INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2003; **43**:555–575 (DOI: 10.1002/fld.505)

Computation of moving boundaries and interfaces and stabilization parameters

Tayfun E. Tezduyar*,†

Mechanical Engineering, Rice University—MS 321, 6100 Main Street, Houston, TX 77005, U.S.A.

SUMMARY

The interface-tracking and interface-capturing techniques we developed in recent years for computation of flow problems with moving boundaries and interfaces rely on stabilized formulations such as the streamline-upwind/Petrov–Galerkin (SUPG) and pressure-stabilizing/Petrov–Galerkin (PSPG) methods. The interface-tracking techniques are based on the deforming-spatial-domain/stabilized space–time formulation, where the mesh moves to track the interface. The interface-capturing techniques, typically used with non-moving meshes, are based on a stabilized semi-discrete formulation of the Navier–Stokes equations, combined with a stabilized formulation of the advection equation governing the time-evolution of an interface function marking the interface location. We provide an overview of the interface-tracking and interface-capturing techniques, and highlight how we determine the stabilization parameters used in the stabilized formulations. Copyright © 2003 John Wiley & Sons, Ltd.

KEY WORDS: moving boundaries and interfaces; SUPG and PSPG formulations; stabilization parameters

1. INTRODUCTION

The finite element techniques we have developed in recent years for computation of flow problems with moving boundaries and interfaces (such as free-surface and two-fluid interface flows, fluid–particle and fluid–structure interactions, and flows with moving mechanical components) can be categorized into interface-tracking and interface-capturing techniques (see References [1–3]). Depending on the complexity of the interface and other aspects of the problem, we can use one class of techniques or the other, or both in some cases, as it was pointed out in References [1–3]. An interface-tracking technique requires meshes that 'track' the interfaces. The mesh needs to be updated as the flow evolves. In interface-capturing techniques, such as one designed for two-fluid flows, the computations are based on spatial domains that are typically not moving or deforming. An interface function, marking the location of the

* Correspondence to: T. E. Tezduyar, Mechanical Engineering, Rice University—MS 321, 6100 Main Street, Houston, TX 77005, U.S.A.
† E-mail: tezduyar@rice.edu

Contract/grant sponsor: US Army Natick Soldier Center; contract/grant number: DAAD16-00-C-9222
Contract/grant sponsor: NASA Johnson Space Center; contract/grant number: NAG9-1059

Received 20 April 2002
Accepted 15 January 2003

interface, needs to be computed to 'capture' the interface over the non-moving mesh. The interface is captured within the resolution of the finite element mesh covering the area where the interface is. This approach can be seen as a special case of interface representation techniques, where the interface is somehow represented over a non-moving fluid mesh, the main point being that the fluid mesh does not move to 'track' the interfaces. A consequence of the mesh not moving to 'track' the interface is that for fluid–solid interfaces, independent of how well the interface geometry is represented, the resolution of the boundary layer will be limited by the resolution of the fluid mesh where the interface is.

The deforming-spatial-domain/stabilized space–time (DSD/SST) formulation [4], developed for moving boundaries and interfaces, is an interface-tracking technique, where the finite element formulation of the problem is written over its space–time domain. At each time step the locations of the interfaces are calculated as part of the overall solution. As the spatial domain occupied by the fluid changes its shape in time, mesh needs to be updated. In general, this is accomplished by moving the mesh with the motion of the nodes governed by the equations of elasticity, and full or partial remeshing (i.e., generating a new set of elements, and sometimes also a new set of nodes) as needed.

In computation of two fluid-flows with interface-tracking techniques, sometimes the interface might be too complex or unsteady to track while keeping the frequency of remeshing at an acceptable level. Not being able to reduce the frequency of remeshing in 3D might introduce overwhelming mesh generation and projection costs, making the computations with the interface-tracking technique no longer feasible. In such cases, interface-capturing techniques, which do not normally require costly mesh update steps, could be used with the understanding that the interface will not be represented as accurately as we would have with an interface-tracking technique. In other words, for comparable levels of spatial discretization, interface-capturing techniques yield less accurate representation of the interface. However, these techniques can be used as practical alternatives in carrying out the simulations when compromising the accurate representation of the interfaces becomes less of a concern than facing major difficulties in updating the mesh to track such interfaces. To increase the accuracy of an interface-capturing technique without adding a major computational cost, we developed the enhanced-discretization interface-capturing technique (EDICT), first introduced in Reference [5]. How EDICT can be used in several other contexts, such as shock-capturing in compressible flows or sub-time-stepping in fluid-structure interactions, is highlighted in References [1–3]. Methods developed to increase the scope and accuracy of the interface-tracking and interface-capturing techniques are also highlighted in References [1–3].

Our interface-tracking and interface-capturing techniques are based on the streamline-upwind/Petrov–Galerkin (SUPG) [6, 7], Galerkin/least-squares (GLS) [8], and pressure-stabilizing/Petrov–Galerkin (PSPG) [4] formulations. In the interface-capturing techniques, stabilized semi-discrete formulations are used for both the Navier–Stokes equations of incompressible flows and the advection equation governing the time-evolution of an interface function marking the interface location. These stabilization techniques prevent numerical oscillations and other instabilities in solving problems with advection-dominated flows and when using equal-order interpolation functions for velocity and pressure. Furthermore, this class of stabilized formulations substantially improve the convergence rate in iterative solution of the large, coupled nonlinear equation system that needs to be solved at every time step of a flow computation. Such nonlinear systems are typically solved with the Newton–Raphson method, which involves, at its every iteration step, solution of a large, coupled linear equation system.

In iterative solution of such linear equation systems using a good stabilized method makes substantial difference in convergence, as it was pointed out in Reference [9].

The SUPG, GLS and PSPG formulations stabilize the method without introducing excessive numerical dissipation. In these formulations, judicious selection of the stabilization parameter, which is almost always known as 'τ', plays an important role in determining the accuracy of the formulation. This stabilization parameter involves a measure of the local length scale (also known as 'element length') and other parameters such as the local Reynolds and Courant numbers. Various 'element length's and 'τ's were proposed starting with those in References [7, 10], followed by the one introduced in Reference [11], and those proposed in the subsequently reported SUPG, GLS and PSPG methods. A number of 'τ's, dependent upon spatial and temporal discretizations, were introduced and tested in Reference [12]. More recently, 'τ's which are applicable to higher-order elements were proposed in Reference [13].

Ways to calculate 'τ's from the element-level matrices and vectors were first introduced in Reference [14]. These new definitions are expressed in terms of the ratios of the norms of the relevant matrices or vectors. They automatically take into account the local length scales, advection field and the element-level Reynolds number. Based on these definitions, a 'τ' can be calculated for each element, or even for each element node or degree of freedom or element equation. Certain variations and complements of these new 'τ's were introduced in References [15, 3]. In this paper, we describe the element–matrix-based and element–vector-based 'τ's designed for the semi-discrete and space–time formulations of the advection–diffusion and Navier–Stokes equations. We also describe approximate versions of these 'τ's, which are based on the local length scales for the advection- and diffusion-dominated limits.

2. GOVERNING EQUATIONS

Let $\Omega_t \subset \mathbb{R}^{n_{sd}}$ be the spatial fluid mechanics domain with boundary Γ_t at time $t \in (0, T)$, where the subscript t indicates the time-dependence of the spatial domain. The Navier–Stokes equations of incompressible flows can be written on Ω_t and $\forall t \in (0, T)$ as

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u} - \mathbf{f}\right) - \nabla\cdot\boldsymbol{\sigma} = 0 \tag{1}$$

$$\nabla\cdot\mathbf{u} = 0 \tag{2}$$

where ρ, $\mathbf{u}$ and $\mathbf{f}$ are the density, velocity and the external force, respectively. The stress tensor $\boldsymbol{\sigma}$ is defined as

$$\boldsymbol{\sigma}(p, \mathbf{u}) = -p\mathbf{I} + 2\mu\boldsymbol{\varepsilon}(\mathbf{u}) \tag{3}$$

Here p is the pressure, $\mathbf{I}$ is the identity tensor, $\mu = \rho\nu$ is the viscosity, ν is the kinematic viscosity, and $\boldsymbol{\varepsilon}(\mathbf{u})$ is the strain-rate tensor:

$$\boldsymbol{\varepsilon}(\mathbf{u}) = \frac{1}{2}((\nabla\mathbf{u}) + (\nabla\mathbf{u})^{\mathrm{T}}) \tag{4}$$

The essential and natural boundary conditions for Equation (1) are represented as

$$\mathbf{u} = \mathbf{g} \text{ on } (\Gamma_t)_{\mathrm{g}}, \quad \mathbf{n}\cdot\boldsymbol{\sigma} = \mathbf{h} \text{ on } (\Gamma_t)_{\mathrm{h}} \tag{5}$$

where $(\Gamma_t)_g$ and $(\Gamma_t)_h$ are complementary subsets of the boundary Γ_t, $\mathbf{n}$ is the unit normal vector, and $\mathbf{g}$ and $\mathbf{h}$ are given functions. A divergence-free velocity field $\mathbf{u}_0(\mathbf{x})$ is specified as the initial condition.

