 CMES, vol.96, no.1, pp.1-29, 2013

Detailed CVFEM Algorithm for Three Dimensional Advection-diffusion Problems

E. Tombarević[1], V. R. Voller[2] and I. Vušanović[1]

Abstract: The Control Volume Finite Element Method (CVFEM) combines the geometric flexibility of the Finite Element Method (FEM) with the physical intuition of the Control Volume Method (CVM). These two features of the CVFEM make it a very powerful tool for solving heat and fluid flow problems within complex domain geometries. In solving problems in the two-dimensional domains the development of the CVFEM has been well documented. For the three-dimensional problems, while there is extensive reporting on the details of the numerical approximation, there is relatively sparse information on important issues related to data structure and interpolation. Here, in the context of a general 3D advection-diffusion problem, a step-by-step derivation of the CVFEM is provided. Significant emphasis is placed on clearly defining an appropriate geometric data structure and detailing the key elements required to arrive at the final discrete equation. The performance and operation of the resulting 3D CVFEM scheme is highlighted by comparing its predictions against existing analytical solutions.

Keywords: CVFEM, three dimensions, unstructured mesh, advection-diffusion, phase-change.

1 Introduction

The advent of the digital computer, in the middle of the 20^{th} century, allows for the wide scale numerical solutions of problems in solid mechanics, heat and mass transfer, and fluid flow. The key component in these solutions is the splitting of the problem domain into a sum of smaller parts (a discretization), which leads to an approximation of the governing equations as an algebraic system. Very broadly speaking, approximations are divided into Finite Difference Methods (FDM) or Finite Element Methods (FEM). In the FDM, the domain is discretized with a rectilinear grid of nodes. On this grid the governing differential equations are ap-

[1] University of Montenegro, Faculty of Mechanical Engineering.
[2] University of Minnesota, Department of Civil Engineering.

proximated by appropriate Taylor series expansions. A drawback, in the basic approaches at least, is the geometric inflexibility arising from the restriction that grid lines of node points must be aligned with the coordinate directions. In contrast, the FEM is based on a mesh of geometric shapes (elements) that tessellate the domain. Typically the elements are polygons or polyhedrons with node points placed at vertices. Within each element of this mesh, functions can be approximated by appropriate interpolation of the element nodal values. Discrete algebraic equations then follow by using such interpolations to evaluate an integral (weak) statement of the governing equations. The significant advantage of FEM is the geometric flexibility to handle complex shaped geometries. In particular, elements can be placed in the domain without forcing node points to comply with a global structure. In other words, the FEM mesh of nodes can be unstructured as opposed to the rectilinear structured mesh required in the FDM.

An alternative to the FDM and FEM is offered by the so-called Control Volume Finite Difference Method (CVFDM) extensively described by Patankar (1980). In this approach, discrete algebraic equations are obtained through balancing transport fluxes across the faces of rectangular or cuboid control volumes. These volumes are constructed by placing faces between neighbouring nodes along a structured grid line. The fluxes across the faces are then approximated by a simple linear interpolation between the neighbouring nodes. This obvious connection with the physics of the problem is the main advantage of the CVFDM, but it still suffers from the geometric inflexibility of the FDM. This problem is overcome by using the Control Volume Finite Element Method (CVFEM), where polygonal or polyhedral control volumes are constructed by inserting faces between neighbouring nodes of an unstructured finite element mesh. In a similar manner to the CVFDM the discrete algebraic equations are also obtained by balancing the fluxes for the control volume. Here, however, fluxes across control volume faces are approximated using the finite element interpolations. In this way, the CVFEM can be seen as a hybrid numerical method that has all desirable features of both CVM and FEM: easy physical interpretation of the discretization procedure, solution which is conservative, even on coarse grids, and geometric flexibility to deal with complex domain geometries.

For completeness we point out that there are many alternative approaches that have similar abilities to the CVFEM. The most obvious alternative which, as noted above, can use an identical mesh structure is the FEM, Zienkiewicz and Taylor (2000). As elegantly pointed out by Idelsohn and Oñate (1994) although the CVFEM and FEM typically start from different points (a direct physical balance vs. a weighted residual) it is a relatively simple matter to demonstrate significant commonalities between the methods. There are also, however, alternative approaches

that use quite different meshing technologies. The classic example is boundary element methods (BEM), Wrobel and Aliabadi (2002). Here an appropriate mathematical treatment of the governing equations may only require a discretization of the domain boundary; thereby reducing the dimensional order of the problem. Finally, we note the rapid development and current interest in so called mesh-less methods, see for examples Sladek et al. (2013) and Šarler (2005). In these methods the full domain mesh of the CVFEM/FEM is essentially replaced by local and adaptive discretizations; an approach which has high utility in dealing with problems that contain sharp, transient discontinuities.

A comprehensive overview of CVFEMs and related methods for fluid flow and heat transfer can be found in Baliga (1997). The first efforts to combine CVM and FEM are presented in works by Winslow (1967) and Williamson (1969). Several variations of CVFEM are proposed in the doctoral dissertations of Baliga (1978), Ramadhyani (1979), and Prakash (1981). Two dimensional CVFEMs based on triangular elements are described in Baliga and Patankar (1980, 1983) and Voller (2009), while three dimensional methods based on tetrahedral elements are elaborated in LeDain-Muir and Baliga (1986) and Baliga and Atabaki (2006).

In any given CVFEM scheme there are two essential ingredients. The first is the methodology for calculating the fluxes across the CV faces. The second is the construction of an appropriate discretization and data structure that allows for the easy assembly of the discrete equations. Unsurprisingly, the major focus of works in the literature is on flux methodologies – e.g. the development of reliable advection diffusion schemes. In comparison, relatively sparse information is available for the data structure and assembly. In the case of two-dimensions such steps are relatively straightforward (see Voller (2009) for example). In three dimensions, however, this task is quite significant, requiring careful accounting and bookkeeping procedures.

The object of this paper is, in the context of advection-diffusion problems, to introduce a straightforward and robust three dimensional CVFEM data scheme that allows for a clear step by step procedure that explicitly links the treatment of the face fluxes to the generation of the final discrete equation, thereby empowering the reader with the essential knowledge to readily construct a working 3D CVFEM code. The operation of the code developed in this fashion is illustrated on solving a number of steady and transient test problems; including advection and diffusion transport and phase change.

2 The governing equation

The governing equation is the well-known transient advection diffusion equation for the conserved scalar ϕ:

$$\frac{\partial \phi}{\partial t} + \nabla \cdot (\mathbf{v}\phi) - \nabla \cdot (\kappa \nabla \phi) - Q = 0 \tag{1}$$

where $\mathbf{v}$ is velocity, ϕ is the conserved quantity, κ is the diffusivity and Q is the source term.

For the clarity of understanding, the 3D CVFEM procedure is first presented in terms of solving a simple steady state advection-diffusion equation. For this case, the integral form of the governing equation is:

$$\int_{\Gamma} (\mathbf{v} \cdot \mathbf{n}) \phi d\Gamma - \int_{\Gamma} \kappa \nabla \phi \cdot \mathbf{n} d\Gamma = 0 \tag{2}$$

where Γ is the surface area of an arbitrary shaped 3-D domain.

3 Domain discretization and data structure

The first step in converting the advection-diffusion governing equation into a system of algebraic equations is to place the node points into the domain. A convenient and flexible approach is to tessellate the domain with a mesh of triangular (2D) or tetrahedral (3D) elements with the node points placed at the vertices. Such meshes can be constructed by using available freeware and commercial software packages. Examples of open source software include *Gmsh* [Geuzaine and Remacle (2009)] which provides a finite element mesh generator with a built-in CAD engine and post-processor and *Distmesh* [Persson (2004)], a collection of *Matlab* functions for generation and manipulation of unstructured 2D and 3D meshes.

A generic four-node tetrahedron is shown in Fig. 1. Its geometry is fully defined by the position of four, non-coplanar, vertex nodes.

A particular 3D tetrahedral mesh will contain *ntet* elements associated with node points and *nnodes* unique node points. A mesh generator will store this information in two matrices:

- A point matrix **p** (dimensioned *nnodes*-by-3), in which each row lists the x, y and z coordinates of each point in the mesh;
- A tetrahedron matrix **t** (dimensioned *ntet*-by-4) in which each row lists the global node numbers of the vertices of the tetrahedral elements.

