INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2006; **52**:723–748
Published online 24 February 2006 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/fld.1199

An efficient second-order accurate cut-cell method for solving the variable coefficient Poisson equation with jump conditions on irregular domains

Hua Ji[1], Fue-Sang Lien[2,*,†] and Eugene Yee[3]

[1]*Waterloo CFD Engineering Consulting Inc., Waterloo, Ont., Canada N2T 2N7*
[2]*Department of Mechanical Engineering, University of Waterloo, Waterloo, Ont., Canada N2L 3G1*
[3]*Defence R&D Canada—Suffield, P.O. Box 4000, Medicine Hat, Alta., Canada T1A 8K6*

SUMMARY

A numerical method is presented for solving the variable coefficient Poisson equation on a two-dimensional domain in the presence of irregular interfaces across which both the variable coefficients and the solution itself may be discontinuous. The approach involves using piecewise cubic splines to represent the irregular interface, and applying this representation to calculate the volume and area of each cut cell. The fluxes across the cut-cell faces and the interface faces are evaluated using a second-order accurate scheme. The deferred correction approach is used, resulting in a computational stencil for the discretized Poisson equation on an irregular (complex) domain that is identical to that obtained on a regular (simple) domain. In consequence, a highly efficient multigrid solver based on the additive correction multigrid (ACM) method can be applied to solve the current discretized equation system. Several test cases (for which exact solutions to the variable coefficient Poisson equation with and without jump conditions are known) have been used to evaluate the new methodology for discretization on an irregular domain. The numerical solutions show that the new algorithm is second-order accurate as claimed, even in the presence of jump conditions across an interface. Copyright © 2006 John Wiley & Sons, Ltd.

KEY WORDS: cut-cell method; second-order accuracy; Poisson equation; irregular boundary; jump conditions; additive correction multigrid (ACM)

1. INTRODUCTION

Many flow simulations involve complex geometries with curved and planar boundaries which are not aligned with the grid line orientation. In a Cartesian coordinate system, such bound-

*Correspondence to: F.-S. Lien, Department of Mechanical Engineering, University of Waterloo, Waterloo, Ont., Canada N2L 3G1.
†E-mail: fslien@uwaterloo.ca

Contract/grant sponsor: Chemical Biological Radiological Nuclear Research and Technology Initiative (CRTI); contract/grant number: CRTI-02-0093RD

Received 16 February 2005
Accepted 23 December 2005

aries are generally approximated as a series of staircase steps (viz., the arbitrary curved boundary is approximated by a domain whose boundary is defined by a set of boundary edges lying on the grid lines). Unless a very fine grid is used, the predicted results adjacent to these (complex) boundaries will be inaccurate. To overcome this problem, various investigators have proposed using other types of grids in order to provide a more accurate resolution of the boundaries corresponding to complex geometries, which have included overlapping orthogonal grids (Chimera grids), curvilinear boundary-fitted grids and unstructured grids. While these approaches simplify the implementation of boundary conditions, each introduces additional difficulties such as extra terms in the governing equations, extra interpolation, large computational stencils, and problems associated with the transfer of information across grid interfaces. This added complexity makes code development more difficult and increases computation time. Discussion of these techniques can be found in Reference [1].

As an alternative approach, the cut-cell method has become increasingly popular in recent years. This method uses a Cartesian grid for all cells except those which are intersected by the complex boundary. These boundary cells (or cut cells) are truncated so that they conform to the shape of the (curved) boundary surfaces. In this way, the advantage of using a conventional Cartesian grid method is retained for the interior cells and a more elaborate treatment is only required for the boundary cells. For example, Udaykumar *et al.* [2, 3] have described their 'ELAFINT' method for solving two-dimensional incompressible fluid flow problems in the presence of irregular stationary and moving boundaries.

Other Cartesian-grid-based methods with application to complex geometries include the immersed boundary method (IBM) [4], the immersed interface method (IIM) [5], the embedded boundary method (EBM) [6] and the ghost fluid method (GFM) [7]. IBM represents the body boundary in the flow field through a forcing term (or a feedback function) that is added to the momentum equations. These forcing terms are evaluated initially at the discrete surface points, and satisfy the no-slip boundary conditions on the surface. Subsequently, a first-order cosine function, which can be interpreted as a discrete δ-function, is used to interpolate and extrapolate information between the immersed boundary and the background grid. The use of the cosine-function formulation smears out the solutions over a thin finite band centred on the boundary, which in general could have an adverse effect on the solution accuracy. Moreover, IBM may induce spurious oscillations and consequently restrict the computational time step, especially when an explicit time-integration method is used in the flow solver. To overcome this difficulty, other IBM variants, such as the ghost-cell immersed boundary method (GCIBM) [8], have been proposed. In contrast to IBM, GCIBM uses ghost cells within the solid objects as boundary conditions without having to explicitly introduce a forcing term into the momentum equation. The ghost cells are reconstructed using either linear or quadratic interpolation of property values at the surrounding fluid nodes in the physical domain and at a boundary node.

IBM and its variants described above frequently do not allow the continuity or discontinuity of the solution to be properly imposed along the boundary curve. In other words, the numerical solution is continuous along the boundary even when the actual physical boundary conditions imply that the solution should be discontinuous, such as for free surface problems (e.g. open channel flows, flows around ships, etc.). Ideally, a physically discontinuous boundary should be represented as a numerically discontinuous boundary in order to ensure consistency between the physical problem and the numerical solution. To this

purpose, IIM [5] is designed to maintain the appropriate jump conditions along a boundary curve where the solution is discontinuous, in contrast to the numerical smearing introduced by the δ-function formulation of IBM. However, to maintain the required jump conditions along a boundary curve, IIM incorporates these boundary conditions into the computational stencil associated with the finite-difference discretization of the governing equations in a rather complicated manner. Unfortunately, this results in a rather complex algorithm with the result that this algorithm has only been extended to three dimensions for the simple case of a stationary boundary [9] (viz., the method has not yet been extended to treat the more complex case of a three-dimensional moving boundary).