If the problem does not involve any moving boundaries or interfaces, the spatial domain does not need to change with respect to time, and the subscript t can be dropped from Ω_t and Γ_t. This might be the case even for flows with moving boundaries and interfaces, if in the formulation used the spatial domain is not defined to be the part of the space occupied by the fluid(s). For example, we can have a fixed spatial domain, and model the fluid–fluid interfaces by assuming that the domain is occupied by two immiscible fluids, A and B, with densities ρ_A and ρ_B and viscosities μ_A and μ_B. In modelling a free-surface problem where fluid B is irrelevant, we assign a sufficiently low density to fluid B. An interface function ϕ serves as a marker identifying fluids A and B with the definition $\phi = \{1 \text{ for fluid A and } 0 \text{ for fluid B}\}$. The interface between the two fluids is approximated to be at $\phi = 0.5$. In this context, ρ and μ are defined as

$$\rho = \phi\rho_\mathrm{A} + (1-\phi)\rho_\mathrm{B}, \quad \mu = \phi\mu_\mathrm{A} + (1-\phi)\mu_\mathrm{B} \tag{6}$$

The evolution of the interface function ϕ, and therefore the motion of the interface, is governed by a time-dependent advection equation, written on Ω and $\forall t \in (0,T)$ as

$$\frac{\partial \phi}{\partial t} + \mathbf{u}\cdot\boldsymbol{\nabla}\phi = 0 \tag{7}$$

3. STABILIZED FORMULATIONS AND STABILIZATION PARAMETERS

3.1. Advection–Diffusion equation

Let us consider over a domain Ω with boundary Γ the following time-dependent advection–diffusion equation, written on Ω and $\forall t \in (0,T)$ as

$$\frac{\partial \phi}{\partial t} + \mathbf{u}\cdot\boldsymbol{\nabla}\phi - \boldsymbol{\nabla}\cdot(v\boldsymbol{\nabla}\phi) = 0 \tag{8}$$

where ϕ represents the quantity being transported (e.g. temperature, concentration, interface function), and v is the diffusivity. The essential and natural boundary conditions associated with Equation (8) are represented as

$$\phi = \mathrm{g} \text{ on } \Gamma_\mathrm{g}, \quad \mathbf{n}\cdot v\boldsymbol{\nabla}\phi = \mathrm{h} \text{ on } \Gamma_\mathrm{h} \tag{9}$$

A function $\phi_0(\mathbf{x})$ is specified as the initial condition.

Let us assume that we have constructed some suitably defined finite-dimensional trial solution and test function spaces $\mathscr{S}_\phi^h$ and $\mathscr{V}_\phi^h$. The stabilized finite element formulation of Equation (8) can then be written as follows: find $\phi^h \in \mathscr{S}_\phi^h$ such that $\forall w^h \in \mathscr{V}_\phi^h$:

$$\int_\Omega w^h\left(\frac{\partial \phi^h}{\partial t} + \mathbf{u}^h\cdot\boldsymbol{\nabla}\phi^h\right)\mathrm{d}\Omega + \int_\Omega \boldsymbol{\nabla}w^h\cdot v\boldsymbol{\nabla}\phi^h\,\mathrm{d}\Omega - \int_{\Gamma_\mathrm{h}} w^h\mathrm{h}^h\,\mathrm{d}\Gamma$$
$$+\sum_{e=1}^{n_\mathrm{el}}\int_{\Omega^e}\tau_\mathrm{SUPG}\mathbf{u}^h\cdot\boldsymbol{\nabla}w^h\left(\frac{\partial \phi^h}{\partial t} + \mathbf{u}^h\cdot\boldsymbol{\nabla}\phi^h - \boldsymbol{\nabla}\cdot(v\boldsymbol{\nabla}\phi^h)\right)\mathrm{d}\Omega = 0 \tag{10}$$

Here n_{el} is the number of elements, Ω^e is the domain for element e, and τ_{SUPG} is the SUPG stabilization parameter.

Let us use the notation $\mathbf{b}: \int_{\Omega^e}(\ldots)\,\mathrm{d}\Omega : \mathbf{b}_{\mathrm{V}}$ to denote the element-level matrix $\mathbf{b}$ and element-level vector $\mathbf{b}_{\mathrm{V}}$ corresponding to the element-level integration term $\int_{\Omega^e}(\ldots)\,\mathrm{d}\Omega$. We now define the following element-level matrices and vectors:

$$\mathbf{m}: \quad \int_{\Omega^e} w^h \frac{\partial \phi^h}{\partial t}\,\mathrm{d}\Omega \quad : \mathbf{m}_{\mathrm{V}} \tag{11}$$

$$\mathbf{c}: \quad \int_{\Omega^e} w^h \mathbf{u}^h \cdot \boldsymbol{\nabla}\phi^h\,\mathrm{d}\Omega \quad : \mathbf{c}_{\mathrm{V}} \tag{12}$$

$$\mathbf{k}: \quad \int_{\Omega^e} \boldsymbol{\nabla} w^h \cdot v \boldsymbol{\nabla}\phi^h\,\mathrm{d}\Omega \quad : \mathbf{k}_{\mathrm{V}} \tag{13}$$

$$\tilde{\mathbf{k}}: \quad \int_{\Omega^e} \mathbf{u}^h \cdot \boldsymbol{\nabla} w^h \mathbf{u}^h \cdot \boldsymbol{\nabla}\phi^h\,\mathrm{d}\Omega \quad : \tilde{\mathbf{k}}_{\mathrm{V}} \tag{14}$$

$$\tilde{\mathbf{c}}: \quad \int_{\Omega^e} \mathbf{u}^h \cdot \boldsymbol{\nabla} w^h \frac{\partial \phi^h}{\partial t}\,\mathrm{d}\Omega \quad : \tilde{\mathbf{c}}_{\mathrm{V}} \tag{15}$$

We define the element-level Reynolds and Courant numbers as follows:

$$Re = \frac{\|\mathbf{u}^h\|^2}{v} \frac{\|\mathbf{c}\|}{\|\tilde{\mathbf{k}}\|} \tag{16}$$

$$Cr_u = \frac{\Delta t}{2} \frac{\|\mathbf{c}\|}{\|\mathbf{m}\|} \tag{17}$$

$$Cr_v = \frac{\Delta t}{2} \frac{\|\mathbf{k}\|}{\|\mathbf{m}\|} \tag{18}$$

$$Cr_{\tilde{v}} = \frac{\Delta t}{2} \tau_{\mathrm{SUPG}} \frac{\|\tilde{\mathbf{k}}\|}{\|\mathbf{m}\|} \tag{19}$$

where $\|\mathbf{b}\|$ is the norm of matrix $\mathbf{b}$.

The components of element–matrix-based τ_{SUPG} are defined as follows:

$$\tau_{\mathrm{S1}} = \frac{\|\mathbf{c}\|}{\|\tilde{\mathbf{k}}\|} \tag{20}$$

$$\tau_{\mathrm{S2}} = \frac{\Delta t}{2} \frac{\|\mathbf{c}\|}{\|\tilde{\mathbf{c}}\|} \tag{21}$$

$$\tau_{\mathrm{S3}} = \tau_{\mathrm{S1}}\, Re = \left(\frac{\|\mathbf{c}\|}{\|\tilde{\mathbf{k}}\|}\right) Re \tag{22}$$

To construct τ_{SUPG} from its components we propose the form

$$\tau_{\text{SUPG}} = \left(\frac{1}{\tau_{\text{S1}}^r} + \frac{1}{\tau_{\text{S2}}^r} + \frac{1}{\tau_{\text{S3}}^r} \right)^{-1/r} \tag{23}$$

which is based on the inverse of τ_{SUPG} being defined as the r-norm of the vector with components $1/\tau_{\text{S1}}$, $1/\tau_{\text{S2}}$ and $1/\tau_{\text{S3}}$. We note that the higher the integer r is, the sharper the switching between τ_{S1}, τ_{S2} and τ_{S3} becomes.

The components of the element–vector-based τ_{SUPG} are defined as follows:

$$\tau_{\text{SV1}} = \frac{\|\mathbf{c}_{\text{V}}\|}{\|\tilde{\mathbf{k}}_{\text{V}}\|} \tag{24}$$

$$\tau_{\text{SV2}} = \frac{\|\mathbf{c}_{\text{V}}\|}{\|\tilde{\mathbf{c}}_{\text{V}}\|} \tag{25}$$

$$\tau_{\text{SV3}} = \tau_{\text{SV1}} Re = \left(\frac{\|\mathbf{c}_{\text{V}}\|}{\|\tilde{\mathbf{k}}_{\text{V}}\|} \right) Re \tag{26}$$

With these three components,

$$(\tau_{\text{SUPG}})_{\text{V}} = \left(\frac{1}{\tau_{\text{SV1}}^r} + \frac{1}{\tau_{\text{SV2}}^r} + \frac{1}{\tau_{\text{SV3}}^r} \right)^{-1/r} \tag{27}$$

Remark 1
The definition of τ_{SUPG} given by Equation (27) can be seen as a non-linear definition because it depends on the solution. However, in marching from time level n to $n+1$ the element vectors can be evaluated at level n. This might be preferable in some cases, as it spares us from ending up with a non-linear semi-discrete equation system.

