COMPUTATIONAL MATHEMATICS
ВЫЧИСЛИТЕЛЬНАЯ МАТЕМАТИКА

Original Theoretical Research

UDC 519.6

https://doi.org/10.23947/2587-8999-2024-8-2-9-23

An Adaptive Mesh Refinement Solver for Regularized Shallow Water Equations

Ivan I. But[1,2], Maria A. Kiryushina[1,2]✉,
Stepan A. Elistratov[1,3], Tatiana G. Elizarova[2], Artem D. Tiniakov[1]
[1]Institute of System Programming of the Russian Academy of Sciences, Moscow, Russian Federation
[2]Keldysh Institute of Applied Mathematics of the Russian Academy of Sciences, Moscow, Russian Federation
[3]Shirshov Institute of Oceanology of the Russian Academy of Sciences, Moscow, Russian Federation
✉m_ist@mail.ru

Abstract

Introduction. We present a novel adaptive mesh refinement (AMR) solver, SWqgdAMR, based on the open software platform AMReX. The new solver is grounded in regularized shallow water equations. This paper details the equations, their discretization, and implementation features within AMReX. The efficacy of SWqgdAMR is demonstrated through two test cases: a two-dimensional circular dam break (collapse of a liquid column) and the collapse of two liquid columns of different heights.

Materials and Methods. The SWqgdAMR solver is developed to extend the applicability of regularized equations in problems requiring high computational power and adaptive grids. SWqgdAMR is the first solver based on the quasigas dynamic (QGD) algorithm within the AMReX framework. The implementation and validation of SWqgdAMR represent a crucial step towards the further expansion of the QGD software suite.

Results. The AMReX-based shallow water equations solver SWqgdAMR with adaptive mesh refinement is described and tested in detail. Validation of SWqgdAMR involved two-dimensional problems: the breach of a cylindrical dam and the breach of two cylindrical dams of different heights. The presented solver demonstrated high efficiency, with the use of adaptive mesh refinement technology accelerating the computation by 56 times compared to a stationary grid calculation.

Discussion and Conclusions. The algorithm can be expanded to include bathymetry, external forces (wind force, bottom friction, Coriolis forces), and the mobility of the shoreline during wetting and drying phases, as has been done in individual codes for regularized shallow water equations (RSWE). The current implementation of the QGD algorithm did not test the potential for parallel computing on graphical cores.

Keywords: shallow water equations, adaptive mesh refinement, quasigas dynamic equations, regularized shallow water equations, AMReX

Funding information. His work was supported by the Moscow Center for Fundamental and Applied Mathematics under Agreement no. 075-15-2022-283 with the Ministry of Science and Higher Education of the Russian Federation.

For citation: But I.I., Kiryushina M.A., Elistratov S.A., Elizarova T.G., Tiniakov A.D. An Adaptive Mesh Refinement Solver for Regularized Shallow Water Equations. *Computational Mathematics and Information Technologies*. 2024;8(2):9–23. https://doi.org/10.23947/2587-8999-2024-8-2-9-23

Оригинальное теоретическое исследование

Решатель с адаптивным измельчением сеток для регуляризованных уравнений мелкой воды

И.И. Бут[1,2] М.А. Кирюшина[1,2]✉,
С.А. Елистратов[1,3], Т.Г. Елизарова[2], А.Д. Тиняков[1]

[1]Институт системного программирования Российской академии наук, Москва, Российская Федерация
[2]Институт прикладной математики им. М.В. Келдыша Российской академии наук, Москва, Российская Федерация
[3]Институт океанологии им. П.П. Ширшова Российской академии наук, Москва, Российская Федерация
✉m_ist@mail.ru

Аннотация

Введение. Представлен новый решатель с адаптивным измельчением сеток SWqgdAMR на базе открытой программной платформы AMReX. Новый решатель основан на регуляризованных уравнениях мелкой воды. В работе описаны уравнения, их дискретизация и особенности реализации в AMReX. Работоспособность SWqgdAMR была показана на двух тестовых задачах: двумерная задача прорыва круговой дамбы (распад столба жидкости) и задача о распаде двух столбов жидкости, разных по высоте.

Материалы и методы. Решатель SWqgdAMR написан в рамках расширения применимости регуляризованных уравнений в задачах, требующих больших вычислительных мощностей и адаптивных сеток. SWqgdAMR является первым решателем на базе КГД алгоритма в программном комплексе AMReX. Реализация и валидация SWqgdAMR является основным шагом на пути дальнейшего расширения комплекса КГД программ.

Результаты исследования. Детально описан и протестирован решатель AMReX уравнений мелкой воды SWqgdAMR с адаптивным измельчением сеток. Для валидации SWqgdAMR использовались две двумерные задачи: о прорыве цилиндрической плотины и о прорыве двух цилиндрических плотин разной высоты. Представленный решатель показал высокую эффективность, а использование технологии адаптивного измельчения сетки позволило ускорить расчёт в 56 раз по сравнению с расчётом на стационарной сетке.

Обсуждение и заключения. В алгоритм может быть включена батиметрия дна, внешние силы (сила ветра, трение о дно, силы Кориолиса), а также учет подвижности береговой линии при осушении-наводнении, как это уже было сделано в рамках индивидуальных кодов для РУМВ. В данной реализации КГД алгоритма не тестировались перспективные возможности применения распараллеливания вычислений на графические ядра.

Ключевые слова: уравнения мелкой воды, адаптивное измельчение сеток, квазигазодинамические (КГД) уравнения, регуляризованные уравнения мелкой воды (РУМВ), AMReX

Финансирование. Работа выполнена при поддержке Московского центра фундаментальной и прикладной математики. Соглашение с Министерством науки и высшего образования РФ № 075-15-2022-283.

Для цитирования. Бут И.И., Кирюшина М.А., Елистратов С.А., Елизарова Т.Г., Тиняков А.Д. Решатель с адаптивным измельчением сеток для регуляризованных уравнений мелкой воды. *Computational Mathematics and Information Technologies*. 2024;8(2);9–23. https://doi.org/10.23947/2587-8999-2024-8-2-9-23

Introduction. Hydro- and gas-dynamics simulations require increasingly precise algorithms and detailed computational grids, which consequently demand substantial computational resources, including methods for parallel computing on GPU cores. Therefore, there is a need to develop a new solver with adaptive mesh refinement (AMR) based on open platforms. This approach offers several advantages over the development of custom codes. Firstly, open platforms typically provide well-established and thoroughly tested frameworks, endorsed by the broader scientific community, reducing the risk of errors and enhancing overall reliability. Secondly, the use of open platforms promotes functional compatibility and reusability, ensuring seamless integration with other tools and facilitating collaboration among researchers. Thirdly, employing existing open platforms can significantly reduce development time and costs, as these platforms often offer a wide range of functionalities, from data processing to visualization and parallel computing. Fourthly, open platforms benefit from continuous development and support from the user community, leading to regular updates, bug fixes, and performance improvements. This contrasts with custom codes, which often depend solely on the resources and expertise of the individual or team that created them.

Among the available open-source software, AMReX was selected as the most optimal framework. AMReX enables:

1. The use of adaptive mesh refinement (AMR) technology.
2. Parallel computation on GPU cores.
3. The immersed boundary method for simulating solid bodies in flow.
4. The construction of structured grids.

5. Integration into The High Performance Software Foundation, established by the Linux Foundation in 2023 [1], ensuring extensive support and ongoing development of this software package.

Numerical simulations of gas dynamics problems have already been conducted using AMReX, including comparisons between AMReX and OpenFOAM [2]. Thus, it was decided to implement a solver for hydro- and gas-dynamics problems based on the quasigas dynamic (QGD) equations within the AMReX framework. A similar solver has already been implemented in OpenFOAM [3, 4] under the general name QGDsolver, demonstrating high efficiency. Unfortunately, as previously noted, OpenFOAM [5] lacks the capabilities for parallel computation on GPU cores and adaptive mesh refinement.

This paper describes the implementation of the QGD algorithm in AMReX in a simplified form. The simplification involves a barotropic variant of the gas dynamics equation system, which allows for the elimination of the energy equation and the equation of state. Under certain assumptions, this barotropic variant takes the form of shallow water equations. It is worth noting that the implementation of the SWqgdAMR solver in the AMReX software package is a key step towards further expanding the suite of solvers based on QGD equations.

The QGD approach itself has been developed for over 30 years for gas dynamics and incompressible flow problems [6–10]. In recent years, the QGD approach has been implemented for shallow water approximation problems [11–16].

Mathematical Model and Numerical Method. Regularized Shallow Water Equations (RSWE). The RSWE can be expressed in vector form, in the absence of external forces and assuming a flat bottom, as follows:

$$\frac{\partial h}{\partial t} + \nabla \cdot \mathbf{j}_m = 0,$$
$$\frac{\partial (hu)}{\partial t} + \nabla \cdot (\mathbf{j}_m \otimes \mathbf{u}) + \nabla \frac{gh^2}{2} = \nabla \Pi, \tag{1}$$

where h is the water layer thickness; $j_m = h\,(\mathbf{u} - \mathbf{w})$ is the mass flux density vector; $\mathbf{u}$ is the velocity vector; g is the acceleration due to gravity; $\Pi = \Pi_{NS} + \Pi_{QGD}$ is the stress tensor; Π_{NS} is the Navier-Stokes viscous stress tensor; $\mathbf{w}$, Π_{QGD} are QGD terms; and $\otimes$ denotes the tensor product. Here, the nabla operator acting on a scalar denotes the gradient, on a vector denotes the divergence, and on a tensor denotes the covariant derivative: $\nabla T \equiv y|y_\beta = \nabla_\alpha T_{\alpha\beta}$. The form of RSWE considering the shape of the bottom and external forces can be found in [11–16].

