EPJ Web of Conferences **108**, 02018 (2016)
DOI: 10.1051/epjconf/201610802018
© Owned by the authors, published by EDP Sciences, 2016

Coordinate Systems, Numerical Objects and Algorithmic Operations of Computational Experiment in Fluid Mechanics

Alexander Degtyarev[1,a] and Vasily Khramushin[1,b]

[1] *St. Petersburg State University, Universitetskii prospect 35,198504 Saint-Petersburg, Russia*

Abstract. The paper deals with the computer implementation of direct computational experiments in fluid mechanics, constructed on the basis of the approach developed by the authors. The proposed approach allows the use of explicit numerical scheme, which is an important condition for increasing the efficiency of the algorithms developed by numerical procedures with natural parallelism. The paper examines the main objects and operations that let you manage computational experiments and monitor the status of the computation process. Special attention is given to a) realization of tensor representations of numerical schemes for direct simulation; b) realization of representation of large particles of a continuous medium motion in two coordinate systems (global and mobile); c) computing operations in the projections of coordinate systems, direct and inverse transformation in these systems. Particular attention is paid to the use of hardware and software of modern computer systems.

1 Introduction

The problems of the Computational Fluid Dynamics (CFD) are among the most complex for implementation on modern computers both because of their strong connectivity and because of the complex nature of the problem. Not surprising that a large number of "grand challenges" to one degree or another gets reduced to CFD problems.

Traditionally, the calculation of flows is associated with the solution of Navier-Stokes equations. The problems in this area can be divided into three broad groups [1]:

1. Incorrectness of the Navier-Stokes equations underlying the simulation, since it is a certain idealization of real processes.
2. Idealization is also present in the geometry of these flows. This concerns sharp edges, corners, etc. in calculations. Such ideal forms were due to the method of approximation using large meshes. This causes a very big problem in the solution, because in reality most of these features are absent. Currently, the size of the mesh may be significantly reduced, which means that the methods of dealing with the problems that arise should be different.
3. Every time when the current problems are mapped on a new computer architecture, there arise porting problems of such applications. Since long ago it became clear that the appearance of

[a] e-mail: a.degtyarev@spbu.ru
[b] e-mail: v.khram@gmail.com

new architectures is related with the appearance of new programming environments, which means the development of new libraries for currents will also be needed. Thus, we need to approach the problem of fluid dynamics programming in a different way.

A distinctive feature of the solution of the first problem is the traditional use of the implicit numerical schemes. Such a way reduces the solution to linear algebra problems that do not allow to efficiently control nonstationary processes and qualitative changes of the continuum for the considered complex problem. Moreover, the disadvantage of the implicit schemes as compared with the explicit ones is in the raise of complexity and parallelization problems. We know that it is a price to pay for the stability of the numerical scheme. However, the attractiveness of the explicit schemes has led to the development of an approach based on the description of the behavior of many large liquid particles [2]. Problems of numerical schemes instability in this case are avoided by providing a large particle with additional degrees of freedom. Geometrically this requires the introduction of a dual basis for describing the state of a large particle [3], [4].

The second group of problems is connected just with the geometric description of these flows. The efficiency of the calculations (especially in the case of direct numerical schemes) depends significantly on the description of the modeled objects, computational meshes and possibilities of their transformation and manipulation [5], [6]. This article is focused on the construction of such formal computational basis. Special attention is paid to programming procedures, which take into account the architecture of the computing systems.

2 The subject of application

The algorithmic realization of direct numerical simulations using explicit schemes has a beautiful historical analogy [2] in the form of calculus of fluxions by Isaac Newton. In the up-to-date algorithms, such implementation is represented as three-dimensional space numeric objects. Between them, connections are established for hydromechanics laws implementation.

Fluxions calculus[1] creates the basis of Newton classic mechanics for the motion in vector space and scalar time. Velocity $\vec{V}$ [m/s] becomes first fluxia in kinematics. It is differential (in accordance with Newton "moment") in multiplication with time step Δt [s] or just t. If we adapt such reasoning with reference to modern algorithms, then motion of control point in space is presented as follows[2]:

$$
{}^{+}\overrightarrow{A} = \overrightarrow{R} + \overrightarrow{V} \cdot t + \overleftarrow{a} \cdot (\overset{\wedge}{\mathbf{r}} + \overset{\wedge}{\mathbf{v}} \cdot t) = {}^{0}\overrightarrow{A} + (\overrightarrow{V} + \overleftarrow{a} \cdot \overset{\wedge}{\mathbf{v}}) \cdot t, \tag{1}
$$

