MSC2010: 65M60, 35Q74, 65Y05

© ***S. P. Kopysov, I. M. Kuz'min, A. K. Novikov, N. S. Nedozhogin, L. E. Tonkov***

# RADIAL BASIS FUNCTION FOR PARALLEL MESH-TO-MESH INTERPOLATION IN SOLVING FLUID-STRUCTURE INTERACTION PROBLEM[1]

In strongly coupled fluid-structure interaction simulations, the fluid dynamics and solid dynamics problems are solved independently on their own meshes. Therefore, it becomes necessary to interpolate the physical properties (pressure, displacement) across two meshes. In the present paper, we propose to accelerate the interpolation process by the method of radial basis functions using the matrix-free solution of the system of equations on a GPU. Also, we reduce the number of equations in the system by using an adaptive algorithm for choosing interpolation points. The adaptive algorithm allows to reduce the number of equations of the interpolation system while preserving the quality of the interpolation. Estimation of the effectiveness of reducing the computational costs based on the matrix-free approach to solving the system, as well as evaluating the quality of interpolation, was carried out using the simulation of the problem of modeling the flow of fluid with a supersonic deformable nozzle.

*Keywords*: parallel computing, hybrid HPC platforms, fluid-structure interaction, radial basis functions, layer-by-layer partitioning.

DOI: 10.20537/2226-3594-2018-51-02

## Introduction

Several approaches are used to simplify algorithms for the numerical solving of the Fluid-Structure Interaction (FSI) problem such as the decreasing of the problem size, **algorithms accelerating the computations**, the analysis methods and the use of the properties of subtasks.

Let us consider the variants of decreasing the computational costs at the interpolation of data between non-matching meshes: the decrease of the interpolation data; the parallelization of algorithms taking into account the geometry of the extended boundaries of axisymmetric bodies.

The main methods for interpolation on non-matching meshes for the FSI simulations are surveyed in [1, 2]. We consider a method based on radial basis functions (RBF) [3], where the coefficients of the interpolant are found from the system of equations, the matrix of which is formed using a radial basis function. The choice of the function determines the condition number and density of the matrix, and, as a result, the computational complexity of solving the system of equations.

The RBF interpolation has the following advantages:

- it does not require any mesh connectivity information;
- it requires solving a sparse system of equations, especially with the compact basis functions;
- it can be efficiently parallelized.

This paper is structured as follows. Section 2 briefly describes the RBF interpolation scheme for the FSI problem. The next section presents a new approach based on layer-by-layer mesh partitioning for reducing the problem size. The fourth section describes a matrix-free solution of the interpolation problem on a GPU.

## § 1. RBF interpolation for FSI problems

Let $P^d_{Q-1}$ be the $d$-dimensional space of the polynomials of a degree smaller than $Q-1$ and $p_1, \ldots, p_q$ be a basis in this space. The main idea of the RBF method is to find the required interpolation function as a linear combination of the following functions:

$$w(x_i) = \sum_{j=1}^{n} \alpha_j \phi(\|x_i - x_j\|) + \sum_{l=1}^{q} \beta_l p_l(x_i), \tag{1.1}$$

[1]This work was supported by RFBR (projects no. 16–01–00129, no. 17–01–00402).

where $q$ is additional degrees of freedom and the coefficients $\alpha_i$ and the polynomials $p_l(x_i)$ satisfy

$$\sum_{j=1}^{n} \alpha_j p_l(x_j) = 0, \quad 1 \leqslant l \leqslant q. \tag{1.2}$$

The solution of the system exists and is unique if

$$p(x_j) = 0, \text{ for all } 1 \leqslant j \leqslant n \text{ and } p \in P^d_{Q-1} \text{ implies } p = 0. \tag{1.3}$$

The system of equations (1.1)–(1.3) is always solvable if $\phi$ is a positive-definite radial basis function.

Let $\Omega$ be the domain with the given pressure $p_\Omega$. The domain with the required pressure is denoted by $\Phi$. The pressure interpolation between the meshes can be expressed in the matrix form as follows:

$$\begin{bmatrix} W_{\Omega\Omega} & P_\Omega \\ P_\Omega^T & O \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} p_\Omega \\ 0 \end{bmatrix} \quad \text{or} \quad A\,\gamma = b, \tag{1.4}$$

here $W_{\Omega\Omega}$ is the $n_\Omega \times n_\Omega$ matrix consisting of the elements $\phi(\,\|\mathrm{x}^i_\Omega - \mathrm{x}^j_\Omega\|\,)$ and $1 \leqslant i,\, j \leqslant n_\Omega$; $P_\Omega$ is the column matrix consisting of the elements $\begin{bmatrix}1 & \mathrm{x}^i_\Omega\end{bmatrix}$; $\alpha$, $\beta$ are the coefficients of the interpolant; $n_\Omega$ is the number of interpolation points of the domain. The target pressure vector $p_\Phi$ is obtained by the matrix-vector product