Discretization of Regularized Shallow Water Equations.

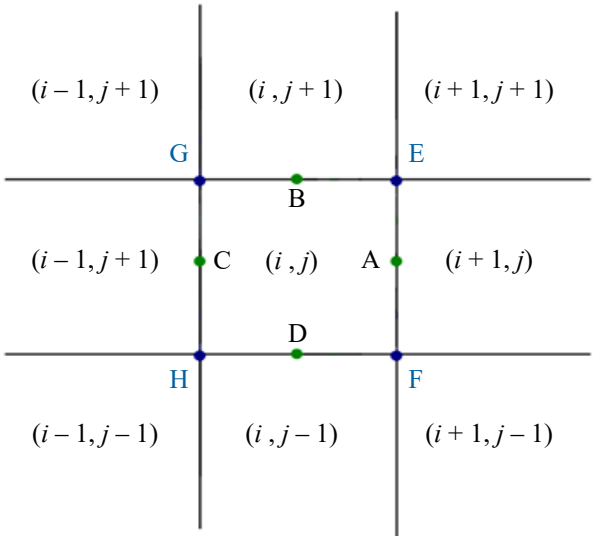

Fig. 1. Numerical stencil. The values of variables h and u are assigned to the cell centers with coordinates (i, j)

The component-wise form of the QGD shallow water equations is as follows:

$$\frac{\partial h}{\partial t} + \frac{\partial j_{mx}}{\partial x} + \frac{\partial j_{my}}{\partial y} = 0,$$
$$\frac{\partial hu_x}{\partial t} + \frac{\partial j_{mx}u_x}{\partial x} + \frac{\partial j_{my}u_x}{\partial y} + \frac{\partial}{\partial x}\left(\frac{gh^2}{2}\right) = \frac{\partial \Pi_{xx}}{\partial x} + \frac{\partial \Pi_{yx}}{\partial y},$$
$$\frac{\partial hu_y}{\partial t} + \frac{\partial j_{mx}u_y}{\partial x} + \frac{\partial j_{my}u_y}{\partial y} + \frac{\partial}{\partial y}\left(\frac{gh^2}{2}\right) = \frac{\partial \Pi_{xy}}{\partial x} + \frac{\partial \Pi_{yy}}{\partial y}.$$

$$j_{mx} = h(u_x - w_x), w_x = \frac{\tau}{h}\left(\frac{\partial(hu_x^2)}{\partial x} + \frac{\partial(hu_x u_y)}{\partial y} + gh\frac{\partial h}{\partial x}\right),$$

$$j_{my} = h(u_y - w_y), w_y = \frac{\tau}{h}\left(\frac{\partial(hu_y^2)}{\partial y} + \frac{\partial(hu_x u_y)}{\partial x} + gh\frac{\partial h}{\partial y}\right).$$

$$\Pi_{xx} = u_x\hat{w}_x + R + \Pi_{xx}^{NS}, \Pi_{xy} = u_x\hat{w}_y + \Pi_{xy}^{NS},$$
$$\Pi_{yx} = u_y\hat{w}_x + \Pi_{yx}^{NS}, \Pi_{yy} = u_y\hat{w}_y + R + \Pi_{yy}^{NS},$$

$$\Pi_{xx}^{NS} = h\nu\left(2\frac{\partial u_x}{\partial x} - \frac{2}{3}div\vec{u}\right), \Pi_{yy}^{NS} = h\nu\left(2\frac{\partial u_y}{\partial y} - \frac{2}{3}div\vec{u}\right),$$

$$\Pi_{xy}^{NS} = \Pi_{yx}^{NS} = h\nu\left(\frac{\partial u_y}{\partial x} + \frac{\partial u_x}{\partial y}\right), \quad \nu = \tau\frac{gh^2}{2}, div\vec{u} = \frac{\partial u_x}{\partial x} + \frac{\partial u_y}{\partial y},$$

$$\hat{w}_x = \tau\left(hu_x\frac{\partial u_x}{\partial x} + hu_y\frac{\partial u_x}{\partial y} + \frac{\partial}{\partial x}\left(\frac{gh^2}{2}\right)\right),$$
$$\hat{w}_y = \tau\left(hu_y\frac{\partial u_y}{\partial y} + hu_x\frac{\partial u_y}{\partial x} + \frac{\partial}{\partial y}\left(\frac{gh^2}{2}\right)\right),$$

$$R = g\tau\left(u_x\frac{\partial}{\partial x}\left(\frac{h^2}{2}\right) + u_y\frac{\partial}{\partial y}\left(\frac{h^2}{2}\right) + h^2\left(\frac{\partial u_x}{\partial x} + \frac{\partial u_y}{\partial y}\right)\right).$$

To discretize the equations spatially, we take into account the values at the half-cell points of the grid (Fig. 1):

$$h_A = h_{i+\frac{1}{2},j} = 0.5(h_{i,j} + h_{i+1,j}), h_B = h_{i,j+\frac{1}{2}} = 0.5(h_{i,j} + h_{i,j+1}),$$
$$h_C = h_{i-\frac{1}{2},j} = 0.5(h_{i,j} + h_{i-1,j}), h_D = h_{i,j-\frac{1}{2}} = 0.5(h_{i,j} + h_{i,j-1}),$$
$$h_E = h_{i+\frac{1}{2},j+\frac{1}{2}} = 0.25(h_{i,j} + h_{i+1,j} + h_{i,j+1} + h_{i+1,j+1}),$$
$$h_F = h_{i+\frac{1}{2},j-\frac{1}{2}} = 0.25(h_{i,j} + h_{i+1,j} + h_{i,j-1} + h_{i+1,j-1}),$$
$$h_G = h_{i-\frac{1}{2},j+\frac{1}{2}} = 0.25(h_{i,j} + h_{i-1,j} + h_{i,j+1} + h_{i-1,j+1}),$$
$$h_H = h_{i-\frac{1}{2},j-\frac{1}{2}} = 0.25(h_{i,j} + h_{i-1,j} + h_{i,j-1} + h_{i-1,j-1}).$$

Similarly, discretization of velocity components at half-cell points is recorded. Discretization of Mass Flux:

$$j_{xA} = j_{xi+\frac{1}{2},j} = h_{i+\frac{1}{2},j}\left(u_{xi+\frac{1}{2},j} - w_{xi+\frac{1}{2},j}\right),$$

$$j_{yB} = j_{yi,j+\frac{1}{2}} = h_{i,j+\frac{1}{2}}\left(u_{yi,j+\frac{1}{2}} - w_{yi,j+\frac{1}{2}}\right),$$

$$j_{xC} = j_{xi-\frac{1}{2},j} = h_{i-\frac{1}{2},j}\left(u_{xi-\frac{1}{2},j} - w_{xi-\frac{1}{2},j}\right),$$

$$j_{yD} = j_{yi,j-\frac{1}{2}} = h_{i,j-\frac{1}{2}}\left(u_{yi,j-\frac{1}{2}} - w_{yi,j-\frac{1}{2}}\right).$$

Discretization of QGD Terms:

$$w_{xA} = w_{xi+\frac{1}{2},j} = \frac{\tau_{i+\frac{1}{2},j}}{h_{i+\frac{1}{2},j}}\left(\frac{h_{i+1,j}u_{xi+1,j}u_{xi+1,j} - h_{i,j}u_{xi,j}u_{xi,j}}{\Delta x} + \right.$$
$$\left. + \frac{h_{i+\frac{1}{2},j+\frac{1}{2}}u_{xi+\frac{1}{2},j+\frac{1}{2}}u_{yi+\frac{1}{2},j+\frac{1}{2}} - h_{i+\frac{1}{2},j-\frac{1}{2}}u_{xi+\frac{1}{2},j-\frac{1}{2}}u_{yi+\frac{1}{2},j-\frac{1}{2}}}{\Delta y} + 0.5g\frac{h_{i+1,j}h_{i+1,j} - h_{i,j}h_{i,j}}{\Delta x}\right),$$

$$w_{yB} = w_{yi,j+\frac{1}{2}} = \frac{\tau_{i,j+\frac{1}{2}}}{h_{i,j+\frac{1}{2}}}\left(\frac{h_{i,j+1}u_{yi,j+1}u_{yi,j+1} - h_{i,j}u_{yi,j}u_{yi,j}}{\Delta y} + \right.$$

$$\left. + \frac{h_{i+\frac{1}{2},j+\frac{1}{2}}u_{xi+\frac{1}{2},j+\frac{1}{2}}u_{yi+\frac{1}{2},j+\frac{1}{2}} - h_{i-\frac{1}{2},j+\frac{1}{2}}u_{xi-\frac{1}{2},j+\frac{1}{2}}u_{yi-\frac{1}{2},j+\frac{1}{2}}}{\Delta x} + 0.5g\frac{h_{i,j+1}h_{i,j+1} - h_{i,j}h_{i,j}}{\Delta y} \right),$$

$$w_{xC} = w_{xi-\frac{1}{2},j} = \frac{\tau_{i-\frac{1}{2},j}}{h_{i-\frac{1}{2},j}}\left(\frac{h_{i,j}u_{xi,j}u_{xi,j} - h_{i-1,j}u_{xi-1,j}u_{xi-1,j}}{\Delta x} + \right.$$

$$\left. + \frac{h_{i-\frac{1}{2},j+\frac{1}{2}}u_{xi-\frac{1}{2},j+\frac{1}{2}}u_{yi-\frac{1}{2},j+\frac{1}{2}} - h_{i-\frac{1}{2},j-\frac{1}{2}}u_{xi-\frac{1}{2},j-\frac{1}{2}}u_{yi-\frac{1}{2},j-\frac{1}{2}}}{\Delta y} + 0.5g\frac{h_{i,j}h_{i,j} - h_{i-1,j}h_{i-1,j}}{\Delta x} \right),$$

