

Supercomputing Frontiers and Innovations

2023, Vol. 10, No. 1

Scope

- Future generation supercomputer architectures
- Exascale computing
- Parallel programming models, interfaces, languages, libraries, and tools
- Supercomputer applications and algorithms
- Novel approaches to computing targeted to solve intractable problems
- Convergence of high performance computing, machine learning and big data technologies
- Distributed operating systems and virtualization for highly scalable computing
- Management, administration, and monitoring of supercomputer systems
- Mass storage systems, protocols, and allocation
- Power consumption minimization for supercomputing systems
- Resilience, reliability, and fault tolerance for future generation highly parallel computing systems
- Scientific visualization in supercomputing environments
- Education in high performance computing and computational science

Editorial Board

Editors-in-Chief

- **Jack Dongarra**, University of Tennessee, Knoxville, USA
- **Vladimir Voevodin**, Moscow State University, Russia

Editorial Director

- **Leonid Sokolinsky**, South Ural State University, Chelyabinsk, Russia

Associate Editors

- **Pete Beckman**, Argonne National Laboratory, USA
- **Arndt Bode**, Leibniz Supercomputing Centre, Germany
- **Boris Chetverushkin**, Keldysh Institute of Applied Mathematics, RAS, Russia
- **Alok Choudhary**, Northwestern University, Evanston, USA
- **Alexei Khokhlov**, Moscow State University, Russia
- **Thomas Lippert**, Jülich Supercomputing Center, Germany

- **Satoshi Matsuoka**, Tokyo Institute of Technology, Japan
- **Mark Parsons**, EPCC, United Kingdom
- **Thomas Sterling**, CREST, Indiana University, USA
- **Mateo Valero**, Barcelona Supercomputing Center, Spain

Subject Area Editors

- **Artur Andrzejak**, Heidelberg University, Germany
- **Rosa M. Badia**, Barcelona Supercomputing Center, Spain
- **Franck Cappello**, Argonne National Laboratory, USA
- **Barbara Chapman**, University of Houston, USA
- **Yuefan Deng**, Stony Brook University, USA
- **Ian Foster**, Argonne National Laboratory and University of Chicago, USA
- **Geoffrey Fox**, Indiana University, USA
- **William Gropp**, University of Illinois at Urbana-Champaign, USA
- **Erik Hagersten**, Uppsala University, Sweden
- **Michael Heroux**, Sandia National Laboratories, USA
- **Torsten Hoefler**, Swiss Federal Institute of Technology, Switzerland
- **Yutaka Ishikawa**, AICS RIKEN, Japan
- **David Keyes**, King Abdullah University of Science and Technology, Saudi Arabia
- **William Kramer**, University of Illinois at Urbana-Champaign, USA
- **Jesus Labarta**, Barcelona Supercomputing Center, Spain
- **Alexey Lastovetsky**, University College Dublin, Ireland
- **Yutong Lu**, National University of Defense Technology, China
- **Bob Lucas**, University of Southern California, USA
- **Thomas Ludwig**, German Climate Computing Center, Germany
- **Daniel Mallmann**, Jülich Supercomputing Centre, Germany
- **Bernd Mohr**, Jülich Supercomputing Centre, Germany
- **Onur Mutlu**, Carnegie Mellon University, USA
- **Wolfgang Nagel**, TU Dresden ZIH, Germany
- **Alexander Nemukhin**, Moscow State University, Russia
- **Edward Seidel**, National Center for Supercomputing Applications, USA
- **John Shalf**, Lawrence Berkeley National Laboratory, USA
- **Rick Stevens**, Argonne National Laboratory, USA
- **Vladimir Sulimov**, Moscow State University, Russia
- **William Tang**, Princeton University, USA
- **Michela Taufer**, University of Delaware, USA
- **Andrei Tchernykh**, CICESE Research Center, Mexico
- **Alexander Tikhonravov**, Moscow State University, Russia
- **Eugene Tyrtshnikov**, Institute of Numerical Mathematics, RAS, Russia
- **Roman Wyrzykowski**, Czestochowa University of Technology, Poland
- **Mikhail Yakobovskiy**, Keldysh Institute of Applied Mathematics, RAS, Russia

Technical Editors

- **Andrey Goglachev**, South Ural State University, Chelyabinsk, Russia
- **Yana Kraeva**, South Ural State University, Chelyabinsk, Russia
- **Dmitry Nikitenko**, Moscow State University, Moscow, Russia
- **Mikhail Zymbler**, South Ural State University, Chelyabinsk, Russia

Contents

An Ant-colony Based Model for Load Balancing in Fog Environments S.L. Mirtaheeri, M. Azari, S. Greco, E. Arianian	4
Simulation of Isolated Propeller Noise Using Acoustic-Vortex Method M.A. Pogosyan, S.F. Timushev, P.A. Moshkov, A.A. Yakovlev	21
Saddle Point Method Interpretation of Transient Processes in Car Tires K.S. Kniazeva, Y. Saito, A.I. Korolkov, A.V. Shanin	31
Simulation of Polyatomic Gas Cloud Expansion under Pulsed Laser Ablation by Model Boltzmann Equation A.A. Frolova	46
Mathematical Modeling of Detonation Initiation in the Channel with a Profiled End Using Parallel Computations A.I. Lopato	52
Overview of SCM Coupler and Its Application for Constructing Climate Models V.S. Gradov, G.A. Platov	58



This issue is distributed under the terms of the Creative Commons Attribution-Non Commercial 3.0 License which permits non-commercial use, reproduction and distribution of the work without further permission provided the original work is properly cited.

An Ant-colony Based Model for Load Balancing in Fog Environments

*Seyedeh Leili Mirtaheri*¹, *Mahya Azari*², *Sergio Greco*³, *Ehsan Arianian*⁴

© The Authors 2023. This paper is published with open access at SuperFri.org

Delay-sensitive applications are becoming more and more in demand as a result of the development of information systems and the expansion of communication in cloud computing technologies. Some of these requests will be overlooked in cloud environments due to the communication delay between the processing center and the client's request. The 'fog-based computing paradigm', a novel processing model, can be added to cloud computing to help with the aforementioned issues. The performance of computing systems is always influenced by latency. In this paper, we focus on balancing the load on the fog nodes to lower the latency. It is also a crucial component of fog computing devices. It necessitates the use of load-balancing algorithms to select the optimal hosts, resulting in an even distribution of the load on the available resources. We provide a load-balancing approach based on the Ant-colony optimization algorithm's latency rate for responding to tasks. A random data set evaluation of this model reveals shorter response times than those of earlier strategies suggested in this field.

Keywords: fog environment, cloud computing, load balancing, Ant-colony.

Introduction

One of the biggest problems in the distributed computing environment is load balancing. Numerous requests made by thousands of users and customers necessitate the utilization of a lot of hardware and bandwidth. A load balancer assists in distributing the burden among the many nodes and making sure that none of them are overwhelmed. It also expedites the completion of the most recent work among the sources that use the phrase [38]. Distributed systems like fog and cloud computing need an effective load-balancer.

Delivery time also dramatically increases as the number of delay-sensitive requests and the computational load on fog nodes both rise. As a result, some nodes must assign jobs to nodes that are less busy. Fog computing systems, on the other hand, have several connected processors that work independently. Every processor has a distinct processing capacity and an initial load. To reduce processing time and CPU downtime, the load is spread among processors based on their processing speeds. Since load balancing is meant to boost performance, its absence in distributed systems is a major issue [15, 48].

Environments that are both homogenous and diverse present a problem for load balancing methods in fog computing. In a homogenous group, all fog nodes have the same capacity and characteristics. Because they are unable to make use of the heterogeneous nature of resources, these nodes rarely adapt to fog settings. Each fog node in a heterogeneous group has a unique capacity, set of processing capabilities, and set of attributes [11]. Depending on the demands of the jobs, the load balancer can choose several nodes. In our suggested paradigm, we presume that all fog nodes are heterogeneous and that each fog node has distinct capacities and hardware architecture.

¹Corresponding Authors, Department of Electrical and Computer Engineering, Faculty of Engineering, Kharazmi University, Tehran, Iran. Mirtaheri@khu.ac.ir

²Faculty of Mathematics and Computer Science, Kharazmi University, Tehran, Iran

³Department of Computer Engineering, Modeling, Electronics and Systems, University of Calabria, Italy

⁴Department of Information Technology, ICT Research Institute (ITRC), Tehran, Iran

The nature of the tasks has a big impact on how well load-balancing algorithms work. Therefore, the greater optimization would be possible, the more information about the tasks would be available at the time of decision-making. Having a general understanding of job execution times and the precise due date for each task increases the likelihood of obtaining the ideal load distribution. Having a set time for each activity to be completed is uncommon; yet, this is an ideal condition. For this reason, a variety of strategies have been taken into consideration for organizing and generating thoughts for the duration of task execution. The maximum execution time needed for each task is taken into account in our proposed approach during load balancing.

An optimization strategy must be used since selecting the optimum hosts (load distribution on resources) is an endless sort of problem with multiple viable solutions. In this study, we employ the Ant-colony algorithm and the meta-heuristic methodology. Glover introduced the term “meta-heuristics” in 1986 in order to help people solve difficulties by identifying solutions that are close to ideal [19]. The meta-heuristic algorithms use governing mechanisms that mimic certain methods found in nature, social behavior, physical laws, etc. to achieve complete exploration of the search space. Meta-heuristic algorithms begin with an initial collection of independent variables and evolve to find the objective function’s global minimum and maximum. We encountered the similar issue when balancing the system’s load, thus we used one of the well-known meta-heuristic algorithms, called Ant-colony, in our proposed model.

We start with balancing the workload between tasks and fog nodes as input for the issue in our suggested model. Depending on the nature of the problem, the list of inputs may change. The current study takes into account a list of tasks with due dates and set intervals for each request. Taking into account the distance to the fog node, each assignment must be completed within the allotted amount of time. You might categorize these lists as static or dynamic. Requests are posted online to the computing system in the dynamic list. In contrast, it is believed that we have a collection of tasks whose order is flexible in the static model. The current study makes the assumption that there are a number of delay-sensitive jobs that ought to be.

It is crucial to know how many fog nodes there are and which ones are capable of carrying out the necessary activities. The list of fog nodes is supplied in our study so that the capacity, traits, and processing power of each node can be noted. The solution to load balancing issues in a fog environment depends on the problem’s objective function. The goal of the issue can be to raise the quality-of-service metrics (QoS), which include optimum resource utilization and decreased throughput, costs, response times, latency, and energy usage. To assess how well the problem satisfies the established aim, the problem’s goal must be modeled. The goal of the current work is to decrease latency in order to speed up task reaction times.

In the following sections of the present paper, we first explained related work. The current definition of load balance, types of load balancing mechanisms, and reasons for its need, benefits, and criteria of load balance in fog environment. The Ant-colony algorithm will be explained according to the classification of previous models. The proposed model is introduced and discussed, and lastly, the research results will be presented.

1. Related Work

Using optimization models, the load balance issue in fog situations can be assessed. Using meta-heuristic models to offer close to optimal solutions for such issues in an acceptable amount of time is one method to problem solving. Large-scale, complicated problems are thought to be amenable to the use of meta-heuristic models, which fall into two categories:

- The first category: is based on individuals and modifies a candidate solution.
- The second category: is population-based and improves several solutions.

In addition, biological behaviors can be categorized based on process strategies in the classification of reproductive strategies or inspired by natural phenomena. Given those various mechanisms have been proposed for load balancing in a fog environment. Some of these mechanisms are based on meta-heuristic algorithms, such as the bee algorithm, particle swarm, hill-climbing, and Ant-colony algorithm. Ant-colony Optimization algorithm is one of the most popular optimization algorithms with the ability to adapt to the system environment and the flexibility to solve any optimization problems. The present research uses Ant-colony algorithms to balance the load in the fog computing environment [43].

Zahid et al. [57] discussed a Hill Climbing technique to manage the load on VMs with the aim of reducing the response time, processing time, and delay. In the proposed framework, four different regions are taken, with each region containing a fog with a cluster of buildings. Results from the simulation reveal that the proposed strategy performs satisfactorily.

Kamal et al. [25] addressed the issue of load balancing by employing a heuristic approach for solving a Constraint Satisfaction Problem (CSP). The algorithm first checks the assignment conflicts and then randomly assigns VMs to requests. The proposed approach is compared in terms of processing time, response time, and cost with Throttled and Round Robin algorithms. It helps in allocating optimal resources to requests, thereby providing better simulation results.

Naqvi et al. [35] has proposed a bio-inspired algorithm called Ant-colony optimization for load balancing. The Ant-colony optimization algorithm is the swarm-based genetic algorithm. It works on the mechanism of real ants using pheromones in order to explore their path. In the same way, the allocation path of the cloudlets is identified based on the minimal path cost in a probabilistic manner.

In the paper [29], a meta-heuristic scheduler Smart Ant-colony Optimization (SACO) task offloading algorithm inspired by nature is proposed to offload the IoT-sensor applications tasks in a fog environment. The proposed algorithm results are compared with Round Robin (RR), throttled scheduler algorithm, and two bio-inspired algorithms such as modified particle swarm optimization (MPSO) and Bee life algorithm (BLA). The numerical result shows the significant improvement in latency by the proposed Smart Ant-colony Optimization (SACO) algorithm in task offloading of IoT-sensor applications comparison to Round Robin (RR), throttled, and MPSO and BLA. The proposed technique reduces the task offloading time by 12.88, 6.98, 5.91, and 3.53% in comparison to Round Robin (RR), throttled, MPSO, and BLA.

In this study [40], utilizing a hybrid optimization algorithm, they introduced a novel energy-aware technique for load balance management in the fog-based VANET. A VANET network is made up of mobile nodes without any infrastructure. The mobility of nodes, limited energy reserves, lack of central management, and providing a guarantee for the quality of services are some of the challenges of this type of network compared to wired networks. The existing nodes of VANET networks are free to move in any direction. This study aimed to present an energy-aware model based on load balancing using the ACO algorithm. Algorithm implementation and routing models were fully described. Then, the effect of increasing the number of nodes on the amount of energy consumed by the fog-based VANET and the residual energy in the battery were investigated. Also, the number of residual nodes was increased by increasing the length of routing periods in the proposed model with the ant and ABC optimization algorithm.

The simulation results in the NS2 environment showed that with increasing the number of nodes, the amount of energy consumed by the fog-based VANET increases. Also, as the number of nodes increases, the amount of residual energy will decrease, making the proposed algorithm outperform the other two algorithms. Examining the amount of residual energy within the periods revealed that the amount of residual energy decreased with increasing periods. The results indicate the good performance of the proposed model compared to the other four models.

In this paper [5], the cost-aware Ant-colony optimization-based load balancing model is proposed to minimize the execution time, response time, and cost in a dynamic environment. This model enables to balance of the load across the virtual machines in the data center and evaluates the overall performance with various load-balancing models. As an average, the proposed model reduces carbon footprint by 45% as compared to existing models.

In this paper [20], they propose a multi-objective fog computing task scheduling algorithm based on an improved Ant-colony algorithm, which optimizes the Ant-colony algorithm to make it more suitable for the characteristics of the fog node, uses time and cost (TAC) to comprehensively consider the cost of the node, and introduce the critical factor in task allocation to improve the convergence speed of the algorithm. Different simulation experiments show that the efficiency of the improved Ant-colony algorithm is enhanced in processing time, cost, and load balance.

In this research [7], they have proposed a new algorithm called Ant-colony Optimization-based Light Weight Container (ACO-LWC) load balancing scheduling algorithm for scheduling various process requests. Initially, the task allocation was done in a round-robin fashion. The CPU usage and memory usage were maintained within a particular optimal range to achieve effective load balancing in this algorithm. Here, the load balancing scheduling was done based on Ant-colony Optimization. Performance analysis showed that the proposed ACO-LWC algorithm achieved better stability performance. Further, the TPS of the cluster for 50 applications was found to be 120, 231, and 1042 for the least connection, round-robin, and proposed ACO-LWC, respectively. Similarly, the response time for the least connection, round-robin, and the proposed scheme with 60 applications was identified as 5282 ms, 2865 ms, and 1593 ms. Furthermore, analysis shows that the overall run time is very low for the proposed scheme. There is around 2.08 times reduction in run time compared to the least connection and 1.607 times reduction in run time compared to the round-robin algorithm.

2. Proposed Model

In this section, we describe the process of using the Ant-colony algorithm in load distribution problem in our proposed model.

2.1. Ant-colony Algorithm

The Ant-colony optimization model is based on the actual movement of ants in nature. Ant-colony optimization algorithms are inspired by natural phenomena and biological behaviors and approach finding near-optimal solutions using a probabilistic model [16].

2.2. Problem Inputs

The problem inputs are mentioned in the following:

- List of fog nodes: This list includes the processing power of each node in terms of the number of cycles per second (cycle/s). Along with this list, the capacity of each fog node is also considered.
- To-do list: This list includes tasks that must be processed in fog nodes. This list is included: the minimum processing power required in terms of bits per second, the volume of tasks in megabytes, the distance of the task producer service from the surrounding fog nodes in meters, and the deadline of each task.
- Fixed parameters: In the Ant-colony optimization algorithm, several fixed parameters are used, the values of which are selected based on [58].

2.3. Latency Model

Latency is the time that each processor spends to process requests, i.e., from the moment the request is transmitted until the customer receives the request after the processing. For nodes in the fog environment that are close to the end devices, it is possible to collect data from sensors / IoT devices and process, analyze, and store them on the network edge. It should be done with minimal delay. The present study aimed to minimize the latency of each task on the fog nodes [28]. The completion time for each task includes the execution time of each task and its transmission time. The execution time of task i on node j is calculated using equation (1) [45]

$$TE_{ij}(t) = \frac{task_size_i \cdot task_proc_i}{fnode_proc_j}. \quad (1)$$

In this equation, $task_size_i$ indicates the size of task i , $task_proc_i$ indicates each task's capacity, and $fnode_proc_j$ indicates the capacity of the node j . At any given time (t), the transfer time, which is the time spent for sending task i to node j , is calculated using equation (2)

$$TR_{ij} = \frac{task_size_i}{task_dist_{ij} \cdot task_deadline_i}. \quad (2)$$

In this equation, $task_dist_{ij}$ indicates the distance of task i to node j , and $task_deadline_i$ indicates the specified time for each task. Lastly, the total delay is calculated using equation (3)

$$total_delay_{ij}(t) = TR_{ij}(t) + TE_{ij}(t). \quad (3)$$

2.4. Mapping of Load Balancing Parameters on the Ant-colony Parameters

As mentioned at the beginning of this section, the load balancing problem uses delay time to reduce the response time. In the case of load balancing in the target fog environment, the load balance between the fog nodes is set so that the objective function, which is the rate of delay, is met and minimized. It can be described as assigning n independent tasks to m fog nodes. Ants place a list of tasks for processing on the fog nodes. Each task has specific processing power, size, and distance from the fog nodes, and also, based on the specific time limit set for each task, the tasks can be executed on their appropriate fog node. For each fog node, a certain amount of processing power is considered in *cycle/s*. As mentioned before, the Ant-colony algorithm is an optimization algorithm, and each optimization algorithm has a certain number of iterations in which the problem is led toward the optimal solution.

2.5. Quantification of the Initial Solution

Every meta-heuristic optimization problem requires starting with an initial solution to explore the search space. The first step in the optimization algorithm is to create an initial solution to start the optimization process. In the Ant-colony algorithm, like other meta-heuristic algorithms, the initial solution must be applied as one of the inputs to the problem. Initial quantification of pheromones is also performed at this stage. The initial pheromone is given to prevent the ants from staying in place and not moving at the starting point. As a result, the initial solution is made using Initial quantification. In the present study, the initial number of pheromones is determined based on the random assignment of each task to the node, and their delay time is calculated. The pheromone left by the ants along the way indicates the tendency of the tasks toward the fog node.

2.6. Pheromone Factor

The number of pheromones in the different pathways drives the ants to reach the goal. In the ant pheromone cloning algorithm, pheromones indicate the desirability of the path. In our proposed model, the delay time factor is used in place of pheromones. The higher the pheromone, the shorter the latency in assigning task i to node j . Equation (4) introduces the pheromone factor

$$\tau_{ij} = \frac{1}{\varepsilon \cdot delay_{ij} + (1-\varepsilon) \cdot dist_{ij}}, \quad (4)$$

where ε is a constant parameter, and its value is determined [58].

2.7. Heuristic Information

Apart from pheromone pathways, another vital factor in the Ant-colony optimization algorithm is the selection of a suitable heuristic, which will be used in combination with pheromone information to construct solutions. At each stage of constructing a solution, the candidate selected for transfer depends on two factors: the pheromone factor and the heuristic factor. Since innovative information is calculated for all the movements in all the ants, it is, therefore, an essential factor that affects the performance of the Ant-colony algorithm. The heuristic information is expressed by the symbol η_{ij} . This information indicates the ant's degree of interest and attention to explore new paths leading to a better fog node. In our proposed model, the calculation of heuristic information is done based on equation (5)

$$\eta_{ij} = \frac{1}{delay_{ij}(s_0)}. \quad (5)$$

Problem's defaults: The following points are considered as defaults to solve the problem:

- Initial assignment of tasks to fog nodes is done.
- Each ant receives a list of tasks and information on the fog nodes from the controller.
- Fixed parameters are based on the values given in the paper [58].
- The algorithm leads toward a near-optimal solution using iterative processes.
- The purpose of the problem is to reduce the response time by considering the delay time parameter.

2.8. Building a Solution (Motion Transfer Law)

The movement of the ant k to place task i on node j is similar to the Pseudo-Random proportional law (equation (6))

$$j = \begin{cases} \max_{u \in \Omega_{k(i)}} \{[\tau_{iu}^\alpha] \cdot [\eta_{iu}^\beta]\}, & \text{if } q \leq q_0 \\ l, & \text{otherwise} \end{cases} \quad (6)$$

In this equation, α and β are weighting coefficients or compatible parameters. These parameters determine the amount of pheromone in the path and the heuristic functions used, respectively. It affects the probability of choosing a particular path. L is a random variable obtained from the probability distribution of equations (7). q is a uniform random number in the range $[0, 1]$ and q_0 is a parameter between zero and one, which controls the balance between discovering routes traveled so far and searching for unmet routes. If q is more minor than q_0 , this process is called exploitation, and we select the fog node in the set $\Omega_{k(i)}$ that has the highest value according to equation (6). If q is more significant than q_0 , the fog node in the set $\Omega_{k(i)}$ is randomly selected using the law of probability distribution p_{ij}^k ; i.e., the probability that the ant k puts task i on the fog node j . This process, called exploration, improves random search and reduces complexity. Exploitation helps ants quickly converge to a high-quality solution while, at the same time, exploration prevents stagnation through providing a more comprehensive search space

$$p_{ij}^k = \begin{cases} \frac{\tau_{ij}^\alpha \cdot \eta_{ij}^\beta}{\sum_{u \in \Omega_{k(i)}} \tau_{iu}^\alpha \cdot \eta_{iu}^\beta}, & \text{if } j \in \Omega_{k(i)} \\ 0, & \text{otherwise} \end{cases}, \quad (7)$$

where τ_{ij} indicates the amount of the pheromones at points (i, j) . $\Omega_{k(i)}$ is a set of fog nodes that can accept i functions. The remaining nodes of ant k are based on node i to construct an acceptable answer. If $\alpha = 0$, the algorithm works greedily so that the selection of the next node does not take into account the number of pheromones; therefore, the selection of the nearest route takes precedence. If $\beta = 0$, the algorithm considers only the number of pheromones, regardless of the path length.

2.9. Updating the Pheromones

In the Ant-colony optimization algorithm, the pheromone needs to be updated. The number of pheromones can increase or decrease. The amount of pheromone increases with sedimentation and decreases with evaporation. The deposition of new pheromones is based on the fact that pheromone pathways indicate the information in some solutions, and the movement in these reasonable solutions is built following the other ants' solutions. However, pheromone evaporation is a practical implementation of forgetting that prevents the algorithm from over-converging to non-optimal regions, leading to the search for new and desirable regions in the search space. In our proposed algorithm, the pheromone upgrade process consists of two stages: local pheromone upgrade and global pheromone upgrade when task i is placed on fog node j , the decrease in pheromone level between task i and fog node j is applied by the local upgrade given in equations (8)

$$\tau_{ij}(t) = (1 - \rho_L) \tau_{ij}(t - 1) + \rho_L \cdot \tau_{ij}^0. \quad (8)$$

In which τ_0 is the level of the initial pheromone and ρ_L is the local pheromone evaporation parameter ($0 < \rho_L < 1$). The global pheromone upgrade applies after the solution for all the ants for one repetition has been completed. Since all non-dominated solutions or the Pareto set are considered optimal or good solutions for optimization problems, we assume that all non-dominated solutions are the same and of high quality and that all dominated solutions must be eliminated. Therefore, the global update is applied to each solution s of the Pareto set using equation (9)

$$\tau_{ij}(t) = (1 - \rho_g) \tau_{ij}(t - 1) + \frac{\rho_g \cdot \lambda}{total_delay}(s_0). \quad (9)$$

In which:

$$\lambda = \frac{N_{ant}}{t - N_{iter.s} + 1}. \quad (10)$$

In equation (10), ρ_g ($0 < \rho_L < 1$) is the pheromone evaporation parameter in the global update, and non-dominated global solutions, in the form of Pareto sets, are stored in an external set.

In equation (10), N_{ant} shows the number of ants, and $N_{iter.s}$ shows the number of repetitions in the solution of s in the external set. λ is the coefficient of adaptation, which helps control the pheromone's information in an external set over time. The global pheromone upgrade aims to increase the ants' learning.

2.10. Pseudo-code

Implementation performed in the following psuedo-code is shown in below.

To determine the fog node (in addition to equation (6)), the probability of the fog node productivity is considered. It is in the range of 30 to 80%.

Based on the flowchart shown in Fig. 1, the fixed values (list of tasks and information on the fog nodes) are received as input for the problem. Then, to determine the initial amount of pheromone, random assignment of tasks is performed on the fog nodes, and their latency is calculated and stored as the initial pheromone. It then enters the repeating loop, and the solution of each ant is made. Each ant makes the solution by sorting the list of tasks at their disposal according to the deadlines in ascending order. Then, remove the first task from the beginning of the list and find the appropriate fog node according to equation (6). The ants also check the productivity of the fog node over and over again. The local pheromone is upgraded based on equations (8), and the ant stores its solution in the optimal solutions. After all the ants make the solution in one repetition, the set of solutions made by the ants are examined, the best ones are left in the list, and the global pheromone upgrade is done based on equations (9). These steps are repeated to ensure that the best solutions remain in the $s_{optimal}$ set. $s_{optimal}$ represents all assignments or assignments that have the shortest response time for tasks.

Since fog calculations are always under investigation and fog nodes have unpredictable behaviors, the real test platform is not yet available to most researchers, and they use simulation and modeling tools to evaluate their work. Since the results of cases such as real-time scheduling can be different from simulation, it can be concluded that implementing the mechanisms discussed in actual experiments is still challenging. Any proposed model to prove its claim; requires the presentation of acceptable results and comparison of the model with other previous models. A data set appropriate to the problem must first be selected and then simulated or modeled inappropriate software to evaluate a model. By using the design proposed in this work, the load

Algorithm 1 The proposed model's algorithm

Input: *fog_node_list*, *job_list*[*size*, *deadline*, *process*]
Output: *pare_to_set*, *s_optimal*
 set values of *n_ant*, *n_itr*, *n_itrmax*, *pl*, *pg*, *q_O*, α , and β
 initialize primary pheromone based on random allocation ▷ initialization
while *n_itr* < *n_itrmax* **do** ▷ iterative loop
 for *ant_k* **do** ▷ construct solution
 while *job_list* is not empty **do**
 sort *job_list* based on the *deadline*
 get *j_i* from *job_list*
 choose *fog_node* with eq. 6-3 to place *j_i*
 flag_underload \leftarrow 0
 flag_overload \leftarrow 0
 detect()
 if *fog_node*[*utilization*] > 0.8 **then**
 flag_overload \leftarrow 1
 else if *fog_node*[*utilization*] < 0.3 **then**
 flag_underload \leftarrow 1
 end if
 remove *j_i* from *job_list*
 apply local pheromone update based one eq 8-3
 end while
 add *ant_k* solution to *s_optimal*
end for
for *s_i* in *s_optimal* **do** ▷ construct optimal solution
 apply local pheomone update based one eq 9-3
end for
n_itr \leftarrow *n_itr* + 1
end while

balance between the fog nodes can be created so that in addition to reducing the delay, the response time of the tasks is also reduced.

3. Evaluation

The load balancing problem in the fog environment is implemented based on the Ant-colony algorithm in Python. The reason for choosing Python for this implementation is to provide the libraries needed for implementation. Since Python is a dynamic language, it has advantages such as brevity in coding, no function limitation, and the ability to replace functions during the performance. Due to its explicit nature, Python, compared to other languages such as Java or C++, helps improve productivity and reduce development time.

