ISSN 1063-7710, Acoustical Physics, 2022, Vol. 68, No. 6, pp. 529–536. © Pleiades Publishing, Ltd., 2022.

NONLINEAR ACOUSTICS

The Use of Graphic Accelerators in Simulation of Nonlinear Ultrasonic Beams with Shock Fronts on the Basis of the Westerwelt Equation

E. O. Konnova[a], *, V. A. Khokhlova[a], **, and P. V. Yuldashev[a], ***

[a] *Laboratory of Medical and Industrial Ultrasound, Faculty of Physics, Moscow State University, Moscow, 119991 Russia*

**e-mail: helen.7aprel@gmail.com*

***e-mail: vera@acs366.phys.msu.ru*

****e-mail: petr@acs366.phys.msu.ru*

Received April 19, 2022; revised April 19, 2022; accepted July 27, 2022

Abstract—The problem of accelerating the algorithm for calculating nonlinear effects is considered, when modeling high-intensity ultrasonic beams based on the one-way Westervelt equation. When constructing a numerical solution for strongly distorted waves with shock fronts, it is necessary to take into account large number of harmonics (up to 1000) on spatial grids with a matrix size of the order of 10000 by 10000, which requires the processing of large amounts of data and a long calculation time. In this paper, the implementation of the nonlinearity operator is carried out in a time representation using a Godunov-type shock-catching scheme, which allows modeling nonlinear waves with shock fronts with a small (3) number of grid nodes on the shock front. The paper compares the efficiency of using this method when it is implemented on central processing units (CPUs) and graphics processing units (GPUs) in comparison with the spectral method implemented earlier for quasi-linear wave propagation. An analysis is made of the speed of algorithm execution on the CPU and GPU, depending on the size of the input data arrays.

Keywords: nonlinear Westervelt equation, Godunov method, graphics accelerators, non-invasive ultrasound surgery

DOI: 10.1134/S1063771022060161

INTRODUCTION

Numerical experiments are today an integral part of the development of methods of noninvasive surgery using nonlinear ultrasonic waves. One of the main objects for the study of which numerical methods are actively used are nonlinear focused ultrasonic beams of high intensity, with the help of which designated structures inside the human body, such as tumors, are destroyed [1]. To generate high-power ultrasound, transducers of various shapes and designs are used; when designing them for specific clinical applications, it is necessary to be able to quantitatively describe the structure of the acoustic fields they create (Fig. 1). Such a task can be effectively implemented using numerical simulation methods [2–4]. One of the most complete wave models for the theoretical description of powerful ultrasonic beams is the Westervelt equation, which allows one to quantify accurately nonlinear shock-wave fields created by focused high-power ultrasound transducers in homogeneous absorbing media [5, 6]. When setting a task, it is usually assumed that the emitter generates a monochromatic wave, the spectrum of which is enriched in higher harmonics during propagation due to the effect of acoustic nonlinearity. In the general case, to solve this equation, when time is an evolutionary variable, it is necessary to use supercomputers [7]. However, a number of simplifications can significantly reduce computational costs.

To optimize the numerical simulation of nonlinear focused ultrasonic beams, special algorithms are being developed. In this paper, the case of a directed three-dimensional beam, which is important for practice, is considered, which makes it possible to simplify the computational problem by passing to a retarded time coordinate system, the axis of which is oriented along the predominant direction of wave propagation in the beam. In this case, the evolutionary variable is the coordinate along the beam's z axis (Fig. 1). The numerical solution of this type of evolution equation is usually constructed using the method of splitting by physical factors [8], according to which, at each step of the grid along the z axis, each physical effect is calculated separately using the most appropriate numerical method. The nonlinear evolutionary Westervelt equation in this formulation was solved for emitters of various shapes and sizes [6, 9–10]. In this case, the typi-

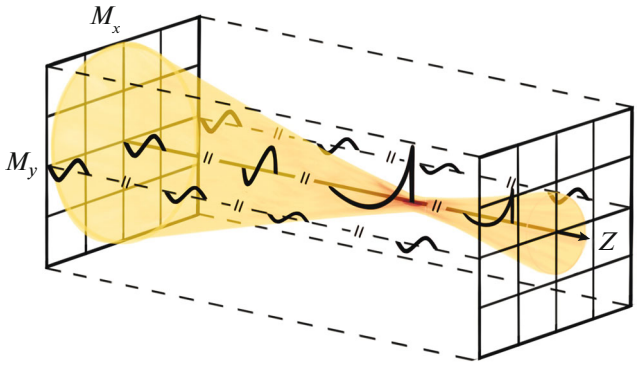

Fig. 1. Scheme of propagation of a focused ultrasonic beam created by an HIFU emitter.

cal size of matrices for storing the parameters of the pressure field is $M_x = 10000$ per $M_y = 10000$ (Fig. 1) for each of $N = 1000$ spectral components of the nonlinear waveform [9]. In this case, the required amount of RAM, which is quite large (several hundred gigabytes), and the corresponding complexity of the calculations still requires the use of supercomputers.