3.2. Navier–Stokes Equations of Incompressible Flows

Given Equations (1)–(2), let us assume that we have some suitably defined finite-dimensional trial solution and test function spaces for velocity and pressure: $\mathcal{S}_{\mathbf{u}}^h$, $\mathcal{V}_{\mathbf{u}}^h$, $\mathcal{S}_p^h$ and $\mathcal{V}_p^h = \mathcal{S}_p^h$. The stabilized finite element formulation of Equations (1)–(2) can then be written as follows: find $\mathbf{u}^h \in \mathcal{S}_{\mathbf{u}}^h$ and $p^h \in \mathcal{S}_p^h$ such that $\forall \mathbf{w}^h \in \mathcal{V}_{\mathbf{u}}^h$ and $q^h \in \mathcal{V}_p^h$:

$$\begin{aligned}
&\int_\Omega \mathbf{w}^h \cdot \rho \left(\frac{\partial \mathbf{u}^h}{\partial t} + \mathbf{u}^h \cdot \boldsymbol{\nabla} \mathbf{u}^h - \mathbf{f} \right) \mathrm{d}\Omega + \int_\Omega \boldsymbol{\varepsilon}(\mathbf{w}^h) : \boldsymbol{\sigma}(p^h, \mathbf{u}^h) \, \mathrm{d}\Omega - \int_{\Gamma_\text{h}} \mathbf{w}^h \cdot \mathbf{h}^h \, \mathrm{d}\Gamma \\
&\quad + \int_\Omega q^h \boldsymbol{\nabla} \cdot \mathbf{u}^h \, \mathrm{d}\Omega + \sum_{e=1}^{n_{\text{el}}} \int_{\Omega^e} \frac{1}{\rho} [\tau_{\text{SUPG}} \rho \mathbf{u}^h \cdot \nabla \mathbf{w}^h + \tau_{\text{PSPG}} \nabla q^h] \cdot \\
&\qquad\qquad \left[\rho \left(\frac{\partial \mathbf{u}^h}{\partial t} + \mathbf{u}^h \cdot \boldsymbol{\nabla} \mathbf{u}^h \right) - \boldsymbol{\nabla} \cdot \boldsymbol{\sigma}(p^h, \mathbf{u}^h) - \rho \mathbf{f} \right] \mathrm{d}\Omega \\
&\quad + \sum_{e=1}^{n_{\text{el}}} \int_{\Omega^e} \tau_{\text{LSIC}} \boldsymbol{\nabla} \cdot \mathbf{w}^h \rho \boldsymbol{\nabla} \cdot \mathbf{u}^h \, \mathrm{d}\Omega = 0
\end{aligned} \tag{28}$$

Here τ_{PSPG} and τ_{LSIC} are the PSPG and LSIC (least-squares on incompressibility constraint) stabilization parameters.

We now define the following element-level matrices and vectors:

$$\mathbf{m}: \quad \int_{\Omega^e} \mathbf{w}^h \cdot \rho \frac{\partial \mathbf{u}^h}{\partial t} \, \mathrm{d}\Omega \quad : \mathbf{m}_{\mathrm{V}} \tag{29}$$

$$\mathbf{c}: \quad \int_{\Omega^e} \mathbf{w}^h \cdot \rho(\mathbf{u}^h \cdot \nabla \mathbf{u}^h) \, \mathrm{d}\Omega \quad : \mathbf{c}_{\mathrm{V}} \tag{30}$$

$$\mathbf{k}: \quad \int_{\Omega^e} \boldsymbol{\varepsilon}(\mathbf{w}^h) : 2\mu \boldsymbol{\varepsilon}(\mathbf{u}^h) \, \mathrm{d}\Omega \quad : \mathbf{k}_{\mathrm{V}} \tag{31}$$

$$\mathbf{g}: \quad \int_{\Omega^e} (\nabla \cdot \mathbf{w}^h) p^h \, \mathrm{d}\Omega \quad : \mathbf{g}_{\mathrm{V}} \tag{32}$$

$$\mathbf{g}^{\mathrm{T}}: \quad \int_{\Omega^e} q^h (\nabla \cdot \mathbf{u}^h) \, \mathrm{d}\Omega \quad : \mathbf{g}_{\mathrm{V}}^{\mathrm{T}} \tag{33}$$

$$\tilde{\mathbf{k}}: \quad \int_{\Omega^e} (\mathbf{u}^h \cdot \nabla \mathbf{w}^h) \cdot \rho(\mathbf{u}^h \cdot \nabla \mathbf{u}^h) \, \mathrm{d}\Omega \quad : \tilde{\mathbf{k}}_{\mathrm{V}} \tag{34}$$

$$\tilde{\mathbf{c}}: \quad \int_{\Omega^e} (\mathbf{u}^h \cdot \nabla \mathbf{w}^h) \cdot \rho \frac{\partial \mathbf{u}^h}{\partial t} \, \mathrm{d}\Omega \quad : \tilde{\mathbf{c}}_{\mathrm{V}} \tag{35}$$

$$\tilde{\boldsymbol{\gamma}}: \quad \int_{\Omega^e} (\mathbf{u}^h \cdot \nabla \mathbf{w}^h) \cdot \nabla p^h \, \mathrm{d}\Omega \quad : \tilde{\boldsymbol{\gamma}}_{\mathrm{V}} \tag{36}$$

$$\boldsymbol{\beta}: \quad \int_{\Omega^e} \nabla q^h \cdot \frac{\partial \mathbf{u}^h}{\partial t} \, \mathrm{d}\Omega \quad : \boldsymbol{\beta}_{\mathrm{V}} \tag{37}$$

$$\boldsymbol{\gamma}: \quad \int_{\Omega^e} \nabla q^h \cdot (\mathbf{u}^h \cdot \nabla \mathbf{u}^h) \, \mathrm{d}\Omega \quad : \boldsymbol{\gamma}_{\mathrm{V}} \tag{38}$$

$$\boldsymbol{\theta}: \quad \int_{\Omega^e} \nabla q^h \cdot \nabla p^h \, \mathrm{d}\Omega \quad : \boldsymbol{\theta}_{\mathrm{V}} \tag{39}$$

$$\mathbf{e}: \quad \int_{\Omega^e} (\nabla \cdot \mathbf{w}^h) \rho (\nabla \cdot \mathbf{u}^h) \, \mathrm{d}\Omega \quad : \mathbf{e}_{\mathrm{V}} \tag{40}$$

Remark 2

In the definition of the element-level matrices listed above, we assume that $\mathbf{u}^h$ appearing in the advective operator (i.e. in $\mathbf{u}^h \cdot \nabla \mathbf{u}^h$ and $\mathbf{u}^h \cdot \nabla \mathbf{w}^h$) is evaluated at time level n rather than $n+1$. The definition would essentially be the same if we, alternatively, assumed that it is evaluated at time level $n+1$ but non-linear iteration level i rather than $i+1$. Except, in the first option, in the advective operator we use $(\mathbf{u}^h)_n$, whereas in the second option we use $(\mathbf{u}^h)_{n+1}^i$. The second option can be seen as a non-linear definition. The first option might be preferable in some cases, as it spares us from another level of nonlinearity coming from the way τ is defined. In the definition of the element-level-vectors, we face the same choices in terms of the evaluation of $\mathbf{u}^h$ in the advective operator.

The element-level Reynolds and Courant numbers are defined the same way as they were defined before, as given by Equations (16)–(19). The components of the element–matrix-based τ_{SUPG} are defined the same way as they were defined before, as given by Equations (20)–(22). τ_{SUPG} is constructed from its components the same way as it was constructed before, as give by Equation (23). The components of the element–vector-based τ_{SUPG} are defined the same way as they were defined before, as given by Equations (24)–(26). The construction of $(\tau_{\mathrm{SUPG}})_{\mathrm{V}}$ is also the same as it was before, given by Equation (27).

The components of the element–matrix-based τ_{PSPG} are defined as follows:

$$\tau_{\mathrm{P1}} = \frac{\|\mathbf{g}^{\mathrm{T}}\|}{\|\boldsymbol{\gamma}\|} \tag{41}$$

$$\tau_{\mathrm{P2}} = \frac{\Delta t}{2}\frac{\|\mathbf{g}^{\mathrm{T}}\|}{\|\boldsymbol{\beta}\|} \tag{42}$$

$$\tau_{\mathrm{P3}} = \tau_{\mathrm{P1}}\, Re = \left(\frac{\|\mathbf{g}^{\mathrm{T}}\|}{\|\boldsymbol{\gamma}\|}\right) Re \tag{43}$$

τ_{PSPG} is constructed from its components as follows:

$$\tau_{\mathrm{PSPG}} = \left(\frac{1}{\tau_{\mathrm{P1}}^r} + \frac{1}{\tau_{\mathrm{P2}}^r} + \frac{1}{\tau_{\mathrm{P3}}^r}\right)^{-1/r} \tag{44}$$

The components of the element–vector-based τ_{PSPG} are defined as follows:

$$\tau_{\mathrm{PV1}} = \tau_{\mathrm{P1}} \tag{45}$$

$$\tau_{\mathrm{PV2}} = \tau_{\mathrm{PV1}}\frac{\|\boldsymbol{\gamma}_{\mathrm{V}}\|}{\|\boldsymbol{\beta}_{\mathrm{V}}\|} \tag{46}$$

$$\tau_{\mathrm{PV3}} = \tau_{\mathrm{PV1}}\, Re \tag{47}$$

With these components,

$$(\tau_{\mathrm{PSPG}})_{\mathrm{V}} = \left(\frac{1}{\tau_{\mathrm{PV1}}^r} + \frac{1}{\tau_{\mathrm{PV2}}^r} + \frac{1}{\tau_{\mathrm{PV3}}^r}\right)^{-1/r} \tag{48}$$