EBM is a second-order accurate method for solving the Poisson equation on irregular two-dimensional domains. In this approach, the irregular boundary in the two-dimensional domain is described using a series of piecewise linear segments. This piecewise linear approximation of the boundary is reminiscent of how the interface front is reconstructed in the volume-of-fluid (VOF) method [10]. The key assumption in EBM is that the solution can be extended smoothly outside the physical domain into a fictitious domain. Similar to the cut-cell methods, fluxes on partial cells (including the interface flux) require special attention. For example, a flux on a partial cell is evaluated based on a linear interpolation between adjacent fluxes, involving nodal values in both the physical and fictitious domains. In contrast, the interface flux is obtained by using a quadratic polynomial based on the surrounding nodal values in the physical domain only. In their application of EBM, Johansen and Colella [6] only considered Dirichlet boundary conditions and did not extend the method to treat jump conditions at an interface.

GFM has been used to solve the variable coefficient Poisson equation in the presence of an interface with jump conditions [11]. GFM uses a ghost cell and assumes that the solution can be smoothly extended beyond the interface to the ghost cell. By carefully constructing ghost cells implicitly satisfying jump conditions on both sides of the interface, the order of accuracy for a first-order derivative discretized with a stencil consisting of the interface is greatly improved. However, Liu *et al.* [11] show that GFM is only first-order accurate when treating problems with jump conditions. Recently, higher-order accuracy has been obtained with GFM [12] for solution of the variable coefficient Poisson equation without jump conditions. In the study of Lui *et al.* [11], it should be mentioned that only Dirichlet boundary conditions on the irregular interface were considered.

In this paper, we present a novel second-order accurate cut-cell method based on a finite volume discretization for solving the variable coefficient Poisson equation in the presence of irregular interfaces where both the variable coefficient and the solution itself may be discontinuous. The irregular interfaces are represented through marker points. Since our final goal is to develop a second-order accurate cut-cell method capable of solving two-fluid free-surface flow problems, the marker points in this study are connected by a piecewise cubic interpolant (or cubic spline), rather than a piecewise linear interpolant (e.g. Reference [6]), so that the curvature of a free surface and, hence, the associated surface tension can be accurately represented [13]. The same interpolant is used to compute the volume, area and the unit normal on each midpoint of the interface for each cut cell. The boundary conditions in the present cut-cell approach are imposed directly on the boundaries in order to avoid interpolation errors associated with jump conditions as when GFM is used. To maintain second-order numerical accuracy, gradient fluxes through all the faces, including those on the cut cells, have to be evaluated at the face centroids.

Instead of using a polynomial interpolating function to evaluate the fluxes [6], the corrections in our proposed method are chosen to ensure that the computational stencil in the implicit part of discretized Poisson equation is the same for both regular and irregular cells. This is clearly advantageous in the sense that we can now apply directly (and, without any modification) computationally efficient iterative methods that have been developed previously to solve Poisson's equation on regular domains to our current problem. To accomplish this, the cell centroid gradient of the solution in the correction formulation is evaluated using the least-squares technique [14]. The corrections of the gradient fluxes of the solution on the Cartesian faces and the flux at the midpoint of the boundary for each cut-cell are incorporated into a source term (the so-called deferred correction approach). This allows the discretized form of the Poisson equation over an irregular domain to have the same form as that obtained over a regular domain. The additive correction multigrid (ACM) method [15, 16] is applied to solve the resulting set of equations arising from the discretization of the Poisson equation over the irregular domain. The advantage of this multigrid method is that the discretized equation on the coarser level can be obtained directly from the associated equation on the finer level without a reconstruction of the irregular boundaries on the coarser level. This advantage is particularly useful for the cut-cell method because we do not have to estimate the volume and cell face areas for the cut-cells at the coarser resolution.

This organization of the paper is as follows. In Section 2, we describe our proposed methodology for constructing an efficient second-order accurate cut-cell method for solving the variable coefficient Poisson equation with jump conditions on an irregular domain in detail. This includes descriptions of the representation of the irregular interface, the calculation of the volume and area of the cut cell, and implementation of our proposed second-order accurate discretization for the Poisson equation on the cut cells. Furthermore, ACM is described briefly in Section 2 as well. In Section 3, we present six different numerical test cases of our methodology for the Poisson equation with and without the jump conditions applied on the irregular domain in order to demonstrate the accuracy of our discretization scheme. Finally, Section 4 contains our conclusions.

2. NUMERICAL METHODS

2.1. Interface representation

The Cartesian grid method is depicted in Figure 1. Since the interface between the two phases can be arbitrary and the Cartesian grid does not have to conform to the interface, the irregular interface is represented using an ordered list of marker points (x_i, y_i), $1 \leqslant i \leqslant N$, where N is the number of markers. A list of connected polynomials $f_i(s)$, $2 \leqslant i \leqslant N$, ($f$ can be the coordinate functions x or y) is constructed using the marker points. This gives a parametric representation of the irregular interface, with the parameter s being the arc length along the interface curve. The arc length along the interface curve from the reference location ($s = 0$) to the ith marker point can be approximated as