However, if we take into account the features of the spatial structure of the nonlinear fields of highly focused emitters, the problem under consideration can be solved on ordinary personal computers with multicore central processing units (CPUs). Since nonlinear effects mainly appear where the wave amplitude is large, i.e., near the focus, then a large number of harmonics must be taken into account precisely in this relatively small region of space [9]. This method makes it possible to reduce computational costs and requirements for the amount of computer RAM by an order of magnitude, however, even with parallel execution of calculations on several processor cores (usually from 2 to 16), which can significantly speed up calculations, field simulation for one set of input parameters can take up to several days [9]. Taking into account the fact that several tens of calculations are required to characterize the field of one emitter in the entire operating power range, such a simulation speed is not satisfactory.

A potential solution to the problem of computational speed is the recently rapidly developing technology of parallel programming on graphics processing units (GPUs). Unlike the CPUs, these processors have up to several thousand highly specialized cores capable of performing a wide range of mathematical operations [11]. Such a number of cores makes it possible to increase the speed of calculations due to a large number of simultaneously launched processes that process the same type of data segments in parallel. The resulting acceleration of calculations is not necessarily proportional to the number of processing threads, which is due to the lower power, performance, and computation speed of GPU cores compared to CPU cores, as well as the peculiarities of storing and transferring data between CPU RAM and graphics memory. Since the storage and recording of data during the execution of each of the steps of the algorithm along the evolutionary coordinate is performed in the RAM of the CPU, additional time is spent on the exchange of data between the central and graphic processors at each step of the algorithm. Due to the considered features of the GPU architecture, it is an important task to find a balance between the maximum sizes of processed at this step of the algorithm, the data that are placed in the RAM of the GPU in order to avoid their transfer in parts, and minimizing the exchange of data between the processors.

Earlier, for this problem, an algorithm was implemented on the GPU, in which kernel functions were written to calculate the diffraction operator by the angular spectrum method, find the exact solution for the absorption operator, and solve the nonlinearity operator in the spectral representation by the Runge-Kutta method of four orders [12]. However, when using the spectral method, the number of computational operations and, accordingly, the computation time are proportional to the square of the number of generated harmonics. Therefore, this method will be effective only with a weak manifestation of nonlinear effects, when a small number of higher harmonics is formed in the wave spectrum. The nonlinearity operator can also be calculated using various methods in time representation; however, in many of them, a large number of grid points along the time axis (50–100) are required to describe the discontinuity region, which increases the total number of grid points and, thus, the amount of data being processed [13, 14]. To solve this problem, it is more efficient to use shock-catching schemes that allow modeling the formation of a shock front using a small number of points on it (about three), for example, a conservative Godunov-type scheme [15].

The purpose of this work is to implement the Godunov-type shock-catching circuit algorithm on a graphical processing unit for greater acceleration of parallel computing compared to using multicore central processing units and to make it possible to model high-power ultrasonic beams with shock fronts on a conventional personal computer.

THEORETICAL MODEL

Nonlinear Westervelt Equation

The Westervelt equation in a retarded time coordinate system can be written in evolutionary form as

$$\frac{\partial^2 p}{\partial\tau\partial z} = \frac{c_0}{2}\left(\frac{\partial^2 p}{\partial x^2} + \frac{\partial^2 p}{\partial y^2} + \frac{\partial^2 p}{\partial z^2}\right) + \frac{\beta}{2\rho_0 c_0^3}\frac{\partial^2 p^2}{\partial\tau^2} + \frac{\delta}{2c_0^3}\frac{\partial^3 p}{\partial\tau^3}, \quad (1)$$

where $p(x, y, z, \tau)$ is the acoustic pressure, c_0 is the speed of sound in the medium, $\tau = t - z/c_0$ is the retarded time, β is the coefficient of nonlinearity, and δ is the thermoviscous absorption coefficient [5]. The differential operators on the right side of the equation, in order from left to right, describe the effects of diffraction, nonlinearity, and thermoviscous absorption.

