Approximate Deconvolution Boundary Conditions for Large Eddy Simulation

J. Borggaard [*] and T. Iliescu [†]

Abstract

We introduce new boundary conditions for Large Eddy Simulation. These boundary conditions are based on an approximate deconvolution approach. They are computationally efficient and general, which makes them appropriate for the numerical simulation of turbulent flows with time-dependent boundary conditions. Numerical results are presented to demonstrate the new boundary conditions in a simplified linear setting.

Key Words and Phrases: Large eddy simulation, turbulence, boundary conditions, approximate deconvolution

1 Introduction

Large Eddy Simulation (LES) is one of the most successful techniques for the numerical simulation of turbulent flow [1, 2]. The goal of LES is to decompose the flow into large and small scales by convolving the flow with a spatial filter [1, 2]. Equations for the large scales (defined by a filter width parameter) are suitable for approximation using discretizations that are computationally tractable. A number of issues arise in LES, including model closure issues similar to those appearing in Reynolds averaging the Navier-Stokes equations [1]. However, one of the main challenges for LES is specification of efficient, general boundary conditions for the filtered variables [3].

There are essentially two ways to treat boundary conditions in LES [1]. The first is to decrease the filter width to zero at the boundaries. This popular approach, known as Near Wall Resolution (NWR) [1], captures the important flow features near the boundary, but has high computational cost since it requires a fine mesh near the wall. The second is referred to as Near Wall Modeling (NWM) [1]. The NWM boundary conditions are developed with the aid of physical modeling such as ensuring conditions on the shear stress or reproducing the logarithmic law of the wall in the mean. Although more ad-hoc (and problem specific), the discretization near boundaries can remain coarse. Hence, the NWM approach is a better candidate for LES of realistic turbulent flows.

In this paper, we propose new boundary conditions for LES based on *approximate deconvolution*. An advantage of these Approximate Deconvolution Boundary Conditions (ADBC) is that they are suited for turbulent flows with time-dependent boundary conditions. There are numerous

[*]Department of Mathematics, Virginia Polytechnic Institute and State University, 528 McBryde Hall, Blacksburg, VA 24061, U.S.A., borggaard@vt.edu; partially supported by AFOSR under grants F49620-00-1-0299 and F49620-03-1-*X*X and NSF grant DMS-0322852.

[†]Department of Mathematics, Virginia Polytechnic Institute and State University, 456 McBryde Hall, Blacksburg, VA 24061, U.S.A., iliescu@math.vt.edu; partially supported by NSF grants DMS-0209309 and DMS-0322852.

applications where the boundary conditions need to be time accurate. One such example is flow control, where for example, blowing and suction on the surface of an airfoil can be used to reduce the skin-friction drag. Note that current LES boundary conditions are not up to this task: NWR would lead to a prohibitively high computational cost and the needed boundary layer theory for NWM is not available. Our new boundary conditions avoid these two roadblocks–they are efficient and general.

2 Approximate Deconvolution Boundary Conditions

In order to develop *efficient* LES algorithms, we consider a constant, and thus "large," filter radius δ near boundaries (see Figure 1). This approach avoids the high computational cost of NWR, where δ (and thus the mesh spacing h)$\rightarrow 0$ near the wall. Using superscripts n and $n+1$ to denote variables at current and next time-steps, respectively, the challenge is to prescribe the boundary condition for the filtered variable at the next time level, $\overline{\mathbf{u}}^{n+1}(\mathbf{x}_0)$. Our approach computes this from known quantities: $\overline{\mathbf{u}}^n$ in the domain and $\mathbf{u}^{n+1}$ on the boundary.

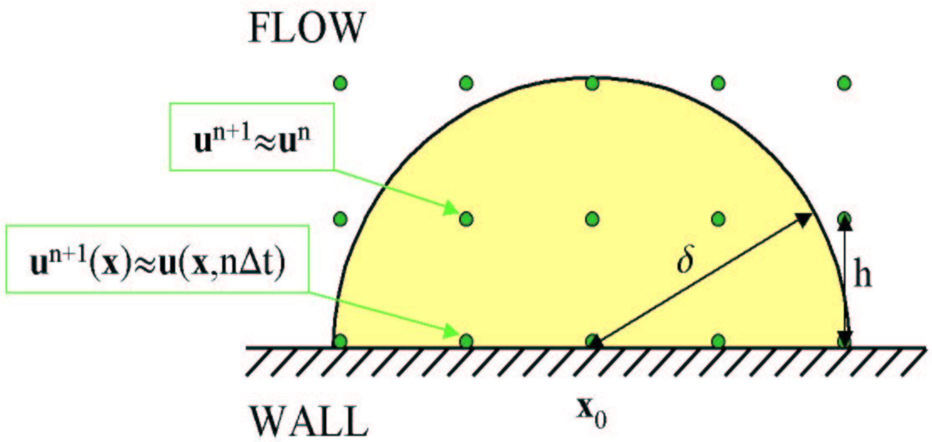
FLOW
$\mathbf{u}^{n+1} \approx \mathbf{u}^n$
$\mathbf{u}^{n+1}(\mathbf{x}) \approx \mathbf{u}(\mathbf{x}, n\Delta t)$
δ
h
WALL
$\mathbf{x}_0$

