Hybrid Wavelet Collocation - Brinkman Penalization Method for Complex Geometry Flows

O. V. Vasilyev
Department of Mechanical Engineering, University of Missouri at Columbia, U.S.A.

N. K.-R. Kevlahan
Department of Mathematics and Statistics, McMaster University, Canada

Abstract

This paper introduces the hybrid wavelet collocation - Brinkman penalization method for solving the Navier–Stokes equations in arbitrarily complex geometries. The main advantages of the wavelet collocation method and penalization techniques are described, and a brief summary of their implementation is given. The hybrid method is then applied to a simple flow around cylinders with both periodic and non-periodic boundary conditions with Reynolds numbers in the range from 100 to 10 000 and the results are discussed.

1 Introduction

Many problems in fluid mechanics are characterized by the presence of a wide range of spatial and temporal scales, which are not distributed uniformly in space and time. In order to solve these problems in a computationally efficient way, the computational grid should adapt dynamically in time. For the numerical algorithm to be robust, grid adaptation, *i.e.* local grid refinement and coarsening, should be based on the local demands of the solution and not on *ad hoc* assumptions. A recently developed dynamically adaptive second generation wavelet collocation method [10, 9] is ideally suited for the solution of such problems. Wavelet collocation method takes advantage of the fact that wavelets are localized in both space and scale, and as a result functions with localized regions of sharp transition are well compressed using wavelet decomposition. The adaptation is achieved by retaining only those wavelets, whose coefficients are greater than an *a priori* given threshold. Thus, high resolution computations are carried out only in those regions, where sharp transitions occur. With this adaptation strategy, a solution is obtained on a near optimal grid, *i.e.* far fewer grid points are needed for wavelets than for conventional finite-difference, finite-element, or spectral methods.

In order to take full advantage of adaptive wavelet collocation method and solve problems of engineering relevance, the algorithm must have the capability to handle realistically complex geometries. The objective of this paper is to investigate the possibility of combining the dynamically adaptive wavelet collocation method [10, 9] and Brinkman penalization [6] as a way of simulating the presence of arbitrarily complex solid boundaries (which may be moving in time). The Brinkman penalization technique allows boundary conditions to be enforced to a specified precision, without changing the numerical method (or grid) used to solve the equations. The main advantage of this method, compared to other penalization type methods, is that the error can be estimated rigorously in terms of the penalization parameter. It can also be shown that the solution of the penalized equations converges to the exact solution in the limit as the penalization parameter tends to zero [6]. The combination of these two methods should allow us to perform numerical simulations of complex flows in geometries of engineering interest. Fluid–structure interactions (where the obstacles move in response

to the fluid forces) are a natural application of this technique, and they are currently under investigation.

2 Penalization method

2.1 General theory

Let us consider a viscous incompressible fluid governed by the Navier–Stokes equations

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} + \nabla P = \nu \Delta \boldsymbol{u}, \tag{2.1a}$$

$$\nabla \cdot \boldsymbol{u} = 0. \tag{2.1b}$$

We consider here the case where the fluid occupies the complement in the plane R^2 of a set of obstacles O_i. The problem is solved in a rectangular domain $\Omega = [L_1, L_2] \times [M_1, M_2]$ containing all the obstacles O_i. To these equations are added appropriate external (inflow, outflow and side boundary conditions), which are discussed further below.

On the surface of the obstacles the velocity must satisfy the no-slip condition,

$$\boldsymbol{u} = 0 \text{ on } \partial O_i, \ \forall i. \tag{2.2}$$

To model the effect of the no-slip boundary conditions on the obstacles O_i without explicitly imposing (2.2) we replace Eqs. (2.1) with appropriate boundary conditions by the following set of *penalized* equations

$$\frac{\partial \boldsymbol{u}_\eta}{\partial t} + \boldsymbol{u}_\eta \cdot \nabla \boldsymbol{u}_\eta + \nabla P_\eta = \nu \Delta \boldsymbol{u}_\eta - \frac{1}{\eta} \chi_0 \boldsymbol{u}_\eta, \tag{2.3a}$$