The element–matrix-based τ_{LSIC} is defined as follows:

$$\tau_{\mathrm{LSIC}} = \frac{\|\mathbf{c}\|}{\|\mathbf{e}\|} \tag{49}$$

We define the element–vector-based τ_{LSIC} as

$$(\tau_{\mathrm{LSIC}})_{\mathrm{V}} = \tau_{\mathrm{LSIC}} \tag{50}$$

Remark 3
We can also calculate a separate τ for each element node, or degree of freedom, or element equation. In that case, each component of τ would be calculated separately for each element node, or degree of freedom, or element equation. For this, we first represent an element matrix $\mathbf{b}$ in terms of its row matrices: $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_{n_{\mathrm{ex}}}$ and an element vector $\mathbf{b}_{\mathrm{V}}$ in terms of it subvectors: $(\mathbf{b}_{\mathrm{V}})_1, (\mathbf{b}_{\mathrm{V}})_2, \ldots, (\mathbf{b}_{\mathrm{V}})_{n_{\mathrm{ex}}}$. If we want a separate τ for each element node, then $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_{n_{\mathrm{ex}}}$ and $(\mathbf{b}_{\mathrm{V}})_1, (\mathbf{b}_{\mathrm{V}})_2, \ldots, (\mathbf{b}_{\mathrm{V}})_{n_{\mathrm{ex}}}$ would be the row matrices and subvectors corresponding to each element node, with $n_{\mathrm{ex}} = n_{\mathrm{en}}$, where n_{en} is the number of element nodes. If we want a separate τ for each degree of freedom, then $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_{n_{\mathrm{ex}}}$ and $(\mathbf{b}_{\mathrm{V}})_1, (\mathbf{b}_{\mathrm{V}})_2, \ldots, (\mathbf{b}_{\mathrm{V}})_{n_{\mathrm{ex}}}$ would be the row matrices and subvectors corresponding to each degree of freedom, with $n_{\mathrm{ex}} = n_{\mathrm{dof}}$, where n_{dof} is the number of degrees of freedom. If we want a separate τ for each element equation, then $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_{n_{\mathrm{ex}}}$ and $(\mathbf{b}_{\mathrm{V}})_1, (\mathbf{b}_{\mathrm{V}})_2, \ldots, (\mathbf{b}_{\mathrm{V}})_{n_{\mathrm{ex}}}$ would be the row matrices and subvectors corresponding to each element equation, with $n_{\mathrm{ex}} = n_{\mathrm{ee}}$, where n_{ee} is the number of element equations. Based on this, the components of τ would be calculated using the norms of these row matrices or subvectors, instead of the element matrices or vectors. For example, a separate τ_{S1} or τ_{SV1} for each element node would be calculated by using the expression

$$(\tau_{\mathrm{S1}})_a = \frac{\|\mathbf{c}_a\|}{\|\tilde{\mathbf{k}}_a\|}, \quad a = 1, 2, \ldots, n_{\mathrm{en}} \tag{51}$$

or

$$(\tau_{\mathrm{SV1}})_a = \frac{\|(\mathbf{c}_{\mathrm{V}})_a\|}{\|(\tilde{\mathbf{k}}_{\mathrm{V}})_a\|}, \quad a = 1, 2, \ldots, n_{\mathrm{en}} \tag{52}$$

Remark 4
The concept of calculating a separate τ for each element node or equation can be extended to calculating a separate τ for each global node or equation. This can be accomplished by first representing a global matrix or vector in terms of its row matrices or subvectors associated with the global nodes or equations, and then by calculating the components of τ using the norms of these global row matrices or subvectors. With this approach, applying the class of stabilization techniques described in this paper to element-free methods would become more direct.

For the purpose of comparison, we define here also the stabilization parameters that are based on an earlier definition of the length scale h [11]:

$$h_{\mathrm{UGN}} = 2\|\mathbf{u}^h\| \left(\sum_{a=1}^{n_{\mathrm{en}}} |\mathbf{u}^h \cdot \boldsymbol{\nabla} N_a| \right)^{-1} \tag{53}$$

where N_a is the interpolation function associated with node a. The stabilization parameters are defined as follows:

$$\tau_{\mathrm{SUGN1}} = \frac{h_{\mathrm{UGN}}}{2\|\mathbf{u}^h\|} \tag{54}$$

$$\tau_{\mathrm{SUGN2}} = \frac{\Delta t}{2} \tag{55}$$

$$\tau_{\text{SUGN3}} = \frac{h_{\text{UGN}}^2}{4\nu} \tag{56}$$

$$(\tau_{\text{SUPG}})_{\text{UGN}} = \left(\frac{1}{\tau_{\text{SUGN1}}^2} + \frac{1}{\tau_{\text{SUGN2}}^2} + \frac{1}{\tau_{\text{SUGN3}}^2} \right)^{-1/2} \tag{57}$$

$$(\tau_{\text{PSPG}})_{\text{UGN}} = (\tau_{\text{SUPG}})_{\text{UGN}} \tag{58}$$

$$(\tau_{\text{LSIC}})_{\text{UGN}} = \frac{h_{\text{UGN}}}{2} \|\mathbf{u}^h\| z \tag{59}$$

Here z is given as follows:

$$z = \begin{cases} \left(\frac{Re_{\text{UGN}}}{3}\right) & Re_{\text{UGN}} \leqslant 3 \\ 1 & Re_{\text{UGN}} > 3 \end{cases} \tag{60}$$

where $Re_{\text{UGN}} = \|\mathbf{u}^h\| h_{\text{UGN}}/2\nu$.

Comparisons between the performances of these earlier stabilization parameters and the ones proposed here can be found in Reference [14]. These comparisons show that, especially for special element geometries, the performances are similar.

Remark 5

The expression for τ_{SUGN1} can be written more directly as

$$\tau_{\text{SUGN1}} = \left(\sum_{a=1}^{n_{\text{en}}} |\mathbf{u}^h \cdot \boldsymbol{\nabla} N_a| \right)^{-1} \tag{61}$$

and based on that, the expression for h_{UGN} can be written as

$$h_{\text{UGN}} = 2\|\mathbf{u}^h\| \tau_{\text{SUGN1}} \tag{62}$$

A rationale for τ_{SUGN1} given by Equation (61) can be provided based on Remark 3 and Equation (52). For that, we apply Equation (52) to the advection–diffusion equation

$$(\tau_{\text{SV1}})_a = \left| \int_{\Omega^e} N_a (\mathbf{u}^h \cdot \boldsymbol{\nabla} \phi^h) \, \mathrm{d}\Omega \right| \Bigg/ \left| \int_{\Omega^e} (\mathbf{u}^h \cdot \boldsymbol{\nabla} N_a)(\mathbf{u}^h \cdot \boldsymbol{\nabla} \phi^h) \, \mathrm{d}\Omega \right| \tag{63}$$

Assuming one-point integration, we can write

$$(\tau_{\text{SV1}})_a = \frac{|N_a|}{|\mathbf{u}^h \cdot \boldsymbol{\nabla} N_a|} \tag{64}$$

Let us define τ_{SUGN1} to be the weighted average (weighted with $|\mathbf{u}^h \cdot \boldsymbol{\nabla} N_a|$) of these nodal τ_{SV1} values:

$$\tau_{\text{SUGN1}} = \left(\sum_{a=1}^{n_{\text{en}}} |\mathbf{u}^h \cdot \boldsymbol{\nabla} N_a| \right)^{-1} \sum_{a=1}^{n_{\text{en}}} |N_a| \tag{65}$$

For linear, bilinear and trilinear elements $|N_a| = N_a$ and therefore $\sum_{a=1}^{n_{\text{en}}} |N_a| = 1$. Consequently:

$$\tau_{\text{SUGN1}} = \left(\sum_{a=1}^{n_{\text{en}}} |\mathbf{u}^h \cdot \boldsymbol{\nabla} N_a| \right)^{-1} \tag{66}$$

As a potential alternative or complement to the LSIC stabilization, we propose the Discontinuity-Capturing Directional Dissipation (DCDD) stabilization. In describing the DCDD stabilization, we first define the unit vectors $\mathbf{s}$ and $\mathbf{r}$:

$$\mathbf{s} = \frac{\mathbf{u}^h}{\|\mathbf{u}^h\|}, \quad \mathbf{r} = \frac{\boldsymbol{\nabla}\|\mathbf{u}^h\|}{\|\boldsymbol{\nabla}\|\mathbf{u}^h\|\,\|} \tag{67}$$

and the element-level matrices and vectors $\mathbf{c}_{\text{r}}$, $\tilde{\mathbf{k}}_{\text{r}}$, $(\mathbf{c}_{\text{r}})_{\text{V}}$, and $(\tilde{\mathbf{k}}_{\text{r}})_{\text{V}}$:

$$\mathbf{c}_{\text{r}}: \quad \int_{\Omega^e} \mathbf{w}^h \cdot \rho(\mathbf{r} \cdot \boldsymbol{\nabla}\mathbf{u}^h)\, \text{d}\Omega \quad : (\mathbf{c}_{\text{r}})_{\text{V}} \tag{68}$$

$$\tilde{\mathbf{k}}_{\text{r}}: \int_{\Omega^e} (\mathbf{r} \cdot \boldsymbol{\nabla}\mathbf{w}^h) \cdot \rho(\mathbf{r} \cdot \boldsymbol{\nabla}\mathbf{u}^h)\, \text{d}\Omega \; : (\tilde{\mathbf{k}}_{\text{r}})_{\text{V}} \tag{69}$$