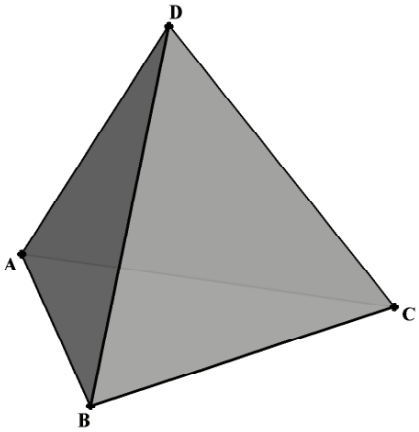

Figure 1: Tetrahedral element

It is important to note, that the first three entries in a row of matrix **t** are arranged such that they form an anti-clockwise path when viewed from the remaining fourth vertex. Hence, for our given element in Fig. 1, possible valid entries in the **t** matrix include: *BDCA*, *ACDB*, *ADBC* and *ABCD*. Such a notation convention eliminates ambiguities about the sign of the dot, cross and triple products of vectors that are extensively used in CVFEM. From this point on we will assume that the **t** row entry for our generic tetrahedron in Fig. 1 is *ABCD* and that the position vectors of the node points (vertices) from a fixed origin point are labelled **a**, **b**, **c** and **d**. From this the volume of the tetrahedral element can be calculated as:

$$V_{ABCD} = \frac{1}{6}(\mathbf{d}-\mathbf{a}) \cdot [(\mathbf{b}-\mathbf{a}) \times (\mathbf{c}-\mathbf{a})] \tag{3}$$

In addition to the matrices **t** and **p** a mesh generator will also store information about the domain boundary. By its nature a mesh of tetrahedral elements will form a domain boundary as a piecewise collection of planar triangles. This information is stored in a matrix **s** dimensioned *ns*-by-4, where *ns* is the number of boundary triangles. The first three entries in a row of **s** contain the global nodal numbers of vertices of a given boundary triangle, arranged anti-clockwise as viewed from the inside of the domain. The fourth entry in a row denotes the boundary region – a contiguous section identified by a common boundary condition, e.g. an imposed value or an imposed flux. On denoting the vertex nodes as A, B and C with position vectors **a**, **b**, and **c** the area of this triangle can be calculated as:

$$\Gamma_{ABC} = \frac{1}{2} \cdot |(\mathbf{b}-\mathbf{a}) \times (\mathbf{c}-\mathbf{a})| \tag{4}$$

4 The interpolation shape function

Values of the dependent variable ϕ are stored at the nodes located on the vertices of tetrahedral elements. The dependent value at an arbitrary point O (x_O, y_O, z_O) within the element can then be expressed as a linear combination of the values at the nodes A, B, C and D:

$$\phi(x_O, y_O, z_O) = N_A\phi_A + N_B\phi_B + N_C\phi_C + N_D\phi_D \tag{5}$$

With linear tetrahedral elements a straightforward geometric derivation for the volume shape functions (the N's in eq. (5)) can be used. On noting that, with appropriate construction about an arbitrary point O (position vector $\mathbf{o}$), a given tetrahedral element can be divided into four sub tetrahedrons (Fig. 2), the volume shape functions can be defined by:

$$\begin{aligned}
N_A &= \frac{V_{BDCO}}{V_{ABCD}} = \frac{1}{6}\frac{\vec{BO}\cdot(\vec{BD}\times\vec{BC})}{V_{ABCD}} = \frac{1}{6}\frac{(\mathbf{o}-\mathbf{b})\cdot[(\mathbf{d}-\mathbf{b})\times(\mathbf{c}-\mathbf{b})]}{V_{ABCD}} \\
N_B &= \frac{V_{ACDO}}{V_{ABCD}} = \frac{1}{6}\frac{\vec{AO}\cdot(\vec{AC}\times\vec{AD})}{V_{ABCD}} = \frac{1}{6}\frac{(\mathbf{o}-\mathbf{a})\cdot[(\mathbf{c}-\mathbf{a})\times(\mathbf{d}-\mathbf{a})]}{V_{ABCD}} \\
N_C &= \frac{V_{ADBO}}{V_{ABCD}} = \frac{1}{6}\frac{\vec{AO}\cdot(\vec{AD}\times\vec{AB})}{V_{ABCD}} = \frac{1}{6}\frac{(\mathbf{o}-\mathbf{a})\cdot[(\mathbf{d}-\mathbf{a})\times(\mathbf{b}-\mathbf{a})]}{V_{ABCD}} \\
N_D &= \frac{V_{ABCO}}{V_{ABCD}} = \frac{1}{6}\frac{\vec{AO}\cdot(\vec{AB}\times\vec{AC})}{V_{ABCD}} = \frac{1}{6}\frac{(\mathbf{o}-\mathbf{a})\cdot[(\mathbf{b}-\mathbf{a})\times(\mathbf{c}-\mathbf{a})]}{V_{ABCD}}
\end{aligned} \tag{6}$$

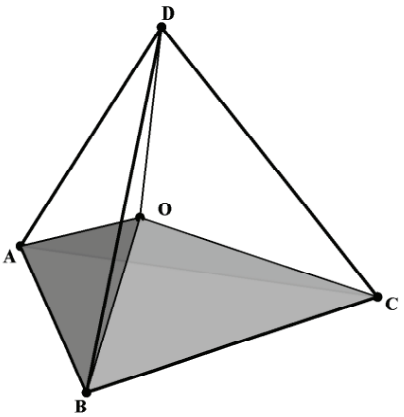

Figure 2: Geometric interpretation of the volume shape functions

These shape functions have some important properties:

$$N_i(x,y,z) = \begin{cases} 1, \text{ at node } i \\ 0, \text{ at all points on face oposite node } i \end{cases}, \quad i = A,\ B,\ C,\ D \tag{7}$$

i.e., when point O coincides with a node $i = A$, B, C or D the shape function$N_i = 1$, and when point O is anywhere on element face opposite node i, the shape function $N_i = 0$.

Due to the fact that the sub volumes about point O tessellate the element, the sum of the shape functions is one:

$$\sum_{i=A,B,C,D} N_i(x,y,z) = 1 \tag{8}$$

The gradient of the shape functions, constant over the element and necessary for the subsequent discretization, can be readily calculated from:

$$\begin{aligned}
\nabla N_A &= \frac{1}{6}\frac{\vec{BD}\times\vec{BC}}{V_{ABCD}} = \frac{1}{6}\frac{(\mathbf{d}-\mathbf{b})\times(\mathbf{c}-\mathbf{b})}{V_{ABCD}} \\
\nabla N_B &= \frac{1}{6}\frac{\vec{AC}\times\vec{AD}}{V_{ABCD}} = \frac{1}{6}\frac{(\mathbf{c}-\mathbf{a})\times(\mathbf{d}-\mathbf{a})}{V_{ABCD}} \\
\nabla N_C &= \frac{1}{6}\frac{\vec{AD}\times\vec{AB}}{V_{ABCD}} = \frac{1}{6}\frac{(\mathbf{d}-\mathbf{a})\times(\mathbf{b}-\mathbf{a})}{V_{ABCD}} \\
\nabla N_D &= \frac{1}{6}\frac{\vec{AB}\times\vec{AC}}{V_{ABCD}} = \frac{1}{6}\frac{(\mathbf{b}-\mathbf{a})\times(\mathbf{c}-\mathbf{a})}{V_{ABCD}}
\end{aligned} \tag{9}$$

In this way, it can be shown that the dot product of $\nabla\phi$ with vector $\mathbf{n}$ can be approximated in the element as:

$$\nabla\phi\cdot\mathbf{n} \approx (\nabla N_A\cdot\mathbf{n})\phi_A + (\nabla N_B\cdot\mathbf{n})\phi_B + (\nabla N_C\cdot\mathbf{n})\phi_C + (\nabla N_D\cdot\mathbf{n})\phi_D \tag{10}$$

5 Construction of the control volumes

Within the generic element the face that separates node A from B can be defined by the four-sided planar surface with corners at the midpoint of tetrahedron $ABCD$, face ABC, face ABD and edge AB.

The unit normal on this face, pointing from A to B, is:

$$(\Gamma\mathbf{n})_{f1} = \tfrac{1}{24}\mathbf{a}\times\mathbf{c} - \tfrac{1}{24}\mathbf{a}\times\mathbf{d} + \tfrac{1}{24}\mathbf{b}\times\mathbf{c} - \tfrac{1}{24}\mathbf{b}\times\mathbf{d} + \tfrac{1}{12}\mathbf{c}\times\mathbf{d} \tag{11a}$$

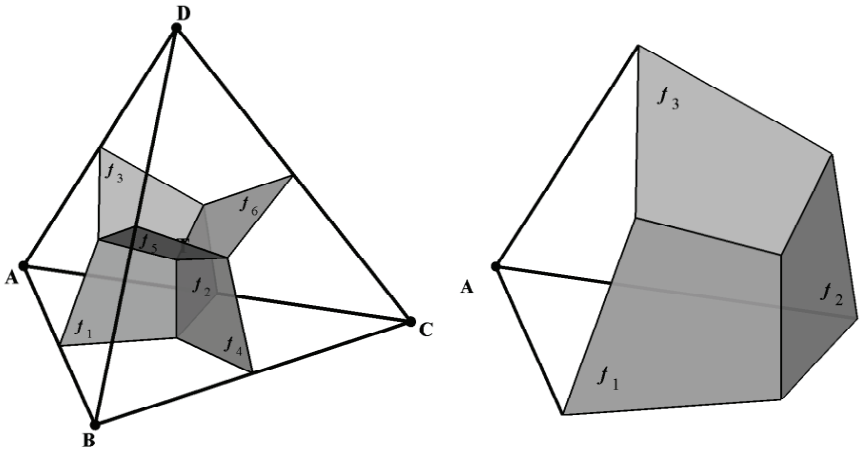

Figure 3: Construction of the control volumes – CV faces within one tetrahedron

where Γ_{f_1} is the face area. Five similar faces, separating each pair of nodes (*AC*, *AD*, *BC*, *BD* and *CD*) in the element can be constructed. The normals on these faces are as follows:

pointing from *A* to *C*

$$(\Gamma\mathbf{n})_{f_2} = -\tfrac{1}{24}\mathbf{a}\times\mathbf{b} + \tfrac{1}{24}\mathbf{a}\times\mathbf{d} + \tfrac{1}{24}\mathbf{b}\times\mathbf{c} - \tfrac{1}{12}\mathbf{b}\times\mathbf{d} + \tfrac{1}{24}\mathbf{c}\times\mathbf{d} \tag{11b}$$

pointing from *A* to *D*

$$(\Gamma\mathbf{n})_{f_3} = \tfrac{1}{24}\mathbf{a}\times\mathbf{b} - \tfrac{1}{24}\mathbf{a}\times\mathbf{c} + \tfrac{1}{12}\mathbf{b}\times\mathbf{c} - \tfrac{1}{24}\mathbf{b}\times\mathbf{d} + \tfrac{1}{24}\mathbf{c}\times\mathbf{d} \tag{11c}$$

pointing from *B* to *C*

$$(\Gamma\mathbf{n})_{f_4} = -\tfrac{1}{24}\mathbf{a}\times\mathbf{b} - \tfrac{1}{24}\mathbf{a}\times\mathbf{c} + \tfrac{1}{12}\mathbf{a}\times\mathbf{d} - \tfrac{1}{24}\mathbf{b}\times\mathbf{d} - \tfrac{1}{24}\mathbf{c}\times\mathbf{d} \tag{11d}$$

pointing from *B* to *D*

$$(\Gamma\mathbf{n})_{f_5} = \tfrac{1}{24}\mathbf{a}\times\mathbf{b} - \tfrac{1}{12}\mathbf{a}\times\mathbf{c} + \tfrac{1}{24}\mathbf{a}\times\mathbf{d} + \tfrac{1}{24}\mathbf{b}\times\mathbf{c} - \tfrac{1}{24}\mathbf{c}\times\mathbf{d} \tag{11e}$$

pointing from *C* to *D*

$$(\Gamma\mathbf{n})_{f_6} = \tfrac{1}{12}\mathbf{a}\times\mathbf{b} - \tfrac{1}{24}\mathbf{a}\times\mathbf{c} - \tfrac{1}{24}\mathbf{a}\times\mathbf{d} + \tfrac{1}{24}\mathbf{b}\times\mathbf{c} + \tfrac{1}{24}\mathbf{b}\times\mathbf{d} \tag{11f}$$

With these constructs it is recognised that the first three faces (11a to 11c) completely separate node *A* from its neighbouring nodes (*B*, *C* and *D*) in the element

(Fig. 3). Furthermore, when all the tetrahedral elements in the grid that have vertices at A are considered, the separating faces associated with node (vertex) A will complexly enclose the point. This enclosing control volume is the core of the CVFEM, the flux balance across its faces leading to an appropriate discrete equation.

Two properties of the separating faces and resulting control volumes need to be noted. (1) The adopted direction of the normal vectors should be respected in the derivation of the discrete equation. For example, face f_1 is the mutual face for the CV around node A and the CV around node B. The outward pointing normal on this face for the CV around node A is$(\Gamma\mathbf{n})_{f_1}$, while for the CV around node B it is $-(\Gamma\mathbf{n})_{f_1}$. (2) By the construction made it is easily confirmed that the each element that has a vertex at node A contributes $1/4$ of its volume to the CV enclosing node A.

6 Auxiliary data structure

As previously noted, the nodes in the domain are uniquely numbered between 1 and *nnodes*. The CV around a given node i involves a number of the tetrahedral elements in the mesh. The nodes in these elements form a set of neighbour (support) nodes for node i, consisting of all the nodes that share a common edge with node i. These nodes are important in the CVFEM derivation because the algebraic governing equation approximation, associated with node i, is developed in terms of the unknown field values at these nodes. As such, it is helpful to introduce an auxiliary data structure that facilitates the derivation of the CVFEM equation for a given node. The auxiliary data structure used here involves two additional matrices.

The first matrix in the auxiliary data structure is labelled **g** (for global) and dimensioned *nnodes*-by-*max-nnb*. The i-th row in **g** lists the global numbers of the neighbours of node i. Not all nodes in the mesh will have the same number of neighbours; the maximum number of neighbours in a given mesh is labelled *max-nnb*. At node i where the number of neighbours nnb_i $<$*max-nnb*, the corresponding row in **g** lists the neighbour node numbers in the first nnb_i locations and pads the remaining spaces with 0's. To illustrate, assume a purely hypothetical mesh with 50 nodes (*nnodes* = 50) where neighbour node counts do not exceed *max-nnb* = 10. Then if the i = 15, node in the mesh has nnb_{15} = 8 neighbours, the 15^{th} row in **g** could look like:

$$[16,\ 17,\ 11,\ 4,\ 25,\ 37,\ 18,\ 9,\ 0,\ 0] \tag{12}$$

Note the ordering of the non-zero node numbers in the row is not important. A method for the construction of **g** from the mesh matrices **t** and **p** is provided in an appendix.

Table 1: Elements of the data structure

Index	**Definition**	
nnodes	number of nodes in the mesh	
ntet	number of tetrahedral elements in the mesh	
ns	number of boundary triangles	
max-nnb	number of neighbours of a node with the most neighbours in the mesh	
Matrix	**Dimension**	**Definition**
p	*nnodes*-by-3	point matrix - the i-th row lists the x, y and z coordinates of the associated node
t	*ntet*-by-4	tetrahedron matrix - the i-th row lists the global node numbers of the vertices of the associated tetrahedral element
nnb	*nnodes*-by-1	number of neighbours vector - the i-th element is the number of neighbours of the associated node
g	*nnodes*-by-*max-nnb*	global data structure matrix - the i-th row lists global numbers of the neighbours of the associated node
l	*nnodes*-by-*nnodes*	local data structure matrix - the i-th row lists unique numbers between 1 and nnb_i for neighbours around associated node (matrix is sparse)
s	*ns*-by-4	boundary matrix - the first three entries in the i-th row are the global node numbers of vertices of the associated boundary triangle; the fourth entry denotes boundary region
ap	*nnodes*-by-1	vector that stores the coefficients of the discrete equation directly associated with the nodes in the domain
anb	*nnodes*-by-*max-nnb*	matrix that stores in each row the coefficients of the discrete equation associated with the neighbours of the respective node
bp	*nnodes*-by-1	vector that stores additional terms in the discrete equation arising from the treatment of sources, transients and boundary conditions
df	6-by-4	matrix of diffusive fluxes
vol	*nnodes*-by-1	CV volume vector, the i-th element is the volume of CV of the associated node

The second matrix in the auxiliary data structure is labelled **l** (for local) and dimensioned *nnodes*-by-*nnodes*. If the node with global number j is a neighbour of node i, the j-th entry on the i-th row of **l** will contain a unique number between 1 and nnb_i . If the node j is not a neighbour of i then the entry is zero. Then, by (12), on the hypothetical example mesh, nonzero elements in the i-th row in matrix **l** will

be:

$$\begin{aligned} &l_{15,16} = 1; \quad l_{15,17} = 2; \quad l_{15,11} = 3; \quad l_{15,4} = 4; \\ &l_{15,25} = 5; \quad l_{15,37} = 6; \quad l_{15,18} = 7; \quad l_{15,9} = 8; \end{aligned} \tag{13}$$

Since the number of nodes in a support nnb_i will be much less than the total number of nodes in the domain *nnodes*, it is to be expected that **l** is a sparse matrix. This matrix is readily obtained from the matrix **g** through the following construction.

$$l_{i,g_{i,j}} = j, \text{ where } i = 1,...,nnodes,\ j = 1,...nnb_i \tag{14}$$

A summary of the indices and matrices in the data structure are provided in Table 1.

7 The control volume point equation

The aim of this paper is to detail the CVFEM discretization procedure that transforms the continuous equation (1) into a system of algebraic equations in terms of the unknown values ϕ located at the node points of a linear tetrahedral finite element mesh. At a specific node point in the mesh the general form of the associated algebraic equation will have the point form:

$$ap_i\phi_i = \sum_{j=1}^{nnb_i} anb_{i,j}\phi_{g_{i,j}} + bp_i \tag{15}$$

where **ap** is a vector dimensioned *nnodes*-by-1 that stores the coefficients directly associated with the nodes in the domain, **anb** is a matrix dimensioned *nnodes*-by-*max-nnb* that stores the coefficients associated with the neighbours of node *i*, and **bp** is a vector of coefficients, dimensioned *nnodes*-by-1, that stores additional terms arising from the treatments of sources, transients and boundary conditions. The essential idea in the control volume method is to arrive at the discrete nodal equation in (15) (i.e. determine the coefficients) through a balance of the fluxes across the faces of the control volume constructed around the node (see Fig. 3). This is a two-step process. In the first step the fluxes across the six faces, as defined in (11), for each element are obtained. In the second step the contributions from each of these faces is attributed to the appropriate node, thus assembling eq. (15).