$$\nabla \cdot \boldsymbol{u}_\eta = 0, \tag{2.3b}$$

with appropriate external boundary conditions. Note that Eqs. (2.3) are valid in the *entire domain* Ω. Here $\eta > 0$ is a penalization coefficient and χ_0 denotes the characteristic (or mask) function

$$\chi_0(\boldsymbol{x}, t) = \begin{cases} 1 & \text{if } \boldsymbol{x} \in O_i, \\ 0 & \text{otherwise.} \end{cases} \tag{2.4}$$

As $\eta \to 0$, it was proved theoretically [1] that the solutions of the penalized equations (2.3) converge to that of the Navier–Stokes equations (2.1) with the correct boundary conditions. More precisely, the upper bound on the global error of the penalization was shown to be [1]

$$||\boldsymbol{u} - \boldsymbol{u}_\eta|| \leq C \eta^{1/4}. \tag{2.5}$$

In the specific case of impulsively started flow over a plane Kevlahan & Ghidaglia [5] showed that the error is actually lower: $O(\eta^{1/2})$.

This penalization has been implemented in a finite difference code [6] for flow around a cylinder and was found to give very good results. In fact, the actual error was slightly better, $O(\eta)$. It is important to note that η is an arbitrary parameter, independent of the spatial or temporal discretization, and thus the boundary conditions can be enforced to any desired accuracy by choosing η appropriately. This property distinguishes the Brinkman method from other penalization schemes and allows the error to be controlled precisely.

Another advantage of the Brinkman penalization is that the force $\boldsymbol{F}_i$ acting on an obstacle O_i can be found by simply integrating the penalization term over the volume of the obstacle: $\boldsymbol{F}_i = 1/\eta \int_{O_i} \boldsymbol{u} \, \mathrm{d}\boldsymbol{x}$. Thus, the calculation of lift and drag on an obstacle can be made simply, accurately and at very low cost.

In practice, we actually solve the penalized form of the vorticity equation,

$$\frac{\partial \omega}{\partial t} + \boldsymbol{u} \cdot \nabla \omega = \nu \Delta \omega - \frac{1}{\eta} \nabla \times (\chi_0 \boldsymbol{u}) \tag{2.6}$$

where the velocity is found from the vorticity using the Biot–Savart law. In two dimensions the Biot–Savart law has the following particularly simple form

$$U(z) = \frac{i}{2\pi} \sum_{n=1}^{N} \frac{\Gamma(z_n)}{z - z_n}, \tag{2.7}$$

where $z = x + iy$, $U = -u + iv$, $\Gamma(z_n)$ is the circulation associated with the point z_n (we estimate this as the local vorticity times the associated grid block), and N is the total number of grid points. Relation 2.7 may be solved approximately to a specified tolerance using the Fast Multipole Method (FMM) developed by Greengard and Rokhlin [4]. This method is very efficient and reduces the computational complexity of the calculating the velocity from the vorticity from $O(N^2)$ to $O(N)$. We employ two types of external boundary conditions: doubly periodic, and uniform inflow (zero vorticity) with non-reflecting side and outflow conditions.

The vorticity formulation is preferred for two reasons. First, the incompressibility equation (2.1b) is automatically satisfied, and we do not need to compute the pressure. Secondly, in the context of two-dimensional flows, the vorticity is the physically relevant variable whose size and shape are directly related to the physical phenomenon involved (instabilities, drag, lift etc.). This means that adaptive wavelet compression is usually higher in the vorticity formulation than in the velocity formulation. The wavelet method used to adapt the grid dynamically to the vorticity and to calculate derivatives is described in the following section.

3 Dynamically adaptive wavelet collocation method

The numerical method is formally derived by evaluating the governing partial differential equations at collocation points, which results in a system of nonlinear ordinary differential-algebraic equations describing the evolution of the solution at these collocation points.