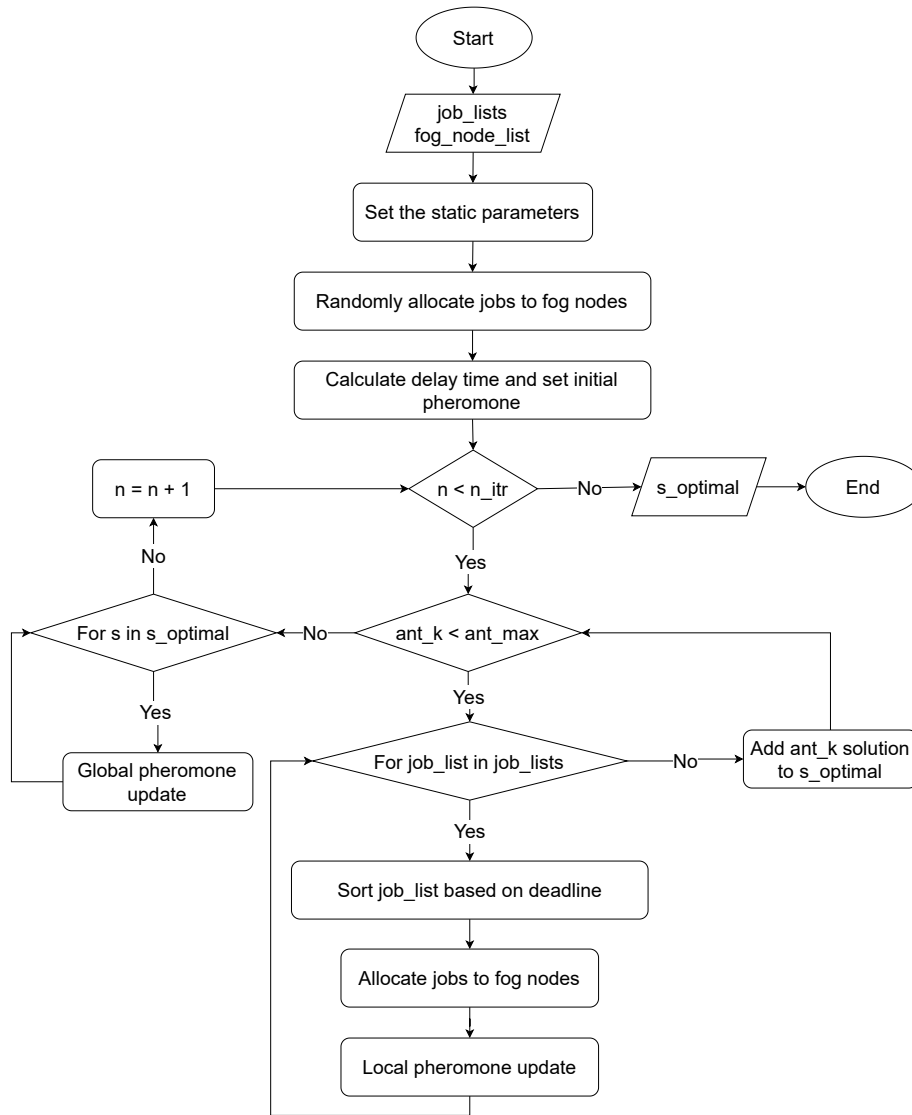


Figure 1. The proposed model's flowchart

3.1. Evaluation Data Set

Evaluation data sets can be examined in two categories: real and artificial data sets. Real data sets are data published by reputable companies used for evaluation. Artificial data sets are produced by different statistical models and are determined based on their parameters. In our proposed model, we use a random data set. In this set, the volume of tasks and their deadlines and processing power are generated and stored randomly. Also, to ensure the obtained results, 200 evaluation tests have been performed, and the announced results are the average of these 200 evaluations.

3.2. Evaluation Criteria

The objective function for our proposed model can be extended to a multi-objective function. Therefore, in this study, we focused on one objective. Our proposed model considers response time as the primary evaluation criterion [1] and shows that making a decision based on delay time can also reduce response time.

3.3. Evaluating the Proposed Model

Our proposed algorithm is implemented with Python programming language and evaluated on three servers. These servers had 24-core physical specifications with an E5-2650 v3 processor and a frequency of 2.3 GHz. Several virtual machines, including an 8-core virtual machine with 32 GB of random memory as the fog control node, 150 dual-core virtual machines with 4 GB of random memory as fog nodes, and 400 Mbps bandwidth are defined on these servers. Table 1 lists the features related to the tasks.

Table 1. Features related to the tasks Description

Description	Value	Parameter
The size of each task	10–150 MB	Task-size
Task processing power	500–2500 Cycle/s	Task-list

3.4. Evaluation Results

We evaluated our proposed model based on the algorithms given in the paper [45]. The criterion measured was the task response time. The following Tab. 2 to Tab. 5 shows the response times for evaluated models. This experiment was repeated with the number of tasks 400, 600, 1000.

Table 2. Table related to 400 tasks

	The Proposed model	The model in [45]
Average response time per task (milliseconds)	12	20
Run time (hours)	1	1.15

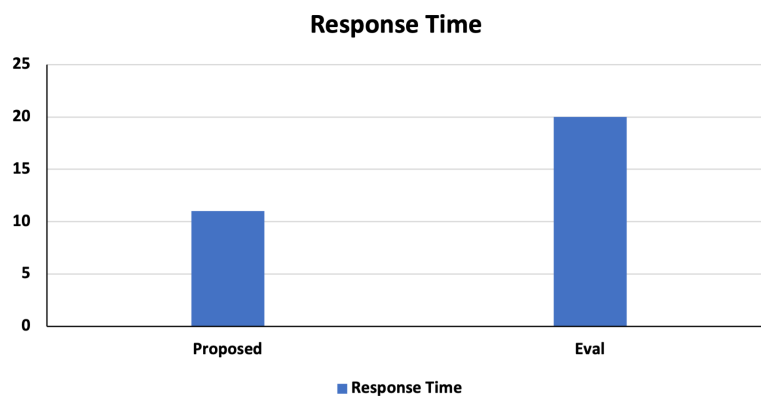
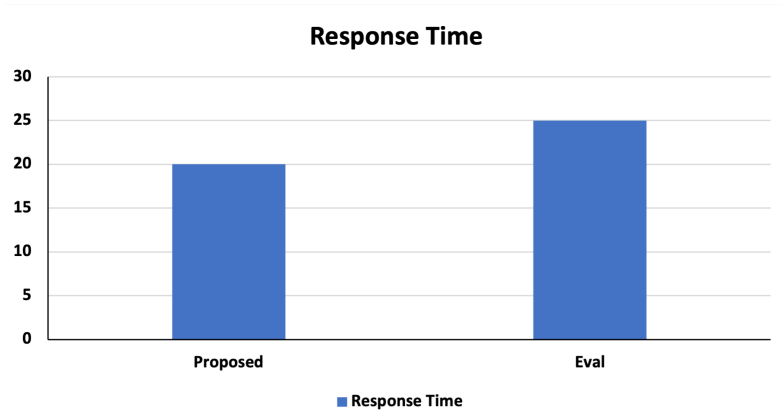


Figure 2. Response time for 400 tasks

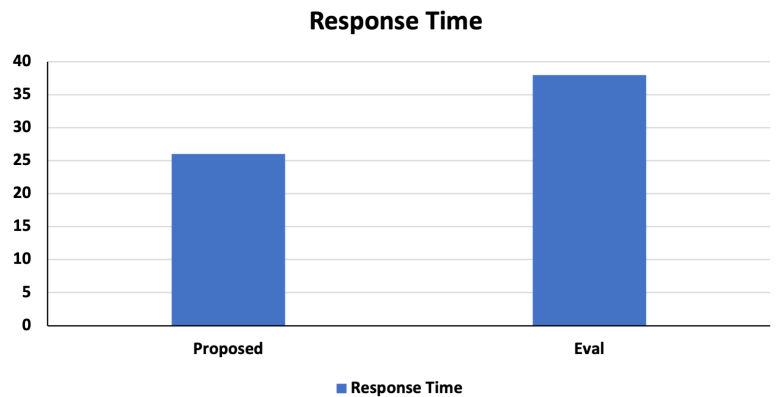
It can be seen in the above tables and the resulting graphs that, as the number of tasks increases, the execution time to find the load balance between the fog nodes increases. Consequently, the average response time of tasks also increases as the load density on the fog nodes increases, and it becomes more challenging to find the correct node for each task over time. Table 5 shows the average latency based on the number of tasks.

Table 3. Table related to 600 tasks

	The Proposed Model	The model in [45]
Average response time per task (milliseconds)	20	25
Run time (hours)	0.06	0.1

**Figure 3.** Response time for 600 tasks**Table 4.** Table related to 1000 tasks

	The Proposed model	The model in [45]
Average response time per task (milliseconds)	27	38
Run time (hours)	0.13	6

**Figure 4.** Response time for 1000 tasks

Conclusion

Due to the Internet of Things explosive expansion, processing speed improvements, and mobile technology, fog-based processing is crucial. This effective processing approach meets the requirements of delay-sensitive applications and eases the network bandwidth bottleneck. One of the key problems with this architecture is load balance, which prevents overloading (and hence lowers the system's efficiency). In order to effectively utilize the resources, decrease response time and latency, and so boost system efficiency, it is necessary to balance the load

Table 5. Average latency based on the number of tasks

Tasks	400	600	1000
Average latency of each task (milliseconds)	9.8	15.3	20.2

amongst the fog nodes using efficient algorithms. In the current work, we demonstrated how the Ant-colony optimization algorithm, along with the delay criterion and the addition of the load criterion for the fog environment, may be utilized to balance the load on the fog nodes and speed up task completion. The load on the fog nodes must be balanced in order to minimize the latency, but identifying the ideal host (load distribution on resources) for loads is an NP-hard task. Because it takes a while to obtain the precise solution, optimization procedures are employed. The Ant-colony algorithm makes a choice based on the latency criterion and the inclusion of the load criterion in the fog computing environment because meta-heuristic models offer better opportunities to create a compromise between the complexity of the search process and optimization. The optimization parameters and mapping load distribution parameters are then thoroughly described. Finally, we designed the load balance between the fog nodes in such a way that reaction time was also lowered in addition to lowering latency using the simulation and comparison with the algorithm described in the paper [45]. We also saw that when the number of tasks increased, it took longer to find a load balance between the fog nodes, which increased the tasks' average response times (due to the increase in density and volume of load on the fog nodes and therefore finding the correct node for each task getting more difficult).

This paper is distributed under the terms of the Creative Commons Attribution-Non Commercial 3.0 License which permits non-commercial use, reproduction and distribution of the work without further permission provided the original work is properly cited.

References

1. Abbasi, S.H., Javaid, N., Ashraf, M.H., *et al.*: Load stabilizing in fog computing environment using load balancing algorithm. In: Advances on Broadband and Wireless Computing, Communication and Applications. pp. 737–750. Springer (2018). https://doi.org/10.1007/978-3-030-02613-4_66
2. Aghdashi, A., Mirtaheri, S.L.: Novel dynamic load balancing algorithm for cloud-based big data analytics. *The Journal of Supercomputing* 78(3), 4131–4156 (2022). <https://doi.org/10.1007/s11227-021-04024-8>
3. Ahmed, A., Arkian, H., Battulga, D., *et al.*: Fog computing applications: Taxonomy and requirements. *CoRR abs/1907.11621* (2019), <https://arxiv.org/abs/1907.11621>
4. Al-Qamash, A., Soliman, I., Abulibdeh, R., Saleh, M.: Cloud, fog, and edge computing: A software engineering perspective. In: 2018 International Conference on Computer and Applications (ICCA). pp. 276–284. IEEE (2018). <https://doi.org/10.1109/COMAPP.2018.8460443>
5. Alagarsamy, M., Sundarji, A., Arunachalapandi, A., Kalyanasundaram, K.: Cost-aware ant colony optimization based model for load balancing in cloud computing. *The International Arab Journal of Information Technology* 18(5), 719–729 (2021)

6. Alboaneen, D.A., Tianfield, H., Zhang, Y.: Metaheuristic approaches to virtual machine placement in cloud computing: a review. In: 2016 15th International Symposium on Parallel and Distributed Computing (ISPDC). pp. 214–221. IEEE (2016). <https://doi.org/10.1109/ISPDC.2016.37>
7. Aruna, K., Pradeep, G.: Ant Colony Optimization-based Light Weight Container (ACO-LWC) Algorithm for Efficient Load Balancing. *Intelligent Automation & Soft Computing* 34(1), 205–219 (2022). <https://doi.org/10.32604/iasc.2022.024317>
8. Atlam, H.F., Walters, R.J., Wills, G.B.: Fog computing and the Internet of Things: A review. *Big Data Cogn. Comput.* 2(2), 10 (2018). <https://doi.org/10.3390/bdcc2020010>
9. Baek, J.y., Kaddoum, G., Garg, S., Kaur, K., Gravel, V.: Managing fog networks using reinforcement learning based load balancing algorithm. In: 2019 IEEE Wireless Communications and Networking Conference (WCNC). pp. 1–7. IEEE (2019). <https://doi.org/10.1109/WCNC.2019.8885745>
10. Beulah Soundarabai, P., Thriveni, J., Venugopal, K., Patnaik, L.: Comparative study on load balancing techniques in distributed systems. *International Journal of Information Technology and Knowledge Management* 6(1), 53–60 (2012)
11. Beraldi, R., Canali, C., Lancellotti, R., Mattia, G.P.: Distributed load balancing for heterogeneous fog computing infrastructures in smart cities. *Pervasive and Mobile Computing* 67, 101221 (2020). <https://doi.org/10.1016/j.pmcj.2020.101221>
12. Bonomi, F., Milito, R., Zhu, J., Addepalli, S.: Fog Computing and Its Role in the Internet of Things. In: Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing. pp. 13–16. ACM, New York, NY, USA (2012). <https://doi.org/10.1145/2342509.2342513>
13. Bukhsh, R., Javaid, N., Ali Khan, Z., *et al.*: Towards fast response, reduced processing and balanced load in fog-based data-driven smart grid. *Energies* 11(12), 3345 (2018). <https://doi.org/10.3390/en11123345>
14. Cao, K., Liu, Y., Meng, G., Sun, Q.: An overview on edge computing research. *IEEE Access* 8, 85714–85728 (2020). <https://doi.org/10.1109/ACCESS.2020.2991734>
15. Chandak, A., Ray, N.K.: A review of load balancing in fog computing. In: International Conference on Information Technology (ICIT). pp. 460–465. IEEE (2019). <https://doi.org/10.1109/ICIT48102.2019.00087>
16. Dorigo, M., Stützle, T.: The ant colony optimization metaheuristic: Algorithms, applications, and advances. In: Handbook of Metaheuristics. pp. 250–285. Springer (2003). https://doi.org/10.1007/0-306-48056-5_9
17. Fan, Q., Ansari, N.: Towards workload balancing in fog computing empowered IoT. *IEEE Transactions on Network Science and Engineering* 7(1), 253–262 (2018). <https://doi.org/10.1109/TNSE.2018.2852762>
18. Ghomi, E.J., Rahmani, A.M., Qader, N.N.: Load-balancing algorithms in cloud computing: A survey. *Journal of Network and Computer Applications* 88, 50–71 (2017). <https://doi.org/10.1016/j.jnca.2017.04.007>

19. Glover, F.: Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research* 13(5), 533–549 (1986). [https://doi.org/10.1016/0305-0548\(86\)90048-1](https://doi.org/10.1016/0305-0548(86)90048-1), applications of Integer Programming
20. Gu, J., Mo, J., Li, B., Zhang, Y., Wang, W.: A multi-objective fog computing task scheduling strategy based on ant colony algorithm. In: 2021 IEEE 4th International Conference on Information Systems and Computer Aided Education (ICISCAE). pp. 12–16. IEEE (2021). <https://doi.org/10.1109/ICISCAE52414.2021.9590674>
21. Hamrioui, S., Lorenz, P.: Load balancing algorithm for efficient and reliable IoT communications within E-health environment. In: GLOBECOM 2017-2017 IEEE Global Communications Conference. pp. 1–6. IEEE (2017). <https://doi.org/10.1109/GLOCOM.2017.8254435>
22. He, X., Ren, Z., Shi, C., Fang, J.: A novel load balancing strategy of software-defined cloud/fog networking in the Internet of Vehicles. *China Communications* 13(Supplement2), 140–149 (2016). <https://doi.org/10.1109/CC.2016.7833468>
23. Ivanisenko, I.N., Radivilova, T.A.: Survey of major load balancing algorithms in distributed system. In: 2015 Information Technologies in Innovation Business Conference (ITIB). pp. 89–92. IEEE (2015). <https://doi.org/10.1109/ITIB.2015.7355061>
24. Jijin, J., Seet, B.C., Chong, P.H.J., Jarrah, H.: Service load balancing in fog-based 5G radio access networks. In: 2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC). pp. 1–5. IEEE (2017). <https://doi.org/10.1109/PIMRC.2017.8292300>
25. Kamal, M.B., Javaid, N., Naqvi, S.A.A., *et al.*: Heuristic min-conflicts optimizing technique for load balancing on fog computing. In: *Advances in Intelligent Networking and Collaborative Systems*. pp. 207–219. Springer (2018). https://doi.org/10.1007/978-3-319-98557-2_19
26. Kaur, S., Kaur, G.: A review of load balancing strategies for distributed systems. *International Journal of Computer Applications* 121(18), 45–47 (2015). <https://doi.org/10.5120/21644-4985>
27. Khaneghah, E.M., Mirtaheri, S.L., Grandinetti, L., *et al.*: A dynamic replication mechanism to reduce response-time of I/O operations in high performance computing clusters. In: 2013 International Conference on Social Computing. pp. 738–743. IEEE (2013). <https://doi.org/10.1109/SocialCom.2013.110>
28. Khattak, H.A., Arshad, H., Ahmed, G., *et al.*: Utilization and load balancing in fog servers for health applications. *EURASIP Journal on Wireless Communications and Networking* 2019(1), 1–12 (2019). <https://doi.org/10.1186/s13638-019-1395-3>
29. Kishor, A., Chakarbarty, C.: Task offloading in fog computing for using smart ant colony optimization. *Wireless Personal Communications* 127, 1–22 (2021). <https://doi.org/10.1007/s11277-021-08714-7>
30. Kumar, A., Pandey, S., Prakash, V.: A survey: Load balancing algorithm in cloud computing. In: *Proceedings of 2nd International Conference on Advanced Computing and Software Engineering (ICACSE)* (2019). <https://doi.org/10.2139/ssrn.3350328>
31. Mahmud, R., Kotagiri, R., Buyya, R.: Fog computing: A taxonomy, survey and future directions. *Internet of Everything* pp. 103–130 (2018). https://doi.org/10.1007/978-981-10-5861-5_5

32. Manju, A., Sumathy, S.: Efficient load balancing algorithm for task preprocessing in fog computing environment. In: Smart Intelligent Computing and Applications. pp. 291–298. Springer (2019). https://doi.org/10.1007/978-981-13-1927-3_31
33. Mirtaheri, S.L., Fatemi, S.A., Grandinetti, L.: Adaptive load balancing dashboard in dynamic distributed systems. Supercomputing Frontiers and Innovations 4(4), 34–49 (2017). <https://doi.org/10.14529/jsfi170403>
34. Mirtaheri, S.L., Grandinetti, L.: Optimized load balancing in high-performance computing for big data analytics. Concurrency and Computation: Practice and Experience 33(16), e6265 (2021). <https://doi.org/10.1002/cpe.6265>
35. Naqvi, S.A.A., Javaid, N., Butt, H., *et al.*: Metaheuristic optimization technique for load balancing in cloud-fog environment integrated with smart grid. In: Advances in Network-Based Information Systems. pp. 700–711. Springer (2019). https://doi.org/10.1007/978-3-319-98530-5_61
36. Neto, E.C.P., Callou, G., Aires, F.: An algorithm to optimise the load distribution of fog environments. In: 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC). pp. 1292–1297. IEEE (2017). <https://doi.org/10.1109/SMC.2017.8122791>
37. Ningning, S., Chao, G., Xingshuo, A., Qiang, Z.: Fog computing dynamic load balancing mechanism based on graph repartitioning. China Communications 13(3), 156–164 (2016). <https://doi.org/10.1109/CC.2016.7445510>
38. Patel, N., Chauhan, S.: A survey on load balancing and scheduling in cloud computing. International Journal for Innovative Research in Science and Technology 1(7), 185–189 (2015)
39. Puthal, D., Obaidat, M.S., Nanda, P., *et al.*: Secure and sustainable load balancing of edge data centers in fog computing. IEEE Communications Magazine 56(5), 60–65 (2018). <https://doi.org/10.1109/MCOM.2018.1700795>
40. Qun, R., Arefzadeh, S.M.: A new energy-aware method for load balance managing in the fog-based vehicular ad hoc networks (VANET) using a hybrid optimization algorithm. IET Communications 15(13), 1665–1676 (2021). <https://doi.org/10.1049/cmu2.12179>
41. Rathod, D., Chowdhary, G.: Load balancing of fog computing centers: minimizing response time of high priority requests. International Journal of Innovative Technology and Exploring Engineering 8(11), 2713–2716 (2019)
42. Shah, J.M., Kotecha, K., Pandya, S., *et al.*: Load balancing in cloud computing: Methodological survey on different types of algorithm. In: 2017 International Conference on Trends in Electronics and Informatics (ICEI). pp. 100–107. IEEE (2017). <https://doi.org/10.1109/ICOEI.2017.8300865>
43. Shams, P., Mirtaheri, S.L., Shahbazian, R., Arianyan, E.: Improving IoT resource management using fog calculations and ant lion optimization algorithm. Journal of Information and Communication Technology 55(56), 1–19 (2023)
44. Sharma, S., Singh, S., Sharma, M.: Performance analysis of load balancing algorithms. World academy of science, engineering and technology 38(3), 269–272 (2008)
45. Sharma, S., Saini, H.: Efficient solution for load balancing in fog computing utilizing artificial bee colony. International Journal of Ambient Computing and Intelligence (IJACI) 10(4), 60–77 (2019). <https://doi.org/10.4018/IJACI.2019100104>

46. Sharma, S., Saini, H.: A novel four-tier architecture for delay aware scheduling and load balancing in fog environment. *Sustainable Computing: Informatics and Systems* 24, 100355 (2019). <https://doi.org/10.1016/j.suscom.2019.100355>
47. Sthapit, S., Hopgood, J.R., Thompson, J.: Distributed computational load balancing for real-time applications. In: 2017 25th European Signal Processing Conference (EUSIPCO). pp. 1385–1189. IEEE (2017). <https://doi.org/10.23919/EUSIPCO.2017.8081436>
48. Talaat, F.M., Ali, S.H., Saleh, A.I., Ali, H.A.: Effective load balancing strategy (ELBS) for real-time fog computing environment using fuzzy and probabilistic neural networks. *Journal of Network and Systems Management* 27(4), 883–929 (2019). <https://doi.org/10.1007/s10922-019-09490-3>
49. Téllez, N., Jimeno, M., Salazar, A., Nino-Ruiz, E.: A tabu search method for load balancing in fog computing. *Int. J. Artif. Intell* 16(2), 1–30 (2018)
50. Verma, M., Bhardwaj, N., Yadav, A.K.: Real time efficient scheduling algorithm for load balancing in fog computing environment. *Int. J. Inf. Technol. Comput. Sci* 8(4), 1–10 (2016). <https://doi.org/10.5815/ijitcs.2016.04.01>
51. Verma, M., Yadav, N.B.A.K.: An architecture for load balancing techniques for Fog computing environment. *International Journal of Computer Science and Communication* 8(2), 43–49 (2015)
52. Verma, S., Yadav, A.K., Motwani, D., *et al.*: An efficient data replication and load balancing technique for fog computing environment. In: 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom). pp. 2888–2895. IEEE (2016)
53. Wan, J., Chen, B., Wang, S., *et al.*: Fog computing for energy-aware load balancing and scheduling in smart factory. *IEEE Transactions on Industrial Informatics* 14(10), 4548–4556 (2018). <https://doi.org/10.1109/TII.2018.2818932>
54. Wang, J., Li, D.: Task scheduling based on a hybrid heuristic algorithm for smart production line with fog computing. *Sensors* 19(5), 1023 (2019). <https://doi.org/10.3390/s19051023>
55. Xu, X., Fu, S., Cai, Q., *et al.*: Dynamic resource allocation for load balancing in fog environment. *Wireless Communications and Mobile Computing* 2018 (2018). <https://doi.org/10.1155/2018/6421607>
56. Yi, S., Hao, Z., Qin, Z., Li, Q.: Fog computing: Platform and applications. In: 2015 Third IEEE workshop on hot topics in web systems and technologies (HotWeb). pp. 73–78. IEEE (2015). <https://doi.org/10.1109/HotWeb.2015.22>
57. Zahid, M., Javaid, N., Ansar, K., *et al.*: Hill climbing load balancing algorithm on fog computing. In: *Advances on P2P, Parallel, Grid, Cloud and Internet Computing*. pp. 238–251. Springer (2018). https://doi.org/10.1007/978-3-030-02607-3_22
58. Zhao, H., Wang, J., Liu, F., *et al.*: Power-aware and performance-guaranteed virtual machine placement in the cloud. *IEEE Transactions on Parallel and Distributed Systems* 29(6), 1385–1400 (2018). <https://doi.org/10.1109/TPDS.2018.2794369>

Simulation of Isolated Propeller Noise Using Acoustic-Vortex Method

*Mikhail A. Pogosyan*¹, *Sergey F. Timushev*¹, *Petr A. Moshkov*¹,
*Alexey A. Yakovlev*¹

© The Authors 2023. This paper is published with open access at SuperFri.org

The widespread development of unmanned aerial vehicles and light propeller-driven aircraft poses the task of reducing the community noise of such vehicles. To solve this problem, tools are needed to calculate the noise of such devices. The paper presents the results of the numerical simulation of the noise of an AV-2 propeller mounted on an AN-2 light propeller-driven aircraft. The authors use the acoustic-vortex method to solve the problem of aeroacoustic modeling of propeller noise in the presence of an incoming flow. The paper shows a good agreement of computed data with the in-flight experiment results and the calculation by the semi-empirical method. For the flight mode with an airspeed of 180 km/h, the deviation of the numerical simulation results from the experimental data does not exceed 2 dB.

Keywords: propeller noise, aeroacoustics, numerical simulation.

Introduction

The problem of numerical simulation of propeller noise is currently relevant due to the following factors:

- the vast development of the subject of uncrewed aerial vehicles (UAVs) [1–3];
- development of advanced configurations of aircraft with a distributed power plant [4, 5];
- development of open-rotor power plants and their integration into aircraft configuration.

For all the types of aircraft described above, the problem of controlling low noise levels to ensure certification and competitiveness is relevant. Several types of modern and promising aircraft configurations with propellers are shown in Fig. 1. Aeroacoustic effects may appear in the presented configurations, such as the noise of the blade-turbulent wake interaction [6, 7], the noise of the vortex blade interaction, the scattering of the power plant noise on the airframe elements [8, 9] and others. These effects must be considered when modeling aircraft noise and when implementing noise reduction technologies [10–12].

The work aims to verify the acoustic-vortex method in modeling the noise of an isolated propeller in the presence of an incoming flow (in-flight conditions). Preliminary verification was previously performed based on the results of static tests of the propeller-driven power plant, and the results are presented in [13].

The paper assumes that in the studied layout of AV-2 propeller on AN-2 aircraft, the propeller in flight conditions can be taken isolated.

The article is organized as follows. Section 1 presents object of study and test procedure for verification. Section 2 describes the numerical modeling method, presents the main equations, the calculated area and grid. Section 3 presents the results of numerical simulation of the aerodynamic and acoustic characteristics of the propeller. Section 4 presents a comparison of the results of numerical modeling with experimental data, as well as the results of calculation using a semi-empirical model.

¹Moscow Aviation Institute (National Research University), Moscow, Russia

1. Object of Study and Test Procedure for Verification

The object of the study is an AV-2 propeller mounted on AN-2 light propeller-driven aircraft (LPDA). The main parameters of the aircraft and its power plant are presented in Tab. 1. The general view of the aircraft and its propeller is shown in Fig. 2.



Figure 1. Modern and advanced configurations of aircraft with propellers

Table 1. The main parameters of AN-2 light propeller-driven aircraft

Engine	ASH-62IR
Available power, kW	735.43
Displacement, l	29.86
Compression ratio	6.4
Power-to-volume ratio, kW/l	24.62
Specific power, kW/kg	1.31
Gear ratio	0.6875
Propeller	AV-2
Number of blades	4
Propeller diameter, m	3.6

The results of aeroacoustic modeling were compared with experimental data on the AV-2 propeller noise to verify the acoustic-vortex method. Flight tests of the aircraft were conducted with level cruising flight conditions at an altitude of 100 m. Table 2 presents the flight modes considered in the framework of this work to verify the numerical solution.

The tests were conducted at the local aerodrome of the Moscow Aviation Institute (National Research University) with an underlying surface in the form of mown grass. During tests, the measuring microphone was located so that the sensitive membrane was parallel to the round



Figure 2. A general view of AN-2 aircraft (a) and its propeller AV-2 (b)

Table 2. Flight modes of AN-2 light propeller-driven aircraft for verifying the numerical solution

No.	Airspeed (V), km/h	Propeller speed (n), rpm
1	160	1100
2	180	1238

metal plate on the earth's surface and was placed 7 mm from it. The microphone was installed at a distance equal to three-fourths between the center and the edge of the plate along the radius and perpendicular to the flight path of LPDA. This microphone installation method is used in the certification tests of light propeller-driven aircraft. The ambient noise levels during the tests were lower than the noise levels of the power plant in the entire studied frequency range by at least 10–15 dB. The results of flight acoustic tests of AN-2 aircraft are presented in detail in [14].