$$w_{yD} = w_{yi,j-\frac{1}{2}} = \frac{\tau_{i,j-\frac{1}{2}}}{h_{i,j-\frac{1}{2}}}\left(\frac{h_{i,j}u_{yi,j}u_{yi,j} - h_{i,j-1}u_{yi,j-1}u_{yi,j-1}}{\Delta y} + \right.$$

$$\left. + \frac{h_{i+\frac{1}{2},j-\frac{1}{2}}u_{xi+\frac{1}{2},j-\frac{1}{2}}u_{yi+\frac{1}{2},j-\frac{1}{2}} - h_{i-\frac{1}{2},j-\frac{1}{2}}u_{xi-\frac{1}{2},j-\frac{1}{2}}u_{yi-\frac{1}{2},j-\frac{1}{2}}}{\Delta x} + 0.5g\frac{h_{i,j}h_{i,j} - h_{i,j-1}h_{i,j-1}}{\Delta y} \right),$$

$$w^*_{xA} = w^*_{xi+\frac{1}{2},j} = \tau_{i+\frac{1}{2},j}\left(h_{i+\frac{1}{2},j}u_{xi+\frac{1}{2},j}\frac{u_{xi+1,j} - u_{xi,j}}{\Delta x} + h_{i+\frac{1}{2},j}u_{yi+\frac{1}{2},j}\frac{u_{xi+\frac{1}{2},j+\frac{1}{2}} - u_{xi+\frac{1}{2},j-\frac{1}{2}}}{\Delta y} + \right.$$

$$\left. + 0.5g\frac{h_{i+1,j}h_{i+1,j} - h_{i,j}h_{i,j}}{\Delta x} \right),$$

$$w^*_{xB} = w^*_{xi,j+\frac{1}{2}} = \tau_{i,j+\frac{1}{2}}\left(h_{i,j+\frac{1}{2}}u_{xi,j+\frac{1}{2}}\frac{u_{xi+\frac{1}{2},j+\frac{1}{2}} - u_{xi-\frac{1}{2},j+\frac{1}{2}}}{\Delta x} + h_{i,j+\frac{1}{2}}u_{yi,j+\frac{1}{2}}\frac{u_{xi,j+1} - u_{xi,j}}{\Delta y} + \right.$$

$$\left. + 0.5g\frac{h_{i+\frac{1}{2},j+\frac{1}{2}}h_{i+\frac{1}{2},j+\frac{1}{2}} - h_{i-\frac{1}{2},j+\frac{1}{2}}h_{i-\frac{1}{2},j+\frac{1}{2}}}{\Delta x} \right),$$

$$w^*_{xC} = w^*_{xi-\frac{1}{2},j} = \tau_{i-\frac{1}{2},j}\left(h_{i-\frac{1}{2},j}u_{xi-\frac{1}{2},j}\frac{u_{xi,j} - u_{xi-1,j}}{\Delta x} + h_{i-\frac{1}{2},j}u_{yi-\frac{1}{2},j}\frac{u_{xi-\frac{1}{2},j+\frac{1}{2}} - u_{xi-\frac{1}{2},j-\frac{1}{2}}}{\Delta y} + \right.$$

$$\left. + 0.5g\frac{h_{i,j}h_{i,j} - h_{i-1,j}h_{i-1,j}}{\Delta x} \right),$$

$$w^*_{xD} = w^*_{xi,j-\frac{1}{2}} = \tau_{i,j-\frac{1}{2}}\left(h_{i,j-\frac{1}{2}}u_{xi,j-\frac{1}{2}}\frac{u_{xi+\frac{1}{2},j-\frac{1}{2}} - u_{xi-\frac{1}{2},j-\frac{1}{2}}}{\Delta x} + h_{i,j-\frac{1}{2}}u_{yi,j-\frac{1}{2}}\frac{u_{xi,j} - u_{xi,j-1}}{\Delta y} + \right.$$

$$\left. + 0.5g\frac{h_{i+\frac{1}{2},j-\frac{1}{2}}h_{i+\frac{1}{2},j-\frac{1}{2}}h_{i-\frac{1}{2},j-\frac{1}{2}}h_{i-\frac{1}{2},j-\frac{1}{2}}}{\Delta x} \right),$$

$$w^*_{yA} = w^*_{yi+\frac{1}{2},j} = \tau_{i+\frac{1}{2},j}\left(h_{i+\frac{1}{2},j}u_{xi+\frac{1}{2},j}\frac{u_{yi+1,j} - u_{yi,j}}{\Delta x} + h_{i+\frac{1}{2},j}u_{yi+\frac{1}{2},j}\frac{u_{yi+\frac{1}{2},j+\frac{1}{2}} - u_{yi+\frac{1}{2},j-\frac{1}{2}}}{\Delta y} + \right.$$

$$\left. + 0.5g\frac{h_{i+\frac{1}{2},j+\frac{1}{2}}h_{i+\frac{1}{2},j+\frac{1}{2}} - h_{i+\frac{1}{2},j-\frac{1}{2}}h_{i+\frac{1}{2},j-\frac{1}{2}}}{\Delta y} \right),$$

$$w^*_{yB} = w^*_{yi,j+\frac{1}{2}} = \tau_{i,j+\frac{1}{2}}\left(h_{i,j+\frac{1}{2}} u_{xi,j+\frac{1}{2}} \frac{u_{yi+\frac{1}{2},j+\frac{1}{2}} - u_{yi-\frac{1}{2},j+\frac{1}{2}}}{\Delta x} + h_{i,j+\frac{1}{2}} u_{yi,j+\frac{1}{2}} \frac{u_{yi,j+1} - u_{yi,j}}{\Delta y} + \right.$$
$$\left. + 0.5g \frac{h_{i,j+1}h_{i,j+1} - h_{i,j}h_{i,j}}{\Delta y} \right),$$

$$w^*_{yC} = w^*_{yi-\frac{1}{2},j} = \tau_{i-\frac{1}{2},j}\left(h_{i-\frac{1}{2},j} u_{xi-\frac{1}{2},j} \frac{u_{yi,j} - u_{yi-1,j}}{\Delta x} + h_{i-\frac{1}{2},j} u_{yi-\frac{1}{2},j} \frac{u_{yi-\frac{1}{2},j+\frac{1}{2}} - u_{yi-\frac{1}{2},j-\frac{1}{2}}}{\Delta y} + \right.$$
$$\left. + 0.5g \frac{h_{i-\frac{1}{2},j+\frac{1}{2}} h_{i-\frac{1}{2},j+\frac{1}{2}} - h_{i-\frac{1}{2},j-\frac{1}{2}} h_{i-\frac{1}{2},j-\frac{1}{2}}}{\Delta y} \right),$$

$$w^*_{yD} = w^*_{yi,j-\frac{1}{2}} = \tau_{i,j-\frac{1}{2}}\left(h_{i,j-\frac{1}{2}} u_{xi,j-\frac{1}{2}} \frac{u_{yi+\frac{1}{2},j-\frac{1}{2}} - u_{yi-\frac{1}{2},j-\frac{1}{2}}}{\Delta x} + h_{i,j-\frac{1}{2}} u_{yi,j-\frac{1}{2}} \frac{u_{yi,j} - u_{yi,j-1}}{\Delta y} + \right.$$
$$\left. + 0.5g \frac{h_{i,j}h_{i,j} - h_{i,j-1}h_{i,j-1}}{\Delta y} \right),$$

$$\mathrm{div}\vec{u}_A = \frac{u_{xi+1,j} - u_{xi,j}}{\Delta x} + \frac{u_{yi+\frac{1}{2},j+\frac{1}{2}} - u_{yi+\frac{1}{2},j-\frac{1}{2}}}{\Delta y}, \mathrm{div}\vec{u}_B = \frac{u_{yi,j+1} - u_{yi,j}}{\Delta y} + \frac{u_{xi+\frac{1}{2},j+\frac{1}{2}} - u_{xi-\frac{1}{2},j+\frac{1}{2}}}{\Delta x},$$
$$\mathrm{div}\vec{u}_C = \frac{u_{xi,j} - u_{xi-1,j}}{\Delta x} + \frac{u_{yi-\frac{1}{2},j+\frac{1}{2}} - u_{yi-\frac{1}{2},j-\frac{1}{2}}}{\Delta y}, \mathrm{div}\vec{u}_D = \frac{u_{yi,j} - u_{yi,j-1}}{\Delta y} + \frac{u_{xi+\frac{1}{2},j-\frac{1}{2}} - u_{xi-\frac{1}{2},j-\frac{1}{2}}}{\Delta x}.$$