In this section we briefly review the dynamically adaptive wavelet collocation (for more details please see [10, 9]). In particular, we will sketch efficient wavelet-based procedures for dynamic grid adaptation and calculation of spatial derivatives.

3.1 Grid adaptation

Grid adaptation occurs naturally in wavelet methods. To illustrate the algorithm, let us consider a function $f(\mathbf{x})$, defined on a closed two-dimensional rectangular domain Ω. We use tensor product second generation wavelets [8] constructed on a set of grids,

$$\mathcal{G}^j = \left\{ \mathbf{x}^j_{\mathbf{k}} \in \Omega : \mathbf{k} \in \mathcal{K}^j \right\}, \quad j \in \mathcal{J}, \tag{3.1}$$

where $\mathbf{k} = (k_1, \ldots, k_n)$ and grid points $\mathbf{x}^j_{\mathbf{k}} = (x^j_{1,k_1}, \ldots, x^j_{n,k_n})$ are constructed as a tensor product of uniformly or non-uniformly spaced one-dimensional grids. The only restriction is that each individual set of one-dimensional grids is nested ($x^j_{m,k_l} = x^{j+1}_{m,2k_l}$, $m = 1, 2$), which guarantees the nestedness of the grids, *i.e.* $\mathcal{G}^j \subset \mathcal{G}^{j+1}$. The procedure of constructing two-dimensional scaling functions $\phi^j_{\mathbf{k}}(\mathbf{x})$ and a family of two-dimensional wavelets $\psi^{\mu,j}_{\mathbf{l}}(\mathbf{x})$,

$\mu = 1, \ldots, 3$ on two dimensional dyadic grid is described in [8, 9]. Once wavelets and scaling functions are constructed, a function $f(\mathbf{x})$ can be decomposed as

$$f(\mathbf{x}) = \sum_{\mathbf{k}\in\mathcal{K}^0} c^0_{\mathbf{k}}\phi^0_{\mathbf{k}}(\mathbf{x}) + \sum_{j=0}^{+\infty}\sum_{\mu=1}^{3}\sum_{\mathbf{l}\in\mathcal{L}^{\mu,j}} d^{\mu,j}_{\mathbf{l}}\psi^{\mu,j}_{\mathbf{l}}(\mathbf{x}). \tag{3.2}$$

For functions which contain isolated small scales on a large-scale background, most wavelet coefficients are small, thus we retain a good approximation even after discarding a large number of wavelets with small coefficients. Intuitively, the coefficient $d^{\mu,j}_{\mathbf{l}}$ will be small unless the $f(\mathbf{x})$ has variation on the scale of j in the immediate vicinity of wavelet $\psi^{\mu,j}_{\mathbf{l}}(\mathbf{x})$. In fact, the error incurred by ignoring coefficients with magnitude lower than ϵ is $O(\epsilon)$. More precisely, if we rewrite Eq. (3.2) as a sum of two terms composed respectively of wavelets whose amplitude is above and below some prescribed threshold ϵ, *i.e.* $f(\mathbf{x}) = f_{\geq}(\mathbf{x}) + f_{<}(\mathbf{x})$, then it can be shown [2, 10, 9] that

$$|f(\mathbf{x}) - f_{\geq}(\mathbf{x})| \leq C_1\epsilon \leq C_2 N^{-p/n}, \tag{3.3}$$

where N is the number of significant wavelet coefficients, p is the order of the wavelets, and n is the dimensionality of the problem.

In order to realize the benefits of the wavelet compression, we need to be able to reconstruct the $f_{\geq}(\mathbf{x})$ from the subset of N grid points. Note that every wavelet $\psi^{\mu,j}_{\mathbf{l}}(\mathbf{x})$ is uniquely associated with the collocation point. Consequently, the collocation point should be omitted from the computational grid if the associated wavelet is omitted from the approximation. This procedure results in a set of nested adaptive computational grids $\mathcal{G}^j_{\geq} \subset \mathcal{G}^j$, such that $\mathcal{G}^j_{\geq} \subset \mathcal{G}^{j+1}_{\geq}$ for any $j < J-1$, where J is the finest level of resolution present in approximation $f_{\geq}(\mathbf{x})$.

