5,200 quadrilateral elements. The time interval is 0.0005 seconds, with a total of 2000 time steps performed, resulting in a total duration of 1 s.

Figs. 14 and 15 respectively show the modal shape of the tuning fork and the velocity time response of the end of the tuning fork arm under the symmetry striking loading computed by Abaqus.

The velocity computed by the FEM is projected onto the element normal direction as the boundary condition with known $q$, and the sound pressure of field points is computed by the KFL-BEM. Fig. 16 shows contour plots of the acoustic velocity potential at different time and under different striking loads showed in Fig. 12. The two arms of the tuning fork are represented by two small rectangles.

Through the contour plots of the sound field around the tuning fork, we observe that the sound radiation direction of the tuning fork varies with different striking loads, which aligns with the directional pattern obtained by Russell [30].

The time-varying pressure curves at the filed point (-0.489, 0.06, 0.003) m of the KFL-BEM and the CBEM for the striking load in Fig. 12 (a) are plotted in Fig. 17.

The computation time for all time steps in the CBEM is 217,260 s, while that for the KFL-BEM it is 146,280 s. Performing an inverse Fourier transform on the data from Fig. 17 yields a vibration frequency of 447 Hz, with a relative error of 1.59% compared to the vibration frequency of the tuning fork (at 440 Hz). The numerical errors in the acoustic-structure analysis are acceptable, indicating the effectiveness of using this method for time-domain acoustic-structure analysis in different scenarios.

By increasing the number of time steps from 2,000 to 8,000, we obtain a set of results depicting the variation of sound pressure over time for the case of the folk under the symmetry strike and for a duration of 4 seconds. Using these results, we can generate a '.wav' file that allows us to directly listen to the sound produced after striking the tuning fork (this sound file is provided as a supplement file with this paper).

## 5. Conclusions

To improve the computational efficiency regarding the memory usage and computing time of the TDBEM for modeling 3D acoustic problems, we propose a kernel-function library time-domain BEM (KFL-BEM) in this paper. This method can effectively remedy the large memory consumption problem in the TDBEM, which can avoid the redundant computations of the same time differences at many different time steps. As a result, the memory consumption of time-domain acoustic BEM is reduced from $O(N^2 N_{t\min})$ to $O(N^2)$. Numerical verifications of the KFL-BEM are provided, showing that the memory usage of the KFL-BEM can be pre-determined, independent of $N_{t\min}$, and is similar to the CBEM when it only stores coefficients for a single time step. However, the KFL-BEM exhibits higher computational efficiency compared to the CBEM in this context, as no coefficients need to be computed repeatedly. Computations of vibro-acoustic problems under transient loads often require smaller time intervals and more time steps to compute the responses. Consequently, the benefit in memory-saving of the KFL-BEM becomes even more relevant. As an example, a vibro-acoustic analysis using a tuning fork under different striking loads is conducted. Based on the computed data, one can directly hear the sound emitted by the vibrating fork using the time-domain BEM, which is intriguing and may find more interesting applications in different fields.

