Fluid Dynamics & Materials Processing

Tech Science Press

DOI: 10.32604/fdmp.2022.021848

ARTICLE

A Variational Multiscale Method for Particle Dispersion Modeling in the Atmosphere

Y. Nishio[1,*], B. Janssens[1], K. Limam[2] and J. van Beeck[3]

[1]Royal Military Academy (RMA), Brussels, 1000, Belgium

[2]Universite de La Rochelle (ULR), La Rochelle, 17000, France

[3]Von Karman Institute for Fluid Dynamics (VKI), Sint-Genesius-Rode, 1640, Belgium

[*]Corresponding Author: Y. Nishio. Email: yoshiyuki.nishio@rma.ac.be

Received: 08 February 2022 Accepted: 04 March 2022

ABSTRACT

A LES model is proposed to predict the dispersion of particles in the atmosphere in the context of Chemical, Biological, Radiological and Nuclear (CBRN) applications. The code relies on the Finite Element Method (FEM) for both the fluid and the dispersed solid phases. Starting from the Navier-Stokes equations and a general description of the FEM strategy, the Streamline Upwind Petrov-Galerkin (SUPG) method is formulated putting some emphasis on the related assembly matrix and stabilization coefficients. Then, the Variational Multiscale Method (VMS) is presented together with a detailed illustration of its algorithm and hierarchy of computational steps. It is demonstrated that the VMS can be considered as a more general version of the SUPG method. The final part of the work is used to assess the reliability of the implemented predictor/multicorrector solution strategy.

KEYWORDS

ABL (Atmospheric boundary layer); CFD; FEM; LES; SUPG (Streamline upwind petrov–Galerkin); Turbulent flow; VMS (Variational MultiScale method)

Nomenclature

A_e [-]	Assembly matrix for the elements
N_i [-]	Shape function
R^X [-]	Residual vector for X
Δt [s]	$= t_{n+1} - t_n$
Ω [m^3]	Space domain
α_X [-]	α-method for the X parameter
v [m^2/s]	Kinematic viscosity
τ_X [-]	Stabilization coefficient for X
$\tilde{u}_{adv}$ [m/s^2]	Reference advection velocity
p [$kg/(ms^2)$]	Pressure
t [s]	Time
u [m/s]	*Velocity*

Superscripts and Subscripts

adv	Advection
BU	Bulk viscosity stabilization coefficient
e	Per element
F,M	parameters for the α-method
l	Multi-corrector iteration number
M, C	Momentum, Continuity
n	Timestep
PS	Pressure stabilization coefficient
SU	Stream-upwind stabilization coefficient
T	Transpose

1 Introduction

1.1 Previous Works and Evolutions

In 2014, a DNS model for fluids laden with small particles was developed in the thesis of Janssens [1]. The model, using FEM for both the fluid and the dispersed phase, was validated by reference and experimental CFD case studies (e.g., "Turbulent channel flow" case from Fig. 1, "Taylor-Green vortex flow" case and "Turbulence-induced coalescence in aerosols" case).

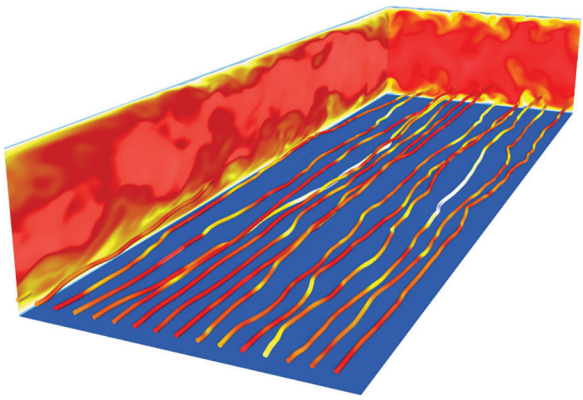

Figure 1: Turbulent channel flow (source: [1])

Consequently to these validations, the aim of the proposed CFD model is to predict the dispersion of particles in the atmosphere, for an area on the order of the hectometer, taking into account detailed geometries of the topography. However, correctly resolving the atmospheric boundary layer is not straight forward and its representation needs to be validated first. In article [2], a Schumann's wall model was implemented and tested in 2D. It used the Streamline Upwind Petrov-Galerkin (SUPG) stabilization method implemented by Janssens [1] to stabilize the numerical oscillations. The results were encouraging but in 3D, spurious numerical fluctuations near the wall were observed. There are suspicions concerning the stabilization currently used in our finite element method. It appears as insufficient to damp the oscillations occurring with high Reynolds number flow. So, we intend to amend it using the techniques from the Variational Multiscale Method (VMS), which provides a combined framework for stabilization and turbulence modeling. In this article, the proposed solution is first described and then compared

theoretically to the SUPG method. In the second part, a comparison is proposed on the Taylor-Green test case and the results are being analyzed.