7.1 Step 1: Approximation of fluxes

7.1.1 Diffusive flux

On recognising that the gradient calculated with the approximating shape functions (9) and the outward normal are constant, the diffusive flux into the CV around node

A across face f_1 separating node A from B can be approximated as:

$$\int_{f_1} \kappa \nabla\phi \cdot \mathbf{n} d\Gamma \approx \kappa_{f_1} \nabla\phi \cdot (\Gamma\mathbf{n})_{f_1} \tag{16}$$

Expanding the right hand side, using the result in (9), leads to:

$$\begin{aligned} \kappa_{f_1} \nabla\phi \cdot (\Gamma\mathbf{n})_{f_1} = \kappa_{f_1} (\nabla N_A \cdot (\Gamma\mathbf{n})_{f_1}) \phi_A + \kappa_{f_1} (\nabla N_B \cdot (\Gamma\mathbf{n})_{f_1}) \phi_B + \\ \kappa_{f_1} (\nabla N_C \cdot (\Gamma\mathbf{n})_{f_1}) \phi_C + \kappa_{f_1} (\nabla N_D \cdot (\Gamma\mathbf{n})_{f_1}) \phi_D \end{aligned} \tag{17}$$

Similar expressions for the discretized diffusive fluxes can be written for the other five faces within the tetrahedron. In a given element, on associating the vertex nodes with global numbers A, B, C and D with element node numbers 1, 2, 3 and 4 respectively the evaluation of the diffusive fluxes across all six faces can be stored in the matrix **df** (for diffusive flux) dimensioned 6-by-4 (faces-by-nodes). In this way example elements in **df** include:

$$df_{2,4} = \kappa_{f_2} \nabla N_D \cdot (\Gamma\mathbf{n})_{f_2}; \quad df_{4,3} = \kappa_{f_4} \nabla N_C \cdot (\Gamma\mathbf{n})_{f_4}; \quad df_{5,1} = \kappa_{f_5} \nabla N_A \cdot (\Gamma\mathbf{n})_{f_5} \tag{18}$$

7.1.2 Advective flux

The advective flux across a given face can be approximated as:

$$\int_{\Gamma_f} (\mathbf{v} \cdot \mathbf{n}) \phi d\Gamma \approx \mathbf{v}\phi \cdot \Gamma\mathbf{n}|_f = q_f \phi_f \tag{19}$$

where q_f is the volume flow out across the face, in the direction of the normal vector defined by (11), e.g., (assuming the nodal velocity field is known) at face 1:

$$q_{f_1} = \mathbf{v}_{f_1} \cdot (\Gamma\mathbf{n})_{f_1} = (v_x)_{f_1} (\Gamma n_x)_{f_1} + (v_y)_{f_1} (\Gamma n_y)_{f_1} + (v_z)_{f_1} (\Gamma n_z)_{f_1} \tag{20}$$

where velocity components at centroids of CV faces are calculated using the nodal values:

$$\begin{aligned} \mathbf{v}_{f_1} &= \tfrac{13}{36}\mathbf{v}_A + \tfrac{13}{36}\mathbf{v}_B + \tfrac{5}{36}\mathbf{v}_C + \tfrac{5}{36}\mathbf{v}_D \\ \mathbf{v}_{f_2} &= \tfrac{13}{36}\mathbf{v}_A + \tfrac{5}{36}\mathbf{v}_B + \tfrac{13}{36}\mathbf{v}_C + \tfrac{5}{36}\mathbf{v}_D \\ \mathbf{v}_{f_3} &= \tfrac{13}{36}\mathbf{v}_A + \tfrac{5}{36}\mathbf{v}_B + \tfrac{5}{36}\mathbf{v}_C + \tfrac{13}{36}\mathbf{v}_D \\ \mathbf{v}_{f_4} &= \tfrac{5}{36}\mathbf{v}_A + \tfrac{13}{36}\mathbf{v}_B + \tfrac{13}{36}\mathbf{v}_C + \tfrac{5}{36}\mathbf{v}_D \\ \mathbf{v}_{f_5} &= \tfrac{5}{36}\mathbf{v}_A + \tfrac{13}{36}\mathbf{v}_B + \tfrac{5}{36}\mathbf{v}_C + \tfrac{13}{36}\mathbf{v}_D \\ \mathbf{v}_{f_6} &= \tfrac{5}{36}\mathbf{v}_A + \tfrac{5}{36}\mathbf{v}_B + \tfrac{13}{36}\mathbf{v}_C + \tfrac{13}{36}\mathbf{v}_D \end{aligned} \tag{21}$$

The face values of ϕ in (19) can be calculated using a central difference or upwind approach. In the former, the value of ϕ at a given face is a simple weighting of the nodal values, where the appropriate weights match those used in the velocity interpolations of eq. (21). In the latter, upwind scheme, the face values are set depending on the direction of flow relative to the outward normal:

$$\begin{aligned}
\phi_{f_1} &= \begin{cases} \phi_A \text{ if } q_{f_1} > 0 \\ \phi_B \text{ if } q_{f_1} < 0 \end{cases} \\
\phi_{f_2} &= \begin{cases} \phi_A \text{ if } q_{f_2} > 0 \\ \phi_C \text{ if } q_{f_2} < 0 \end{cases} \\
\phi_{f_3} &= \begin{cases} \phi_A \text{ if } q_{f_3} > 0 \\ \phi_D \text{ if } q_{f_3} < 0 \end{cases} \\
\phi_{f_4} &= \begin{cases} \phi_B \text{ if } q_{f_4} > 0 \\ \phi_C \text{ if } q_{f_4} < 0 \end{cases} \\
\phi_{f_5} &= \begin{cases} \phi_B \text{ if } q_{f_5} > 0 \\ \phi_D \text{ if } q_{f_5} < 0 \end{cases} \\
\phi_{f_6} &= \begin{cases} \phi_C \text{ if } q_{f_6} > 0 \\ \phi_D \text{ if } q_{f_6} < 0 \end{cases}
\end{aligned} \tag{22}$$

Advective fluxes through faces within the tetrahedron can then be written as:

$$\begin{aligned}
q_{f_1}\phi_{f_1} &= \max[q_{f_1},0]\phi_A - \max[-q_{f_1},0]\phi_B \\
q_{f_2}\phi_{f_2} &= \max[q_{f_2},0]\phi_A - \max[-q_{f_2},0]\phi_C \\
q_{f_3}\phi_{f_3} &= \max[q_{f_3},0]\phi_A - \max[-q_{f_3},0]\phi_D \\
q_{f_4}\phi_{f_4} &= \max[q_{f_4},0]\phi_B - \max[-q_{f_4},0]\phi_C \\
q_{f_5}\phi_{f_5} &= \max[q_{f_5},0]\phi_B - \max[-q_{f_5},0]\phi_D \\
q_{f_6}\phi_{f_6} &= \max[q_{f_6},0]\phi_C - \max[-q_{f_6},0]\phi_D
\end{aligned} \tag{23}$$

7.2 Step 2: Assembly of the discrete equation

Assembly of the coefficients in (15) is done an element at a time. The process is initiated by setting the coefficient matrices **ap**, **anb** and **bp** to zero. Then a sweep through each of the tetrahedral elements in the mesh is made. In each element the diffusive and advective fluxes across each of the six faces, using the constructions in (18) and (23) (assuming upwind), are calculated and stored. These values are

then used to incrementally update the coefficients as follows:

Node A :

$$
\begin{aligned}
&ap_A = ap_A - df_{1,1} - df_{2,1} - df_{3,1} + \max[q_{f_1}, 0] + \max[q_{f_2}, 0] + \max[q_{f_3}, 0] \\
&anb_{A,l_{A,B}} = anb_{A,l_{A,B}} + df_{1,2} + df_{2,2} + df_{3,2} + \max[-q_{f_1}, 0] \\
&anb_{A,l_{A,C}} = anb_{A,l_{A,C}} + df_{1,3} + df_{2,3} + df_{3,3} + \max[-q_{f_2}, 0] \\
&anb_{A,l_{A,D}} = anb_{A,l_{A,D}} + df_{1,4} + df_{2,4} + df_{3,4} + \max[-q_{f_3}, 0]
\end{aligned}
$$

Node B :

$$
\begin{aligned}
&ap_B = ap_B + df_{1,2} - df_{4,2} - df_{5,2} + \max[-q_{f_1}, 0] + \max[q_{f_4}, 0] + \max[q_{f_5}, 0] \\
&anb_{B,l_{B,A}} = anb_{B,l_{B,A}} - df_{1,1} + df_{4,1} + df_{5,1} + \max[q_{f_1}, 0] \\
&anb_{B,l_{B,C}} = anb_{B,l_{B,C}} - df_{1,3} + df_{4,3} + df_{5,3} + \max[-q_{f_4}, 0] \\
&anb_{B,l_{B,D}} = anb_{B,l_{B,D}} - df_{1,4} + df_{4,4} + df_{5,4} + \max[-q_{f_5}, 0]
\end{aligned}
$$

Node C :

$$
\begin{aligned}
&ap_C = ap_C + df_{2,3} + df_{4,3} - df_{6,3} + \max[-q_{f_2}, 0] + \max[-q_{f_4}, 0] + \max[q_{f_6}, 0] \\
&anb_{C,l_{C,A}} = anb_{C,l_{C,A}} - df_{2,1} - df_{4,1} + df_{6,1} + \max[q_{f_2}, 0] \\
&anb_{C,l_{C,B}} = anb_{C,l_{C,B}} - df_{2,2} - df_{4,2} + df_{6,2} + \max[q_{f_4}, 0] \\
&anb_{C,l_{C,D}} = anb_{C,l_{C,D}} - df_{2,4} - df_{4,4} + df_{6,4} + \max[-q_{f_6}, 0]
\end{aligned}
$$