Then the DCDD stabilization is defined as

$$S_{\text{DCDD}} = \sum_{e=1}^{n_{\text{el}}} \int_{\Omega^e} \rho \nu_{\text{DCDD}} \boldsymbol{\nabla}\mathbf{w}^h : ([\mathbf{rr} - (\mathbf{r} \cdot \mathbf{s})^2 \mathbf{ss}] \cdot \boldsymbol{\nabla}\mathbf{u}^h)\, \text{d}\Omega \tag{70}$$

where the element–matrix-based and element–vector-based DCDD viscosities are

$$\nu_{\text{DCDD}} = |\mathbf{r} \cdot \mathbf{u}^h| \frac{\|\mathbf{c}_{\text{r}}\|}{\|\tilde{\mathbf{k}}_{\text{r}}\|} \tag{71}$$

$$(\nu_{\text{DCDD}})_{\text{V}} = |\mathbf{r} \cdot \mathbf{u}^h| \frac{\|(\mathbf{c}_{\text{r}})_{\text{V}}\|}{\|(\tilde{\mathbf{k}}_{\text{r}})_{\text{V}}\|} \tag{72}$$

An approximate version of the expression given by Equation (71) can be written as

$$\nu_{\text{DCDD}} = |\mathbf{r} \cdot \mathbf{u}^h| \frac{h_{\text{RGN}}}{2} \tag{73}$$

where

$$h_{\text{RGN}} = 2 \left(\sum_{a=1}^{n_{\text{en}}} |\mathbf{r} \cdot \boldsymbol{\nabla} N_a| \right)^{-1} \tag{74}$$

A different way of determining ν_{DCDD} can be expressed as

$$\nu_{\text{DCDD}} = \tau_{\text{DCDD}} \|\mathbf{u}^h\|^2 \tag{75}$$

where

$$\tau_{\mathrm{DCDD}} = \frac{h_{\mathrm{DCDD}}}{2\|\mathbf{U}\|} \frac{\|\boldsymbol{\nabla}\|\mathbf{u}^h\|\|h_{\mathrm{DCDD}}}{\|\mathbf{U}\|} \tag{76}$$

Here $\mathbf{U}$ represents a global velocity scale, and h_{DCDD} can be calculated by using the expression

$$h_{\mathrm{DCDD}} = 2\frac{\|\mathbf{c}_{\mathrm{r}}\|}{\|\tilde{\mathbf{k}}_{\mathrm{r}}\|} \tag{77}$$

or the approximation

$$h_{\mathrm{DCDD}} = h_{\mathrm{RGN}} \tag{78}$$

Combining Equations (75) and (76), we obtain

$$v_{\mathrm{DCDD}} = \frac{1}{2}\left(\frac{\|\mathbf{u}^h\|}{\|\mathbf{U}\|}\right)^2 (h_{\mathrm{DCDD}})^2 \|\boldsymbol{\nabla}\|\mathbf{u}^h\|\| \tag{79}$$

It was shown by Mittal [16] that flow computations with the SUPG and PSPG formulations (based on stabilization parameters very much like those given by Equations (53)–(60)) and very high aspect-ratio elements in the boundary layers might in some cases exhibit convergence problems and inaccuracies. As a remedy, Mittal [16] proposed to use an element length definition that represents the 'minimum dimension' of an element and gives the minimum edge length in the special case of a rectangular element.

In Reference [3], we proposed to re-define τ_{PSPG} by modifying the definitions of τ_{P3} and τ_{PV3} given by Equations (43) and (47). We proposed to accomplish that by using the expressions

$$\tau_{\mathrm{P3}} = \tau_{\mathrm{P1}} \frac{\|\mathbf{c}\|}{v\|\tilde{\mathbf{k}}_{\mathrm{r}}\|}, \quad \tau_{\mathrm{PV3}} = \tau_{\mathrm{PV1}} \frac{\|\mathbf{c}\|}{v\|\tilde{\mathbf{k}}_{\mathrm{r}}\|} \tag{80}$$

or the approximations

$$\tau_{\mathrm{P3}} = \tau_{\mathrm{P1}}\, Re\left(\frac{h_{\mathrm{RGN}}}{h_{\mathrm{UGN}}}\right)^2, \quad \tau_{\mathrm{PV3}} = \tau_{\mathrm{PV1}}\, Re\left(\frac{h_{\mathrm{RGN}}}{h_{\mathrm{UGN}}}\right)^2 \tag{81}$$

In Reference [3], we further stated that these modifications can also be applied to τ_{S3} and τ_{SV3} given by Equations (22) and (26). Here we write that explicitly

$$\tau_{\mathrm{S3}} = \tau_{\mathrm{S1}} \frac{\|\mathbf{c}\|}{v\|\tilde{\mathbf{k}}_{\mathrm{r}}\|}, \quad \tau_{\mathrm{SV3}} = \tau_{\mathrm{SV1}} \frac{\|\mathbf{c}\|}{v\|\tilde{\mathbf{k}}_{\mathrm{r}}\|} \tag{82}$$

$$\tau_{\mathrm{S3}} = \tau_{\mathrm{S1}}\, Re\left(\frac{h_{\mathrm{RGN}}}{h_{\mathrm{UGN}}}\right)^2, \quad \tau_{\mathrm{SV3}} = \tau_{\mathrm{SV1}}\, Re\left(\frac{h_{\mathrm{RGN}}}{h_{\mathrm{UGN}}}\right)^2 \tag{83}$$

However, if we are dealing with just an advection–diffusion equation, rather than the Navier–Stokes equations of incompressible flows, then the definition of the unit vector $\mathbf{r}$ changes as follows:

$$\mathbf{r} = \frac{\boldsymbol{\nabla}|\phi^h|}{\|\boldsymbol{\nabla}|\phi^h|\,\|} \tag{84}$$

We also propose to re-define τ_{SUGN3} given by Equation (56) as follows:

$$\tau_{\text{SUGN3}} = \frac{h_{\text{RGN}}^2}{4\nu} \tag{85}$$

Furthermore, we propose to replace $(\tau_{\text{LSIC}})_{\text{UGN}}$ given by Equation (59) as follows:

$$(\tau_{\text{LSIC}})_{\text{UGN}} = (\tau_{\text{SUPG}})_{\text{UGN}}\|\mathbf{u}^h\|^2 \tag{86}$$

Remark 6
The 'element length's h_{UGN} (given by Equation (53)) and h_{RGN} (Equation (74)) can be viewed as the local length scales corresponding to the advection- and diffusion-dominated limits, respectively.

4. DSD/SST FINITE ELEMENT FORMULATION

In the DSD/SST method, the finite element formulation of the governing equations is written over a sequence of N space–time slabs Q_n, where Q_n is the slice of the space–time domain between the time levels t_n and t_{n+1}. At each time step, the integrations involved in the finite element formulation are performed over Q_n. The space–time finite element interpolation functions are continuous within a space–time slab, but discontinuous from one space–time slab to another. Typically we use first-order polynomials as interpolation functions. The notation $(\cdot)_n^-$ and $(\cdot)_n^+$ denotes the function values at t_n as approached from below and above respectively. Each Q_n is decomposed into space–time elements Q_n^e, where $e = 1, 2, \ldots, (n_{\text{el}})_n$. The subscript n used with n_{el} is to account for the general case in which the number of space–time elements may change from one space–time slab to another. The Dirichlet- and Neumann-type boundary conditions are enforced over $(P_n)_{\text{g}}$ and $(P_n)_{\text{h}}$, the complementary subsets of the lateral boundary of the space–time slab. The finite element trial function spaces $(\mathscr{S}_{\mathbf{u}}^h)_n$ for velocity and $(\mathscr{S}_p^h)_n$ for pressure, and the test function spaces $(\mathscr{V}_{\mathbf{u}}^h)_n$ and $(\mathscr{V}_p^h)_n = (\mathscr{S}_p^h)_n$ are defined by using, over Q_n, first-order polynomials in both space and time.