$$s_i = \sum_{j=1}^{i-1} [(x_{j+1} - x_j)^2 + (y_{j+1} - y_j)^2]^{1/2} \tag{1}$$

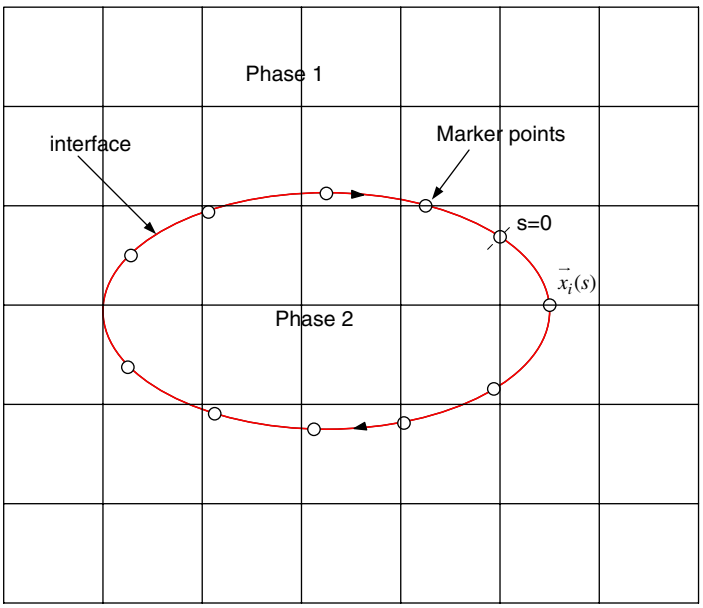

Figure 1. Schematic of a Cartesian grid and the irregular interface representation.

It is assumed that the phase 1 region always lies on the left side, and the phase 2 region lies on the right side when one traverses the interface curve in the clockwise direction, as illustrated in Figure 1.

We use a piecewise cubic interpolant (or cubic spline) [17] to provide a smooth and efficient approximation for the interface curve. In particular, on each interval of the interface curve between marker points some piecewise cubic polynomial $f_i(s)$ is chosen so that it agrees with the values of the coordinate functions ($x(s)$ or $y(s)$) at the marker points defining the interval and is continuous and has a continuous first derivative on the interval. If it is assumed that the second derivative of the ith cubic polynomial piece $f_i(s)$ $(2 \leqslant i \leqslant N)$ is linear, we can write

$$f_i''(s) = f''(s_i)\frac{s_{i+1} - s}{s_{i+1} - s_i} + f''(s_{i+1})\frac{s - s_i}{s_{i+1} - s_i} \tag{2}$$

in which f'' at each marker location can be obtained by solving the following set of linear algebraic equations:

$$\frac{\Delta_{i-1}}{6} f''(s_{i-1}) + \frac{(\Delta_{i-1} + \Delta_i)}{6} f''(s_i) + \frac{\Delta_i}{3} f''(s_{i+1}) = \frac{f(s_{i+1}) - f(s_i)}{\Delta_i} - \frac{f(s_i) - f(s_{i-1})}{\Delta_{i-1}} \tag{3}$$

where $\Delta_i \equiv (s_{i+1} - s_i)$ for $2 \leqslant i \leqslant N$. For $i = 1$ or N, continuity of f' is not applicable, so Equation (3) represents only $N - 2$ equations for the N unknown second derivatives, and two

further equations are needed. These must be based on assumptions or approximations applied at the endpoints. For example, for a periodic curve, we can set

$$f''(s_1) = f''(s_{N-1}), \qquad f''(s_N) = f''(s_2) \tag{4}$$

to get two additional equations.

The resulting piecewise cubic polynomial equation can then be rewritten as

$$f_i(s) = a_f s^3 + b_f s^2 + c_f s + d_f \tag{5}$$

where

$$\begin{aligned}
a_f &= \frac{f''(s_{i+1}) - f''(s_i)}{6\Delta_i} \\
b_f &= \frac{s_{i+1} f''(s_i) - s_i f''(s_{i+1})}{2\Delta_i} \\
c_f &= \frac{f(s_{i+1}) - f(s_i)}{\Delta_i} + \frac{\Delta_i [f''(s_i) - f''(s_{i+1})]}{6} + \frac{[s_i^2 f''(s_{i+1}) - s_{i+1}^2 f''(s_i)]}{2\Delta_i} \\
d_f &= \frac{s_{i+1} f(s_i) - s_i f(s_{i+1})}{\Delta_i} + \frac{\Delta_i [s_i f''(s_{i+1}) - s_{i+1} f''(s_i)]}{6} \\
&\quad + \frac{s_{i+1}^3 f''(s_i) - s_i^3 f''(s_{i+1})}{6\Delta_i}
\end{aligned} \tag{6}$$

To compute the volume and surface areas of the Cartesian control volumes which are required for the discretization of the Poisson equation with the finite volume method, the coordinates of the intersection points between the irregular interface and the Cartesian grids have to be determined first. Let us denote by Ω^+ the volume of the Cartesian cell that lies within the phase 1 region. In consequence, Ω^+ equals $\Delta x \Delta y$ for Cartesian cells that lie completely within the phase 1 region, where Δx and Δy are the grid spacings in the x and y directions, respectively. Ω^+ equals 0 for those cells that lie completely outside the phase 1 region. The remaining problem is to determine appropriate values of Ω^+ for those cells that have one or more Cartesian faces that intersect the interface (i.e. cut cells). Once the volume Ω^+ is known, the volume of a cell in the phase 2 region, denoted Ω^-, is simply $\Delta x \Delta y - \Omega^+$. Similarly, let us denote by l^+ the (surface) area of a Cartesian cell face that lies within the phase 1 region. In view of this, l^+ equals Δx or Δy for a Cartesian cell face that lies completely within the phase 1 region, and 0 for a cell face that lies completely outside the phase 1 region. Furthermore, an appropriate value of the area l^+ for a cell face that intersects the interface curve will need to be determined. Once the area l^+ is known, the area of a Cartesian cell face in the phase 2 region, denoted l^-, is simply Δx (or Δy) $- l^+$.