Node D :

$$
\begin{aligned}
&ap_D = ap_D + df_{3,4} + df_{5,4} + df_{6,4} + \max[-q_{f_3}, 0] + \max[-q_{f_5}, 0] + \max[-q_{f_6}, 0] \\
&anb_{D,l_{D,A}} = anb_{D,l_{D,A}} - df_{3,1} - df_{5,1} - df_{6,1} + \max[q_{f_3}, 0] \\
&anb_{D,l_{D,B}} = anb_{D,l_{D,B}} - df_{3,2} - df_{5,2} - df_{6,2} + \max[q_{f_5}, 0] \\
&anb_{D,l_{D,C}} = anb_{D,l_{D,C}} - df_{3,3} - df_{5,3} - df_{6,3} + \max[q_{f_6}, 0]
\end{aligned}
\tag{24}
$$

During the assembly it is also useful to calculate and store the volumes of the CV associated with each node. These are stored in the vector **vol** with dimensions *nnodes*-by-1. Before the sweep through the tetrahedral elements in the mesh, this vector is initiated with a zero value. During the sweep, in each tetrahedron, the elements of **vol** are incrementally updated as:

$$
\begin{aligned}
&vol_A = vol_A + \tfrac{1}{4} V_{ABCD} \\
&vol_B = vol_B + \tfrac{1}{4} V_{ABCD} \\
&vol_C = vol_C + \tfrac{1}{4} V_{ABCD} \\
&vol_D = vol_D + \tfrac{1}{4} V_{ABCD}
\end{aligned}
\tag{25}
$$

where V_{ABCD} is calculated from (3).

7.3 Implementation of boundary conditions

The above assembly only takes care of faces on the interior of the domain. Thus after this assembly it is necessary to update the coefficients to account for specified flux, q_{in}, crossing the domain boundary. As noted is section 3 the discrete domain boundary is made up of a piecewise planar triangular mesh. For the i-th triangle on this boundary with global nodes A, B, and C (stored in first three elements of the i-th row of the boundary matrix **s**) the specified flux takes the general form:

$$\int_{\Gamma_{ABC}} q_{in} d\Gamma = \int_{\Gamma_{ABC}} h(\phi_{amb} - \phi)\, d\Gamma \approx \frac{\Gamma_{ABC}}{3} \left[h(\phi_{amb} - \phi_A) + h(\phi_{amb} - \phi_B) + h(\phi_{amb} - \phi_C) \right] \tag{26}$$

where the area of the area triangle Γ_{ABC} is calculated with eq. (4). As illustrated in Table 2, this form allows, on setting values of convective coefficient h and ambient value ϕ_{amb}, insulated, fixed value, convective, and specified flux boundaries. For the i-th triangle on the domain boundary, the appropriate choice of these values is associated with the 4^{th} component on the i-th row of **s,** which provides a unique identifier for each contiguous region of the domain boundary.

Table 2: Possible boundary condition settings

Convective Boundary	Finite setting of h and ϕ_{amb}
Insulated Boundary	h=0, finite value of ϕ_{amb}
Fixed Value Boundary	h=10^{16}, ϕ_{amb}=ϕ_{value} (ϕ_{value} is a prescribed ϕ value)
Fixed Flux Boundary	h=10^{-16}, ϕ_{amb}=q_{in}/h (q_{in} is a prescribed flux value)

In this way, assuming that $h \geq 0$, the previously calculated coefficients are updated on sweeping over the *ns* boundary triangles and setting:

$$\begin{aligned} ap_{s_{i,k}} &= ap_{s_{i,k}} + \tfrac{\Gamma_{ABC}}{3} h_{s_{i,4}}, \;\; k = 1,2,3 \\ bp_{s_{i,k}} &= bp_{s_{i,k}} + \tfrac{\Gamma_{ABC}}{3} h_{s_{i,4}} (\phi_{amb})_{s_{i,4}}, \;\; k = 1,2,3 \end{aligned} \tag{27}$$

7.4 Implementation of source terms

When source terms are present in the problem under consideration they are included in the governing equation (2) as an additional term:

$$\int_{vol} Q dV + \int_{\Gamma} (\mathbf{v} \cdot \mathbf{n}) \phi d\Gamma - \int_{\Gamma} \kappa \nabla \phi \cdot \mathbf{n} d\Gamma = 0 \tag{28}$$

For the CV around a given node i this volume source term (quantity/volume-time) can be approximated by a one-point integration:

$$\int_{vol_i} QdV \approx Q_i vol_i \tag{29}$$

where $Q_i = Q(x_i, y_i)$ is the value of the source evaluated at the location of node i and vol_i is the volume of i-th control volume. In a standard approach the source is represented in the linearized form (Patankar, 1980):

$$Q_i vol_i = -Q_{C_i}\phi_i + Q_{B_i} \tag{30}$$

Then a sweep through the nodes can be used to update the coefficients as follows:

$$ap_i = ap_i + Q_{C_i}; \quad bp_i = bp_i + Q_{B_i} \tag{31}$$

7.5 Solution

Following the assembly and correction for boundary conditions and sources, the formation of the algebraic point equations for steady state advection diffusion problems, i.e., equation (15) is complete. As written the solution for the unknown field can be readily obtained using a convenient point iteration scheme, e.g., Jacobi, Gauss-Siedel or SOR.

8 Transients

In the steady state problems the net flow rate of quantity into the control volume around node i is identically zero:

$$Net_i = -ap_i\phi_i + \sum_{j=1}^{nnb_i} ap_{i,j}\phi_{g_{i,j}} = 0 \tag{32}$$

In the case of transient problems, this net flow rate into the control volume is non-zero which results in a change of accumulation of the quantity in the control volume:

$$vol_i \frac{\phi_i^{new} - \phi_i}{\Delta t} = (1-\theta)Net_i + \theta Net_i^{new} \tag{33}$$

where Δt is the time step. The weighting factor $0 \leq \theta \leq 1$ approximates the net flow into the control volume i within the time interval $[t,\ t+\Delta t]$ in terms of the values at the beginning and at the end of the time step. Values at time $t+\Delta t$ are indicated with superscript *new*.

For $\theta = 0$ the scheme is fully explicit:

$$vol_i \phi_i^{new} = vol_i \phi_i + \Delta t \left(\sum_{j=1}^{nnb_i} ap_{i,j} \phi_{g_{i,j}} - ap_i \phi_i \right) \tag{34}$$

This explicit scheme does not require solution of a system of equation, since the new time level values are directly updated from the current time values. However, in order to ensure positive coefficients and stability of the solution it is necessary that the chosen time step is such that:

$$\Delta t < \min \left(\frac{vol_i}{ap_i} \right), \; i = 1, ..., nnodes \tag{35}$$

For $\theta = 1$ the scheme is fully implicit:

$$\left(\frac{vol_i}{\Delta t} + ap_i \right) \phi_i^{new} = \frac{vol_i}{\Delta t} \phi_i + \sum_{j=1}^{nnb_i} ap_{i,j} \phi_{g_{i,j}}^{new} \tag{36}$$

On the other hand, the implicit schemes are unconditionally stable, but require the solution of a system of equations.

9 Example problems

9.1 Overview

Matlab code that follows the discretization algorithm described above has been written. In the first place, the reliability and accuracy of this code is tested by comparing predictions with known advective-diffusive solutions for axisymmetric problems. This is seen as a true test of the proposed approach since the unstructured 3-D Cartesian mesh and subsequent Cartesian frame numerical solution has zero ability to take any advantage of the symmetric nature of the given problems.

9.2 Steady state diffusion in an annulus

In axisymmetric coordinates, steady state diffusion in an annulus, with constant diffusivity and no sources, is governed by:

$$\frac{d}{dr} \left(r \frac{\partial \phi}{\partial r} \right) = 0 \tag{37}$$

With boundary conditions $\phi \, (r = r_{in}) = \phi_{in}$ and $\phi \, (r = r_{out}) = \phi_{out}$, the solution is:

$$\phi = \phi_{in} + (\phi_{out} - \phi_{in}) \frac{\ln r - \ln r_{in}}{\ln r_{out} - \ln r_{in}} \tag{38}$$

A suitable three-dimensional Cartesian domain for this problem is a one-quarter of annulus with unit wall thickness. The governing equation is:

$$\frac{\partial^2\phi}{\partial x^2}+\frac{\partial^2\phi}{\partial y^2}+\frac{\partial^2\phi}{\partial z^2}=0 \tag{39}$$

with the following boundary conditions:

$$\begin{aligned}&\phi=\phi_{in} \text{ on } \sqrt{x^2+y^2}=r_{in},\ \phi=\phi_{out} \text{ on } \sqrt{x^2+y^2}=r_{out}\\&\frac{\partial\phi}{\partial y}=0 \text{ on } y=0,\ \frac{\partial\phi}{\partial x}=0 \text{ on } x=0\\&\frac{\partial\phi}{\partial z}=0 \text{ on } z=0 \text{ and } z=1\end{aligned} \tag{40}$$

The mesh of tetrahedral elements for the Cartesian domain (setting $r_{in}=1$ and $r_{out}=2$) is shown in Fig. 4. This discretization has 3,431 nodes and 17,471 elements. The solution on this mesh (symbols), when $\phi_{in}=1$ and $\phi_{out}=0$, is compared with the axisymmetric solution (38) (shown as a line) in the lower plot of Fig. 5. The comparison is excellent. Note that the plot for the CVFEM was built by simply randomly selecting 60 points in the domain and making the scatter plot of ϕ_i values against $r_i=\sqrt{x_i^2+y_i^2}$. The realization shown in Fig. 5 is typical. Plotting in this way tests the ability of the CVFEM throughout the domain as opposed to just along a chosen radial line.