$$p_\Phi = [W_{\Phi\Omega} P_\Phi] \begin{bmatrix} \alpha \\ \beta \end{bmatrix}, \tag{1.5}$$

where $W_{\Phi\Omega}$ is the $n_\Phi \times n_\Omega$ matrix consisting of the elements $\phi(\,\|\mathrm{x}^i_\Phi - \mathrm{x}^j_\Omega\|\,)$, $1 \leqslant i \leqslant n_\Phi$ and $1 \leqslant j \leqslant n_\Omega$; $n_\Phi$ is the number of interpolation points on the domain $\Phi$; $P_\Phi$ is the column matrix $\begin{bmatrix}1 & \mathrm{x}^i_\Phi\end{bmatrix}$. The dimension of the matrices $P_\Omega$ and $P_\Phi$ depends on the type of basis functions. For example, the dimension of the matrices $W_{\Omega\Omega}$ and $W_{\Phi\Omega}$ for the global radial basis function Thin-Plate Spline is $3 \times n_\Omega$ and $3 \times n_\Phi$, respectively.

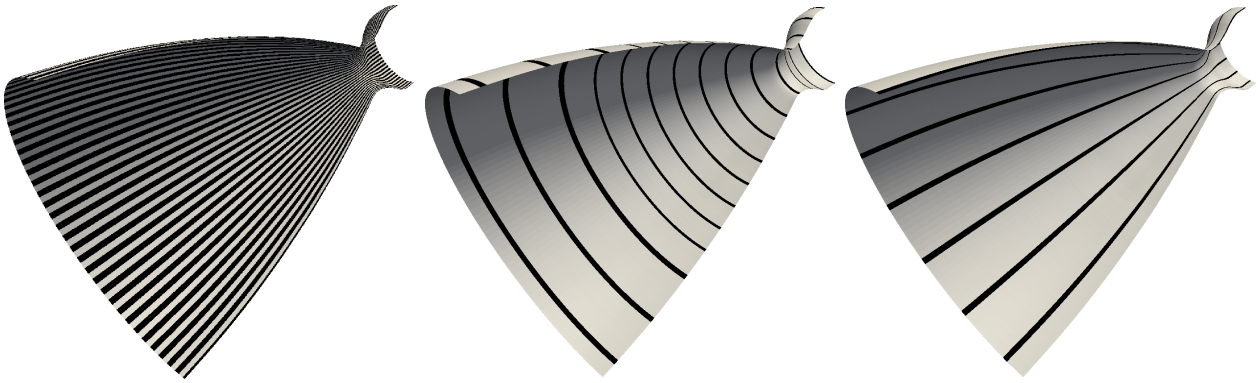
Solving the system of equations (1.4) is the most computationally expensive part of the interpolation. In [4], it was shown that the choice of basis functions affected both the quality of the interpolation and the solution time. The functions providing more accurate interpolation may require a large amount of time for the solution. The computational cost can be optimized by (i) reducing the system and (ii) parallelizing the steps of the preconditioning and solution of sparse/dense systems of equations.

## § 2. Reducing the size of the system of equations

In this section, we demonstrate reducing the size of the system of equations for the fluid-structure interaction of a supersonic flow with a nozzle wall that has a high geometric expansion ratio [5]. The boundary along which the computational data are interpolated is quite long and the pressure is irregularly distributed along the boundary $\Omega$ (the nozzle wall). The solution of the above problems is considered within the framework of the layer-by-layer mesh partitioning method proposed in our previous work [6]. The method provides a conflict-free data access during the parallel summation of the components of the finite element vectors in the shared memory of the multi-core computing systems.

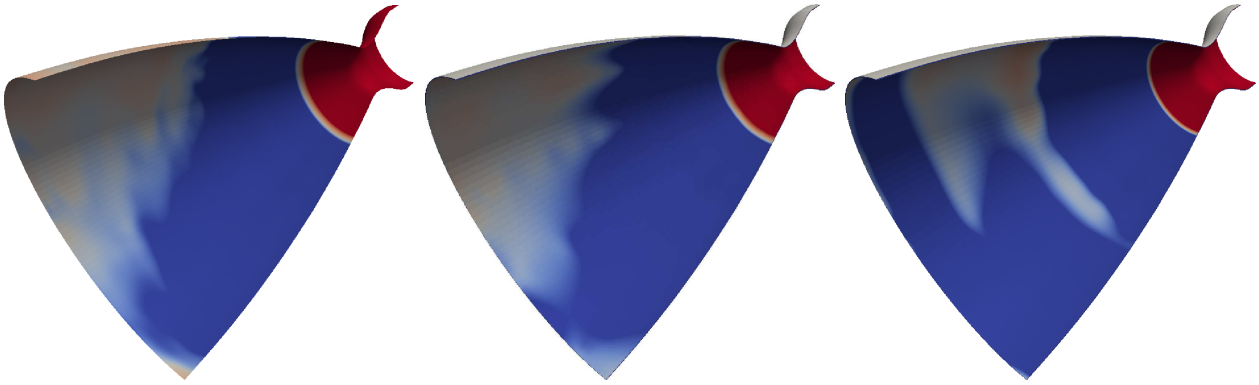
Let us divide the interface part $\Gamma_{\Omega^h} = \partial\Omega^h$ of the mesh $\Omega^h$ into layers. To do this, we use the neighborhood criterion where any two mesh cells are considered adjacent if they have at least one common node.