where t is the calculated time interval [s]; $\overleftarrow{a}$ is the vector count in the local basis (of the large particle) [m^{-2}]; ${}^{+}\overrightarrow{A}$ and ${}^{0}\overrightarrow{A}$ are new and initial positions of the control point in the global reference system [m]; $\overrightarrow{R}$ is the location of the local basis in the global reference system [m]; $\overrightarrow{V}$ is the velocity of the local basis translational displacement [m/s]; $\overset{\wedge}{\mathbf{v}}$ [m^3/s] is the tensor of rotation and deformation of the basis axes of the tensor form[3] $\overset{\wedge}{\mathbf{r}}$ [m^3].

This equation presents a point shift during one step of the direct computational experiment. As shown, it can be represented in the form of the recalculation from the local coordinates of the point into the global reference system in accordance with the time interval t.

[1]*fluens, fluentis* are functions x, y, z on the argument of time t; *fluxio* $\dot{x}, \dot{y}, \dot{z}$ are time derivatitives x, y, z.

[2]In scalar notation: ${}^{+}A_x = {}^{0}A_x + (V_x + v_{xx}a^x + v_{xy}a^y + v_{xz}a^z) \cdot t;$
${}^{+}A_y = {}^{0}A_y + (V_y + v_{yx}a^x + v_{yy}a^y + v_{yz}a^z) \cdot t;$
${}^{+}A_z = {}^{0}A_z + (V_z + v_{zx}a^x + v_{zy}a^y + v_{zz}a^z) \cdot t.$

[3]Tensor object without indices here and later are marked by bold.

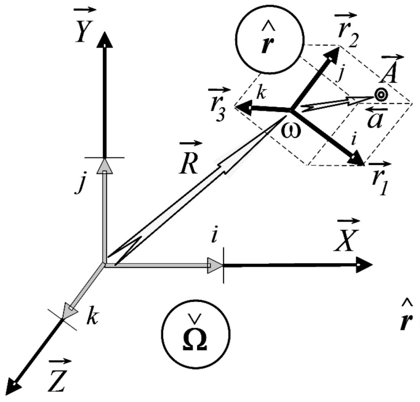
Renewed spatial field of velocities and deformations are used at conjugate stage of computational experiment. They are necessary for the control of the dynamic state as well as for redefining rheological characteristics of fluid, which are presented in the form of scalar, vector and tensor objects. In up-to-date complex computational models similar hydrodynamic characteristics are associated with spatial distribution of polarized dipole cores, initiated sources of vortexes, etc.

Obviously, the direct computational experiment efficiency directly depends on the manipulation efficiency with reference systems, numerical objects and algorithmic operations on these objects.

3 Geometric synthesis of computational objects and related algorithmic operations

Let us consider the principles of construction of the computational objects in direct computational experiment. The described approach allows to partly automate the validation of code writing and to improve its computational efficiency.

The geometrical construction of spatial problems includes scalar, vector and tensor numerical objects. Algorithmic procedures and arithmetic-logic operations are defined in the dimensional physical form and associate numerical objects and interpolation basis in a tensor mesh space.



$$\vec{A} = \vec{R} + \overleftarrow{a}\cdot\hat{\boldsymbol{r}} \qquad \overleftarrow{a} = (\vec{A} - \vec{R})\cdot\check{\boldsymbol{r}}$$

$$\check{\boldsymbol{r}} = \overleftarrow{r}^{\,i} = \begin{pmatrix} \overleftarrow{r}^x \\ \overleftarrow{r}^y \\ \overleftarrow{r}^z \end{pmatrix} = r^{ij} = \begin{pmatrix} r^{xx} & r^{xy} & r^{xz} \\ r^{yx} & r^{yy} & r^{yz} \\ r^{zx} & r^{zy} & r^{zz} \end{pmatrix}$$

$$\hat{\boldsymbol{r}} = \vec{r}_i = (\vec{r}_x \;\; \vec{r}_y \;\; \vec{r}_z) = r_{ij} = \begin{pmatrix} r_{xx} & r_{yx} & r_{zx} \\ r_{xy} & r_{yy} & r_{zy} \\ r_{xz} & r_{yz} & r_{zz} \end{pmatrix}$$

Figure 1. Geometry of global space ($\check{\Omega}$) and local basis ($\hat{\mathbf{r}}$ (ω)); i, j, k denote unitary vectors in the connected reference system

Elementary numerical objects are formed by non-coplanar basis vectors (Fig. 1). They serve to build indissoluble physical fields in the vicinity of adjacent mesh nodes ($^T_\Omega\vec{R}$) and centers of mass (ω). Products of vector and tensor quantities are performed with convolution, i.e. by summation over a repeated index in the monomial product ($\overleftarrow{a} = \vec{a}\cdot\check{\mathbf{r}}$ or $a^j = a_i \cdot r^{ij}$), the transition to local basis and back ($\vec{a} = \overleftarrow{a}\cdot\hat{\mathbf{r}}$ or $a_k = a^j \cdot r^{jk}$). The latter occurs at the return to absolute coordinates.