2. Method of Numerical Simulation

2.1. Main Equations

Aeroacoustics modeling is based on decomposing compressible medium motion equations into the vortex (vortex motion of an incompressible medium) and acoustic modes [15, 16]. The velocity is the sum of the vortex flow velocity and the velocity of acoustic motion, which gives an acoustic-vortex equation for the fluctuations of the enthalpy (i) in the isentropic flow of the compressible medium:

$$\frac{1}{a^2} \frac{\partial^2 i}{\partial t^2} - \Delta i = \nabla \cdot \left(\nabla (1/2 U^2) - U \times (\nabla \times U) \right), \quad (1)$$

where a – mean sound speed, U – the velocity field of the vortex mode (pseudo-sound).

The equations of moments are used to model the vortex mode:

$$\frac{\partial \rho U}{\partial t} + \nabla \cdot (\rho U \otimes U) = -\nabla P + \nabla \cdot \left((\mu + \mu_t) \left(2\hat{S} - \frac{2}{3} (\nabla \cdot U) \hat{I} \right) \right) \quad (2)$$

and continuity

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \cdot U) = 0, \quad (3)$$

where ρ – air density, P – pressure in the vortex flow, μ – molecular coefficient of dynamic viscosity, μ_t – turbulent coefficient of dynamic viscosity, \hat{S} – strain rate tensor, \hat{I} – unit tensor.

The solution methodology discussed is based on the finite volume method. The finite volume method involves the integration of the equations of fluid motion and the transfer of scalar quantities in partial derivatives with respect to the volumes of computational cells – polyhedra. This ensures the conservatism of the mass, momentum, energy and other desired quantities in the computational domain. For the finite-difference solution of the equations of the vortex mode, the method of splitting by physical processes is used, which provides the second order of approximation accuracy in spatial variables and the first order in time. To solve the wave equation, an explicit method is used in time with the second order of approximation accuracy in space and time.

The acoustic-vortex method is implemented as a beta version of the single-processor software code FlowVision 2.5x. Currently, the development of a multiprocessor version of the acoustic-vortex module based on FlowVision 3.12 is being completed.

The k - ϵ turbulence model determines the turbulent viscosity with setting of solid wall boundary condition in the form of “wall functions” (numerical implementation of the logarithmic law for the velocity profile in the boundary layer).

2.2. Calculated Area and Grid

The calculation is carried out on a grid of the third level of adaptation with the number of calculation cells of 230000. The calculation time on the Intel(R) Core (TM) i5-7400 CPU 3.00 GHz processor for solving the vortex mode equations from zero initial conditions is 40 hours.

The calculated area is a cylinder with a diameter of 20 m and a height of 20 m (Fig. 3). The propeller locates in the center of the computation domain. The calculation is carried out in a fixed coordinate system with a simulation of the propeller rotation.

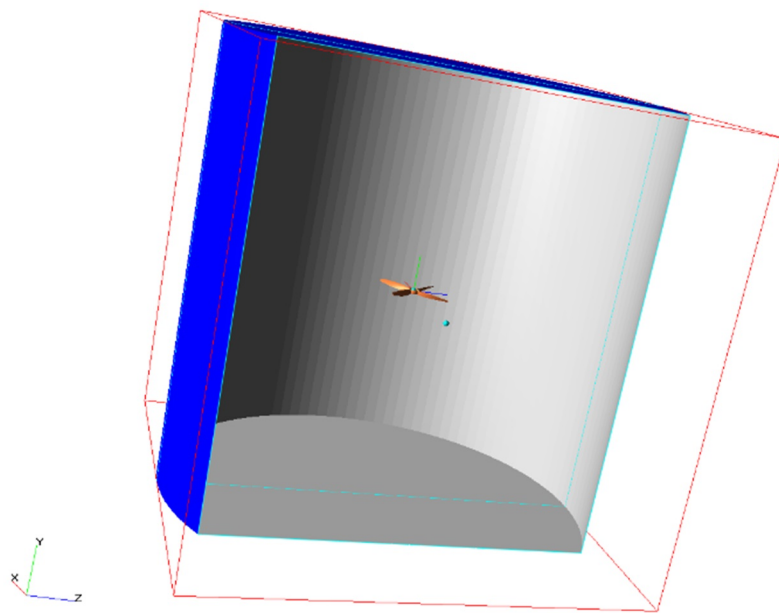


Figure 3. Computation region

In the computation study, an adapted grid of the third level is applied – near the propeller, the cells of the initial rectangular grid are divided into eight smaller cells – this procedure repeats three times. The computation grid in the vicinity of the propeller is shown in Fig. 4.

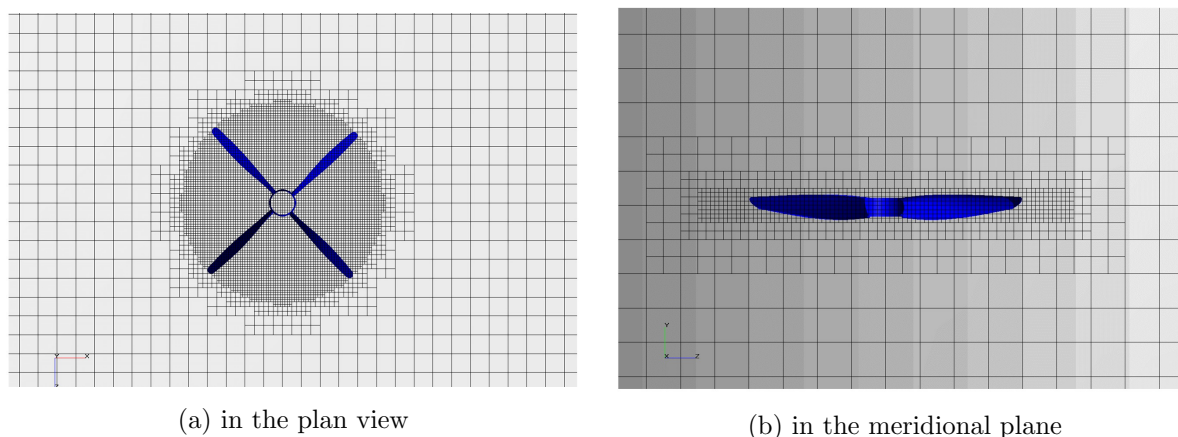


Figure 4. Computation grid

3. Results of Numerical Simulation

Numerical calculations were performed for two flight modes presented in Tab. 2. This section presents computational data on the airspeed of 180 km/h at a propeller speed of 1238 rpm.

3.1. Aerodynamic Results

The aerodynamic results are presented in Fig. 5 in the form of an instantaneous static pressure field (Pa) in the propeller rotation plane and the meridional plane.

One can see (Fig. 5a) that local pressure drop zones are formed when the propeller blades flow around. The propeller at work (Fig. 5b) throws off the flow and creates thrust, thus the aerodynamic pattern of the problem under consideration is realistic.

The results of the calculated evaluation of the thrust and power of the propeller for the flow conditions of the power plant considered in the work are presented in Tab. 3.

Table 3. Thrust and power of the propeller according to the results of numerical simulation

No.	Airspeed (V), km/h	Propeller speed (n), rpm	Thrust, N	Power, kW
1	160	1100	506.8	133.8
2	180	1238	315.8	139.3

3.2. Acoustic Results

The acoustic results in the sound pressure field (Pa) of the first BPF tone of propeller noise in the plane of rotation of the propeller and the meridional plane are shown in Fig. 6. The distribution of the amplitude (Pa) of the first BPF tone of propeller noise in the plane of rotation and the meridional plane is shown in Fig. 7.

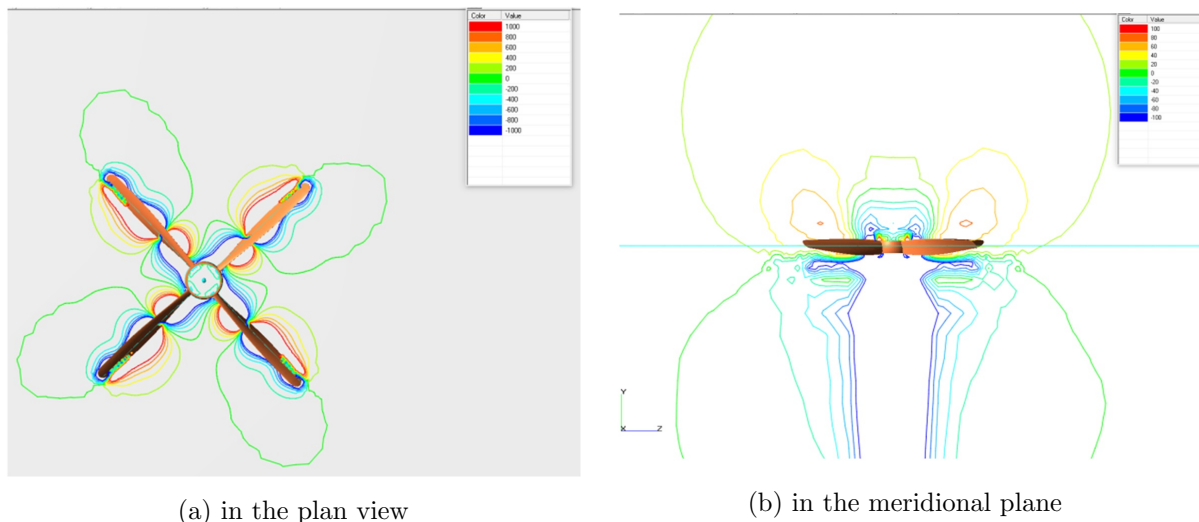


Figure 5. Instantaneous static pressure field

The presented results indicate the dipole nature of the noise emission of the propeller [17, 18]. The maximum noise levels are observed in the direction of the azimuth angle of 90, which is consistent with the directional pattern of the first BPF tone of the studied propeller obtained under static conditions when the maximum was observed in the direction of 105 [13]. There are four characteristic directions of radiation maximums located symmetrically in the plane of rotation.

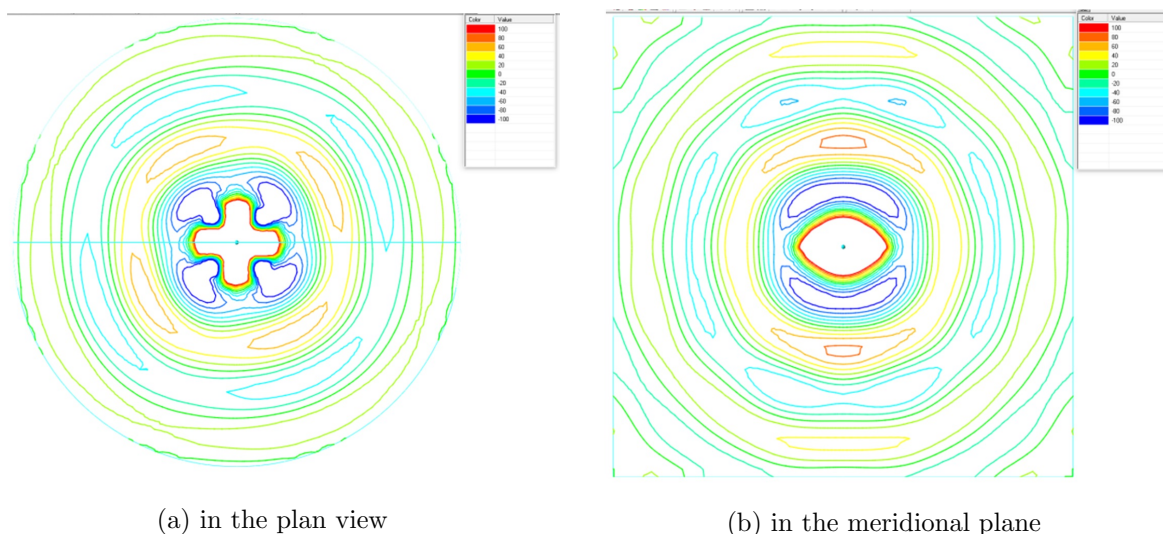


Figure 6. Sound pressure field (Pa) of the 1st BPF tone of the propeller noise (flight direction along the Y-axis)

The distribution of sound pressure of the 1st BPF tone at the boundary of the calculation domain is shown in Fig. 8. The sound pressure is distributed unevenly and symmetrically on the lateral surface, revealing four maximums. One can see that at the inlet and outlet of the cylindrical region, the sound pressure is distributed evenly and decreases to the periphery.

4. Comparison of Calculated and Experimental Data

The comparison of calculated and experimental data is based on comparing the sound power levels of the first three tones of the propeller noise. The evaluation results are shown in Fig. 9

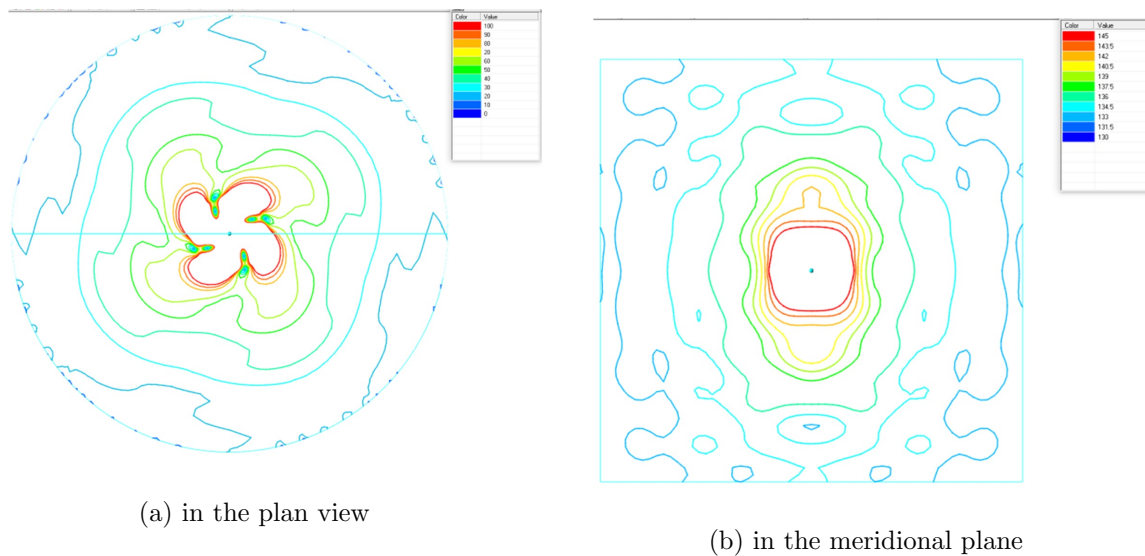


Figure 7. Amplitude (Pa) of the 1st BPF tone of the propeller noise (flight direction along the Y-axis)

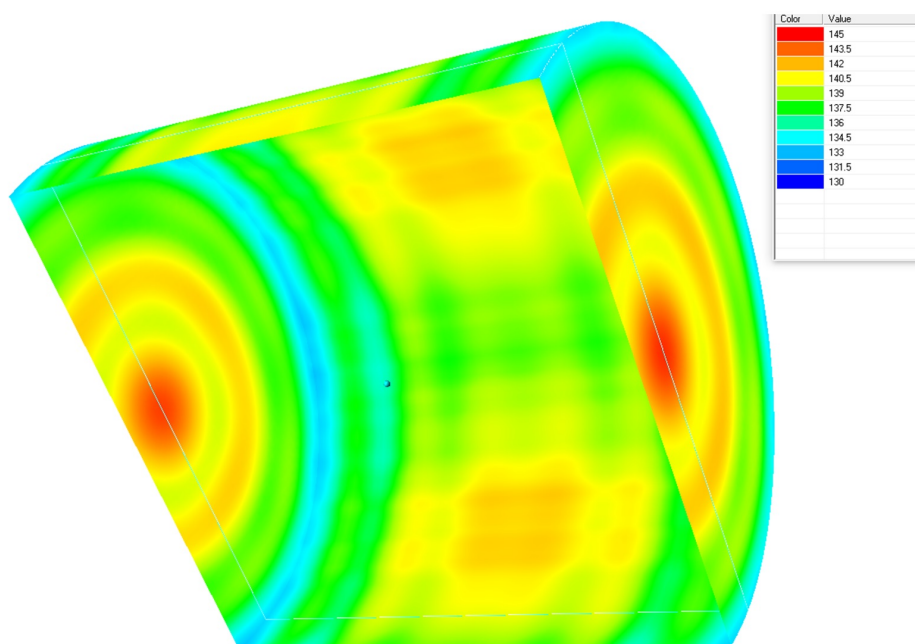


Figure 8. Sound pressure distribution (dB) of the first tone at the boundary of the computational region

for two flight modes. Additionally, the graph shows the results of the calculation performed by the semi-empirical method [19, 20].

For a flight mode with an airspeed of 160 km/h, a good agreement was obtained between the results of numerical simulation and experimental data for the first two tones of propeller noise. For the flight mode with a speed of 180 km/h, the deviation from the experimental data does not exceed 2 dB.

Note that the semiempirical method provides an acceptable calculation accuracy for solving engineering problems. For the two presented calculated cases, the deviation of the measured and calculated sound power levels of the above three tones does not exceed 2 dB.

According to the authors of the work, in this case, a deviation within 2 dB is a good result of modeling the sound field of a propeller, since the measurement error of the sound pressure level of the measuring system used is 0.7 dB. A deviation of more than 3 dB when assessing the sound power level leads to an overestimation or underestimation of the sound power by 2 times.

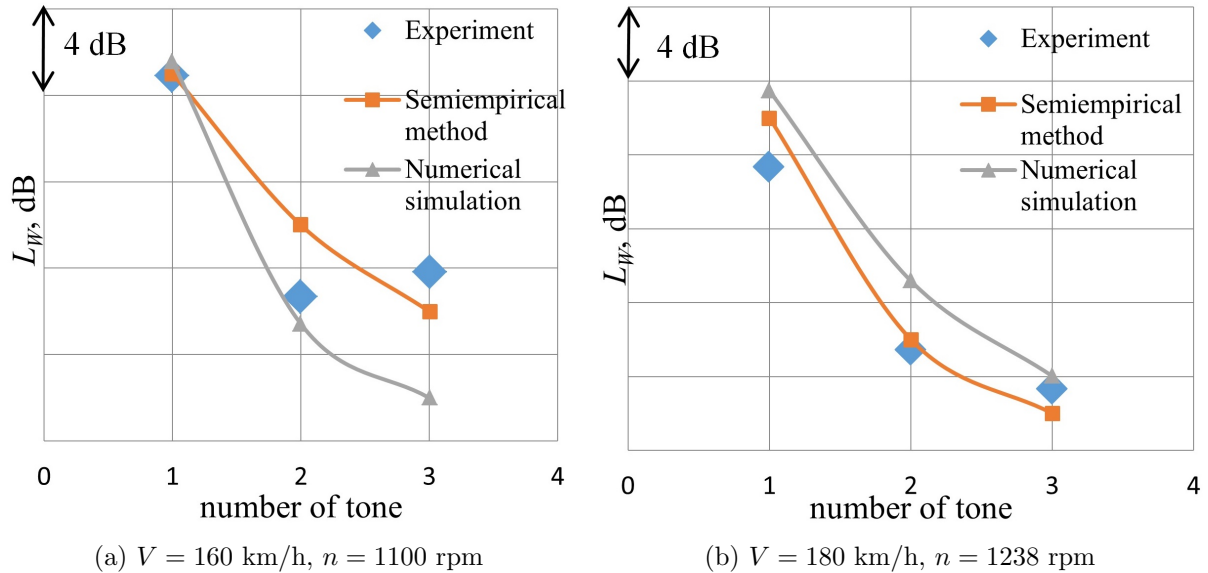


Figure 9. Comparison of sound power levels of the first three propeller noise tones obtained in the experiment with numerical and semi-empirical modeling

Conclusions

A numerical simulation of the noise of the AV-2 propeller installed on AN-2 LPDA is performed. The calculations were performed for the cruising flight modes of the aircraft with airspeeds of 160 and 180 km/h corresponding to the propeller speeds of 1100 and 1238 rpm, respectively. The description of the acoustic-vortex method, the calculation area, and the computation grid are presented. The aerodynamic results are presented in a distribution of the instantaneous pressure field. The acoustic results are presented in the form of a sound pressure field and the amplitude of the first BPF tone of the propeller noise. The data indicates the dominance of dipole noise at the frequency of the first tone, that is, noise from a steady aerodynamic load. For the studied propeller, the thickness noise is insignificant in the flight modes considered, consistent with well-known propeller noise theories and experimental data.

The numerical simulation results are compared with the results of the in-flight experiment and the results of calculating the propeller noise by the semi-empirical method. The data obtained in the experiment agree with the results of numerical and semi-empirical modeling. Verification results indicate the possibility of using the acoustic-vortex method in solving the problem of computing the noise of propellers in tractor configurations. As a parameter for comparative evaluation, the sound power level for the first three tones at frequencies multiple of the blade passing frequency was selected. This parameter does not depend on the distance from the source and characterizes the propeller sound energy emission.

The presented work continues to investigate the possibility of using the acoustic-vortex method to simulate the propeller noise in actual aircraft configurations, considering the installation effects (blade-wake interaction noise, blade-vortex interaction noise, scattering noise of airframe elements, etc.).

Acknowledgements

The authors thank Kozhevnikov Yevgeniy, head of the Moscow Aviation Institute (National Research University) local aerodrome, for organizing acoustics tests.

This paper is distributed under the terms of the Creative Commons Attribution-Non Commercial 3.0 License which permits non-commercial use, reproduction and distribution of the work without further permission provided the original work is properly cited.

References

1. Pettingill, N.A., Zawodny, N., Lopes, L.V.: Acoustic and Performance Characteristics of an Ideally Twisted Rotor in Hover. AIAA Scitech 2021 Forum, AIAA Paper No. 2021-1928 (2021). <https://doi.org/10.2514/6.2021-1928>
2. Yang, Y., Liu, Y., Li, Y., *et al.*: Aerodynamic and Aeroacoustic Performance of an Isolated Multicopter Rotor During Forward Flight. AIAA Journal 58(2) (2020). <https://doi.org/10.2514/1.J058459>
3. Zhou, T. Jiang, H., Sun, Y., *et al.*: Acoustic Characteristics of a Quad-Copter Under Realistic Flight Conditions. 25th AIAA/CEAS Aeroacoustics Conference, AIAA Paper No. 2019-2587 (2019). <https://doi.org/10.2514/6.2019-2587>
4. Moshkov, P., Samokhin, V., Yakovlev, A.: About the community noise problem of the light propeller aircraft. Akustika 34, 68–73 (2019). <https://doi.org/10.36336/akustika20193466>
5. Moshkov, P., Samokhin, V., Yakovlev, A., Bolun, C.: The problems of selecting the power plant for light propeller-driven aircraft and unmanned aerial vehicle taking into account the requirements for community noise. Akustika 39, 164–169 (2021). <https://doi.org/10.36336/akustika202139162>
6. Titarev, V.A., Faranosov, G.A., Chernyshev, S.A., Batrakov, A.S.: Numerical modeling of the influence of the relative positions of a propeller and pylon on turboprop aircraft noise. Acoust. Phys. 64, 760–773 (2018). <https://doi.org/10.1134/S1063771018060118>
7. Drofelnik, J., Andrejasic, M., Mocan, B., *et al.*: Measurement and modelling of aero-acoustic installation effects in tractor and pusher propeller architectures. 2021 AIAA AVIATION Forum, AIAA Paper No. 2021-2301 (2021). <https://doi.org/10.2514/6.2021-2301>
8. Vieira, A., Snellen, M., Malgoezar, A.M.N., Merino-Martinez, R., Simons, D.G.: Analysis of shielding of propeller noise using beamforming and predictions. JASA 146(2), 1085–1098 (2019). <https://doi.org/10.1121/1.5121398>

9. Vieira, A., Malgoezar, A., Snellen, M., Simons, D.G.: Experimental study of shielding of propeller noise by a wing and comparison with model predictions. *Euronoise-2018*, 237–244 (2018).
10. Rathgeber, R., Sipes, D.: The Influence of Design Parameters on Light Propeller Aircraft Noise. SAE Technical Paper No. 770444 (1977). <https://doi.org/10.4271/770444>
11. Dahan, C, Avezard, L., Guillien, G., *et al.*: Propeller Light Aircraft Noise at Discrete Frequencies. *Journal of Aircraft* 18(6), 480–486 (1981). <https://doi.org/10.2514/3.57515>
12. Berton, J.J., Nark, D.M.: Low-Noise Operating Mode for Propeller-Driven Electric Airplanes. *Journal of Aircraft* 56(4), 1708–1714 (2019). <https://doi.org/10.2514/1.C035242>
13. Timushev, S., Yakovlev, A., Moshkov, P.: Numerical simulation of the light aircraft propeller noise under static condition. *Akustika* 41, 100–106 (2021). <https://doi.org/10.36336/akustika202141100>
14. Moshkov, P.: Study of the influence of in-flight conditions on the light propeller-driven aircraft noise. *Aerospace systems* 5, 131–140 (2022). <https://doi.org/10.1007/s42401-021-00127-5>
15. Timushev, S., Klimenko, D., Aksenov, A., *et al.*: On a new approach for numerical modeling of the quadcopter rotor sound generation and propagation. *Proceedings of 2020 International Congress on Noise Control Engineering, INTER-NOISE* (2020).
16. Timushev, S., Yakovlev, A., Klimenko, D.: CFD-CAA Method for Prediction of Pseudosound and Emitted Noise in Quadcopter Propeller. *International Journal of Modeling and Optimization* 12(1), 21–25 (2022). <https://doi.org/10.7763/IJMO.2022.V12.794>
17. Herniczek, M.T.K., Feszty, D., Meslioui, S., Park, J.: Applicability of Early Acoustic Theory for Modern Propeller Design. *23rd AIAA/CEAS Aeroacoustics Conference, AIAA Paper No. 2017-3865* (2017). <https://doi.org/10.2514/6.2017-3865>
18. Herniczek, M.T.K., Feszty, D., Meslioui, S., *et al.*: Evaluation of Acoustic Frequency Methods for the Prediction of Propeller Noise. *AIAA Journal* 57(6) (2019). <https://doi.org/10.2514/1.J056658>
19. Samokhin, V.F.: Semiempirical method for estimating the noise of a propeller. *Journal of Engineering Physics and Thermophysics* 85(5), 1157–1166 (2012). <https://doi.org/10.1007/s10891-012-0758-y>
20. Moshkov, P.A, Samokhin, V.F.: Integral model of noise of an engine-propeller power plant. *Journal of Engineering Physics and Thermophysics* 91(2), 332–338 (2018). <https://doi.org/10.1007/s10891-018-1753-8>

Saddle Point Method Interpretation of Transient Processes in Car Tires

*Kseniia S. Kniazeva*¹, *Yoshinori Saito*², *Andrey I. Korolkov*¹,
*Andrey V. Shanin*¹

© The Authors 2023. This paper is published with open access at SuperFri.org

The problem of mechanical excitation of a suspended tire is studied experimentally and theoretically. The tire is considered as an elastic waveguide. Its numerical description is provided by the Waveguide Finite Element Method (WFEM). A case of tire excitation by a δ -shaped pulse is considered, which corresponds to a short kick applied to some point of the tire. The paper focuses on asymptotic analysis of the formal solution. Mainly, a forerunner is evaluated, which is a fast non-stationary wave having an exponential decay. A modification of the saddle point method, namely, a multi-contour saddle point method, is applied for such an estimation. In the framework of this method, we look for the saddle points of the analytical continuation of the dispersion diagram of the waveguide, taking into account that the contours of integration form a family of curves on the dispersion diagram. The tire pulse response is also measured experimentally. A good agreement between the experimentally observed forerunner and its theoretical prediction is shown.

Keywords: transient processes in car tires, forerunner, carcass of the dispersion diagram, complex dispersion diagram, multi-contour saddle point method.

Introduction

Tire road noise is known to be a significant contributor to the environmental noise. Naturally, design of “quiet” tires is of great importance for different tire manufacturers. Investigations on how tire noise is born are held since nearly the middle of the last century. A notable progress is achieved, however there are still some unanswered questions on this topic.

The mechanisms of exterior noise generation by tires are usually divided into aero-acoustic induced and vibration-induced ones [27]. The first type is connected to turbulent air motion mostly, and noise of the second type is produced as a result of interaction of tire vibrations to the surrounding air. In the current paper we describe the generation mechanisms of the second type, namely, we are interested in noise, which can be associated with certain tire eigenmodes.

We consider oscillations in a suspended tire, i. e., a tire not touching the ground. Such a study makes sense since a lot of important processes happen on the part of the tire far from the contact patch.

Since a tire is a resonator, it is natural to represent its motion as a sum of modes:

$$U(t, \varphi, Y) = W_{\nu,j}(Y)e^{-i\omega_{\nu,j}t+i\nu\varphi},$$

where U is the (vector) field of displacements in the tire, φ is the azimuth (or circumferential) angle ($0 \leq \varphi \leq 2\pi$), $\omega_{\nu,j}$ is an eigenfrequency, j is a mode number, Y is the set of two variables in the transversal cross-section of the tire. Function $W_{\nu,j}(Y)$ is a profile of the mode, its three components describe polarization. Value ν is an integer *circumferential order* of the mode.

There exists a number of papers (for example, [13, 20, 26, 28]) devoted to classification of tire modes. A good classification based on mechanisms of modes behavior could help in control of resonances of a tire. However, it is not a simple task to categorize mode profiles $W_{\nu,j}(Y)$.