The considered physical area and the computational mesh are symmetrical. Therefore, the choice of the initial set of interpolation points is carried out in accordance with the distribution of the layers. Here, there are two possibilities for selecting layers: along the generatrix and along the directrix.



(a) (b) (c)

**Fig. 1.** The interface boundary partitioned into 150 layers (a); the partition into 15 layers parallel to the directrix (b) and the partition into 15 layers by the surface generatrix (c)



(a) (b) (c)

**Fig. 2.** The given pressure distribution at the boundary of the domain $\Omega$ (a); the pressure distribution obtained at the use of 15 out of 150 layers at the radial partition (b) and at the partition along the generatrix (c)

The mesh $\Gamma_{\Omega^h}$ is the discrete description of the rotation surface $\Gamma_\Omega$ with the closed directrix. To form layers in parallel to the directrix $\Gamma_\Omega$ (see Fig. 1, *b*) or along the surface generatrix (see Fig. 1, *c*), we use the algorithm proposed in [6].

The layer-by-layer partitioning is used to reduce the number of interpolation points. To construct the interpolant, we choose those layers of the interface surface which most accurately represent the distribution of the interpolated data (pressure). Figure 1 shows the partitioning of the surface mesh into 150 layers. The dark layers correspond to 15 layers involved in the pressure interpolation (Fig. 1, *b*, *c*).

The quality of interpolation is compared for the local $\phi(\|\mathrm{x}\|) = 1 - \|\mathrm{x}\|$, $\phi(\|\mathrm{x}\|) = (1 - \|\mathrm{x}\|)^2$, the global $\phi(\|\mathrm{x}\|) = e^{-\|\mathrm{x}\|^2}$, $\phi(\|\mathrm{x}\|) = \|\mathrm{x}\|^2 \log \|\mathrm{x}\|$ basis functions and Inverse Distance Weighting (IDW) [7], using different partitions and numbers of layers. The quality of the pressure interpolation can be estimated as the relative error computed by the ratio of the norms of the resultant forces of the pressure on the interface boundary.

Table 1 shows the results for the pressure interpolation in parallel to the directrix (Radial partitioning) and along the surface generatrix (Longitudinal partitioning). In the second column, the evaluation of the interpolation quality is given for all possible interpolation points of $\Gamma_{\Omega^h}$.

The quality of the interpolation with the data reduction depends not only on the number of interpolation points but also on the choice of the points (Fig. 2). When compact basis functions in the form $(1 - \|\mathrm{x}\|)^2$ are used, the orientation of the pressure distribution after the interpolation depends on the partitioning. The best interpolation is achieved for the radial partitioning of the domain. The RBF method using the global basis function $\phi(\mathrm{x}) = \|\mathrm{x}\|^2 \log \|\mathrm{x}\|$ gives the best results.

It allows to reduce the number of equations in system (1.4) by a factor of 15 with the acceptable quality of the interpolation. The IDW interpolation gives the greatest error, even in the case of the full data.

**Table 1.** Relative error of the pressure interpolation, %

| $n_\Omega$ | 28800 | 9600 | 5760 | 2880 | 960 |
|---|---|---|---|---|---|
| Longitudinal distribution | | | | | |
| $1 - \|\mathrm{x}\|$ | 0.24 | 3.79 | 10.4 | 15.2 | 349.6 |
| $(1 - \|\mathrm{x}\|)^2$ | 0.78 | 7.79 | 69.3 | 74.3 | 365.4 |
| $\mathrm{IDW}_{p=3}$ | 14.2 | 50.6 | 105 | 189 | 407.0 |
| $e^{-\|\mathrm{x}\|^2}$ | 0.54 | 1.18 | 2.68 | 8.04 | 147.6 |
| $\|\mathrm{x}\|^2 \log \|\mathrm{x}\|$ | 0.01 | 0.23 | 1.07 | 5.95 | 28.61 |
| Radial distribution | | | | | |
| $1 - \|\mathrm{x}\|$ | 0.24 | 0.34 | 0.92 | 10.8 | 214.3 |
| $(1 - \|\mathrm{x}\|)^2$ | 0.78 | 1.58 | 9.34 | 15.9 | 36.5 |
| $\mathrm{IDW}_{p=3}$ | 14.2 | 53.9 | 111 | 205 | 534.2 |
| $e^{-\|\mathrm{x}\|^2}$ | 0.54 | 1.07 | 3.99 | 5.41 | 101.6 |
| $\|\mathrm{x}\|^2 \log \|\mathrm{x}\|$ | 0.01 | 0.01 | 0.21 | 1.13 | 21.6 |
| Adaptive distribution | | | | | |
| $1 - \|\mathrm{x}\|$ | 0.24 | 1.23 | 1.14 | 0.93 | 4.89 |
| $(1 - \|\mathrm{x}\|)^2$ | 0.78 | 0.67 | 1.75 | 2.41 | 1360 |
| $\mathrm{IDW}_{p=3}$ | 14.2 | 26.1 | 27.3 | 30.8 | 85.1 |
| $e^{-\|\mathrm{x}\|^2}$ | 0.54 | 0.94 | 1.12 | 0.82 | 9.61 |
| $\|\mathrm{x}\|^2 \log \|\mathrm{x}\|$ | 0.01 | 0.37 | 1.25 | 1.31 | 3.65 |