The proposed notation is similar to that in [7] and [8]. The symbol notations and the principles of their construction are summarized below.

– A Left upper index marks the current time, which may be indicated by a capital letter $^T\Omega$ in absolute terms or the calculated step in time ^tR. In addition, badges $^+\omega$ and $^-\omega$ designate links to the next or previous time interval.

– A Left low index marks a location in the mesh space ${}_{\{X,Y,Z\}}\overset{\wedge}{\mathbf{r}}$, or links to adducent knots ${}_{\{+\}}\overset{\wedge}{\mathbf{r}}$, or centers of mass of liquid particles ${}_{\{-\}}\overset{\vee}{\mathbf{r}}$. It is performed on conjugate stages of the computational experiment.

Right indices connect vector and tensor components in absolute and local bases. They serve to a strict definition of the dynamics and deformation of numeric cells (particles of a continuous medium).

– Low right indices, tensor "box" and the right arrow show the belonging to an absolute coordinate system (Fig. 1). For example, the tensor $\overset{\wedge}{\mathbf{r}}$ [m^3] is a collection by columns of basis vectors $\overrightarrow{r_i}$ in matrices of geometric transformations (like $\overrightarrow{a} = \overleftarrow{a} \cdot \overset{\wedge}{\mathbf{r}}$ [m]).

– Upper right indices mark projections inside mobile and deformable mesh cells. The display of unitary vectors of absolute coordinates lies in row vectors in matrix of inverse coordinate transformations, $\overset{\vee}{\mathbf{r}} = \overleftarrow{r^j} = r^{jk} = \overset{\wedge}{\mathbf{r}}^{-1}$ [m^{-3}]. They are marked by tensor "tick" and vector left arrow $\overleftarrow{a} = \overrightarrow{a} \cdot \overset{\vee}{\mathbf{r}}$ [m^{-2}].

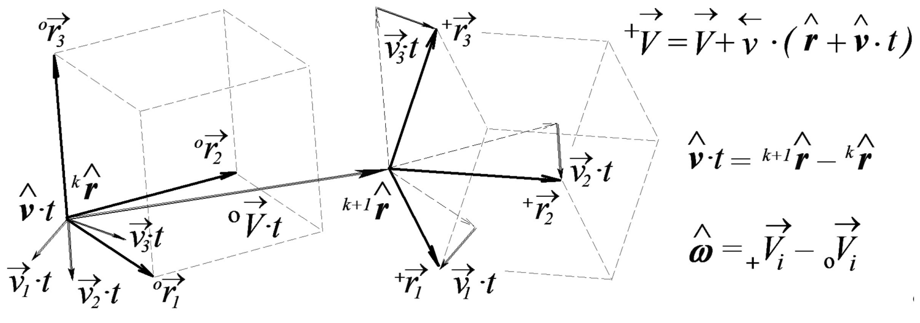
– Capital letters are used for big numerical values measured in scale of global space (Ω) and general absolute time (T);

– Lowercase letters are used for especially small quantities or finite differences which are commensurable with the physical dimensions of local bases of particle continuum ω, as well as in the range of the current time step t.

The absolute time T can contain the Julian date and time from the beginning of the day[4]: ${}^{k}T = T + k \cdot t$. Absolute values in space may also be presented[5]: $\overrightarrow{A} = \overrightarrow{R} + \overleftarrow{a} \cdot \overset{\wedge}{\mathbf{r}}$ [m] (geographical and other generalized coordinates). The need of involvement of absolute encoder in space ${}^{T}_{\Omega}\overrightarrow{R}$ and time T is eliminated in the balanced numerical schemes. In this case, the use of numerical values at nodes and centers of mass of conjugate mesh cells is sufficient at all stages of calculations $\hat{r}$ [m^3].