In numerical solution of Eq. (1), at each step along the z axis, discretized pressure field $p(x, y, z, \tau)$ is represented in the computer memory in the form of a three-dimensional matrix, which contains a set of complex amplitudes N harmonics of the wave spectrum in the expansion of its waveform into a finite Fourier series at each spatial point of plane xy on a uniform grid with number of points M_x and step Δx along the x axis and M_y with step Δy along the y axis:

$$p(x, y, z, \tau) = \sum_{n=-N}^{N} p_n(x, y, z) e^{-i\omega_n \tau}. \quad (2)$$

Here, $\omega_n = \omega n$ is the circular harmonic frequencies with number n, ω is the circular frequency of the monochromatic source, and p_n is the complex amplitudes of the harmonics. Note that it is sufficient to store only one-half of the spectrum of positive frequencies in the computer memory, since the amplitude of the second half of the negative frequencies is complex conjugate to the first.

Using spectral representation (2) for the pressure field in the scheme of splitting by physical factors of solving Westervelt equation (1), it is possible to calculate the diffraction operator using the angular spectrum method and accurately calculate the absorption operator for each of the spectral components of the wave independently of each other [12]. Features of parallel computations of the nonlinearity operator in the spectral and temporal representations are considered below. We consider the implementation of only a nonlinear operator in the Westervelt equation; i.e., in fact, the equation of simple waves is solved. Since this equation is solved independently in each spatial coordinate, to simplify the representation of data in the computer memory, we will proceed to consider the row of the original array in terms of spatial coordinate x with M elements, which will not affect the study of the efficiency of the algorithm. Thus, the wave propagates along the z axis, nonlinear effects are calculated at each step by z independently M times for all grid nodes along coordinate x (Fig. 2), optimization of calculations is carried out by parallelizing data flows along this coordinate. In the future, when modeling a complete nonlinear diffraction problem, calculations will be carried out on a three-dimensional grid. For this purpose, data from a two-dimensional spatial grid in plane xy with full number of nodes $M = M_x M_y$ can be processed by splitting them into one-dimensional arrays of size described above M.

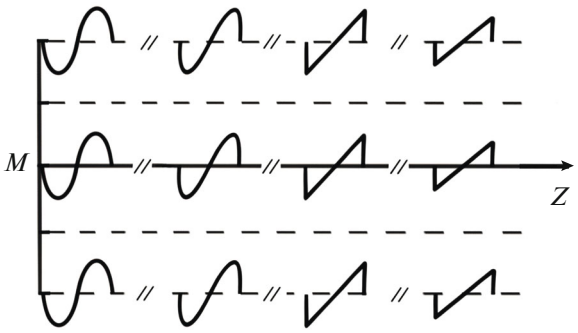

Fig. 2. Scheme for the implementation of parallel calculation of a nonlinear operator, where M is number of spatial points.

Nonlinearity operator

If the diffraction and absorption operators are not considered in Eq. (1), as a result, we obtain the equation of simple waves, which describes the nonlinear distortions of the plane waveform:

$$\frac{\partial p}{\partial z} = \frac{\beta}{2\rho_0 c_0^3} \frac{\partial p^2}{\partial \tau}. \quad (3)$$

As mentioned above, in a three-dimensional nonlinear problem, Eq. (3) is solved independently for each of points $M = M_x M_y$ spatial grid in plane xy (Fig. 2). In the case of weak manifestation of nonlinear effects, when the formation of shock fronts is not expected, it is convenient to use a spectral algorithm in which the system of coupled nonlinear equations is solved numerically for the amplitudes of harmonics (2) [16]. Usually, the Runge–Kutta method with four orders of accuracy is used as a numerical method.

Earlier, in the works of the authors of this article, such a spectral algorithm was implemented to perform calculations on graphics processors when solving the complete nonlinear diffraction problem (1) [12]. Parallel computing is organized by splitting the total data array into segments of size $L_x L_y N$, where L_x is the size of the buffer along the spatial coordinates x, L_y is that along spatial coordinate y, and N is the number of harmonics. This approach allows one to process the amount of data that does not necessarily fit entirely in the amount of GPU memory. Portions of data from temporary buffers are transferred from the CPU memory to the GPU memory, where they are processed by a parallel algorithm, in which, for each process on the graphics core, a set of N harmonics was selected out of total number of processes $L_x L_y$. Thus, for each data segment sent to the GPU, the algorithm uses data parallelism. When implementing an algorithm on a GPU, it is an important task to find a balance between the amount of data being processed and the capabilities of the processor's memory. When evaluating computational performance, one must also take into account

