Symmetry plane boundary conditions for cell-centred finite-volume continuum mechanics

I. Demirdžić[1] and P. Cardiff[2]

[1]Mašinski fakultet Sarajevo, Vilsonovo šetalište 9, 71000 Sarajevo, Bosnia-Herzegovina
[2]University College Dublin, School of Mechanical and Materials Engineering, Belfield, Ireland

March 2022

Abstract

A general approach to deriving the symmetry plane boundary conditions for cell-centred finite-volume continuum mechanics is presented. It is equally applicable to scalar, vector, and tensor solution variables. The total contribution of the symmetry plane cell faces to the next-to-symmetry-plane cells is decomposed into implicit and explicit parts, allowing the use of segregated solution algorithms. Using several unstructured mesh test cases, the derived symmetry plane discretisations are shown to be consistent with the discretisation on the internal faces.

Keywords: *Symmetry plane; Finite-volume method; Continuum mechanics*

1 Introduction

Many problems in engineering are symmetric about one, two or even three planes. Defining symmetry plane(s) on the solution domain boundary enables one to reduce the solution domain, the computer memory and the CPU time by a factor of two or more. However, while imposing correct boundary conditions for scalar solution variables is relatively simple, treating vector and especially tensor variables at symmetry planes is not a straightforward matter. This is particularly the case when the symmetry plane normal is not aligned with any of the coordinate system directions or when unstructured numerical meshes are used.

The starting point is the fact that by using the symmetry plane boundary conditions, *the same results* should be obtained by solving for the complete domain. Thus, to arrive at the correct symmetry plane boundary conditions, in addition to the physical problem symmetry, one has to assume *numerical mesh symmetry*. This can be achieved by constructing a fictitious mirror-image cell M of the next-to-symmetry-plane cell P (Figure 1).

The convection flux through the symmetry plane face, if it exists, is zero, and the diffusion flux through the symmetry-plane surface S is

$$D_\phi \;=\; \int_S \Gamma_\phi \, \mathbf{n} \cdot \operatorname{grad} \phi \, \mathrm{d}S \;\approx\; \Gamma_\phi \, \frac{\mathbf{s} \cdot \mathbf{s}}{\boldsymbol{\Delta} \cdot \mathbf{s}} \, (\phi_M - \phi_P) \;+\; \text{cross-diffusion term} \tag{1}$$

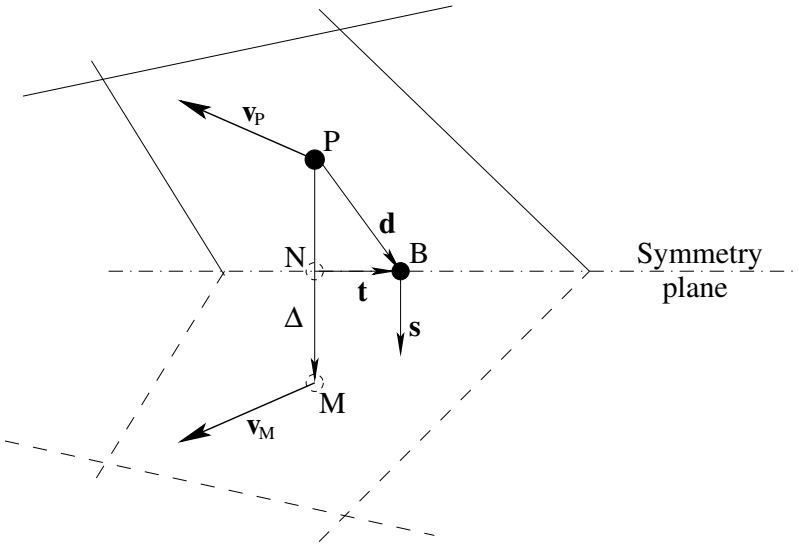


Figure 1: Symmetry plane.

where the unit normal vector is

$$\mathbf{n} = \frac{\mathbf{s}}{|\mathbf{s}|} \tag{2}$$

and $\mathbf{s}$ is the surface vector with its magnitude $|\mathbf{s}|$ equal to the area of the boundary cell face [1]. The generic variable ϕ stands for the scalar ψ, the vector $\mathbf{v}$, or the tensor $\mathbf{T}$ variable, Γ_ϕ is the diffusion coefficient, ϕ_M and ϕ_P are the values of the variable ϕ at points M and P, and the vector joining the centres of the cell P and its mirror image M is

$$\boldsymbol{\Delta} \;=\; \overrightarrow{PM} \;=\; 2\,(\mathbf{n}\otimes\mathbf{n})\cdot\mathbf{d} \;=\; 2\,\frac{\mathbf{d}\cdot\mathbf{s}}{\mathbf{s}\cdot\mathbf{s}}\,\mathbf{s} \tag{3}$$

Vector $\mathbf{d}$ joins the cell P centre to its symmetry-plane boundary face B. Since vector $\boldsymbol{\Delta}$ is collinear with vector $\mathbf{s}$ (orthogonal to the symmetry plane), the cross-diffusion term in Eq. (1) is zero and the flux through the symmetry-plane cell face is

$$D_\phi \;\approx\; \Gamma_\phi\,\frac{\mathbf{s}\cdot\mathbf{s}}{2\,\mathbf{d}\cdot\mathbf{s}}(\phi_M - \phi_P) \tag{4}$$

Thus, the problem of calculating the symmetry plane boundary face contribution to the next-to-symmetry-plane cell P reduces to the calculation of ϕ_M.