¹Lomonosov Moscow State University, Moscow, Russian Federation

²Nihon Michelin Tire Co. Ltd., Ota Site, Ota, Japan

Indeed, for any tire, one can distinguish a number of groups of modes, the profiles of which are similar. For example, there can be identified a class of flexural modes, the cross-section of which bends like a plate, while the circumferential motion is weak. Another class features a strong circumferential motion. Each of these classes is quite numerous. The problem is that a tire also has a lot of mixed modal shapes, which combine properties of different classes. This fact makes the number of modal shapes groups big, such a classification is too complex and arbitrary.

A helpful idea is to introduce a waveguide associated with a given tire [7, 8], which we call a *spiral tire*. To build such a spiral tire, we assume that angle coordinate φ can take real values from $-\infty$ to ∞ . Then the circumferential order ν becomes an (angular) wavenumber, and it can take real values rather than integer ones. The modes in the spiral tire have the form

$$W_j(Y; \nu) e^{-i\omega_j(\nu)t + i\nu\varphi}.$$

Possible values of ν and ω form a *real dispersion diagram* with continuous modal branches $\omega_j(\nu)$. Modes of the initial non-spiral tire can be obtained from the spiral tire dispersion diagram by picking out the modes with integer ν . Indeed, a tire in this formulation is a segment of a waveguide, $0 \leq \varphi \leq 2\pi$, on the ends of which the periodicity conditions are imposed.

The modes of the spiral tire with similar modal shapes form branches of the dispersion diagram. The mixed modal shapes appear, where these branches come closer to each other. Usually, such behavior of the modal branches is referred to as *avoided crossing* [15], *avoiding crossing* [22], *mode repulsion* [14], *veering* [26], *diabolical points* [1], *mode splitting*, *double cusps*, *osculation points* [10], etc.

The most important reason for building the dispersion diagram of a “spiral tire” (or of any other waveguide) is that it enables to predict the propagation speed for narrow-band modal pulses. This concept is known as *group velocity*, and it can be considered as the speed of energy transfer in a waveguide [25]. The group velocity reasoning is based on the stationary phase method. Namely, for given large φ and t , the point on the dispersion diagram with

$$\frac{d\omega}{d\nu} = \frac{\varphi}{t} \tag{1}$$

is a stationary phase point for the corresponding Fourier integral, and the field is estimated by a local integration near this point. Thus, $d\omega/d\nu$ can be treated as an angular group velocity. The linear group velocity is

$$v_{gr} = r \frac{d\omega}{d\nu}, \tag{2}$$

where r is the radius of the tire. In the standard analysis, this reasoning is held only on a real dispersion diagram. The resulting field components may have some decay due to dispersion, but the amplitude behaves like a power of propagation distance φ (not exponentially).

A research held by Bolton et al. [3] showed that waveguide propagation in tires can be observed experimentally. The authors excited a tire by a random force and measured velocity on a tire surface at different points by a laser vibrometer. After the data had been processed, the pulse response was obtained. Authors observed exponentially decaying waves, i. e., with ν and ω having non-zero imaginary parts. Applying Fourier transform, the authors identified $\text{Re}[\nu]$ and $\text{Re}[\omega]$ for the most effectively excited modes. To calculate $\text{Im}[\nu]$, the authors applied the Prony series method. The results showed that excited waves were formed by some spots of the *complex dispersion diagram* (see below) of a “spiral tire”. Another result of the study was the identification of the modal shapes of the excited modes. They are flexural modes with different

transversal index j and a fast mode connected with extension of the tread and having strong displacement in the circumferential displacement.

It was shown in [18, 19], and later on in [21, 22, 24] that the dispersion diagram for a waveguide can be analytically continued into the domain of complex values of ν and ω to obtain the *complex dispersion diagram*. In other words, the complex dispersion diagram is a set of all, complex and real, solutions of the dispersion equation. Studying of the complex dispersion diagram instead of the real one can help one to describe exponentially decaying transient processes in a waveguide. Indeed, instead of the stationary phase method one should use the saddle point method (which is essentially the same).

In the current paper, we develop a method to estimate a response on a δ -shaped force applied to some point on a suspended tire. Our method allows one to predict pulses in a tire and find modal shapes corresponding to these pulses. We are especially interested in *forerunners*, the propagation velocity of which is higher than any group velocity on the real dispersion diagram. Such pulses are of interest in the context of tire noise because they can effectively radiate sound into the surrounding air [3, 27]. The forerunners typically possess exponential decay, but this decay is not strong enough to make the amplitude negligibly small.

Note that the forerunners are a diffraction phenomenon and their attenuation is not connected to damping in tire compounds materials. In this paper we consider only the tires made of materials without energy losses. Indeed, this is a slightly idealized formulation. Beside the simplifications mentioned above, we ignore the interaction of elastic modes with the cavity modes and the radiation of sound into the outer air.

In this paper we describe the wave process in the tire by a WFEM equation (3) (see below) and apply to it a technique similar to the one described in [24], i. e., we find the solution in the form of a Fourier transform and estimate the integrals by the saddle point method assuming that φ is large enough. As it is known from the standard theory, the saddle point method starts with finding of the saddle (or stationary) points on the dispersion diagram, i. e., the points for which the equation (1) is valid for a given value φ/t . Then, one should deform the contours of integration in the initial integrals in such a way that they pass through some of the saddle points and the integrand decays exponentially outside the saddle points. Finally, one should estimate the integral near the saddle points.

Each saddle point on the complex dispersion diagram is related to three values: ν , ω , and the ratio φ/t . A set of all complex saddle points for all admissible (i. e., real) values of φ/t is called a *carcass* of the dispersion diagram. The real dispersion diagram belongs to the carcass as a whole, but there are also important complex branches of the carcass that describe decaying pulses. The latter can give significant contribution into the wave field for moderately large values of φ .

Thus, to estimate the wave field, we make the following steps:

- study the complex dispersion diagram for a tire;
- search for carcass points in the complex domain of ν and ω ;
- estimate their contribution into the field;
- sum up contributions from different carcass points.

The plan of the paper is as follows. Some details of the experimental data are described in Section 1. In Section 2 a WFEM equation is formulated. In Section 3 we find a solution in the form of double integral over ν and ω . Then, applying the residue method, we find modal decomposition for the pulse response of a spiral tire. Also, a complex dispersion diagram is discussed. The

carcass for a tire is built in Section 4. The procedure of estimation of the contributions of the carcass points is described in Section 5. In Section 6 we compare the experimental results and the theoretical predictions of the fast transient waves field. Then we make a conclusion.

1. Experimental Evidence of Forerunners in a Tire

Here we describe an experiment for measurement of the pulse response in a suspended tire. The geometry of the experiment is shown in Fig. 1 and Fig. 2. A tire is excited by a force applied to some point on the outer surface of the tire located at the sagittal (central) plane. The force is directed radially towards the tire center. Let the azimuth coordinate of the source be equal to 0° . The time profile of the force is a δ -function³. A detector is placed also in the sagittal plane in different positions characterized by the azimuth angle φ . The detector is one-axis, polarized normally to the tire. The detector provides the oscillation acceleration a_n of a point of the tire.

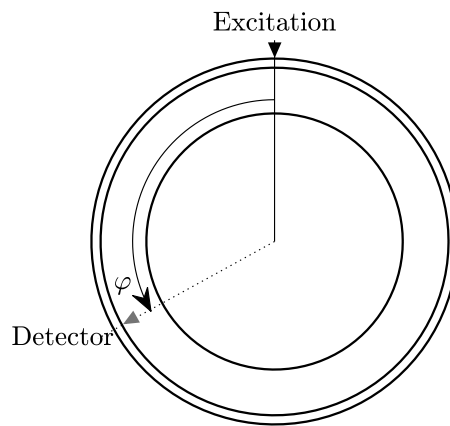


Figure 1. Side view of a tire

The front parts of three typical profiles of pulse response are shown in Fig. 3. From this data, one can easily find that velocity of propagation of the maxima in the area $t < 4$ s is about 350 m/s, and velocity of propagation of maxima in the right part of the figure is about 70 m/s. Let us compare these values to the values of the group velocities on the real dispersion diagram.

A real dispersion diagram for the considered spiral tire is shown in Fig. 4. Corresponding group velocities are plot in Fig. 5. One can see that the maximum value of the group velocity is about 250 m/s. Pulses propagating with velocities higher than this value are the forerunners, in particular, the first pulse in Fig. 3 is a forerunner. The slow pulse appears to be described by the real dispersion diagram, and it corresponds mainly to flexural modes.

Due to the way of the excitation and the fact that the considered tire was almost symmetrical with respect to the central plane, the most effectively excited modes must have significant radial displacement in the central part of a cross-section. One can check that points satisfying this characteristic are located below the line $v_{gr} = 142$ m/s, this area is shown by grey in Fig. 5. That is, we can expect that the wave field for $142 < r\varphi/t < 250$ (m/s) is also formed by some pulses, whose nature is close to the forerunners.

³Indeed, such time profile is a result of a correlation-based data processing. We omit details here.



Figure 2. Tire profiles at the source and the detector azimuth positions

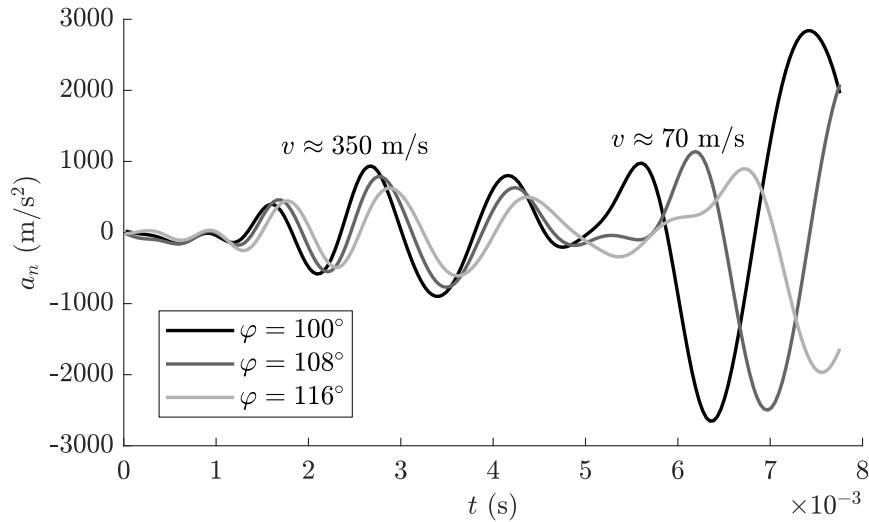


Figure 3. Pulse response of a tire

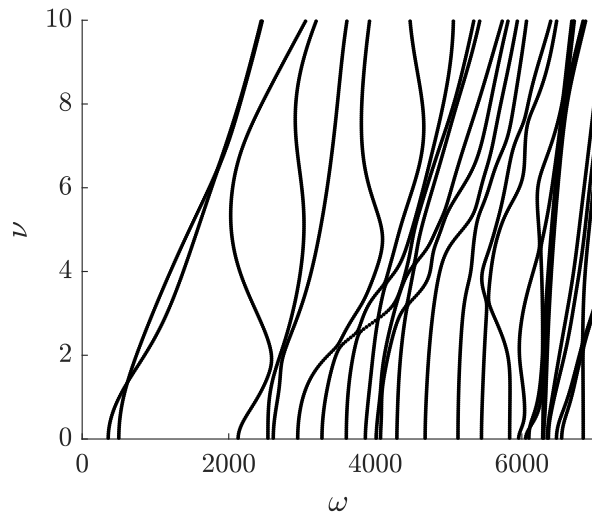


Figure 4. Real dispersion diagram for the chosen tire

We should note that the critical velocity of the pulse in noise-related application is the velocity of sound in air, that is 343 m/s under normal conditions. An elastic pulse that travels faster can radiate a powerful airborne noise due to Cherenkov’s mechanism. Indeed, formally, this should be the *phase* velocity of the pulse, i. e., the velocity of pulse maxima/minima. However, as we see below, in our case the phase and group velocities are close to each other.

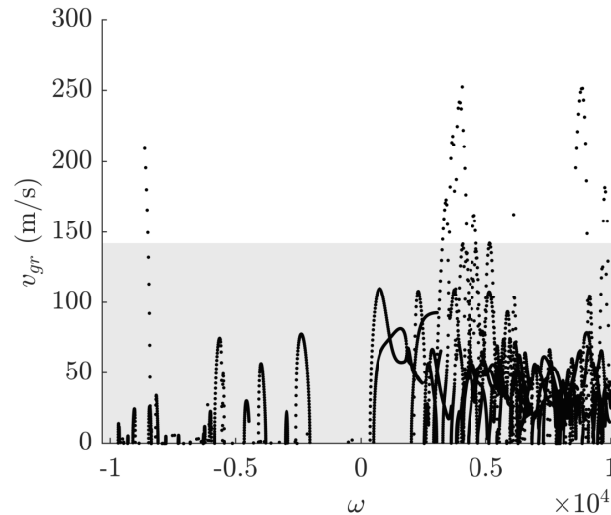


Figure 5. Group velocities for real ω for the chosen tire. Stationary points significantly contributing to the field (due to the excitation way) are located in the grey area

2. Numerical Description of the Tire Using WFEM

We assume that the tire is homogeneous in the angular direction and that the tire cross-section is described by a dense enough mesh. The field is described by values of three spatial components of displacement in each of the mesh nodes. These values form a vector of unknowns $U(t; \varphi)$. The dimension of this vector is $3N$, where N is the number of mesh nodes on the cross-section of the tire. It is shown in [7, 8, 11, 16] that a good mathematical model for such a tire is *Waveguide Finite Element Method* (WFEM). The method is also called *Semi-analytical finite element method* [14], *Hybrid two-dimensional finite element method* [11], *Matrix Klein-Gordon equation (MKGE)* [12]. In the framework of this method, the field is described by a partial differential equation for U :

$$(\mathbf{D}_2 \partial_\varphi^2 + \mathbf{D}_1 \partial_\varphi + \mathbf{D}_0 - \mathbf{M} \partial_t^2) U(t, \varphi) = F(t, \varphi), \quad -\infty < \varphi, t < \infty. \quad (3)$$

The right-hand side $F(t, \varphi)$ is a vector that describes the excitation. Matrices $\mathbf{D}_2, \mathbf{D}_1, \mathbf{D}_0, \mathbf{M}$ are constant; they depend on the tire structure and materials. Since we ignore the attenuation in the materials forming the tire, matrices $\mathbf{D}_2, \mathbf{D}_1, \mathbf{D}_0, \mathbf{M}$ are assumed to be real and having the following properties:

$$\mathbf{M}^T = \mathbf{M}, \quad \mathbf{D}_0^T = \mathbf{D}_0, \quad \mathbf{D}_1^T = -\mathbf{D}_1, \quad \mathbf{D}_2^T = \mathbf{D}_2. \quad (4)$$

Besides, matrices \mathbf{M} and \mathbf{D}_2 should be positive-definite, and \mathbf{D}_0 should be a non-negative definite matrix. The matrices \mathbf{D}_m and \mathbf{M} are provided by mechanical modeling (say, FEM-based) and contain the whole mechanical structure of the tire. Below we do not consider this mechanical matter and concentrate on the mathematical properties of (3). A brief derivation of WFEM (3) is described in [23].

According to the experimental set-up, we assume that the excitation is delta-shaped with respect to t and φ :

$$F \delta(t) \delta(\varphi),$$

where \mathbf{F} is a constant vector. Since the force is applied to a small spot on the tire surface and has a radial polarization, we assume that \mathbf{F} has a single non-zero component corresponding to a node of application and a polarization of the force.

3. Formal Solution of the Governing Equation

Let us find a formal solution of (3). Let us apply the Fourier transform with respect to φ and Laplace transform with respect to t . The result is a linear algebraic system:

$$\mathbf{P}\tilde{\mathbf{U}} = \mathbf{F}, \tag{5}$$

where $\tilde{\mathbf{U}} = \tilde{\mathbf{U}}(\omega, \nu)$ is a Fourier-Laplace image of $\mathbf{U}(t, \varphi)$, \mathbf{P} is a matrix defined as

$$\mathbf{P} = \mathbf{P}(\omega, \nu) \equiv \omega^2\mathbf{M} - \nu^2\mathbf{D}_2 + i\nu\mathbf{D}_1 + \mathbf{D}_0. \tag{6}$$

Matrix equation (5) can be solved provided \mathbf{P} is non-singular. By inverting the Fourier and Laplace transforms, let us obtain a solution in a double integral form:

$$\mathbf{U}(t, \varphi) = \frac{1}{4\pi^2} \int_{-\infty}^{\infty} \int_{-\infty+i\epsilon}^{\infty+i\epsilon} \mathbf{P}^{-1}\mathbf{F}e^{i\nu\varphi-i\omega t} d\omega d\nu, \tag{7}$$

where ϵ is an arbitrary small positive value (it comes from the Mellin's transform and enforces the causality of (7)).

Let us fix some value ν and consider the inner integral. According to the Jordan's lemma one can close the integral in the lower half of the complex ω -plane. Applying the residue method, one can obtain the series-integral representation:

$$\mathbf{U}(t, \varphi) = -\frac{i}{2\pi} \sum_j \int_{-\infty}^{\infty} \text{Res}[\mathbf{P}^{-1}\mathbf{F}e^{i\nu\varphi-i\omega t}, \omega_j(\nu)] d\nu, \tag{8}$$

where $\omega_j(\nu)$ are solutions of the dispersion equation

$$\det \mathbf{P}(\omega, \nu) = 0 \tag{9}$$

with respect to ω that lie inside the enclosed integral contour on the ω -plane. According to the reality statement [7] all poles of the integrand of the integral in (7) belong to the real ω -axis, and, consequently, all of them participate in (8).

One can derive (see, for example, [16]) that

$$\text{Res}[\mathbf{P}^{-1}, \omega_j(\nu)] = \frac{\mathbf{W}_j\mathbf{V}_j}{2\omega_j(\nu)\mathbf{V}_j\mathbf{M}\mathbf{W}_j}, \tag{10}$$

where $\mathbf{W}_j = \mathbf{W}(\nu, \omega_j(\nu))$ and $\mathbf{V}_j = \mathbf{V}(\nu, \omega_j(\nu))$ are eigenvectors of the left and right eigenvalue problems:

$$\mathbf{P}\mathbf{W} = 0, \quad \mathbf{V}\mathbf{P} = 0. \tag{11}$$

Hence,

$$\mathbf{U}(t, \varphi) = -\frac{i}{4\pi} \sum_j \int_{-\infty}^{\infty} \frac{\mathbf{W}_j\mathbf{V}_j\mathbf{F}}{\omega_j(\nu)\mathbf{V}_j\mathbf{M}\mathbf{W}_j} e^{i\nu\varphi-i\omega_j(\nu)t} d\nu. \tag{12}$$

The dispersion equation (9) can be considered as an equation for ω with a parameter ν . The equation has generally $6N$ solutions $\omega_j(\nu)$ ($3N$ is the dimension of U). It has been shown in [18, 19, 21, 22] that solutions $\omega_j(\nu)$ can be analytically continued into the complex domain of ν to form a Riemann surface R of a multi-valued function $\omega(\nu)$, which is defined implicitly by (9). The index j over which the summation (12) is performed corresponds to number of a sheets of R . Taking this into consideration, one can rewrite (12) as follows:

$$U(t, \varphi) = -\frac{i}{4\pi} \int_{\sum_j \gamma_j} \frac{WVF}{\omega(\nu)VMW} e^{i\nu\varphi - i\omega(\nu)t} d\nu, \quad (13)$$

where γ_j are all preimages of the positively oriented real axis ν on the Riemann surface R , $W = W(\nu, \omega(\nu))$, $V = V(\nu, \omega(\nu))$.

The obtained solution (13) is valid for any waveguide. Its problem is that it does not give a clear understanding of the field structure and of wave pulses propagating in the waveguide. To obtain an insight into the transient processes, we apply a *multi-contour saddle point method* [24]. Unlike the usual saddle point method, to estimate (13), we need to perform the integration on the Riemann surface R having a complicated set of branch points. Thus, before applying the multi-contour saddle point method we need to study the structure of R .

In [23, 24] we demonstrated that it is relevant to consider the Riemann surface R as a projection of a manifold H onto the ν -plane. The manifold H is a set of all possible complex solutions (ω, ν) of the dispersion equation, we call it a *complex dispersion diagram*. Note that since points of H are restricted by one equation (9), the manifold H has complex dimension 1, which is real dimension 2.

Instead of projecting of H onto ν -plane, one can project it onto ω -plane. The result is the Riemann surface R' of a multivalued function $\nu(\omega)$, the sheets of which are solutions of (9) with ω taken as a parameter. Such projections result in branch points on the Riemann surfaces, which should be taken into account while deforming the integration contours. Compared to R and R' , manifold H is smooth, since it is defined by a polynomial equation (9). Thus, the contours of integration can be *freely* deformed on H .

Another consequence of the smoothness of H is that the corresponding modal shapes (eigenvectors) change smoothly when the point (ω, ν) is moving along a continuous curve on H . This fact explains the interchange of the modal shapes near avoiding crossings of the real dispersion diagram.

Also, we can make an obvious conjecture that manifold H is not connected, if, physically, the waveguide can be split into isolated subsystems.

Let us make a short note on numerical solution of the dispersion equation (9). Indeed, this is a polynomial equation of a high dimension. We prefer to consider this equation as a generalized eigenvalue problem, in which ω^2 is the eigenvalue and ν is a complex parameter. Modern systems of linear algebra solve such problems very efficiently, namely, they are capable of finding, say, about 50 smallest or largest eigenvalues. This fits our needs. All complex and real dispersion diagrams below are built this way. As a by-product of this procedure, we obtain right or left eigenvectors, i. e., solutions of (11).

We are going to use a formula for the angular group velocity [2, 7, 14, 16]:

$$\Omega_{gr} \equiv \frac{d\omega}{d\nu} = \frac{2\nu V D_2 W - i V D_1 W}{2\omega V M W}. \quad (14)$$

The formula can be easily derived from (11). This formula allows us to avoid numerical differentiation of the function $\omega(\nu)$.

4. Carcass of the Dispersion Diagram

The main goal of the current paper is to estimate asymptotically the expression (13) and indicate the parts of the complex dispersion diagram mainly contributing into the field by the multi-contour saddle point method.

The equation (13) can be rewritten:

$$U(t, \varphi) = -\frac{i}{4\pi} \int_{\sum_j \gamma_j} \frac{\text{WVF}}{\omega(\nu)\text{VMW}} e^{i\varphi g(\nu, \omega)} d\nu, \quad (15)$$

where $g(\omega, \nu)$ is a *phase function*

$$g(\omega, \nu) = \nu - \omega/\Omega, \quad (16)$$

Ω is a *formal angular velocity*

$$\Omega \equiv \varphi/t. \quad (17)$$

According to the saddle point method, asymptotic estimation of an integral is a sum of saddle point integrals, and the saddle points are the points where derivative of the phase function is zero:

$$\frac{dg(\omega, \nu)}{d\nu} = 0, \quad p = (\omega, \nu) \in H. \quad (18)$$

Substituting (16), one can obtain (1). Note that we assume that

$$\left. \frac{d^2g(\omega, \nu)}{d\nu^2} \right|_p \neq 0$$

at each saddle point. If this is not true, a more elaborate asymptotic (e.g., an Airy one) is needed to be built.

Using function (16), we obtain the condition for finding the saddle point

$$\Omega_{gr}(p) = \Omega, \quad (19)$$

where the function $\Omega_{gr}(p)$ is defined by (14).

Let us consider all points $p \in H$, real and complex, and find the set of all such points that $\Omega(p)$ is real. Indeed, potentially any such point may become a saddle point for some Ω . This set is called a *carcass of the dispersion diagram*.

Not all points of the carcass contribute into the field. Namely, to estimate the integral (13) for certain (t, φ) , one should deform the initial integration contour into the steepest descend contour. The latter will pass through some points obeying the relation (19) (and thus contribute to the field), but not necessarily through each of them. The points $p \in H$, through which the steepest descend contour passes when $\varphi/t = \Omega_{gr}(p)$, are called *active* points of carcass, and the rest ones are *not active*.

Obviously, points of the real dispersion diagram satisfy (1), i. e., belong to carcass. Moreover, these points are active, because they define far field, according to the stationary phase method.

A good example of active and not active saddle points is provided by an example based on the integral representation of the Airy function:

$$\text{Ai}(\xi) = \frac{\sqrt{\xi}}{2\pi} \int \exp\{i\xi^{3/2}(\tau^3/3 + \tau)\}d\tau.$$

For large positive ξ , there are two saddle points, $\tau = \pm i$, but only the point $\tau = i$ is active, since the contour of integration should be shifted into the upper half-plane.

The procedure of deformation of the initial contour of integration into a steepest descend one is quite complicated. Fortunately, there exists an efficient algorithm of finding the active points on the carcass without making the whole deformation [6, 24]. The algorithm is as follows:

1. We consider only $\varphi, t > 0$, so carcass points with $\Omega_{gr} < 0$ are marked as not active.
2. Carcass points p with $\text{Im}[g(p)] < 0$ are marked as not active.
3. From other carcass points two contours of steepest descend of $\text{Im}[g]$ are built. If
 - both of them hit the real axis ν , then the point is not active;
 - both of the contours go to infinity on the complex plane ν , then the point is also not active;
 - only one of the contours hits the real axis, then the point is active.

The results of building a carcass and finding active points on the carcass are shown in Fig. 6. As one can see the carcass is composed of 1D curves (branches). We assume that each branch corresponds to a certain mode shape (at least that the mode shape varies continuously along the branch). Thus, one can associate a mode with each branch (corresponding pulses are attenuating).

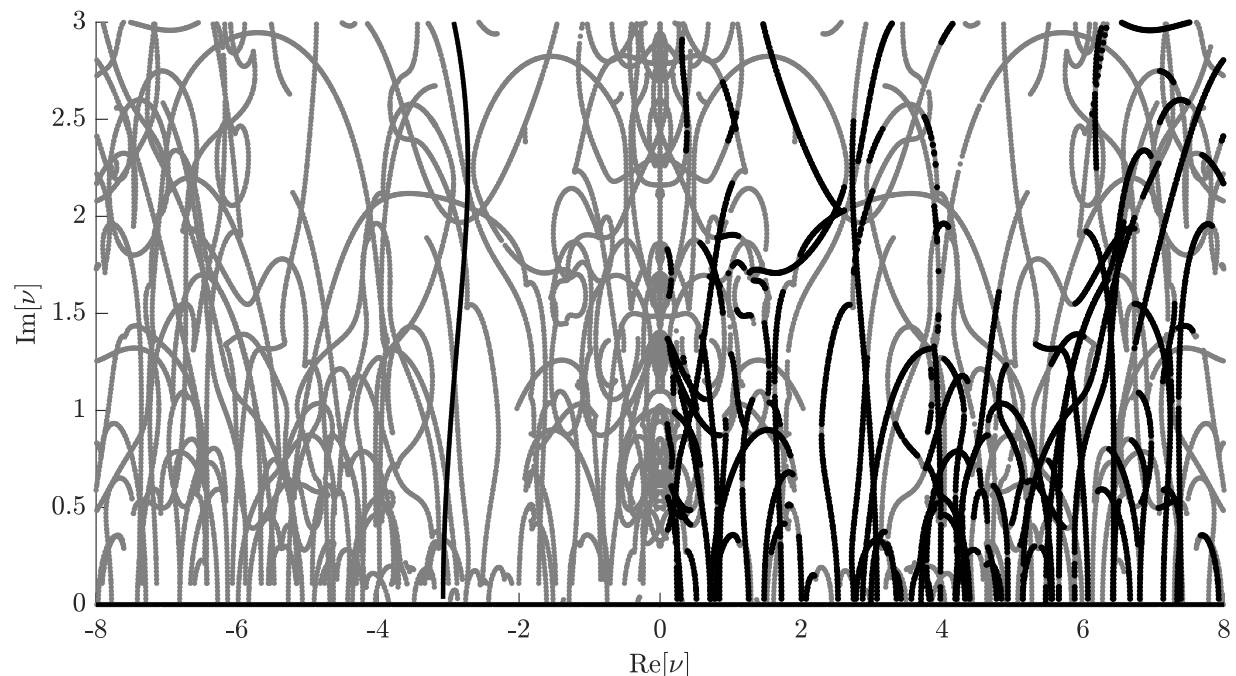


Figure 6. Projection of the carcass on ν -plane: found not active points (grey), active branches (black)

Most of the active carcass branches start from the real axis. The points, where the complex carcass branches start to grow from the real axis, are the points of the group velocity extremes (maxima or minima). As it is known [17] such points provide Airy-type asymptotics.

There can exist points with negative group velocities on the real dispersion diagram (it can be seen in the real dispersion diagram Fig. 4). It follows from the dispersion equation (9) that if a point with positive real ν and ω has a negative group velocity, then the point $(-\nu, \omega)$ has the opposite (positive) value of the group velocity. Such points can also become start points for some complex carcass branches. One of such active carcass branches can be found in the domain $\text{Re}[\nu] < 0$ in Fig. 6.

5. Computing Field Asymptotics at Active Saddle Points

According to the saddle point method, (13) can be represented as a sum of saddle point integrals:

$$U(t, \varphi) \approx \sum_m U_{*m}(t, \varphi), \quad (20)$$

where $U_{*m}(t, \varphi)$ is a contribution of the m -th active carcass point.

Since each active carcass branch is a curve on which Ω changes monotonically, a carcass branch forms a pulse. Our plan is to calculate the field for each of the active carcass branches and then sum them up.