The obtained matrix $A$ is ill-conditioned. When using global basis functions, the condition number depends on the number and location of the interpolation points, as well as on the type of the basis functions. The distance between the interpolation points influences the condition number. As the number of interpolation points increases, the distance between them decreases, and the condition number increases.

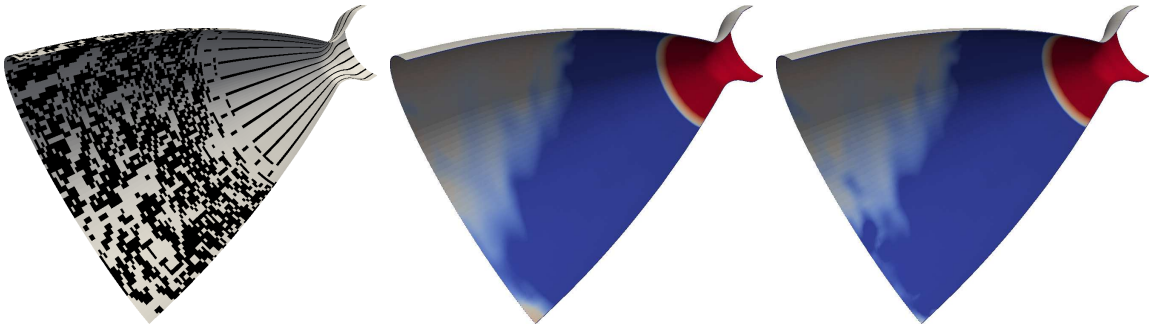
For interpolation it is practical to use an adaptive algorithm. It should be noted that the interpolation error is not the only reason for its use. In [8], it was shown that in the RBF-based interpolation, the optimal distribution for the case of a two-dimensional space was the asymptotically uniform distribution of points. In the considered case, at the initial non-uniform distribution of the data, the adaptive choice of the interpolation points is preferable since it allows obtaining the interpolation point distribution providing a minimum error. Note that in the case of the asymptotically uniformly distributed interpolation points, theoretically, an arbitrarily high error can be obtained for some functions; however, for others it is unattainable. Even the use of greedy algorithms [9] does not provide an optimal interpolation by the RBF method for an arbitrary function.

The aim of the adaptive algorithm is to achieve a sufficiently small error for the interpolation with the use of $n_a \ll n_\Omega$ points only, where $n_\Omega$ is the initial number of interpolation points.

The adaptive algorithm starts with a very small number of points and then refines the data set by adding new points of the interpolation where the observed interpolation error is largest. The algorithm constructs an interpolant for the reduced set of points, which reproduces the interpolation function in certain tolerance limits. Furthermore, limits can be imposed on the number of iterations and the total number of interpolation points ($n_a$).

In the fluid dynamics problem on a flow in a supersonic overexpanded nozzle, the line $\ell = \{(\arg\min_x p(x, \varphi_*), \varphi_*) \in \Gamma : 0 \leqslant \varphi_* < 2\pi\}$ divides the set of interpolation points $\Omega$ into $\Gamma_1$, where $p = p(x)$ is the axial symmetry region, and $\Gamma_2$, where $p = p(x, \varphi)$ (Fig. 3–5, *a*).

Let us apply the adaptive algorithm to the pressure interpolation for the considered fluid-structure interaction problem. At the first step, we apply it to the layers to which the set of



(a) (b) (c)

**Fig. 3.** Pressure distribution: (a) adaptive distribution with $n_\Omega = 9600$; (b) $e^{-\|x\|^2}$; (c) $1 - \|x\|$

interpolation points is divided, i.e. we reduce the set of interpolation points along the coordinate $x$. Thus, from all the layers we leave the minimum number of layers satisfying the error of interpolation. At the second step, we apply this algorithm to each remaining layer to select the interpolation points with the largest error in the coordinate $\varphi$. Thus, we reduce the number of the interpolation points preserving the interpolation error.

**Adaptive algorithm for selecting interpolation points**

Let us

1) select an initial set of layers and solve $A\gamma = b$ for $n_a$ init points;

2) form and solve the linear system of the interpolation coefficients for the initial layers $A\gamma = b$;

3) evaluate the interpolant at all $n_\Omega$ data points $p_\Phi = A\gamma$;

4) compute the residual vector or find errors $E(\mathrm{x}) = p(\mathrm{x}) - p_\Omega(\mathrm{x})$;

5) check the stopping criteria, and if they are not satisfied, increase the iteration count and add new points with the largest error $E(\mathrm{x})$.