With the piecewise cubic polynomial equation of (5), the coordinates of the intersection points of the interface curve with the cell faces can be computed using a combination of the bisection and Newton–Raphson methods [17]. The computation of the volume can be reduced to the computation of a circulation along the cut lines between the Cartesian cell and the interface curve, as shown in Figure 2. If P and Q are two (arbitrary) functions of the

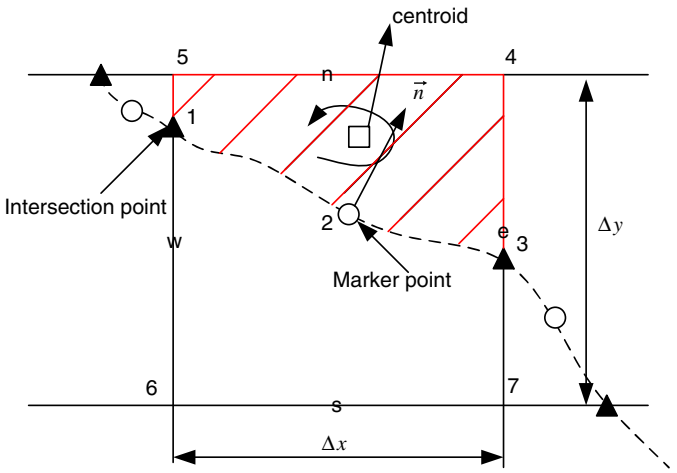

Figure 2. Computation of the volume fraction in a cut cell.

coordinates (x, y) and if we apply Stokes' theorem to the phase 1 region, we can write

$$\oint_{S^+} P\,\mathrm{d}x + Q\,\mathrm{d}y = \int_{V^+} \left(\frac{\partial Q}{\partial x} - \frac{\partial P}{\partial y}\right) \mathrm{d}x\,\mathrm{d}y \tag{7}$$

where S^+ is a simple closed curve enclosing domain V^+ of the plane in the phase 1 region. Now, if we choose $P = 0$ and $Q = x$, then

$$\oint_{S^+} x\,\mathrm{d}y = \int_{V^+} \mathrm{d}x\,\mathrm{d}y = \Omega^+ \tag{8}$$

Using Equation (8) and referring to Figure 2, the line integral on the left-hand side can be written as

$$\oint_{S^+} x\,\mathrm{d}y = \int_1^2 x\,\mathrm{d}y + \int_2^3 x\,\mathrm{d}y + \int_3^4 x\,\mathrm{d}y + \int_4^5 x\,\mathrm{d}y + \int_5^1 x\,\mathrm{d}y \tag{9}$$

The line integrals along the horizontal and vertical lines (i.e. $\overline{34}$, $\overline{45}$, $\overline{51}$) can be simply computed based on the coordinates of the end points of each line. The other integrals along the interface curve (i.e. $\overline{12}$, $\overline{23}$) can be computed with the constructed cubic polynomials described by Equation (5). To this purpose, note that the cubic polynomials for the x and y coordinate functions can be expressed explicitly as

$$x(s) = a_x s^3 + b_x s^2 + c_x s + d_x, \qquad y(s) = a_y s^3 + b_y s^2 + c_y s + d_y \tag{10}$$

Then, the contribution of the piecewise interface curve $\overline{12}$ to the line integral is given by

$$\int_{y_1}^{y_2} x(s)\,\mathrm{d}y(s) = \int_{s_1}^{s_2} x(s)y'(s)\,\mathrm{d}s = I(s_2) - I(s_1) \tag{11}$$

where

$$I(s) = \tfrac{1}{2}a_x a_y s^6 + \tfrac{1}{5}(3b_x a_y + 2a_x b_y)s^5 + \tfrac{1}{4}(a_x c_y + 2b_x b_y + 3c_x a_y)s^4 + \tfrac{1}{3}(b_x c_y + 2c_x b_y + 3d_x a_y)s^3 + \tfrac{1}{2}(c_x c_y + 2d_x b_y)s^2 + d_x c_y s \tag{12}$$

Here, a_x, a_y, b_x, b_y, c_x, c_y, d_x and d_y are defined as in Equation (6). The line integral along the piecewise curve $\overline{23}$ can be obtained similarly as in Equation (11). The sum of line integrals along each segment then gives us the volume of the Cartesian cut-cell that lies within the phase 1 region. The corresponding volume in the phase 2 region is then given simply by $\Omega^- = (\Delta x \Delta y - \Omega^+)$.

In terms of the definition, the four areas of the faces of the cut-cell in the phase 1 region (see Figure 2) can be obtained as

$$l_{\mathrm{e}}^+ = \Delta y_{\overline{34}}, \qquad l_{\mathrm{w}}^+ = \Delta y_{\overline{51}}, \qquad l_{\mathrm{n}}^+ = \Delta x, \qquad l_{\mathrm{s}}^+ = 0 \tag{13}$$

where l_{e}^+, l_{w}^+, l_{n}^+ and l_{s}^+ are the areas of the east, west, north and south faces of the cut-cell in the phase 1 region, respectively; $\Delta y_{\overline{34}}$, and $\Delta y_{\overline{51}}$ are the length of the line segments $\overline{34}$ and $\overline{51}$. The corresponding four areas of the faces of the same cut-cell in the phase 2 region are then

$$l_{\mathrm{e}}^- = \Delta y - \Delta y_{\overline{34}} = \Delta y_{\overline{73}}, \qquad l_{\mathrm{w}}^- = \Delta y - \Delta y_{\overline{51}} = \Delta y_{\overline{16}}, \qquad l_{\mathrm{n}}^- = 0, \qquad l_{\mathrm{s}}^- = \Delta x \tag{14}$$

To compute the interface area $l_{\overline{13}}^+$ of the cut-cell, we divide the curve $\overline{13}$ into two line segments $\overline{12}$ and $\overline{23}$ (viz., $\overline{13} = \overline{12} \cup \overline{23}$). From the coordinates of the intersection points 1, 3 between the cell face and the interface curve and the marker point 2, we can compute the interface area $l_{\overline{13}}^+$ as $l_{\overline{13}}^+ = l_{\overline{12}}^+ + l_{\overline{23}}^+$, which is also equal to the interface area $l_{\overline{13}}^-$ in the phase 2 region.