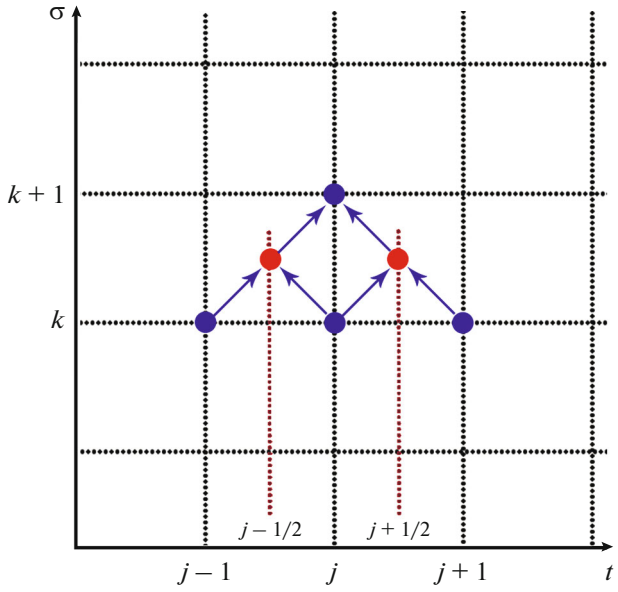

Fig. 3. Scheme for a conservative Godunov-type numerical scheme for the simple wave equation.

the time spent transferring data between CPU and GPU memory.

With a strong manifestation of nonlinear effects, a large number of harmonics are required to describe the process of waveform distortion, information about which must be stored at each step along beam axis z. Since the number of computational operations in the spectral approach grows in proportion to the square of the number of harmonics, the spectral algorithm quickly becomes inefficient. An alternative way to solve Eq. (3) is to use numerical schemes that work in time representation. In this case, it is most efficient to use shock-catching schemes that allow one to model highly distorted nonlinear waves using a small number of points on shock fronts [13, 15]. Note that, when storing data in spectral representation (2), in order to pass to waveforms, it is necessary to perform a fast Fourier transform, which can also be performed on the GPU. To construct a numerical solution, it is convenient to reduce Eq. (3) to a dimensionless form:

$$\frac{\partial V}{\partial \sigma} = \frac{1}{2}\frac{\partial V^2}{\partial \theta}, \tag{4}$$

where $V = p/p_0$ is acoustic pressure normalized to wave amplitude p_0, $\theta = \omega_0\tau$ and $\sigma = z/z_s$ are the dimensionless time and coordinate, respectively; and $z_s = c_0^3\rho_0/\beta p_0\omega_0$ is the shock formation distance for an initially harmonic wave with frequency ω_0 specified in dimensionless variables as

$$V(\sigma = 0, \theta) = \sin\theta. \tag{5}$$

Godunov's method

The numerical solution of Eq. (4) at each grid step along the evolutionary coordinate σ was constructed using a Godunov-type scheme (Fig. 3). The calculation algorithm is an explicit six-point conservative scheme of the second order of accuracy in time and the first order of accuracy in the propagation coordinate [13, 15]:

$$V_j^{k+1} = V_j^k - \frac{h_\sigma}{h_\theta}\left(H^k_{j+\frac{1}{2}}(\sigma) - H^k_{j-\frac{1}{2}}(\sigma)\right), \tag{6}$$

where flows $H^k_{j+\frac{1}{2}}$ through the centers of the cells of the numerical grid on the kth step by coordinate σ are set as follows:

$$H^k_{j+\frac{1}{2}}(\sigma) = -\frac{1}{4}\left(\left(V^+_{j+\frac{1}{2}}\right)^2 + \left(V^-_{j+\frac{1}{2}}\right)^2\right) - \frac{a^k_{j+\frac{1}{2}}(\sigma)}{2}\left[V^+_{j+\frac{1}{2}}(\sigma) - V^-_{j+\frac{1}{2}}(\sigma)\right]. \tag{7}$$

Here, h_σ is the scheme step along coordinate σ, h_θ is the grid step along dimensionless time axis θ, and local flow velocity in the grid cell is

$$a^k_{j+\frac{1}{2}}(\theta) = \max\left|V^-_{j+\frac{1}{2}}, V^+_{j+\frac{1}{2}}\right|. \tag{8}$$

To obtain the second order of accuracy, this algorithm uses a piecewise linear reconstruction of pressure values $V(\sigma_k, \theta_j)$ on the right $V^+_{j+\frac{1}{2}}(\sigma)$ and left $V^-_{j+\frac{1}{2}}(\sigma)$ of the node of the numerical grid (k, j):

$$V^+_{j+\frac{1}{2}} = V^k_{j+1}(\sigma) - \frac{h_\theta}{2}\left(\frac{\partial V}{\partial \theta}\right)^k_{j+1}, \quad V^-_{j+\frac{1}{2}} = V^k_j(\sigma) + \frac{h_\theta}{2}\left(\frac{\partial V}{\partial \theta}\right)^k_j. \tag{9}$$