Formula for a contribution U_{*m} of a carcass point (ω_{*m}, ν_{*m}) into the full wave field can be derived in a rather standard way (see, for example, [4]):

$$U_{*m} = \pm i \sqrt{\frac{1}{2\pi t \left| \frac{d^2\omega}{d\nu^2} \Big|_{\nu_{*m}} \right|}} \exp \left\{ -i \left(\frac{1}{2} \arg \left(\frac{d^2\omega}{d\nu^2} \Big|_{\nu_{*m}} \right) + \pi/4 \right) \right\} \frac{W_{*m} V_{*m} F e^{i\nu_{*m}\varphi - i\omega_{*m}t}}{2\omega_{*m} V_{*m} M W_{*m}}, \quad (21)$$

where $W_{*m} = W(\omega_{*m}, \nu_{*m})$ and $V_{*m} = V_{*m}(\omega_{*m}, \nu_{*m})$ are the (right/left) eigenvectors at the carcass point. The sign in (21) is chosen as follows:

- “minus” if

$$\frac{\pi}{2} < \arg \left(\frac{d^2\omega}{d\nu^2} \Big|_{\nu_{*m}} \right) \leq \pi,$$

- “plus” if

$$-\pi < \arg \left(\frac{d^2\omega}{d\nu^2} \Big|_{\nu_{*m}} \right) < \frac{\pi}{2}.$$

Let us recall that the detector measures the radial acceleration a_n in the inner central point of the tire cross-section φ (see Fig. 2). Similar to (20), value a_n can be represented as a sum of the saddle point contributions:

$$a_n(t) \approx \sum_m (a_n)_{*m}(t). \quad (22)$$

Each of the acceleration contributions $(a_n)_{*m}$ is the second time derivative of the corresponding saddle point contribution into the displacement field $(u_n)_{*m}$:

$$(a_n)_{*m}(t) = -\omega_{*m}^2 (u_n)_{*m}(t). \quad (23)$$

A displacement contribution $(u_n)_{*m}$ is the necessary component of the vector U_{*m} calculated by (21).

In Fig. 7 we plot examples of the pulses formed by some of the active carcass branches. The detector position was $\varphi = 100^\circ$. The legend of the figures shows the point on the real ν -axis,

where the active carcass branches start to grow into the complex domain. For example, the most left black branch in Fig. 6 intersects the real axis at $\nu \approx -3.08$. According to the legend for Fig. 7b), this value corresponds to the black dashed line.

One can see that most of the pulses are oscillating and attenuating for small t . Another observation is that there is no prevailing pulse, which determines the field for small t . Instead, there are few dozens of pulses, the amplitude of which are of the same order.

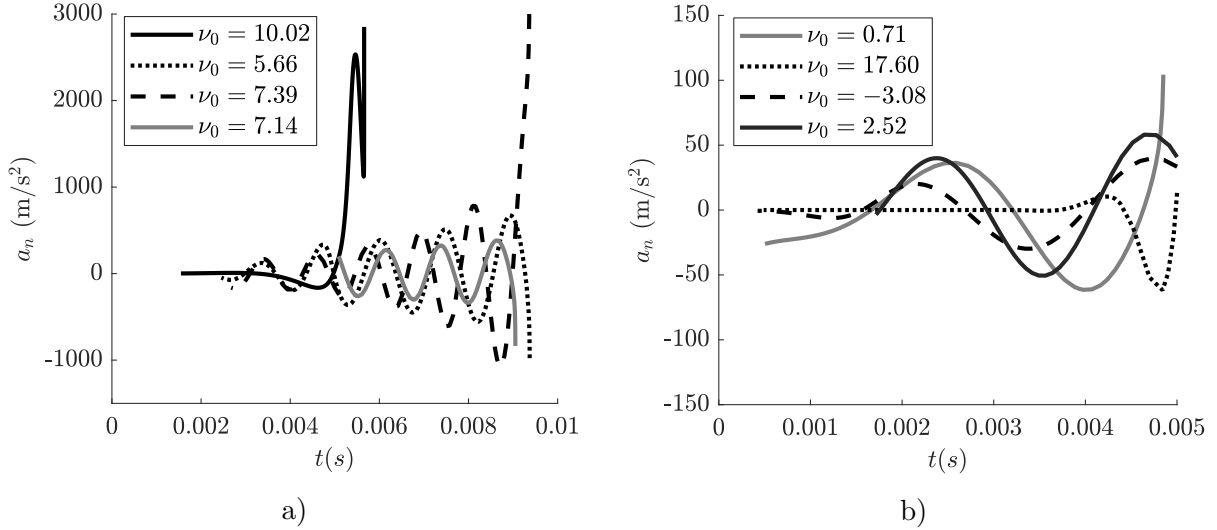


Figure 7. Examples of pulses formed by some of the found active carcass branches. The detector is placed at $\varphi = 100^\circ$

6. Comparison of the Experimental and Theoretical Results

Finally, we find the sum of the calculated pulses for the tire and compare it to the experimental results (see Fig. 8). The black curves in the figure show the saddle point asymptotic, the grey ones show the experiment data.

The area, where the stationary points of the real dispersion diagram can contribute to the field, is highlighted by light grey. As it was discussed in Section 1, the boundary of this area is defined by the maximum group velocity on the real dispersion diagram corresponding to a modal shape with strong radial displacement in the center of a tire cross-section. Since we have considered only the complex branches of the carcass, the curves are not supposed to coincide in the grey area.

We observe that *before* the light grey domain our method describes the wave field adequately. We obtain that the forerunners field is formed by approximately 20 complex carcass branches.

It is worth to recall also that we excluded dissipation losses from our model. Probably, if there were a good instrument to describe the attenuation correctly, even better results could be obtained. However, the problem of description of energy losses in tires is known to be difficult. In a suspended tire the attenuation occurs, mainly, due to the hysteretic character of rubber deformation (see, for example, [5, 9]).

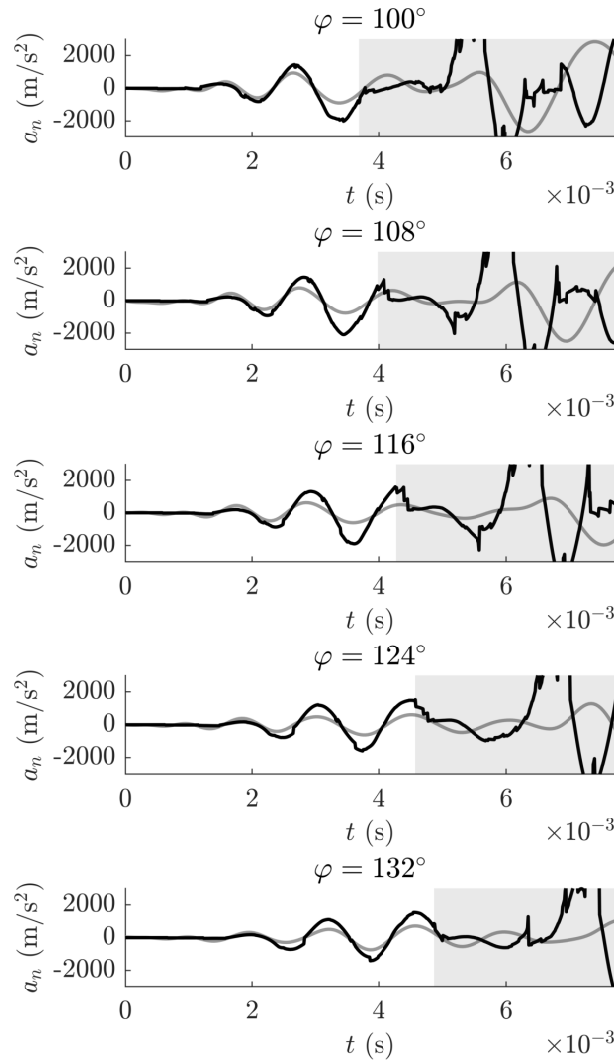


Figure 8. δ -pulse response of the tire for different detector positions: experimental results (grey curves) and the saddle point asymptotics (black curves). Light grey area is the area, where the stationary points of the real dispersion diagram start to contribute

Conclusion

In the current paper, we study non-stationary forerunners in a mechanically excited suspended tire. The consideration is held experimentally and theoretically.

The theoretical model of the tire is provided by a WFEM equation (3). A formal solution is obtained in the form of a sum of integrals (12). We apply the multi-contour saddle point method to evaluate the field contribution corresponding to forerunners.

In the framework of the method, firstly, we compute the complex dispersion diagram of the tire by numerical solution of the eigenproblem (11). After that, we find a carcass, i. e., a set of points on the complex dispersion diagram, on which the value of $d\omega/d\nu$ is real. Then, we classify the found carcass points into active and not active ones by means of the algorithm developed in [6, 24]. The active carcass points are shown to form branches, each of which corresponds to a certain pulse. Finally, to obtain an asymptotic estimation of the non-stationary field, we sum all the calculated pulses.

An asymptotic estimation of a pulse response for a tire is calculated by this method. Then it is compared to the tire pulse response obtained by the experiment briefly discussed in the paper.

A good agreement is observed between the theoretically predicted shape of the forerunners and the experimentally measured pulses.

An important and slightly unexpected conclusion is that the forerunner is described well only by a considerable number of carcass branches (about 20). This probably means that the forerunner is a *ray pulse* rather than a *modal pulse*. Still, we are not aware of a convenient theory of rays in systems described by WFEM equation. Our finding may be a good motivation for building of such a theory.

This paper is distributed under the terms of the Creative Commons Attribution-Non Commercial 3.0 License which permits non-commercial use, reproduction and distribution of the work without further permission provided the original work is properly cited.

References

1. Berry, M.V., Wilkinson, M.: Diabolical points in the spectra of triangles. Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences 392(1802), 15–43 (mar 1984). <https://doi.org/10.1098/rspa.1984.0022>
2. Biot, M.A.: General theorems on the equivalence of group velocity and energy transport. Physical Review 105(4), 1129–1137 (feb 1957). <https://doi.org/10.1103/physrev.105.1129>
3. Bolton, J.S., Kim, Y.J.: Wave number domain representation of tire vibration. In: Proceedings of Inter-noise 2000. vol. 1, pp. 184–190 (2000)
4. Borovikov, V.A.: Uniform stationary phase method. IEEE (1994)
5. Collins, J.M., Jackson, W.L., Oubridge, P.S.: Relevance of elastic and loss moduli of tire components to tire energy losses. Rubber Chemistry and Technology 38(2), 400–414 (may 1965). <https://doi.org/10.5254/1.3535661>
6. Feldbrugge, J., Lehnert, J.L., Turok, N.: Lorentzian quantum cosmology. Physical Review D 95(10) (may 2017). <https://doi.org/10.1103/physrevd.95.103508>
7. Finnveden, S.: Evaluation of modal density and group velocity by a finite element method. Journal of Sound and Vibration 273(1-2), 51–75 (may 2004). <https://doi.org/10.1016/j.jsv.2003.04.004>
8. Finnveden, S., Fraggstedt, M.: Waveguide finite elements for curved structures. Journal of Sound and Vibration 312(4-5), 644–671 (may 2008). <https://doi.org/10.1016/j.jsv.2007.11.020>
9. Kainradl, P., Kaufmann, G.: Heat generation in pneumatic tires. Rubber Chemistry and Technology 49(3), 823–861 (jul 1976). <https://doi.org/10.5254/1.3534981>
10. Kausel, E., Malischewsky, P., Barbosa, J.: Oscillations of spectral lines in a layered medium. Wave Motion 56, 22–42 (jul 2015). <https://doi.org/10.1016/j.wavemoti.2015.01.004>
11. Kim, Y.J., Bolton, J.S.: Analysis of tire vibration by using a hybrid two-dimensional finite element based on composite shell theory. In: Proceedings of INTER-NOISE 2003. pp. 1344–1351. No. 294 (2003)
12. Korolkov, A., Shanin, A., Kniazeva, K.: Asymptotical study of two-layered discrete waveguide with a weak coupling. In: INTER-NOISE and NOISE-CON Congress and Conference Proceedings. vol. 261, pp. 5133–5144. Institute of Noise Control Engineering (2020)

13. Kung, L.E., Soedel, W., Yang, T.Y., Charek, L.T.: Natural frequencies and mode shapes of an automotive tire with interpretation and classification using 3-D computer graphics. *Journal of Sound and Vibration* 102(3), 329–346 (oct 1985). [https://doi.org/10.1016/s0022-460x\(85\)80146-2](https://doi.org/10.1016/s0022-460x(85)80146-2)
14. Loveday, P.W., Long, C.S., Ramatlo, D.A.: Mode repulsion of ultrasonic guided waves in rails. *Ultrasonics* 84, 341–349 (2018). <https://doi.org/10.1016/j.ultras.2017.11.014>
15. Neumann, J., Wigner, E.: ber das verhalten von eigenwerten bei adiabatischen prozessen. *Phys. Zschr.* 30, 467–470 (1929)
16. Nilsson, C.M.: Waveguide finite elements applied on a car tyre. Ph.D. thesis, Department of Aeronautical and Vehicle Engineering, Royal Institute of Technology (2004)
17. Press, F., Ewing, M., Tolstoy, I.: The Airy phase of shallow-focus submarine earthquakes. *Bulletin of the Seismological Society of America* 40(2), 111–148 (1950)
18. Randles, P.W.: Modal representation for the high-frequency response of elastic plates. Ph.D. thesis, Caltech, Pasadena, CA, 235 (1969)
19. Randles, P.W., Mlkowitz, J.: Modal representations for the high-frequency response of elastic plates. *International Journal of Solids and Structures* 7(8), 1031–1055 (aug 1971). [https://doi.org/10.1016/0020-7683\(71\)90079-5](https://doi.org/10.1016/0020-7683(71)90079-5)
20. Sabiniarz, P., Kropp, W.: A waveguide finite element aided analysis of the wave field on a stationary tyre, not in contact with the ground. *Journal of Sound and Vibration* 329(15), 3041–3064 (jul 2010). <https://doi.org/10.1016/j.jsv.2010.02.008>
21. Shanin, A.V.: Precursor wave in a layered waveguide. *The Journal of the Acoustical Society of America* 141(1), 346–356 (jan 2017). <https://doi.org/10.1121/1.4973958>
22. Shanin, A.V., Knyazeva, K.S., Korolkov, A.I.: Riemann surface of dispersion diagram of a multilayer acoustical waveguide. *Wave Motion* 83, 148–172 (dec 2018). <https://doi.org/10.1016/j.wavemoti.2018.09.010>
23. Shanin, A.V., Korolkov, A.I., Kniazeva, K.S.: Integral representations of a pulsed signal in a waveguide. *Acoustical Physics* 68(4), 316–325 (aug 2022). <https://doi.org/10.1134/s1063771022040108>
24. Shanin, A.V., Korolkov, A.I., Kniazeva, K.S.: Saddle point method for transient processes in waveguides. *Journal of Theoretical and Computational Acoustics* 30(04) (mar 2022). <https://doi.org/10.1142/s2591728521500183>
25. Tolstoy, I.: Modes, rays, and travel times. *Journal of Geophysical Research* 64(7), 815–821 (jul 1959). <https://doi.org/10.1029/jz064i007p00815>
26. Waki, Y., Mace, B.R., Brennan, M.J.: Free and forced vibrations of a tyre using a wave/finite element approach. *Journal of Sound and Vibration* 323(3-5), 737–756 (jun 2009). <https://doi.org/10.1016/j.jsv.2009.01.006>
27. Wang, X. (ed.): *Automotive tire noise and vibrations*. Butterworth-Heinemann (2020)
28. Wheeler, R.L., Dorfi, H.R., Keum, B.B.: Vibration modes of radial tires: measurement, prediction, and categorization under different boundary and operating conditions. *SAE Transactions* 114, 2823–2837 (2005), <http://www.jstor.org/stable/44725319>

Simulation of Polyatomic Gas Cloud Expansion under Pulsed Laser Ablation by Model Boltzmann Equation

Anna A. Frolova¹ 

© The Author 2023. This paper is published with open access at SuperFri.org

Polyatomic gas cloud expansion into vacuum under pulsed laser ablation is studied on the basis of one-dimensional model kinetic equation. To account for the influence of internal energy on the vapor-gas cloud parameters (density, temperature, and velocity) a model kinetic equation of the BGK-type is applied, which considers the energy exchange using a two-temperature model. In this approach, the collision integral is approximated by the sum of two terms corresponding to elastic (translational relaxation) and inelastic (rotational relaxation) collisions. Note that the vibrational energy is not taken into account in this paper. The independence of the differential part of transfer equation from the rotational energy makes it possible to reduce the equation to a system with two functions. A comparison of the gas flow macroparameters obtained by the kinetic equations and the DSMC is carried out. The influence of the number of rotational degrees of freedom on the evolution of average temperature and on the gas parameters is shown. The calculations are performed on non-uniform grids in the phase space with global dynamic velocity mesh adaptation to suppress the “ray effect”.

Keywords: model kinetic equations, rotational degrees of freedom, pulsed laser ablation, non-stationary problems.

Introduction

This paper is devoted to the simulation of a widely used technology for nanomaterials synthesis due to nanosecond pulsed laser ablation (PLA). A large number of numerical, theoretical, and experimental studies is focused on the physical phenomena caused by PLA, see, for example, [1–4] and references therein. However, further research is needed because of the rapid development of technology and the large number of factors affecting the structure of the vapor-gas cloud.

Since the vapor cloud dynamics is accompanied by a transition from a continuum regime to the free-molecular one, numerical modeling of the process at large time intervals is a rather complicated computational problem. At present, the calculations are most often carried out by the direct simulation Monte-Carlo (DSMC). The kinetic approach based on integrating the exact Boltzmann equation or its model approximations by discrete ordinate method (DOM) is rarely used because of high computational cost.

It should be noted that the usage of a fixed velocity grid in the DOM in the case of discontinuous boundary or initial conditions can lead to unphysical oscillations of macroparameters (“ray effect”). Furthermore in the considered problem, there is an additional source of “ray effect”, due to the narrow kernel of the velocity distribution function (VDF) in the velocity space. This is a consequence of boundary values transfer of the distribution function along the characteristics in velocity interval, that decreases with time. The non-uniform velocity grid with refinement in the neighborhood of zero velocity [5, 6] allows to obtain monotonic behavior of macroparameters for high intensity evaporation regime, and in comparison with the results obtained by the DSMC shows a close solution. However, more rarefied flow regimes require dynamic adaptation of the velocity grid. Despite the computational difficulties, the usage of kinetic equations for modeling physical processes represents an important alternative approach.

¹Federal Research Center “Computer Science and Control” of the Russian Academy of Sciences, Moscow, Russia

The vapor-gas cloud after laser ablation can consist of polyatomic particles, so it seems important to study the influence of the molecular internal energy on the vapor dynamics, since the energy exchange affects the density, temperature, and flow velocity, which, in turn, affect the deposition rate and coating quality.

The molecular vapor cloud dynamics was studied using DSMC and continuum approach in [7, 8] and in other papers referenced in [8]. The goal of this paper is to determine the possibility of numerical simulation of the molecular cloud expansion by integration of model kinetic equation and comparing the obtained results with DSMC. Note that only rotational degrees of freedom are considered, since accounting for vibrational degrees of freedom based on model equations in the laser ablation problem requires additional theoretical studies.

The article is organized as follows. In Section 1, we give the formulation of the problem and a generalized form of the model kinetic equation of BGK type for a polyatomic gas. In Section 2, we introduce characteristic quantities for normalizing the kinetic equation in the problem of gas cloud expansion into vacuum during pulsed laser ablation. In Section 3, we compare the results obtained by the model equation and DSMC and show the influence of internal energy on the vapor-gas cloud parameters for different numbers of vaporized monolayers. The good agreement obtained in comparison allows us to consider the model equation as an alternative approach to reduce the computational complexity of solving these kinds of problems.

1. Formulation of the Problem and Kinetic Equations

Model equation, that takes into account the rotational degrees of freedom, describes the evolution of the distribution function $f(t, \mathbf{r}, \boldsymbol{\xi}, I_r)$ that depends on the coordinate vector $\mathbf{r} = (x, y, z)$, velocity vector $\boldsymbol{\xi}$, time t , and the continuous variable I_r corresponding to the rotational energy.

The macroparameters of the gas (particle number density n , velocity vector \mathbf{u} , and temperatures T_{tr}, T_{rot}) are determined using the intrinsic molecular velocity $\mathbf{C} = \boldsymbol{\xi} - \mathbf{u}$ and the notation

$$\langle\langle f \rangle\rangle = \int_{R^3 \times R^+} f(t, \mathbf{r}, \boldsymbol{\xi}, I_r) d\boldsymbol{\xi} dI_r$$

as follows (the lower index tr corresponds to translational variables and rot to rotational ones):

$$n = \langle\langle f \rangle\rangle, \quad n\mathbf{u} = \langle\langle \boldsymbol{\xi} f \rangle\rangle, \quad 3nk_B T_{tr} = \langle\langle m\mathbf{C}^2 f \rangle\rangle, \quad (k_{rot}/2)nk_B T_{rot} = \langle\langle I_r f \rangle\rangle,$$

where m is the molecular mass, k_B is the Boltzmann constant, and k_{rot} is the number of rotational degrees of freedom. The translational pressure p_{tr} and the equilibrium temperature T_{eqr} , which is set due to the exchange of translational and rotational energies, are determined according to the formulas

$$p_{tr} = nk_B T_{tr}, \quad T_{eqr} = \frac{3T_{tr} + k_{rot}T_{rot}}{3 + k_{rot}}.$$

In this study, the calculations for a polyatomic gas are based on the generalized form of BGK equation [9], which takes into account the internal energies. This model is a special case of the Rykov equation [9] when the rotational and translational heat fluxes are equal to zero. In this approach, the collision integral is approximated by the sum of two relaxation terms which correspond to the elastic and inelastic collisions.

By averaging the VDF over internal energies and integrating over the variable I_r with weight coefficients $+1, I_r$,

$$f_0 = \int f dI_r, \quad f_1 = \int I_r f dI_r,$$

we reduce the kinetic equation to the system of two model equations,

$$\frac{\partial f_j}{\partial t} + \left(\boldsymbol{\xi}, \frac{\partial f_j}{\partial \mathbf{r}} \right) = \nu_{tr}(n f_j^{tr} - f_j) + \nu_{rot}(n f_j^{rot} - f_j), \quad j = 0, 1.$$

Here

$$f_0^{tr} = f_M(T_{tr}), \quad f_0^{rot} = f_M(T_{eqr}), \quad f_1^{tr} = 0.5 k_{rot} k_B T_{rot} f_0^{tr}, \quad f_1^{rot} = 0.5 k_{rot} k_B T_{eqr} f_0^{rot},$$

$$f_M(T) = \left(\frac{m}{2\pi k_B T} \right)^{\frac{3}{2}} \exp \left(-\frac{m \mathbf{C}^2}{2k_B T} \right), \quad \nu_{rot} = \frac{\nu}{Z_r}, \quad \nu_{tr} = \nu \left(1 - \frac{1}{Z_r} \right),$$

where Z_r is the rotational collision number, $\nu = p_{tr}/\mu_{tr}$ is the collision frequency, and μ_{tr} is the viscosity coefficient, which depends on the translational temperature. The obtained system of model equations satisfies the conservation laws of density, momentum, and total energy.

2. Details of Calculations

The calculations use dimensionless quantities by introducing characteristic values of the evaporation spot radius R , surface temperature T_s , number density n_s , the most probable molecular velocity of the evaporating gas $v_m = \sqrt{2RT_s}$, and the time scale $t_0 = R/v_m$. The hard spheres intermolecular interactions are assumed.

The introduced characteristic quantities lead to a dimensionless form of the kinetic equation with the Knudsen number $Kn = \lambda_s/R$, where $\lambda_s = 1/(n_s \sigma_T \sqrt{2})$ is the mean free path and σ_T is the collision cross section. Assuming that the particle flux $\Psi_{VAP} = n_s u_t/4$ is constant, where $u_t = 2v_m/\sqrt{\pi}$ is the thermal molecular velocity, the Knudsen number can be related to the number of evaporated monolayers $\Theta = N/N_0$, where $N = \tau_{imp} \Psi_{VAP} \pi R^2$ is the total number of molecules evaporated from the surface, N_0 is the number of molecules in one layer, and $\tau_{imp} = \tau t_0$ is the laser pulse duration. Then $Kn = \tau \sqrt{2/\pi}/16/\Theta$ [5].

In the general case, the rotational collision number Z_r depends on the temperature, but for the considered range of the evaporation parameters as indicated in [7], the value of Z_r weakly affects the vapor cloud dynamics, so $Z_r = 4$.

3. Results

The problem of particle evaporation from a surface with expansion into vacuum is considered in one dimensional geometry. The VDF $f(x=0, \xi_x > 0, t)$ at $t < \tau$ is Maxwellian with particle number density $n_1 = 1$, temperature $T_1 = 1$, and $\mathbf{u}_1 = 0$ (in dimensionless form). For $t > \tau$, $f(x=0, \xi_x > 0, t) = 0$, since full absorption is assumed. The initial value of the VDF for the vacuum simulation $f(x > 0, \xi_x, t)$ is Maxwellian with parameters $n_0 = 1.e^{-12}$, $T_0 = 300K/T_s$, $\mathbf{u}_0 = 0$.

To study the effect of internal energy on the vapor-gas cloud dynamics, a series of calculations with different numbers of evaporated monolayers $\Theta = 1, 10, 100$ ($Kn = 0.04986\tau/\Theta$) and different numbers of rotational degrees of freedom are carried out. The numerical solution is

found by the discrete ordinate method with a finite volume TVD scheme. At the beginning of the calculations, the cell size near the target for $\Theta \leq 10$ is chosen of the order of the mean free path. For $\Theta = 100$ the cell size is larger than mean free path, but its reducing does not lead to noticeable changes in the solution. The boundary of the computational physical domain is defined by the estimate of the position cloud boundary at time $t = 1000\tau$.

The calculations are carried out by two software codes. The first of them is Unified Flow Solver (UFS) [10] in which the distribution functions are reduced to two-dimensional functions in the velocity space. This code is used for $\Theta = 100$, when high refine level necessary for Knudsen layer and the “ray effect” due to the intense relaxation of VDF to equilibrium is weak. The stationary non-uniform velocity grid with fine meshes near zero velocity can be used.

At the beginning of the calculations, the high level of grid refinement is implemented in the boxes closest to the $x = 0$. In other boxes, the refinement level is set to zero. At time moments $t > \tau$, the grid is changed using the adaptive mesh refinement algorithm according to predefined patterns, which take into account the cloud move and the increasing Knudsen number. In this case, the calculation can utilize about 10^8 cells in the phase space and efficient parallel implementation is essential. The parallel implementation of the UFS is single-level and uses the decomposition in the physical space with optimal balance between processors. The calculations were run in Joint Supercomputing Center of RAS using up to 1024 processors.

For $\Theta = 10$ and $\Theta = 1$, another kinetic module with the reduction to a one-dimensional distribution functions is used. This 1D code tests the algorithm of velocity dynamic adaptation to suppress “ray effect”. At the beginning of calculations, velocity intervals (boxes) of an arbitrary size are given, and a uniform partition is introduced in each of them with the increasing refinement level in the neighborhood of zero velocity. The grid is changed by doubling the number of cells in the velocity interval at given times if $2h_v/\sqrt{T} > C$ (where C is a given constant and h_v is a velocity step in the corresponding interval). The mapping of the distribution function to a new grid is carried out with the density conservation.

Calculations show a significant change in the behavior of macroparameters. Thus, during impulse expansion of the polyatomic gas, intermolecular collisions lead to intensive energy transfer from internal degrees of freedom to translational ones. Increasing translational temperature leads to a significant acceleration of the cloud. Figure 1 shows comparison of numerical results obtained by DSMC [7] and the kinetic equation for different number of rotational degrees of freedom at time moment $t = 5\tau$. Figure 1 shows that the profiles of macroparameters are very close.

The effect of internal energy on the cloud dynamics over a large time interval can be determined from the time dependence of the average values of gas parameters (translational temperature, velocity, or density). For this purpose, the mean value \bar{F} is defined as

$$\bar{F} = \int nFdx / \int ndx.$$

The average temperatures and velocities for monoatomic gas and for $k_{rot} = 3$ are given in Fig. 2, which shows that the results obtained by the model equations and by the DSMC practically coincide.

The effect of the number of evaporated monolayers on the vapor-gas cloud dynamics is presented in Fig. 3 for $k_{rot} = 3$ in time moments $t = 100\tau$. Figure 3 shows that as the number of monolayers decreases, the energy exchange becomes weaker and the difference between the rotational and translational temperatures increases.