The regularization parameter of the algorithm

$$\tau = \frac{\alpha \Delta_h}{\sqrt{gh}} \tag{2}$$

is calculated as

$$\tau_A = \tau_{i+\frac{1}{2},j} = \alpha \sqrt{\frac{\Delta x^2 + \Delta y^2}{g h_{i+\frac{1}{2},j}}}, \quad \tau_B = \tau_{i,j+\frac{1}{2}} = \alpha \sqrt{\frac{\Delta x^2 + \Delta y^2}{g h_{i,j+\frac{1}{2}}}}, \tag{3}$$

where α is a tuning parameter between 0 and 1; g is the acceleration due to gravity. A similar discretization is applied for terms τ_C, τ_D. The time step on the base computational grid is chosen to satisfy the stability condition for the explicit scheme, expressed as the Courant condition (Courant number $0 < \beta < 1$):

$$\Delta t = \beta \left(\frac{\Delta x + \Delta y}{2\sqrt{gh}} \right)_{\min}, \tag{4}$$

$$R_A = R_{i+\frac{1}{2},j} = \frac{g\tau_{i+\frac{1}{2},j}}{2}\left(u_{xi+\frac{1}{2},j} \frac{h_{i+1,j}h_{i+1,j} - h_{i,j}h_{i,j}}{\Delta x} + \right.$$
$$\left. + u_{yi+\frac{1}{2},j} \frac{h_{i+\frac{1}{2},j+\frac{1}{2}} h_{i+\frac{1}{2},j+\frac{1}{2}} - h_{i+\frac{1}{2},j-\frac{1}{2}} h_{i+\frac{1}{2},j-\frac{1}{2}}}{\Delta y} + 2h_{i+\frac{1}{2},j} h_{i+\frac{1}{2},j} \mathrm{div}(u)_A \right),$$

$$R_B = R_{i,j+\frac{1}{2}} = \frac{g\tau_{i,j+\frac{1}{2}}}{2}\left(u_{yi,j+\frac{1}{2}} \frac{h_{i,j+1}h_{i,j+1} - h_{i,j}h_{i,j}}{\Delta y} + \right.$$
$$\left. + u_{xi,j+\frac{1}{2}} \frac{h_{i+\frac{1}{2},j+\frac{1}{2}} h_{i+\frac{1}{2},j+\frac{1}{2}} - h_{i-\frac{1}{2},j+\frac{1}{2}} h_{i-\frac{1}{2},j+\frac{1}{2}}}{\Delta x} + 2h_{i,j+\frac{1}{2}} h_{i,j+\frac{1}{2}} \mathrm{div}(u)_B \right),$$

$$R_C = R_{i-\frac{1}{2},j} = \frac{g\tau_{i-\frac{1}{2},j}}{2}\left(u_{xi-\frac{1}{2},j}\frac{h_{i,j}h_{i,j} - h_{i-1,j}h_{i-1,j}}{\Delta x} + \right.$$
$$\left. + u_{yi-\frac{1}{2},j}\frac{h_{i-\frac{1}{2},j+\frac{1}{2}}h_{i-\frac{1}{2},j+\frac{1}{2}} - h_{i-\frac{1}{2},j-\frac{1}{2}}h_{i-\frac{1}{2},j-\frac{1}{2}}}{\Delta y} + 2h_{i-\frac{1}{2},j}h_{i-\frac{1}{2},j}\mathrm{div}(u)_C \right),$$

$$R_D = R_{i,j-\frac{1}{2}} = \frac{g\tau_{i,j-\frac{1}{2}}}{2}\left(u_{yi,j-\frac{1}{2}}\frac{h_{i,j}h_{i,j} - h_{i,j-1}h_{i,j-1}}{\Delta y} + \right.$$
$$\left. + u_{xi,j-\frac{1}{2}}\frac{h_{i+\frac{1}{2},j-\frac{1}{2}}h_{i+\frac{1}{2},j-\frac{1}{2}} - h_{i-\frac{1}{2},j-\frac{1}{2}}h_{i-\frac{1}{2},j-\frac{1}{2}}}{\Delta x} + 2h_{i,j-\frac{1}{2}}h_{i,j-\frac{1}{2}}\mathrm{div}(u)_D \right).$$

Discretization of the viscous stress tensor:

$$\Pi_{xxA} = \Pi_{xxi+\frac{1}{2},j} = u_{xi+\frac{1}{2},j}w^*_{xi+\frac{1}{2},j} + R_{xi+\frac{1}{2},j},\ \Pi_{xxB} = \Pi_{xxi-\frac{1}{2},j} = u_{xi-\frac{1}{2},j}w^*_{xi-\frac{1}{2},j} + R_{xi-\frac{1}{2},j},$$

$$\Pi_{xyA} = \Pi_{xyi+\frac{1}{2},j} = u_{xi+\frac{1}{2},j}w^*_{yi+\frac{1}{2},j},\qquad \Pi_{xyB} = \Pi_{xyi-\frac{1}{2},j} = u_{xi-\frac{1}{2},j}w^*_{yi-\frac{1}{2},j},$$

$$\Pi_{yxC} = \Pi_{yxi,j+\frac{1}{2}} = u_{yi,j+\frac{1}{2}}w^*_{xi,j+\frac{1}{2}},\qquad \Pi_{yxD} = \Pi_{yxi,j-\frac{1}{2}} = u_{yi,j-\frac{1}{2}}w^*_{xi,j-\frac{1}{2}},$$

$$\Pi_{yyC} = \Pi_{yyi,j+\frac{1}{2}} = u_{xi,j+\frac{1}{2}}w^*_{xi,j+\frac{1}{2}} + R_{xi,j+\frac{1}{2}},\ \Pi_{yyD} = \Pi_{xxi,j-\frac{1}{2}} = u_{xi,j-\frac{1}{2}}w^*_{xi,j-\frac{1}{2}} + R_{xi,j-\frac{1}{2}}.$$

Discretization of the mass conservation equation:

$$\hat{h}_{i,j} = h_{i,j} - \frac{\Delta t}{\Delta x}\left((j_x)_{i+\frac{1}{2},j} - (j_x)_{i-\frac{1}{2},j} \right) - \frac{\Delta t}{\Delta y}\left((j_y)_{i,j+\frac{1}{2}} - (j_y)_{i,j-\frac{1}{2}} \right).$$

Discretization of the momentum balance equations:

$$\hat{h}_{i,j}(\hat{u}_x)_{i,j} - \frac{\Delta t}{\Delta x}\left((u_x)_{i+\frac{1}{2},j}(j_x)_{i+\frac{1}{2},j} - (u_x)_{i-\frac{1}{2},j}(j_x)_{i-\frac{1}{2},j} \right) - \frac{\Delta t}{\Delta y}\left((u_x)_{i,j+\frac{1}{2}} - (j_y)_{i,j+\frac{1}{2}} - (u_x)_{i,j-\frac{1}{2}}(j_y)_{i,j-\frac{1}{2}} \right) -$$
$$- 0.5g\frac{\Delta t}{\Delta x}\left(h_{i+\frac{1}{2},j}h_{i+\frac{1}{2},j} - h_{i-\frac{1}{2},j}h_{i-\frac{1}{2},j} \right) + \frac{\Delta t}{\Delta x}\left((\Pi_{xx})_{i+\frac{1}{2},j} - (\Pi_{xx})_{i-\frac{1}{2},j} \right) + \frac{\Delta t}{\Delta y}\left((\Pi_{yx})_{i,j+\frac{1}{2}} - (\Pi_{yx})_{i,j-\frac{1}{2}} \right),$$

$$\hat{h}_{i,j}(\hat{u}_y)_{i,j} = h_{i,j}(u_y)_{i,j} - \frac{\Delta t}{\Delta y}\left((u_y)_{i,j+\frac{1}{2}}(j_y)_{i,j+\frac{1}{2}} - (u_y)_{i,j-\frac{1}{2}}(j_y)_{i,j-\frac{1}{2}} \right) -$$
$$- \frac{\Delta t}{\Delta x}\left((u_y)_{i+\frac{1}{2},j}(j_x)_{i+\frac{1}{2},j} - (u_y)_{i-\frac{1}{2},j}(j_x)_{i-\frac{1}{2},j} \right) - 0.5g\frac{\Delta t}{\Delta y}\left(h_{i,j+\frac{1}{2}}h_{i,j+\frac{1}{2}} - h_{i,j-\frac{1}{2}}h_{i,j-\frac{1}{2}} \right) +$$
$$+ \frac{\Delta t}{\Delta x}\left((\Pi_{yy})_{i,j+\frac{1}{2}} - (\Pi_{yy})_{i,j-\frac{1}{2}} \right) + \frac{\Delta t}{\Delta x}\left((\Pi_{xy})_{i+\frac{1}{2},j} - (\Pi_{xy})_{i-\frac{1}{2},j} \right).$$

Implementation in AMReX. The numerical solution of the shallow water equations is implemented in C++ using the open-source software AMReX. This software was chosen as the foundation because it facilitates ready-to-use adaptive mesh refinement (AMR) logic and offers straightforward portability of computations to GPU cores via macros, significantly reducing computational time.

Figure 2 shows the structure of the developed software.

The main solver class, AmrSWQGD, is declared in the file AmrSWQGD.H and implemented in the file AmrSWQGD.cpp. It inherits from the AmrLevel class, defined in the AMReX core. Inheriting from this class allows straightforward adaptive mesh refinement across multiple levels.