Kinematics of internal streams is defined by the speed difference tensor (Fig. 2). It is given on the large liquid particles basis vectors form shifted in time,

$$\overset{\wedge}{\mathbf{v}} \cdot t = \overrightarrow{v_i} \cdot t = {}_{\Delta}^{\Delta}\overrightarrow{r_i} = {}_{+}^{t}\overrightarrow{r_i} - {}_{\Omega}^{0}\overrightarrow{r_i}, \tag{2}$$



${}^{o}\overrightarrow{r_3}$
${}^{o}\overrightarrow{r_2}$
${}^{k}\hat{r}$
$\hat{v} \cdot t$
$\overrightarrow{v_3} \cdot t$
${}^{o}\overrightarrow{V} \cdot t$
$\overrightarrow{v_1} \cdot t$
$\overrightarrow{v_2} \cdot t$
${}^{o}\overrightarrow{r_1}$
$\overrightarrow{v_3} \cdot t$
${}^{+}\overrightarrow{r_3}$
${}^{k+1}\hat{r}$
$\overrightarrow{v_2} \cdot t$
${}^{+}\overrightarrow{r_2}$
${}^{+}\overrightarrow{r_1}$
$\overrightarrow{v_1} \cdot t$
${}^{+}\overrightarrow{V} = \overrightarrow{V} + \overleftarrow{v} \cdot (\hat{r} + \hat{v} \cdot t)$
$\hat{v} \cdot t = {}^{k+1}\hat{r} - {}^{k}\hat{r}$
$\overset{\wedge}{\omega} = {}_{+}\overrightarrow{V_i} - {}_{0}\overrightarrow{V_i}$

Figure 2. Movement of basis vectors of calculation cell in space

[4]The real time is set by the numeric structure `Event` with Julian data: `D` (from 4713 BC), and local time in hours from the day beginning: `T`.

[5]In software environment points in global coordinates (`Point`) are separated with free vectors in local bases (`Vector`). It unifies computing operations with tensor numerical objects `Tensor` and `Basis`.

The tensor $\overset{\wedge}{\mathbf{v}}$ [m^3/s] defines the current speed of the displacement of the liquid particle basis vectors on a local scale (lowercase letters) that is measured in the projection of the global coordinate system (lower indices). The independent convective rates tensor describing the local motion of the fluid is obtained after transformation of the velocity tensor reference frame[6] to the local basis of the large liquid particle (geometric normalization): $\overset{<}{\mathbf{v}}=\overset{\wedge}{\mathbf{v}} \cdot \overset{\vee}{\mathbf{r}}$ [1/s].

The tensor ($\overset{<}{\mathbf{v}}$ [1/s]) contains the extended set of kinematic elements of the differential equations with cross derivative components of deformable liquid particle motion:

$$\overset{<}{\mathbf{v}}=\overset{\wedge}{\mathbf{v}} \cdot \overset{\vee}{\mathbf{r}}= v_{ij}r^{jk} = \begin{pmatrix} v_{xx}r^{xx} + v_{xy}r^{yx} + v_{xz}r^{zx} & v_{xx}r^{xy} + v_{xy}r^{yy} + v_{xz}r^{zy} & v_{xx}r^{xz} + v_{xy}r^{yz} + v_{xz}r^{zz} \\ v_{yx}r^{xx} + v_{yy}r^{yx} + v_{yz}r^{zx} & v_{yx}r^{xy} + v_{yy}r^{yy} + v_{yz}r^{zy} & v_{yx}r^{xz} + v_{yy}r^{yz} + v_{yz}r^{zz} \\ v_{zx}r^{xx} + v_{zy}r^{yx} + v_{zz}r^{zx} & v_{zx}r^{xy} + v_{zy}r^{yy} + v_{zz}r^{zy} & v_{zx}r^{xz} + v_{zy}r^{yz} + v_{zz}r^{zz} \end{pmatrix}. \quad (3)$$

Alternatively, such products can be presented in the form of complete differentiation $\overset{<}{\mathbf{v}}=\overset{\wedge}{\mathbf{v}} \;/\; \overset{\wedge}{\mathbf{r}}= {}^{\Delta}_{\omega}\overrightarrow{v_i}/{}^{\Delta}_{\omega}\overrightarrow{r_i}$ executed without artificial exceptions of "small" or convective elements in substantial derivative approximations. Thus correct physical interpretation of rheological characteristics of liquid and living conditions of currents is remained.

4 Construction of tensor numerical objects and modeling algorithms

Computing space is constructed on fixed nonregularized nodes (`Point`) in indexed set of mesh cells. Three-dimensional interpolation is carried out using Euclidean bases (`Base`). Inside and in the vicinity, coordination of physical laws with the use of free vectors (`Vector`) is realized.