The DSD/SST formulation is written as follows: given $(\mathbf{u}^h)_n^-$, find $\mathbf{u}^h \in (\mathscr{S}_{\mathbf{u}}^h)_n$ and $p^h \in (\mathscr{S}_p^h)_n$ such that $\forall \mathbf{w}^h \in (\mathscr{V}_{\mathbf{u}}^h)_n$ and $q^h \in (\mathscr{V}_p^h)_n$:

$$\int_{Q_n} \mathbf{w}^h \cdot \rho\left(\frac{\partial \mathbf{u}^h}{\partial t} + \mathbf{u}^h \cdot \boldsymbol{\nabla}\mathbf{u}^h - \mathbf{f}^h\right) \mathrm{d}Q + \int_{Q_n} \boldsymbol{\varepsilon}(\mathbf{w}^h) : \boldsymbol{\sigma}(p^h, \mathbf{u}^h)\,\mathrm{d}Q$$
$$- \int_{(P_n)_{\text{h}}} \mathbf{w}^h \cdot \mathbf{h}^h\,\mathrm{d}P + \int_{Q_n} q^h \boldsymbol{\nabla} \cdot \mathbf{u}^h\,\mathrm{d}Q + \int_{\Omega_n} (\mathbf{w}^h)_n^+ \cdot \rho((\mathbf{u}^h)_n^+ - (\mathbf{u}^h)_n^-)\,\mathrm{d}\Omega$$

$$+\sum_{e=1}^{(n_{\rm el})_n}\int_{Q_n^e}\frac{\tau_{\rm LSME}}{\rho}\,Ł(q^h,\mathbf{w}^h)\cdot[Ł(p^h,\mathbf{u}^h)-\rho\mathbf{f}^h]\,\mathrm{d}Q$$

$$+\sum_{e=1}^{n_{\rm el}}\int_{Q_n^e}\tau_{\rm LSIC}\,\boldsymbol{\nabla}\cdot\mathbf{w}^h\rho\boldsymbol{\nabla}\cdot\mathbf{u}^h\,\mathrm{d}Q=0 \tag{87}$$

where

$$Ł(q^h,\mathbf{w}^h)=\rho\left(\frac{\partial\mathbf{w}^h}{\partial t}+\mathbf{u}^h\cdot\boldsymbol{\nabla}\mathbf{w}^h\right)-\boldsymbol{\nabla}\cdot\boldsymbol{\sigma}(q^h,\mathbf{w}^h) \tag{88}$$

and $\tau_{\rm LSME}$ and $\tau_{\rm LSIC}$ are the stabilization parameters (see Reference [17]). This formulation is applied to all space–time slabs $Q_0, Q_1, Q_2, \ldots, Q_{N-1}$, starting with $(\mathbf{u}^h)_0^- = \mathbf{u}_0$. For an earlier, detailed reference on this stabilized formulation see Reference [4].

Remark 7 (*Space-time extension of the τ calculations described in Section 3*)
Let us write a DSD/SST formulation that is slightly different than the one given by Equation (87). We do that by neglecting the $(\tau_{\rm LSME}/\rho)\boldsymbol{\nabla}\cdot(2\mu\boldsymbol{\varepsilon}(\mathbf{w}^h))$ term and replacing $\tau_{\rm LSME}$ with $\tau_{\rm SUPG}$ and $\tau_{\rm PSPG}$:

$$\int_{Q_n}\mathbf{w}^h\cdot\rho\left(\frac{\partial\mathbf{u}^h}{\partial t}+\mathbf{u}^h\cdot\boldsymbol{\nabla}\mathbf{u}^h-\mathbf{f}^h\right)\mathrm{d}Q+\int_{Q_n}\boldsymbol{\varepsilon}(\mathbf{w}^h):\boldsymbol{\sigma}(p^h,\mathbf{u}^h)\,\mathrm{d}Q$$

$$-\int_{(P_n)_{\rm h}}\mathbf{w}^h\cdot\mathbf{h}^h\,\mathrm{d}P+\int_{Q_n}q^h\boldsymbol{\nabla}\cdot\mathbf{u}^h\,\mathrm{d}Q+\int_{\Omega_n}(\mathbf{w}^h)_n^+\cdot\rho((\mathbf{u}^h)_n^+-(\mathbf{u}^h)_n^-)\,\mathrm{d}\Omega$$

$$+\sum_{e=1}^{(n_{\rm el})_n}\int_{Q_n^e}\frac{1}{\rho}\left[\tau_{\rm SUPG}\rho\left(\frac{\partial\mathbf{w}^h}{\partial t}+\mathbf{u}^h\cdot\nabla\mathbf{w}^h\right)+\tau_{\rm PSPG}\nabla q^h\right]\cdot[Ł(p^h,\mathbf{u}^h)-\rho\mathbf{f}^h]\,\mathrm{d}Q$$

$$+\sum_{e=1}^{n_{\rm el}}\int_{Q_n^e}\tau_{\rm LSIC}\,\boldsymbol{\nabla}\cdot\mathbf{w}^h\rho\boldsymbol{\nabla}\cdot\mathbf{u}^h\,\mathrm{d}Q=0 \tag{89}$$

For extensions of the τ calculations based on matrix norms, we define the space–time augmented versions of the element-level matrices and vectors given by Equations (30), (34), and (38):

$$\mathbf{c}_{\rm A}: \quad \int_{Q_n^e}\mathbf{w}^h\cdot\rho\left(\frac{\partial\mathbf{u}^h}{\partial t}+\mathbf{u}^h\cdot\boldsymbol{\nabla}\mathbf{u}^h\right)\mathrm{d}Q \quad :(\mathbf{c}_{\rm A})_{\rm V} \tag{90}$$

$$\tilde{\mathbf{k}}_{\rm A}: \quad \int_{Q_n^e}\left(\frac{\partial\mathbf{w}^h}{\partial t}+\mathbf{u}^h\cdot\boldsymbol{\nabla}\mathbf{w}^h\right)\cdot\rho\left(\frac{\partial\mathbf{u}^h}{\partial t}+\mathbf{u}^h\cdot\boldsymbol{\nabla}\mathbf{u}^h\right)\mathrm{d}Q \quad :(\tilde{\mathbf{k}}_{\rm A})_{\rm V} \tag{91}$$

$$\boldsymbol{\gamma}_{\rm A}: \quad \int_{Q_n^e}\boldsymbol{\nabla}q^h\cdot\left(\frac{\partial\mathbf{u}^h}{\partial t}+\mathbf{u}^h\cdot\boldsymbol{\nabla}\mathbf{u}^h\right)\mathrm{d}Q \quad :(\boldsymbol{\gamma}_{\rm A})_{\rm V} \tag{92}$$

The components of element–matrix-based τ_{SUPG} are defined as follows:

$$\tau_{\mathrm{S12}} = \frac{\|\mathbf{c}_{\mathrm{A}}\|}{\|\tilde{\mathbf{k}}_{\mathrm{A}}\|} \tag{93}$$

$$\tau_{\mathrm{S3}} = \tau_{\mathrm{S12}} \frac{\|\mathbf{c}_{\mathrm{A}}\|}{v\|\tilde{\mathbf{k}}_{\mathrm{r}}\|} \tag{94}$$

where $\tilde{\mathbf{k}}_{\mathrm{r}}$ is the space–time version (i.e. integrated over the space–time element domain Q_n^e) of the element-level matrix given by Equation (69). To construct τ_{SUPG} from its components we propose the form

$$\tau_{\mathrm{SUPG}} = \left(\frac{1}{\tau_{\mathrm{S12}}^r} + \frac{1}{\tau_{\mathrm{S3}}^r} \right)^{-1/r} \tag{95}$$

The components of the element–vector-based τ_{SUPG} are defined as follows:

$$\tau_{\mathrm{SV12}} = \frac{\|(\mathbf{c}_{\mathrm{A}})_{\mathrm{V}}\|}{\|(\tilde{\mathbf{k}}_{\mathrm{A}})_{\mathrm{V}}\|} \tag{96}$$

$$\tau_{\mathrm{SV3}} = \tau_{\mathrm{SV12}} \frac{\|\mathbf{c}_{\mathrm{A}}\|}{v\|\tilde{\mathbf{k}}_{\mathrm{r}}\|} \tag{97}$$

From these two components,

$$(\tau_{\mathrm{SUPG}})_{\mathrm{V}} = \left(\frac{1}{\tau_{\mathrm{SV12}}^r} + \frac{1}{\tau_{\mathrm{SV3}}^r} \right)^{-1/r} \tag{98}$$

The components of element–matrix-based τ_{PSPG} are defined as follows:

$$\tau_{\mathrm{P12}} = \frac{\|\mathbf{g}^{\mathrm{T}}\|}{\|\boldsymbol{\gamma}_{\mathrm{A}}\|} \tag{99}$$

$$\tau_{\mathrm{P3}} = \tau_{\mathrm{P12}} \frac{\|\mathbf{c}_{\mathrm{A}}\|}{v\|\tilde{\mathbf{k}}_{\mathrm{r}}\|} \tag{100}$$

where $\mathbf{g}^{\mathrm{T}}$ is the space–time version of the element-level matrix given by Equation (33). To construct τ_{PSPG} from its components we propose the form

$$\tau_{\mathrm{PSPG}} = \left(\frac{1}{\tau_{\mathrm{P12}}^r} + \frac{1}{\tau_{\mathrm{P3}}^r} \right)^{-1/r} \tag{101}$$

The components of the element–vector-based τ_{PSPG} are defined as follows:

$$\tau_{\mathrm{PV12}} = \frac{\|\mathbf{g}_{\mathrm{V}}^{\mathrm{T}}\|}{\|(\boldsymbol{\gamma}_{\mathrm{A}})_{\mathrm{V}}\|} \tag{102}$$

$$\tau_{\mathrm{PV3}} = \tau_{\mathrm{PV12}} \frac{\|\mathbf{c}_{\mathrm{A}}\|}{v\|\tilde{\mathbf{k}}_{\mathrm{r}}\|} \tag{103}$$

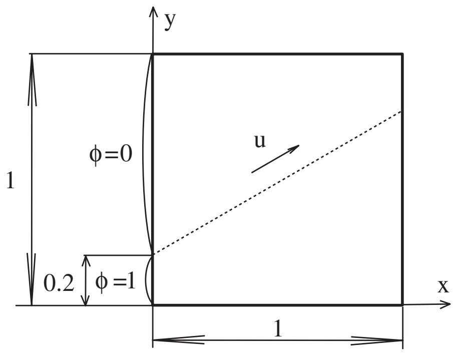


Figure 1. 2D Advection skew to mesh. Problem set up.