Based on the definition, the coordinates of the centroid $(x_{\mathrm{c}}^+, y_{\mathrm{c}}^+)$ of each cut-cell in the phase 1 region can be written in

$$x_{\mathrm{c}}^+ = \frac{\int_{V^+} x \,\mathrm{d}x\,\mathrm{d}y}{\int_{V^+} \mathrm{d}x\,\mathrm{d}y}, \qquad y_{\mathrm{c}}^+ = \frac{\int_{V^+} y \,\mathrm{d}x\,\mathrm{d}y}{\int_{V^+} \mathrm{d}x\,\mathrm{d}y} \tag{15}$$

where V^+ is the volume of the Cartesian cell in the phase 1 region. For the evaluation of the cell centroid in Equation (15), it is useful to apply Stokes' theorem with $Q = \frac{1}{2}x^2$, $P = 0$ for computation of x_{c}^+, and $P = -\frac{1}{2}y^2$, $Q = 0$ for computation of y_{c}^+, so

$$x_{\mathrm{c}}^+ = \frac{\oint_{S^+} \frac{1}{2}x^2 \,\mathrm{d}y}{\oint_{S^+} x \,\mathrm{d}y}, \qquad y_{\mathrm{c}}^+ = -\frac{\oint_{S^+} \frac{1}{2}y^2 \,\mathrm{d}x}{\oint_{S^+} x \,\mathrm{d}y} \tag{16}$$

The surface integral in Equation (16) is along the curve $\overline{123451}$. The coordinates of the centroid $(x_{\mathrm{c}}^-, y_{\mathrm{c}}^-)$ of each cut-cell in the phase 2 region can be computed with Equation (16) as well, but the surface integral is now along the curve $\overline{673216}$.

Based on the polynomial representation of the interface curve, the normal direction of any point on the curve can be evaluated using the following analytical formula:

$$n_x = \frac{-y'(s)}{\sqrt{x'(s)^2 + y'(s)^2}}, \qquad n_y = \frac{x'(s)}{\sqrt{x'(s)^2 + y'(s)^2}} \tag{17}$$

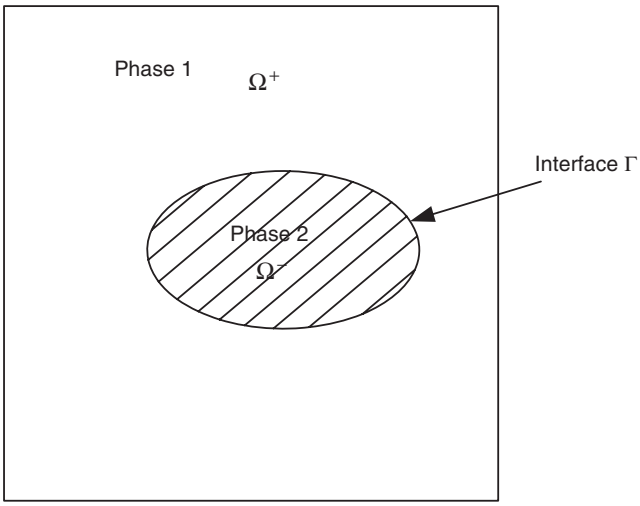

Figure 3. A two-dimensional computational domain includes two phases that are divided by an irregular interface.

where n_x and n_y are the x and y components of the unit normal vector to the interface curve. The normal direction points from the phase 2 region towards the phase 1 region, as shown in Figure 2.

2.2. *Numerical discretization*

Consider a two-dimensional Cartesian computational domain, Ω, with the exterior boundary, $\partial\Omega$, and a one-dimensional irregular interface Γ, that divides the computational domain into two adjacent subdomains, Ω^+ and Ω^-, as shown in Figure 3. The variable coefficient Poisson equation can be written as

$$\nabla \cdot (\beta(x, y)\nabla\phi(x, y)) = f(x, y) \tag{18}$$

where $\nabla \equiv (\partial/\partial x, \partial/\partial y)$ is the gradient operator, and $\beta(x, y)$ is assumed continuous in each subdomain, Ω^+ and Ω^-, but may be discontinuous across the interface Γ. On $\partial\Omega$, either Dirichlet boundary conditions ($\phi(x, y) = g(x, y)$) or Neumann boundary conditions ($\phi_n(x, y) = h(x, y)$) can be specified. Here, $\phi_n \equiv \nabla\phi \cdot \mathbf{n}$ and $\mathbf{n}$ is the unit normal vector to the interface. The jump conditions across the interface are specified by

$$[\phi]_\Gamma = \phi_\Gamma^+ - \phi_\Gamma^- = A(x, y) \tag{19}$$

and

$$[\beta\phi_n]_\Gamma = \beta_\Gamma^+(\nabla\phi \cdot \mathbf{n})_\Gamma^+ - \beta_\Gamma^-(\nabla\phi \cdot \mathbf{n})_\Gamma^- = B(x, y) \tag{20}$$

where the superscripts $+$, $-$ refer to the phase 1 and phase 2 property values, respectively. Hence, ϕ_Γ^+ is the value of the variable ϕ adjacent to the interface on the phase 1 side, and ϕ_Γ^- is the value of the variable ϕ adjacent to the interface on the phase 2 side.