In the case of steady state diffusion, with radial dependant diffusivity and source term defined as:

$$\kappa=\frac{1}{r},\ Q=\frac{\alpha}{r},\ \alpha=const \tag{41}$$

the axisymmetric problem can be written as:

$$\frac{d^2\phi}{dr^2}+\alpha=0 \tag{42}$$

With the boundary conditions $\phi\,(r=r_{in})=\phi_{in}=1$ and $\phi\,(r=r_{out})=\phi_{out}=0$ the analytical solution is:

$$\phi=-\frac{\alpha}{2}r^2-\left(1-\frac{3}{2}\alpha\right)r+2-\alpha \tag{43}$$

In the three-dimensional 1/4 annulus Cartesian domain the corresponding governing equation is:

$$\frac{\partial}{\partial x}\left(\kappa\frac{\partial\phi}{\partial x}\right)+\frac{\partial}{\partial y}\left(\kappa\frac{\partial\phi}{\partial y}\right)+\frac{\partial}{\partial z}\left(\kappa\frac{\partial\phi}{\partial z}\right)+Q=0 \tag{44}$$

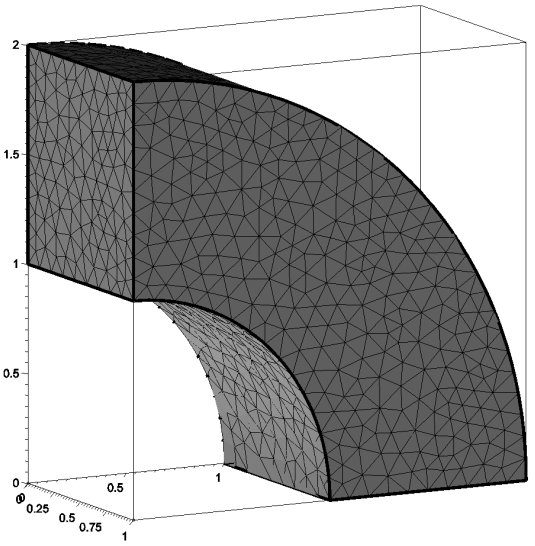

Figure 4: Annulus meshed with tetrahedral elements.

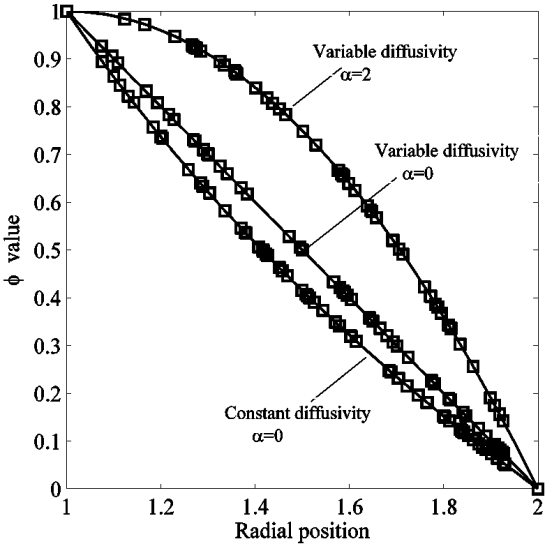

Figure 5: Steady state diffusion in an annulus: axisymmetric analytical solution (solid line) vs. CVFEM solution (symbols)

with boundary conditions as in (40) and diffusivity and source defined as:

$$\kappa = \frac{1}{\sqrt{x^2+y^2}}, \quad Q = \frac{\alpha}{\sqrt{x^2+y^2}} \tag{45}$$

Once again, for two choices of the source term, the Cartesian CVFEEM solution (symbols) is in agreement with the axisymmetric analytical solution (lines); see upper curves in Fig. 5.

9.3 Steady state advection-diffusion in an annulus

In the case of steady state advection-diffusion, with radial velocity defined as $v_r = \frac{1}{r}$ and constant diffusivity $\kappa = 1$ with no heat source, the axisymmetric governing equation is:

$$\frac{d}{dr}\left(r\frac{d\phi}{dr}\right) = 0 \tag{46}$$

With boundary conditions $\phi\,(r = r_{in}) = \phi_{in} = 1$ and $\phi\,(r = r_{out}) = \phi_{out} = 0$ the analytical solution is:

$$\phi = 2\phi_{in} - \phi_{out} + (\phi_{out} - \phi_{in})\, r \tag{47}$$

In three-dimensional Cartesian coordinates the governing equation is:

$$\frac{\partial\,(v_x\phi)}{\partial x} + \frac{\partial\,(v_y\phi)}{\partial y} + \frac{\partial\,(v_z\phi)}{\partial z} - \frac{\partial}{\partial x}\left(\frac{\partial\phi}{\partial x}\right) - \frac{\partial}{\partial y}\left(\frac{\partial\phi}{\partial y}\right) - \frac{\partial}{\partial z}\left(\frac{\partial\phi}{\partial z}\right) = 0 \tag{48}$$

with boundary conditions as defined in (40) and:

$$v_x = \frac{\cos\theta}{\sqrt{x^2+y^2}}, \quad v_y = \frac{\sin\theta}{\sqrt{x^2+y^2}}, \quad \theta = \arctan\frac{y}{x} \tag{49}$$

The CVFEM Cartesian solution (symbols, lower curve) in Fig. 6, for the case compares well with the axisymmetric solution.

When the radial velocity is defined as $v_r = \frac{1}{r}$ and variable diffusivity defined as in (41), the axisymmetric equation is:

$$\frac{\partial\phi}{\partial r} = \frac{\partial^2\phi}{\partial r^2} \tag{50}$$

With the same boundary conditions as above the analytical solution is:

$$\phi = \frac{e^r - e^2}{e - e^2} \tag{51}$$

In three-dimensional Cartesian coordinates the corresponding equation is:

$$\frac{\partial (v_x \phi)}{\partial x} + \frac{\partial (v_y \phi)}{\partial y} + \frac{\partial (v_z \phi)}{\partial y} - \frac{\partial}{\partial x}\left(\kappa \frac{\partial \phi}{\partial x}\right) - \frac{\partial}{\partial y}\left(\kappa \frac{\partial \phi}{\partial y}\right) - \frac{\partial}{\partial z}\left(\kappa \frac{\partial \phi}{\partial z}\right) = 0 \quad (52)$$

with variable diffusivity as defined by (45). The Cartesian CVFEM solution from these equations and the mesh in Fig. 4 is once more very close to the axisymmetric analytical solutions, upper curve in Fig. 6.

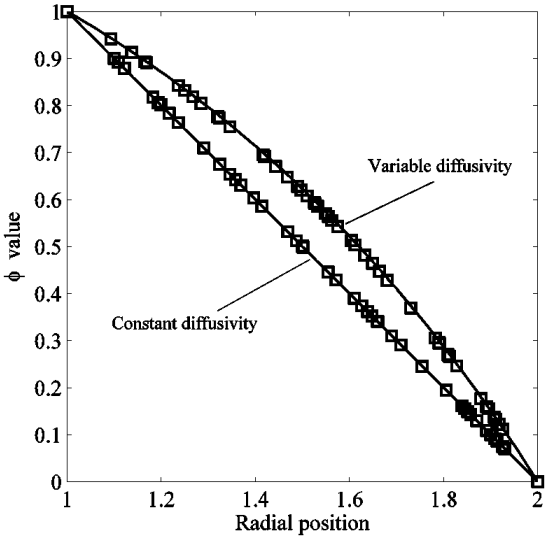

Figure 6: Steady state advection-diffusion in an annulus: axisymmetric analytical solution (solid line) vs. CVFEM solution (symbols)

9.4 Steady state diffusion in a spherical shell

In an axisymmetric coordinates, steady state diffusion in a spherical shell, with constant diffusivity and no sources, is governed by:

$$\frac{d}{dr}\left(r^2 \frac{d\phi}{dr}\right) = 0 \quad (53)$$

With boundary conditions $\phi\,(r = r_{in}) = \phi_{in}$ and $\phi\,(r = r_{out}) = \phi_{out}$, the analytical solution is:

$$\phi = \phi_{in} - r_{out} \frac{\phi_{in} - \phi_{out}}{r_{out} - r_{in}} \frac{r - r_{in}}{r} \quad (54)$$

A suitable three-dimensional Cartesian domain for this problem is a one-eighth of the spherical shell. The governing equation in this domain is:

$$\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} + \frac{\partial^2 \phi}{\partial z^2} = 0 \tag{55}$$

with the following boundary conditions:

$$\begin{gathered} \phi = \phi_{in} \text{ on } \sqrt{x^2+y^2+z^2} = r_{in}, \ \phi = \phi_{out} \text{ on } \sqrt{x^2+y^2+z^2} = r_{out} \\ \frac{\partial \phi}{\partial x} = 0 \text{ on } x = 0, \ \frac{\partial \phi}{\partial y} = 0 \text{ on } y = 0, \ \frac{\partial \phi}{\partial z} = 0 \text{ on } z = 0 \end{gathered} \tag{56}$$

The mesh of tetrahedral elements for this domain (setting$r_{in} = 1$ and $r_{out} = 2$) is shown in Fig. 7. This discretization has 5,361 nodes and 28,544 elements. The solution on this mesh (symbols), when $\phi_{in} = 1$ and $\phi_{out} = 0$, is compared with the axisymmetric solution (line) in Fig. 8. Once again the comparison is excellent.