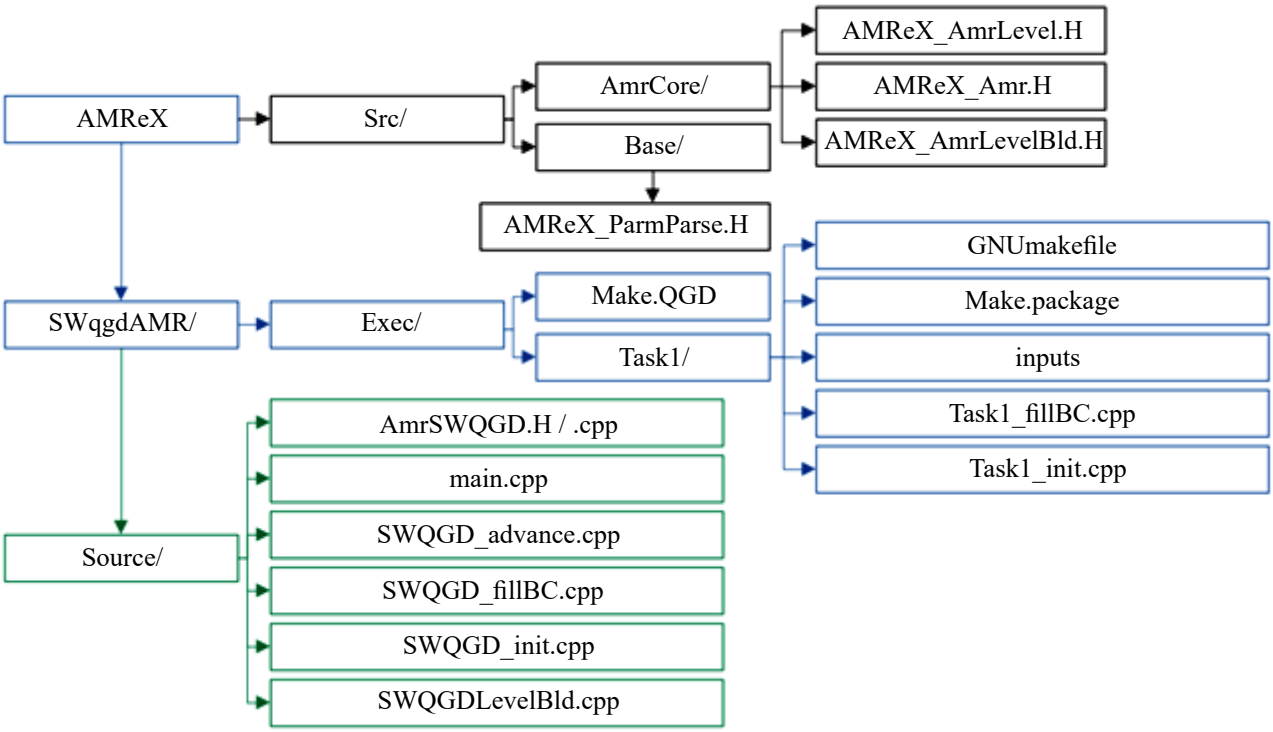

Fig. 2. Solver structure

The solver iteration logic is defined in the file SWQGD_advance.cpp within the advance method. In this method, a ParallelFor loop updates the fields h, ux and uy at each iteration. Here, huOld is the container for old variables, and huNew is for new variables. Since the solution occurs on a multi-level grid (Fig. 3), where each level has its own refinement (level 0 is the coarsest grid, and higher levels have increased accuracy), each level introduces its own time step (Fig. 4).

For example, if the grid has two levels, 0 and 1, and the grid at level 1 is twice as fine in each direction as at level 0, then one iteration of the solution proceeds as follows: calculations are performed at level 0 with a time step dt, two iterations of calculations are performed at level 1 with a time step dt/2, and then the grids are synchronized. This algorithm enhances computational accuracy.

It is important to note that it is not necessary to refine the entire grid at each level, only specific parts of it. To achieve this, the solver class defines the errorEst method in the file AmrSWQGD.cpp. This method takes a reference to an instance of the TagBoxArray container. Using a ParallelFor loop, each grid cell is examined and marked for refinement if it meets certain conditions (defined within an if statement). Additionally, some surrounding cells are marked for refinement. Cells that do not meet the condition are marked with the clearval tag and will not be refined.

The computational tasks themselves are located in the Exec directory, which contains the inputs files with initial and boundary conditions.

The inputs file contains the settings for the solution, including various parameters that control the behavior of the solver. Here are the key parameters and their descriptions:

• max_step: The maximum number of iterations;

• stop_time: The computational time in seconds at which the solution stops. Essentially, the calculations continue until either the number of iterations exceeds max_step or the computational time reaches stop_time;

• geometry.is_periodic: An array of three boolean variables (e.g., 0 0 0, 0 1 1, or 1 0 1) that determine whether the boundaries in each direction are periodic (1) or not (0);

• geometry.coord_sys: The coordinate system used for the solution. The recommended value is 0, which corresponds to the Cartesian coordinate system. There is no guarantee that the solver will work correctly in other coordinate systems;

• geometry.prob_lo: The xyz coordinates of the lower left corner of the physical rectangular domain (e. g., 0.0 0.0 0.0);

• geometry.prob_hi: The xyz coordinates of the upper right corner of the physical domain (e.g., 10.0 10.0 1.0);

• amr.n_cell: An array of three integers representing the grid resolution in each direction at level 0 (e.g., 512 512 1);

• amr.max_level: An integer indicating the maximum allowable level of grid refinement;

• amr.ref_ratio: The refinement ratio of the grid levels;

• amr.regrid_int: An integer representing the number of steps after which the grid should be regenerated;

• amr.max_grid_size: The solver divides the grid into domains, with the size of each domain not exceeding max_grid_size in each direction;

• amr.plot_int: The solver writes output to files every plot_int iterations.

The boundary conditions are defined in the structure SWQGDBCFill located in the file Task1_fillBC.cpp (where Task1 is the task name and can be changed to any other name). In this structure: ilo and ihi represent the left and right boundaries of the computational domain, respectively, jlo and jhi represent the bottom and top boundaries, respectively. The equations are solved for the variables h, u_x and u_y, which represent the water column height, the velocity in the x-direction, and the velocity in the y-direction, respectively. The variable dest is a multidimensional array that stores the values of all variables throughout the computational domain. To access the variable h at cell (i, j, k) you use dest(i, j, k, 0). To access the variable ux in the same cell, you use dest(i, j, k, 1), and for uy, you use dest(i, j, k, 2). ince we are dealing with 2D equations, the index k is assumed to be 1.

Initial conditions are set in the file Task1_init.cpp. Here, the variable snew[bi] acts as a container similar to the dest container in the boundary conditions structure. Initial conditions are computed in a loop using amrex::ParallelFor. This loop, along with the macro-lambda function AMREX_GPU_DEVICE, allows for the parallel computation of initial conditions across the entire domain on GPU cores.

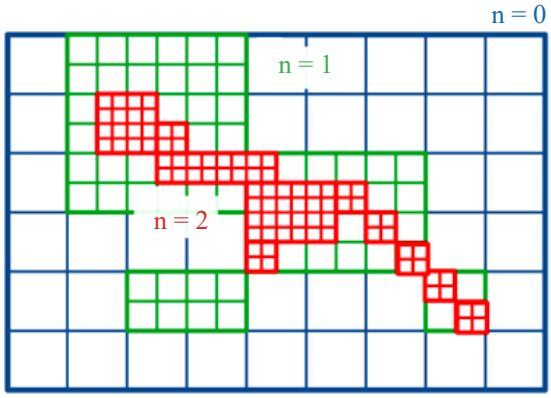

Fig. 3. Adaptive Mesh Refinement Algorithm

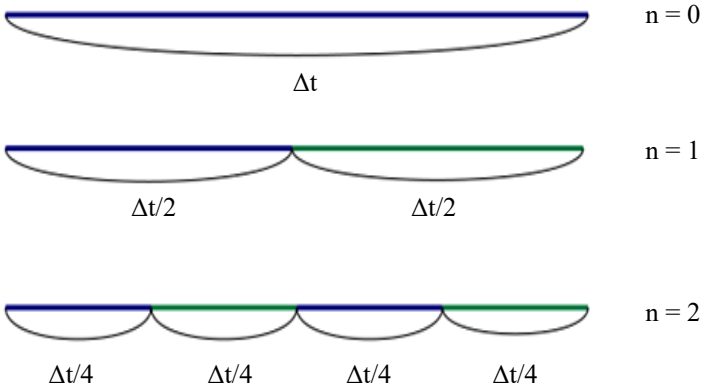

Fig. 4. Time Step Splitting Algorithm in Subcycle

To compile the program, navigate to the task folder (Task1) and then run the command make –j n in the terminal, where n is the number of cores for parallelization. To run the utility without parallelization, simply run make. After make completes, a file named main2d.gnu.MPI.ex (the name may vary slightly) will appear in the folder. To start the calculation, execute the following command in the terminal: **mpiexec –np n ./main2d.gnu.MPI.ex inputs**, where n is the number of cores for parallelization. This will start the computation.

Results. Two 2D problems are used for the validation and verification of the developed solver:

The dam break problem, for which an analytical solution is well known.

The collapse of two liquid columns of different heights.

Two-Dimensional Circular Dam-Break Problem. The problem of liquid column collapse or the breakthrough of a circular dam (Circular Dam-Break) is widely used in the validation and verification of new solvers [17–20]. Consider a 2D plane with dimensions 40×40 m, where at the center resides a liquid column with height h = 2.5 m and radius R = 2.5 m. The height of the liquid in the rest of the domain is h_0 = 0.5 m (see Fig. 5). The computational domain is divided into 40,000 uniform cells, i. e., 200 cells in each direction. The time step is chosen as $\Delta t = 10^{-4}$ s, and the calculation is carried out until time t = 4.7 s.

Visualization in Figure 6 illustrates the liquid column collapse. Initially, the wall is removed, allowing the water to move in all directions. As the circular shock wave propagates outward, a rarefaction wave moves inward into the original cylinder until it completely converges at the center of the computational domain, where it reflects, causing a height gradient and hence a secondary shock wave. Results of numerical experiments compared with the analytical solution from [19] at time t = 4.7 s are shown in Fig. 7. Panel (a) of Fig. 7 demonstrates the dependence of the solution on the algorithm tuning parameter α. The optimal value is α = 0.2. Panel (b) of Figure 7 illustrates the convergence of the solution with grid refinement. The characteristic Courant number is 0.2.