Object-oriented programming allows to move the control of the basic math operations correctness at the stage of the source program compilation, if they are applied to uniquely determined numerical objects (in accordance with physical characteristics).

The main computational objects (see, e.g., [9]) are defined by the requirement of fast and independent performance of computational operations. Particular attention is paid to the possibility of quick adjustment of the calculations for the application of hybrid algorithms depending upon conditions of the physical phenomena existing in a local subdomain. The following objects are introduced:

```
typedef double Real;   // scalar quantity in global space and time
typedef double real;   // local or differential counts in space and time
struct Tensor;         // tensor object without contextual links for quick calculation
struct Base;           // location coordinates and related Euclidean basis
struct Cell;           // numerical cell with contextual links to adjacent particles
struct Point;          // point in scale of absolute reference system
struct Vector;         // free difference vector in scale of local reference system
struct Space;          // space of nodal elements for net area in general
struct Volume;         // set of free/moving and deformable cells
```

The control of the correctness of the mathematical operations with respect to the listed objects can be illustrated as follows. The automatic conversion of a complex variable `Vector` or `Tensor` to a local value `real` determines the length of the vector and the determinant of the matrix. Any other transformation are marked as wrong by a translator. The difference between values of type `Point` can

[6]Prohibition improving rank in product operation enables automatic permutation of factors in geometric transformations: $\overset{\wedge}{\mathbf{v}} \cdot \overleftarrow{a} =\overset{\wedge}{\mathbf{v}} \cdot(\overrightarrow{a}\cdot \overset{\vee}{\mathbf{r}}) =\overset{\wedge}{\mathbf{v}}\overset{\vee}{\mathbf{r}} \cdot\overrightarrow{a} =\overset{<}{\mathbf{v}} \cdot\overrightarrow{a}$ [m/s].

only give a value of `Vector`. Addition of objects of type `Vector` with `Vector` or `Point` values will result in the same type of data, and no other.

The development of modern computers goes rather towards adding more RAM, than towards improving the processing of large amounts of data. The basic data arrows `Space` and `Volume` have to be retained at current, previous and subsequent cuts in time for the technical support of the parallelization. This allows to synchronize the parallel processing in the case of complete separation of the stages of the experiment into independent physical processes, or to create any cycles for matching parameters of the computing environment in complex and ill-conditioned mathematical models, if necessary.

5 Conclusion

The solution of any computational problem consists of several stages:

– problem identification;
– formalization of the physical problem;
– construction of a computational procedure;
– creation of code for the chosen computer architecture.

At the transition between any successive stages, qualitative change in the essence of the described phenomenon can occur. Problems associated with incorrect mathematical models, numerical schemes to replace them, mapping problems on the architecture of a computer system, etc. are quite well known. For an adequate creation of a complete solution of the problem, it is necessary to develop consistent mapping of one stage onto another. The above approach is an attempt to adequately represent the final stage of solving the overall problem.

Acknowledgements

The research was carried out using computational resources of Resource Center Computer Center of Saint-Petersburg State University and supported by Russian Foundation for Basic Research (project N 13-07-00747) and St. Petersburg State University (projects N 9.38.674.2013, 0.37.155.2014).

References

[1] A. V. Bogdanov, A. B. Degtyarev and V. N. Khramushin, Computer Research and Modeling **7**, 3, 429–438 (2015)

[2] A. B. Degtyarev and V. N. Khramushin, Mathematical Modeling **26**, 11, 4–17 (2014)

[3] V. N. Khramushin, Vestnik of Saint-Petersburg University, Series 10 : Applied Mathematics. Computer Science. Control Processes, 2, 108–117 (2015)

[4] V. N. Khramushin, *3D Tensor Mathematics for the Computational Fluid Mechanics Experience* (FEB RAS, Vladivostok, 2005)

[5] I. D. Faux and M. J. Pratt,*Computational Geometry for Design and Manufacture (Mathematics and Its Applications)* (Ellis Horwood, 1981)

[6] A. J. McConnell, *Applications of Tensor Analysis* (Dover Publications, 2011)

[7] G. Astarita and G. Marrucci, *Principles of Non-Newtonian Fluid Mechanics* (Mc Graw-Hill, 1974)

[8] N. E. Kochin, *Vector calculus and beginnings of tensor calculus* (9th ed., Nauka, Moscow, 1965)

[9] Code for numerical objects construction and functions of three-dimensional tensor mathematics for computational experiments in fluid mechanics (Tensor), Saint-Petersburg State University. FIPS N 2013619727