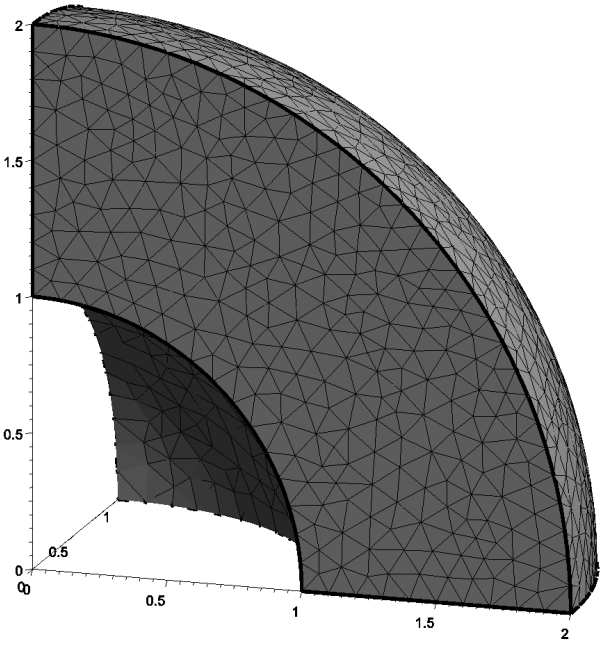

Figure 7: Spherical shell meshed with tetrahedral elements

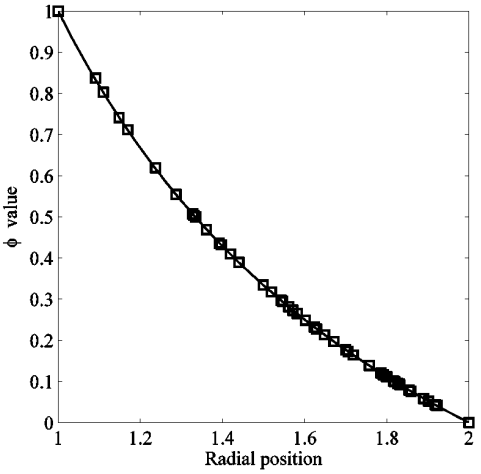

Figure 8: Steady state advection-diffusion in a spherical shell: axisymmetric analytical solution (solid line) vs. CVFEM solution (symbols)

9.5 Unsteady diffusion with phase change

The ability of the three-dimensional CVFEM scheme to handle transients is demonstrated for the case of a heat conduction controlled melting of a bar (x-by-y-by-z dimensions 20-by-1-by-1), initially in solid form at the phase change temperature T=0. Assuming a unit density, diffusivity, specific heat and latent heat, the governing single domain enthalpy equation can be written as (see Crank (1984)):

$$\frac{\partial H}{\partial t} = \frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} + \frac{\partial^2 T}{\partial z^2} \tag{57}$$

where the enthalpy is defined as the sum of sensible and latent heats:

$$H = T + f \tag{58}$$

where f is the liquid fraction ($0 \leq f \leq 1$, f=0 solid, f=1 liquid). In this system melting, moving as a planar front in the positive x direction, is initiated by instantly setting the boundary surface at x=0 to have temperature T=1 and liquid fraction f=1. An analytical treatment of this problem [Crank (1984)], shows that the melt front x-position with time $s(t)$ can be determined as:

$$s(t) = 1.24 \cdot \sqrt{t} \tag{59}$$

In terms of a CVFEM solution an explicit time stepping scheme results in the following nodal enthalpy update:

$$(vol_i + B_{C_i})\, H_i^{new} = vol_i H_i + \Delta t \left(\sum_{j=1}^{nnb_i} anb_{i,j} T_{g_{i,j}} - ap_i T_i + B_{B_i} \right) \tag{60}$$

where terms B_{C_i} and B_{B_i} take into account boundary conditions. In this case, those terms are zero for all nodes except the nodes on the boundary x=0 where $B_{C_i} = 10^{18}$ and $B_{B_i} = 2 \cdot 10^{18}$.

At each time step in the solution an updated enthalpy field H_i^{new} is obtained from (59) and liquid fraction is calculated as:

$$f_i = \min\left(H_i^{new}, 1\right) \tag{61}$$

To move forward to the next time step, updated temperature field needs to be extracted using the following equation:

$$T_i = H_i^{new} - f_i \tag{62}$$

These equations are solved on a grid of 1,501 nodes and 5,718 elements, see Fig. 9. Note in this mesh the average distance between nodes along the x-direction is approximately 0.3.

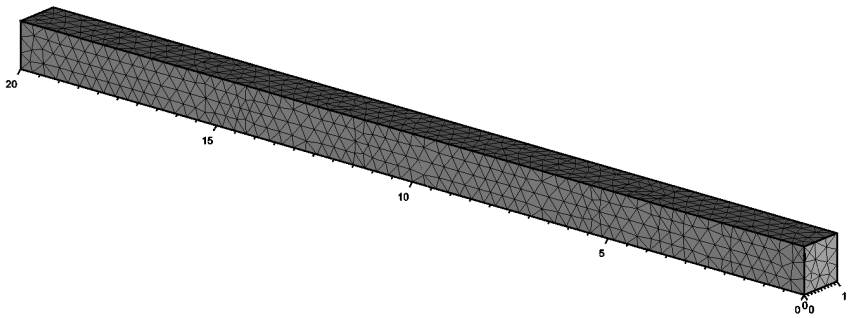

Figure 9: Domain meshed with tetrahedral elements

The stability criterion (35) requires that the time step used is less than 0.0029, here a constant time step of half this value, i.e. Δt=0.00145 is used. At each time step the position of the melt front is determined by:

$$s(t) = 20 \cdot \frac{\sum_{i=1}^{nnodes} f_i vol_i}{\sum_{i=1}^{nnodes} vol_i} \tag{63}$$

The position of the front obtained in this fashion is compared with the analytical solution (59) in Fig. 10. It is seen that the CVFEM solution, plotted every 5,000 time steps, provides a smooth and close approximation to the analytical solution for the melt front position.

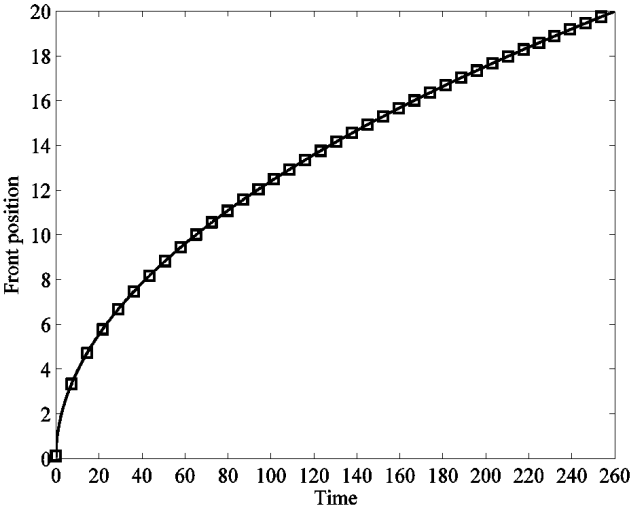

Figure 10: Position of phase change front in time: analytical solution (solid line) vs. CVFEM solution (symbols)

The result in Fig. 10 clearly indicates the ability of the CVFEM to handle problems with transients. There is, however, another interesting attribute to note. It is well known that structured fixed grid enthalpy solutions of unidirectional phase change problems (as studied immediately above) can be subjected to quite severe step like oscillations in the predictions of the movement of the phase front position [Voller (1980) and Crank (1984)]. This is due to the fact that in the enthalpy method on a fixed structured grid, the $y-z$ co-planar node points closest to the phase front can remain fixed at the phase change temperature for quite a considerable number of solution time steps. In problems with Dirichlet boundary conditions this leads to intermittent "locking" of the temperature field in a steady state which is only periodically punctuated by the movement of the phase front from one plane of nodes to the next [Bell (1982)]. In the unstructured CVFEM solution used here, however, it appears as if the non-coplanar arrangement of nodes is able, to a large degree, to "quench" this stepping behavior. This is illustrated in Fig. 11 which plots the relative error of the CVFEM front movement predictions

(black line) against the relative error for predictions of a structured grid which has similar node spacing along the x-dimension (grey line). Relative error is defined as $\left(s(t)_{numerical} - s(t)_{analytical}\right)/s(t)_{analytical}$. Oscillations in the CVFEM3D solution still occur but the precise periodicity is broken and the amplitude of the errors is severely curtailed. Thus suggesting that fixed grid enthalpy CVFEM schemes could be competitive with schemes that employ more sophisticated deforming grid front tracking algorithms (e.g. Becket, Mackenzie and Robertson (2001)).