From these components,

$$(\tau_{\mathrm{PSPG}})_{\mathrm{V}} = \left(\frac{1}{\tau^r_{\mathrm{PV12}}} + \frac{1}{\tau^r_{\mathrm{PV3}}} \right)^{-1/r} \tag{104}$$

The element–matrix-based τ_{LSIC} is defined as

$$\tau_{\mathrm{LSIC}} = \frac{\|\mathbf{c}_{\mathrm{A}}\|}{\|\mathbf{e}\|} \tag{105}$$

where $\mathbf{e}$ is the space–time version of the element-level matrix given by Equation (40).

The element–vector-based τ_{LSIC} is defined as

$$(\tau_{\mathrm{LSIC}})_{\mathrm{V}} = \tau_{\mathrm{LSIC}} \tag{106}$$

The space–time versions of τ_{SUGN1}, τ_{SUGN2}, τ_{SUGN3}, $(\tau_{\mathrm{SUPG}})_{\mathrm{UGN}}$, $(\tau_{\mathrm{PSPG}})_{\mathrm{UGN}}$, and $(\tau_{\mathrm{LSIC}})_{\mathrm{UGN}}$, given respectively by Equations (54), (55), (85), (57), (58), and (86), are defined as follows:

$$\tau_{\mathrm{SUGN12}} = \left(\sum_{a=1}^{n_{\mathrm{en}}} \left| \frac{\partial N_a}{\partial t} + \mathbf{u}^h \cdot \boldsymbol{\nabla} N_a \right| \right)^{-1} \tag{107}$$

$$\tau_{\mathrm{SUGN3}} = \frac{h^2_{\mathrm{RGN}}}{4\nu} \tag{108}$$

$$(\tau_{\mathrm{SUPG}})_{\mathrm{UGN}} = \left(\frac{1}{\tau^2_{\mathrm{SUGN13}}} + \frac{1}{\tau^2_{\mathrm{SUGN3}}} \right)^{-1/2} \tag{109}$$

$$(\tau_{\mathrm{PSPG}})_{\mathrm{UGN}} = (\tau_{\mathrm{SUPG}})_{\mathrm{UGN}} \tag{110}$$

$$(\tau_{\mathrm{LSIC}})_{\mathrm{UGN}} = (\tau_{\mathrm{SUPG}})_{\mathrm{UGN}} \|\mathbf{u}^h\|^2 \tag{111}$$

Here, n_{en} is the number of nodes for the space–time element, and N_a is the space–time interpolation function associated with node a.

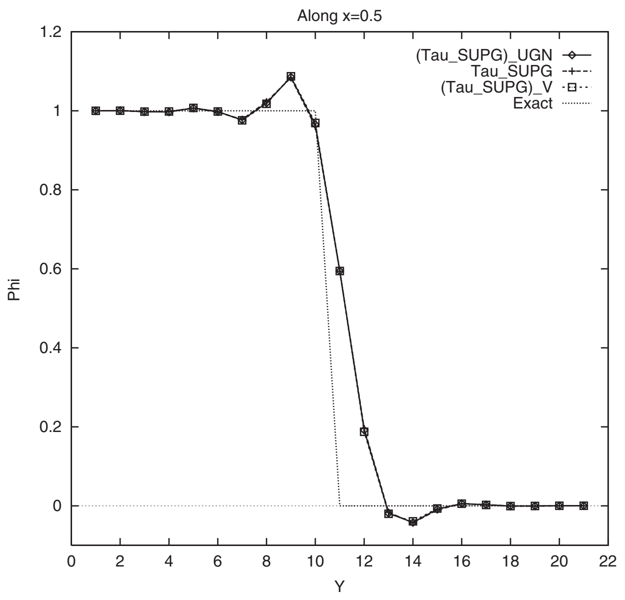


Figure 2. 2D advection skew to mesh. Solution along $x = 0.5$, obtained with three different stabilization parameters, compared to the exact solution.

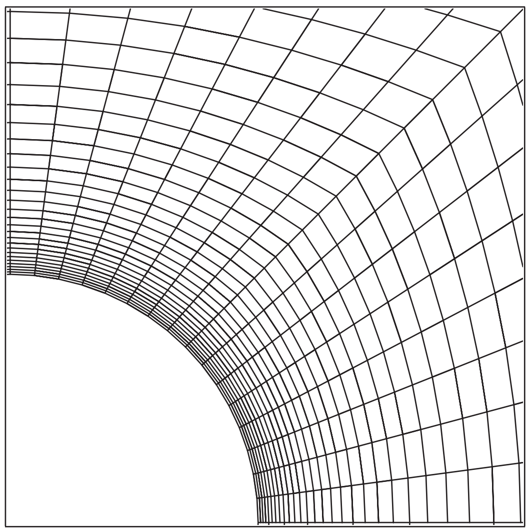
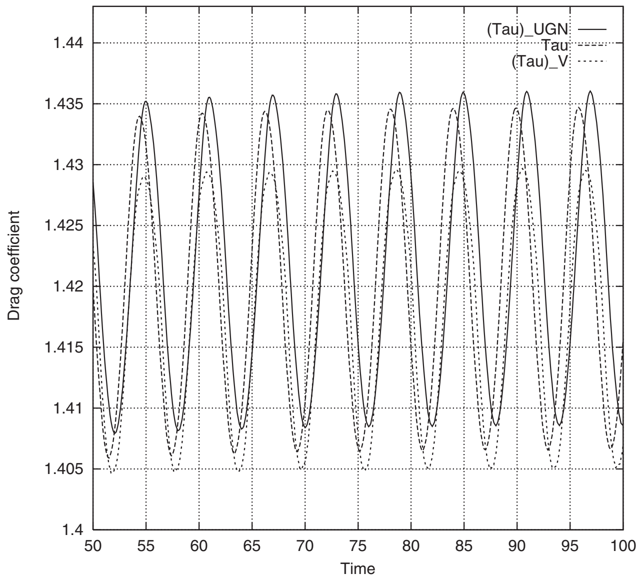


Figure 3. 2D Incompressible flow past a cylinder. Mesh near cylinder.



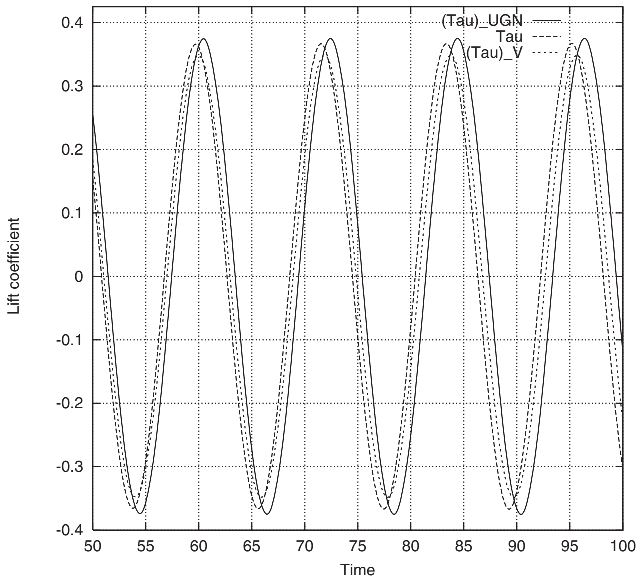
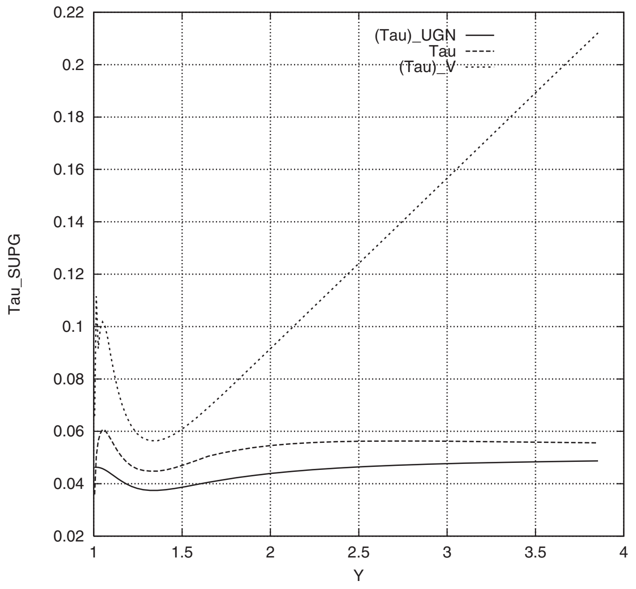


Figure 4. 2D incompressible flow past a cylinder. Time history of the drag (upper) and lift (lower) coefficients.