An example of the adaptive mesh refinement algorithm can be seen in Figure 8. Depending on the chosen adaptation criterion (in our case, the gradient of the water column height), the mesh is refined across levels (in our case $n_{amr} = 4$, where n_{am} is the maximum level in the current calculation), significantly accelerating the computation. Detailed investigation is presented in the section on performance evaluation of the SWqgdAMR solver.

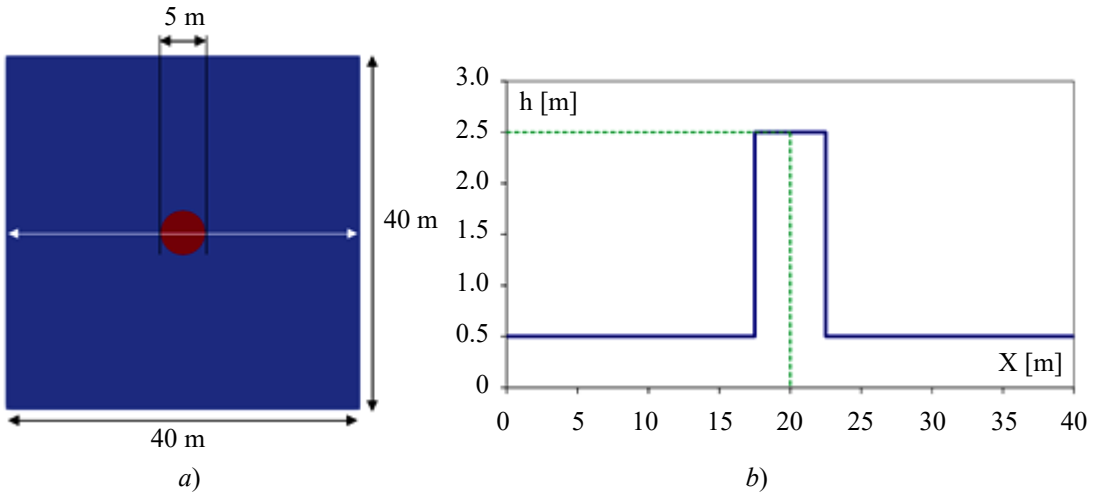

Fig. 5. Initial Conditions of the Circular Dam-Break Problem: *a* — Computational domain geometry and initial distribution; *b* — Height of the liquid column along the white line

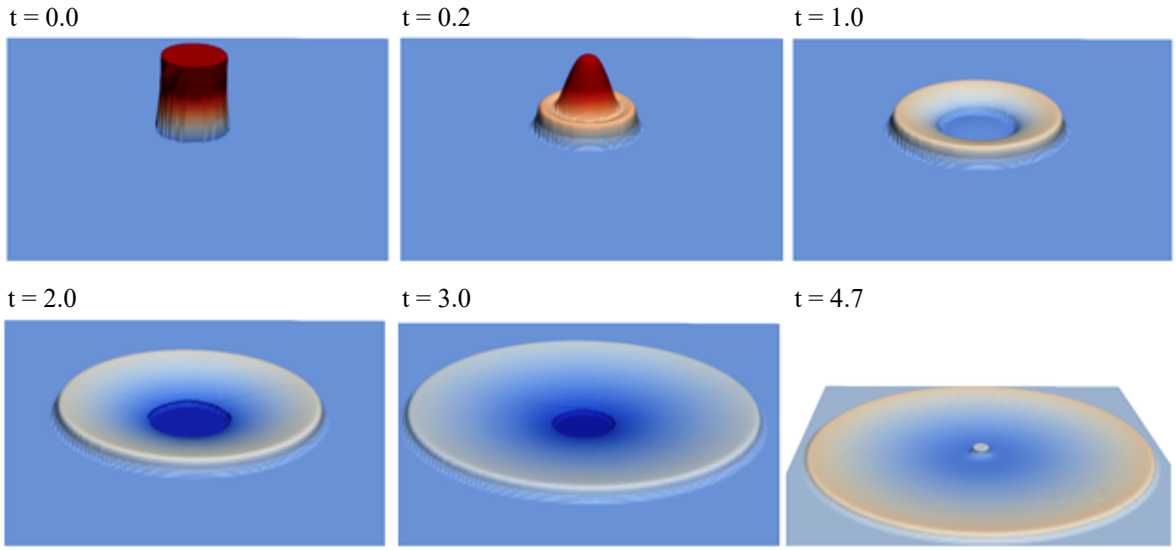

Fig. 6. Visualization of Liquid Column Collapse Over Time α = 0,2, $\Delta t = 10^{-4}$ s.
Time in seconds on the figure

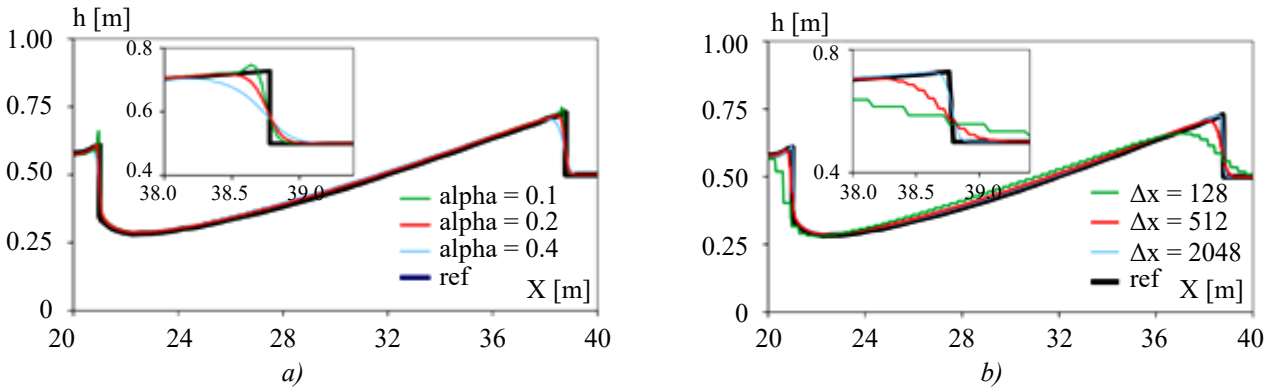

Fig. 7. Distribution of Liquid Column Height with Time Step $\Delta t = 10^{-4}$ s at $t = 4.7$ s:
a — Dependence on parameter α on a fixed grid $\Delta x = 1024$;
b — Grid convergence with constant α = 0.2

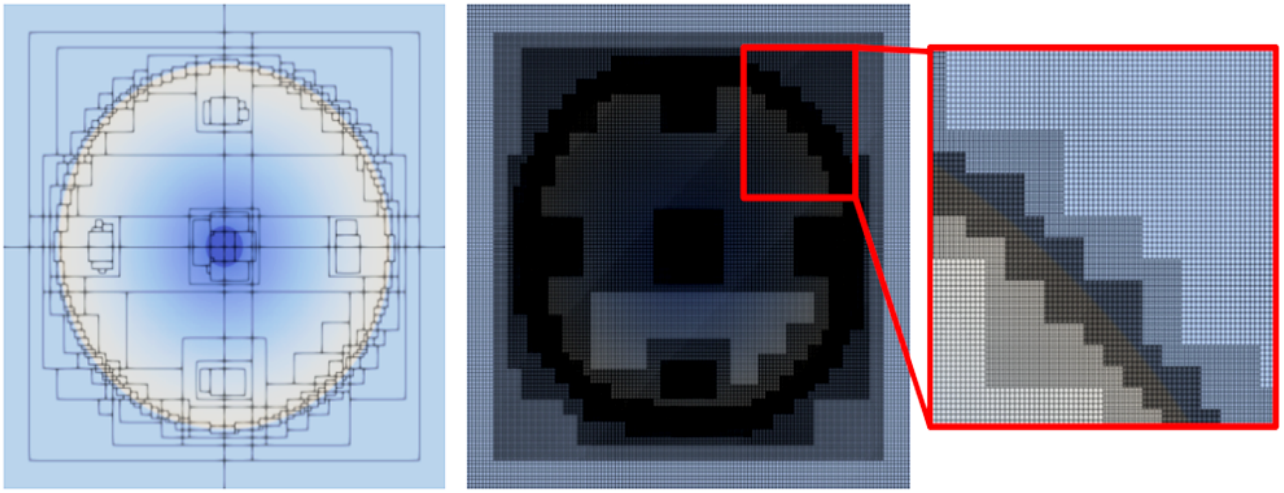

Fig. 8. Visualization of Adaptive Mesh Refinement Algorithm at α = 0.2 at time t = 3 s. The first image shows the block decomposition, while the second image depicts the adaptive mesh

The collapse of two different-height liquid columns problem allows the solver's capability to reproduce complex flow structures to be tested, similar to the previous test. Considered is a 2D plane of size 2000×2000 m, with the first water column located at (875.0) having a radius R_1 = 125 m and height h_1 = 15 m. At coordinates (1375.0) the second water column has radius R_2 = 125 m, h_2 = 20 m, with the water level in the remaining area set to h_1 = 10 m (see Fig. 9). The computational domain is divided into 160,000 uniform cells, i. e., 400 cells in each direction. The time step is chosen as $\Delta t = 10^{-4}$ s, and the computation concludes at time t = 30 s.

Fig. 10 and 11 visualize the collapse and subsequent interaction of the two liquid columns. Initially, the walls are removed, allowing water to move in all directions from each column. Subsequently, two shock waves collide, resulting in significant deformation of the wave fronts.