Note that the value of the dependent variable ϕ at the boundary point B, which can be required to calculate the cell-centres gradients or for postprocessing, has remained unknown. Since the symmetry-plane face is essentially an internal cell face, the calculation of the symmetry-plane values must be consistent with interpolations on the internal faces. The dependent variables at the internal cell faces are frequently obtained by linear interpolation between two points straddling the cell face. In that case

$$\phi_B \approx \phi_N = \frac{1}{2}\,(\phi_P + \phi_M) \tag{5}$$

However, if the numerical mesh is highly skewed or in case of significant variation of ϕ along the symmetry plane, a more accurate value would be

$$\phi_B = \phi_N + \mathbf{t} \cdot (\text{grad } \phi)_N = \phi_N + \mathbf{t} \cdot \frac{1}{2} \left[(\text{grad } \phi)_P + (\text{grad } \phi)_M \right] \tag{6}$$

where **t** is the vector joining points N and B

$$\mathbf{t} = \overrightarrow{NB} \; = \; (\mathbf{I} - \mathbf{n} \otimes \mathbf{n}) \cdot \mathbf{d} \; = \; \mathbf{d} - \frac{\mathbf{d} \cdot \mathbf{s}}{\mathbf{s} \cdot \mathbf{s}} \mathbf{s} \tag{7}$$

where **I** is the second-order identity tensor.

Some aspects of this approach have been used by Harlow and Welch [2] for Cartesian meshes and extended to boundary-fitted curvilinear grids for scalar and vector variables by Oliveira [3] and more recently by Moukalled et al. [4].

In this article, the symmetry-plane boundary conditions for the scalar (e.g. energy, electric scalar potential), the vector (e.g. the fluid velocity, the solid body displacement), and the tensor (e.g. the Reynolds stress tensor, the stress tensor in the Oldroyd viscoelastic models) solution variables are presented. Several test cases are designed to verify this approach.

2 Calculation of diffusion flux

In this section, the contribution to the next-to-symmetry-plane cell coming from the diffusion flux through the symmetry-plane cell face is derived for scalar, vector and tensor variables by employing the following *reflection tensor*

$$\mathbf{R} = \mathbf{I} - 2\,\mathbf{n} \otimes \mathbf{n} = R_{ij}\,\mathbf{e}_i \otimes \mathbf{e}_j \tag{8}$$

where $\mathbf{e}_i$ $(i = 1, 2, 3)$ are the Cartesian unit base vectors.

2.1 Scalar variable

The mirror reflection ψ_M of the scalar ψ_P is simply

$$\psi_M = \psi_P \tag{9}$$

and the diffusion flux (Equation 4) is

$$D_\psi = \Gamma_\psi \, \frac{\mathbf{s} \cdot \mathbf{s}}{2\,\mathbf{d} \cdot \mathbf{s}} \, (\psi_M - \psi_P) = 0 \tag{10}$$

which means that there is no contribution to the next-to-symmetry-plane cell P balance from the symmetry plane boundary faces.

2.2 Vector variable

The mirror reflection $\mathbf{v}_M$ of the vector $\mathbf{v}_P$ can be obtained by multiplying that vector with the reflection tensor (Equation 8)

$$\mathbf{v}_M = \mathbf{R} \cdot \mathbf{v}_P = (\mathbf{I} - 2\,\mathbf{n} \otimes \mathbf{n}) \cdot \mathbf{v}_P = \mathbf{v}_P - 2\,\mathbf{n}\,(\mathbf{n} \cdot \mathbf{v}_P) = \mathbf{v}_P - 2\,\frac{\mathbf{s}\,(\mathbf{s} \cdot \mathbf{v}_P)}{\mathbf{s} \cdot \mathbf{s}} \tag{11}$$

where the use of Equation (2) has been made and the diffusion flux (Equation (4)) for vector variables takes the following form

$$\mathbf{D}_v = \Gamma_v\,\frac{\mathbf{s} \cdot \mathbf{s}}{2\,\mathbf{d} \cdot \mathbf{s}}(\mathbf{v}_M - \mathbf{v}_P) = -\Gamma_v\,\frac{\mathbf{s} \cdot \mathbf{v}_P}{\mathbf{d} \cdot \mathbf{s}}\,\mathbf{s} \tag{12}$$

With a segregated solution algorithm in mind, one can write Equation (12) in terms of the Cartesian components

$$D_{vi} = -\Gamma_v\,\frac{s_1 v_{P1} + s_2 v_{P2} + s_3 v_{P3}}{d_1 s_1 + d_2 s_2 + d_3 s_3}\,s_i \quad (i = 1, 2, 3) \tag{13}$$

and see that the contribution of these terms can be split into the explicit and the implicit part as [5]

$$D_{vi} = D_{vi_{ex}} - D_{vi_{im}}\,v_{Pi} \quad (i = 1, 2, 3) \tag{14}$$

The implicit contribution to the individual Cartesian component can be written as

$$D_{vi_{im}} = \Gamma_v\,\frac{s_i s_i}{d_1 s_1 + d_2 s_2 + d_3 s_3} \quad (i = 1, 2, 3) \tag{15}$$

and the explicit contribution as

$$D_{vi_{ex}} = -\Gamma_v\,\frac{s_1 v_{P1} + s_2 v_{P2} + s_3 v_{P3} - s_i v_{Pi}}{d_1 s_1 + d_2 s_2 + d_3 s_3}\,s_i \quad (i = 1, 2, 3) \tag{16}$$

2.3 Tensor variable

Similar to vectors, the reflection of a tensor can be obtained by multiplying that tensor by the reflection tensor (Equation 8). Thus, the mirror image $\mathbf{T}_M$ of the tensor $\mathbf{T}_P$ is

$$\begin{aligned}\mathbf{T}_M &= \mathbf{R} \cdot \mathbf{T}_P \cdot \mathbf{R}^{\mathrm{T}} = (\mathbf{I} - 2\,\mathbf{n} \otimes \mathbf{n}) \cdot \mathbf{T}_P \cdot (\mathbf{I} - 2\,\mathbf{n} \otimes \mathbf{n}) \\ &= \mathbf{T}_P - 2\,(\mathbf{n} \otimes \mathbf{n}) \cdot \mathbf{T}_P - 2\,\mathbf{T}_P \cdot (\mathbf{n} \otimes \mathbf{n}) + 4\,(\mathbf{n} \otimes \mathbf{n} : \mathbf{T}_P)\,\mathbf{n} \otimes \mathbf{n}\end{aligned} \tag{17}$$