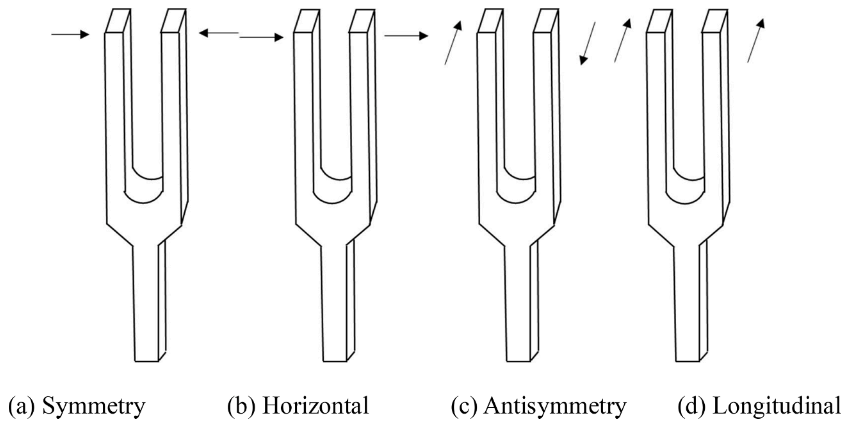


**Fig. 12.** Four different striking loads.

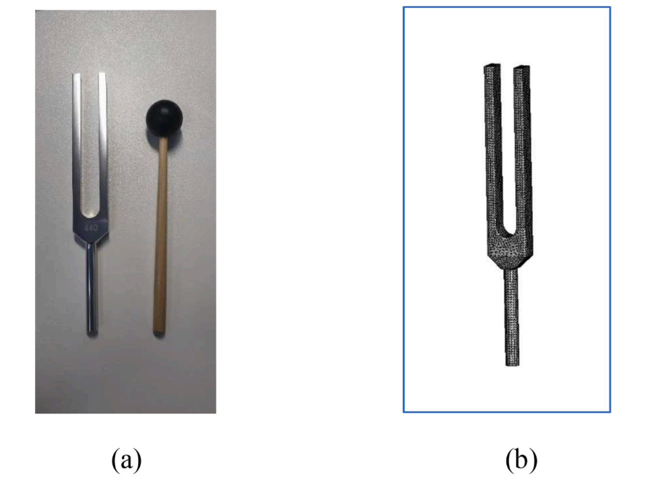


**Fig. 13.** The tuning fork used (a) and the boundary elements applied (b).

The advantages of the KFL-BEM in reducing memory usage can be extended to the BEM for solving other time-domain problems, such as transient elastodynamic and electromagnetic problems. Fast solution methods, such as the FMM, ACA and fast direct solvers, can also be implemented with the KFL-BEM to further improve the computational efficiencies in solving large-scale time-domain problems.

## CRediT authorship contribution statement

**Zhenyu Gao:** Writing – original draft, Software, Investigation, Formal analysis, Data curation. **Zonglin Li:** Investigation, Formal analysis, Data curation. **Yijun Liu:** Writing – review & editing, Supervision, Software, Resources, Project administration, Methodology, Funding acquisition, Conceptualization.