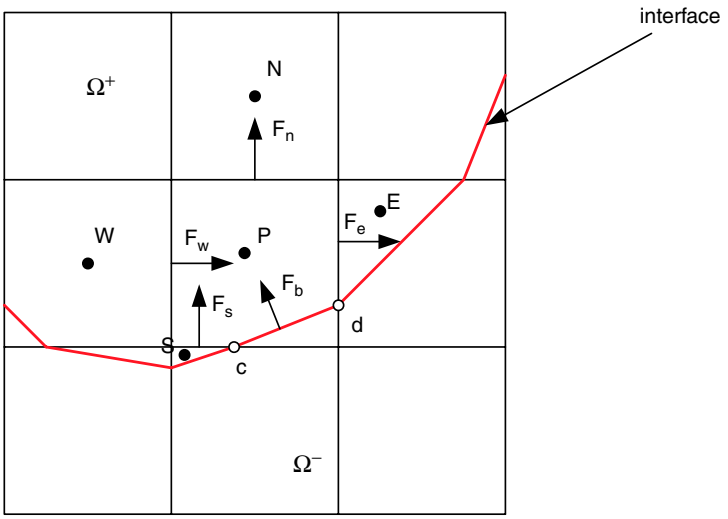

Figure 4. Discretization of the Poisson equation in the phase 1 region in the irregular grid system.

Applying Stokes' theorem, the integral form of the Poisson equation (18) can be expressed as

$$\oint_S \beta \nabla \phi \cdot \mathbf{n}\, \mathrm{d}s = \int_V f\, \mathrm{d}V \tag{21}$$

where V is the control domain, S is the control surface, and $\mathbf{n}$ is the unit normal direction to the control surface S. Equation (21) applies to both phases. For sake of brevity, only the discretization of the Poisson equation in the phase 1 region will be described in detail below (the discretization in the phase 2 region can be obtained in an analogous manner). In the irregular Cartesian grid system (i.e. Cartesian grid over an irregular domain) shown in Figure 4, the Poisson equation (21) can be discretized as follows (on using the midpoint rule):

$$F_{\mathrm{e}} - F_{\mathrm{w}} + F_{\mathrm{n}} - F_{\mathrm{s}} - F_b^+ = \int_{V^+} f\, \mathrm{d}V \tag{22}$$

where F_{e}, F_{w}, F_{n} and F_{s} are the gradient fluxes of ϕ through the east, west, north and south faces, respectively, and F_b^+ is the gradient flux of ϕ through the piece (section) $\overline{cd}$ of the interface curve that intersects the cell P in the phase 1 region. It should be noted that the gradient flux F_b^- of ϕ through the same piece of the interface $\overline{cd}$ in the phase 2 region can be different from F_b^+ due to the jump condition of Equation (20) across the interface. The evaluation of F_b^+ and F_b^- will be addressed later in this section.

To achieve second-order accuracy in the discretization, we will use the values at the auxiliary nodes P' and E' to approximate the gradient flux of ϕ on the east face (i.e. F_{e}). As shown in Figure 5, P' lies at the intersection of the normal to the east (or west) face through the east face midpoint e' and the normal to the north (or south) face through cell centroid P

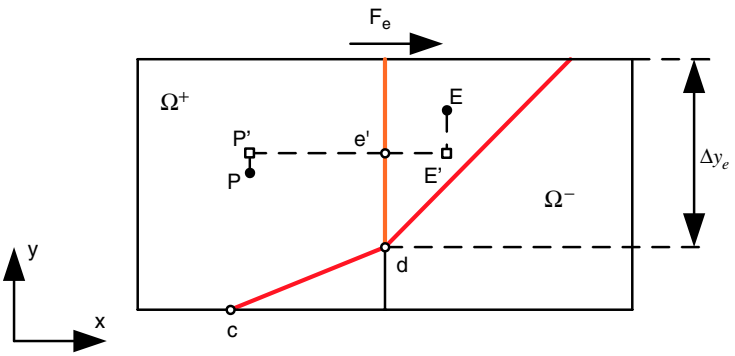

Figure 5. Approximation of the gradient flux F_e of ϕ using two property values at the auxiliary nodes P' and E'.

(with a similar definition for E'). Hence, the gradient flux F_e of ϕ can be expressed as

$$F_e = \beta_{e'} \frac{\phi_{E'} - \phi_{P'}}{x_{E'} - x_{P'}} \Delta y_e \tag{23}$$

It should be noted that Δy_e in Equation (23) is the area on the east face of the cut cell, not the grid spacing in this case. Furthermore, $\beta_{e'}$ corresponds to the value of the variable coefficient function $\beta(x, y)$ evaluated at the midpoint e' of the east face.

The values of ϕ at the auxiliary nodes P' and E' are evaluated based on the gradient of ϕ at the cell centroids P and E, that is

$$\phi_{P'} = \phi_P + \left(\frac{\partial \phi}{\partial y}\right)_P (y_{P'} - y_P) \tag{24}$$

and

$$\phi_{E'} = \phi_E + \left(\frac{\partial \phi}{\partial y}\right)_E (y_{E'} - y_E) \tag{25}$$

Substituting Equations (24) and (25) into the expression for the gradient flux F_e of Equation (23), we obtain

$$F_e = \beta_{e'} \left(\frac{\phi_E - \phi_P}{x_E - x_P}\right) \Delta y_e + \frac{\beta_{e'} \Delta y_e}{x_E - x_P} \left[\left(\frac{\partial \phi}{\partial y}\right)_E (y_{e'} - y_E) - \left(\frac{\partial \phi}{\partial y}\right)_P (y_{e'} - y_P)\right] \tag{26}$$

In derivation of Equation (26), we used the fact that $x_{E'} = x_E$, $x_{P'} = x_P$ and $y_{e'} = y_{E'} = y_{P'}$. Other gradient fluxes through the west, north and south faces can be evaluated similarly to Equation (26). To ensure that the coefficients of the discretized Poisson equation for the cut-cell here are the same as those for a regular cell, we will incorporate the second term of Equation (26) into the source term. The gradient flux of ϕ through the interface $\overline{cd}$, $F_b^+ [\equiv (\beta(\partial \phi / \partial n))_b^+ l_{\overline{cd}}]$, is also incorporated into the source term. In so doing, the discretized Poisson equation on an irregular Cartesian grid system has the same form as the corresponding

equation on a regular Cartesian grid system; viz.,

$$A_E\phi_E + A_W\phi_W + A_N\phi_N + A_S\phi_S + A_P\phi_P = \text{RHS} \tag{27}$$

where

$$\begin{aligned} A_W &= \frac{\beta_{w'}\Delta y_{\mathrm{w}}}{x_P - x_W} \\ A_E &= \frac{\beta_{e'}\Delta y_{\mathrm{e}}}{x_E - x_P} \\ A_N &= \frac{\beta_{n'}\Delta x_{\mathrm{n}}}{y_N - y_P} \\ A_S &= \frac{\beta_{s'}\Delta x_{\mathrm{s}}}{y_P - y_S} \\ A_P &= -(A_W + A_E + A_N + A_S) \end{aligned} \tag{28}$$