The adaptation is successfully completed if the residual is smaller than the given tolerance.

In addition to the pressure $p = p(x, \varphi)$ determined on $\Omega$, the gradient $\operatorname{grad} p$ is also known. It is used as an indicator of the addition of interpolation points. Layers and interpolation points are added to the minimal set of points when they are located in regions of the largest gradient.

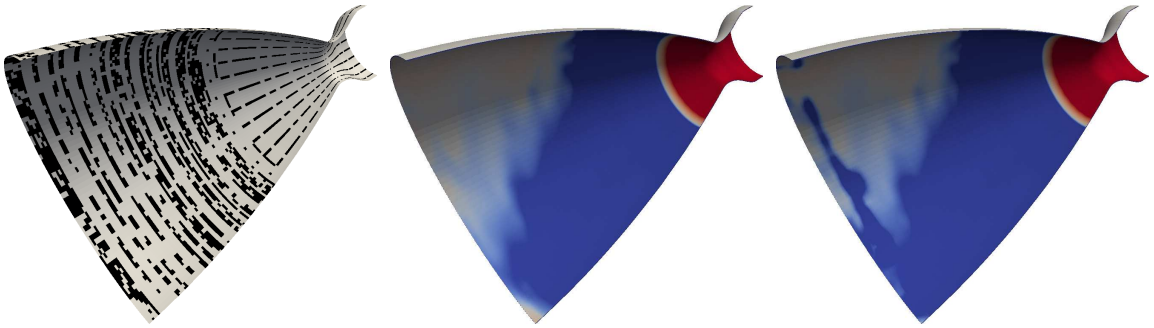
In the absence of any additional information on the distribution of the interpolated data (e.g., in the case of the pressure interpolation it is a gradient), the indicator of the interpolation error is formed on the basis of local basis functions or by the IDW method.

Table 1 shows the error of the pressure interpolation using the adaptive distribution of interpolation points. It should be noted that the adaptive choice of interpolation points increases of the condition number of the matrix of the system of equations (1.4), and, thus, the iterative solution process converges slowly.

The adaptive distribution of the interpolation points using the local basis function $1 - \|x\|$ substantially reduces the error for any number of interpolation points, and another local basis function $(1 - \|x\|)^2$ has a large error for a small number of points. The Gaussian global function (Fig. 5, *b*) shows good results for a small number of the adaptively chosen points $n_\Omega = 960$.

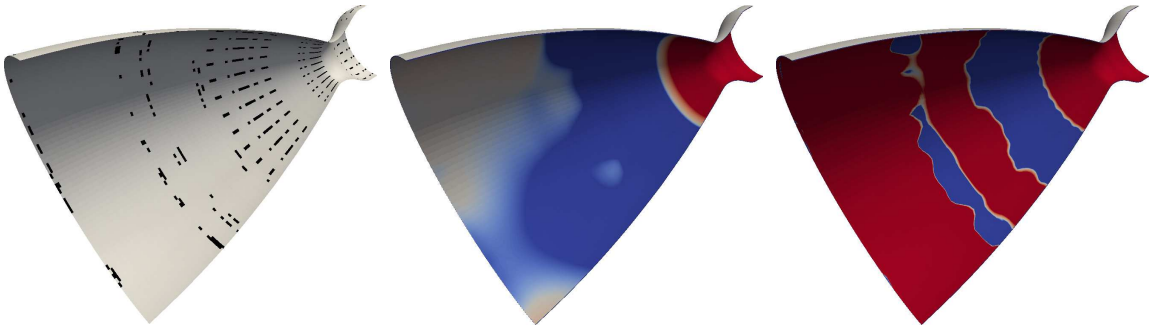
## § 3. Matrix-free solution of interpolation problem on GPU

One of the specific features of the system (1.4) is a dense matrix, which imposes some restrictions on the GPU use due to the small capacity of the available GPU memory. The problem can be resolved by (i) using several GPUs, thereby increasing the total memory available for the system solution; (ii) solving the system of equations without the formation of a matrix (Matrix-Free Algorithm or MFA). In this case, the matrix elements are computed as they are required in the algorithm of the system solution. The solution of the system by the RBF method is possible without the formation of



(a) (b) (c)

**Fig. 4.** Pressure distribution: (a) adaptive distribution with $n_\Omega = 5760$; (b) $e^{-\|x\|^2}$; (c) $1 - \|x\|$



(a) (b) (c)

**Fig. 5.** Pressure distribution: (a) adaptive distribution with $n_\Omega = 960$; (b) $e^{-\|x\|^2}$; (c) $1 - \|x\|$

a matrix. since the matrix elements are computed by the chosen basis function. This improves the data locality and arithmetic intensity for matrices and vectors. The memory requirements and CPU-GPU communications are reduced. The efficiency of the algorithm can be improved if multi-GPUs are used in the similar way to that in [10].