When solving the evolution equations an additional criterion for grid adaptation should be added. The computational grid should consist of grid points associated with wavelets whose coefficients are significant or could become significant during a time step. In other words, at any instant in time, the computation grid should include points associated with wavelets belonging to an *adjacent zone* of wavelets for which the magnitude of their coefficients is greater then an *a priori* prescribed threshold. We found that the optimal adjacent zone includes the nearest points at the same, one higher, and one lower level of resolution.

3.2 Calculation of spatial derivatives on an adaptive grid

When solving partial differential equations numerically, it is necessary to obtain derivatives of a function from its values at collocation points. In this section we describe an efficient procedure for calculating spatial derivatives [10, 9], which takes advantage of the multiresolution wavelet decomposition, fast wavelet transform, and uses finite difference differentiation. In other words we make wavelets do what they do well: compress and interpolate and make finite difference do the rest: differentiate polynomials.

The differentiation procedure is based on the interpolating properties of second generation wavelets. We recall that wavelet coefficients $d^{\mu,j}_{\mathbf{l}}$ measure the difference between the approximation of the function at the $j+1$ level of resolution and its representation at the j level of resolution. Thus if there are no points in the immediate vicinity of a grid point $\mathbf{x}^j_{\mathbf{k}}$, *i.e.* $|d^{\mu,j}_{\mathbf{m}}| < \epsilon$ for all the neighboring points, and points $\mathbf{x}^{j+1}_{(2k_1\pm1,2k_2\pm1)}$ are not present in $\mathcal{G}^{j+1}_{\geq}$, then there exist some neighborhood of $\mathbf{x}^j_{\mathbf{k}}$, where the actual function is well approximated by a wavelet interpolant based on $c^j_{\mathbf{m}}$.

Thus differentiating this local polynomial gives us the value of the derivative of the function at that particular location. Let us denote by $\mathcal{D}^j_{\geq}$ a collection of such points at each level of resolution. Then the procedure for finding derivatives at all grid points consists of the following steps: First, knowing the values of a function on an adaptive computational grid $\mathcal{G}_{\geq}$, perform wavelet transform. Next, recursively reconstruct the function starting from the coarsest level of resolution. On each level of resolution j find derivatives of the function at grid points that belong to $\mathcal{D}^j_{\geq}$. At the end of the inverse wavelet transform we have derivatives of the function at all grid points. The computational cost of calculating spatial derivatives is roughly the same as the cost of forward and inverse wavelet transforms.

3.3 Numerical algorithm

The three basic steps of the numerical algorithm are then as follows (bold symbols denote n-dimensional vectors $\mathbf{u} \equiv (u_1, \ldots, u_n)$ and $\mathbf{k} \equiv (k_1, \ldots, k_n)$):

1. Knowing the values of the solution $\mathbf{u_k}(t)$ on $\mathcal{G}^t_{\geq}$, we perform the wavelet transform for each component of the solution. For a given threshold ϵ we adjust $\mathcal{G}^{t+\Delta t}_{\geq}$ based on the magnitude of the wavelet coefficients.
2. If $\mathcal{G}^t_{\geq}$ and $\mathcal{G}^{t+\Delta t}_{\geq}$ are identical, we go directly to step 3. Otherwise, we interpolate the solution to the collocation points $\mathcal{G}^{t+\Delta t}_{\geq}$, which are not included in $\mathcal{G}^t_{\geq}$.
3. We integrate the resulting system of ordinary differential equations to obtain new values $\mathbf{u_k}(t + \Delta t)$ at positions on the grid $\mathcal{G}^{t+\Delta t}_{\geq}$ and go back to step 1.

