Application of the Parallel INMOST Platform to Subsurface Flow and Transport Modelling

Igor Konshin[1,2( )], Ivan Kapyrin[1,3], Kirill Nikitin[1,3], and Kirill Terekhov[1,4]

[1] Institute of Numerical Mathematics of the Russian Academy of Sciences, Moscow 119333, Russia
igor.konshin@gmail.com
[2] Dorodnicyn Computing Centre of the Russian Academy of Sciences, Moscow 119333, Russia
[3] Nuclear Safety Institute of the Russian Academy of Sciences, Moscow 115191, Russia
[4] Stanford University, Stanford, CA 94305, USA

Abstract. INMOST (Integrated Numerical Modelling and Object-oriented Supercomputing Technologies) is a tool for supercomputer simulations characterized by a maximum generality of supported computational meshes, distributed data structure flexibility and cost-effectiveness, as well as crossplatform portability. INMOST is a software platform for developing parallel numerical models on general meshes. User guides, online documentation, and the open-source code of the library is available at http://www.inmost.org.

To demonstrate the power and efficiency of the specified technology the solutions of subsurface flow and transport problems was considered. The efficiency of the parallel solution of the multiphase flow model was shown for up to several thousands of cores. Real-life examples of advective–diffusive–dispersive transport with sorption and decay modeling as well as a reactive transport problem were also considered.

Keywords: Numerical modelling · Software platform · Distributed meshes · Subsurface flow and transport

1 Introduction

The solution of industrial boundary-value problems requires high quality approximation and discretization of the problem. In addition to the high accuracy discretization it is necessary to use the general unstructured meshes that fit the problem geometry. On the other hand, to increase the approximation accuracy the usage of very large dimension meshes is required. It results in exploiting of the distributed computations on a modern parallel computers. A developer requires a tool that helps operating with distributed mesh data.

In general, the boundary-value problems solution consist of the following stages:

– mesh generation;

R. Wyrzykowski et al. (Eds.): PPAM 2015, Part II, LNCS 9574, pp. 277–286, 2016.
DOI: 10.1007/978-3-319-32152-3_26

– distribution of mesh data to processors;
– problem discretization;
– assembling linear system;
– solution of linear system;
– visualization of initial data and solution results.

A lot of existing softwares (STK, FMDB, MOAB, MSTK, OpenFOAM, Salome and some others) try to operate with distributed mesh data. While solving a particular problem we were not able to find a library that completely satisfies our requirements. Some of the libraries does not support operations with arbitrary polygon elements, some are insufficiently reliable and their realizations are not effective, several libraries support only one layer of ghost cells, some libraries does not support mesh modification during modelling, there is no possibility (or some difficulties) with incorporating of user approximation schemes, or there is no code portability between different computer platforms (Windows, Linux). That is why a decision to develop our own mesh platform INMOST was done [3–5,9].

2 Algorithmic Specification of INMOST Kernel

Mesh platform should support the following mesh elements:

– Vertex, which contains information on the position in the space;
– Edge, which consists of 2 or more vertices;
– Face, polygon in general case, which is based on a set of edges;
– Cell, polyhedron in general case, which is based on a set of faces.

These mesh elements suppose the following hierarchy (see Fig. 1):

$$\text{Cell} \Rightarrow \text{Face} \Rightarrow \text{Edge} \Rightarrow \text{Vertex}.$$

On the other hand the same hierarchy can be presented in the reverse order (see Fig. 2).

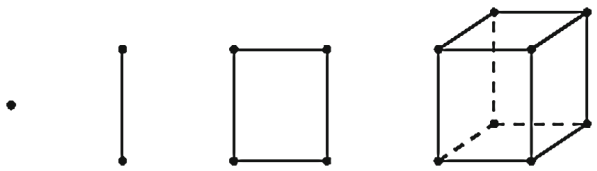


Fig. 1. Basic mesh elements: Vertex, Edge, Face, Cell.

In addition, the mesh elements can be specified by their dimension (see Fig. 3):

– (0D) Node: Vertex;

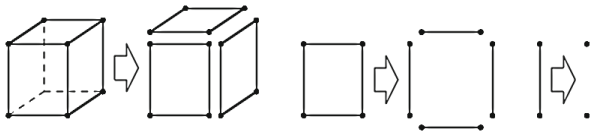


Fig. 2. Elements composition: Cell ⇒ Face ⇒ Edge ⇒ Vertex.