2 Theory

In current work, the described methods are implicit. Thus, the system of equations is coupling the velocity and the pressure.

2.1 Navier-Stokes in FEM

The model that is being implemented is based on the incompressible two Navier-Stokes transport equations. These can be written following [1]:

$$\nabla \cdot u = 0$$

$$\frac{\partial u}{\partial t} + (u \cdot \nabla)u + \frac{u(\nabla \cdot u)}{2} + \nabla p - \nu \nabla^2 u = 0 \tag{1}$$

The finite element formulation of the continuity and momentum equations can be achieved by multiplying these equations with weighting functions. The unknown variables can be interpolated between the discrete nodes by using shape functions and eventually, these equations can be integrated over the whole domain. The expressions can be simplified by choosing weighting functions equal to shape functions, which leads to a Galerkin formulation. After applying a θ-method to the time derivative and because the shape functions are non-zero only on their respective node and surrounding element, the integrals for all the elements can be replaced by a sum of the integral on each element. By solving this discrete system, the pressure and the velocity for the next timestep can be found:

$$\sum_{e=1}^{N} \left(\frac{1}{\Delta t} T_e + \theta A_e \right) (x_e^{n+1} - x_e^n) = -A_e x_e^n \tag{2}$$

where θ is set to 1 for a forward Euler scheme or 0.5 for a Crank-Nicolson scheme. By default, it is set to 0.5 in [1]. The unknown x_e^n for each element, grouped by nodal values, have following format for a 3D element with m + 1 nodes:

$$x_e^n = [p_0^n \cdots p_m^n (u_0^n)_0 \cdots (u_0^n)_m \cdots (u_2^n)_m] \tag{3}$$

Matrices A_e and T_e have the following structure:

$$A_e = \begin{bmatrix} A_{pp} & A_{pu} \\ A_{up} & A_{uu} \end{bmatrix} = \begin{bmatrix} A_{pp} & A_{pu_0} & A_{pu_1} & A_{pu_2} \\ A_{u_0p} & A_{u_0u_0} & A_{u_0u_1} & A_{u_0u_2} \\ A_{u_1p} & A_{u_1u_0} & A_{u_1u_1} & A_{u_1u_2} \\ A_{u_2p} & A_{u_2u_0} & A_{u_2u_1} & A_{u_2u_2} \end{bmatrix} \tag{4}$$

2.2 SUPG Stabilization Method

By applying the SUPG stabilization method detailed in [3] to Eq. (1), each block of A_e can be formulated as follows:

$$A_{pp} = \int_{\Omega_e} \tau_{PS} \nabla N_p^T \nabla N_p d\Omega_e \tag{5}$$

$$A_{pu_i} = \int_{\Omega_e} \left(\left(N_p + \frac{\tau_{PS} \tilde{u}_{adv} \nabla N_p}{2} \right)^T (\nabla N_u)_i + \tau_{PS} (\nabla N_p)_i^T \tilde{u}_{adv} \nabla N_u \right) d\Omega_e \tag{6}$$

$$A_{u_ip} = \int_{\Omega_e} (N_u + \tau_{SU}\tilde{u}_{adv}\nabla N_u)^T \nabla N_p d\Omega_e \tag{7}$$

$$A_{u_iu_i} = \int_{\Omega_e} (\nu \nabla N_u^T \nabla \mathrm{N_u} + (N_u + \tau_{SU}\tilde{u}_{adv}\nabla N_u)^T \tilde{u}_{adv}\nabla N_u) d\Omega_e + A_{u_iu_j} \tag{8}$$

$$A_{u_iu_j} = \int_{\Omega_e} \left(\tau_{BU}(\nabla N_u)_i + \frac{1}{2}(\tilde{u}_{adv})_i(N_u + \tau_{SU}u\nabla N_u)\right)^T (\nabla N_u)_j d\Omega_e \tag{9}$$

