Available online at www.sciencedirect.com

SCIENCE DIRECT®

Comput. Methods Appl. Mech. Engrg. 195 (2006) 4594–4603

Computer methods in applied mechanics and engineering

www.elsevier.com/locate/cma

A two-level variational multiscale method for convection-dominated convection–diffusion equations

Volker John [a,*], Songul Kaya [b], William Layton [c,1]

[a] FR 6.1—Mathematik, Universität des Saarlandes, Postfach 15 11 50, 66041 Saarbrücken, Germany
[b] Department of Mathematics, Middle East Technical University, 06531 Ankara, Turkey
[c] Department of Mathematics, University of Pittsburgh, Pittsburgh, PA 15260, USA

Received 23 February 2005; received in revised form 26 September 2005; accepted 7 October 2005

Abstract

This paper studies the error in, the efficient implementation of and time stepping methods for a variational multiscale method (VMS) for solving convection-dominated problems. The VMS studied uses a fine mesh C^0 finite element space X^h to approximate the concentration and a coarse mesh discontinuous vector finite element space L^H for the large scales of the flux in the two scale discretization. Our tests show that these choices lead to an efficient VMS whose complexity is further reduced if a (locally) L^2-orthogonal basis for L^H is used. A fully implicit and a semi-implicit treatment of the terms which link effects across scales are tested and compared. The semi-implicit VMS was much more efficient. The observed global accuracy of the most straightforward VMS implementation was much better than the artificial diffusion stabilization and comparable to a streamline-diffusion finite element method in our tests.

Keywords: Convection-dominated convection–diffusion equation; Variational multiscale method; Two-level method; Efficient implementation

1. The variational multiscale method

We consider a time-dependent scalar convection–diffusion equation

$$\begin{aligned} &u_t - \varepsilon\Delta u + \mathbf{b}\cdot\nabla u + cu = f \quad \text{in } (0,T]\times\Omega,\\ &u = 0 \quad \text{on } [0,T]\times\partial\Omega,\\ &u(0,\mathbf{x}) = u_0(\mathbf{x}) \quad \text{in } \Omega. \end{aligned} \tag{1}$$

Here, $\Omega\subset\mathbb{R}^d$, $d\in\{2,3\}$, is a bounded polyhedral domain. The functions $\mathbf{b}\in(L^\infty(0,T;L^\infty(\Omega)))^d$, $c\in L^\infty(0,T;L^\infty(\Omega))$ with $c(\mathbf{x},t)\geqslant 0, f\in L^2(0,T;L^2(\Omega))$, $u_0(\mathbf{x})\in H_0^1(\Omega)$ and the constant $\varepsilon>0$ are given. The use of homogeneous Dirichlet boundary conditions is only for convenience of presentation. Non-homogeneous Dirichlet boundary conditions are considered in the numerical studies. In these studies, only the case $c(\mathbf{x},t)=0$ will be considered. Let $X=H_0^1(\Omega)$ and let $(\cdot,\cdot)$ denote the $L^2(\Omega)$-inner product. The variational solution of (1) is a strongly differentiable map: $u:[0,T]\to X$ satisfying $u(0,\mathbf{x})=u_0(\mathbf{x})\in X$ and

$$(u_t,v)+a(u,v)=(f,v)\quad \forall v\in X, \tag{2}$$

[*] Corresponding author. Tel.: +49 681 302 4445; fax: +49 681 302 4443.
E-mail addresses: john@math.uni-sb.de (V. John), songul@math.metu.edu.tr (S. Kaya), wjl@pitt.edu (W. Layton).
URLs: http://www.math.uni-sb.de/ag/john/ (V. John), http://www.math.pitt.edu/~wjl (W. Layton).

[1] Partially supported by NSF Grant DMS0207 627.

0045-7825/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.cma.2005.10.006

where

$$a(u,v)=(\varepsilon\nabla u,\nabla v)+(\mathbf{b}\cdot\nabla u+cu,v).$$

We consider the case that ε is small compared to $\|\mathbf{b}\|_{(L^\infty(0,T;L^\infty(\Omega)))^d}$. The convection–diffusion equation (1) above occurs in many practical problems in which the diffusion coefficient is very small compared to the velocity field $\mathbf{b}$ which drives the convection, precisely the case which is most difficult to solve accurately. The solution then contains many scales composed of a complex collection of boundary and interior layers. Usual (centered) finite element methods typically produce approximate solutions with large, non-physical oscillations unless either the mesh width h is globally small with respect to the diffusion coefficient ε or enough is known about the exact solution to generate Shishkin-like meshes which are locally small with respect to ε in all transition regions in a very precise sense [30]. Thus, various stabilizations have proven to be essential computational tools. Recently, Hughes et al. [13,11] have developed the variational multiscale method (VMS) which is motivated by the inherent multiscale structure of the solution of (1). We consider in this paper a method which arose from related considerations [26] and is, in fact, a VMS. The method (3) below introduces global stabilization and then anti-diffuses these effects on the large scales of the solution. Thus, effective stabilization is retained only on the smallest resolved scales (in which the non-physical oscillations occur).

Let $\mathscr{T}^H(\Omega)$ be a conforming triangulation of Ω and let $\mathscr{T}^h(\Omega)$ be a refinement of $\mathscr{T}^H(\Omega)$ or $\mathscr{T}^h(\Omega)=\mathscr{T}^H(\Omega)$. The finite element approximation of the solution of (2) is sought in the conforming finite element space $X^h\subset X$. Let L^H denote a vector-valued finite element subspace of $(L^2(\Omega))^d$. The discretization we study adds additional diffusion acting on all discrete scales and then anti-diffuses on the scales resolvable on $\mathscr{T}^H(\Omega)$ as follows: find $u^h:[0,T]\to X^h$, $\mathbf{g}^H:[0,T]\to L^H$ satisfying

$$\begin{aligned}&(u_t^h,v^h)+a(u^h,v^h)+(\varepsilon_{\text{add}}\nabla u^h,\nabla v^h)-(\varepsilon_{\text{add}}\mathbf{g}^H,\nabla v^h)=(f,v^h)\quad\forall v^h\in X^h,\\&(\mathbf{g}^H-\nabla u^h,\mathbf{l}^H)=0\quad\forall\mathbf{l}^H\in L^H.\end{aligned}\tag{3}$$

Here, ε_{add} is a non-negative function depending on the mesh size h. This method is similar to the orthogonal subscale stabilization method proposed in [4]. The main difference consists in the functions which are projected. In contrast to (3), the method from [4] projects $\mathbf{b}\cdot\nabla u^h$.