– (1D) Edge: Line;
– (2D) Face: Triangle, Quad, Polygon;
– (3D) Cell: Tetrahedron, Hexahedron, Prism, Pyramid, Polyhedron.

One of the most important mesh function used on discretization stage is a search of neighboring elements:

$$\begin{array}{ccc} \text{Cell} & \Leftrightarrow & \text{Face} \\ \Updownarrow & & \Updownarrow \\ \text{Vertex} & \Leftrightarrow & \text{Edge} \end{array}$$

Not only the above mentioned connections but a complete set of element connections can be applied as well.

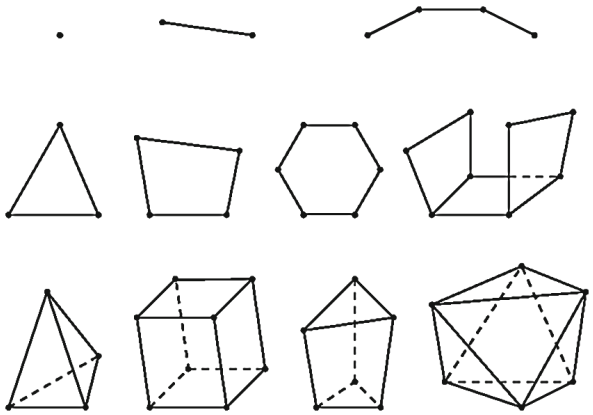


Fig. 3. Elements types.

Except for elements description to operate with mesh elements data is also required. INMOST mesh platform gives such an opportunity: Elements – to store the mesh configuration; Data – to store information in the mesh elements (data types can be dense or sparse; integer, float, or binary; a single value or an array of values); Tags – to connect the mesh data to the elements. To realize the above mentioned opportunity the mesh functions should operate with: Data, Tags, Elements, Set of elements, and Mesh.

But in spite of the wide set of the mesh functions, INMOST is not a mesh generator although such a generator can be written on its base (see example GridGen in [3]).

Except for the mentioned features, a user can require the following operations to handle distributed mesh data:

– Distribute mesh between processors;
– Specify ghost elements;
– Store data for elements in tags;
– Exchange tag data for ghost elements;
– Generate problem matrix from distributed data;
– Call parallel linear solver for distributed matrix;
– Perform global operations (min, max, sum, etc.);
– Save mesh data in a parallel format file (.pvtk, .pmf).

All the functions above are implemented in INMOST.

To distribute, redistribute, and balance the mesh data external packages ParMetis and Zoltan can used as an internal INMOST function. The solution of distributed linear systems generated during discretization can be performed by PETSc, Trilinos, or by a set of internal linear solvers in the same interface.

3 Numerical Experiments

3.1 Two- and Three-Phase Black-Oil Modelling

We consider parallel two- and three-phase black-oil models with the nonlinear monotone flux approximation presented in [7].

The first numerical test for a parallel version of three-phase black-oil model was performed on the BlueGene/P cluster located in the Moscow State University and two parts of the INM cluster:

– BG/P system consists of relatively slow PowerPC 450 (850 MHz) cores with 2 GB RAM each.
– The first part of the INM cluster (INM-1) consists of nodes with two quad-core Intel Xeon X5355 (2.66 GHz) or Intel Xeon E5462 (2.80 GHz) processors and 8 GB RAM per node.
– The second part of the INM cluster (INM-2) consists of nodes with two six-core Intel Xeon X5650 (2.67 GHz) and 24 GB RAM per node.

The problem set-up is the following. The square region contains two wells in the opposite corners: one injector and one producer with given bottom hole pressures.

In our parallel simulation, we use parallel grid generation. At the first stage the computational domain is split into subdomains which are distributed between available cores. At the second stage each core constructs a local grid inside the associated subdomain and exchanges ghost cells with neighbours. Only one layer of ghost cells is sufficient due to the compact stencil of discrete operators. Grid partitioning example is shown on Fig. 4.

The total grid dimensions are $128 \times 128 \times 16$ which gives us total of 304 192 nodes (cells + boundary entities).

Linear systems were solved with the PETSc package. The chosen solver is BCG iterations combined with the additive Schwarz preconditioner and ILU0 preconditioners in subdomains.