With such an algorithm the grid of collocation points adapts dynamically in time to follow local structures that appear in the solution. Note that by omitting wavelets with coefficients below a threshold parameter ϵ we automatically control the error of approximation. Thus the wavelet collocation method has another important feature: active control of the accuracy of the solution. The smaller ϵ is chosen to be, the smaller the error of the solution is. In typical applications the value of ϵ varies between 10^{-2} and 10^{-6}, assuming that the unknown dependent variables have been properly normalized. As the value of ϵ increases, fewer grid points are used in the solution.

3.4 Time integration algorithm

In this paper we use the stiffly stable Krylov subspace time-integration algorithm [3]. The main motivations to use the Krylov time-integration algorithm are the following: first, the Krylov time integration method is stiffly stable, uses an adaptive time step, and allows the order of accuracy of the time integration to be adjusted easily. Secondly, the Krylov time integration algorithm does not require explicit construction of the discretized linear operator, but rather evaluation of its action. The latter property is particularly important, since the wavelet collocation algorithm calculates derivatives directly without construction of discrete spatial derivative operators. At high Reynolds numbers the Laplacian term of the vorticity equation is another source of stiffness, and thus a stiffly-stable method is also necessary for calculating high Reynolds number flow.

4 Results

We have applied the adaptive wavelet based method described in the previous sections to calculate flow around a stationary cylinder with periodic and non-periodic external boundary

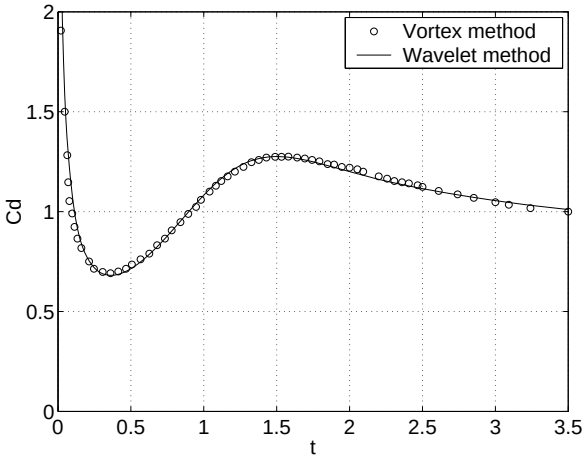

Figure 1: Drag around an impulsively started cylinder at $Re = 550$. Comparison with the vortex method [7].

conditions with Reynolds numbers from 100 to 10 000. In this section we present results from a simulation at $Re = 550$. This example is especially illustrative since we can make direct comparisons with results from the vortex method of Koumoutsakos and Leonard [7].

The cylinder has a diameter of one and is placed at the origin of a domain with dimensions $[-2.5, 7.5] \times [-5, 5]$. The maximum resolution is 1024^2, $\epsilon = 10^{-6}$, $\eta = 10^{-3}$. Numerical differentiation is eighth-order accurate (*i.e.* we use wavelets with eight zero moments). The tolerance of the Krylov time integration scheme is set to 10^{-4}, and the dimension of the Krylov subspace is 20. The flow is impulsively started at $t = 0$.

Figure 1 shows a comparison of the drag results obtained here with those obtained by Koumoutsakos and Leonard [7]. Note the good qualitative and quantitative agreement, despite the difference in the methods.

Figure 2 shows the vorticity field and dynamically adapted grid at $t = 40$, when the vortex shedding is well-established. It is interesting to see how the adaptive wavelet method has placed grid points only where they are needed to resolve the developing vorticity and the boundary layer near the obstacle.

5 Future work

The combination of the penalization method (for introducing obstacles with complicated shapes) with the wavelet method (for solve the vorticity equation on a dynamically adapting grid) appears to be working reasonably well for a stationary obstacle and two-dimensional flow. We have also confirmed numerically that the overall complexity of the method is $O(N)$ where N is the number of grid points (or wavelets) actually used.