Using Equation (2), the diffusion flux (Equation (4)) for tensor variables takes the following form

$$\mathbf{D}_T = \Gamma_T\,\frac{\mathbf{s} \cdot \mathbf{s}}{2\,\mathbf{d} \cdot \mathbf{s}}(\mathbf{T}_M - \mathbf{T}_P) = -\Gamma_T\,\frac{(\mathbf{s} \otimes \mathbf{s}) \cdot \mathbf{T}_P + \mathbf{T}_P \cdot (\mathbf{s} \otimes \mathbf{s}) - \dfrac{2\,(\mathbf{s} \otimes \mathbf{s} : \mathbf{T}_P)\,\mathbf{s} \otimes \mathbf{s}}{\mathbf{s} \cdot \mathbf{s}}}{\mathbf{d} \cdot \mathbf{s}} \tag{18}$$

Noting the index notation form of the dot product of two tensors ($\mathbf{A} \cdot \mathbf{B} = A_{ik} B_{kj} \mathbf{e}_i \otimes \mathbf{e}_j$) and the double-dot product of two tensors ($\mathbf{A} : \mathbf{B} = A_{ij} B_{ij}$)), Equation (18) can be written in terms of

Cartesian components as

$$D_{Tij} = -\Gamma_T \frac{s_i(s_k T_{Pkj}) + s_j(s_k T_{Pik}) - \dfrac{2(s_p s_q T_{Ppq}) s_i s_j}{s_k s_k}}{d_k s_k} \quad (i,j = 1,2,3) \tag{19}$$

Like in the vector case, this term can be split into explicit and implicit parts

$$D_{Tij} = D_{Tij_{ex}} - D_{Tij_{im}} T_{Pij} \quad (i,j = 1,2,3) \tag{20}$$

Thus, the implicit contribution to the individual Cartesian components can be written as (no summation on i and j)

$$D_{Tij_{im}} = \Gamma_T \frac{s_i s_i + s_j s_j}{d_k s_k} \quad (i,j = 1,2,3) \tag{21}$$

and the explicit contribution as (no summation on i and j)

$$D_{Tij_{ex}} = -\Gamma_T \frac{s_i(s_k T_{Pkj}) + s_j(s_k T_{Pik}) - \dfrac{2(s_p s_q T_{Ppq}) s_i s_j}{s_k s_k} - (s_i s_i + s_j s_j) T_{Pij}}{d_k s_k} \quad (i,j = 1,2,3) \tag{22}$$

3 Calculation of boundary-point value

This section calculates the value of the dependent variable ϕ at the boundary point B.

3.1 Scalar variable

By using Equations (5) and (9) the value of the scalar ψ at the boundary point B is

$$\psi_B \approx \frac{1}{2}(\psi_P + \psi_M) = \psi_P \tag{23}$$

or according to Equation (6)

$$\psi_B = \psi_N + \mathbf{t} \cdot \frac{1}{2}[(\text{grad } \psi)_P + (\text{grad } \psi)_M] = \psi_P + \mathbf{t} \cdot \frac{1}{2}(\mathbf{g}_P + \mathbf{g}_M) \tag{24}$$

The mirror image $\mathbf{g}_M$ of the scalar gradient

$$\mathbf{g}_P = (\text{grad } \psi)_P = \left(\frac{\partial \psi}{\partial x_i}\right)_P \mathbf{e}_i \tag{25}$$

is according to Equation (11)

$$\mathbf{g}_M = \mathbf{R} \cdot \mathbf{g}_P = \mathbf{g}_P - 2\frac{\mathbf{s} \cdot \mathbf{g}_P}{\mathbf{s} \cdot \mathbf{s}}\mathbf{s} \tag{26}$$

Since $\mathbf{t} \cdot \mathbf{s} = 0$, this gives the boundary value

$$\psi_B = \psi_P + \mathbf{t} \cdot \left(\mathbf{g}_P - \frac{\mathbf{s} \cdot \mathbf{g}_P}{\mathbf{s} \cdot \mathbf{s}}\mathbf{s}\right) = \psi_P + \mathbf{t} \cdot \mathbf{g}_P \tag{27}$$

or in terms of Cartesian components

$$\psi_B = \psi_P + t_i \, g_{Pi} \tag{28}$$

3.2 Vector variable

Using Equations (5) and (11) the value of the vector **v** at the boundary point B is

$$\mathbf{v}_B \approx \frac{1}{2} \left(\mathbf{v}_P + \mathbf{v}_M \right) = \mathbf{v}_P - \frac{\mathbf{s} \cdot \mathbf{v}_P}{\mathbf{s} \cdot \mathbf{s}} \, \mathbf{s} \tag{29}$$

or in terms of the Cartesian components

$$v_{Bi} \approx v_{Pi} - \frac{s_j v_{Pj}}{s_j s_j} \, s_i \quad (i = 1, 2, 3) \tag{30}$$

According to Equation (6)

$$\mathbf{v}_B = \mathbf{v}_P - \frac{\mathbf{s} \cdot \mathbf{v}_P}{\mathbf{s} \cdot \mathbf{s}} \, \mathbf{s} + \mathbf{t} \cdot \frac{1}{2} \left(\mathbf{G}_P + \mathbf{G}_M \right) \tag{31}$$

The mirror image $\mathbf{G}_M$ of the vector gradient

$$\mathbf{G}_P = (\text{grad } \mathbf{v})_P = \left(\frac{\partial v_j}{\partial x_i} \right)_P \mathbf{e}_i \otimes \mathbf{e}_j \tag{32}$$

is according to Equation (17)

$$\begin{aligned} \mathbf{G}_M &= \mathbf{R} \cdot \mathbf{G}_P \cdot \mathbf{R}^{\mathrm{T}} = (\mathbf{I} - 2\,\mathbf{n} \otimes \mathbf{n}) \cdot \mathbf{G}_P \cdot (\mathbf{I} - 2\,\mathbf{n} \otimes \mathbf{n}) \\ &= \mathbf{G}_P - \frac{2\,\mathbf{s} \otimes \mathbf{s} \cdot \mathbf{G}_P - 2\,\mathbf{G}_P \cdot \mathbf{s} \otimes \mathbf{s} + \dfrac{4\,(\mathbf{s} \otimes \mathbf{s} : \mathbf{G}_P)\,\mathbf{s} \otimes \mathbf{s}}{\mathbf{s} \cdot \mathbf{s}}}{\mathbf{s} \cdot \mathbf{s}} \end{aligned} \tag{33}$$