$$T_{pu_i} = \int_{\Omega_e} \tau_{PS}(\nabla N_p)_i^T N_u d\Omega_e \tag{10}$$

$$T_{u_iu_i} = \int_{\Omega_e} (N_u + \tau_{SU}\tilde{u}_{adv}\nabla N_u)^T N_u d\Omega_e \tag{11}$$

where $\tilde{u}_{adv}$ is the advection velocity, obtained, in [1], by a Taylor expansion of the previous velocities. And where the stabilization terms, multiplied by their respective stabilization coefficients (τ_{PS}, $_{SU}$, $_{BU}$), are presented in [4]. These coefficients have to be chosen in such a manner that the numerical values are properly stabilized but not over-dissipated. This stabilization method was implemented by Janssens in [1] and successfully helped to solve oscillations for reasonable Reynolds simulation (e.g., the channel flow test case). Nevertheless, to be able to tackle the spurious numerical oscillations observed in [5], the idea is to use a generalized version of the SUPG stabilization method, called Variational Multiscale method (VMS), introduced by Hughes in [6]. The next section will present an implementation of the VMS algorithm proposed by Bazilevs [7].

2.3 Variational Multiscale Method (VMS)

The VMS applies the scale separation (coarse *vs.* fine) primarily and only approximate the fines scales. In these small scales, the stabilization terms appear "naturally".

2.3.1 Structure

The proposed VMS method follows a predictor/multi-corrector structure, schematized in Fig. 2. This method has more parameters than the standard SUPG method, allowing it to better respond to numerical oscillations.

2.3.2 Stages

Predictor stage

For each timestep, during the first stage, the predictor stage, the three variables (velocity u, derivative of the velocity $\dot{u}$ and pressure p) are set according to previous timestep.

$$U_{n+1,(0)} = U_n \tag{12}$$

$$\dot{U}_{n+1,(0)} = \frac{\gamma - 1}{\gamma}\dot{U}_n \tag{13}$$

$$P_{n+1,(0)} = P_n \tag{14}$$

γ (as well as α_f and α_m that are presented in this section) is a real-valued parameter defining the α-method. It is developed in [8,9].

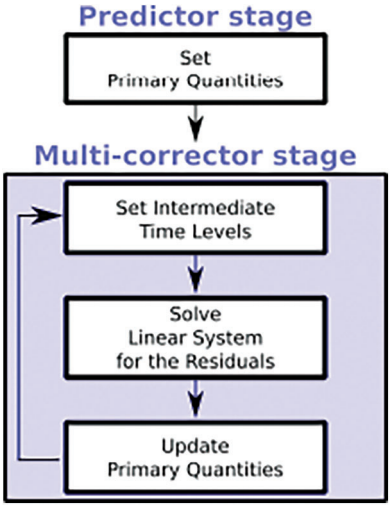

Figure 2: Predictor/Multi-corrector's structure

Multi-corrector stage

Then, a number of iterations ($l = 1, 2, \ldots, l_{max}$) is performed (typically between 2 and 4) to converge to steady values, or in other words, the residuals for, both, the continuity and momentum equations of Eq. (1) are minimized. To be more explicit, during these iterations, the following steps are performed:

1) The first step is to set the intermediate time levels:

$$\dot{U}_{n+\alpha_m,(l)} = \dot{U}_n + \alpha_m(\dot{U}_{n+1,(l-1)} - \dot{U}_n) \tag{15}$$

$$U_{n+\alpha_f,(l)} = U_n + \alpha_f(U_{n+1,(l-1)} - U_n) \tag{16}$$

$$P_{n+1,(l)} = P_{n+1,(l-1)} \tag{17}$$

α_f and α_m are intermediate time level parameters. They (as well as previously mentioned γ) are selected, taking accuracy and stability considerations into account. According to [9], obtaining second-order accuracy in time is possible if:

$$\gamma = \frac{1}{2} + \alpha_m - \alpha_f \tag{18}$$

and the method is unconditionally stable if:

$$\alpha_m \geq \alpha_f \geq \frac{1}{2} \tag{19}$$

2) These intermediate values are used to assemble the residuals of the continuity and momentum equations and solve the following linear system:

$$A_{iu}\Delta\dot{U}_{n+1,(l)} + A_{ip}\Delta P_{n+1,(l)} = -R^M_{(l)} \tag{20}$$

$$A_{pu}\Delta\dot{U}_{n+1,(l)} + A_{pp}\Delta P_{n+1,(l)} = -R^C_{(l)} \tag{21}$$

that can be rearranged as below:

$$\begin{bmatrix} A_{\dot{u}\dot{u}} & A_{\dot{u}p} \\ A_{p\dot{u}} & A_{pp} \end{bmatrix} \begin{bmatrix} \Delta\dot{U}_{n+1,(l)} \\ \Delta P_{n+1,(l)} \end{bmatrix} = \begin{bmatrix} -R^M_{(l)} \\ -R^C_{(l)} \end{bmatrix} \tag{22}$$

3) This will provide values for the derivative of the velocity and the pressure that can be updated in the last step of the multi-stage corrector:

$$\dot{U}_{n+1,(l)} = \dot{U}_{n+1,(l-1)} + \Delta\dot{U}_{n+1,(l)} \tag{23}$$

$$U_{n+1,(l)} = U_{n+1,(l-1)} + \gamma\Delta t\Delta\dot{U}_{n+1,(l)} \tag{24}$$

$$P_{n+1,(l)} = P_{n+1,(l-1)} + \Delta P_{n+1,(l)} \tag{25}$$

In terms of computation, the second step of the multi-corrector stage, namely, assembling and solving the linear system, is the costliest.

2.3.3 VMS Assembly Matrix A_e

Below, compared to Eqs. (5)–(11), a slightly different assembly matrix A_e is provided:

$$A_e = \begin{bmatrix} A_{\dot{u}\dot{u}} & A_{\dot{u}p} \\ A_{p\dot{u}} & A_{pp} \end{bmatrix} \tag{26}$$

Indeed, in this case, the first block depends on the derivative of the velocity $\dot{u}$ and the last block on the pressure p. This structure was chosen to match the proposed implementation of [7] but it is equivalent to the one presented in the SUPG Section 2.2.

For each block, the expression is detailed hereafter:

$$A_{\dot{u}_i\dot{u}_i} = \int_{\Omega_e} (\alpha_m N^T_{\dot{u}} N_{\dot{u}} + \alpha_m (u\tau_M \nabla N_{\dot{u}})^T N_{\dot{u}} + \alpha_f \gamma\Delta t N^T_{\dot{u}} u \nabla N_{\dot{u}} + \alpha_f \gamma\Delta t (\nabla N_{\dot{u}} v)^T \nabla N_{\dot{u}} + \alpha_f \gamma\Delta t (u\nabla N_{\dot{u}}\tau_M)^T (u\nabla N_{\dot{u}}))d\Omega_e \tag{27}$$

$$A_{\dot{u}_i\dot{u}_j} = \int_{\Omega_e} (\alpha_f \gamma\Delta t (\nabla N_{\dot{u}})^T_j v (\nabla N_{\dot{u}})_i + \alpha_f \gamma\Delta t (\nabla N_{\dot{u}})^T_i \tau_C (\nabla N_{\dot{u}})_j)d\Omega_e \tag{28}$$

$$A_{\dot{u}_i p} = \int_{\Omega_e} -(\nabla N_{\dot{u}})^T_i N_p + (u^T \tau_M (\nabla N_{\dot{u}})_i)^T \nabla N_p d\Omega_e \tag{29}$$

$$A_{p\dot{u}_i} = \int_{\Omega_e} (\alpha_f \gamma\Delta t N^T_p (\nabla N_{\dot{u}})_i + \alpha_f \gamma\Delta t (\nabla N_p)^T_i \tau_M u \nabla N_{\dot{u}} + \alpha_m (\nabla N_p)^T_i \tau_M N_{\dot{u}})d\Omega_e \tag{30}$$

$$A_{pp} = \int_{\Omega_e} (\nabla N_p)^T \tau_M \nabla N_p d\Omega_e \tag{31}$$

As previously described, the $N_{[\dot{u},p]}$ represent the shape functions; $\alpha_{[f,m]}$ and γ the intermediate time level parameters; Δt the difference between timestep t_{n+1} and t_n; u is the advection velocity (different of $\tilde{u}_{adv}$ because computed during the simulation and not constructed from a Taylor serie); v is the kinematic viscosity; and $\tau_{[C,M]}$ are the stabilizing coefficients that still needs to be described thoroughly.