Figure 1: Setting for the ADBC Algorithm.

Our new boundary conditions are inspired by deconvolution approaches used in model closure [4, 5]. The derivation begins with the formula

$$\overline{\mathbf{u}}^{n+1}(\mathbf{x}_0) = (g_\delta * \mathbf{u}^{n+1})(\mathbf{x}_0) = \int_\Omega g_\delta(\mathbf{x}_0 - \mathbf{y}) u^{n+1}(\mathbf{y})\, d\mathbf{y}.$$

The convolution integral is computed using given Dirichlet values $\mathbf{u}^{n+1}$ on the boundary while approximation to $\mathbf{u}^{n+1}$ in the interior is performed using approximate deconvolution of the filtered variables. This may be implemented explicitly (using $\overline{\mathbf{u}}^n$) or implicitly (using $\overline{\mathbf{u}}^{n+1}$). The approximate deconvolution uses the filtered velocity and pressure ($\overline{\mathbf{u}}^k$ and $\overline{p}^k$, for $k = n$ or $n+1$) to construct an approximation to the original unfiltered variables ($\mathbf{u}^{n+1}$ and p^{n+1}) in the interior, e.g.

$$\mathbf{u}^{n+1} \approx \overline{\mathbf{u}}^k - \frac{\delta^2}{24} \Delta \overline{\mathbf{u}}^k. \tag{1}$$

Since the ADBC filters through the boundaries, we need to account for the *boundary commutation error* (BCE) term introduced in [6, 7] and analyzed mathematically in [8, 9],

$$\int_{\partial\Omega} g_\delta(\mathbf{x}_0 - \mathbf{s})\ [Re^{-1}\nabla\mathbf{u}(\mathbf{s})\ \mathbf{n}(\mathbf{s}) - p(\mathbf{s})\ \mathbf{n}(\mathbf{s})]\ d\mathbf{s}, \tag{2}$$

where $\mathbf{n}(\mathbf{s})$ is the outward unit normal vector to $\partial\Omega$ at the point $\mathbf{s} \in \partial\Omega$. Specifically, we need to find approximations for $\nabla\mathbf{u}^{n+1}(\mathbf{s})$ and $p^{n+1}(\mathbf{s})$. Again, we choose to use an approximate deconvolution approach, as in (1).

Quadrature is used in the ADBC method to approximate the convolution integral

$$\overline{\mathbf{u}}^{n+1} \approx w_0\mathbf{u}^{n+1}(\mathbf{x}_0) + \sum_{i\in\mathcal{I}} w_i \left(\overline{\mathbf{u}}^k(\mathbf{x}_i) - \frac{\delta^2}{24}\Delta\overline{\mathbf{u}}^k(\mathbf{x}_i)\right) + \sum_{i\in\mathcal{B}} w_i^b\mathbf{u}^{n+1}(\mathbf{x}_i^b), \tag{3}$$

where $\{\mathbf{x}_i\}$ represent quadrature points and $\{w_i\}$, the corresponding weights (including g_δ). The index sets $\mathcal{I}$ and $\mathcal{B}$ represent interior and boundary points, respectively. As we expect, $\delta \to 0$ leads to $w_i \to 0$ for $i > 0$ since they include the filter function. Thus, the original unfiltered boundary condition is recovered. We can consider the sums as correction terms. In practice, the rapid decay of the g_δ function away from x_0 allows us to set weights to zero outside of, for example, the shaded area in Figure 1. In our numerical experiments, we use an explicit implementation. Complex flows may require the implicit formulation making the implementation more challenging since they are nonlocal boundary conditions.

Explicit ADBC Algorithm

Step 1. Compute an approximate deconvolution approximation for $\mathbf{u}^n$ at the mesh-points inside the shaded area in Figure 1 not on the boundary.