which, since $\mathbf{t} \cdot \mathbf{s} = 0$, gives the boundary value

$$\begin{aligned} \mathbf{v}_B &= \mathbf{v}_P - \frac{\mathbf{s} \cdot \mathbf{v}_P}{\mathbf{s} \cdot \mathbf{s}} \, \mathbf{s} + \mathbf{t} \cdot \mathbf{G}_P - \mathbf{t} \cdot \frac{\mathbf{s} \otimes \mathbf{s} \cdot \mathbf{G}_P - \mathbf{G}_P \cdot \mathbf{s} \otimes \mathbf{s} + \dfrac{2\,(\mathbf{s} \otimes \mathbf{s} : \mathbf{G}_P)\,\mathbf{s} \otimes \mathbf{s}}{\mathbf{s} \cdot \mathbf{s}}}{\mathbf{s} \cdot \mathbf{s}} \\ &= \mathbf{v}_P - \frac{\mathbf{s} \cdot \mathbf{v}_P}{\mathbf{s} \cdot \mathbf{s}} \, \mathbf{s} + \mathbf{t} \cdot \mathbf{G}_P - \mathbf{t} \cdot \frac{\mathbf{G}_P \cdot \mathbf{s} \otimes \mathbf{s}}{\mathbf{s} \cdot \mathbf{s}} \end{aligned} \tag{34}$$

or in terms of Cartesian components

$$v_{Bi} = v_{Pi} - \frac{s_j v_{Pj}}{s_j s_j} \, s_i + t_j \, G_{Pji} - t_j \, \frac{G_{Pjk} s_k}{s_k s_k} \, s_i \tag{35}$$

3.3 Tensor variable

According to Equation (5), the value of the tensor $\mathbf{T}$ at the boundary point B is

$$\begin{aligned}\mathbf{T}_B \approx \frac{1}{2}\left(\mathbf{T}_P + \mathbf{T}_M\right) &= \mathbf{T}_P - \mathbf{n}\otimes\mathbf{n}\cdot\mathbf{T}_P - \mathbf{T}_P\cdot\mathbf{n}\otimes\mathbf{n} + 2\left(\mathbf{n}\otimes\mathbf{n}:\mathbf{T}_P\right)\mathbf{n}\otimes\mathbf{n} \\ &= \mathbf{T}_P - \frac{1}{\mathbf{s}\cdot\mathbf{s}}\left[\mathbf{s}\otimes\mathbf{s}\cdot\mathbf{T}_P + \mathbf{T}_P\cdot\mathbf{s}\otimes\mathbf{s} - \frac{2\left(\mathbf{s}\otimes\mathbf{s}:\mathbf{T}_P\right)\mathbf{s}\otimes\mathbf{s}}{\mathbf{s}\cdot\mathbf{s}}\right] \end{aligned} \quad (36)$$

where the use of Equation (2) has been made, or in terms of Cartesian components

$$T_{Bij} \approx T_{Pij} - \frac{s_i(s_k T_{Pkj}) + s_j(s_k T_{Pik}) - \dfrac{2(s_p s_q\, T_{Ppq})s_i s_j}{s_k s_k}}{s_k s_k} \quad (i,j=1,2,3) \quad (37)$$

and the value at the boundary point B, cf. Equation (6), is

$$\mathbf{T}_B = \mathbf{T}_P - \frac{\mathbf{s}\otimes\mathbf{s}\cdot\mathbf{T}_P + \mathbf{T}_P\cdot\mathbf{s}\otimes\mathbf{s} - \dfrac{2\left(\mathbf{s}\otimes\mathbf{s}:\mathbf{T}_P\right)\mathbf{s}\otimes\mathbf{s}}{\mathbf{s}\cdot\mathbf{s}}}{\mathbf{s}\cdot\mathbf{s}} + \mathbf{t}\cdot\frac{1}{2}\left(\boldsymbol{\mathcal{G}}_P + \boldsymbol{\mathcal{G}}_M\right) \quad (38)$$

The mirror image $\boldsymbol{\mathcal{G}}_M$ of the tensor gradient

$$\boldsymbol{\mathcal{G}}_P = (\text{grad }\mathbf{T})_P = \left(\frac{\partial T_{jk}}{\partial x_i}\right)_P \mathbf{e}_i\otimes\mathbf{e}_j\otimes\mathbf{e}_k = (\mathcal{G}_{ijk})_P\,\mathbf{e}_i\otimes\mathbf{e}_j\otimes\mathbf{e}_k \quad (39)$$

is

$$\boldsymbol{\mathcal{G}}_M = R_{li}R_{mj}R_{nk}\,(\mathcal{G}_{lmn})_P\,\mathbf{e}_i\otimes\mathbf{e}_j\otimes\mathbf{e}_k \quad (40)$$

It can be seen that the expanded form of Equation (38) becomes quite cumbersome and is not presented here.

4 Example

Consider a symmetry plane parallel to the $x-y$ coordinate plane. In that case, the surface vector

$$\mathbf{s} = (0,0,s_3) \quad (41)$$

and the only non-zero contribution from the symmetry plane to the cell P for vector variables is (Equations (13) to (16))

$$D_{v3} = -\Gamma_v\frac{s_3}{d_3}\,v_{P3}, \quad \Rightarrow \quad D_{v3im} = \Gamma_v\frac{s_3}{d_3}, \quad D_{v3ex} = 0 \quad (42)$$

and for tensor variables (Equations (19) to (22))

$$D_{Tij} = -\Gamma_T\frac{s_3}{d_3}\,T_{Pij}, \quad \Rightarrow \quad D_{Tij\,im} = \Gamma_T\frac{s_3}{d_3}, \quad D_{Tij\,ex} = 0 \quad (ij = 13,31,23,32) \quad (43)$$