The RHS (source) term in Equation (27) is given by

$$\begin{aligned} \text{RHS} = &-\frac{\beta_{e'}\Delta y_{\mathrm{e}}}{x_E - x_P}\left[\left(\frac{\partial\phi}{\partial y}\right)_E (y_{e'} - y_E) - \left(\frac{\partial\phi}{\partial y}\right)_P (y_{e'} - y_P)\right] \\ &+ \frac{\beta_{w'}\Delta y_{\mathrm{w}}}{x_P - x_W}\left[\left(\frac{\partial\phi}{\partial y}\right)_P (y_{w'} - y_P) - \left(\frac{\partial\phi}{\partial y}\right)_W (y_{w'} - y_W)\right] \\ &- \frac{\beta_{n'}\Delta x_{\mathrm{n}}}{y_N - y_P}\left[\left(\frac{\partial\phi}{\partial x}\right)_N (x_{n'} - x_N) - \left(\frac{\partial\phi}{\partial x}\right)_P (x_{n'} - x_P)\right] \\ &+ \frac{\beta_{s'}\Delta x_{\mathrm{s}}}{y_P - y_S}\left[\left(\frac{\partial\phi}{\partial x}\right)_P (x_{s'} - x_P) - \left(\frac{\partial\phi}{\partial x}\right)_S (x_{s'} - x_S)\right] \\ &+ F_b^+ + \int_V f \,\mathrm{d}V \end{aligned} \tag{29}$$

To approximate the contribution of Equation (29) to the source term in the discretized Poisson equation, we have to evaluate the gradient of ϕ at each control volume centroid. There are two common techniques for evaluating cell property gradients: namely, using a Green–Gauss theorem [18] or applying a least-squares approach [14]. In this study, we choose the least-squares approach for the reason that it is exact for a linear profile, whereas some effort is needed to make the Green–Gauss method exact for the same profile.

To understand how the least-squares algorithm works, consider a cut-cell P and its set of immediate neighbours η_P, as illustrated in Figure 6. It should be noted that the interface face must also be considered as a neighbour. Therefore, there are three different kinds of cut-cells which include four, three and five neighbours as shown in Figures 6(a), (b) and (c), respectively. The change in value of ϕ between an immediate neighbour cell j and central

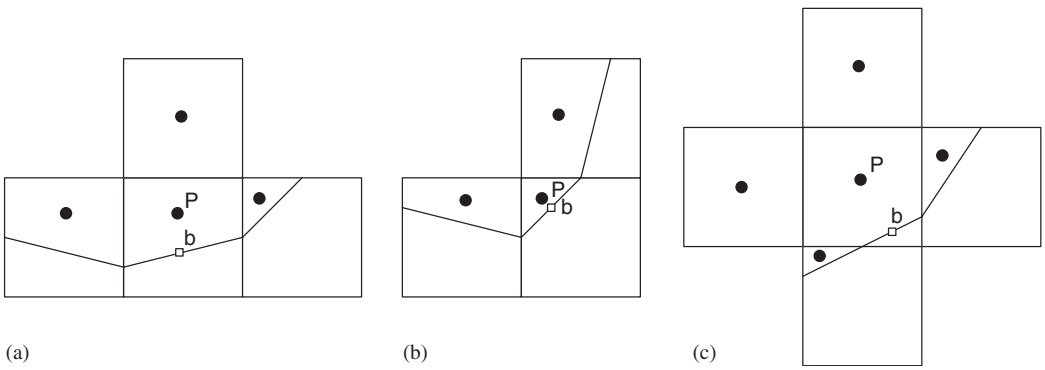

Figure 6. Least-squares stencil used in calculating cell gradient vectors of ϕ.

cell P is given by $\phi_j - \phi_P$, $j \in \eta_P$. If the cell gradient $(\nabla\phi)_P$ of ϕ at P is exact, then this difference is also

$$\phi_j - \phi_P = (\nabla\phi)_P \cdot (\mathbf{r}_j - \mathbf{r}_P) \tag{30}$$

where $\mathbf{r}_j - \mathbf{r}_P$ is the vector from cell P to cell j. But unless the solution ϕ is linear, the cell gradient cannot be exact, for cell P has more neighbours than the gradient vector has components. The least-squares gradient is that which minimizes the cost function given by

$$E = \sum_{j \in \eta_P} w_j [(\nabla\phi)_P \cdot (\mathbf{r}_j - \mathbf{r}_P) - (\phi_j - \phi_P)]^2 \tag{31}$$

where w_j are the weights. We choose $w_j = 1/|\mathbf{r}_j - \mathbf{r}_P|^2$, which places greater weight on neighbours nearer P in the stencil.

The solution to this least-squares problem requires the solution of the following matrix equation:

$$\begin{bmatrix} \sum w_j \Delta x_j \Delta x_j & \sum w_j \Delta x_j \Delta y_j \\ \sum w_j \Delta x_j \Delta y_j & \sum w_j \Delta y_j \Delta y_j \end{bmatrix} \begin{bmatrix} \phi_x \\ \phi_y \end{bmatrix} = \begin{bmatrix} \sum w_j \Delta x_j \Delta \phi_j \\ \sum w_j \Delta y_j \Delta \phi_j \end{bmatrix} \tag{32}$$

where

$$\Delta x_j \equiv x_j - x_P$$

$$\Delta y_j \equiv y_j - y_P$$

$$\Delta \phi_j \equiv \phi_j - \phi_P$$

and ϕ_x and ϕ_y are the components of the cell gradient $(\nabla\phi)_P$ of ϕ at P.