The second equation of (3) states that $\mathbf{g}^H=\overline{P}_H\nabla u^h$ where $\overline{P}_H$ is the L^2-orthogonal projection into L^H. Consider the case that ε_{add} is a non-negative constant. Then, the second equation of (3) gives

$$(\varepsilon_{\text{add}}(I-\overline{P}_H)\nabla u^h,\overline{P}_H\nabla v^h)=\varepsilon_{\text{add}}((I-\overline{P}_H)\nabla u^h,\overline{P}_H\nabla v^h)=0$$

since $\overline{P}_H\nabla v^h\in L^H$. Now (3) can be reformulated as follows: find $u^h:[0,T]\to X^h$ satisfying

$$(u_t^h,v^h)+a(u^h,v^h)+(\varepsilon_{\text{add}}(I-\overline{P}_H)\nabla u^h,(I-\overline{P}_H)\nabla v^h)=(f,v^h)\tag{4}$$

for all $v^h\in X^h$. Since L^H represents the large scales of the gradients, $(I-\overline{P}_H)\nabla u^h$ clearly represents the small fluctuations of ∇u^h. Thus, the method (4) introduces additional diffusion acting directly only on the fluctuating components of ∇u^h. In the case that ε_{add} is a constant, a straightforward computation, using the orthogonality of the second equation of (3), gives

$$(\varepsilon_{\text{add}}(I-\overline{P}_H)\nabla u^h,(I-\overline{P}_H)\nabla v^h)=(\varepsilon_{\text{add}}\nabla u^h,\nabla v^h)-(\varepsilon_{\text{add}}\overline{\nabla u^h},\overline{\nabla v^h}),$$

where $\overline{\nabla u^h}=\overline{P}_H\nabla u^h$. Then (4) can be rewritten in the form

$$(u_t^h,v^h)+a(u^h,v^h)+(\varepsilon_{\text{add}}\nabla u^h,\nabla v^h)-(\varepsilon_{\text{add}}\overline{\nabla u^h},\overline{\nabla v^h})=(f,v^h)\quad\forall v^h\in X^h.\tag{5}$$

This paper studies algorithmic aspects of the two formulations (3) and (5). In both, the large scale space L^H must be chosen. If X^h is a higher order finite element space on a given mesh, one approach is to define the large scale space using lower order finite elements on the same mesh. The implementation of this choice was discussed in [18]. For low order elements, which are the only elements available in many codes, the only option is to define the large scale space L^H on a coarse mesh leading to a two-level discretization, considered herein. Low order elements are also the most common choice for diffusion-transport problems in geophysics because of the very large scales of the problems studied. The goal of this paper is to study efficient implementations of the two-level VMS idea and to delineate pros and cons of different time stepping methods for multiscale discretizations.

Theoretical studies of this method began in [26] and were continued in [10,18,21–25,29]. We note that it is inspired by both physical ideas in turbulence modeling and algorithmic ideas developed for simulation of non-Newtonian fluids, [5]. It can also be thought of as a finite element realization of the method of spectral viscosity, e.g., see Maday and Tadmor [28] or Chen et al. [2].

Based on ideas developed in [13,9], multiscale discretizations have recently attracted attention for the simulation of turbulent flows [11,12,15,6–8]. The VMS idea is to use a variationally consistent discretization for the large scales and to stabilize directly only small scales. Since small and large scales are coupled, the stabilization acts indirectly also on the large scales. The stabilization accounts for the effects of the unresolved solution scales upon the smallest resolved scales in the approximate solution and its choice must be based on physical ideas in turbulence modeling. Within the framework of finite element methods, this approach was tested, e.g., in [7,8]. In these studies, standard finite element spaces were used for the large scale velocity/pressure and bubble functions to model the small scales. The implementation of this approach is straightforward because the bubble functions vanish on every face of the mesh cells, so their contribution to the global stiffness matrix can be eliminated by static condensation. On the other hand, this choice imposes a constraint for computational convenience rather than physical fidelity on the small scales that they must vanish on all mesh cell boundaries. Thus, we consider herein a more complex model for fluctuations. With extra complexity, the question of its computational difficulty and implementation becomes more important.

2. Algorithmic aspects

We will start by considering the formulation (3) of the VMS. As discretization in time, an implicit θ-scheme is applied. This leads in the discrete time t_k to the following fully discrete equations:

$$\begin{aligned}
&(u_k^h, v^h) + \theta_1 \Delta t_k \left[((\varepsilon + \varepsilon_{\text{add}}) \nabla u_k^h, \nabla v^h) + (\mathbf{b} \cdot \nabla u_k^h + c u_k^h, v^h) - (\varepsilon_{\text{add}} \mathbf{g}_k^H, \nabla v^h) \right] \\
&\quad = (u_{k-1}^h, v^h) - \theta_2 \Delta t_k \left[((\varepsilon + \varepsilon_{\text{add}}) \nabla u_{k-1}^h, \nabla v^h) + (\mathbf{b} \cdot \nabla u_{k-1}^h + c u_{k-1}^h, v^h) - (\varepsilon_{\text{add}} \mathbf{g}_{k-1}^H, \nabla v^h) \right] \\
&\qquad + \theta_3 \Delta t_k (f_{k-1}, v^h) + \theta_4 \Delta t_k (f_k, v^h) \quad \forall v^h \in V^h \\
&(\mathbf{g}_k^H - \nabla u_k^h, \mathbf{l}^H) = 0 \ \forall \mathbf{l}^H \in L^H.
\end{aligned} \tag{6}$$

Here, $\Delta t_k = t_k - t_{k-1}$. Different choices of the parameters $\theta_1, \ldots, \theta_4$ give different time stepping schemes, see Table 1. The parameters in the fractional-step θ-scheme are given by

$$\theta = 1 - \frac{\sqrt{2}}{2}, \quad \tilde{\theta} = 1 - 2\theta, \quad \tau = \frac{\tilde{\theta}}{1 - \theta}, \quad \eta = 1 - \tau.$$