Step 2. Using the approximations in Step 1 and the exact values of $\mathbf{u}^{n+1}$ on the boundary, compute an approximation for the BCE term (2).

Step 3. Compute the approximation to $\overline{\mathbf{u}}^{n+1}(\mathbf{x}_0)$ using 3 ($k = n$).

The convolution in $\overline{\mathbf{u}}^{n+1}(\mathbf{x}_0)$ in Step 3 is carried out in an efficient way: at the beginning of the simulation, we pre-compute convolutions of the form $g_\delta * \phi_j$, where ϕ_j are finite element basis functions corresponding to the mesh-points inside the shaded area in Figure 1; then, at each time-step, we estimate the convolution integral with simple algebraic operations of the form $u_j^{n+1}(g_\delta * \phi_j)$.

3 Numerical Results

We use the heat equation to illustrate our approach. The advantage of considering this linear problem is that we can use it to isolate the issue of boundary conditions from closure and commutation issues associated with nonlinear PDEs such as the Navier-Stokes equations. We stress that this is just the first step in the numerical validation of the ADBC method and that investigations in realistic turbulent flows are needed.

The heat equation in a domain $\Omega \subset \mathbb{R}^3$ has the following form:

$$\begin{cases} u_t - \Delta u = f & \text{in} \quad \Omega \times (0,T], \\ u = g & \text{on } \partial\Omega \times (0,T], \\ u(\mathbf{x},0) = u_0(\mathbf{x}) & \quad\text{in} \quad \Omega. \end{cases} \tag{1}$$

To convolve the above equation with a spatial filter, one needs to extend first u, f, g and u_0 to $\mathbb{R}^3$. By following the approach in [9], we obtain the space-averaged momentum equations

$$\overline{u}_t - \Delta \overline{u} - \int_{\partial\Omega} g_\delta(\mathbf{x}-\mathbf{s}) \; \nabla u(\mathbf{s}) \cdot \mathbf{n}(\mathbf{s}) \; d\mathbf{s} = \overline{f} \qquad \text{in} \quad \mathbb{R}^3 \times (0,T). \tag{2}$$

where $\mathbf{n}(\mathbf{s})$ is the outward unit normal vector to $\partial\Omega$ at the point $\mathbf{s} \in \partial\Omega$.

For simplicity, we consider the 1D heat equation. We chose f, g and u_0 such that

$$u(x,t) = t + \sin(2\,\pi\,x) + \sin(8\,\pi\,x) \tag{3}$$

be the exact solution for (1). This yielded $f(x,t) = 1 + 4\,\pi^2 \, \sin(2\,\pi\,x) + 64\,\pi^2 \, \sin(8\,\pi\,x)$, $g(t) = t$, and $u_0(x) = \sin(2\,\pi\,x) + \sin(8\,\pi\,x)$. The above functions need to be extended outside $(0,1)$ in order to convolve equations (1) with g_δ. We extended u by its values at the boundaries, that is, $u(x,t) = t$ for $x \notin (0,1)$ This yields the extension for f and u_0 $f(x,t) = 1$ and $u_0(x) = t$ for $x \notin (0,1)$. We consider g_δ to be the Gaussian filter $g_\delta(x) = (6/\pi\delta^2)^{1/2} \; exp(-6x^2/\delta^2)$, with $\delta = 0.2$.

A finite element approximation of (1) with piecewise linear basis functions, and an explicit Euler time discretization is used. The interval $(0,1)$ is divided into 20 equidistant subintervals ($\Delta x = 0.05$). To compute the convolutions with the Gaussian filer, we extended the computational domain $(0,1)$ to the left and to the right by $\delta = 0.2$. To eliminate effects of time integration (and our explicit implementation) we use a small time-step, $\Delta t = 0.0001$.

The space-averaged momentum equations (2) have the following one-dimensional form

$$\overline{u}_t - \Delta \overline{u} + g_\delta(x-0) \, \frac{\partial u}{\partial x}(0,t) - g_\delta(x-1) \, \frac{\partial u}{\partial x}(1,t) = \overline{f} \qquad \text{in} \quad \mathbb{R} \times (0,T). \tag{4}$$

The first challenge in a numerical implementation of (4) is that the terms $\frac{\partial u}{\partial x}(0,t)$ and $\frac{\partial u}{\partial x}(1,t)$ are not known a priori. A straightforward approximation to these terms is by using finite differences

$$\frac{\partial u}{\partial x}(0,t) \approx \frac{u(\Delta x, t) - u(0,t)}{\Delta x} \qquad \text{and} \qquad \frac{\partial u}{\partial x}(1,t) \approx \frac{u(1,t) - u(1-\Delta x, t)}{\Delta x}. \tag{5}$$

Unfortunately, this approximation is based on the values of u at the mesh points near the boundary, which are not known a priori either. The solution is to use an approximate deconvolution approach:

$$u(\Delta x, t) \;\; \approx \;\; \overline{u}(\Delta x) - \left(\frac{\delta^2}{24}\right) \frac{\overline{u}(0) - 2\,\overline{u}(\Delta x) + \overline{u}(2\Delta x)}{\Delta x^2}, \tag{6}$$

and similarly for $u(1-\Delta x, t)$.