The non-zero components of the vector $\mathbf{v}_B$ are (Equation (30))

$$v_{B1} \approx v_{P1}, \quad v_{B2} \approx v_{P2} \quad (44)$$

and of the tensor $\mathbf{T}_B$ (Equation (37))

$$T_{B33} \approx T_{P33} \tag{45}$$

5 Verification

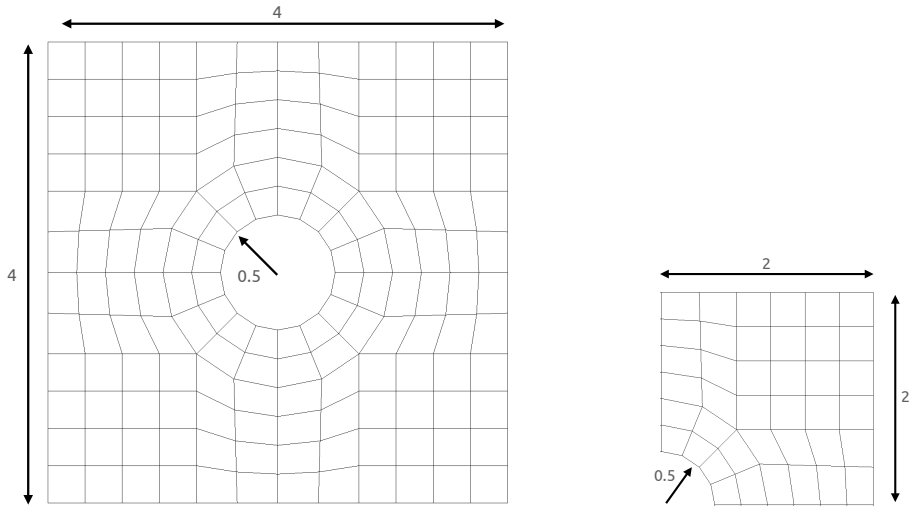
5.1 Problem definition

In this section, three cases are shown to numerically verify the current approach, i.e. to show that identical results are obtained when using symmetry plane(s) compared with solving for the complete domain.

The three cases all use the same 2-D geometry and mesh (Figure 2), with each solving a different form of the steady-state diffusion equation:

1. Scalar diffusion equation: $\mathscr{D}\boldsymbol{\nabla}^2\phi = 0$;
2. Vector diffusion equation: $\mathscr{D}\boldsymbol{\nabla}^2\boldsymbol{v} = \mathbf{0}$;
3. Tensor diffusion equation: $\mathscr{D}\boldsymbol{\nabla}^2\boldsymbol{T} = \mathbf{0}$.

The diffusivity, $\mathscr{D}$, is assumed here to be uniform and equal to unity. The full domain mesh contains 160 quadrilateral cells, while the symmetry unit mesh contains 40 cells. This relatively coarse mesh has been chosen to allow differences to be more easily seen, if any.



(a) Full domain (b) Symmetry unit

Figure 2: Geometry used for all three verification cases. The full domain mesh contains 160 cells, whereas the symmetry unit contains 40 cells. All dimensions are given in metres. The x direction is the horizontal, y is the vertical direction, and z is directed out of the plane.

The boundary conditions are shown in Figure 3, with Dirichlet conditions on the left, right and centre hole boundaries and Neumann conditions on the top and bottom boundaries. The boundary

condition definitions are chosen such that they may be easily applied regardless of the rigid rotation of the initial domains.

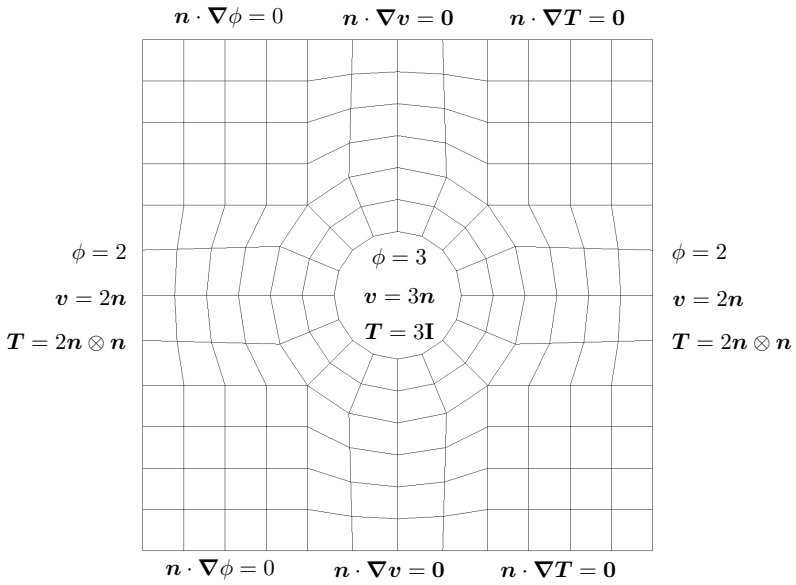


Figure 3: Boundary conditions for the full domain. The symmetry unit uses the same boundary conditions in addition to two symmetry conditions.

In each case, the case is solved for three different configurations:

(a) Geometry is not rotated, and is as given in Figure 2;

(b) Geometry is rotated in the xy plane, based on the angle between the vectors $(1\ 0\ 0)$ and $(2\ 1\ 0)$;

(c) Geometry is rotated in the x, y and z directions, based on the angle between the vectors $(1\ 0\ 0)$ and $(2\ 1\ 3)$.

A preconditioned conjugate gradient iterative solver is used to solve the linear systems, with a tolerance of 1×10^{-12}. An outer loop is performed to allow the explicit deferred correction terms to converge, also with a tolerance of 1×10^{-12}. These tolerances effectively represent the machine limit using double-precision floating-point numbers. The convergence of the outer loop residual is compared for each case.

For each case, the solutions from the symmetry units and the rotated full domain problems are compared with the solution from the corresponding not-rotated full domain problem. The cell-by-cell difference in solution fields (ϕ, $\boldsymbol{v}$ and $\boldsymbol{T}$) are calculated and the L_1, L_2 and L_∞ norms of the difference fields are presented.