For greater stability of the numerical algorithm, the time derivatives involved in (8) of the solution at step k by coordinate σ are chosen in such a way that their values are minimal in absolute value from the possible values of the derivatives—right, left, and central with a weighting factor of $1 \le b \le 2$:

$$\left(\frac{\partial V}{\partial \theta}\right)^k_j = \min\operatorname{mod}\left(\frac{b\left(V^k_j - V^k_{j-1}\right)}{h_\theta}, \frac{V^k_{j+1} - V^k_{j-1}}{2h_\theta}, \frac{b\left(V^k_{j+1} - V^k_j\right)}{h_\theta}\right). \tag{10}$$

Weight coefficient $b = 2$ corresponds to the most accurate solution with minimum grid absorption and $b = 1$ corresponds to a more stable numerical algo-

Table 1. Comparison of the speed of algorithms based on the Godunov method for a CPU and GPU and using the spectral method for a GPU, when performing calculations on a laptop, for number of harmonics $N = 1000$ for different numbers of M waveforms

Number of points in space M		100	500	1000	5000	10000
Time T, s	CPU (TD Godunov)	30.7	156	324	1562	3023
	GPU (TD Godunov)	3.5	12.2	22	55	108
	GPU (FD RK4)	2	11.6	36.8	141	282
T_{CPU_G}/T_{GPU_G}		8.3	12.8	14.8	28.4	28
$T_{GPU_{RK4}}/T_{GPU_G}$		0.57	0.95	1.67	2.56	2.6

rithm, which is achieved by increasing grid absorption. In this work, the value of b assumed to be 1. The presented scheme makes it possible to calculate the propagation of narrow shock fronts with high accuracy using only three grid nodes on the shock front [15]. Note that, when solving three-dimensional problems, artificial absorption is usually additionally introduced, which makes it possible to smear the width of the shock front to the required values (usually 7 or 8 points per front) in order to reduce large spatial gradients of the pressure field along transverse coordinates [6].

Calculations were carried out on two different computers: on a laptop with a CPU Intel Core i5-8250U with a GPU Nvidia GeForce MX150 video card (380 cores, 1.5 GHz) and on a personal computer (PC) with a CPU Intel Core i7 4790 with an Nvidia GTX1070 graphics card (1980 cores, 1.7 GHz). To implement calculations on GPUs, a program was written in C containing a CUDA core function that implemented algorithms (6)–(10). To determine the efficiency of this program in the C language, a similar single-threaded algorithm for the CPU was implemented, as well as an algorithm for the GPU based on the spectral method, with the calculation speed of which a comparison was made. When solving Eqs. (6)–(10) numerically, at each step in σ, the discretized pressure field was represented in the computer memory in the form of a two-dimensional matrix that contained the set of M waveforms specified in M spatial points. Such a representation of the data is necessary to implement data parallelization, in which each waveforms from among M is processed in parallel with other waveforms by different GPU cores. The calculations were carried out for a different number of spatial coordinates M (100–10000 points), as well as for a different number of points per wave period (100–3000 points) with time grid steps $\Delta t = 0.0063$ and $\Delta\sigma = 0.2378$ satisfying the stability of the Godunov-type scheme for the maximum considered number of points per wave period. To determine the correctness of the implemented algorithm in the time representation, we used an analytical solution of a simple wave equation with an infinitely narrow shock front (4) with initial waveform (5) presented in an implicit form [17]:

$$V_{th} = \sin(\theta + \sigma V_{th}). \quad (11)$$

The position and magnitude of the shock front were determined by the rule of equality of areas.

RESULTS

Figure 4 shows the wave profiles at different distances obtained numerically using the Godunov method implemented on a laptop GPU, as well as the theoretical solution described by Eq. (11). There is good agreement between the numerical and analytical solutions with an error of no more than 0.03% in the amplitude of the shock front. The discrepancy is observed only in the discontinuity region, since, in contrast to the analytical solution, the numerical scheme leads to the formation of a shock front of finite width with three points on the discontinuity.in. Thus, the width of the front when using 1000 points per period is about 0.2% of the wave period. These results confirm the correct implementation of the algorithm for the GPU. Identical results were obtained for algorithms implemented on a PC using one of the CPU cores and on a laptop with and without a video card.