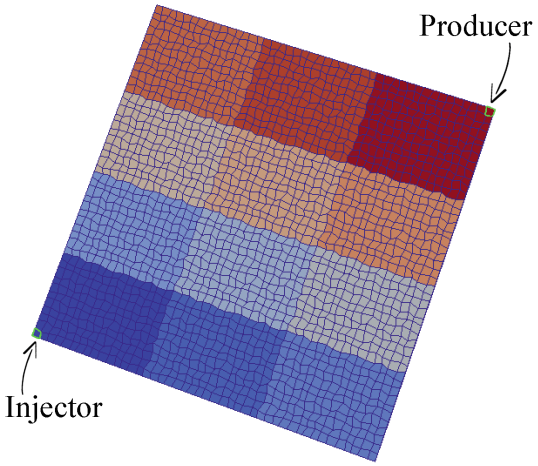


Fig. 4. Grid partitioning example.

Table 1. Relative speed-up of simulation, BG/P.

#cores	Nodes/core	#lit	t_{init}	t_{sol}	speed-up
8	38 024	71 024	68.8 s	28 549 s	1x
16	19 012	71 042	37.2 s	14 471 s	1.97x
32	9 506	71 648	19.6 s	7 464 s	3.82x
64	4 753	72 174	10.5 s	3 874 s	7.36x
128	2 377	73 806	5.9 s	2 059 s	13.86x

Table 1 shows the results of the parallel experiment on BG/P for 200 days simulation. One can see good relative speed-up for up to 128 cores (2.4 k nodes per core). The number of total nonlinear iterations is 648 and does not depend on the number of cores. The number of linear iterations increases slightly as #cores grows, while the initialization and computation times decrease almost linearly. We note that the BG/P system has fast connection with relatively slow computational cores.

Table 2 shows the results for INM-1 and INM-2 which have much faster cores than BG/P. As expected, the relative speed-up is lower albeit satisfactory: up to 11x for 8-to-128 cores on INM-1 and 4.5x for 8-to-64 cores on INM-2.

Figure 5 presents the relative speed-up of the parallel computation which in case of BG/P cluster is close to the ideal linear speed-up. Figure 6 shows the diagram with computational times on three clusters.

The presented results demonstrate good quality of the developed parallel data structure and algorithms, although we use the third-party PETSc linear solver which also can be improved.

The second experiment deals with two-phase flow model on a massively parallel BG/P system with up to 8192 cores. Problem setup and grid construction method is similar to the first test case. We consider 50 days simulation on 0.9 million cells nonorthogonal hexahedral grid (1.8 million unknowns).

Table 2. Relative speed-up of simulation, INM-1 and INM-2 clusters.

	INM-1			INM-2		
#cores	t_{init}	t_{sol}	speed-up	t_{init}	t_{sol}	speed-up
8	9.9 s	12 506 s	1x	6.2 s	4 909 s	1x
16	5.2 s	6 182 s	2.02x	3.8 s	2 980 s	1.65x
32	3.0 s	3 756 s	3.33x	2.4 s	1 957 s	2.51x
64	1.7 s	1 926 s	6.49x	2.0 s	1 092 s	4.50x
128	1.0 s	1 131 s	11.06x	–	–	–

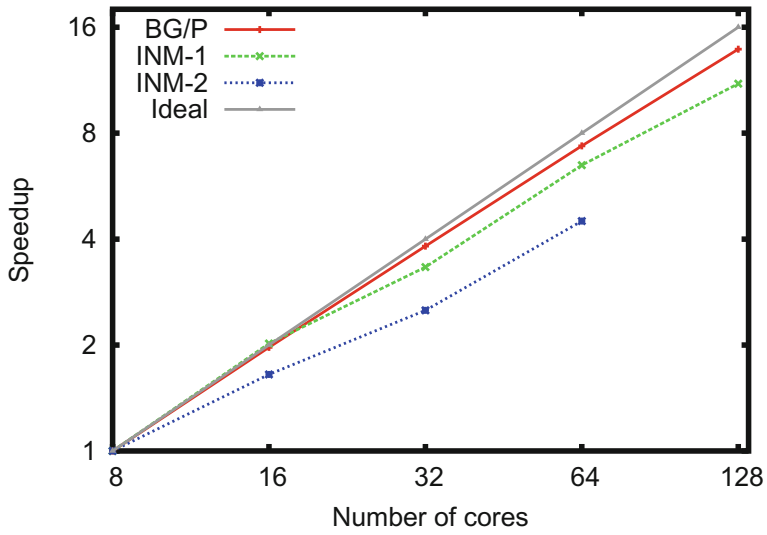


Fig. 5. Relative speed-up, BG/P, INM-1 and INM-2.