2.3.4 Defining the Stabilizing Coefficients

τ_M and τ_C are the stabilizing coefficients associated to the momentum and continuity equations. Below, their respective expressions:

$$\tau_M = \left(\frac{4}{\Delta t^2} + u^h \cdot G u^h + C_1 v^2 G : G \right)^{-\frac{1}{2}} \tag{32}$$

$$\tau_C = (\tau_M g \cdot g)^{-1} \tag{33}$$

where C_1 is a positive constant, derived from an element-wise inverse estimate described in [10] and set to 1 in our work. And where three specific operations have to be introduced:

1. The tensor product $G{:}\, G = \sum_{i,j=1}^{3} G_{ij} G_{ij}$
2. The tensor product $u^h \cdot G u^h = \sum_{i,j=1}^{3} u_i^h G_{ij} (u_j^h)^T$
3. The tensor product $g \cdot g = \sum_{i=1}^{3} g_i (g_i)^T$

where $G_{ij} = \sum_{k=1}^{3} J_{k,i}^{-1} (J_{k,j}^{-1})^T$ is the element metric tensor, computed from the element shape function inverse Jacobian matrix $J^{-1} = \left(\frac{\partial x_i}{\partial \xi_k} \right)^{-1}$. For the last item $g_i = \sum_{j=1}^{3} (J^{-1})_{j,i}$.

2.4 SUPG vs. VMS Assembly Matrix Comparison

Now that both SUPG (2.2) and VMS (2.3) are described, it is convenient to show that, indeed, the VMS method can be considered as a generalized version of the older and more used SUPG method.

To ease the comprehension, the intermediate time level parameters are set to unity: $\alpha_f = \alpha_m = \gamma = 1$, and Δt is set equal to 1 making the value of $\dot{u}$ and u equal.

For each block of the assembly matrices, the observation is given:

1. *PP* block (Eqs. (5), (31)): Both equations are expressing the shape functions related to the pressure and can be considered equivalent, assuming $\tau_{PS} \equiv \tau_M$
2. *PU*$_i$ block (Eqs. (6), (10), (30)): Both A_e and T_e for SUPG are gathered into A_e in VMS. Besides, Eq. (6) is proportional to the two first terms of Eq. (30), while the third term in Eq. (30) is equivalent to Eq. (10).

Note that the second term of the SUPG formulation is written in skew symmetric form to improve its stability and accuracy properties.

3. *U*$_i$*P* block (Eqs. (7), (29)): After an integration by parts, the first term of Eq. (7) can be expressed as the first term of Eq. (29). The second terms are directly equivalent.

This also implies that, for setting the pressure to 0 on a boundary (i.e., at the outlet), nothing has to be done. It is the "do-nothing" boundary condition [11].

4. *U*$_i$*U*$_i$ block (Eqs. (8), (11), (27)): All the terms are identical.
5. *U*$_i$*U*$_j$ block (Eqs. (9), (28)): In the last block, there is a clear distinction between the SUPG and the VMS formulation. For the first term of Eq. (9), supposing $\tau_{BU} \equiv \tau_C$, it can be considered equivalent to the second term of Eq. (28). The second term of Eq. (9) is the skew-symmetric form in the SUPG matrix while the viscous tensor in VMS is treated differently.

3 Results

Now that the VMS theory that is implemented has been described, a few tests will be presented.

Note that, to reduce the computer cost, all tests are performed in 2D.

The first section is focused on the structure of the algorithm whereas the second, on the implementation.

3.1 VMS Structure: Poisson and Taylor-Green

The VMS structure described in Section 2.3.1 is a predictor/multi-corrector structure. To confirm the correctness of our implementation, it was first tested qualitatively, with classical equations that have analytical values.

The two chosen tests are the Poisson equation (with a uniform velocity boundary condition) for its simple implementation and the Taylor-Green equation for its periodical boundary conditions (Fig. 3).