Tables 1 and 2 compare the results of calculations on a laptop based on the run time of a single-threaded algorithm based on the time-domain method and implemented on the CPU with the corresponding algorithm for the GPU, as well as with an algorithm that implements the spectral method on the GPU, depending on number of spatial points in buffer M and the quantity of harmonics N. As expected, the computation time using the spectral method on the GPU increases quadratically with an increase in the number of harmonics, which demonstrates its inefficiency in the case of strong nonlinear effects and a large number of harmonics. In this case, the time of calculations based on the Godunov method is proportional to the number of harmonics. For the dimensions of the pressure-field matrix with 10000 spatial points and, at 1000 harmonics, calculations implemented on the

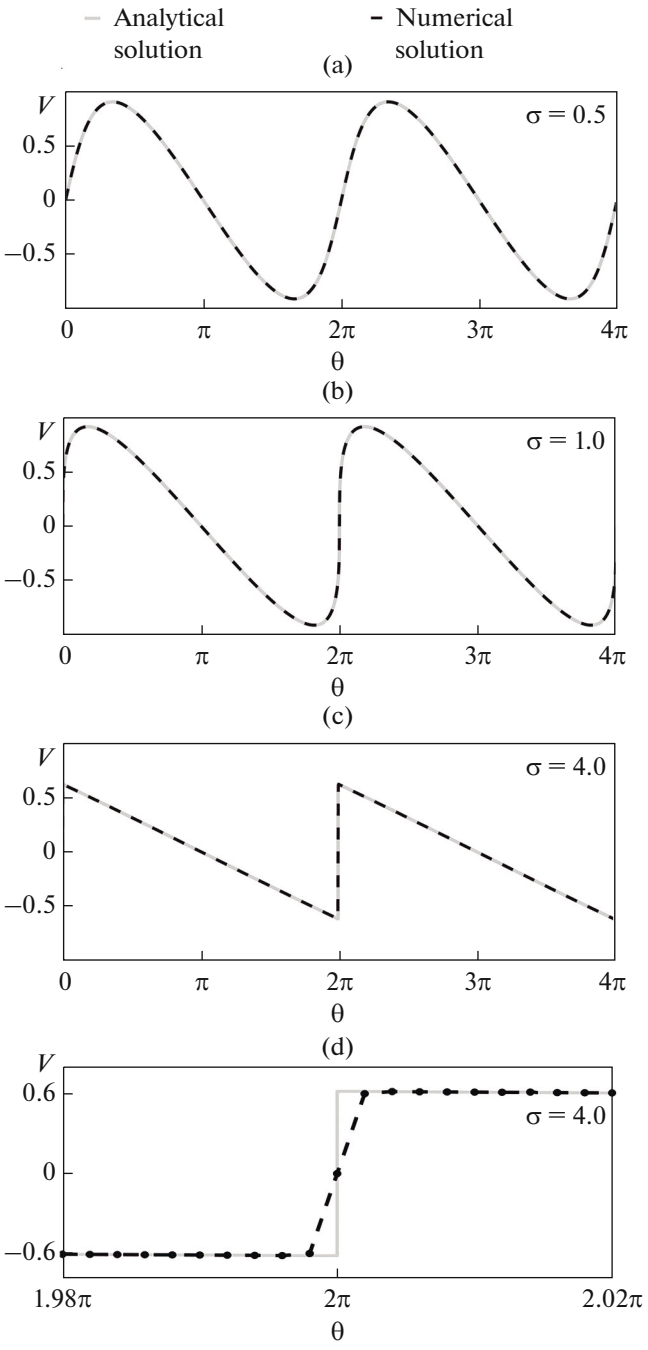

Fig. 4. Comparison of the analytical (solid gray line) and numerical (dashed black line) solutions for plane waves with a harmonic initial profile at different distances from the source: σ = (a) 0.5, (b) 1.0, and (c) 4.0. (d) Comparison of analytical and numerical solutions near the shock front at σ = 4.0.

GPU using the time approach are 26 times faster compared to the CPU and 2.6 times faster than the spectral algorithm implemented on the GPU.

Tables 3 and 4 compare the speed of the same programs on a more powerful computer (a PC). There are similar dependences of the time spent on calculations

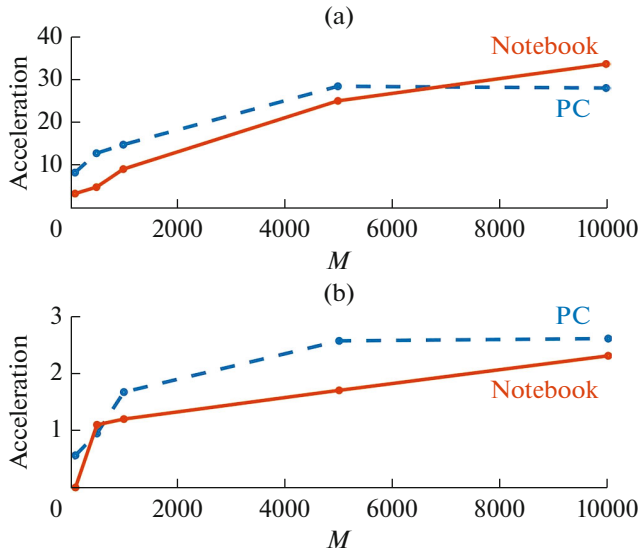

Fig. 5. Dependence of the acceleration of the algorithm based on the Godunov method implemented on the GPU: (a) compared to the version for the CPU, (b) compared to the spectral algorithm on the GPU, depending on buffer size *M*.

on the number of harmonics used for both methods. For matrix sizes with 10000 spatial points and 1000 harmonics, calculations implemented on the GPU using the time approach are 33 times faster compared to the CPU and 2.3 times faster than the spectral algorithm implemented on the GPU.