We consider for convenience of presentation the two-dimensional case. The same ideas can be applied in a straightforward way to three-dimensional convection–diffusion equations. The finite element spaces are equipped with bases

$$X^h = \operatorname{span}\{\phi_i^h\}, \quad i = 1, \ldots, N_X, \quad L^H = \operatorname{span}\left\{ \begin{pmatrix} \psi_i^H \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ \psi_i^H \end{pmatrix} \right\}, \quad i = 1, \ldots, N_L, \tag{7}$$

where N_X is the dimension of X^h and N_L is the dimension of one component of L^H, i.e., $\dim(L^H) = 2N_L$. Then, the algebraic representation of (6) looks as follows:

$$\begin{aligned}
\mathscr{A} \begin{pmatrix} u_k \\ g_{k,1} \\ g_{k,2} \end{pmatrix} &= \begin{pmatrix} M + \theta_1 \Delta t_k A & \theta_1 \Delta t_k B_1 & \theta_1 \Delta t_k B_2 \\ C_1 & D & 0 \\ C_2 & 0 & D \end{pmatrix} \begin{pmatrix} u_k \\ g_{k,1} \\ g_{k,2} \end{pmatrix} \\
&= \begin{pmatrix} \theta_3 \Delta t_k f_{k-1} + \theta_4 \Delta t_k f_k \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} M - \theta_2 \Delta t_k A & -\theta_2 \Delta t_k B_1 & -\theta_2 \Delta t_k B_2 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} u_{k-1} \\ g_{k-1,1} \\ g_{k-1,2} \end{pmatrix},
\end{aligned} \tag{8}$$

where

$$\begin{aligned}
M &= (\phi_j^h, \phi_i^h)_{i,j=1,\ldots,N_X}, \\
A &= (((\varepsilon + \varepsilon_{\text{add}}) \nabla \phi_j^h, \nabla \phi_i^h) + (\mathbf{b} \cdot \nabla \phi_j^h + c\phi_j^h, \phi_i^h))_{i,j=1,\ldots,N_X}, \\
B_1 &= -(\varepsilon_{\text{add}} \psi_j^H, (\phi_i^h)_x)_{i=1,\ldots,N_X, j=1,\ldots,N_L}, \\
B_2 &= -(\varepsilon_{\text{add}} \psi_j^H, (\phi_i^h)_y)_{i=1,\ldots,N_X, j=1,\ldots,N_L}, \\
C_1 &= -(\psi_i^H, (\phi_j^h)_x)_{i=1,\ldots,N_L, j=1,\ldots,N_X}, \\
C_2 &= -(\psi_i^H, (\phi_j^h)_y)_{i=1,\ldots,N_L, j=1,\ldots,N_X}, \\
D &= (\psi_j^H, \psi_i^H)_{i,j=1,\ldots,N_L}.
\end{aligned}$$

Table 1
Implicit θ-schemes

	θ_1	θ_2	θ_3	θ_4	t_{k-1}	t_k	Δt_k
Backward Euler	1	0	0	1	t_{n-1}	t_n	Δt_n
Crank–Nicolson	0.5	0.5	0.5	0.5	t_{n-1}	t_n	Δt_n
Fractional-step							
Step 1	$\tau\theta$	$\eta\theta$	$\eta\theta$	$\tau\theta$	t_{n-1}	$t_{n-1} + \theta\Delta t_n$	$\theta\Delta t_n$
Step 2	$\eta\tilde{\theta}$	$\tau\tilde{\theta}$	$\tau\tilde{\theta}$	$\eta\tilde{\theta}$	$t_{n-1} + \theta\Delta t_n$	$t_n - \theta\Delta t_n$	$\tilde{\theta}\Delta t_n$
Step 3	$\tau\theta$	$\eta\theta$	$\eta\theta$	$\tau\theta$	$t_n - \theta\Delta t_n$	t_n	$\theta\Delta t_n$

Note, the blocks B_1, B_2 have to be scaled in the same way in (8) as the block A since the additional diffusion ε_{add} has to be the same in all of these blocks.

The matrix blocks M, A and D are sparse since they are build from inner products with finite element functions from only one space. Thus, the sparsity of these blocks is a standard property. However, the sparsity of the matrix blocks B_1, B_2, C_1, C_2 depends heavily on the choice of L^H. An inner product defining an entry of these matrices does not vanish if the intersection of the support of the two factors has a positive measure

$$\operatorname{meas}(\operatorname{supp}(\psi_i^H) \cap \operatorname{supp}(\phi_j^h)) > 0, \quad \psi_i^H \in L^H, \ \phi_j^h \in X^h.$$

The number of non-zero entries connected to the basis function ψ_i^H becomes the smaller, the smaller the support of ψ_i^H is. The smallest possible support is one mesh cell on $\mathcal{T}^H(\Omega)$. This can be realized if L^H is a discontinuous finite element space.

The fully implicit VMS introduces $2N_L$ additional equations for the unknowns $g_{k,1}$, $g_{k,2}$. A way to avoid this problem is to use a semi-implicit version of (6) as

$$\begin{aligned}
&(u_k^h, v^h) + \theta_1 \Delta t_k \left[((\varepsilon + \varepsilon_{\text{add}})\nabla u_k^h, \nabla v^h) + (\mathbf{b}\cdot\nabla u_k^h + cu_k^h, v^h) \right] \\
&\quad = (u_{k-1}^h, v^h) - \theta_2 \Delta t_k \left[((\varepsilon + \varepsilon_{\text{add}})\nabla u_{k-1}^h, \nabla v^h) + (\mathbf{b}\cdot\nabla u_{k-1}^h + cu_{k-1}^h, v^h) \right] + \Delta t_k (\varepsilon_{\text{add}} \mathbf{g}_{k-1}^H, \nabla v^h) \\
&\qquad + \theta_3 \Delta t_k (f_{k-1}, v^h) + \theta_4 \Delta t_k (f_k, v^h) \quad \forall v^h \in V^h \\
&(\mathbf{g}_{k-1}^H, \mathbf{l}^H) = (\nabla u_{k-1}^h, \mathbf{l}^H) \ \ \forall \mathbf{l}^H \in L^H.
\end{aligned} \tag{9}$$

In this semi-implicit version, the subtraction of the additional diffusion in the large scales is treated explicitly.

Note, scheme (9) might provide still a stabilization even in the case that the projection g_{k-1}^H is equal to ∇u_{k-1}^h. Then, on the right-hand side there is the term $(1-\theta_2)\Delta t_k(\varepsilon_{\text{add}}\nabla u_{k-1}^h, \nabla v^h)$ and on the left-hand side, we have $\theta_1 \Delta t_k(\varepsilon_{\text{add}}\nabla u_k^h, \nabla v^h)$. These two terms are in general not the same and they are additional terms in comparison to the Galerkin finite element discretization. For example, in the Crank–Nicolson scheme, which will be used in the numerical studies, collecting terms (on the left-hand side) gives $0.5\Delta t_k(\varepsilon_{\text{add}}\nabla(u_k^h - u_{k-1}^h), \nabla v^h)$. If the time step is sufficiently small, there are only small changes in the solution from time $k-1$ to k and $\nabla(u_k^h - u_{k-1}^h)$ represents small scales. In this way, one obtains a second small scale stabilization by the explicit treatment of the projection term.