Let us consider in more detail the MFA computing expenses. Table 2 shows the time of the sequential and parallel formation of the matrix $A$ of the system (1.4). In the MFA, the formation time is excluded. For comparison, the time of the solution of the system with an assembled matrix is given. The time of copying the matrix A of the system (1.4) to the GPU memory is also presented. In addition, the time is given for solving the system with the use of both the algorithm with an assembled matrix and the matrix-free solution algorithm.

The CPU parallelization is carried out with OpenMP. The solution of the system of equations on several GPUs is carried out by CUDA in conjunction with OpenMP. The system of equations is solved by the conjugate gradient method with the diagonal preconditioner [10]. The precision is equal to $10^{-6}$. In the computations, double-precision arithmetic is used. The analysis and performance estimations are performed on a computing node consisting of 2×quad-core Intel Xeon processor E5-2609, 2 × GeForce GTX 980 with 4Gb GDDR.

When the system of equations is solved using the assembled matrix on the CPU, the step of the matrix formation is added. The use of GPU increases the cost due to the necessity of copying the data to the GPU. When the system is solved using the MFA the cost is not increased because there in no need to copy the data. In the last line of Table 2, the total time is given for each of the above approaches.

The numerical computations show that the use of eight CPU threads within one computing node reduces the solution time almost by a factor of seven. One GPU allows to speed up solving the system by a factor of 250 compared with one CPU thead and by a factor of 50 compared with $8 \times$ CPU. The GPU efficiency increases with the increase of the system size. Using two GPUs reduces the time by a factor of 1.5 compared with one GPU and by a factor of 350 compared with the CPU. With an increase in the number of GPUs, the strong scalability can be provided only when

**Table 2.** The execution time of the interpolation for $e^{-\|x\|^2}$, s

| | | Number of equation | | | |
|---|---|---|---|---|---|
| | | 960 | 9600 | 19200 | 28800 |
| Forming $A$ | $1 \times$ CPU | 0.098 | 5.655 | 33.91 | 60.85 |
| | $8 \times$ CPU | 0.018 | 0.907 | 6.166 | 12.16 |
| | MFA | 0.0 | 0.0 | 0.0 | 0.0 |
| Copy of $A$ to GPU | $1 \times$ GPU | 0.009 | 0.312 | 0.842 | — |
| | $2 \times$ GPU | 0.006 | 0.125 | 0.433 | 1.393 |
| | MFA | 0.0 | 0.0 | 0.0 | 0.0 |
| System solution $A\gamma = b$ | $1 \times$ CPU | 3.343 | 592.8 | 4732 | 10359 |
| | $8 \times$ CPU | 0.271 | 86.82 | 946.6 | 2273 |
| | $1 \times$ GPU | 0.288 | 2.282 | 12.11 | — |
| | $2 \times$ GPU | 1.262 | 2.354 | 8.643 | 16.19 |
| | Matrix-free algorithm | | | | |
| | $8 \times$ CPU | 1.712 | 249.5 | 1489 | 4492 |
| | $1 \times$ GPU | 0.471 | 14.01 | 91.59 | 191.1 |
| | $2 \times$ GPU | 1.226 | 8.664 | 47.03 | 95.58 |
| Total time | $1 \times$ CPU | 3.438 | 598.5 | 4765 | 10419 |
| | $8 \times$ CPU | 0.288 | 86.82 | 952.7 | 2285 |
| | $1 \times$ GPU | 0.317 | 3.53 | 19.12 | — |
| | $2 \times$ GPU | 1.283 | 3.38 | 15.24 | 29.74 |
| | Matrix-free algorithm | | | | |
| | $8 \times$ CPU | 1.712 | 249.6 | 1489 | 4492 |
| | $1 \times$ GPU | 0.472 | 14.01 | 91.59 | 191.1 |
| | $2 \times$ GPU | 1.226 | 8.664 | 47.03 | 95.58 |

the sizes of the submatrices on each GPU are preserved.

The matrix-free solution of the system using $8 \times$ CPU reduces the solution time by a factor of 2.5. However, the solution with the assembled matrix is twice as fast as the matrix-free solution. When one GPU is used, the time for the matrix-free solution of the system of equations is 5 times larger than that for the solution with the assembled matrix, and in the case of using two GPUs, the matrix-free solution is 3 times longer. The speedup obtained at the use of one CPU thread is 55 times smaller than that when using one GPU and 110 times smaller than that when using two GPUs. It should be noted that the use of local basis functions with an introduced radius of influence increases the MFA efficiency.