Table 3 presents number of linear and nonlinear iteration, initialization, grid generation and total simulation times of the parallel experiment for two-phase flow model. The reference results are taken for 512 cores run. One can see that the total simulation time decreases, yet there is almost no speed-up for 1024-to-2048 and 4096-to-8192 pairs (see Fig. 7, left). This is explained by the reduction of the subproblem sizes and the sharp increase of the number of the linear iterations in these pairs (see Fig. 7, right).

The performed parallel experiments show that the INMOST platform is useable for parallel simulations even with severely low unknowns per core numbers (down to 220 in the last test) and allows to achieve good scalability of the numerical model with minimal changes of the serial code.

3.2 Groundwater Flow and Contaminant Transport Modelling

The platform may be efficiently applied for complex multidimensional problems demanding high-performance simulation of various coupled processes. An example is the groundwater flow and contaminant transport in porous media modellings [6]. One can observe the general trends in the development of

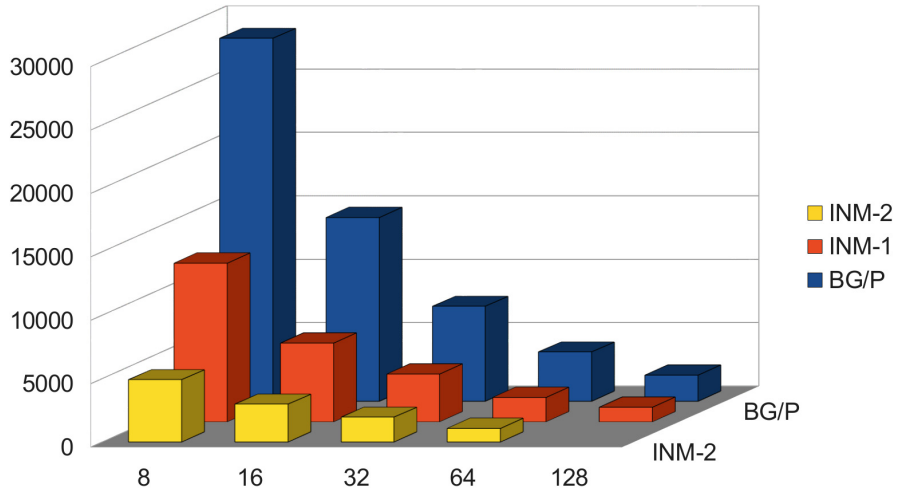


Fig. 6. Solution times for parallel computation, BG/P, INM-1 and INM-2.

Table 3. Relative speed-up of the two-phase flow simulation, BG/P.

#cores	#nonlit	#lit	t_{init}		t_{grid} t_{sol}	speed-up
512	151	247 919	2.69	1.19	1478.7	1x
1024	151	190 099	1.80	0.70	559.0	2.64x
2048	150	333 369	1.34	0.49	536.2	2.76x
4096	148	291 533	1.53	0.41	296.7	4.98x
8192	147	402 742	1.91	0.46	296.0	5.0x

hydrogeological modelling codes. From the numerical point of view these are the use of unstructured adaptive grids, the development of discretizations suitable for this type of grids and parallelization (examples are the MODFLOW-USG code [8] and the ASCEM project of US DOE [2].

The means for mesh and data storage, matrix and vector assembly implemented in the platform were used to create the models of the following processes on unstructured grids:

– saturated and unsaturated groundwater flow;
– advective–diffusive–dispersive transport with sorption and decay;
– reactive transport;
– density-driven flow.