The algebraic form of the second equation of the coupled system (9) is

$$\begin{pmatrix} D & 0 \\ 0 & D \end{pmatrix} \begin{pmatrix} g_{k-1,1} \\ g_{k-1,2} \end{pmatrix} = -\begin{pmatrix} C_1 u_{k-1} \\ C_2 u_{k-1} \end{pmatrix}. \tag{10}$$

If the mass matrix D of L^H is diagonal, that means iff the basis functions of L^H are L^2-orthogonal, the solution of this system is very simple. This property can be achieved easily for discontinuous finite element spaces, e.g., by using a basis consisting of Legendre polynomials. Inserting the solution of (10) into the first equation of (8) gives

$$(M + \theta_1 \Delta t_k A) u_k = \theta_3 \Delta t_k f_{k-1} + \theta_4 \Delta t_k f_k + (M - \theta_2 \Delta t_k A) u_{k-1} + \Delta t_k B_1 D^{-1} C_1 u_{k-1} + \Delta t_k B_2 D^{-1} C_2 u_{k-1}. \tag{11}$$

Note, the matrix A includes an additional diffusion in the diffusive term. Thus, the operator on the left-hand side of (11) is stable if the amount of additional diffusion is sufficiently large. In addition, many standard solvers and preconditioners work well for such problems. The only difference to the simple artificial diffusion stabilization of a convection–diffusion equation consists in the last two terms on the right-hand side of (11).

The summary of the algorithmic aspects of using the VMS (3) is as follows:

- the additional matrix blocks B_1, B_2, C_1, C_2 and D are needed, at least one dimension of these blocks is N_L,
- the sparsest structure of B_1, B_2, C_1, C_2 is achieved if L^H is a discontinuous finite element space,
- the fully implicit approach (8) requires the additional vectors $g_{k,1}$, $g_{k,2}$,
- the algebraic system (8) possesses $2N_L$ additional equations,
- the semi-implicit approach (11) can be implemented easily if the basis functions of L^H are L^2-orthogonal,
- the system matrix of (11) corresponds to a stable operator.

Now, we will consider the formulation (5) of the VMS. The application of an implicit θ-scheme leads in each discrete time to a scalar equation of the form

$$\begin{aligned}
&(u_k^h, v^h) + \theta_1 \Delta t_k \left[((\varepsilon + \varepsilon_{\text{add}})\nabla u_k^h, \nabla v^h) + (\mathbf{b}\cdot\nabla u_k^h + cu_k^h, v^h) - (\varepsilon_{\text{add}}\overline{\nabla u_k^h}, \overline{\nabla v^h}) \right] \\
&\quad = (u_{k-1}^h, v^h) - \theta_2 \Delta t_k \left[((\varepsilon + \varepsilon_{\text{add}})\nabla u_{k-1}^h, \nabla v^h) + (\mathbf{b}\cdot\nabla u_{k-1}^h + cu_{k-1}^h, v^h) - (\varepsilon_{\text{add}}\overline{\nabla u_{k-1}^h}, \overline{\nabla v^h}) \right] \\
&\qquad + \theta_3 \Delta t_k (f_{k-1}, v^h) + \theta_4 \Delta t_k (f_k, v^h) \quad \forall v^h \in V^h.
\end{aligned} \tag{12}$$

In this equation, the large scales $\mathbf{g}^H$ are eliminated. But the term $(\varepsilon_{\text{add}}\overline{\nabla u_k^h}, \overline{\nabla v^h})$ couples the variables of X^h across the mesh cells of the coarse triangulation $\mathscr{T}^H(\Omega)$. Let ϕ_i^h, ϕ_j^h be two basis functions of X^h. Since $\text{supp}(\phi_i^h) \subset \text{supp}(\nabla\phi_i^h)$, the term $(\varepsilon_{\text{add}}\overline{\nabla\phi_j^h}, \overline{\nabla\phi_i^h})$ does not vanish if

$$\text{meas}\left(\text{supp}\left(\overline{\nabla\phi_i^h}\right) \cap \text{supp}(\overline{\nabla\phi_j^h})\right) > 0. \tag{13}$$

To minimize the number of non-vanishing terms of this form, the support of the projections has to be minimized. This is achieved by using a discontinuous finite element space for L^H. The algebraic representation of (12) is

$$(M + \theta_1\Delta t_k(A+B))u_k = \theta_3\Delta t_k f_{k-1} + \theta_4\Delta t_k f_k + (M - \theta_2\Delta t_k(A+B))u_{k-1}. \tag{14}$$

This can be derived also from (8) by solving for $g_{k,1}$, $g_{k,2}$ which shows that

$$B = -B_1D^{-1}C_1 - B_2D^{-1}C_2.$$

However, the matrix B in (14) is not given in this product form but it is assembled directly by evaluating terms of the form

$$(B)_{ij} = \left(\varepsilon_{\text{add}}\overline{\nabla\phi_j^h}, \overline{\nabla\phi_i^h}\right). \tag{15}$$

This can be done in the following way:

1. Compute in a pre-processing step the matrix structure of B by checking (13).
2. If $(B)_{ij}$ is a member of this matrix structure, then compute $\overline{\nabla\phi_i^h}$, and $\overline{\nabla\phi_j^h}$. Taking the basis (7) and using the ansatz

   $$\overline{\nabla\phi_i^h} = \sum_{j=1}^{N_L}\phi_j\psi_j^H,$$

   one gets for the evaluation of $\overline{\nabla\phi_i^h}$

   $$\sum_{j=1}^{N_L}(\psi_j^H, \psi_k^H)\phi_j = (\nabla\phi_i^h, \psi_k^H), \quad k = 1, \ldots, N_L. \tag{16}$$

   The coefficients ϕ_j can be easily computed if the system matrix is diagonal, i.e., if the basis functions of L^H are L^2-orthogonal. But even if L^H is solely a discontinuous finite element space, (16) decouples in a number of small problems which can be solved in parallel.
3. Compute the inner product (15).