We conducted four sets of numerical experiments for (1). In all these experiments, we integrated in time for 500 time-steps, which corresponds to a final time $T_{final} = 0.5$. The errors in approximating $\overline{u}$ in all four sets of experiments are plotted in Figure 2.

Test 1: Exact Boundary Term, Exact Boundary Conditions (Figure 2, top-left). This represents our benchmark.

Test 2: No Boundary Term, Exact Boundary Conditions (Figure 2, top-right). This experiment was conducted to illustrate the influence of the boundary term in (4). By dropping the boundary term in (4), we got an increase by *three orders of magnitude* in the error in $\overline{u}$ compared to that in Test 1. This indicates that (2) should be included in the LES model. These numerical results confirm the exquisite theoretical considerations in [8].

Test 3: Approximate Boundary Term, Exact Boundary Conditions (Figure 2, bottom-left). The boundary term is approximated through a combined finite-differences, approximate deconvolution approach (5)-(6). The results are not as good as those in Test 1, since we do not use the exact boundary term anymore. However, they are significantly better than those in Test 2 - there is a *four-fold reduction* of the error in $\overline{u}$.

Test 4: Approximate Boundary Term, Approximate Boundary Conditions (Figure 2, bottom-right). Both the boundary term and the boundary conditions are approximated. There is basically just a slight decrease of accuracy from Test 3, especially at the boundary. This decrease, however, does not seem to degrade the overall accuracy.

4 Conclusions

We have introduced the ADBC algorithm, a new set of boundary conditions for LES. The ADBC algorithm is based on an approximate deconvolution approach. The new boundary conditions are computationally efficient and general (they function in cases where the boundary layer theory is not available). These two features make the ADBC algorithm well suited for turbulent flows with time-dependent boundary conditions, such as those in a closed-loop flow control setting.

We tested ADBC in a finite element discretization of the one-dimensional heat equation. We chose a linear problem in order to decouple the boundary treatment from the closure problem (due to the nonlinearity in the Navier-Stokes equations). The numerical results proved that the boundary commutation error term (2) should be included in the LES model. They also suggested that our approximations to the boundary commutation error term (2) and boundary conditions are appropriate.

The error in $\overline{u}$ in Tests 1 and 2 in Section 3 illustrate the importance of including the boundary commutation error (2). Test 3 shows the appropriate treatment of (2) in ADBC. Finally, Test 4 shows the efficient treatment of boundary conditions in ADBC. These first numerical tests with ADBC were promising. We will continue our careful numerical validation of ADBC in the numerical simulation of realistic turbulent flows, such as turbulent channel flows with time-dependent boundary conditions.

References

[1] S.B. Pope. *Turbulent flows.* Cambridge University Press, 2000.

[2] P. Sagaut. *Large eddy simulation for incompressible flows.* Springer-Verlag, Berlin, 2002.

[3] U. Piomelli and E. Balaras. Wall-layer models for large-eddy simulations. *Annu. Rev. Fluid Mech.*, 34:349–374, 2002.

[4] L.C. Berselli and T. Iliescu. A higher-order subfilter-scale model for large eddy simulation. *J. Comput. Appl. Math.*, 159(2):411–430, 2003.

[5] G.P. Galdi and W.J. Layton. Approximation of the larger eddies in fluid motions. II. A model for space-filtered flow. *Math. Models Methods Appl. Sci.*, 10(3):343–350, 2000.

[6] S. Ghosal and P. Moin. The basic equations for the large eddy simulation of turbulent flows in complex geometry. *J. Comput. Phys.*, 118(1):24–37, 1995.

[7] O.V. Vasilyev, T.S. Lund, and P. Moin. A general class of commutative filters for LES in complex geometries. *Journal of Computational Physics*, 146:82–104, 1998.