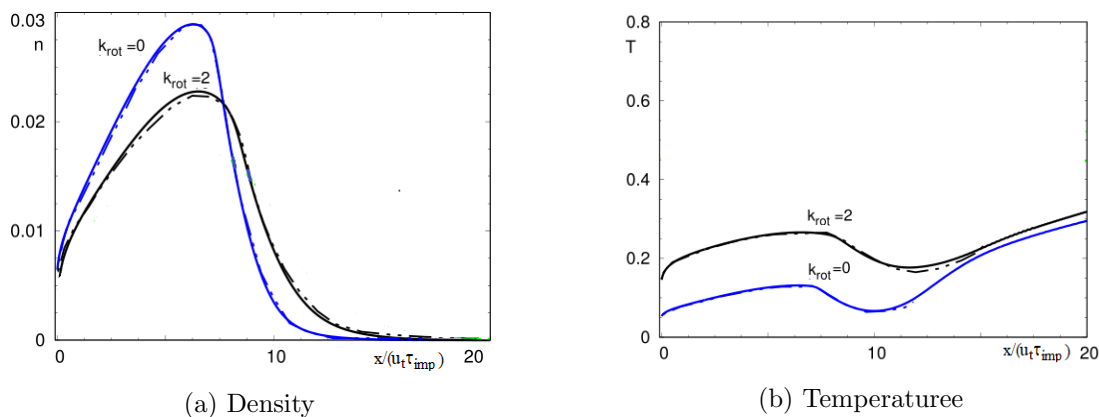


Figure 1. Density and temperature at time moment $t = 5\tau$ and $\Theta = 100$, solid lines correspond to model equations, dashed lines to DSMC

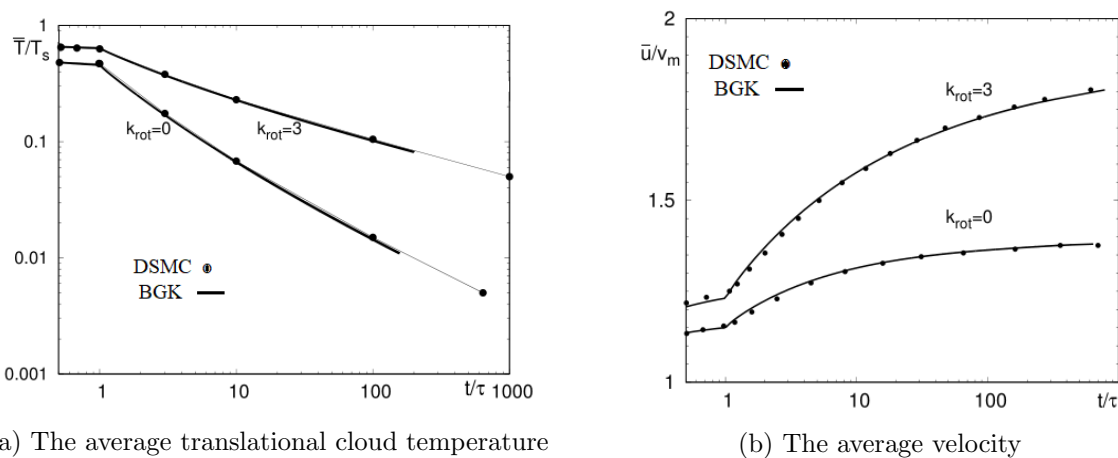


Figure 2. Influence of the number of rotational degrees of freedom on the temporal evolution of the average temperature and velocity

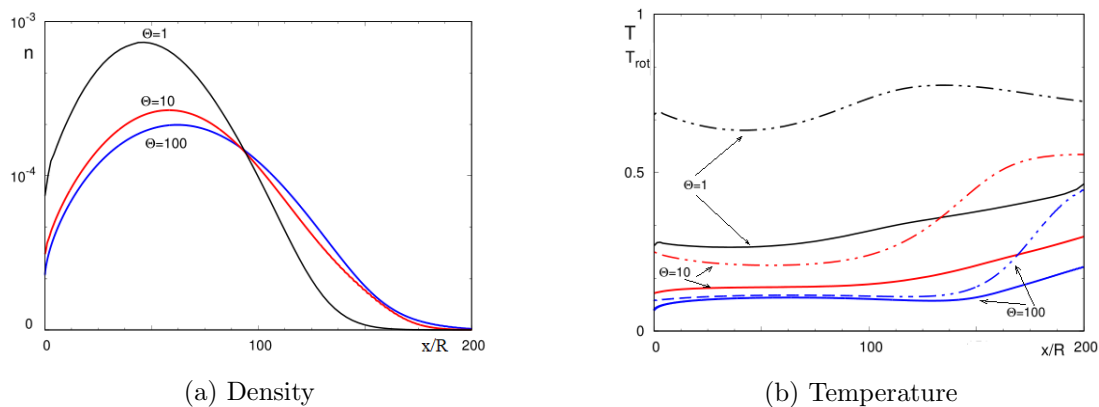


Figure 3. Density and temperature at moment time $t = 100\tau$ for different numbers of evaporated monolayers and for $k_{rot} = 3$, solid lines on Fig. 3b translation temperature, dashed lines rotational temperature

Conclusion

Calculations of evaporation of molecular gas into vacuum induced by a nanosecond laser pulse using model kinetic equations, that take into account the rotational degrees of freedom, are

carried out. These calculations describe the experiment more correctly, because they correspond to a more complete formulation of the problem.

A comparison of the molecular vapor cloud dynamics with the case of a monoatomic gas shows an increase in the forward temperature and velocity of the gas and comparison of the results obtained using the DSMC and the integration of model kinetic equations demonstrates a very close behavior of the macroparameters that makes it possible to use an alternative approach, based on model equations and to study the complex physical process of laser ablation.


Calculations were carried out on the resources of the Joint Supercomputer Center of the Russian Academy of Sciences.

This paper is distributed under the terms of the Creative Commons Attribution-Non Commercial 3.0 License which permits non-commercial use, reproduction and distribution of the work without further permission provided the original work is properly cited.

References

1. Bykov, N.Y., Bulgakova, N.M., Bulgakov, A.V., Loukianov, G.A.: Pulsed laser ablation of metals in vacuum: DSMC study versus experiment. *Appl. Phys. A* 79, 1097–1100 (2004). <https://doi.org/10.1007/s00339-004-2654-6>
2. Morozov, A.A., Mironova, M.L.: Numerical analysis of time-of-flight distributions of neutral particles for pulsed laser ablation of binary substances into vacuum. *Appl. Phys. A* 123(12), article 783 (2017). <https://doi.org/10.1007/s00339-017-1400-9>
3. Morozov, A.A.: Analysis of time-of-flight distributions under pulsed laser ablation in vacuum based on the DSMC calculations. *Appl. Phys. A* 111(4), 1107–1111 (2013). <https://doi.org/10.1007/s00339-012-7325-4>
4. Itina, T.E., Hermann, J., Delaporte, P., Sentis, M.: Laser-generated plasma plume expansion: Combined continuous-microscopic modeling. *Physical Review E* 66, 066406 (2002). <https://doi.org/10.1103/PhysRevE.66.066406>
5. Morozov, A.A., Frolova, A.A., Titarev, V.A.: On different kinetic approaches for computing planar gas expansion under pulsed evaporation into vacuum. *Phys. Fluids* (2020). <https://doi.org/10.1063/5.0028850>
6. Titarev, V.A., Morozov, A.A.: Arbitrary Lagrangian-Eulerian discrete velocity method with application to laser-induced plume expansion. *Applied Mathematics and Computation* 429, 127241 (2022). <https://doi.org/10.1016/j.amc.2022.127241>
7. Morozov, A.A.: Dynamics of pulsed expansion of polyatomic gas cloud: Internal translational energy transfer contribution. *Phys. Fluids* 19(8), 087101 (2007). <https://doi.org/10.1063/1.2754347>
8. Morozov, A.A.: Analytical model for polyatomic gas expansion under pulsed. *Physics of Fluids* 20, 027103 (2008). <http://doi.org/10.1063/1.2841624>
9. Titarev, V.A., Frolova, A.A.: Application of Model Kinetic Equations to Calculations of Super- and Hypersonic Molecular Gas Flows. *Fluid Dynamics* 53(4), 536–551 (2018). <https://doi.org/10.1134/S0015462818040110>
10. Kolobov, V.I., Arslanbekov, R.R., Aristov, V.V., *et al.*: Unified Solver for Rarified and Continuum Flows with Adaptive Mesh and Algorithm Refinement. *J. Comp. Phys.* 223, 589–608 (2007). <https://doi.org/10.1016/j.jcp.2006.09.021>

Mathematical Modeling of Detonation Initiation in the Channel with a Profiled End Using Parallel Computations

Alexander I. Lopato^{1,2} 

© The Author 2023. This paper is published with open access at SuperFri.org

The paper is devoted to a numerical study of detonation initiation in the gas mixture in the channel with a profiled end. Initiation occurs as a result of the reflection of the shock wave of relatively low intensity from the end of the channel. Numerical calculations are carried out using unstructured triangular grids. The numerical algorithm is parallelized by the computational domain decomposition method using the METIS library. The exchange of grid function values between computing cores is performed using the MPI library. Numerical calculations are conducted on grids with different numbers of triangular cells. Detonation initiation patterns are obtained, which correspond to each other. The differences are mainly related to the degree of resolution of the elements of the gas mixture flow in the channel.

Keywords: detonation initiation, unstructured triangular grids, parallel computations, Kelvin–Helmholtz instability.

Introduction

Mathematical modeling of gasdynamic processes with chemical reactions, shock waves (SWs) and detonation waves (DWs) is of great interest from the point of view of fundamental and applied sciences. Scales of physical quantities and chemical reactions can significantly vary in space during the study of the process. In addition, gasdynamic processes with DWs have a complex nonlinear nature (see, for example, [1]), the comprehensive research of which in experiments is often associated with some difficulties. One of such problems is the study of the process of suppression of DWs that occur in tunnels and mines. Another, quite important on an industrial field, is the problem of creating a detonation engine based on rotating detonation in an annular channel. The issues of detonation initiation in channels are also of interest.

In this work, the initiation of a DW in a gas mixture under reflection of a SW from the curved end of the channel is considered. The SW propagation along the channel is accompanied by its interaction with the channel walls with the formation of new waves. The interaction of waves with each other and with the channel walls leads to the formation of so-called hot spots that are regions of high pressure and temperature (see, for example, [2]). In this case, hot spots become initiators of the mixture.

The problems noted above show that the study of the processes of initiation, propagation, and suppression of DWs can be carried out in a domain of complex geometry and with curved boundaries. In such cases, the domain is typically discretized using unstructured grids. Indeed, as noted in [3], it is impossible to generate automatically block-structured grids on arbitrary geometries. However, calculations on unstructured grids with a large number of cells are usually quite demanding on computer resources. One of the solutions is the use of parallel computational technologies, which makes it possible to carry out calculations in various studies in adequate periods of time.

The aim of the work is mathematical modeling of detonation initiation as a result of shock wave reflection from the profiled end of the channel using parallel computations.

¹Institute for Computer Aided Design RAS, Moscow, Russian Federation

²Institute for Computer Science and Mathematical Modeling, I.M. Sechenov First Moscow State Medical University (Sechenov University), Moscow, Russian Federation

The article is organized as follows. Section 1 provides the statement of the problem and the mathematical model. Section 2 is devoted to the description of the numerical algorithm. Section 3 contains the results obtained in the calculations. Conclusion summarizes the study.

1. Problem Statement and Mathematical Model

The plane channel with the end consisting of three sections of a plane wall and two semi-ellipses arranged symmetrically relative to the axis of the channel is considered. The channel is filled with a model hydrogen-oxygen mixture. The pressure in the channel is 0.04 atm, the temperature is 298 K. The plane SW with a Mach number of 2.5 propagates towards the end of the channel. Taking into account the problem statement and the symmetry of the channel, we consider half of the channel to reduce the number of calculations. The geometry of the computational domain, as well as the boundary conditions, are shown in Fig. 1. Note that the geometry and the statement correspond to the data from the experimental work [1].

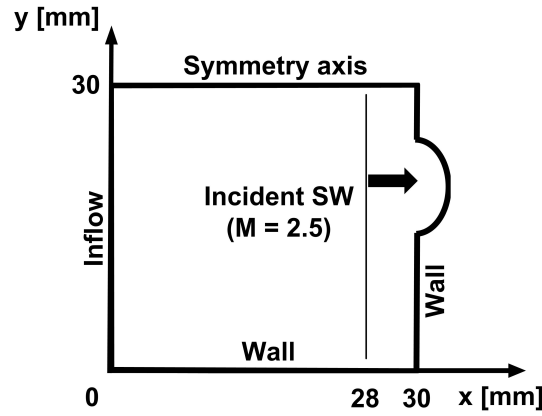


Figure 1. Schematic of the computational domain

The mathematical model is based on the 2D Euler equations written in the Cartesian frame (x, y) and supplemented by one-stage chemical kinetics model. The system of governing equations has the following view:

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}}{\partial x} + \frac{\partial \mathbf{G}}{\partial y} = \mathbf{S},$$

$$\mathbf{U} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ e \\ \rho Z \end{bmatrix}, \mathbf{F} = \begin{bmatrix} \rho u \\ p + \rho u^2 \\ \rho uv \\ (p + e)u \\ \rho Zu \end{bmatrix}, \mathbf{G} = \begin{bmatrix} \rho v \\ \rho uv \\ p + \rho v^2 \\ (p + e)v \\ \rho Zv \end{bmatrix}, \mathbf{S} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \rho \omega \end{bmatrix}, \quad (1)$$

$$e = \frac{\rho}{2}(u^2 + v^2) + \rho \varepsilon, \quad \varepsilon = \frac{p}{\rho(\gamma - 1)} + ZQ, \quad p = \frac{\rho}{\mu}RT, \quad \omega = -A\rho Z \exp\left(-\frac{E}{RT}\right).$$

Here, t is the time, ρ is the density of the gas mixture, u and v are the components of the gas velocity, p is the pressure, T is the temperature, e is the total energy per unit the unit of the volume, Z is the mass fracture of the reactive component of the mixture, ω is the rate of the chemical reaction, ε is the specific internal energy of the gas, Q is the heat release of the chemical reaction, μ is the molar mass of the gas, R is the universal gas constant, A is the pre-exponent

factor. The gas is considered to be ideal with the constant specific heat ration γ . The following values of the parameters are used:

$$\gamma = 1.23, \mu = 12 \frac{\text{g}}{\text{mole}}, Q = 7.37 \frac{\text{MJ}}{\text{kg}}, E = 76.2 \frac{\text{kJ}}{\text{mole}}, A = 9.16 \times 10^8 \frac{\text{m}^3}{\text{kg s}}.$$

The parameters γ , Q , E and A are taken from the database [4]. The parameter values correspond to a pressure of 0.2 atm, which is equal to the pressure behind the SW reflected from the plane wall for the considered Mach number value and the initial pressure value of 0.04 atm.

2. Numerical Algorithm

The main feature of the computational technique is the use of completely unstructured numerical grids with triangular cells. A Delaunay triangulation is carried out to construct the grids. The computational algorithm is based on the technique of splitting physical processes. First, the gasdynamic equations are integrated on a time step. Then, the chemical kinetics equations are integrated on the time step using parameters obtained on the previous stage and not taking into account the effect of convection. On the gasdynamic stage, spatial discretization is conducted using the finite volume method. The numerical flux is calculated using the AUSM scheme. The reconstruction of grid functions with limiter minmod is applied. The numerical method employs a large stencil on triangular grids suitable for the second order of the accuracy. Time integration is carried out using a second order explicit Runge–Kutta scheme with two stages. On the second stage the system of 2 ordinary differential equations is solved numerically to find the values of gas temperature and mass fracture of the gas reactive component. The serial computational algorithm is described in more detail in [5].

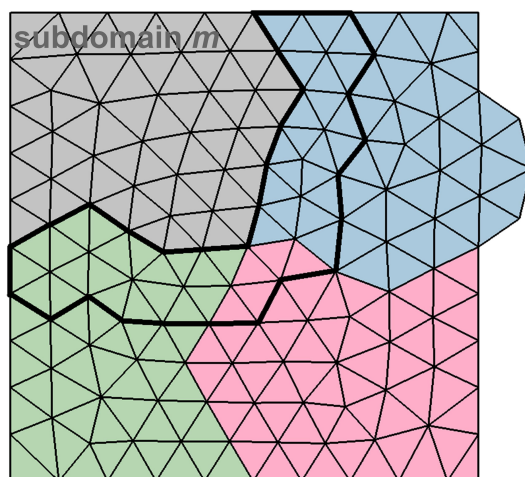


Figure 2. Example of decomposition of the computational domain into 4 subdomains. The bold line indicates the boundary of the halo-cells corresponding to the subdomain m

The computational algorithm is parallelized by the method of numerical grid decomposition. The realization of the parallel algorithm is carried out using the Message Passing Interface (MPI) technology. The computational domain is divided into subdomains using the METIS library. The number of subdomains is equal to the number of computing cores used in the computations. Thus, the computing core with the number m corresponds to the values of gasdynamic parameters in cells from the subdomain m , as well as the values of the parameters in the halo-cells [6] of the subdomain m . An example of the decomposition of the computational domain

into 4 subdomains is shown in Fig. 2. The bold line indicates the boundary of the halo-cells corresponding to the subdomain m . The set of halo cells for each subdomain is determined by the location of the boundaries of the subdomains and the algorithm of gasdynamic parameters calculation. At each time step, parameter values in halo-cells are exchanged between computing cores. The exchange of values is carried out using standard methods of the MPI library. The MPI_Send and MPI_Recv blocking functions are quite effective in cases of such decompositions when a subdomain has one or two adjacent subdomains. In this work, the number of adjacent subdomains can be more than two, as shown in Fig. 2. So, we use the MPI_Alltoallv function that sends data from all to all computing cores. Then, the calculation of gasdynamic parameters in each subdomain is carried out using the actual values of the parameters in halo-cells. The computations are performed on the supercomputer MVS-10P (Joint Supercomputer Center of RAS).

During the computations, the distributions of gasdynamic parameters in the subdomain m are periodically written to a CGNS file with a name corresponding to the computing core with the number m . Visualization of the spatial distributions of gasdynamic flow parameters is carried out using the open source tool Visit based on the Visualization ToolKit (VTK) library.

3. Results

A series of calculations of the detonation problem for a fixed numerical grid and a different number of computing cores is carried out. The numerical grid consists of 6 mln cells. The dependence of the parallel speedup on the number of cores is shown in Fig. 3. The deviation of the obtained dependence from the linear one is due to the following fact. The greater the number of computing cores, the greater the number of halo-cells. Moreover, in the case of a multidimensional computational domain, the number of halo-cells for various cores can be different. Thus, an increase in the number of cores can lead to the fact that the percentage of time associated with data exchange between cores will grow, and the percentage of time associated with calculations will fall due to a decrease in the number of triangular cells per core.

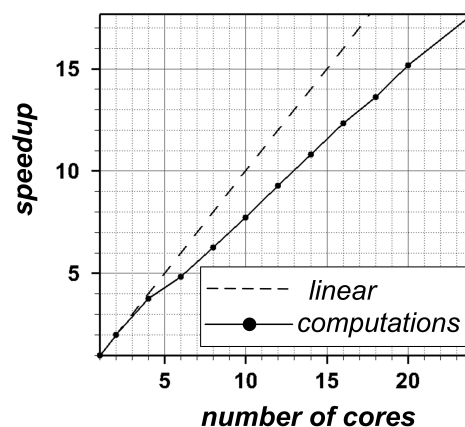


Figure 3. Parallel speedup on supercomputer MVS-10P

Let us consider the process of detonation wave initiation in the case of a numerical grid consisting of approximately 28 million triangular cells. The reflection of the incident plane SW from the end of the wall is associated with the formation of a series of waves that interact with each other and with the walls of the channel. The presence of vertical walls leads to an

amplification in the gasdynamic parameters inside the elliptic cavities. Ignition of the mixture occurs in the region of focusing of the flow at about $20 \mu s$. During the combustion of the mixture, the formation of combustion waves propagating from the ignition region occurs. Detonation initiation occurs at about $41 \mu s$ in two regions on the symmetry axis of the channel outside the elliptic cavities. Figure 4a shows the spatial distribution of temperature (lower half of the picture) and density gradients (upper half of the picture) at $41 \mu s$. An important role in the considered process of detonation initiation is played by the interference of combustion zones and waves formed after the reflections of the incident SW from the profiled end wall. The presence of regions with distributed elliptic surfaces leads to a decrease in the detonation initiation time compared to the case when the rectangular channel without a profiled end is considered. This statement is in agreement with both experimental [1] and numerical [5] works.

A series of calculations of detonation initiation in the channel with the proposed geometry is carried out. Numerical grids with different numbers of triangular cells are considered: 1.2 mln, 6 mln, 18 mln and 28 mln cells. The spatial distribution of temperature and density gradients on the grid consisting of 1.2 mln cells at $41 \mu s$ are shown in Fig. 4b. The results obtained using different grids correspond to each other. In particular, the initiation of the mixture in all calculations occurs at about $41 \mu s$. However, the detailed resolution of elements and structures such as the Kelvin–Helmholtz instability formed during the gas flow is observed on the most detailed grids. On the detailed grids the thickness of SWs becomes smaller that is associated with a decrease in the effect of numerical smearing in calculations.

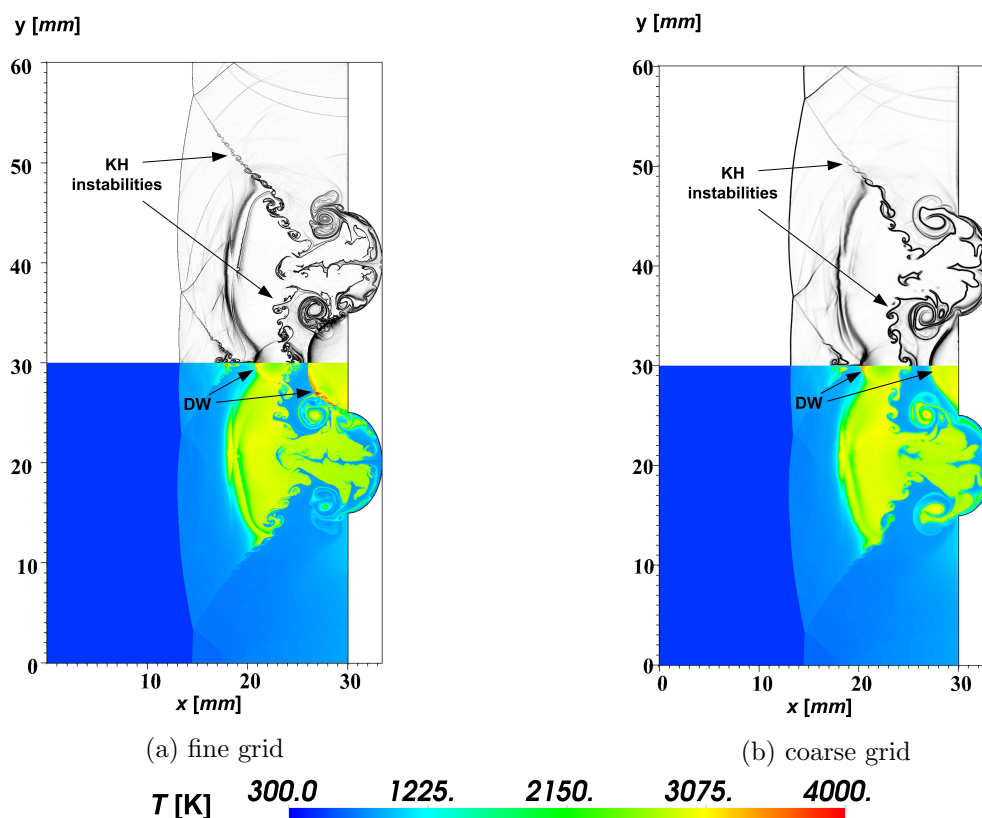


Figure 4. Predicted numerical schlieren images (upper half) and temperature distributions (lower half) at the time instant $41 \mu s$

Conclusions

A parallel computational algorithm for calculating gasdynamic flows with detonation waves on unstructured triangular grids is proposed. Parallelization of the computational algorithm is conducted by the domain decomposition method using the METIS library. Data exchange between computing cores is carried out using MPI library functions. The acceleration of the realized computational algorithm in the problem of detonation initiation in a model hydrogen-oxygen mixture is demonstrated. Calculations of detonation initiation on several numerical grids with different numbers of triangular cells are carried out. The results show that the detonation initiation patterns obtained using different grids correspond to each other. Detailed resolution of flow structures, including Kelvin–Helmholtz instabilities, is observed on computational grids with a large number of triangular cells.

Acknowledgements

The work is carried out under the state task of the ICAD RAS.

This paper is distributed under the terms of the Creative Commons Attribution-Non Commercial 3.0 License which permits non-commercial use, reproduction and distribution of the work without further permission provided the original work is properly cited.

References

1. Vasilev, A.A.: Cellular structures of a multifront detonation wave and initiation (review). *Combust. Explos. Shock Waves* 51(1), 1–20 (2015). <https://doi.org/10.1134/S0010508215010013>
2. Lopato, A.I.: Numerical simulation of shock-to-detonation transitions using one-stage and detailed chemical kinetics mechanism. In: Favorskaya, M.N., Favorskaya, A.V., Petrov, I.B., Lakhmi, C.J. (eds.) *Smart Modeling for Engineering Systems*, pp. 79–88. Springer (2021). <https://doi.org/10.1007/978-981-33-4619-2>
3. Hirsch, C.: *Numerical computation of internal and external flows: the fundamentals of computational fluid dynamics*. Elsevier, Oxford (2007). <https://doi.org/10.1016/B978-0-7506-6594-0.X5037-1>
4. Schultz, E., Shepherd, J.: Validation of detailed reaction mechanisms for detonation simulation. Caltech Explosion Dynamics Lab, Report No. FM99-5 (2000).
5. Utkin, P.S., Lopato, A.I., Vasil'ev, A.A.: Mechanisms of detonation initiation in multi-focusing systems. *Shock Waves* 30(4), 741–753 (2020). <https://doi.org/10.1007/s00193-020-00969-6>
6. Besnard, J., Malony, A., Shende, S., *et al.*: An MPI halo-cell implementation for zero-copy abstraction. In: *EuroMPI 15: Proceedings of the 22nd European MPI Users Group Meeting*, pp. 1–9. ACM Press (2015). <https://doi.org/10.1145/2802658.2802669>

Overview of SCM Coupler and Its Application for Constructing Climate Models

Viacheslav S. Gradov¹ , Gennady A. Platov¹ 

© The Authors 2023. This paper is published with open access at SuperFri.org

Numerical modeling is one of the leading research tools for the climate problems. Among numerical models, researchers use, in particular, coupled models, that are numerical models describing more than one climate component dynamics and their interactions. The simulation results with such models depend on the way how these interactions are configured. Therefore, a proper configuration of the exchanges is crucial. The software called a “coupler” is often used to configure these interactions. The coupler is most helpful if the model components are independent of each other modules, their number exceeds two, and they have their own computational grid and time integration step. Key functions of the coupler are managing data exchange between models, setting up synchronous interaction between them based on the time integration step, and interpolating data from one model’s computational grid to the other model’s grid. Additional functions can also be implemented, e.g., fluxes computation between model components, data assimilation, working with the file system, etc. The coupler has one crucial feature: if there is a set of different models of climate system components, one can construct new coupled models by coupling various subsets of these components with coupler. This paper gives an overview of the SCM (SibCIOM Coupling Module) coupler we first developed for the model SibCIOM (Siberian Coupled Ice and Ocean Model). The description of this coupler has not been published before. The SCM coupler is a separate module to which the main climate system model component, such as atmospheric, oceanic, sea ice and land components, can be attached. Additional functions of this coupler include computation of atmosphere-to-ocean and atmosphere-to-ice fluxes and ocean and sea ice state correction using a tidal model. This paper also gives examples of two models constructed with the SCM coupler.

Keywords: coupler, climate modeling, numerical modeling, coupled models.

Introduction

Numerical modeling is one of the main tools in climate research nowadays. There are many numerical models of different complexity, from simple null or one-dimensional models to ensembles of coupled three-dimensional models. Every model type is used for various aims depending on the investigated problem. For example, simpler models can be used to estimate the influence of a specific physical process on the climate, to analyze particular climate drivers or to estimate some parameters needed for the more complex models to work correctly [9, 19, 25]. More complex models are often used to describe the dynamics and interactions of climate system components in detail [33]. Coupled models also describe some climate system components, such as coupled ocean and sea ice models, e.g., FEMA0 [30], SibCIOM [14, 15], coupled atmosphere and land models [16], etc.

All components of most coupled models are usually independent and interact, or coupled, with each other in a certain way. Various coupling techniques are possible. A program code of one model component can be nested in the code of another as a subroutine, or the model components can run concurrently and communicate with each other through any interface (MPI, file system, UNIX pipe, etc.). An example of the model that uses both these coupling techniques is the INMCM climate model [35]. It includes the General Atmospheric Circulation Model (GACM) component and the General Ocean Circulation Model (GOCM) component, running concurrently and interacting with each other through MPI. At the same time, the land, vegetation and

¹Institute of Computational Mathematics and Mathematical Geophysics, Novosibirsk, Russian Federation

atmospheric aerosol models are implemented in the GACM component as subroutines, and the ice model is implemented in the GOCM component.

Other coupling techniques are associated with a software called “coupler”. The main functions of the coupler are management of data transfer between model components, execution control based on the calendar and time step of model components, and interpolation of coupling data from the computational grid of one model component to the computational grid of the other component. Additional functions can also be delegated to the coupler, such as flux calculation between model components, working with the file system, data assimilation, etc. The set of these functions may depend on the specific problem.

In general, couplers can be referred to one of three types: stand-alone coupler, integrated coupling framework and coupling library. Figure 1 shows the interaction diagrams of some couplers with model components.