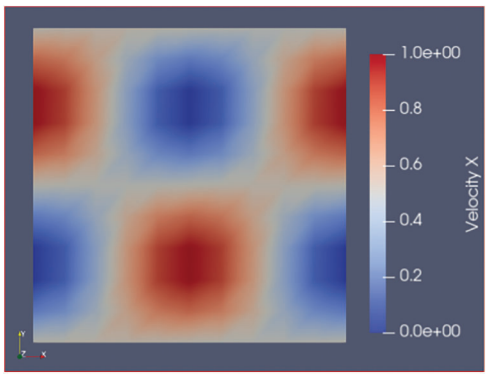

Figure 3: Taylor-Green test case

The objective of these tests was to ensure the output, with the new structure, would correlate with the one produced with a standard Navier-Stokes structure.

3.2 Tests for the VMS Implementation

Having acquired confidence in the structure, the next step was to test the algorithm's implementation. The latter is more complicated because the only comparison that can be performed is to compare the assembly matrix A_e between the SUPG implementation and the VMS algorithm implementation, when both the viscosity ν and the stabilization coefficients τ_x are reduced to zero.

To better analyse how each equation described in Section 2.3.3 is influencing A_e, they are schemed with blocks in Table 1. Note that each block is actually a 3×3 matrix.

Table 1: A_e structure

$A_{u_iu_i}$	$A_{u_iu_j}$	A_{u_ip}
$A_{u_iu_j}$	$A_{u_iu_i}$	A_{u_ip}
A_{pu_i}	A_{pu_i}	A_{pp}

Note: To ease the reading, the A_e matrix will be printed, limited to 3 decimals (sufficient to see the differences).

3.2.1 Results with $v = \tau_x = 0$

A simulation was performed, for an identical mesh, with both the SUPG implementation and the VMS implementation. Table 2 displays A_e for the SUPG implementation. This is the reference result.

Table 2: A_e with $v = 0$ for SUPG

−0.182 0.000 0.182 −0.182 0.000 0.182 −0.182 0.000 0.182	0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000	−0.083 0.083 0.000 −0.083 0.083 0.000 −0.083 0.083 0.000
0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000	−0.182 0.000 0.182 −0.182 0.000 0.182 −0.182 0.000 0.182	0.000 −0.083 0.083 0.000 −0.083 0.083 0.000 −0.083 0.083
−0.083 0.083 0.000 −0.083 0.083 0.000 −0.083 0.083 0.000	0.000 −0.083 0.083 0.000 −0.083 0.083 0.000 −0.083 0.083	0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000

In Table 2, one can see the matrix is symmetrical, considering the blocks. Another observation is that $A_{u_i u_j}$ and A_{pp} blocks are null value. Eventually, all blocks have a column structure (values varying only column-wise).

The next table (Table 3) proposes the VMS implementation (Eqs. (27)–(31)).

Table 3: A_e with $u = 0$ for VMS

−0.083 0.000 0.083 −0.083 0.000 0.083 −0.083 0.000 0.083	0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000	0.083 0.083 0.083 −0.083 −0.083 −0.083 0.000 0.000 0.000
0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000	−0.083 0.000 0.083 −0.083 0.000 0.083 −0.083 0.000 0.083	0.000 0.000 0.000 0.083 0.083 0.083 −0.083 −0.083 −0.083
−0.083 0.083 0.000 −0.083 0.083 0.000 −0.083 0.083 0.000	0.000 −0.083 0.083 0.000 −0.083 0.083 0.000 −0.083 0.083	0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000

When analysing Table 3 for the similarities, the observation is that the null valued blocks are identical. For the differences, the first remark is that A_e is column-wise except for A_{uip} that is row-wise and has the opposite sign, compared to SUPG values. Second, the Auiui values are not equal to SUPG ones.

After modifying $A_{u_i p}$ by transposing it and taking its opposite value, Table 3 becomes Table 4.

In the latter, the only difference resides in $A_{u_i u_i}$. This issue is still under investigation. When the issue will be solved, the next step will be to reactivate the stabilization coefficients and the viscosity to study their behavior.