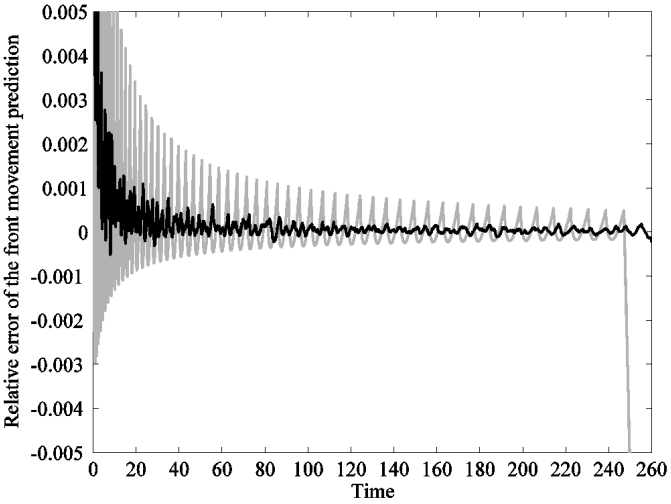

Figure 11: Relative error of the front movement prediction in time: CVFEM3D (black line) vs. Finite Difference 1D (grey line)

10 Conclusions

The Control Volume Finite Element Method (CVFEM) offers the combination of a physically appealing solution approach on a mesh of unstructured nodes. In this paper a detailed, step-by-step unstructured CVFEM discretization of the general three dimensional transient advection-diffusion equation has been given. Emphasis has been placed on ensuring that a clear exposition of the required data structure and assembly, not readily found in the open literature, is provided. The performance and operation of the resulting code has been illustrated and rigorously tested by solving a number of three-dimensional problems exhibiting results that can be compared to known analytical solutions. Of particular note is a transient melting phase change problem which indicates a potential significant advantage for front tracking on fixed

unstructured 3D meshes.

In closing it is noted that the approach used here may not be the most efficient but it is direct and straightforward. This best serves the main purpose of the work which is to provide sufficient guidance for the reader to fully construct a working three dimensional unstructured CVFEM scheme.

Appendix: Creation of auxiliary data structure

It is desirable to have a data structure in order to automate the discretization process and to make it more efficient. In doing so, global and local numbering of neighbouring nodes around each node of the mesh is used. Global numbering of neighbouring nodes can be stored in the two dimensional array **g**. Data structure matrix **g** can be constructed through the loop over tetrahedrons *itet*=1,...,*ntet*) using the following algorithm:

$$
\begin{aligned}
&A = t_{itet,1}, \quad B = t_{itet,2}, \quad C = t_{itet,3}, \quad D = t_{itet,4} \\
&g_{A,counter_A} = B, \quad g_{A,counter_A+1} = C, \quad g_{A,counter_A+2} = D \\
&g_{B,counter_B} = A, \quad g_{B,counter_B+1} = C, \quad g_{B,counter_B+2} = D \\
&g_{C,counter_C} = A, \quad g_{C,counter_C+1} = B, \quad g_{C,counter_C+2} = D \\
&g_{D,counter_D} = A, \quad g_{D,counter_D+1} = B, \quad g_{D,counter_D+2} = C \\
&counter_A = counter_A + 3 \\
&counter_B = counter_B + 3 \\
&counter_C = counter_C + 3 \\
&counter_D = counter_D + 3
\end{aligned}
\tag{64}
$$

where at the beginning of the loop, auxiliary vector *counter* has all elements equal to one. Some elements are repeated one or more times in each row of matrix **g** created using algorithm (64). Therefore, it can be further condensed to eliminate duplicated elements. To perform this elimination and sorting *Matlab* build in function *unique* could be useful. The final data structure matrix **g** contains in each row *i*=1,...,*nnodes* neighbouring nodes of node *i*. Number of columns is equal to the number of neighbours of a node that has the most neighbours. Most of the nodes in the mesh have less neighbours and the rest of their associated rows in **g** is padded with zeros. Number of neighbouring nodes around node *i* is equal to the number of nonzero elements in row *i* of matrix **g** and it is stored in vector *nnb*.

Acknowledgement: Research for this article was supported in part by the Junior Faculty Development Program, which is funded by the Bureau of Educational and Cultural Affairs (ECA) of the United States Department of State, under authority of

the Fulbright-Hays Act of 1961 as amended and administered by American Councils for International Education: ACTR/ACCELS. The opinions expressed herein are the author's own and do not necessarily express the views of either ECA or American Councils.

References

Baliga, B. R. (1978): *A Control-Volume Based Finite Element Method for Convective Heat and Mass Transfer, PhD Thesis.* University of Minnesota.

Baliga, B. R. (1997): Control-Volume Finite Element Methods for Fluid Flow and Heat Transfer, in: Minkowycz, W.J.; Sparrow, E.M. (ed.) *Advances in Numerical Hear Transfer, vol. 1.* Taylor&Francis, Bristol.

Baliga, B. R.; Patankar, S. V. (1980): A New Finite Element Formulation for Convection-Diffusion Problems. *Numerical Heat Transfer*, vol. 3, pp. 393-409.

Baliga, B. R.; Patankar, S. V. (1983): A Control-Volume Finite Element Method for Two-Dimensional Incompressible Fluid Flow and Heat Transfer. *Numerical Heat Transfer*, vol. 6, pp. 245-261.

Baliga, B. R.; Atabaki, N. (2006): Control-volume-based finite-difference and finite-element methods, in: Minkowycz, W.J.; Sparrow, E. M.; Murthu, J. Y. (ed.) *Handbook of Numerical Heat Transfer,* Wiley, Hoboken.

Beckett, G.; Mackenzie, J. A., Robertson M. L. (2001): A Moving Mesh Finite Element Method for the Solution of Two-Dimensional Stefan Problems. *Journal of Computational Physics*, vol. 168, pp. 500-518.

Bell, G. (1982): On the Performance of the Enthalpy Method. *International Journal of Heat and Mass Transfer*, vol. 25, pp. 587-589.

Crank, J. (1984): *Free and Moving Boundary Problems*, Clarendon Press, Oxford.

Geuzaine, C.; Remacle, J.-F. (2009): Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering*, vol. 79, pp. 1309-1331.

Idelshon, S. R.; Oñate E. (1994): Finite volumes and finite elements: Two 'good friends'. *International Journal for Numerical Methods in Engineering*, vol. 37, pp. 3323-3341.

LeDain-Muir, B.; Baliga, B. R. (1986): Solution of Three-Dimensional Convection-Diffusion Problems Using Tetrahedral Elements and Flow-Oriented Upwind Interpolation Functions. *Numerical Heat Transfer*, vol. 9, pp. 143-162.

Patankar, S. V. (1980): *Numerical Heat Transfer and Fluid Flow*. Hemisphere, Washington.

Persson, P-O.; Strang, G. (2004): A simple mesh generator in MATLAB. *SIAM Review*, vol. 46, pp. 329-345.

Prakash, C. (1981): *A Finite Element Method for Predicting Flow through Ducts with Arbitrary Cross-Sections, PhD Thesis.* University of Minnesota.

Ramadhyani, S. (1979): *Solution of the Equations of Convective Heat, Mass, and Momentum Transfer by the Finite Element Method using Quadrilateral Elements, PhD Thesis.* University of Minnesota.

Sladek, J.; Stanak, P.; Han, Z. D.; Sladek, V.; Atluri, S. N. (2013): Applications of the MLPG Method in Engineering & Sciences: A Review. *CMES: Computer Modelling in Engineering & Sciences*, vol. 92, pp. 423-475.

Šarler, B. (2005): A Radial Basis Function Collocation Approach in Computational Fluid Dynamics. *CMES: Computer Modelling in Engineering & Sciences*, vol. 7, pp. 185-193.

Voller, V. R. (2009): *Basic Control Volume Finite Element Methods for Fluids and Solids, IISc Research Monographs – Vol. 1.* World Scientific, Singapore.

Voller, V.; Cross, M. (1981): Accurate Solutions of Moving Boundary Problems Using the Enthalpy Method. *International Journal of Heat and Mass Transfer*, vol. 24, pp. 545-556.

Williamson, D. (1969): Numerical Integration of Fluid Flow over Triangular Grids. *Monthly Weather Reviews*, vol. 97, pp. 885-895.

Winslow, A. M. (1967): Numerical Solution of the Quasilinear Poisson Equation in a Nonuniform Triangle Mesh. *Journal of Computational Physics*, vol. 2, pp. 149-172.

Wrobel, L. C.; Aliabadi, M. H. (2002): *The Boundary Element Method, Volume 2: Applications in Solids and Structures*, Wiley, New York.

Zienkiewicz, O. C.; Taylor, R. L. (2000): *The Finite Element Method, Volume 1: The Basis* (4th ed.), Butterworth–Heinemann, Oxford.