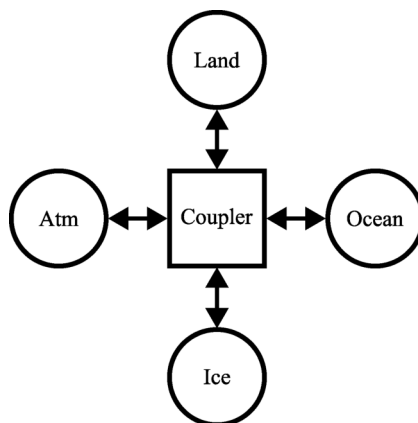
The stand-alone coupler is usually a separate component of the model. In this case, the model components are separate executable modules that run concurrently. Usually, such a coupler is serial and its performance is bottleneck when the array sizes and the number of model components increase, because only the master processes of the model components are involved in interaction with the coupler. The memory usage restrictions arise from this as well. The most advanced examples of such couplers are OASIS3 [34] and older versions of the NCAR CSM Flux coupler (cpl3, cpl4, cpl5), that was used in CCSM model (Fig. 1a).

The integrated coupling framework (e.g. Fig. 1b) is the software that uses component-level interfaces assembling into a single integrated application. This type of system is managed by a high-level driver, and the model component code is usually reorganized as a sequence of method calls, such as initialize, run and finalize, etc. Many systems of this type are usually designed for a predefined set of components, and any changes may require complicated work on the coupler and model component codes. However, the ability of such systems to execute model components sequentially, concurrently or in a mixed mode may give additional opportunities for performance optimization. The most advanced systems of this type are ESMF [6], NCAR CSM Flux Coupler (cpl7) [8], and GFDL FMS [2].

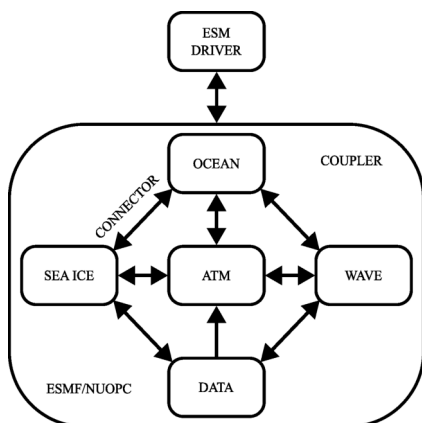
It is worth noting that such systems may have the coupler as a separate module, keeping only a part of the coupler functions, e.g. interpolation or data transfer, and giving other functions, e.g. execution control, to a high-level driver. The NCAR CSM Flux coupler is one of such couplers. Starting with the cpl6 version, this type of coupler works in a parallel mode. There are also couplers which can additionally use abstract interfaces, that enable to couple any number of models with different characteristics. One of them is the CMF (Compact Modeling Framework) coupler [21].

The coupling library (Fig. 1c) is a set of coupling tools that are plugged into each model component as a library. In this case, there is no stand-alone coupler component. Also, a high-level driver is not required in this case. The coupler functions are performed concurrently by a subset of the cores of each of the model components, resulting in improved performance. However, as the number of model components increases, it becomes harder to manage and debug the model code. The OASIS3-MCT [7] and MCT [20, 24] are the most advanced couplers of this type.

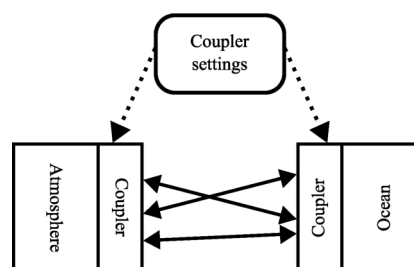
There is no “one-size-fits-all” coupler that is optimal for all coupled models. There is a wide range of couplers that may be suitable for a particular purposes, but not to the user, like such couplers as black box. Nevertheless, it can be convenient to create unique coupler that performs all the additional functions the user needs. In addition, having specific software solution allows



(a) Stand-alone sequential coupler (CCSM model)



(b) Integrated coupling framework (ESMF/NUOPC coupling framework)



(c) Coupling library (OASIS3-MCT)

Figure 1. Interaction diagrams of model components with coupler for different coupler types

to fully control its versions and make the necessary changes without any communication with coupler holder. Thus, we decided to develop the SCM (SibCIOM Coupling Module) coupler.

The development of the SCM coupler began in the early 2000s, when it was planned to couple the ocean model developed at ICMMG SB RAS [14, 15] and any sea ice model for Arctic ocean simulations. The CICE model [4, 10, 18, 27], which, at that time, was part of the CCSM (Community Climate System Model), was taken as the sea-ice model. Interaction between the components of this model was performed using the NCAR CSM Flux coupler (cpl4) [23]. Therefore, to avoid modifications in the CICE model, the concept of interaction protocols with the NCAR coupler was taken and the SCM coupler was developed based on it.

Since this coupler accounted for the features of our ocean model and the CICE model, the first version of the coupled sea ice and ocean model SibCIOM (Siberian Coupled Ice-Ocean Model) was developed on its basis. Later, atmosphere and land components were added to the model as pure model components, i.e. they transfer NCEP/NCAR reanalysis data [22] through the coupler to ocean and sea-ice components.

Note that the SCM coupler is designed for coarse-resolution models, in contrast to many modern couplers designed for use on high and ultra-high-resolution models on massively parallel supercomputers. At the time of its development, we were not interested in performance issues. Nevertheless, some changes were made over time, which allowed us to slightly reduce the running

time of the coupler within the SibCIOM model. Note that the coupler code was developed from the ground up.

The main purpose of this article is to describe the design of the resulting SCM coupler and its functionality, since its description has never been published. The new coupled climate models using this coupler are presented. The SCM is written in FORTRAN90 programming language using MPI. The source code is licensed under the GPL-3.0 license and is available on Github [3].

The article consists of three sections. The first and second sections describe SCM coupler design and functionality, respectively. The third section gives examples of constructing two climate models using this coupler.

1. SCM Coupler Design

To interact with the coupler, the model components must first be prepared correctly. Then, the work of the coupler and model components can be split into three phases: initialization; exchange of configurations and initial data; and synchronized exchange in time cycle.

1.1. Preparation of Model Component

The model being coupled must be either sequential or paralleled using MPI. Note that it is recommended to use the MPI subroutines from [3]: MPI/mpi_tools.F. The first step is to insert an initialization procedure *setup_mpi('model_name')* (see Section 1.2) at the beginning of program and make sure that previous MPI initialization procedures (if they exist) do not override it. Before starting any calculations, the model must receive a one-dimensional integer array of length 100, the first two records of which contain information about the start and end dates of the experiment. This array is then filled one by one with the following data before sending it back to the coupler:

- diagnostic flag (0 – if the model works correctly, 1 – if the model does not work correctly, in which case the whole model will stop);
- the current date of the model component;
- time in seconds elapsed since the beginning of the model's day;
- grid size by longitude;
- grid size by latitude;
- number of model steps per day.

Next, it is necessary to create subroutines for sending calculated model data to the coupler and for receiving data from the coupler, similar to subroutines *send_atmos_data* and *get_atm_state* (see [3]: Coupler_8/coupler.F), for example.

The send subroutine should pack a set of two-dimensional arrays into one two-dimensional array with the number of rows equal to the number of arrays to be sent and send it to the coupler. The rows themselves are the original two-dimensional arrays, flattened into one-dimensional arrays. The receive subroutine must get a two-dimensional array from the coupler and reshape each of its rows back into a two-dimensional array. In this case, it is necessary to ensure the consistency of the transmitted data between the coupler and the model component.

These subroutines must then be used before the main model cycle to send the model grid and initial data to the coupler and receive the necessary data from the other models through the coupler.

1.2. Initialization of Model Components

The subroutine *setup_mpi('model_name')* (see [3]: MPI/mpi_tools.F) should be called at the beginning of the model components and coupler programs. It initializes all model components in global communicator MPI.COMM_WORLD, creates the internal communicators and distribute all processes over them (Fig. 2).

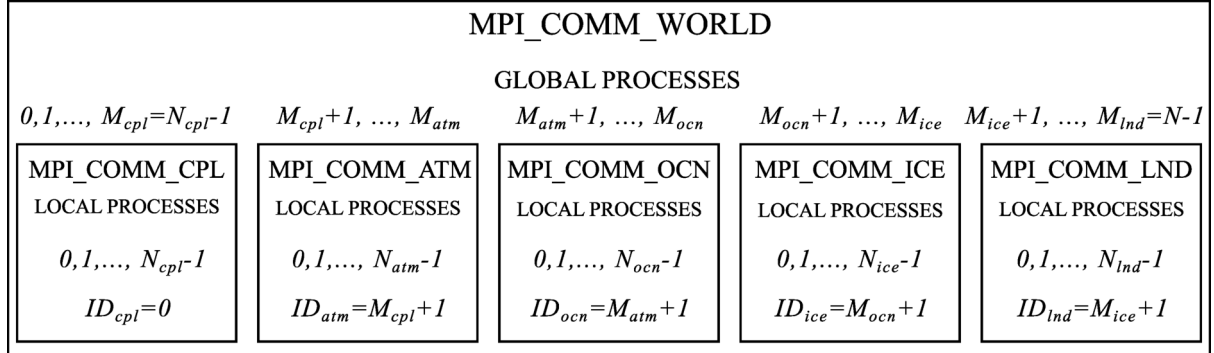


Figure 2. Scheme of the *setup_mpi('model_name')* subroutine result

Each model component process has a global and a local number. Let N be total number of processes used by model, and $N_{cpl}, N_{atm}, N_{ocn}, N_{ice}, N_{lnd}$ are the number of a process used by each model component providing equality $N_{cpl} + N_{atm} + N_{ocn} + N_{ice} + N_{lnd} = N$. These numbers also define the maximum number of local processes for each component. Enumeration of global processes is changed for every program run, therefore for certainty we assume that $M_{cpl}, M_{atm}, M_{ocn}, M_{ice}, M_{lnd}$ are the global numbers of the last process of each component, such that $N_{cpl} - 1 = M_{cpl} < M_{atm} < M_{ocn} < M_{ice} < M_{lnd} = N - 1$. Procedures MPI_GROUP_RANGE_INCL and MPI_COMM_CREATE are used to create the processes groups and individual communicators (MPI_COMM_CPL, MPI_COMM_ATM, etc.) for all model components. Within these communicators, model components can perform their own internal communications. Finally, the master processes for each component are defined as global processes with numbers $ID_{cpl} = 0, ID_{atm} = M_{cpl} + 1, ID_{ocn} = M_{atm} + 1, ID_{ice} = M_{ocn} + 1, ID_{lnd} = M_{ice} + 1$.

1.3. Initial Data and Configuration Exchange

The second phase starts with configuration exchange (Fig. 3). First, the coupler sends the start and end date of the experiment via the *buff* array of size 100 to the model components. Tags *msgtypes_c2ii, msgtypes_c2ai, msgtypes_c2li, msgtypes_c2oi* (see [3]: Coupler_8/msgtypes.F) show that the coupler sends initial data to the ice, atmospheric, land and oceanic components, respectively. Similarly, parameters *msgtypes_i2ci, msgtypes_a2ci, msgtypes_l2ci, msgtypes_o2ci* shows that model components send initial data to coupler. Then coupler receives updated array *buff* and grid configuration from each model. The data structures that the coupler uses will be described below. (see [3]: Coupler_8/struct.F for details)

The first are structures of type “grid” and “state” that store buff array data and grid configuration. They are defined for each model component. A structure of “state” type contains the next fields: the current date *idate*, the time in seconds elapsed since the beginning of the

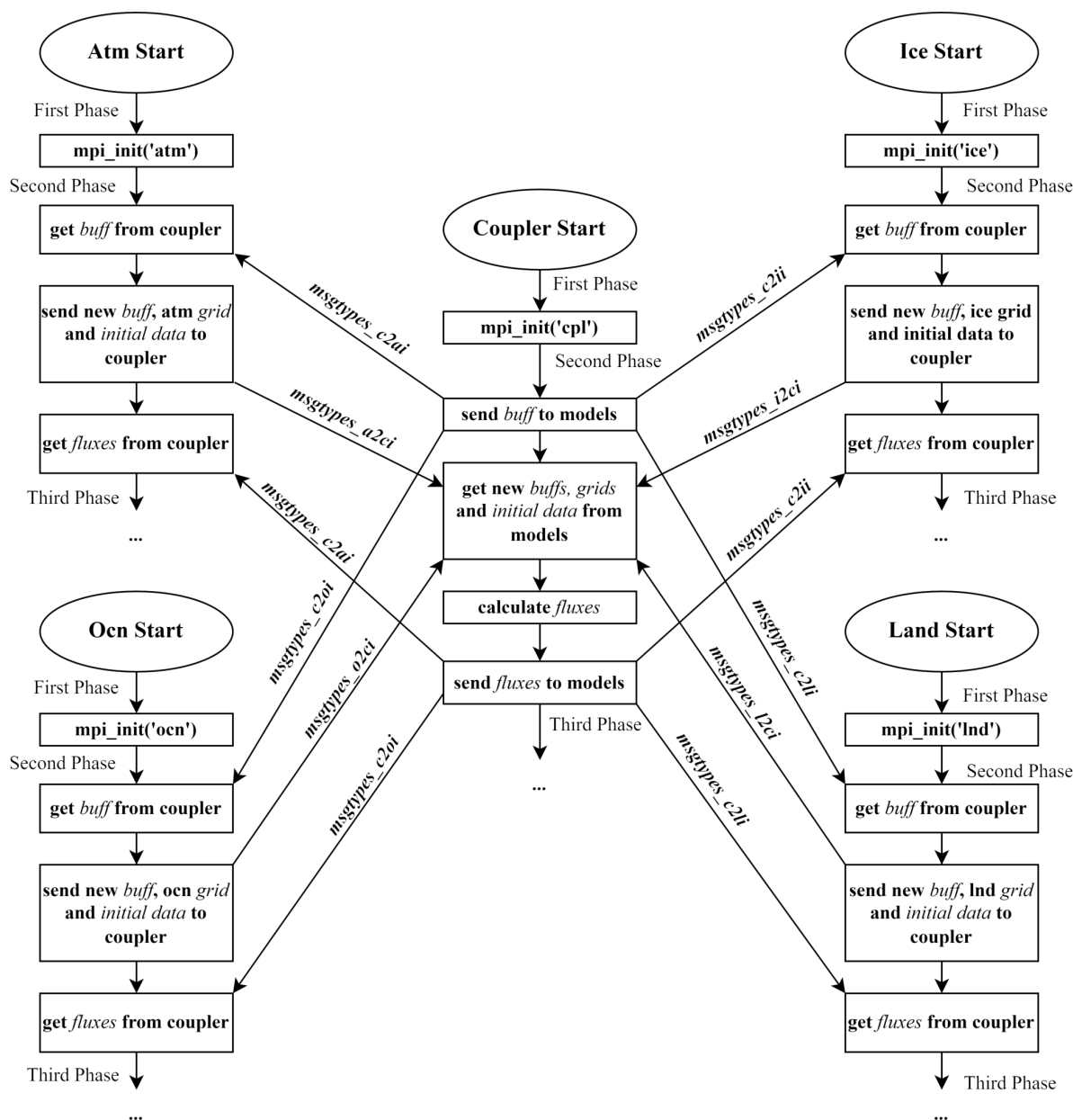


Figure 3. Flow chart diagram of the first and second phases of coupler and model components

model’s day *sec*, the size of the computation grid *imt*, *jmt*, the number of time steps per day *nadv* and the next time moment of model component *next*.

A structure of “grid” type contains the next two-dimensional arrays: geographical coordinates *lat*, *lon*, the area of the box surrounding the grid points *tarea*, domain mask *mask* and an auxiliary array *work* that can store any additional grid data.

The next step is receiving the initial data of each model component. This data is written in data structures of types “from_ocn”, “from_ice”, “from_atm” and “from_lnd”. These structures must be modified manually according to the data being sent by the model components. Fluxes after their computation are stored in structures “to_ice”, “to_ocn”, “to_lnd”, “to_atm” types. The final step of second phase is sending the new data to the model components.

The second phase of any model component includes receiving the coupler configuration, sending its configuration, grid and initial data to the coupler and receiving the necessary data from the other model components (Fig. 3).

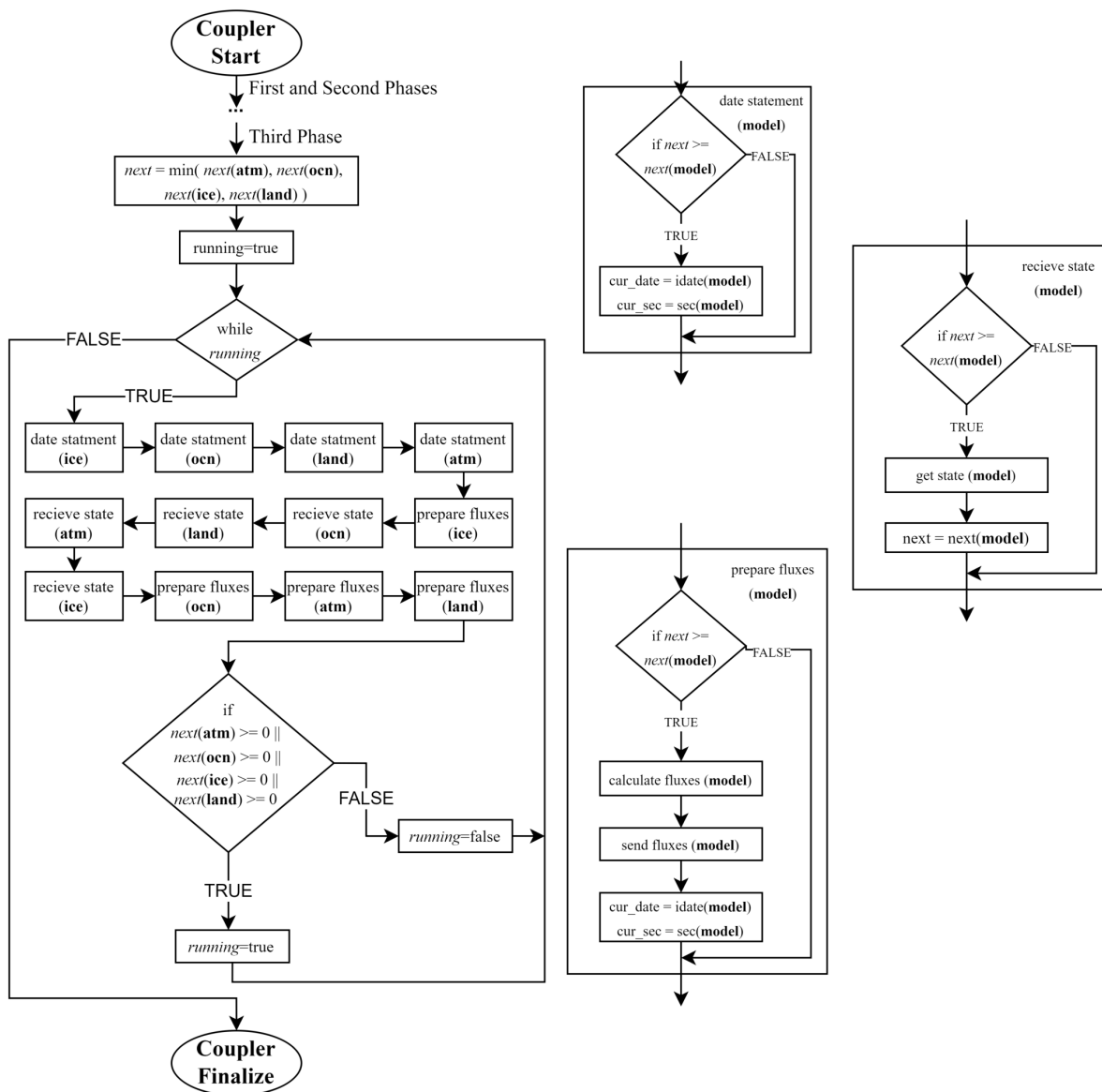


Figure 4. Flow chart diagram of the coupler third phase

1.4. Data Exchange Synchronization

The third phase of coupler includes a “while” loop (Fig. 4). The exit from the loop occurs in two cases: after reaching the final simulation date (the “state” structure field *next* is assumed to be -999 for any model component) or after stopping any of the model components for any reason.

At the same time, the third phase of any model component includes “do” loop where the synchronized data exchange occurs. Moreover, the order of interaction with the coupler remains the same as before the loop (Fig. 3). Within the loop, the *buff* is sent, the current model state is sent, and the necessary data is received from other components.

The coupler knows next time moments $next(\mathbf{atm})$, $next(\mathbf{ice})$, $next(\mathbf{ocn})$, $next(\mathbf{land})$ for each step of the “while” loop. Among all these moments, the minimum moment $next$ is chosen, and further data exchange and flux calculation are performed for model components where this minimum moment is greater or equal to one of the next time moments. One should note that this condition can be satisfied for several components at the same time.

Fluxes are calculated using the data from each model component obtained at different time steps, i.e., if $next \geq next(\mathbf{ice})$, then ice fluxes are calculated using the atmospheric, land and ocean data from the previous time step and ice data received at the current moment.

Since a coupler is a sequential program, the order established for receiving data, calculating fluxes, and sending updated data is very important. The ocean model is the most time-consuming one; these types of operations are a priority for it. For other model components, these operations take much less time. If the order is changed, the waiting time for other components increases. And this slows down the model operation.

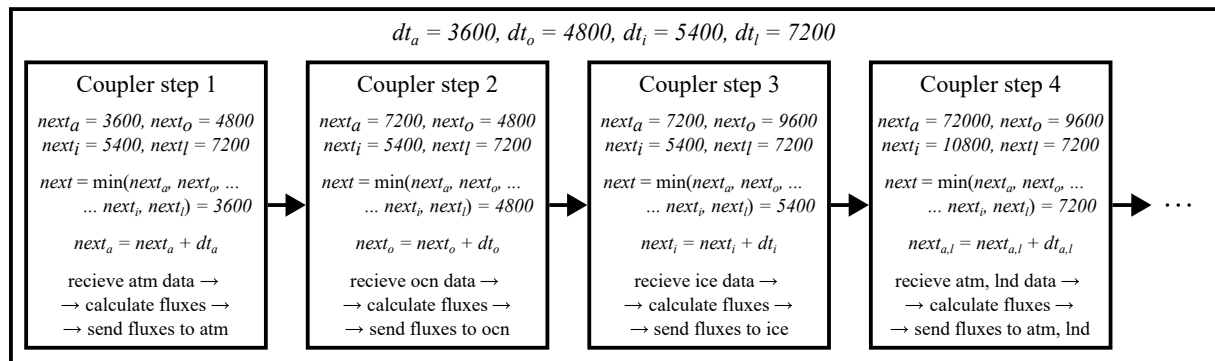


Figure 5. Example of the data exchange synchronization in SCM coupler

Finally, let us consider an example of how the process of data exchanges synchronization by the coupler is performed (Fig. 5). Let $dt_a = 3600$, $dt_o = 4800$, $dt_i = 5400$, $dt_l = 7200$ are time steps (in seconds) of each model component. Let us also assume that $next_a = next(\mathbf{atm})$, $next_o = next(\mathbf{ocn})$, $next_i = next(\mathbf{ice})$ and $next_l = next(\mathbf{land})$. Before the while loop starts, we have $next_a = dt_a$, $next_o = dt_o$, $next_i = dt_i$, $next_l = dt_l$. At the first step, $next = next_a = 3600$, i.e., $next \geq next_a$. Therefore, data exchange and flux calculation are performed only for the atmospheric component. The next time moment for atmosphere is $next_a = next_a + dt_a = 7200$. A similar process takes place for the second and third steps within the ocean and ice components, respectively. For the fourth step, we have $next \geq next_a$ and $next \geq next_l$, thus, in this step, data exchange and flux calculation are performed first for the atmosphere, then for the land.

2. SCM Coupler Functionality

This chapter describes other functions of the SCM coupler, such as interpolation, flux calculation and tidal effects accounting. These functions are used in the data preparation subroutines before being sent to the model components. The preparation subroutines are called *prepare_atmos_support*, *prepare_ocean_support*, *prepare_ice_support* and *prepare_land_support* (see [3]: Coupler_8/prepare.F).

2.1. Fluxes Calculation

Flux is the exchange rate of some media characteristics specified for a unit area per unit time. Fluxes determine how the climate components interact with each other. For each pair of model components, different fluxes are calculated e.g., heat and radiation fluxes from the atmosphere to the ocean ice, greenhouse gases fluxes from the land to the atmosphere, fresh water fluxes from ice to ocean, etc.

The flux calculation problem can be solved either by the coupler, which has information about the state of all components at any given time, or by the major component at the moment when it has all the necessary data from the other components. Note that in the general case it is difficult to determine where the fluxes should be calculated, either in the coupler or in any model component, because it depends on the features of a particular model. Therefore, the solution of this problem is left to the model developers.

The current version of the SCM coupler fully computes fluxes from atmosphere to ocean (see [3]: Coupler_8/flux_ao.F). Fluxes from atmosphere to ice are computed by the ice model, because the CICE model has its own flux calculation procedures. Moving these procedures from the ice component to the coupler requires a significant rearrangement in the model code and may cause incorrect model behavior. Nevertheless, the SCM coupler provides the option of the atmosphere-to-ice fluxes computation.

At the atmosphere-ocean interface, the values of wind stress $\vec{\tau}$, humidity E , sensible heat flux H , upward long-wave radiation flux L_{\uparrow} and CO_2 flux from the atmosphere to ocean must be calculated. The bulk aerodynamic formulae are used to evaluate $\vec{\tau}$, E and H . The general expressions are the following:

$$\begin{aligned} \vec{\tau} &= \rho_A (u^*)^2 \frac{\Delta\vec{U}}{|\Delta\vec{U}|}, & E_s &= \rho_A u^* Q^*, & H_s &= \rho_A C_{P_A} u^* \theta^*, \\ u^* &= \sqrt{C_U} |\Delta\vec{U}|, & Q^* &= \frac{C_E |\Delta\vec{U}| \Delta q}{u^*}, & \theta^* &= \frac{C_T |\Delta\vec{U}| \Delta\theta}{u^*}, \\ \Delta\vec{U} &= \vec{U} - \vec{U}_A, & \Delta q &= q - q_A, & \Delta\theta &= T - \theta_A, \end{aligned}$$

where ρ_A is the atmospheric surface density, C_{P_A} is the specific heat capacity of atmosphere, C_U, C_E, C_T are dimensionless drag-coefficients for wind stress, humidity and sensible heat, respectively, \vec{U}_A, q_A, θ_A are the horizontal wind velocity vector, the specific air humidity and the potential atmospheric temperature at some reference height ζ_r , respectively, and \vec{U}, q, T are the vector of horizontal ocean current velocity, the saturation humidity at the atmosphere-ocean interface, and the ocean surface temperature, respectively. The differences $\Delta\vec{U}, \Delta q, \Delta\theta$ are defined so that the downward fluxes are positive. It is worth noting that the velocities in the atmosphere are considerably higher than in the ocean, so one can neglect the ocean velocity in the formulas and put $\Delta\vec{U} = -\vec{U}_A$.

All bulk formulas differ from each other in a manner how the parametric coefficients C_U, C_E, C_T are evaluated. Two evaluation approaches are implemented in the SCM coupler: one is based on the COARE3.0 algorithm [1, 11], the other is based on the algorithm presented in the CSM Flux coupler (cpl4) user guide [23].

The general formula to calculate L_{\uparrow} is as in [23]

$$L_{\uparrow} = -\varepsilon\sigma T^4 + \alpha^L L_{\downarrow}, \quad (1)$$

where $\sigma = 5.67 \times 10^{-8} \frac{W}{m^2 K^4}$ is the Stefan-Boltzmann constant, ε is the emissivity of the interface, and α^L is the surface albedo for incident longwave radiation L_\downarrow . The value of L_\uparrow also depends on the air humidity, cloud cover and difference between the surface (ocean or ice) temperature T_S and near- surface air temperature T_A . Thus, the formula (1) becomes (see [5]):

$$L_\uparrow = L_\uparrow^0 + 4\varepsilon\sigma T_S^3 (T_S - T_A). \quad (2)$$

Two ways to determine L_\uparrow^0 can be used in the coupler:

$$\begin{aligned} L_\uparrow^0 &= \varepsilon\sigma T_S^4 (0.39 - 0.05\sqrt{e_a}) (1 - 0.6c^2), \\ L_\uparrow^0 &= \varepsilon\sigma T_S^4 (0.39 - 0.05\sqrt{e_a}) (1 - 0.8c), \end{aligned}$$

where e_a is the water vapor pressure in atmosphere, and c is the fractional cloud cover. The first expression is described in [13], the second one in [32]. The second expression was used by various Arctic Ocean models, including SibCIOM, in the framework of the AOMIP (Arctic Ocean Inter comparison Project).

There are also two ways to account L_\uparrow . In the first case, the fluxes L_\uparrow^I and L_\uparrow^O for the ocean surface and sea ice are calculated separately by formula (2) and the total flux is calculated as $L_\uparrow = \beta L_\uparrow^I + (1 - \beta) L_\uparrow^O$, where β is ice compactness. In the second case, it is assumed that $T_S = \varepsilon_I T_I \beta + \varepsilon_O T_O (1 - \beta)$ and $\varepsilon = \varepsilon_I \beta + \varepsilon_O (1 - \beta)$, where $\varepsilon_I, \varepsilon_O$ are emissivity for ice and ocean. Then the calculation is carried out according (2).

2.2. Data Interpolation

Let for some model component, for example, the ice model, the influence of atmospheric, ocean and land components have been described. These components send their current states and fluxes, based on these states, to the ice model using the coupler. Each component has its own computational grid. Let us call the grids of the atmosphere, ocean, and land models the source grids, and the ice model grid as the destination grid. To use all necessary data by the ice component, the data from the source grids must be interpolated onto the destination grid. This requires an interpolation subroutine.