   The dimension of B is $N_X \times N_X$. In comparison to the matrix blocks M and A, the block B creates a substantial fill-in. The problem of having this substantial fill-in does not arise if a semi-implicit version of (12) is used

$$\begin{aligned}(u_k^h, v^h) &+ \theta_1\Delta t_k\left[((\varepsilon + \varepsilon_{\text{add}})\nabla u_k^h, \nabla v^h) + (\mathbf{b}\cdot\nabla u_k^h + cu_k^h, v^h)\right]\\ &= (u_{k-1}^h, v^h) - \theta_2\Delta t_k\left[((\varepsilon + \varepsilon_{\text{add}})\nabla u_{k-1}^h, \nabla v^h) + (\mathbf{b}\cdot\nabla u_{k-1}^h + cu_{k-1}^h, v^h)\right]\\ &\quad + \Delta t_k(\varepsilon_{\text{add}}\overline{\nabla u_{k-1}^h}, \overline{\nabla v^h}) + \theta_3\Delta t_k(f_{k-1}, v^h) + \theta_4\Delta t_k(f_k, v^h) \quad \forall v^h \in V^h.\end{aligned} \tag{17}$$

To evaluate the term $(\varepsilon_{\text{add}}\overline{\nabla u_{k-1}^h}, \overline{\nabla v^h})$ on the right-hand side, one can use the same approach as for computing the entries of B. The algebraic form of the semi-implicit system is like (11). Here, the additional terms on the right-hand side of (11) are not computed by matrix–vector products but by computing the explicit projection of the finite element function and the appropriate inner products. Additional matrices are not necessary. Which of the two approaches should be preferred depends certainly on the code where the method will be implemented. The approach of assembling the matrix blocks B_1, B_2, C_1, C_2 and the computing of the matrix–vector products as in (11) requires more memory and the evaluation of $(\varepsilon_{\text{add}}\overline{\nabla u_{k-1}^h}, \overline{\nabla v^h})$ in (17) needs the implementation of a routine like those, which was described for assembling (15). Based on the data structures in our code, we preferred to implement the first approach.

The algorithmic aspects for the VMS of form (5) are summarized as follows:

- for the fully implicit approach (14), the additional matrix B of dimension $N_X \times N_X$ is needed, this matrix is considerably less sparse than the other blocks,
- the sparsest structure of B is achieved if L^H is a discontinuous finite element space,
- the fully implicit approach (14) does not generate additional equations,
- matrix–vector products with the system matrix become considerably more expensive due to the additional fill-in of the matrix B,

- if L^H is a discontinuous finite element space, the computation of the projections decouples in a number of small systems which can be solved in parallel,
- for an easy computation of the projections, it is also advantageous if the basis functions of L^H are L^2-orthogonal.

The attractive semi-implicit strategy (9) and (17) has been studied in [1]. In [1], the method (17) was proven to be unconditionally stable. Concurrently, this method and stability results were obtained in a different context and to a different end by Johnston and Liu in [20]. A second order generalization is considered in [20]. Although a fully satisfactory stability proof for the second order extension is still open, preliminary analysis and computations in [20] look promising.

The finite element error in the energy norm for this semi-discrete VMS was analyzed in [10] for convection–diffusion equations and in [17] for the Navier–Stokes equations. Let $e = u - u^h$, one will obtain an optimal order of convergence for $\|\nabla e\|_{L^2(0,T;L^2)}$ if the coarse space L^H is sufficiently fine and if u is sufficiently smooth. Denote by h and H the mesh widths for the fine and the coarse mesh, respectively, the fineness condition on L reads $h \sim H^\beta$ with $\beta \leqslant 2$. Beyond this, many theoretical problems are still open for (5).

3. Numerical studies

This section presents numerical studies comparing the fully implicit approach (6) and the semi-implicit approach (9). The algebraic representations of these approaches are given in (8) and (11), respectively. As mentioned above, the implementation of (6) and (9) seems to us easier than of (12) and (17). In addition, the basis equation (5) of (12) and (17) is equivalent to the VMS (3) only in the case of constant ε_{add}.

The numerical studies were carried out with the code MooNMD [16,19]. The finite element space X^h consists of continuous piecewise linear or bilinear functions, i.e., $X^h = P_1$ on a triangular mesh $\mathcal{T}^h(\Omega)$ and $X^h = Q_1$ on a quadrilateral mesh $\mathcal{T}^h(\Omega)$. For the finite element space L^H on $\mathcal{T}^H(\Omega)$, we have used the simplest choice, namely piecewise constant functions. Note, this choice satisfies all conditions on L^H given in Section 2. For the additional diffusion, numerical tests with $\varepsilon_{\text{add}} = ch$ were performed. Below, the studies with $c = 0.1$ are presented. We found that for larger values of c, with $c \leqslant 1$, the results are qualitatively the same. On the one hand, the errors increased slightly, but on the other hand, the orders of convergence are similar and the computing times became shorter. If c was chosen to be too small, the additional viscosity was not sufficient to stabilize the method. The necessary size of c for the additional viscosity to work properly was dependent on, e.g., the level of the grid hierarchy where the space L^H was defined.

The efficiency of the solver for the algebraic systems is crucial for the efficiency of the whole numerical simulation. We applied a flexible GMRES method [31], with a multigrid method as preconditioner. In the case of the scalar system (11), the preconditioner is a geometric multigrid method. We used the $F(1,1)$-cycle, ILU$_\beta$ with $\beta = 1$ as smoother and the coarse grid system was solved directly by Gaussian elimination. Note, the ILU-decomposition has to be computed only once in the initial time step.

For the coupled system (8), an algebraic multigrid method was applied as preconditioner. Here, the $W(2,2)$-cycle was used, ILU$_\beta$ with $\beta = 1$ as smoother and also as coarse grid solver. The algebraic multigrid method, which is described in [27], belongs to the class of aggregation methods, i.e., the unknowns of the coarse grid are defined by an appropriate clustering of the unknowns of the fine grid. A constant prolongation is applied and the restriction is defined to be the adjoint operator. The coarse grids and coarse grid matrices of the algebraic multigrid have to be constructed before the iterative solution of the linear system can start. For equi-distant time-steps, this has to be done only once at the first discrete time since the matrices are the same for all discrete times. The diagonal entry a_{ii}^H of the coarse grid matrix is the sum of all couplings of the fine grid nodes which are forming cluster i. The off diagonal entry a_{ij}^H, $i \neq j$, is the sum of all fine grid entries a_{kl}^h, $k \neq l$, where node k belongs to cluster i and node l to cluster j.

The numerical studies were performed for (1) with the prescribed solution

$$u(t,\mathbf{x}) = t^2 \cos(x_1 x_2^2), \tag{18}$$

with $\mathbf{x} = (x_1, x_2)$, $\varepsilon = 10^{-8}$, $\mathbf{b} = (2,-1)^T$, $c = 1$, $\Omega = (0,1)^2$ and $T = 10$. The non-homogeneous Dirichlet boundary conditions and the right-hand side f were chosen such that $u(t,\mathbf{x})$ fulfills (1). We decided to use as prescribed solution (18), because the Crank–Nicolson scheme is an exact time integrator in this case and all errors are due to the discretization in space and the stabilization of the convective term. The Crank–Nicolson scheme was applied with an equi-distant time step of length $\Delta t_n = 0.125$.