5.2 Results

The predicted solution fields are graphically shown in Figure 4 for the scalar cases, in Figure 5 for the vector cases, and in Figure 6 for the tensor cases.

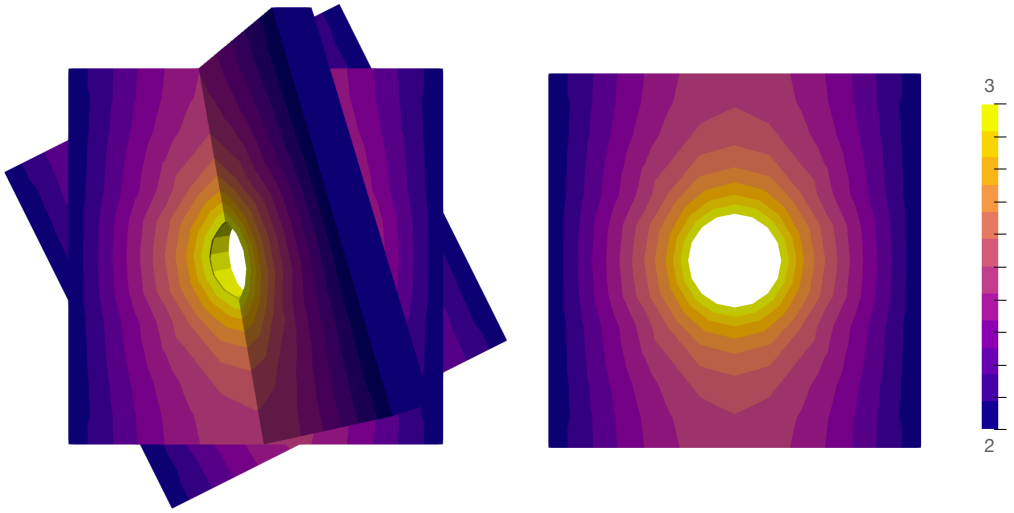


Figure 4: Results for the scalar problems, where ϕ is shown. On the left, the results for the three configurations are shown and on the right the results for the not-rotated geometry is shown on its own. The displayed fields are interpolated for the cell-centres to the points.

Table 1 gives the differences between the predictions from all the cases and the not-rotated full domain problem for the scalar cases. The corresponding results for the vector cases can be found in Table 2 and for the tensor cases in Tables 3. From these tables, all converged solutions are shown to match the not-rotated full domain solutions to the machine limit.

Case	L_1	L_2	L_∞
Symmetry unit (not rotated)	1.65e-11	9.35e-13	1.20e-12
Symmetry unit (rotate XY)	1.65e-11	9.36e-13	1.20e-12
Symmetry unit (rotate XYZ)	1.65e-11	9.33e-13	1.20e-12
Full domain (rotate XY)	1.41e-13	1.78e-15	2.66e-15
Full domain (rotate XYZ)	9.55e-14	4.44e-16	2.22e-15

Table 1: Scalar equation problem: L_1, L_2 and L_∞ norms of the differences between the non-rotated full domain problem and the other cases (all symmetry units, and rotated full domains)

As a final comparison between the cases, the convergence of the outer loop residuals is given in Figure 7 for the scalar cases, in Figure 8 for the vector cases, and Figure 9 for the tensor cases. For the scalar cases (Figure 7), the convergence behaviour is the same for all cases; this is expected as the scalar symmetry discretisation does not depend on the symmetry plane normal. The 15 outer

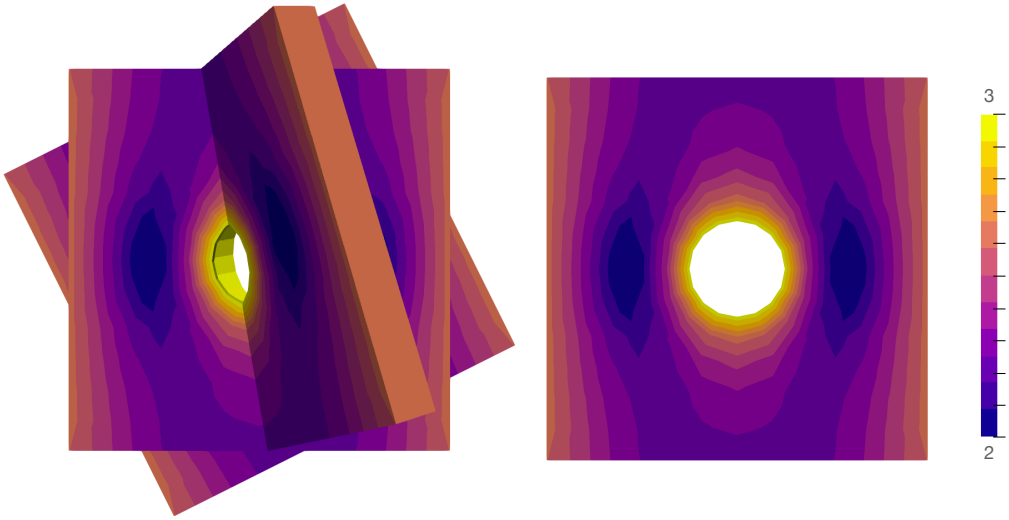


Figure 5: Results for the vector problems, where the magnitude of $\boldsymbol{v}$ is shown. On the left, the results for the three configurations are shown and on the right the results for the not-rotated geometry is shown on its own. The displayed fields are interpolated for the cell-centres to the points.

Case	L_1	L_2	L_∞
Symmetry unit (not rotated)	3.14e-11	2.28e-12	2.28e-12
Symmetry unit (rotate XY)	4.78e-10	2.90e-11	3.75e-11
Symmetry unit (rotate XYZ)	1.15e-10	5.30e-12	6.04e-12
Full domain (rotate XY)	3.14e-10	6.48e-12	7.21e-12
Full domain (rotate XYZ)	5.99e-10	8.88e-12	1.10e-11

Table 2: Vector equation problem: L_1, L_2 and L_∞ norms of the differences between the non-rotated full domain problem and the other cases (all symmetry units, and rotated full domains)

correctors are due to the non-orthogonal corrections terms on internal faces, which are dealt with in a deferred manner. Inspecting the results for the vector (Figure 8) and tensor (Figure 9) cases, several observations can be made:

- The full domain cases show the same convergence behaviour regardless of whether they are rotated or not;
- The rotated symmetry units take 2 to 20 times as many outer iterations to convergence as the full domain cases;
- In the tensor case, the not-rotated symmetry unit took the greatest number of iterations to converge.