On Fig. 5, where the data on comparing the speed of the algorithms are summarized in the form of graphs, it can be seen that the acceleration for a laptop with an increase in both the number of spatial points and the number of harmonics (Fig. 6) reaches a certain value and then does not change. This is because the GPU on a laptop has a small number of cores capable of running parallel processes, as well as a small amount of memory, which limits the capacity of processed data. Therefore, with large parameters of the input array, it becomes necessary to transfer it from CPU memory to GPU memory in parts, which takes additional time. Also, a small number of cores makes it possible to process only the corresponding number of threads at the same time. The acceleration of calculations implemented on a more powerful computer (a PC) grows for all kinds of constructed dependences, due to the greater power and memory of its GPU.

In the case of comparing the speed of the algorithms implemented using the Godunov method on the CPU and GPU, the acceleration achieved on a PC is less than on a laptop. This is due to the higher power of the PC's CPU. An important result is a significant acceleration achieved in PC calculations with an increase in the number of harmonics in the case of

Table 2. Comparison of the speed of algorithms based on the Godunov method for a CPU and GPU and using the spectral method for a GPU, when performing calculations on a laptop, for number of points in space $M = 2500$ and different quantities of harmonics N

Number of harmonics N		50	250	500	1000	1500
Time T, s	CPU (TD Godunov)	73	371.5	738.6	1476	2214
	GPU (TD Godunov)	3.8	14.4	27.7	56.1	83.2
	GPU (FD RK4)	0.8	19.7	79.1	316	713
$T_{\mathrm{CPU_G}}/T_{\mathrm{GPU_G}}$		19.2	25.8	26.7	26.4	26.6
$T_{\mathrm{GPU_{RK4}}}/T_{\mathrm{GPU_G}}$		0.21	1.37	2.85	5.6	8.6

Table 3. Comparison of the speed of algorithms based on the Godunov method for a CPU and GPU and using the spectral method for a GPU, when performing calculations on a stationary computer, for number of harmonics $N = 1000$ for different-quantity M waveforms

Number of points in space M		100	500	1000	5000	10000
Time T, s	CPU (TD Godunov)	7.1	35.8	71.1	355	712
	GPU (TD Godunov)	2.1	7.3	7.8	14.1	21.2
	GPU (FD RK4)	0.03	8.0	9.0	24.4	48
$T_{\mathrm{CPU_G}}/T_{\mathrm{GPU_G}}$		3.4	4.9	9.1	25	33.6
$T_{\mathrm{GPU_{RK4}}}/T_{\mathrm{GPU_G}}$		0.01	1.1	1.2	1.7	2.3

Table 4. Comparison of the speed of algorithms based on the Godunov method for a CPU and GPU and using the spectral method for a GPU, when performing calculations on a stationary computer, for number of points in space $M = 2500$ of different numbers of harmonics N

Number of harmonics N		50	100	500	1000	1500
Time T, s	CPU (TD Godunov)	18.7	91.4	182.8	384	576
	GPU (TD Godunov)	1.6	7.0	13.7	27.4	40.9
	GPU (FD RK4)	1.4	35.6	142.4	569	1281
$T_{\mathrm{CPU_G}}/T_{\mathrm{GPU_G}}$		11.7	13.0	13.34	14.0	14.1
$T_{\mathrm{GPU_{RK4}}}/T_{\mathrm{GPU_G}}$		0.88	5.0	10.39	20.7	31.3