The initial quadrilateral grid, level 0, consists of four squares with edge length 0.5. As usual, the size of a mesh cell is the longest distance between two of its vertices and the mesh size is the maximum of the sizes of the mesh cells. Accordingly, the mesh size is $h_0 = \sqrt{2}2^{-1}$. The initial triangular grid was obtained from the initial quadrilateral grid by dividing the squares using the diagonals from the left lower corner to the right upper corner. The initial grids were refined uniformly.

The computations have been carried out on a PC with Intel Pentium 4 Processor, with 3 GHz.

In the first numerical study, a fixed fine mesh $\mathcal{T}^h(\Omega)$ is considered and the coarse mesh $\mathcal{T}^H(\Omega)$ is varied. The computations were performed on a quadrilateral mesh. The fine mesh is given by refinement level 6 of the initial grid such that $h_6 = \sqrt{2}2^{-7}$. The number of degrees of freedom (d.o.f.) on this mesh is 16,641 (including Dirichlet nodes). The coarse mesh is varied between level $L = 1$ and $L = 6$ giving $H = \sqrt{2}2^{-(L+1)}$. The results are presented in Table 2.

In order to provide an impression on the accuracy of the computed results with the VMS, the results obtained with the simple artificial diffusion stabilization and the Streamline-Diffusion FEM (SDFEM) are also presented. In the artificial diffusion stabilization, the same parameter ε_{add} was used as in the VMS. The SDFEM has in each discrete time the form, see [14,3]: find $u_k^h \in X^h$ such that for all $v^h \in X^h$

$$\begin{aligned}
&(u_k^h, v^h) + \sum_{K\in\mathcal{T}_h} \tau_K (u_k^h, \mathbf{b}\cdot\nabla v^h)_K + \theta_1 \Delta t_k \left[(\varepsilon \nabla u_k^h, \nabla v^h) + (\mathbf{b}\cdot\nabla u_k^h + c u_k^h, v^h) + \sum_{K\in\mathcal{T}_h} \tau_K (\mathbf{b}\cdot\nabla u_k^h + c u_k^h, \mathbf{b}\cdot\nabla v^h)_K \right] \\
&\quad = (u_{k-1}^h, v^h) + \sum_{K\in\mathcal{T}_h} \tau_K (u_{k-1}^h, \mathbf{b}\cdot\nabla v^h)_K - \theta_2 \Delta t_k \left[(\varepsilon \nabla u_{k-1}^h, \nabla v^h) + (\mathbf{b}\cdot\nabla u_{k-1}^h + c u_{k-1}^h, v^h) + \sum_{K\in\mathcal{T}_h} \tau_K (\mathbf{b}\cdot\nabla u_{k-1}^h + c u_{k-1}^h, b\cdot\nabla v^h)_K \right] \\
&\qquad + \theta_3 \Delta t_k (f_{k-1}, v^h) + \theta_3 \Delta t_k \sum_{K\in\mathcal{T}_h} \tau_K (f_{k-1}, \mathbf{b}\cdot\nabla v^h)_K + \theta_4 \Delta t_k (f_k, v^h) + \theta_4 \Delta t_k \sum_{K\in\mathcal{T}_h} \tau_K (f_k, \mathbf{b}\cdot\nabla v^h)_K.
\end{aligned}$$

Here, $(\cdot,\cdot)_K$ denotes the inner product in $L^2(K)$ where K is a mesh cell. The parameter τ_K is computed as follows:

$$\tau_K = \frac{\alpha_K h_K}{2\|\mathbf{b}\|_2}, \quad Pe_K = \frac{\|\mathbf{b}\|_2 h_K}{2\varepsilon}, \quad \alpha_K = \coth(Pe_K) - \frac{1}{Pe_K}.$$

Here, h_K is the size of the mesh cell K and $\|\mathbf{b}\|_2$ is the Euclidean norm of $\mathbf{b}$.

Let $e = u - u^h$. The results presented in Table 2 show that the semi-implicit approach (9) is much more efficient than the fully implicit one (6). For the fully implicit approach, we were not able to solve the linear systems of equations in the cases $L \in \{5, 6\}$. The stabilization which is introduced by the VMS is not sufficient. In contrast, the semi-implicit approach still works well for $L \in \{5, 6\}$. This results from the extra stabilization inherited in this approach, see the paragraph after (9). Since there is no stabilization coming directly from the projection for $L = 6$, the extra stabilization by itself is sufficient in the considered example. The accuracy of the results with both approaches is practically the same for $L \leqslant 4$. The results become more accurate the finer the coarse grid becomes. Even for a very coarse space L^H, the results with the VMS are considerably more accurate than the results with the artificial diffusion stabilization. If L^H is defined on finer and finer grids, the results become more and more accurate. Finally, if the coarse space is defined on the same grid as the fine space, $L = 6$, the results are nearly as accurate as for the SDFEM. Here we like to emphasize that a very simple model for ε_{add} has been used. The accuracy of the VMS can be certainly improved by applying more sophisticated models. An important observation is that the use of finer coarse grids $\mathcal{T}^H(\Omega)$ and the corresponding increase of degrees of freedom in L^H practically does not lead to an increase in the computing times for the semi-implicit approach (9). The computational overhead of this method, four matrix–vector products and multiplication with the diagonal matrix D^{-1}, is small in comparison to the time needed for solving the linear system.

Table 2
Results for a fixed fine mesh and varying coarse mesh

Method	L	D.o.f. L^H	$\|e\|_{L^\infty(0,T;L^2)}$	$\|e\|_{L^2(0,T;L^2)}$	$\|\nabla e\|_{L^2(0,T;L^2)}$	Time
(6)	1	32	1.078e−2	1.520e−2	1.637e+0	184.4
	2	128	5.927e−3	8.365e−3	1.227e+0	181.9
	3	512	3.316e−3	4.679e−3	9.141e−1	183.0
	4	2048	2.099e−3	2.960e−3	7.349e−1	373.1
(9)	1	32	1.076e−2	1.518e−2	1.621e+0	103.5
	2	128	5.960e−3	8.425e−3	1.214e+0	103.2
	3	512	3.381e−3	4.800e−3	9.050e−1	103.1
	4	2048	2.191e−3	3.130e−3	7.281e−1	102.8
	5	8192	1.762e−3	2.531e−3	6.604e−1	103.0
	6	32,768	7.052e−4	1.094e−3	3.526e−1	103.1
Art. diff.			1.806e−1	2.538e−1	5.490e+0	56.3
SDFEM			5.213e−4	7.344e−4	3.414e−1	89.2