Based on these observations, if the symmetry planes are not oriented in the Cartesian axis direc-

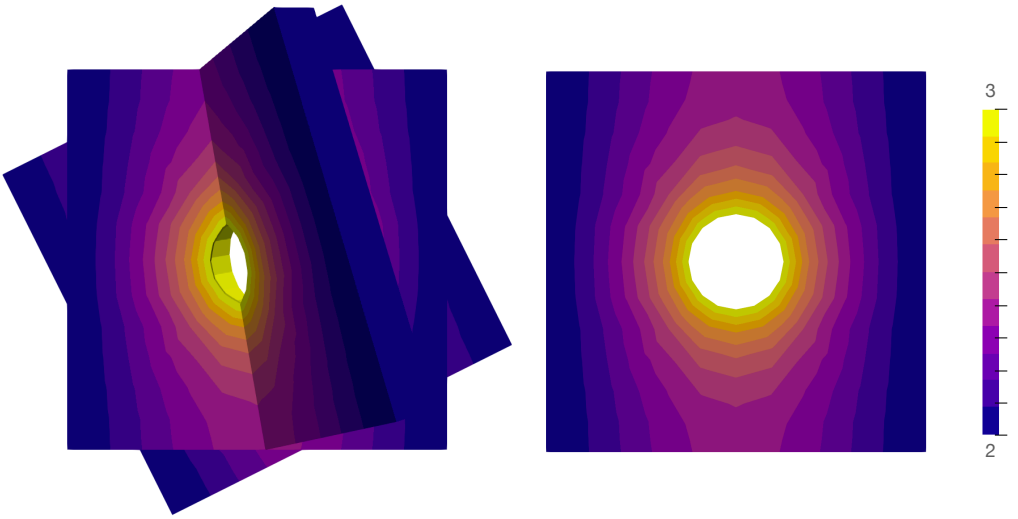


Figure 6: Results for the tensor problems, where the magnitude of $\boldsymbol{T}$ is shown. On the left, the results for the three configurations are shown and on the right the results for the not-rotated geometry is shown on its own. The displayed fields are interpolated for the cell-centres to the points.

Case	L_1	L_2	L_∞
Symmetry unit (not rotated)	6.56e-10	2.93e-11	3.37e-11
Symmetry unit (rotate XY)	6.08e-10	3.33e-11	3.83e-11
Symmetry unit (rotate XYZ)	9.33e-10	5.17e-11	6.16e-11
Full domain (rotate XY)	3.07e-13	1.26e-15	6.49e-15
Full domain (rotate XYZ)	1.96e-13	8.99e-16	3.24e-15

Table 3: Tensor equation problem: L_1, L_2 and L_∞ norms of the differences between the non-rotated full domain problem and the other cases (all symmetry units, and rotated full domains)

tions, then in some cases, the full domain problem may take less CPU time to solve; however, it should be noted that the full domain in these cases has four times as many cells as the symmetry units and hence requires at least four times the CPU time per outer iteration.

6 Conclusions

This article concisely presents the cell-centred finite volume discretisation for symmetry planes, including the implicit-explicit split for segregated solution algorithms. The presented general approach is demonstrated for scalar, vector and tensor problems, where unstructured polyhedral meshes are assumed. While the results of Section 2.2 for the vector variable could have been obtained by relying on the decomposition of vector $\mathbf{v}_P$ into the normal and tangential components and requiring that the normal component at the symmetry plane and the gradient of the tangen-

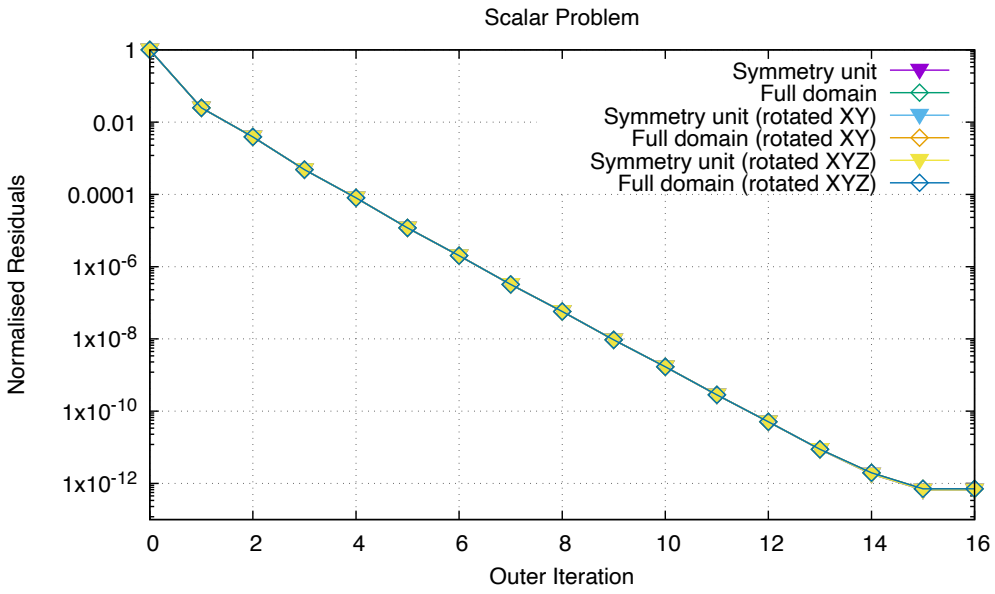


Figure 7: Residuals convergence for the scalar problems.

tial component normal to the symmetry plane is zero, such an analysis is not possible for tensor variables.