The subroutine used in the coupler is based on the interpolation function described in [26]. Let $f : \mathbb{R}^n \rightarrow \mathbb{R}$ be a function of n variables and $\{\vec{x}_k \in \mathbb{R}^n, k = 1, \dots, M\}$ be a grid of any domain. The value of $f(\vec{x})$ in some grid point \vec{x}_k is denoted by $f_k = f(\vec{x}_k)$. The model component grid can include both points used and not used for calculation, e.g., the ocean model grid can include land points that are not used in the calculation (Fig. 2). Therefore, let us introduce additionally the mask of the domain as a function $m : \mathbb{R}^n \rightarrow \{0, 1\}$ such that $m(\vec{x}_k) = m_k$, where $m_k = 1$ if \vec{x}_k is a computational point and $m_k = 0$ if \vec{x}_k is not a computational point. The interpolation function is defined by the formula:

$$u(\vec{x}) = \frac{\sum_{k=1}^M W_k(\vec{x}) f_k}{\sum_{k=1}^M W_k(\vec{x})}, \quad W_k(\vec{x}) = \begin{cases} \exp\left(-E \frac{r_k^2}{R^2}\right), & r_k \leq R \vee m_k = 1 \\ 0, & r_k > R \vee m_k = 0 \end{cases},$$

where R is the radius of the circle centered at $\vec{x} \in \mathbb{R}^n$, $r_k = \rho(\vec{x}, \vec{x}_k) = |\vec{x} - \vec{x}_k|$ and $E = 4$.

The interpolation subroutine is time consuming enough, so if the model grids are fixed, it is better to compute the coefficients $W_k(\vec{x})$ once and use them for every time step. During the first

fluxes calculation, the coupler creates the files containing the matrices of interpolation coefficients $W_k(\vec{x})$ for each pair of model components. Then these matrices are stored in memory and used during computation. If the grid configuration for another model simulation has not changed, the interpolation coefficients are read from the saved files during the first flux calculation and used further. If the grid of any component has changed, it is necessary to delete manually the files with interpolation coefficients, related to this component. In this case, only part of the matrices will be recalculated.

Note that in addition to scalar data, vector data can also be sent to the components. Since the source grid and the destination grid can be rotated relative to each other by a certain angle, there is another interpolation procedure for vector data, based on the same interpolation function: it can rotate the vector on the destination grid by the required angle.

2.3. Tidal Model

The tidal phenomena have a significant impact on the dynamics of sea ice in the Arctic. For example, they can affect the sea ice velocity, its formation and degradation, etc. A more detailed description of the tide effect can be found in [17]. The coupler has a subroutine for describing the tidal phenomena (see [3]: Coupler_8/tides.F). It is based on the barotropic inverse model of the Northern Ice Ocean [29]. This module is called in the ice flux calculation procedure and it is universal for any ice model attached to the coupler, since it uses the sea ice model mask and the current date as input data.

3. Constructing New Climate Models Using SCM Coupler

One of the important tasks in Arctic climate research is to investigate the causal relationships between the Earth's and Arctic's climate changes. The SibCIOM model is useful for solving these kinds of problems. However, this model does not use dynamical atmosphere and land models, which makes it impossible to perform more detailed research. Therefore, it was decided to modify the SibCIOM model using the existing GACM instead of pure data components. The atmospheric model PUMA (Portable University Model of Atmosphere) of the intermediate complexity climate model PlaSim was taken first, and then the GACM of the INMCM48 climate model.

3.1. Overview of Basic Models

SibCIOM model

The SibCIOM model is a coupled ice-ocean model and has four independent model components: oceanic, atmospheric, sea-ice and land components interacting through the SCM coupler (Fig. 1a). There exist regional and global versions model.

The grid for the ocean component is a composite tripolar grid [28] composed of two components. The first component is spherical grid with a pole at 90°S. The second component is an asymmetric bipolar grid (“polar cap”) with poles on land: one in North America and the second in Eurasia.

The grid of the global version (Fig. 6a) has a horizontal resolution of $1^\circ \times 1^\circ$ in longitude and latitude in the spherical part and from 35 to 94 km within the polar cap. The regional version uses a grid (Fig. 6b) that covers only the Arctic Ocean and a part of the Atlantic Ocean above

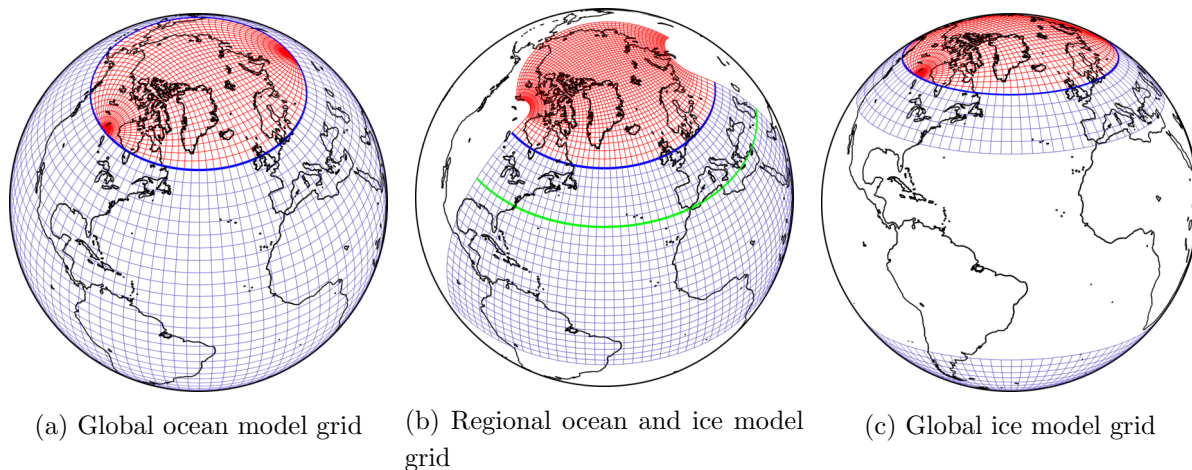


Figure 6. The SibCIOM composite grid composed of a spherical grid (blue) with a pole at 90°S and a bipolar asymmetric grid (red) with two poles allocated in North America and Eurasia

the latitude of 20°S . There are grids with a resolution of $0.5^{\circ} \times 0.5^{\circ}$ and $0.25^{\circ} \times 0.25^{\circ}$ in latitude and longitude directions in the spherical part and from 7 to 55 km and from 3 to 27 km in the polar cap, respectively. Both versions of the model have 38 vertical levels to a depth 5500 m.

The ice model uses the same grid as the ocean model for both model versions, though this is not necessary. The spherical area from 35°S to 35°N (Fig. 6c) excluded in the global version because there is no ice in this area. The regional version uses the ocean model grid (Fig. 6b) above 35°N (green line).

Current model configurations use different time steps ΔT_{ocn} and ΔT_{ice} . Time steps for the global version with resolution $1^{\circ} \times 1^{\circ}$ are $\Delta T_{ocn} = 4800 \text{ s}$ and $\Delta T_{ice} = 5400 \text{ s}$. The time steps of the regional version of the ocean model at resolutions $0.5^{\circ} \times 0.5^{\circ}$ and $0.25^{\circ} \times 0.25^{\circ}$ are $\Delta T_{ocn} = 2400 \text{ s}$ and $\Delta T_{ocn} = 1800 \text{ s}$, respectively. At the same time, the step of the ice model in both cases is $\Delta T_{ice} = 3600 \text{ s}$. The model components are synchronized according to the diagram described in Section 1.4.

SibCIOM is written in FORTRAN90 programming language using MPI. Oceanic and ice components are parallelized using MPI while atmospheric, land components and coupler are serial programs. MPI functions are also used for data exchange between coupler and other model components.

INMCM48 model

INMCM48 is a global climate model developed at the INM RAS. It is able to simulate the state of the atmosphere, ocean, sea ice, land and vegetation, taking into account the greenhouse gases. This model has the atmospheric (GACM) and oceanic (GOCM) components with mutual interaction.

The atmospheric component calculates the state of the atmosphere, land, vegetation and dynamics of aerosols. Here the horizontal resolution is $2^{\circ} \times 1.5^{\circ}$ in longitude and latitude, respectively. Vertically, 21 σ -levels up to $\sigma = 0.01$ are used. The oceanic component simulates the state of the ocean and sea ice. It has a resolution $1^{\circ} \times 0.5^{\circ}$ in longitude and latitude and 40 vertical σ -levels. The maximum depth is 5500 m. In current version time step of the GACM is $\Delta T_{atm} = 240 \text{ s}$, while the GOCM time step is $\Delta T_{atm} = 1800 \text{ s}$.

The GACM and GOCM components of the INMCM48 model can be run either in stand-alone mode (Fig. 7a) or in a coupled mode (Fig. 7b). The coupled run is controlled by a simple driver that distributes all MPI processes within the MPI_COMM_WORLD communicator to the four communicators: atmospheric (MPI_COMM_ATM), oceanic (MPI_COMM_OCEAN), atmospheric-trace (MPI_COMM_ATM_XX) and ocean-trace (MPI_COMM_OCEAN_XX). The tracer models are optional and work only with their corresponding main models (Fig. 7). For stand-alone running, there are individual drivers for each component.

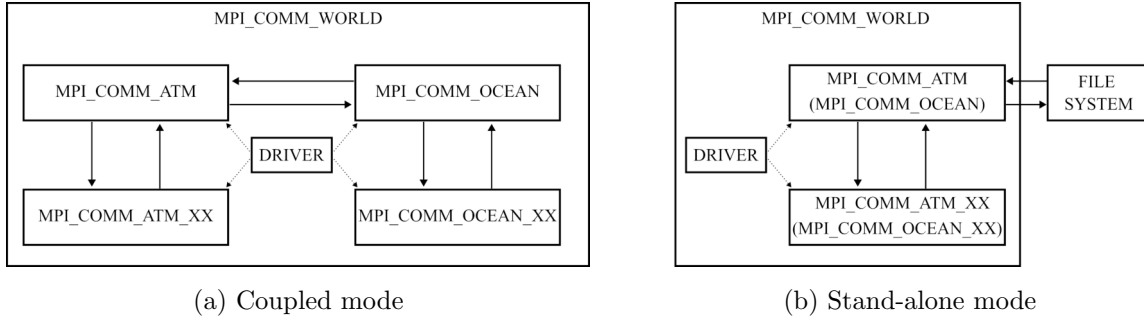


Figure 7. INMCM48 model interaction diagrams

As mentioned in the introduction, the GACM and GOCM components are coupled directly through master processes without any coupler. The GACM sends to GOCM data every 2 hours, while GOCM sends data every 1 hour. At the same time, the GACM each next time step after the ocean data received from the GOCM uses the same data for the computations.

INMCM48 is coded with FORTRAN77 language. GACM and GOCM components parallelized using MPI. This model is unique and has been used in various climate researches. It has also been verified many times during the CMIP experiments.

PlaSim model

Intermediate complexity climate models are the models allowing various simplifications in describing some physical processes. Such models are suitable in investigating particular physical processes and relationships between them when some details can be neglected, e.g. subscale processes. One such model is the PlaSim (Planet Simulator) global climate model developed at the Institute of Meteorology (the University of Hamburg).

PlaSim model includes atmospheric, ocean, sea ice, land and biosphere components. The dynamic core of the model is the PUMA (Portable University Model of the Atmosphere). The other components are nested as subroutines in the atmospheric model.

The PlaSim model works with fine resolution grids and has a limited number of grid resolution configurations. The number of grid points by latitude is 2^n , by longitude is 2^{n+1} , where $n = 1, \dots, 8$. The vertical coordinate is σ , and the minimum number of vertical levels is 5. Note, that all components use the same grid. Default time step of the atmosphere model is $\Delta T_{atm} = 2700 s$, while the sea ice and ocean model use a time step of one day.

The model does not have an external coupler and works in SIMD mode. Atmosphere and sea ice models coupled directly every 32 time steps and the sea ice and ocean model are coupled every time step.

Table 1. Arrays sent by the atmospheric component to the coupler

Atmosphere → Coupler				
Variable name	Name in model	SibCIOM	PlaSim-ICMMG1.0	INMCM-SibCIOM
Wind velocity height	zlvl	+	+	+
Temperature and humidity height	zvlv	+	+	+
<i>U</i> -wind velocity components	uatm	+	+	+
<i>V</i> -wind velocity components	vatm	+	+	+
Potential temperature	potT	+	+	+
Air temperature	Tair	+	+	+
Specific humidity	Qa	+	+	+
Air density	rhoa	+	+	+
Downward shortwave radiation	Fsw	+	+	+
Downward longwave radiation	Flw	+	+	+
Precipitation rate (rain)	Frain	+	+	+
Precipitation rate (snow)	Fsnow	+	+	+
Cloudiness	cld	+	+	–
CO ₂ flux to ocean	cco2	+	+	+
Greenland river discharge transport	small_riv	+	+	+
<i>U</i> -wind stress component	taux	–	–	+
<i>V</i> -wind stress component	tauy	–	–	+
Upward longwave radiation	lwup	–	–	+
Sensible heat flux	hsn	–	–	+
Latent heat flux	hlt	–	–	+

Table 2. Arrays delivered by the coupler to the atmospheric component

Coupler → Atmosphere				
Variable name	Name in model	SibCIOM	PlaSim-ICMMG1.0	INMCM-SibCIOM
Surface temperature	tsurf	–(+)	+	+
Drag-coefficient for wind stress	cu	–(+)	+	–
Drag-coefficient for sensible heat flux	ct	–(+)	+	–
Drag-coefficient for evaporation flux	ce	–(+)	+	–
Surface albedo	albed	–(+)	+	–
Ice compactness	dicec	–(+)	+	+
CO ₂ flux to the atmosphere	fco2	–(+)	+	+
Sensible heat flux	hsn	–(+)	–	+
Latent heat flux	hlt	–(+)	–	+

3.2. Construction of New Climate Models

To couple a new model component to the coupler, the component must be configured as described in Section 1.1. Then, the coupling is carried out through MPI data exchange procedures.

During data exchange step, each model component sends to the coupler its current state described by a set of two-dimensional arrays. The current states of the models are required to compute the fluxes. After the fluxes have been computed, the coupler sends back to each component the other sets of two-dimensional arrays. For example, in the SibCIOM model, the atmospheric component sends to the coupler 15 arrays (Tab. 1), which are sufficient for calculating the atmospheric fluxes between atmosphere and ocean and atmosphere and ice. The coupler may send any data to the atmospheric component, but the data will not be used by it since the atmospheric component is a pure model component (Tab. 2). In the general case, the set of arrays is not fixed and depends on the coupled model. Note that the data transfer from the coupler to the atmospheric component can be enabled when testing the coupler's communication capabilities.

Two examples of constructing the climate models using the SCM coupler are described below. The main principle is replacing the atmospheric component of the SibCIOM model with a General Atmospheric Circulation Model. The first model is obtained by coupling the GACM of INMCM48 model [36], the second model is obtained by using the GACM of PlaSim model [12].

INMCM-SibCIOM model

When building the INMCM-SibCIOM model, the original driver of the GACM component was modified by replacing the original initialization subroutines with the *setup_mpi('model_name')* (see Section 1.2). The MPI communicator of the GACM becomes the sub-communicator created in this subroutine.

The data exchange process between the components was also changed. In the original INMCM48 model, the GACM component received 5 two-dimensional arrays, and then sent 11 two-dimensional arrays, one one-dimensional array and one constant. To synchronize the exchanges between coupler and GACM, the order of exchanges was reversed: sending to coupler first, then receiving from coupler. The original subroutines were replaced with the new ones using functions from [3]: *MPI/mpi_tools.F*. The arrays sent to the coupler are shown in Tab. 1. The arrays received from the coupler remained the same as for the original model (see. Tab. 2). The coupling also began to be performed every step of the atmosphere model instead of two, as in the original model.

It is worth noting that flux calculation functions from the atmosphere to the ocean were disabled in the coupler to avoid a significant heat imbalance. The program code of the atmosphere model is too complex to make various changes to it, so the best solution was to send the fluxes calculated by GACM.

Thus, the INMCM-SibCIOM model is a global climate model composed of three model components coupled with each other through the SCM coupler. Since the GACM includes a nested land model, the interaction diagram of the model components corresponds to diagram from Fig. 1a with the missing land component. The configurations of the grids and time steps correspond to ones that global SibCIOM model and GACM of the INMCM48 model use (Section 3.1).

The availability of such a climate model gives more opportunities to investigate a wide range of problems related, for example, to the study of the response of atmospheric circulation to climate change in the Arctic.

PlaSim-ICMMG1.0 model

The PUMA model is coupled to the SCM coupler in the same way as the GACM of the INMCM48 model. Subroutines for data exchange, consistent with the SCM coupler protocol, were developed and the necessary synchronization points were created. The process of model development is described in more detail in [31].

The PUMA also has a nested land model like the GACM of INMCM48. Therefore, the interaction diagram corresponds to that of the INMCM-SibCIOM model. The set of arrays to be exchanged is presented in Tab. 1 and Tab. 2. In this model, the fluxes from the atmosphere to the ocean are computed by the coupler in the same way as for the SibCIOM model.

The grid configurations and time steps for the ocean and ice models are the same as in the global version of the SibCIOM model. The grid of the atmospheric model has a resolution $2.8125^\circ \times 2.8125^\circ$ in latitude and longitude (64 by 128 nodes), respectively. The coupling is performed each time step $\Delta T_a = 1200$ s.

Thus, the PlaSim-ICMMG1.0 model is an intermediate complexity model since it has a PUMA as the atmospheric component, but with complete ocean and sea ice models from the SibCIOM model. Therefore, this model can be used, e.g., to study both changes in the large-scale atmospheric dynamics occurring on large time scales due to changes in the Arctic climate and, conversely, the long-term effect of large-scale atmospheric phenomena on the climate of the Arctic region. In particular, this model was used to investigate the mechanisms that influence the recently observed Arctic climate warming [31].

Conclusions

The SCM coupler performs its primary functions, such as exchange synchronization and interpolation, and has other optional functions, such as flux calculation and tidal effect accounting. But, it is less functional concerning the already existing couplers. For example, the OASIS coupler is implemented in several programming languages (Fortran, and C), has coupling library and integrated coupling framework versions, and includes several types of interpolation functions. Nevertheless, having our software product makes it significantly easier to control its versions and allows us to modify it with less effort.

The SCM coupler allows coupling different models of climate system components, as was demonstrated by the GACM components of the INMCM48 and PlaSim models. Thus, climate model INMCM-SibCIOM and climate model of intermediate complexity PlaSim-ICMMG1.0 were built. These models allow solving various problems of climate variability on different time scales. Moreover, they enable us to analyze and understand in more detail the drivers of climate change in the Arctic region and estimate its impact on the global climate.

Several modifications are planned for the next version of SCM coupler. First is the implementation of its parallelized version to improve performance. In particular, it would be useful to implement parallel flux calculation and parallel interpolation in the coupler as it is done in modern couplers. This is especially important when increasing the grid resolution of the model. Also, multiple MPI processes can concurrently exchange data with model components, which will reduce coupler idle time. Second is adding an interpolation procedure with the conservation

property for arbitrarily shaped grids, since the standard interpolation procedure may over- or underestimate the interpolated value in a grid cell. It is crucial, for example, when calculating radiation and heat fluxes since there is a balance of heat and radiation in the simulated climate system, and disbalance can lead to errors. The third task is creating more flexible versions of the coupler suitable for coupling to any number of different components.

The source code SCM coupler is available on Github [3].

Acknowledgements

This work was conducted within the framework of the budget project 0251-2022-0003 for ICMMG SB RAS. We would like to thank the reviewers for their time and effort in reviewing the manuscript. We sincerely appreciate all the meaningful comments and suggestions that have helped us to improve the quality of the manuscript.

This paper is distributed under the terms of the Creative Commons Attribution-Non Commercial 3.0 License which permits non-commercial use, reproduction and distribution of the work without further permission provided the original work is properly cited.

References

1. COARE algorithm, Version 3.0, <https://github.com/NOAA-PSL/COARE-algorithm.git>
2. Flexible Modeling System (FMS) Coupler, <https://github.com/NOAA-GFDL/FMSCoupler.git>
3. SCM (SibCIOM Coupling Module) coupler, <https://github.com/SibCIOM/SCM-coupler.git>
4. Bitz, C.M., Lipscomb, W.H.: An energy-conserving thermodynamic model of sea ice. *JGR: Oceans* 104(C7), 15669–15677 (1999). <https://doi.org/10.1029/1999JC900100>
5. Budyko, M.I.: *Climate and Life*. International geophysics series, Academic Press (1974), <https://books.google.ru/books?id=U0IazQEACAAJ>
6. Collins, N., Theurich, G., DeLuca, C., *et al.*: Design and implementation of components in the earth system modeling framework. *The International Journal of High Performance Computing Applications* 19(3), 341–350 (2005). <https://doi.org/10.1177/1094342005056120>
7. Craig, A., Valcke, S., Coquart, L.: Development and performance of a new version of the OASIS coupler, OASIS3-MCT_3.0. *Geoscientific Model Development* 10(9), 3297–3308 (2017). <https://doi.org/10.5194/gmd-10-3297-2017>
8. Craig, A., Vertenstein, M., Jacob, R.: A new flexible coupler for earth system modeling developed for CCSM4 and CESM1. *The International Journal of High Performance Computing Applications* 26(1), 31–42 (2012). <https://doi.org/10.1177/1094342011428141>
9. Cummins, D.P., Stephenson, D.B., Stott, P.A.: Optimal Estimation of Stochastic Energy Balance Model Parameters. *Journal of Climate* 33(18), 7909–7926 (2020). <https://doi.org/10.1175/JCLI-D-19-0589.1>

10. Dukowicz, J.K., Baumgardner, J.R.: Incremental Remapping as a Transport/Advection Algorithm. *Journal of Computational Physics* 160(1), 318–335 (2000). <https://doi.org/10.1006/jcph.2000.6465>
11. Fairall, C.W., Bradley, E.F., Hare, J.E., *et al.*: Bulk Parameterization of AirSea Fluxes: Updates and Verification for the COARE Algorithm. *Jour. of Climate* 16(4), 571–591 (2003). [https://doi.org/10.1175/1520-0442\(2003\)016<0571:BPOASF>2.0.CO;2](https://doi.org/10.1175/1520-0442(2003)016<0571:BPOASF>2.0.CO;2)
12. Fraedrich, K., Jansen, H., Kirk, E., *et al.*: The Planet Simulator: Towards a user friendly model. *Meteorologische Zeitschrift* 14(3), 299–304 (2005). <https://doi.org/10.1127/0941-2948/2005/0043>
13. Gill, A.: *Atmosphere-Ocean Dynamics*. International Geophysics Series, Academic Press (1982), <https://books.google.ru/books?id=UOIazQEACAAJ>
14. Golubeva, E.N.: Numerical modeling of the Atlantic Water circulation in the Arctic Ocean using QUICKEST scheme. *Computational technologies* 13(5), 11–24 (2008), <http://www.ict.nsc.ru/jct/annotation/1166?l=eng>, (in Russian)
15. Golubeva, E.N., Platov, G.A.: On improving the simulation of Atlantic Water circulation in the Arctic Ocean. *Journal of Geophysical Research: Oceans* 112(C4) (2007). <https://doi.org/10.1029/2006JC003734>
16. Haggag, M., Yamashita, T., Lee, H., Kim, K.: A coupled atmosphere and multi-layer land surface model for improving heavy rainfall simulation. *Hydrology and Earth System Sciences Discussions* 5, 1067–1100 (2008). <https://doi.org/10.5194/hessd-5-1067-2008>
17. Holloway, G., Proshutinsky, A.: Role of tides in Arctic ocean/ice climate. *Journal of Geophysical Research: Oceans* 112(C4) (2007). <https://doi.org/10.1029/2006JC003643>
18. Hunke, E.C., Dukowicz, J.K.: An ElasticViscousPlastic Model for Sea Ice Dynamics. *Journal of Physical Oceanography* 27(9), 1849–1867 (1997). [https://doi.org/10.1175/1520-0485\(1997\)027<1849:AEVPMF>2.0.CO;2](https://doi.org/10.1175/1520-0485(1997)027<1849:AEVPMF>2.0.CO;2)
19. Jackson, L.S., Maycock, A.C., Andrews, T., *et al.*: Errors in Simple Climate Model Emulations of Past and Future Global Temperature Change. *Geophysical Research Letters* 49(15), e2022GL098808 (2022). <https://doi.org/10.1029/2022GL098808>
20. Jacob, R., Larson, J., Ong, E.: $M \times N$ Communication and Parallel Interpolation in Community Climate System Model Version 3 Using the Model Coupling Toolkit. *The International Journal of High Performance Computing Applications* 19(3), 293–307 (2005). <https://doi.org/10.1177/1094342005056116>
21. Kalmykov, V.V., Ibrayev, R.A., Kaurkin, M.N., Ushakov, K.V.: Compact Modeling Framework v3.0 for high-resolution global oceaniceatmosphere models. *Geoscientific Model Development* 11(10), 3983–3997 (2018). <https://doi.org/10.5194/gmd-11-3983-2018>
22. Kalnay, E., Kanamitsu, M., Kistler, R., *et al.*: The NCEP/NCAR 40-Year Reanalysis Project. *Bulletin of the American Meteorological Society* 77(3), 432–472 (1996). [https://doi.org/10.1175/1520-0477\(1996\)077<0437:TNYRP>2.0.CO;2](https://doi.org/10.1175/1520-0477(1996)077<0437:TNYRP>2.0.CO;2)
23. Kauffman, B.G., Large, W.G.: NCAR CSM Flux Coupler, version 4.0. <https://www2.cesm.ucar.edu/models/cpl/doc4> (1998), accessed: 2023-01-15

24. Larson, J., Jacob, E., Ong, E.: The Model Coupling Toolkit: A New Fortran90 Toolkit for Building Multiphysics Parallel Coupled Models. *The International Journal of High Performance Computing Applications* 19(3), 277–292 (2005). <https://doi.org/10.1177/1094342005056115>
25. Leach, N.J., Jenkins, S., Nicholls, Z., *et al.*: FaIRv2.0.0: a generalized impulse response model for climate uncertainty and future scenario exploration. *Geoscientific Model Development* 14(5), 3007–3036 (2021). <https://doi.org/10.5194/gmd-14-3007-2021>
26. Levitus, A., Boyer, T.P.: World ocean atlas 1994. Vol. 4, Temperature. NOAA atlas NESDIS; 4 (1994), <https://repository.library.noaa.gov/view/noaa/1381>
27. Lipscomb, W.H., Hunke, E.C.: Modeling Sea Ice Transport Using Incremental Remapping. *Monthly Weather Review* 132(6), 1341–1354 (2004). [https://doi.org/10.1175/1520-0493\(2004\)132<1341:MSITUI>2.0.CO;2](https://doi.org/10.1175/1520-0493(2004)132<1341:MSITUI>2.0.CO;2)
28. Murray, R.J.: Explicit Generation of Orthogonal Grids for Ocean Models. *Journal of Computational Physics* 126(2), 251–273 (1996). <https://doi.org/10.1006/jcph.1996.0136>
29. Padman, L., Erofeeva, S.: A barotropic inverse tidal model for the Arctic Ocean. *Geophysical Research Letters* 31(2) (2004). <https://doi.org/10.1029/2003GL019003>
30. Perezhugin, P., Chernov, I., Iakovlev, N.: Advanced parallel implementation of the coupled ocean–ice model FEMAO (version 2.0) with load balancing. *Geoscientific Model Development* 14(2), 843–857 (2021). <https://doi.org/10.5194/gmd-14-843-2021>
31. Platov, G., Krupchatnikov, V., Martynova, Y., *et al.*: A new earths climate system model of intermediate complexity, PlaSim-ICMMG-1.0: description and performance. *IOP Conference Series: Earth and Environmental Science* 96(1), 012005 (2017). <https://doi.org/10.1088/1755-1315/96/1/012005>
32. Rosati, A., Miyakoda, K.: A General Circulation Model for Upper Ocean Simulation. *Journal of Physical Oceanography* 18(11), 1601–1626 (1988). [https://doi.org/10.1175/1520-0485\(1988\)018<1601:AGCMFU>2.0.CO;2](https://doi.org/10.1175/1520-0485(1988)018<1601:AGCMFU>2.0.CO;2)
33. Stute, M., Clement, A., Lohmann, G.: Global climate models: Past, present, and future. *Proceedings of the National Academy of Sciences* 98(19), 10529–10530 (2001). <https://doi.org/10.1073/pnas.191366098>
34. Valcke, S.: The OASIS3 coupler: a European climate modelling community software. *Geoscientific Model Development* 6(2), 373–388 (2013). <https://doi.org/10.5194/gmd-6-373-2013>
35. Volodin, E.M., Gritsun, A.S.: Simulation of Possible Future Climate Changes in the 21st Century in the INM-CM5 Climate Model. *Izvestiya, Atmospheric and Oceanic Physics* 56(3), 218–228 (2018). <https://doi.org/10.1134/S0001433820030123>
36. Volodin, E.M., Mortikov, E.V., Kostykin, S.V., *et al.*: Simulation of the modern climate using the INM-CM48 climate model. *Russian Journal of Numerical Analysis and Mathematical Modelling* 33(6), 367–374 (2018). <https://doi.org/10.1515/rnam-2018-0032>