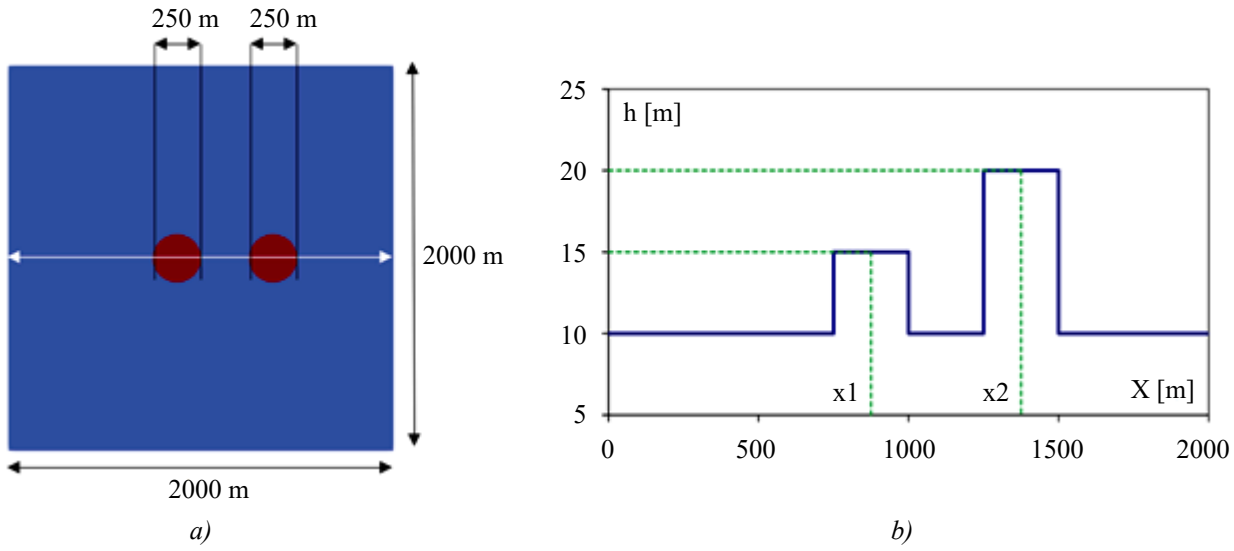

Fig. 9. Initial Conditions of the Liquid Column Collapse Problem: *a* — geometry of the computational domain; *b* — height of the liquid column along the white line, $x1$ = 875 m, $x2$ = 1375 m

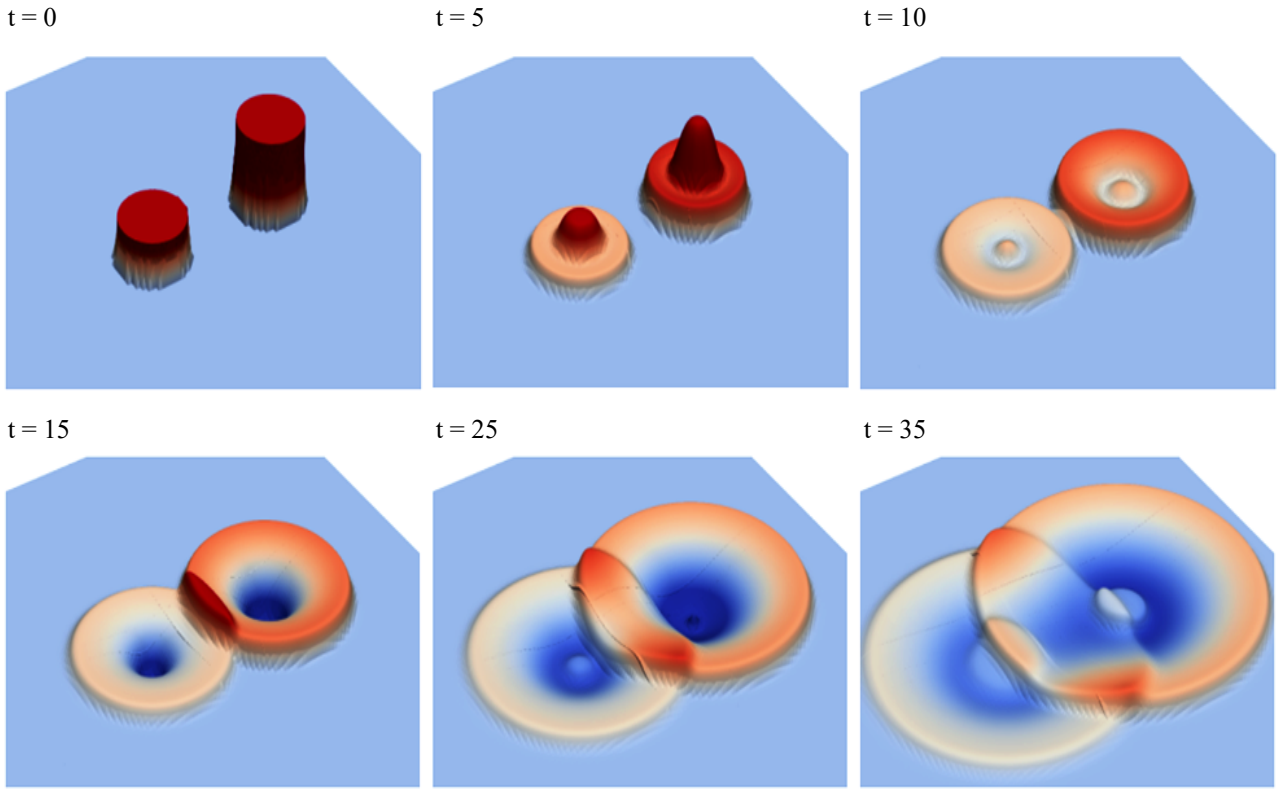

Fig. 10. Visualization of the collapse of two liquid columns over time $\alpha = 0.2$, $\Delta t = 10^{-4}$ s.
Time in the figure is in seconds

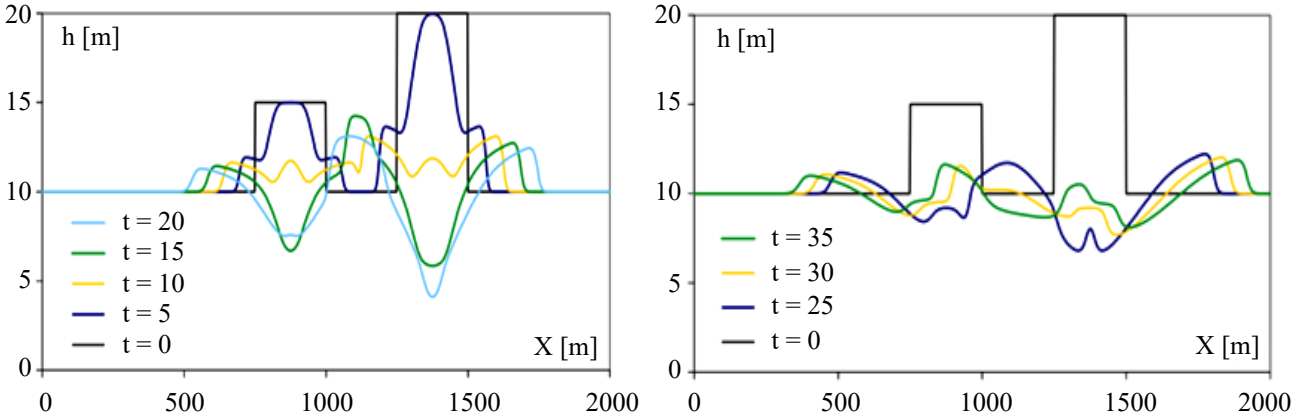

Fig. 11. Plot of the collapse of two liquid columns over time in the central cross-section.
Time in the figure is in seconds, $\Delta x = 1024$, $\alpha = 0.2$, $\Delta t = 10^{-4}$ s

Performance of the SWqgdAMR Solver. One of the crucial criteria in developing a new solver is assessing its performance and the efficiency of parallelization. For this purpose, the problem of the collapse of two different-height liquid columns was utilized. A computational grid of 1.048.576 cells, a time step of $\Delta t = 10^{-4}$ s, and computation completion at $t = 0.1$ c. Performance evaluation was conducted on an Intel(R) Core(TM) i7-9700 CPU @ 3.00GHz with 8 cores, and the results are presented in Table 1.

Table 1

Performance Evaluation of the SWqgdAMR Solver

Number of Cores	Number of Cells	Computation Time t, s	Efficiency, %
1	1 048 576	71	–
2	524 288	40	89
4	262 144	27	66

Using the same processor with 2 cores, acceleration of computations was investigated through the use of adaptive mesh refinement technology. The data are presented in Table 2. The number of computational cells at $n_{amr} = 0$ is 4 194 304, with a base grid at $n_{amr} = 1$ of 1 048 576, at $n_{amr} = 2$ of 262 144, at $n_{amr} = 4$ base computational grid of 65 536. In all cases, flow resolution remains constant, with quality varying depending on the adaptation criterion (an example of mesh refinement algorithm operation is shown in Figure 8). In our calculations, the gradient of the liquid column height was chosen as the mesh adaptation criterion.

Table 2

Computation time in seconds depending on levels of mesh adaptation

Number of Cores	Number of Cells	$n_{amr} = 0$	$n_{amr} = 1$	$n_{amr} = 2$	$n_{amr} = 4$
2	4 194 304	2288	359	132	41

Thus, the use of adaptive mesh refinement technology allows for significant acceleration of computations. In our case, acceleration of up to 56 times was achieved. In [21], it was demonstrated that on identical stationary grids, AMReX is 4 times faster than OpenFOAM, indicating that with mesh refinement, speed gains of up to 232 times can be achieved.