Let us estimate the maximum size of the system, which can be solved using the MFA on a one GPU. For the matrix $A$ formation, the coordinates of the interpolation points are used. Then for interpolation in a three-dimensional space, it is necessary to allocate memory for the vector of coordinates of length equal to $n_\Omega \times 3$. The required memory size for solving the system with the assembled matrix is $n_\Omega \times n_\Omega$. The remaining vectors participating in the conjugate gradient method coincide for both algorithms. Thus, the memory size for interpolating the mesh data in the three-dimensional space is decreased by a factor of $n_\Omega/3$. The maximum system size solved by the MFA increases by the same factor. The algorithm of the conjugate gradient method with a diagonal preconditioner involves the use of memory to store a matrix of size $n_\Omega \times 3$ (the MFA) and six vectors $n_\Omega \times 1$. Thus, for solving the system using the MFA and double precision arithmetic, $n_\Omega \times (3+6) \times 8$ bytes are required. Consequently, the maximum size of a system for the GPU with a 4Gb GDDR is about $6 \times 10^8$ equations. Using two graphics cards, the possible size of the system is increased to $1.2 \times 10^9$ equations. Thus, for the dense matrices obtained on the basis of global basis functions, a parallel method of conjugate gradients is constructed. The computations are distributed among several GPUs. The use of the matrix-free approach makes it possible to remove any limitations on the amount of memory.

## § 4. Conclusion

We propose an algorithm for constructing a uniform distribution of interpolation points on an unstructured mesh in the interpolation based on RBF. For solving the problems on unstructured meshes, the adaptive algorithm is proposed for sampling the data for the interpolation based on the RBF method. The obtained results show that the interpolation on the uniformly distributed data is of high quality and applicable for meshes of super-large dimensions. The use of the adaptive data reduction based on the layer-by-layer partition of the mesh makes it possible to reduce the number of interpolation points, but requires additional information about the data. At the same time, the quality of the interpolation for the irregularly-spaced data is preserved both for global and for local basis functions. Using a matrix-free algorithm on large meshes significantly reduces the memory costs associated with the formation of the interpolation matrix. At the same time, the computation locality of the matrix-vector product computations increases when solving the system of equations by iterative methods. The solution of systems with dense matrices by the MFA on the CPU does not lead to any significant time reductions. Since the time of the matrix formation is less than 1% of the solution time, the use of the MFA in conjunction with the CPU is inefficient. The matrix-free approach is most effective when using a GPU, especially when it is not possible to achieve a large reduction of points without the interpolation quality loss. Using a GPU for solving larger systems of equations allows minimizing the cost of additional computations associated with the formation of the matrix elements.

A further increase in the efficiency of the MFA is associated with a decrease in the number of iterations of the algorithm for solving the system of equation by constructing effective parallel preconditioners and by using local basis functions with the radius of influence.

REFERENCES


1. Berndt M., Breil J., Galera S., Kucharik M., Maire P.-H., Shashkov M. Two-step hybrid conservative remapping for multimaterial arbitrary Lagrangian–Eulerian methods, *Journal of Computational Physics*, 2011, vol. 230, no. 17, pp. 6664–6687. DOI: 10.1016/j.jcp.2011.05.003
2. Farrell P.E., Piggott M.D., Pain C.C., Gorman G.J., Wilson C.R. Conservative interpolation between unstructured meshes via supermesh construction, *Computer Methods in Applied Mechanics and Engineering*, 2009, vol. 198, no. 33–36, pp. 2632–2642. DOI: 10.1016/j.cma.2009.03.004
3. de Boer A., van der Shoot M.S., Bijl H. Mesh deformation based on radial basis function interpolation, *Computers and Structures*, 2007, vol. 85, no. 11–14, pp. 784–795. DOI: 10.1016/j.compstruc.2007.01.013
4. De Boer A., Van der Shoot M.S., Bijl H. New method for mesh moving based on radial basis function interpolation, *ECCOMAS CFD 2006: Proceedings of the European Conference on Computational Fluid Dynamics*, Egmond aan Zee, Netherlands, 2006, pp. 1–16.
5. Wang T.-S., Zhao X., Zhang S., Chen Y.-S. Development of an aeroelastic modeling capability for transient nozzle flow analysis, *Journal of Propulsion and Power*, 2014, vol. 30, no. 6, pp. 1692–1700. DOI: 10.2514/1.B35277
6. Novikov A., Piminova N., Kopysov S., Sagdeeva Yu. Layer-by-layer partitioning of finite element meshes for multicore architectures, *Communications in Computer and Information Science*, 2016, vol. 687, pp. 106–117. DOI: 10.1007/978-3-319-55669-7_9
7. Shepard D. A two-dimensional interpolation function for irregularly-spaced data, *Proceedings of the 1968 23rd ACM National Conference*, 1968, pp. 517–524. DOI: 10.1145/800186.810616
8. De Marchi S., Schaback R., Wendland H. Near-optimal data-independent point locations for radial basis function interpolation, *Advances in Computational Mathematics*, 2005, vol. 23, no. 3, pp. 317–330. DOI: 10.1007/s10444-004-1829-1
9. Rendall T.C.S., Allen C.B. Efficient mesh motion using radial basis functions with data reduction algorithms, *Journal of Computational Physics*, 2009, vol. 228, no. 17, pp. 6231–6249. DOI: 10.1016/j.jcp.2009.05.013
10. Kopysov S., Kuzmin I., Nedozhogin N., Novikov A., Sagdeeva Yu. Scalable hybrid implementation of the Schur complement method for multi-GPU systems, *The Journal of Supercomputing*, 2014, vol. 69, no. 1, pp. 81–88. DOI: 10.1007/s11227-014-1209-7