Table 4: Modified A_e with $u = 0$ for VMS

−0.083 0.000 0.083	0.000 0.000 0.000	−0.083 0.083 0.000
−0.083 0.000 0.083	0.000 0.000 0.000	−0.083 0.083 0.000
−0.083 0.000 0.083	0.000 0.000 0.000	−0.083 0.083 0.000
0.000 0.000 0.000	−0.083 0.000 0.083	0.000 −0.083 0.083
0.000 0.000 0.000	−0.083 0.000 0.083	0.000 −0.083 0.083
0.000 0.000 0.000	−0.083 0.000 0.083	0.000 −0.083 0.083
−0.083 0.083 0.000	0.000 −0.083 0.083	0.000 0.000 0.000
−0.083 0.083 0.000	0.000 −0.083 0.083	0.000 0.000 0.000
−0.083 0.083 0.000	0.000 −0.083 0.083	0.000 0.000 0.000

4 Conclusion

In this article, after having acknowledged the purpose of the work and its context, the aim was to present the Variational Multiscale Method, to describe the chosen implementation, to demonstrate that it can be considered as a more general version of the SUPG method, and to present its similarities and differences with that method.

In the first part, the main objective of the work and its context were briefly recalled.

Then, in the second part, a theoretical explanation was provided for these two methods. Their respective assembly matrices were described and a comparison revealed their characteristics.

The third part was more focused on the results of the implementation. It was shown that the predictor/multi-corrector stages structure was properly implemented. It was also shown that the implementation of the algorithm itself is not yet correct although evolving in the right direction.

Eventually, when the VMS implementation will reflect perfectly the SUPG (for the non-viscous case), it will offer a robust foundation to explore the VMS capacities.

Acknowledgement: This study is promoted by the Royal Military Academy (RMA), in cotutelle with the Université de La Rochelle (ULR) and the von Karman Institute for Fluid Dynamics (VKI). The author is thankful to these three institutions to give him the opportunity to work on this subject.

Funding Statement: The authors received the funding of the Royal Higher Institute for Defence (MSP16-06).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Janssens, B. (2014). *Numerical modeling and experimental investigation of fine particle coagulation and dispersion in dilute flows (Ph.D. Thesis)*. Universite de La Rochelle.
2. Nishio, Y., Janssens, B., van Beeck, J., Limam, K. (2019). CBRN dispersion modeling in the atmosphere. *Review of the VKI Doctoral Research 2018–2019*, pp. 85–93. Von Karmann Institute for Fluid Dynamics, Belgium.
3. Banyai, T., van Abeele, D., Deconinck, H. (2006). A fast fully-coupled solution algorithm for the unsteady incompressible Navier stokes equations. *Conference on Modelling Fluid Flow (CMFF'06)*, pp. 91–98. Budapest, Hungary.
4. Braack, M., Burman, E., John, V., Lube, G. (2007). Stabilized finite element methods for the generalized oseen problem. *Computer Methods in Applied Mechanics and Engineering, 196(4–6),* 853–866. DOI 10.1016/j.cma.2006.07.011.

5. Nishio, Y., Janssens, B., Limam, K., van Beeck, J. (2018). Multiphase flow modeling for CBRN applications. *International Conference on Materials and Energy (ICOME'18)*, San Sebastian, Spain.
6. Hughes, T. J., Mazzei, L., Jansen, K. E. (2000). Large Eddy Simulation and the variational multiscale method. *Computing and Visualization in Science, 3(1–2),* 47–59. DOI 10.1007/s007910050051.
7. Bazilevs, Y., Calo, V., Cottrell, J., Hughes, T. J., Reali, A. et al. (2007). Variational multiscale residual-based turbulence modeling for large eddy simulation of incompressible flows. *Computer Methods in Applied Mechanics and Engineering, 197(1–4),* 173–201. DOI 10.1016/j.cma.2007.07.016.
8. Chung, J., Hulbert, G. M. (1993). A time integration algorithm for structural dynamics with improved numerical dissipation: The generalized-α method. *Journal of Applied Mechanics, 60(2),* 371–375. DOI 10.1115/1.2900803.
9. Jansen, K. E., Whiting, C. H., Hulbert, G. M. (2000). A generalizedα method for integrating the filtered Navier–stokes equations with a stabilized finite element method. *Computer Methods in Applied Mechanics and Engineering, 190(3),* 305–319. DOI 10.1016/S0045-7825(00)00203-6.
10. Johnson, C. (2012). Numerical solution of partial differential equations by the finite element method. In: *Dover books on mathematics series,* Courier Corporation.
11. Braack, M., Mucha, P. B. (2014). Directional do-nothing condition for the Navier-stokes equations. *Journal of Computational Mathematics, 32(5),* 507–521. DOI 10.4208/jcm.1405-m4347.