Discussion and Conclusions. The AMReX solver for shallow water equations (SWqgdAMR) with adaptive mesh refinement (AMR) was comprehensively described and tested in this work. Two 2D test cases were used for validation: the breach of a cylindrical dam and the breach of two cylindrical dams of different heights. The presented solver demonstrated high efficiency, and the use of adaptive mesh refinement technology accelerated computations by a factor of 56 compared to computations on a stationary grid.

The SWqgdAMR solver was developed as part of efforts to expand the applicability of regularized equations in problems requiring significant computational resources and adaptive grids. It represents the first solver based on the shallow water equations algorithm within the AMReX software framework. The implementation and validation of SWqgdAMR represent a crucial step towards further expanding the suite of shallow water equations programs. Future work will include incorporating quasi-gasdynamic equations into AMReX for simulating gas dynamics problems.

In this implementation, the prospective capabilities of leveraging graphics processing unit (GPU) computing for parallel computation were not tested. Additionally, it is noteworthy that the algorithm could be extended to include bathymetry, external forces (such as wind force, bottom friction, and Coriolis forces), and consideration of shoreline mobility during flooding and drying, as has been implemented in individual codes for hydrodynamic simulations.

References

1. TheLinuxFoundation. Linux Foundation Announces Intent to Form the High Performance Software Foundation. URL: https://www.linuxfoundation.org/press/linux-foundation-announces-intent-to-form-high-performance-software-foundation-hpsf (accessed: 16.04.2024).

2. Epikhin A., But I. Numerical Simulation of Supersonic Jet Noise Using Open Source Software. *International Conference on Computational Science*. Springer. 2023. P. 292–302.

3. Kraposhin M.V. et al. Development of a new OpenFOAM solver using regularized gas dynamic equations. *Computers & Fluids*. 2018;166:163–175.

4. Kraposhin M.V., Ryazanov D.A., Elizarova T.G. Numerical algorithm based on regularized equations for incompressible flow modeling and its implementation in OpenFOAM. *Computer Physics Communications*. 2022;271: 108216.

5. QGDsolver. URL: https://github.com/unicfdlab/QGDsolver (accessed: 16.04.2024).

6. Elizarova T.G. *Quasi-gas-dynamic Equations*. Springer. 2009.

7. Elizarova T.G. *Quasi-gas-dynamic Equations and Methods for Calculating Viscous Flows*. Moscow: Nauchnyi Mir; 2007. 350 p. (in Russ.).

8. Chetverushkin B.N. *Kinetic Schemes and Quasi-gas-dynamic System of Equations*. Moscow: MAKS Press; 2004. 328 p. (in Russ.).

9. Sheretov Yu.V. *Regularized Equations of Hydrodynamics*. Tver: Tver State University; 2016. 222 p. (in Russ.).

10. Sheretov Yu.V. *Dynamics of Continua under Space-Time Averaging*. Moscow; Izhevsk: Regular and Chaotic Dynamics; 2009. 400 p. (in Russ.).

11. Bulatov O., Elizarova T.G. Regularized shallow water equations and an efficient method for numerical simulation of shallow water flows. *Computational mathematics and mathematical physics*. 2011;51:160–173.

12. Elizarova T.G., Ivanov A.V. Regularized equations for numerical simulation of flows in the two-layer shallow water approximation. *Computational Mathematics and Mathematical Physics*. 2018;58:714–734.

13. Saburin D.S., Elizarova T.G. Modelling the Azov Sea circulation and extreme surges in 2013–2014 using the regularized shallow water equations. *Russian Journal of Numerical Analysis and Mathematical Modelling*. 2018;33(3):173–185.

14. Bulatov O.V., Elizarova T.G. Regularized Shallow Water Equations and an Efficient Method for Numerical Simulation of Flows in Shallow Water Bodies. *Journal of Computational Mathematics and Mathematical Physics*. 2011;51(1):170–184. (in Russ.).

15. Elizarova T.G., Ivanov A.V. Regularized Equations for Numerical Simulation of Flows in the Two-Layer Shallow Water Approximation. *Journal of Computational Mathematics and Mathematical Physics*. 2018;58(5):741–761. (in Russ.).

16. Ivanov A.V. On the Implementation of the Shallow Water Model Based on the Quasi-gas-dynamic Approach in the OpenFOAM Open Source Software. *Preprints of the Institute of Applied Mathematics named after M.V. Keldysh RAS*. 2023;28:27. (in Russ.).

17. Delis A., Nikolos I. A novel multidimensional solution reconstruction and edge-based limiting procedure for unstructured cell-centered finite volumes with application to shallow water dynamics. *International Journal for Numerical Methods in Fluids*. 2013;71(5):584–633.

18. Delis A.I., Katsaounis T. Numerical solution of the two-dimensional shallow water equations by the application of relaxation methods. *Applied Mathematical Modelling*. 2005;29(8):754–783.

19. Ginting B.M., Mundani R.-P. Comparison of shallow water solvers: Applications for dam-break and tsunami cases with reordering strategy for efficient vectorization on modern hardware. *Water*. 2019;11(4):639.

20. Soares-Fraz̃ao S., Zech Y. Experimental study of dam-break flow against an isolated obstacle. *Journal of Hydraulic Research.* 2007;45(1):27–36.

21. Britov A. et al. Numerical Simulation of Propeller Hydrodynamics Using the Open Source Software. *International Conference on Computational Science*. Springer. 2023. P. 279–291.

Received 20.05.2024
Received 03.06.2024
Accepted 04.06.2024

About the Authors:

Ivan I. But, research engineer, Open-source Software Laboratory for Digital Modelling of Technical Systems, Ivannikov Institute for System Programming of the RAS (25, Aleksandr Solzhenitsyn st., Moscow, 109004, RF), Junior Researcher, Keldysh Institute of Applied Mathematics of RAS (4, Miusskaya sq., Moscow, 125047, RF), ORCID, ivan.but@ispras.ru

Maria A. Kiryushina, research associate of Keldysh Institute of Applied Mathematics of Russian Academy of Sciences (KIAM RAS), Ivannikov Institute for System Programming of the Russian Academy of Sciences (ISP RAS) (25, Aleksandr Solzhenitsyn st., Moscow, 109004, RF), phd, ORCID, ResearcherID, m_ist@mail.ru

Stepan A. Elistratov, junior researcher, Shirshov Institute of oceanology of RAS (36, Nakhimovsky ave., Moscow, 117218, RF), research intern, Ivannikov Institute for system programming of RAS (25, Aleksandra Solzhenitsyna st., Moscow, 109004, RF), research engineer, Sobolev Institute of mathematics of Siberian branch of RAS (4, Akademika Koptyuga ave., Novosibirsk, 630090, RF), ORCID, sa.elist-ratov@yandex.ru

Tatiana G. Elizarova, chief researcher of Keldysh Institute of Applied Mathematics of Russian Academy of Sciences (KIAM RAS) (4, Miusskaya sq., Moscow, 125047, RF), doctor of physico-mathematical sciences, professor, ORCID, ResearcherID, ScopusID, telizar@mail.ru

Artem D. Tiniakov, 5th year student of the Department of Information Transmission and Data Analysis Problems, Moscow Institute of Physics and Technology (9, Institutskiy pereulok, Dolgoprudny, 141701, RF), ORCID, tiniakov.ad@gmail.com

Contributions of the co-authors:

All authors have made an equivalent contribution to the preparation of the publication.

Conflict of interest statement

The authors do not have any conflict of interest.

All authors have read and approved the final manuscript.

Поступила в редакцию 20.05.2024
Поступила после рецензирования 03.06.2024
Принята к публикации 04.06.2024

Об авторах:

Бут Иван Игоревич, инженер-исследователь лаборатории цифрового моделирования технических систем, Институт системного программирования им. В.П. Иванникова РАН (РФ, 109004, Москва, ул. Александра Солженицына, 25); младший научный сотрудник, Институт прикладной математики им. М.В. Келдыша РАН (РФ, 125047, Москва, Миусская пл., 4), ORCID, ivan.but@ispras.ru

Кирюшина Мария Александровна, научный сотрудник ИПМ им. М.В. Келдыша РАН (РФ, 125047, Москва, Миусская пл., 4); ИСП им. В.П. Иванникова РАН (РФ, 109004, Москва, ул. Александра Солженицына, 25), кандидат физико-математических наук, ORCID, ResearcherID, m_ist@mail.ru

Елистратов Степан Алексеевич, младший научный сотрудник, лаборатория геофизической гидродинамики, Институт океанологии имени П.П. Ширшова РАН (РФ, 117218, Москва, Нахимовский пр., 65), стажер-исследователь, лаборатория цифрового моделирования технических систем, Институт системного программирования им. В.П. Иванникова РАН (РФ, 109004, Москва, ул. Александра Солженицына, 25); инженер-исследователь, отделение научно-исследовательской и проектной деятельности, Институт математики им. С.Л. Соболева СО РАН (РФ, 630090, Новосибирск, проспект Академика Коптюга, 4), ORCID, sa.elist-ratov@yandex.ru

Елизарова Татьяна Геннадьевна, главный научный сотрудник ИПМ им. М.В. Келдыша РАН (125047, Москва, Миусская пл., 4), доктор физико-математических наук, профессор, ORCID, ResearcherID, ScopusID, telizar@mail.ru

Тиняков Артём Дмитриевич, студент 5 курса кафедры проблем передачи информации и анализа данных, Московский физико-технический институт (национальный исследовательский университет) (РФ, 141701, Долгопрудный, Институтский переулок, 9), ORCID, tiniakov.ad@gmail.com

Заявленный вклад соавторов:

Все авторы сделали эквивалентный вклад в подготовку публикации.

Конфликт интересов

Авторы заявляют об отсутствии конфликта интересов.

Все авторы прочитали и одобрили окончательный вариант рукописи.