Received 12.05.2018

Kopysov Sergei Petrovich, Doctor of Physics and Mathematics, Leading Researcher, Udmurt Federal Research Center of the Ural Branch of the Russian Academy of Sciences, ul. T. Baramzinoi, 34, Izhevsk, 426067, Russia;
Professor, Department of Computational Mechanics, Udmurt State University, ul. Universitetskaya, 1, Izhevsk, 426034, Russia. E-mail: s.kopysov@gmail.com

Kuz'min Igor Mikhailovich, Researcher, Udmurt Federal Research Center of the Ural Branch of the Russian Academy of Sciences, ul. T. Baramzinoi, 34, Izhevsk, 426067, Russia.
E-mail: i.m.kuzmin@gmail.com

Nedozhogin Nikita Sergeevich, Researcher, Udmurt Federal Research Center of the Ural Branch of the Russian Academy of Sciences, ul. T. Baramzinoi, 34, Izhevsk, 426067, Russia.
E-mail: nedozhogin@inbox.ru

Novikov Aleksandr Konstantinovich, Candidate of Physics and Mathematics, Senior Researcher, Udmurt Federal Research Center of the Ural Branch of the Russian Academy of Sciences, ul. T. Baramzinoi, 34, Izhevsk, 426067, Russia.
E-mail: sc_work@inbox.ru

Tonkov Leonid Evgen'evich, Candidate of Physics and Mathematics, Head of Laboratory, Udmurt Federal Research Center of the Ural Branch of the Russian Academy of Sciences, ul. T. Baramzinoi, 34, Izhevsk, 426067, Russia.
E-mail: tnk@udman.ru

***С. П. Копысов, И. М. Кузьмин, А. К. Новиков, Н. С. Недожогин, Л. Е. Тонков***

**Параллельная межсеточная интерполяция радиальными базисными функциями при решении сопряженных задач**

**Цитата:** *Известия Института математики и информатики Удмуртского государственного университета.* 2018. Т. 51. С. 42–51.

*Ключевые слова:* параллельные вычисления, гибридные платформы, сопряженные задачи, радиальные базисные функции, послойное разделение.

УДК: 519.63, 530.145.6

DOI: 10.20537/2226-3594-2018-51-02

При моделировании сопряженной задачи взаимодействия жидкости/газа и деформируемого твердого тела в разделенной постановке каждая из задач решается независимо, с использованием собственной расчетной сетки. Обычно расчетные сетки физических задач является несогласованными, поэтому возникает необходимость интерполирования физических данных (давления, перемещения) на границе сопряжения между двумя расчетными сетками. В представленной статье рассматривается сокращение затрат интерполяции на основе метода радиальных базисных функций с использованием безматричного решения системы уравнений на графических процессорах. Кроме того, представлен адаптивный алгоритм выбора точек интерполяции, позволяющий сократить размер системы уравнений с сохранением качества интерполяции. Оценка эффективности сокращения вычислительных затрат на основе безматричного подхода решения системы, а также оценка качества интерполяции осуществлялись на примере задачи моделирования истечения потока газа из сверхзвукового деформируемого сопла.

Поступила в редакцию 12.05.2018

Копысов Сергей Петрович, д. ф.-м. н., главный научный сотрудник, Удмуртский федеральный исследовательский центр УрО РАН, 426067, Россия, г. Ижевск, ул. Т. Барамзиной, 34;
профессор, кафедра вычислительной механики, Удмуртский государственный университет, 426034, Россия, г. Ижевск, ул. Университетская, 1.
E-mail: s.kopysov@gmail.com

Кузьмин Игорь Михайлович, научный сотрудник, Удмуртский федеральный исследовательский центр УрО РАН, 426067, Россия, г. Ижевск, ул. Т. Барамзиной, 34.
E-mail: i.m.kuzmin@gmail.com

Недожогин Никита Сергеевич, научный сотрудник, Удмуртский федеральный исследовательский центр УрО РАН, 426067, Россия, г. Ижевск, ул. Т. Барамзиной, 34.
E-mail: nedozhogin@inbox.ru

Новиков Александр Константинович, к. ф.-м. н., старший научный сотрудник, Удмуртский федеральный исследовательский центр УрО РАН, 426067, Россия, г. Ижевск, ул. Т. Барамзиной, 34.
E-mail: sc_work@inbox.ru

Тонков Леонид Евгеньевич, к. ф.-м. н., заведующий лабораторией, Удмуртский федеральный исследовательский центр УрО РАН, 426067, Россия, г. Ижевск, ул. Т. Барамзиной, 34.
E-mail: tnk@udman.ru