The second numerical test studies the convergence of the errors if the ratio of the fine mesh width h and the coarse mesh width H is kept (nearly) constant. We tested the scalings $h \sim H^{\beta}$, $\beta \in \{5/4, 3/2, 7/4, 2\}$. The scaling $h \sim H^2$ was proposed in [26] for the steady state equations. This scaling is the limit case for an optimal order of convergence of $\|\nabla e\|_{L^2(0,T;L^2)}$, see the discussion of finite element error estimates at the end of Section 2. Neglecting for simplicity the factor $\sqrt{2}$ in the mesh widths, then $h = 2^{-(l+1)}$ and $H = 2^{-(L+1)}$. Let l be given, one obtains $L = \beta^{-1}(l+1) - 1$. Since this number is in general not an integer, we used in the computations the nearest integer to $\beta^{-1}(l+1) - 1$ as value for L. The computations were performed on triangular grids with piecewise linear finite elements. The coarsest grid (level 0) consisted of 8 triangles. The degrees of freedom for the spaces X^h and L^H are given in Table 3.

In Tables 4–7, errors in several norms are presented. The orders of convergence are given with respect to h. They were computed using the values on the two finest levels.

The numerical results show that one obtains first order of convergence for $\|\nabla e\|_{L^2(0,T;L^2)}$ if $\beta \in \{5/4, 3/2, 7/4\}$. For the limit value $\beta = 2$, a first order of convergence is not yet reached. A second order convergence for $\|e\|_{L^\infty(0,T;L^2)}$ and $\|e\|_{L^2(0,T;L^2)}$ cannot be observed. These observations correspond to the available analytical results, see the end of Section 2. For the SDFEM, we found for this example first order convergence in $\|\nabla e\|_{L^2(0,T;L^2)}$ and second order in $\|e\|_{L^2(0,T;L^2)}$.

Table 3
Degrees of freedom on the triangular grids

Level	X^h	L^H
0		16
1	25	64
2	81	256
3	289	1024
4	1089	4096
5	4225	16,384
6	16,641	65,536
7	66,049	262,144
8	263,169	
9	1,050,625	

Table 4
Results for a fixed ratio $h \sim H^{5/4}$ of the fine mesh width and the coarse mesh width

L	l	$\|e\|_{L^\infty(0,T;L^2)}$	$\|e\|_{L^2(0,T;L^2)}$	$\|\nabla e\|_{L^2(0,T;L^2)}$
1	2	5.225e−1	7.335e−1	1.619e+1
2	3	1.554e−1	2.185e−1	9.432e+0
3	4	4.443e−2	6.254e−2	5.241e+0
4	5	1.218e−2	1.719e−2	2.771e+0
5	6	3.250e−3	4.611e−3	1.422e+0
5	7	7.358e−4	1.062e−3	5.221e−1
6	8	2.046e−4	3.038e−4	2.623e−1
7	9	6.262e−5	9.767e−5	1.316e−1
Order		1.708	1.637	0.995

Table 5
Results for a fixed ratio $h \sim H^{3/2}$ of the fine mesh width and the coarse mesh width

L	l	$\|e\|_{L^\infty(0,T;L^2)}$	$\|e\|_{L^2(0,T;L^2)}$	$\|\nabla e\|_{L^2(0,T;L^2)}$
1	2	5.225e−1	7.335e−1	1.619e+1
2	3	1.554e−1	2.185e−1	9.432e+0
2	4	3.901e−2	5.506e−2	3.957e+0
3	5	1.046e−2	1.481e−2	2.038e+0
4	6	2.762e−3	3.934e−3	1.035e+0
4	7	9.844e−4	1.409e−3	5.613e−1
5	8	2.655e−4	3.878e−4	2.820e−1
6	9	7.683e−5	1.168e−4	1.415e−1
Order		1.789	1.731	0.995

Table 6
Results for a fixed ratio $h \sim H^{7/4}$ of the fine mesh width and the coarse mesh width

L	l	$\|e\|_{L^\infty(0,T;L^2)}$	$\|e\|_{L^2(0,T;L^2)}$	$\|\nabla e\|_{L^2(0,T;L^2)}$
1	3	1.407e−1	1.983e−1	7.507e+0
2	4	3.901e−2	5.506e−2	3.957e+0
2	5	1.426e−2	2.015e−2	2.188e+0
3	6	3.748e−3	5.318e−3	1.113e+0
4	7	9.844e−4	1.409e−3	5.613e−1
4	8	4.246e−4	6.096e−4	3.422e−1
5	9	1.157e−4	1.700e−4	1.718e−1
Order		1.876	1.842	0.994

Table 7
Results for a fixed ratio $h \sim H^2$ of the fine mesh width and the coarse mesh width

L	l	$\|e\|_{L^\infty(0,T;L^2)}$	$\|e\|_{L^2(0,T;L^2)}$	$\|\nabla e\|_{L^2(0,T;L^2)}$
0	1	1.696e+0	2.380e+0	2.708e+1
1	3	1.407e−1	1.983e−1	7.507e+0
2	5	1.426e−2	2.015e−2	2.188e+0
3	7	1.614e−3	2.292e−3	6.794e−1
4	9	1.980e−4	2.846e−4	2.224e−1
Order		1.514	1.505	0.806

4. Summary

The paper studied a two-level variational multiscale method for convection–diffusion equations. This method possesses two parameters: an additional diffusion ε_{add} and a vector-valued coarse finite element space L^H. The two main topics of the study were the conditions on L^H which are necessary for an efficient implementation of the method and the treatment of the additional terms of the VMS within the temporal discretization. It was shown that an efficient implementation of the VMS can be achieved if L^H consists of discontinuous finite element functions and if the basis of L^H is L^2-orthogonal. In this case, the additional matrices possess a very sparse structure and the inversion of the mass matrix of L^H, which is needed in the semi-implicit approach, can be done easily. Numerical tests at a model problem showed that a semi-implicit temporal discretization, which treats the subtraction of the additional diffusion from the large scales explicitly, is much more efficient than a fully implicit discretization. The computing times in the semi-implicit VMS practically did not depend on the dimension of the coarse space L^H since the computational overhead is small in comparison to the time which is needed to solve the linear system. This property will be shared by other temporal discretizations which treat the last term on the left-hand side of (3) explicitly. A comparison of this method with the SDFEM showed a similar order of convergence in several norms for appropriate scalings of the fine and the coarse mesh.

An extension of the two-level method to the Navier–Stokes equations requires similar algorithmic considerations as for scalar convection–diffusion equations. Based on the results obtained in this paper, discontinuous finite elements for the space L^H and the semi-implicit approach should be used. For the Navier–Stokes equations, an additional viscosity has to be chosen based on physical ideas in turbulence modeling. This will lead to non-linear models, e.g., the Smagorinsky model [32].