On Fig. 8 we show the application of the code to the safety assessment of a surface radioactive waste disposal facility. The problem features an adaptive mesh composed of triangular prisms, and quite a specific geological structure featuring 10 layers with heterogeneous hydraulic conductivity tensor. Combined with Qt and VTK libraries the INMOST platform allows to organize a full workflow for hydrogeological modelling: creation of a geological model, model data setting, grid generation, numerical flow and transport modelling with the ability to run in parallel, and finally the visualization of results. The broad

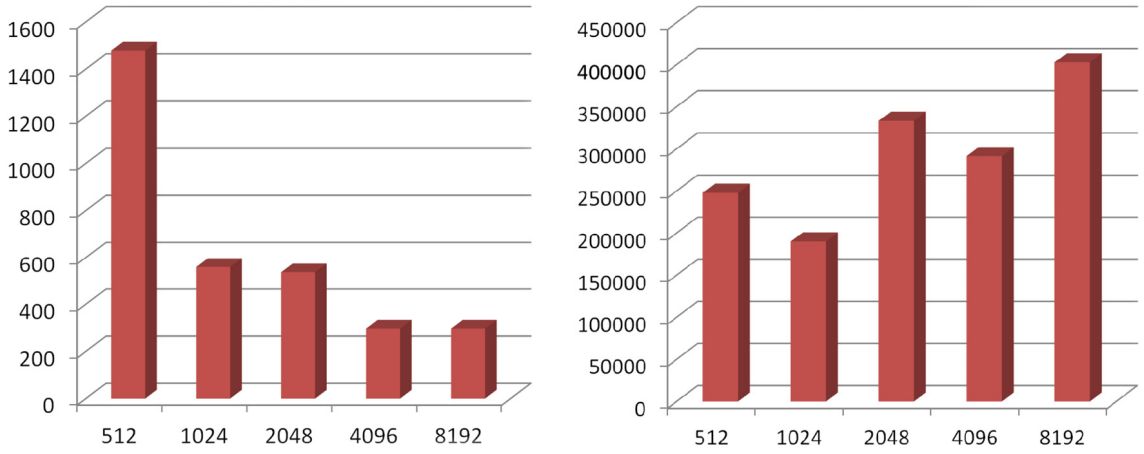


Fig. 7. Left: reduction of computation time compared to 512-cores experiment. Right: total number of linear iterations for simulation.

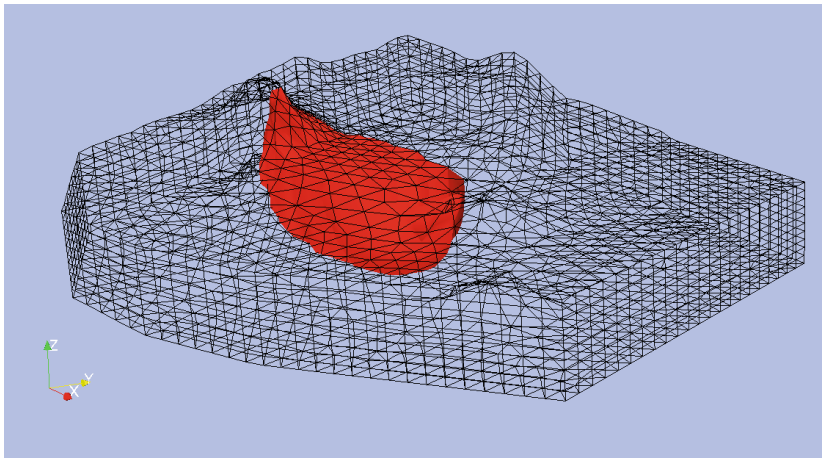
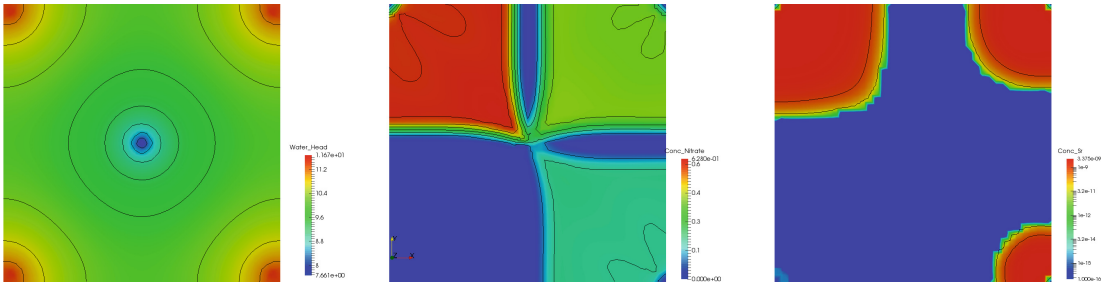


Fig. 8. Modelling the evolution of a contaminant plume from a surface waste disposal in a realistic geological media.

options of the platform to attach data to mesh entities is widely used to impose the boundary conditions, express the heterogeneity of the domain properties and take into account various objects affecting the flow, namely wells, drains, rivers and lake, pollution sources.