From the results, the symmetry plane discretisations are consistent with the internal face discretisation, with solution differences in the order of double-floating point precision. One motivation for using symmetry planes is to reduce the CPU time and memory requirements; however, if the symmetry planes are not aligned with the Cartesian axis directions, then the solution time may be slower when using symmetry planes. Consequently, it is recommended to align symmetry planes with Cartesian axes. Nonetheless, the use of coupled solution algorithms (e.g. [6]), where the entire diffusion term can be treated implicitly, are expected to not suffer from this limitation.

Acknowledgements

The second author gratefully acknowledges financial support from Bekaert through the University Technology Centre (UTC), from the Irish Research Council through the Laureate programme, grant number IRCLA/2017/45, and from I-Form, via a research grant from Science Foundation Ireland (SFI) under grant number 16/RC/3872 and is co-funded under the European Regional Development Fund.

References

[1] I. Demirdžić, On the discretisation of diffusion term in the finite-volume continuum mechanics, *Numerical Heat Transfer B*, vol. 68, pp. 1–10, 2015.

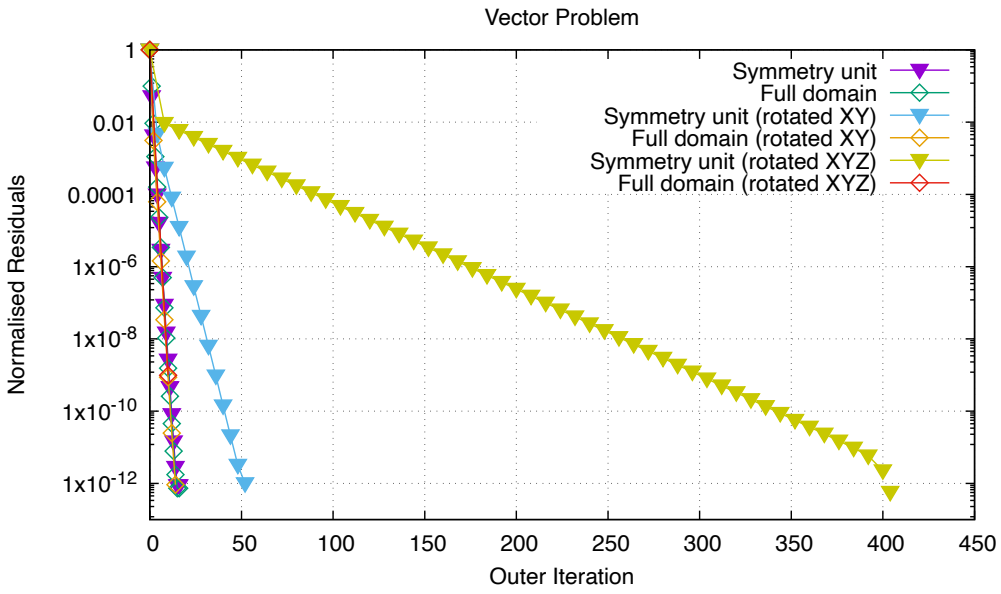


Figure 8: Residuals convergence for the vector problems.

[2] F.H. Harlow, J.E. Welch, Numerical calculation of the time-dependent viscous incompressible flow of fluid with a free surface, *Phys. Fluids*, vol. 8, pp. 2182–2189, 1965.

[3] P.J. Oliveira, Computer modelling of multidimensional multiphase flow and application to T-junctions, PhD Thesis, Imperial College, University of London, 1992.

[4] F. Moukalled, L. Mangani, M. Darwish, Implementation of boundary conditions in the finite-volume pressure-based method - Part I: Segregated solvers, Numerical Heat Transfer, Part B: Fundamentals, 69:6, 534-562, 2016.

[5] S.V. Patankar, Numerical heat transfer and fluid flow, Hemisphere Publishing Co., 1980.

[6] P. Cardiff P, Ž. Tuković, H. Jasak, A. Ivanković, A block- coupled finite volume methodology for linear elasticity and unstructured meshes, *Comput Struct*, vol. 17, pp. 100–122, 2016.

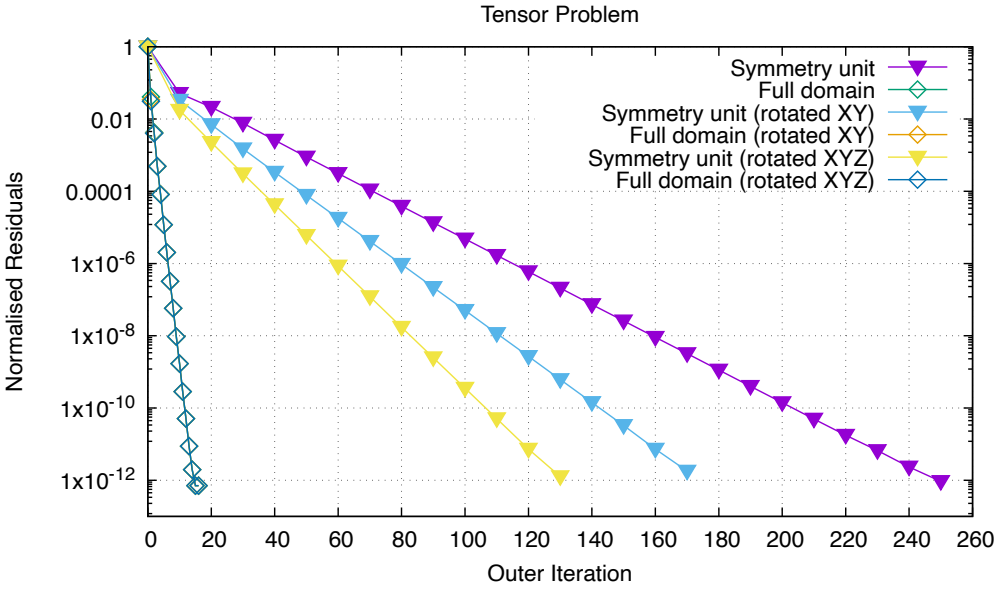


Figure 9: Residuals convergence for the tensor problems.