A one-level variant of the VMS presented in this paper applied in the simulation of turbulent flows can be found in [18]. Using higher order finite elements for the velocity, the coarse scale space L^H was defined on the same grid using low order polynomials. The implementation of the two-level variant for the Navier–Stokes equations and its comparison with the variant presented in [18] will be future work.

Acknowledgements

The authors would like to thank Ramon Codina and the referee of this paper for their very useful comments and suggestions.

References

[1] M. Anitescu, W. Layton, F. Pahlevani, Implicit for local effects and explicit for nonlocal effects is unconditionally stable, ETNA 18 (2004) 174–187.
[2] G.Q. Chen, Q. Du, E. Tadmor, Spectral viscosity approximations for multi-dimensional scalar conservation laws, Math. Comput. 61 (1993) 629–643.

[3] R. Codina, Comparison of some finite element methods for solving the diffusion–convection–reaction equation, Comput. Methods Appl. Mech. Engrg. 156 (1998) 185–210.

[4] R. Codina, Stabilization of incompressibility and convection through orthogonal sub-scales in finite element methods, Comput. Methods Appl. Mech. Engrg. 13 (2000) 505–512.

[5] M. Fortin, R. Guenette, R. Pierre, Numerical analysis of the EVSS method, Comput. Methods Appl. Mech. Engrg. 143 (1997) 79–95.

[6] L.P. Franca, A. Nesliturk, On a two-level finite element method for the incompressible Navier–Stokes equations, Int. J. Numer. Methods Engrg. 52 (2001) 433–453.

[7] V. Gravemeier, The variational multiscale method for laminar and turbulent incompressible flow, Ph.D. thesis, Institute of Structural Mechanics, University of Stuttgart, 2003.

[8] V. Gravemeier, W.A. Wall, E. Ramm, A three-level finite element method for the instationary incompressible Navier–Stokes equation, Comput. Methods Appl. Mech. Engrg. 193 (2004) 1323–1366.

[9] J.L. Guermond, Stabilization of Galerkin approximations of transport equations by subgrid modelling, Math. Modell. Numer. Anal. (1999) 1293–1316.

[10] N. Heitmann, Subgrid stabilization of evolutionary diffusive transport problems, Ph.D. thesis, University of Pittsburgh, 2004.

[11] T.J. Hughes, L. Mazzei, K.E. Jensen, Large eddy simulation and the variational multiscale method, Comput. Vis. Sci. 3 (2000) 47–59.

[12] T.J. Hughes, A. Oberai, L. Mazzei, Large eddy simulation of turbulent channel flows by the variational multiscale method, Phys. Fluids 13 (2001) 1784–1799.

[13] T.J.R. Hughes, Multiscale phenomena: Green's functions, the Dirichlet-to-Neumann formulation, subgrid-scale models, bubbles and the origin of stabilized methods, Comput. Methods Appl. Mech. Engrg. 127 (1995) 387–401.

[14] T.J.R. Hughes, A.N. Brooks, A multidimensional upwind scheme with no crosswind diffusion, in: T.J.R. Hughes (Ed.), Finite Element Methods for Convection Dominated Flows, AMD, vol. 34, ASME, New York, 1979, pp. 19–35.

[15] T.J.R. Hughes, L. Mazzei, A.A. Oberai, A.A. Wray, The multiscale formulation of large eddy simulation: decay of homogeneous isotropic turbulence, Phys. Fluids 13 (2001) 505–512.

[16] V. John, Large eddy simulation of turbulent incompressible Flows, in: Analytical and Numerical Results for a Class of LES Models, Lecture Notes in Computational Science and Engineering, vol. 34, Springer-Verlag, Berlin, Heidelberg, New York, 2004.

[17] V. John, S. Kaya, Finite element error analysis of a variational multiscale method for the Navier–Stokes equations, Adv. Comput. Math., in press.

[18] V. John, S. Kaya, A finite element variational multiscale method for the Navier–Stokes equations, SIAM J. Sci. Comput. 26 (2005) 1485–1503.

[19] V. John, G. Matthies, MooNMD—a program package based on mapped finite element methods, Comput. Vis. Sci. 6 (2004) 163–170.

[20] H. Johnston, J.-L. Liu, Accurate, stable and efficient Navier–Stokes solvers based on explicit treatment of the pressure term, J. Comput. Phys. 199 (2004) 221–259.

[21] S. Kaya, Numerical analysis of subgrid-scale eddy viscosity for the Navier–Stokes equations, Technical Report, University of Pittsburgh, 2002.

[22] S. Kaya, W. Layton, Subgrid-scale eddy viscosity methods are variational multiscale methods, Technical Report, University of Pittsburgh, 2002.

[23] S. Kaya, B. Riviere, A discontinuous subgrid eddy viscosity method for the time dependent Navier–Stokes equations, SIAM J. Numer. Anal. 43 (2005) 1572–1595.

[24] S. Kaya, B. Riviere, A two-grid stabilization method for solving the steady-state Navier–Stokes equations, Numer. Methods Part. Differen. Equat., in press.

[25] W. Layton, Variational multiscale methods and subgrid scale eddy viscosity, in: H. Deconinck (Ed.), 32nd, 2002–2006, Computational Fluid Dynamics Multiscale Methods, Von Karman Institute for Fluid Dynamics, Rhode-Saint-Genèse, Belgium, 2002.

[26] W.J. Layton, A connection between subgrid scale eddy viscosity and mixed methods, Appl. Math. Comput. 133 (2002) 147–157.

[27] R. Lonsdale, An algebraic multigrid solver for the Navier–Stokes equations on unstructured meshes, Int. J. Numer. Methods Heat Fluid Flow 3 (1993) 3–14.

[28] Y. Maday, E. Tadmor, Analysis of the spectral vanishing method for periodic conservation laws, SIAM J. Numer. Anal. 26 (1989) 854–870.

[29] F. Pahlevani, Sensitivity analysis of eddy viscosity models, Ph.D. thesis, University of Pittsburgh, 2004.

[30] H.-G. Roos, M. Stynes, L. Tobiska, Numerical Methods for Singularly Perturbed Differential Equations, Springer, 1996.

[31] Y. Saad, A flexible inner–outer preconditioned GMRES algorithm, SIAM J. Sci. Comput. 14 (2) (1993) 461–469.

[32] J.S. Smagorinsky, General circulation experiments with the primitive equations, Mon. Weather Rev. 91 (1963) 99–164.