5. TEST COMPUTATIONS

5.1. 2D Advection skew to mesh

Here we compare the performance of some of the stabilization parameters for a 2D advection–diffusion problem with negligible diffusivity and with advection skew to the mesh.



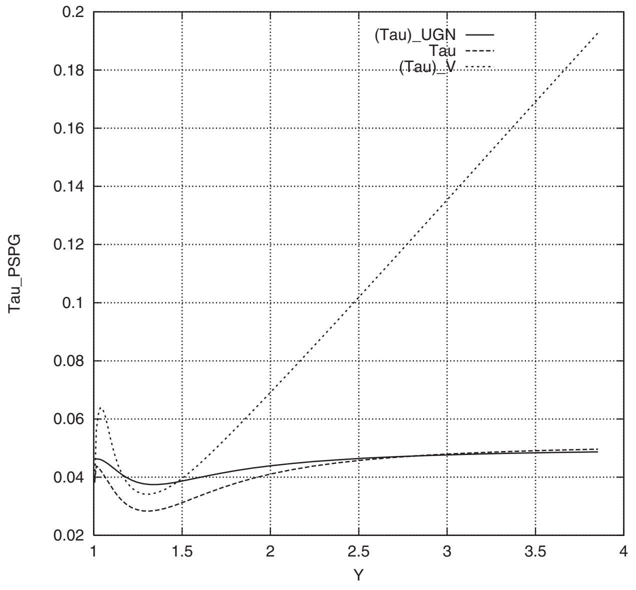


Figure 5. 2D incompressible flow past a cylinder. Stabilization parameters: τ_{SUPG} (upper) and τ_{PSPG} (lower).

Figure 1 shows the problem set up. The advection direction is 30° from the x-axis. The domain is square, the mesh is uniform with 20×20 square elements, and $\|\mathbf{u}^h\|\Delta t/\Delta x = 2.0$. The stabilization parameters tested are: $(\tau_{\mathrm{SUPG}})_{\mathrm{UGN}}$ (given by Equation (57)), τ_{SUPG} (Equation (23)), and $(\tau_{\mathrm{SUPG}})_{\mathrm{V}}$ (Equation (27)). Figure 2 shows the solution along $x = 0.5$. The exact solution is for the pure advection case. We observe that the solutions obtained with the stabilization parameters tested are almost identical. For more details on this test computation, see Reference [14].

5.2. 2D Incompressible flow past a cylinder

In this case we compare the stabilization parameters in computation of 2D incompressible flow past a cylinder at $Re = 100$. The mesh near the cylinder is shown in Figure 3. In this computation $U_\infty \Delta t / R = 0.1$, where U_∞ is the free-stream velocity and R is the cylinder radius. Figure 4 shows, at later stages of the computation, time history of the drag and lift coefficients. For $(\tau)_{\mathrm{UGN}}$, τ, and $(\tau)_{\mathrm{V}}$, respectively, the average drag coefficient is 1.422, 1.421, and 1.417, and the average Strouhal number is 0.167, 0.169, and 0.168. Here $(\tau)_{\mathrm{UGN}}$ represents the group of stabilization parameters $(\tau_{\mathrm{SUPG}})_{\mathrm{UGN}}$ (given by Equation (57)), $(\tau_{\mathrm{PSPG}})_{\mathrm{UGN}}$ (Equation (58)), and $(\tau_{\mathrm{LSIC}})_{\mathrm{UGN}}$ (Equation (59)); τ represents τ_{SUPG} (Equation (23)), τ_{PSPG} (Equation (44)), and τ_{LSIC} (Equation (49)); and $(\tau)_{\mathrm{V}}$ represents $(\tau_{\mathrm{SUPG}})_{\mathrm{V}}$ (Equation (27)), $(\tau_{\mathrm{PSPG}})_{\mathrm{V}}$ (Equation (48)), and $(\tau_{\mathrm{LSIC}})_{\mathrm{V}}$ (Equation (50)). Figure 5 shows the values of the stabilization parameters along the vertical line passing through the cylinder centre, starting from the upper cylinder surface. For more details on this test computation, see Reference [14].

6. CONCLUDING REMARKS

We provided an overview of the interface-tracking and interface-capturing techniques we developed for computation of flow problems with moving boundaries and interfaces. These techniques rely on stabilized formulations such as the streamline-upwind/Petrov–Galerkin and pressure-stabilizing/Petrov–Galerkin methods. The interface-tracking techniques are based on the deforming-spatial-domain/stabilized space–time formulation, where the mesh moves to track the interface. The interface-capturing techniques, typically used with non-moving meshes, are based on a stabilized semi-discrete formulation of the Navier–Stokes equations, combined with a stabilized formulation of an advection equation. The advection equation governs the time-evolution of an interface function marking the interface location. We highlighted how we determine the stabilization parameters ('τ's) used in the stabilized formulations. For the Navier–Stokes equations and the advection equation, we described the element–matrix-based and element–vector-based 'τ's designed for semi-discrete and space–time formulations. These 'τ' definitions are expressed in terms of the ratios of the norms of the relevant matrices or vectors. They automatically take into account the local length scales, advection field and the element-level Reynolds number. Based on these definitions, a 'τ' can be calculated for each element, or even for each element node or degree of freedom or element equation. We also described certain variations and complements of these new 'τ's, including the approximate versions that are based on the local length scales for the advection- and diffusion-dominated limits.

ACKNOWLEDGEMENTS

This work was supported by the US Army Natick Soldier Center (contract no. DAAD16-00-C-9222) and by NASA Johnson Space Center (grant no. NAG9-1059).

REFERENCES

1. Tezduyar TE. Finite element methods for flow problems with moving boundaries and interfaces. *Archives of Computational Methods in Engineering* 2001; **8**:83–130.
2. Tezduyar TE. Finite element interface-tracking and interface-capturing techniques for flows with moving boundaries and interfaces. In *Proceedings of the ASME Symposium on Fluid-Physics and Heat Transfer for Macro- and Micro-Scale Gas-Liquid and Phase-Change Flows* (*CD-ROM*), ASME Paper IMECE2001/HTD-24206. ASME: New York, 2001.
3. Tezduyar TE. Stabilized finite element formulations and interface-tracking and interface-capturing techniques for incompressible flows. In *Proceedings of the Workshop on Numerical Simulations of Incompressible Flows*, Half Moon Bay: California, 2001, to appear.
4. Tezduyar TE. Stabilized finite element formulations for incompressible flow computations. *Advances in Applied Mechanics* 1991; **28**:1–44.
5. Tezduyar TE, Aliabadi S, Behr M. Enhanced-discretization interface-capturing technique. In *Proceedings of the ISAC'97 High Performance Computing on Multiphase Flows*, Matsumoto Y, Prosperetti A (eds.), Nos. 1–6. Japan Society of Mechanical Engineers, 1997.
6. Hughes TJR, Brooks AN. A multi-dimensional upwind scheme with no crosswind diffusion. In *Finite Element Methods for Convection Dominated Flows*, Hughes TJR (ed.), AMD-Vol. 34. ASME: New York, 1979; 19–35.
7. Brooks AN, Hughes TJR. Streamline upwind/Petrov–Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier–Stokes equations. *Computer Methods in Applied Mechanics and Engineering* 1982; **32**:199–259.
8. Hughes TJR, Franca LP, Hulbert GM. A new finite element formulation for computational fluid dynamics: VIII. the Galerkin/least-squares method for advective-diffusive equations. *Computer Methods in Applied Mechanics and Engineering* 1989; **73**:173–189.
9. Tezduyar T, Aliabadi S, Behr M, Johnson A, Mittal S. Parallel finite-element computation of 3D flows. *IEEE Computer* 1993; **26**:27–36.
10. Tezduyar TE, Hughes TJR. Finite element formulations for convection dominated flows with particular emphasis on the compressible Euler equations. In *Proceedings of AIAA 21st Aerospace Sciences Meeting*, AIAA Paper 83-0125, Reno, Nevada, 1983.
11. Tezduyar TE, Park YJ. Discontinuity capturing finite element formulations for nonlinear convection–diffusion-reaction problems. *Computer Methods in Applied Mechanics and Engineering* 1986; **59**:307–325.
12. Tezduyar TE, Ganjoo DK. Petrov–Galerkin formulations with weighting functions dependent upon spatial and temporal discretization: Applications to transient convection–diffusion problems. *Computer Methods in Applied Mechanics and Engineering* 1986; **59**:49–71.
13. Franca LP, Frey SL, Hughes TJR. Stabilized finite element methods: I. Application to the advective-diffusive model. *Computer Methods in Applied Mechanics and Engineering* 1992; **95**:253–276.
14. Tezduyar TE, Osawa Y. Finite element stabilization parameters computed from element matrices and vectors. *Computer Methods in Applied Mechanics and Engineering* 2000; **190**:411–430.
15. Tezduyar TE, Adaptive determination of the finite element stabilization parameters. In *Proceedings of the ECCOMAS Computational Fluid Dynamics Conference 2001* (*CD-ROM*), Swansea: Wales, United Kingdom, 2001.
16. Mittal S. On the performance of high aspect-ratio elements for incompressible flows. *Computer Methods in Applied Mechanics and Engineering* 2000; **188**:269–287.
17. Tezduyar TE, Osawa Y. Methods for parallel computation of complex flow problems. *Parallel Computing* 1999; **25**:2039–2066.