[8] L. C. Berselli and V. John. On the comparison of a commutation error and the Reynolds stress tensor for flows obeying a wall law. *Quaderno 2004/18 del Dipartimento di Matematica Applicata "U.Dini"*, 2004.

[9] A. Dunca, V. John, and W.J. Layton. The commutation error of the space averaged Navier-Stokes equations on a bounded domain. *J. Math. Fluid Mech.*, 2004. to appear.

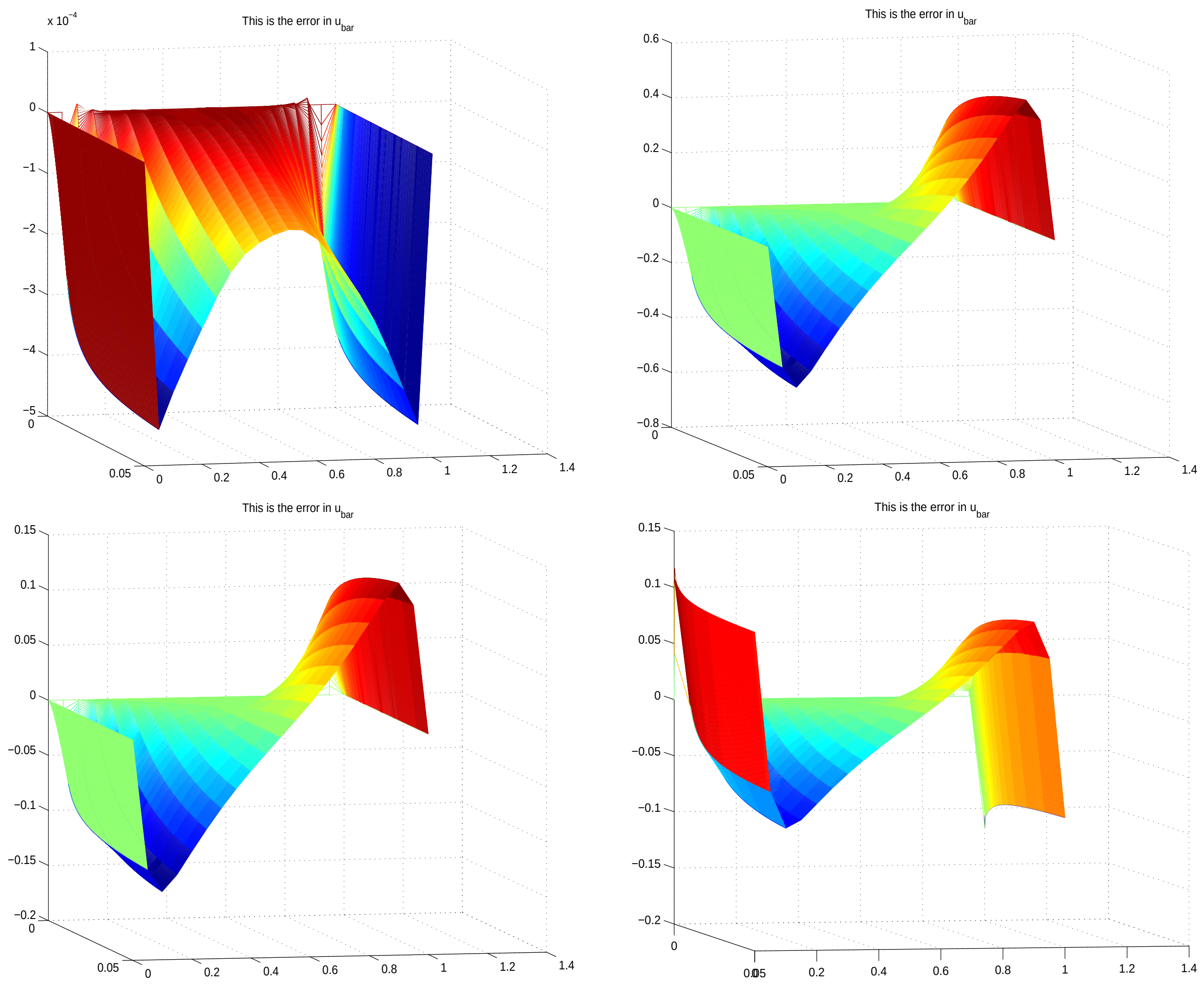

Figure 2: Error in $\overline{\mathbf{u}}$ (top to bottom and left to right): exact Commutation Error, exact Boundary Conditions; no Commutation Error, exact Boundary Conditions; approximate Commutation Error, exact Boundary Conditions; approximate Commutation Error, approximate Boundary Conditions;