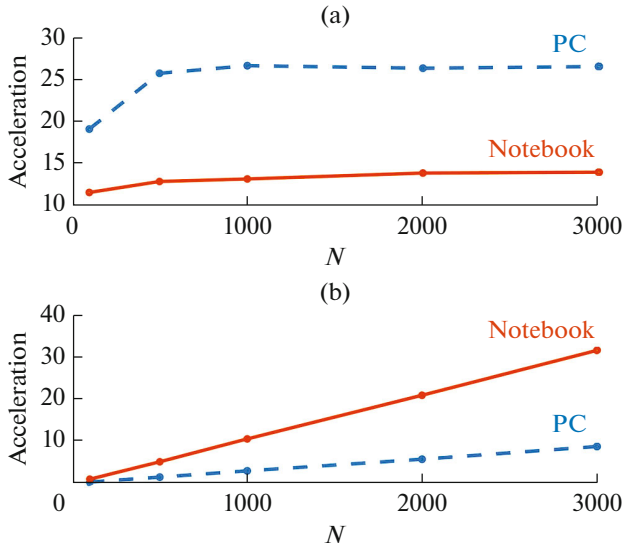

Fig. 6. Dependence of the acceleration of the algorithm based on the Godunov method implemented on the GPU: (a) in comparison with the version for the CPU, (b) in comparison with the spectral algorithm on the GPU, depending on number of harmonics N.

using the time method compared to the spectral method.

CONCLUSIONS

The paper implements a numerical algorithm for calculating nonlinear acoustic effects on graphics processing units using a Godunov-type shock-catching circuit. Calculations on a GPU made it possible to speed up the calculations compared to the algorithm for a CPU by an order of magnitude or more for both a laptop and a more powerful personal computer for all the sizes of the data array under consideration. In addition, for the characteristic number of harmonics ($N = 1000$) required to describe high-amplitude shock fronts, the implementation of the Godunov scheme on the GPU gave a gain of an order of magnitude in time compared to the spectral algorithm in calculations on a personal computer and one of five times on a laptop. Thus, it was shown that computations on graphics processors can be effectively used to solve nonlinear wave problems using shock-catching numerical schemes operating in time representation.

FUNDING

This study was financed by Russian Science Foundation grant no. 20-12-00145.

REFERENCES

1. M. R. Bailey, V. A. Khokhlova, O. A. Sapozhnikov, S. G. Kargl, and L. A. Crum, Acoiust. Phys. **49** (4), 369 (2003).
2. P. B. Rosnitskiy, P. V. Yuldashev, O. A. Sapozhnikov, et al., IEEE Trans. Ultrason. Ferr. Freq. Control **64** (2), 374 (2016).
3. I. A. Shvetsov, S. A. Shcherbinin, P. A. Astafiev, M. O. Moysa, and A. N. Rybyanets, Bull. Russ. Acad. Sci.: Phys. **82** (3), 355 (2018).
4. M. M. Karzova, P. V. Yuldashev, P. B. Rosnitskiy, and V. A. Khokhlova, Bull. Russ. Acad. Sci.: Phys. **81** (8), 927 (2017).
5. P. J. Westervelt, J. Acoust. Soc. Am. **35** (4), 535 (1963).
6. W. Kreider, P. V. Yuldashev, O. A. Sapozhnikov, N. Farr, A. Partanen, M. R. Bailey, and V. A. Khokhlova, IEEE Trans. Ultrason. Ferr. Freq. Control **60** (8), 1683 (2013).
7. K. Okita, K. Ono, S. Takagi, and Y. Matsumoto, Int. J. Numer. Methods Fluids **65** (1-3), 43 (2010).
8. W. F. Ames, *Numerical Methods for Partial Differential Equations,* 3rd ed. (Academic, San Diego, 2014), p. 380.
9. P. V. Yuldashev and V. A. Khokhlova, Acoust. Phys. **57** (3), 334 (2011).
10. C. R. Bawiec, T. D. Khokhlova, O. A. Sapozhnikov, et al., IEEE Trans. Ultrason. Ferroelectr. Freq. Control **68** (5), 1496 (2021).
11. E. E. Perepelkin, B. I. Sadovnikov, and N. G. Inozemtseva, *Calculations with Graphic Processors for the Problems of Mathematical and Theoretical Physics* (Lenand, Moscow, 2014) [in Russian].
12. E. O. Konnova, P. V. Yuldashev, and V. A. Khokhlova, Bull. Russ. Acad. Sci.: Phys. **85** (6), 632 (2021).
13. M. V. Aver'yanov, Candidate's Dissertation in Mathematics and Physics (MSU, Moscow, 2008).
14. Y.-S. Lee and M. F. Hamilton, J. Acoust. Soc. Am. **97** (2), 906 (1995).
15. A. Kurganov and E. Tadmor, J. Comput. Phys. **160** (1), 241 (2000).
16. S. S. Kashcheeva, O. A. Sapozhnikov, V. A. Khokhlova, M. A. Averkiou, and L. A. Crum, Acoust. Phys. **46** (2), 170 (2000).
17. O. V. Rudenko and S. I. Soluyan, *Theoretical Foundations for Nonlinear Acoustics* (Nauka, Moscow, 1975), p. 288 [in Russian].