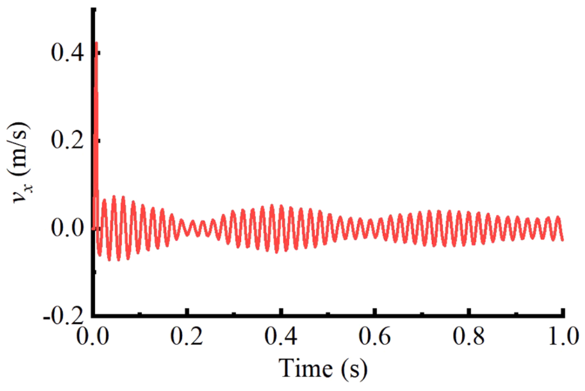


**Fig. 15.** the velocity time response of the end of the tuning fork arm.

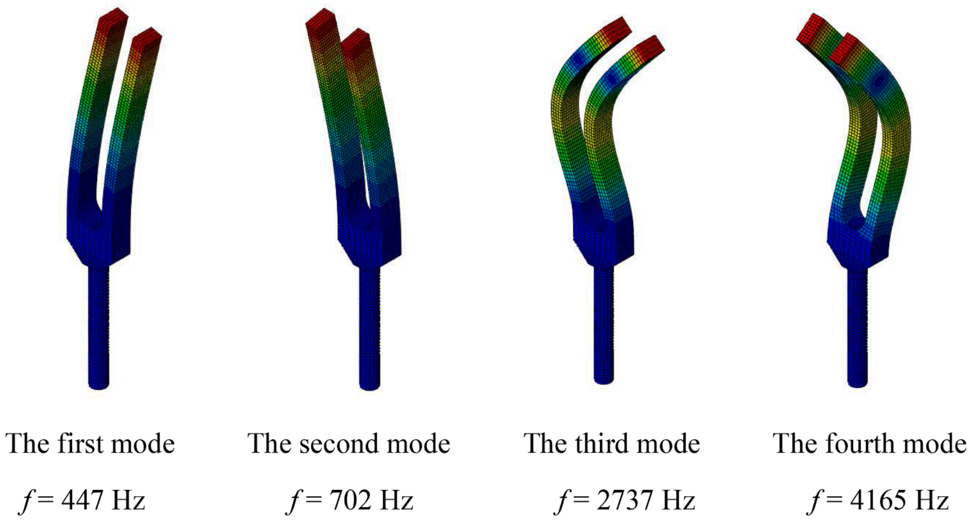


**Fig. 14.** Modal shape of tuning fork.

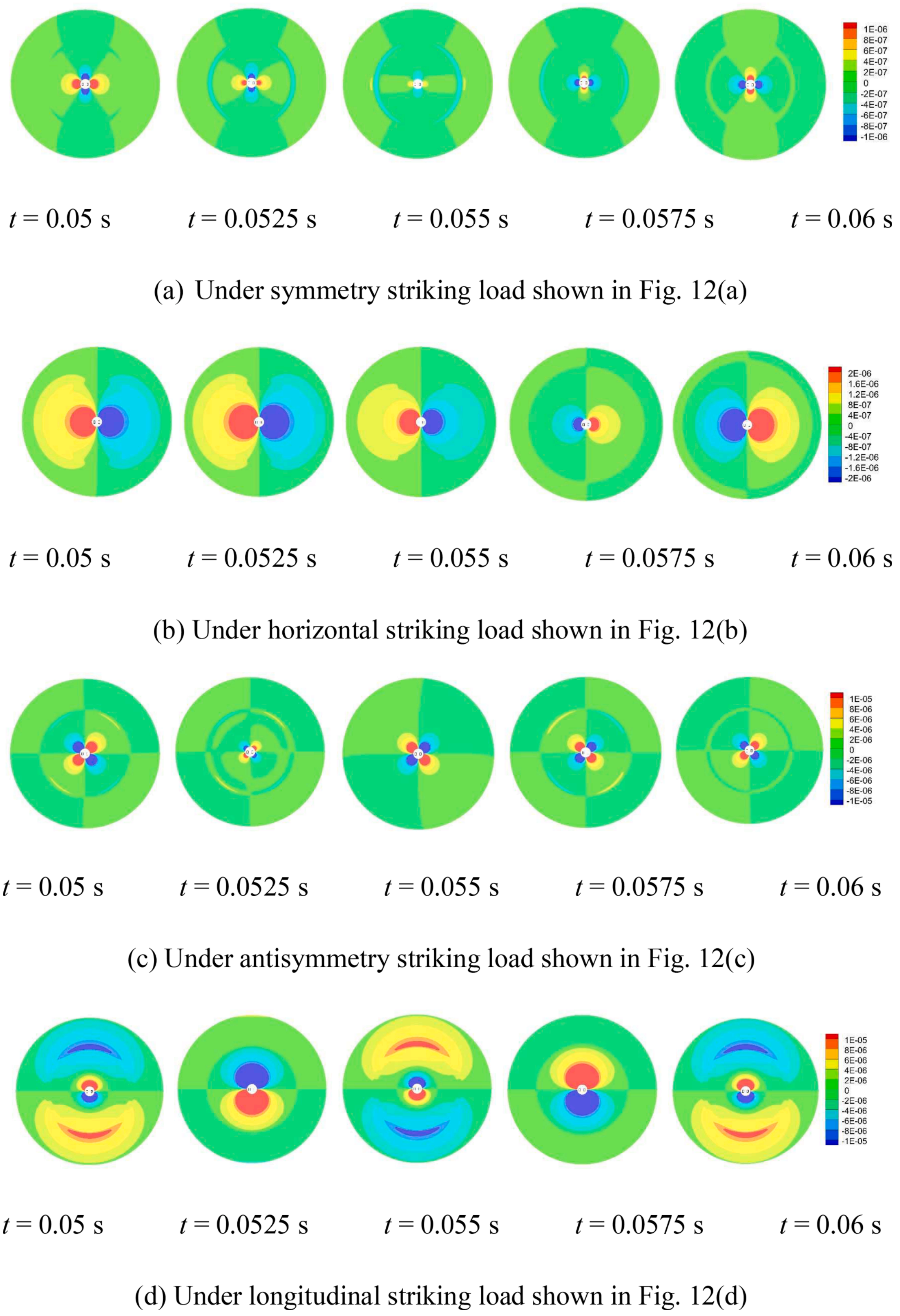


**Fig. 16.** Contour plots of the sound field at different time steps with the tuning fork under different striking loads.

## Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Yijun Liu reports financial support was provided by National Natural Science Foundation of China. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

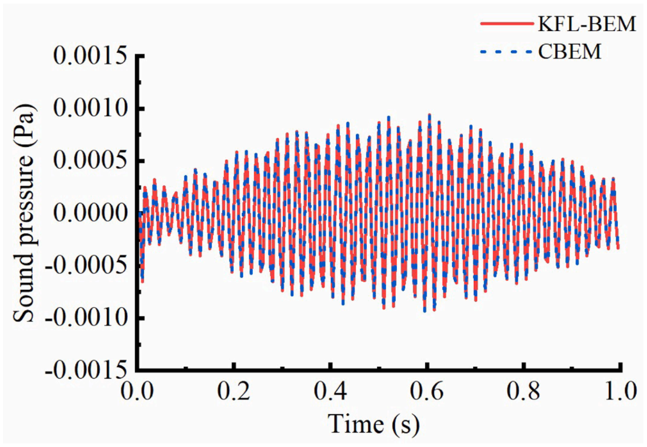


**Fig. 17.** Sound pressure-time diagram of sound radiation at the filed point after striking load on the tuning fork.

## Data availability

No data was used for the research described in the article.

## Acknowledgement

The authors would like to thank the following financial support for this work: the National Natural Science Foundation of China (Project Nos. 11972179 and 12372198), the Natural Science Foundation of Guangdong Province (No. 2020A1515010685), and the Department of Education of Guangdong Province (No. 2020ZDZX2008).

## Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.enganabound.2024.01.001.