Two extensions of this work are currently under investigation. First, the application of the method to moving obstacles. The obstacle may move due to an external force, or may move in response to the fluid forces it experiences. Applying our method to moving obstacles will allow us to investigate a variety of problems in fluid–structure interaction. The second research direction is to extend the method to three dimensional flow. This is primarily a technical problem (albeit a large one!), since all the basic components of our approach can be extended in a straightforward way to three dimensions.

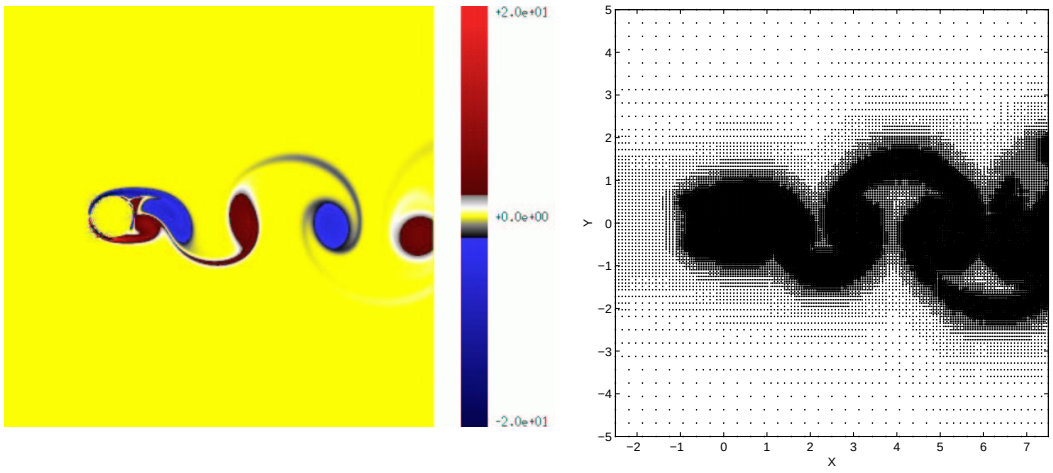

Figure 2: Vorticity field and adapted grid at $t = 40$.

References

[1] P. Angot, C.-H. Bruneau, and P. Fabrie. A penalization method to take into account obstacles in viscous flows. *Numerische Mathematik*, 81:497–520, 1999.

[2] D. L. Donoho. Interpolating wavelet transforms. Technical Report 408, Department of Statistics, Stanford University, 1992.

[3] W. S. Edwards, L. S. Tuckerman, R. A. Friesner, and D. C. Sorensen. Krylov methods for the incompressible Navier–Stokes equations. *J. Comp. Phys.*, 110:82–102, 1994.

[4] L. Greengard and V. Rokhlin. A fast algorithm for particle simulations. *J. Comput. Phys.*, 73:325–348, 1987.

[5] N. Kevlahan and J.-M. Ghidaglia. Computation of turbulent flow past an array of cylinders using a spectral method with Brinkman penalization. *Eur. J. Mech./B*, 2001. To appear.

[6] K. Khadra, P. Angot, S. Parneix, and J. P. Caltagirone. Fictitious domain approach for numerical modelling of Navier-Stokes equations. *Int. J. Num. Meth. Fluids*, 34:651–684, 2000.

[7] P. Koumoutsakos and A. Leonard. High-resolution simulations of the flow around an impulsively started cylinder using vortex methods. *J. Fluid Mech.*, 296:1–38, 1995.

[8] W. Sweldens. The lifting scheme: A construction of second generation wavelets. *SIAM J. Math. Anal.*, 29(2):511–546, 1998.

[9] O. V. Vasilyev. Solving multi-dimensional evolution problems with localized structures using second generation wavelets. To apper in *Int. J. Comp. Fluid Dyn.,* Special issue on High-resolution methods in Computational Fluid Dynamics, 2001.

[10] O. V. Vasilyev and C. Bowman. Second generation wavelet collocation method for the solution of partial differential equations. *J. Comp. Phys.*, 165:660–693, 2000.