Another application is the reactive transport modelling using a combination of a domestic flow and transport code with a third-party chemical code PHREEQC [1]. On Fig. 9 the results of five-spot test case are shown. The domain is a parallelepiped $[-100; 100] \times [-100; 100] \times [-5; 5]$ (in meters). Four injection wells are located close to the corners of the domain: points $(-95; -95)$, $(95; -95)$, $(95; 95)$, $(-95; 95)$ in the X-Y plane with the well screen in the range $(-2; 2)$ meters along the Z-axis. A production well is located in the middle of the domain. The injection intensity is $10\,\mathrm{m}^3/\mathrm{day}$ for each injection well, the production rate is $40\,\mathrm{m}^3/\mathrm{day}$. The boundaries are impervious. The porous media is homogeneous and contains an ionic exchanger X. The injected liquids have an equal concentration of strontium $3.3 \cdot 10^{-9}$ Mol/l while the concentration of



(a) (b) (c)

Fig. 9. Modelling groundwater flow and reactive transport of solute containing sodium nitrate and strontium in the presence of ionic exchanger on the rock ($T = 3000$ days). (a) Water head; (b) Nitrate concentration in groundwater; (c) Strontium concentration in groundwater.

sodium nitrate varies from zero (lower left injection well) to 0.6 Mol/l (upper left injection well) with a step 0.2 Mol/l (thus having 0.2 Mol/l in the lower right and 0.4 in the upper right injection well). The initial solution in the media also contains K, Cl, Ca, Mg, S, C in equilibrium with the media. The results of the modelling are in good agreement with observed experimental data showing a decrease in the strontium attenuation caused by the presence of sodium nitrate in the injected solution. On Fig. 9 one can see that the higher the sodium nitrate concentration will be, the less will strontium be sorbed causing a quicker pollution propagation. Note that in sequential mode the modelling of this problem for 1000 days on a very coarse mesh containing 1600 hexahedral cells took around 3 h on an Intel CoreI7 machine substantiating a strong need in parallelization.

Conclusions

In conclusion we would like to formulate the major benefits of INMOST:

- Cross-platform code;
- Supports parallel mesh generation;
- Supports various input/output mesh formats (.gmsh, .vtk, .pvtk, .gmv, internal .pmf);
- Mesh can be distributed and redistributed in parallel (works with Zoltan, Parmetis and internal partitioner);
- Full set of mesh elements;
- Supports element markers and tag data of different types (integer, double, byte, element);
- Basis for parallel grid modification (is under development now).

Acknowledgements. This work has been supported in part by RFBR grants 14-01-00830, 15-35-20991.

References

1. Charlton, S.R., Parkhurst, D.L.: Modules based on the geochemical model PHREEQC for use in scripting and programming languages. Comput. Geosci. **37**(10), 1653–1663 (2011)
2. Freedman, V.L., et al.: A high-performance workflow system for subsurface simulation. Environ. Model. Softw. **55**, 176–189 (2014)
3. INMOST: a toolkit for distributed mathematical modelling. http://www.inmost.org
4. INMOST: user guides. http://wiki.inmost.org
5. INMOST: an online documentation. http://doxygen.inmost.org
6. Kapyrin, I., Vassilevski, Y., Utkin, S.: Concept of the design and application of the GeRa numerical code for radioactive waste disposal safety assessment. Voprosy Atomnoi Nauki i Tekhniki. Ser.: Math. Model. Phys. Process Iss. **4**, 44–54 (2014)
7. Nikitin, K.D., Terekhov, K.M., Vassilevski, Y.V.: A monotone nonlinear finite volume method for diffusion equations and multiphase flows. Comput. Geosci. **18**(3), 311–324 (2014)
8. Panday, S., Langevin, C.D., Niswonger, R.G., Ibaraki, M., Hughes, J.D.: MODFLOW-USG version 1: An unstructured grid version of MODFLOW for simulating groundwater flow and tightly coupled processes using a control volume finite-difference formulation: U.S. Geological Survey Techniques and Methods, Book 6, chap. A45, p. 66 (2013)
9. Vassilevski, Y.V., Konshin, I.N., Kopytov, G.V., Terekhov, K.M.: INMOST - a software platform and a graphical environment for development of parallel numerical models on general meshes. Moscow State University Publishing, Moscow, p. 144 (2013)