## References

[1] Brebbia CA. The boundary element method for engineers. London: Pentech Press; 1978.
[2] Zienkiewicz OC, Taylor RL, Zhu JZ. The finite element method : its basis and fundamentals. Seventh ed. Oxford, UK: Butterworth-Heinemann; 2013.
[3] Liu YJ. On the BEM for acoustic wave problems. Eng Anal Bound Elem 2019;107:53–62. No.
[4] Burton AJ, Miller GF. The application of integral equation methods to the numerical solution of some exterior boundary-value problems. Proc R Soc. A, Math Phys Eng Sci 1971;323(1553):201.
[5] Meyer WL, Bell WA, Zinn BT, Stallybrass MP. Boundary integral solutions of three dimensional acoustic radiation problems. J Sound Vib 1978;59(2):245–62.
[6] Cunefare KA, Koopmann G, Brod K. A boundary element method for acoustic radiation valid for all wavenumbers. J Acoust Soc Am 1989;85(1):39–48.
[7] Liu YJ, Chen S. A new form of the hypersingular boundary integral equation for 3-D acoustics and its implementation with C0 boundary elements. Comput Methods Appl Mech Eng 1999;173(3):375–86.
[8] Greengard L, Rokhlin V. A fast algorithm for particle simulations. J Comput Phys 1997;135(2):280–92.
[9] Nishimura N, Yoshida K-i, Kobayashi S. A fast multipole boundary integral equation method for crack problems in 3D. Eng Anal Bound Elem 1999;23(1):97–105.
[10] Liu YJ, Nishimura N, Yao ZH. A fast multipole accelerated method of fundamental solutions for potential problems. Eng Anal Bound Elem 2005;29(11):1016–24.
[11] Shen L, Liu YJ. An adaptive fast multipole boundary element method for three-dimensional acoustic wave problems based on the Burton–Miller formulation. Comput Mech 2007;40(3):461–72.
[12] Huang S, Liu YJ. A new simple multidomain fast multipole boundary element method. Comput Mech 2016;58(3):533–48.
[13] Liu YJ. Fast multipole boundary element method : theory and applications in engineering. Cambridge: Cambridge University Press; 2009.
[14] Hackbusch W. A sparse matrix arithmetic based on h-matrices. Part I : Introduction to h-matrices. Computing 1999;62(2):89–108.
[15] Bebendorf M. Approximation of boundary element matrices. Numer Math (Heidelb) 2000;86(4):565–89.
[16] Martinsson PG, Rokhlin V. A fast direct solver for boundary integral equations in two dimensions. J Comput Phys 2005;205(1):1–23.
[17] Lai J, Ambikasaran S, Greengard LF. A fast direct solver for high frequency scattering from a large cavity in two dimensions. SIAM J Sci Comput 2014;36(6):B887–903.
[18] Li R, Liu YJ, Ye W. A fast direct boundary element method for 3D acoustic problems based on hierarchical matrices. Eng Anal Bound Elem 2023;147:171–80.
[19] Langer S, Schanz M. Time domain boundary element method. In: Marburg S, Nolte B, editors. Computational acoustics of noise propagation in fluids - finite and boundary element methods. Berlin, Heidelberg: Springer Berlin Heidelberg; 2008. p. 495–516.
[20] Ergin AA, Shanker B, Michielssen E. Fast evaluation of three-dimensional transient wave fields using diagonal translation operators. J Comput Phys 1998;146(1):157–80.
[21] Ergin AA, Shanker B, Michielssen E. Fast transient analysis of acoustic wave scattering from rigid bodies using a two-level plane wave time domain algorithm. J Acoust Soc Am 1999;106(5):2405–16.
[22] Ergin AA, Shanker B, Michielssen E. Fast analysis of transient acoustic wave scattering from rigid bodies using the multilevel plane wave time domain algorithm. J Acoust Soc Am 2000;107(3):1168–78.
[23] Takahashi T. An interpolation-based fast-multipole accelerated boundary integral equation method for the three-dimensional wave equation. J Comput Phys 2014;258:809–32.
[24] Takahashi T, Tanigawa M, Miyazawa N. An enhancement of the fast time-domain boundary element method for the three-dimensional wave equation. Comput Phys Commun 2022;271:108229.
[25] Takahashi T. A fast time-domain boundary element method for three-dimensional electromagnetic scattering problems. J Comput Phys 2023;482:112053.
[26] Aimi A, Desiderio L, Di Credico G. Partially pivoted ACA based acceleration of the energetic BEM for time-domain acoustic and elastic waves exterior problems. Comput Math Appl 2022;119:351–70.
[27] Thirard C, Parot J-M. On a way to save memory when solving time domain boundary integral equations for acoustic and vibroacoustic applications. J Comput Phys 2017;348:744–53.
[28] Yoshikawa NNH. An improved implementation of time domain elastodynamic BIEM in 3D for large scale problems and its application to ultrasonic NDE. Electron J Bound Elem 2003;1(2):201–17.
[29] Pan L, Adams DO, Rizzo FJ. Boundary element analysis for composite materials and a library of Green's functions. Comput Struct 1998;66(5):685–93.
[30] Russell DA. On the sound field radiated by a tuning fork. Am J Phys 2000;68(12):1139–45. No.