In the phase 2 region, the discretized variable coefficient Poisson equation is the same as that in the phase 1 region (i.e. Equations (27)–(29)), but with a different coefficient $\beta(x, y)$ and source $f(x, y)$. The jump conditions of Equations (19) and (20) also need be discretized to couple solutions in the two regions. The discretization of Equation (20)

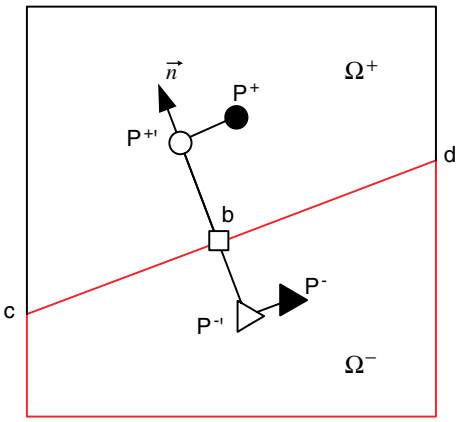

Figure 7. Calculating the gradient flux F_b at a boundary face.

requires the discretization of $(\nabla\phi\cdot\mathbf{n})^+$ and $(\nabla\phi\cdot\mathbf{n})^-$, which are also needed to evaluate F_b^+ $(\equiv\beta_b^+(\nabla\phi\cdot\mathbf{n})_b^+ l_{\overline{cd}})$ in Equation (29). Here, we use the same approach as that used to evaluate the gradient flux of ϕ through an internal cell face (i.e. Equation (26)), except for the fact that we now need to use a one-sided approximation. As shown in Figure 7, the gradient flux of ϕ, F_b^+, in the phase 1 region can be approximated by

$$F_b^+=\beta_b^+(\nabla\phi\cdot\mathbf{n})_b^+ l_{\overline{cd}}=\beta_b^+\frac{\phi_{P^{+\prime}}-\phi_b^+}{|\mathbf{r}_{P^{+\prime}}-\mathbf{r}_b|}l_{\overline{cd}} \tag{33}$$

where the subscript $P^{+\prime}$ denotes the intersection point between the line running through the midpoint b of the interface along normal vector direction $\mathbf{n}$ and the line running through the centroid P^+ perpendicular to the normal vector direction $\mathbf{n}$ in the phase 1 region. In terms of the gradient of ϕ at the cell centroid P^+, the value at the auxiliary point $P^{+\prime}$ can be expressed as

$$\phi_{P^{+\prime}}=\phi_{P^+}+(\nabla\phi)_{P^+}\cdot(\mathbf{r}_{P^{+\prime}}-\mathbf{r}_{P^+}) \tag{34}$$

Substituting Equation (34) into Equation (33), we then obtain

$$F_b^+=\beta_b^+\left[\frac{\phi_{P^+}-\phi_b^+}{|\mathbf{r}_{P^{+\prime}}-\mathbf{r}_b|}+(\nabla\phi)_{P^+}\cdot\frac{(\mathbf{r}_{P^{+\prime}}-\mathbf{r}_{P^+})}{|\mathbf{r}_{P^{+\prime}}-\mathbf{r}_b|}\right]l_{\overline{cd}} \tag{35}$$

Similarly, with reference to Figure 7, the gradient flux of ϕ through the interface $\overline{cd}, F_b^-$, in the phase 2 region can be evaluated to give

$$F_b^-=-\beta_b^-\left[\frac{\phi_{P^-}-\phi_b^-}{|\mathbf{r}_{P^{-\prime}}-\mathbf{r}_b|}+(\nabla\phi)_{P^-}\cdot\frac{(\mathbf{r}_{P^{-\prime}}-\mathbf{r}_{P^-})}{|\mathbf{r}_{P^{-\prime}}-\mathbf{r}_b|}\right]l_{\overline{cd}} \tag{36}$$

where the subscripts $P^{-\prime}$ and P^- in the phase 2 region have definitions similar to those of $P^{+\prime}$ and P^+ in the phase 1 region. With the jump conditions of Equations (19) and (20), i.e.

$$\phi_b^+-\phi_b^-=A(x,y) \tag{37}$$

$$F_b^+ - F_b^- = B(x, y) l_{\overline{cd}} \tag{38}$$

where F_b^+, F_b^- are obtained from Equations (35) and (36), we can solve Equations (37) and (38) to obtain

$$\phi_b^+ = \frac{\beta_b^+((\phi_{P^+}/dr^+) + C^+) + \beta_b^-(((A(x, y) + \phi_{p^-})/dr^-) + C^-) - B(x, y)}{(\beta_b^+/dr^+) + (\beta_b^-/dr^-)} \tag{39}$$

$$\phi_b^- = \phi_b^+ - A(x, y) \tag{40}$$

where

$$dr^+ \equiv |\mathbf{r}_{P^{+\prime}} - \mathbf{r}_b| \tag{41}$$

$$dr^- \equiv |\mathbf{r}_{P^{-\prime}} - \mathbf{r}_b| \tag{42}$$

$$C^+ \equiv (\nabla\phi)_{P^+} \cdot \frac{(\mathbf{r}_{P^{+\prime}} - \mathbf{r}_{P^+})}{dr^+} \tag{43}$$

$$C^- \equiv (\nabla\phi)_{P^-} \cdot \frac{(\mathbf{r}_{P^{-\prime}} - \mathbf{r}_{P^-})}{dr^-} \tag{44}$$

2.3. Additive correction multigrid

The discretized Poisson equation (27) can be written in the form

$$\mathscr{L}(\phi) = \mathrm{RHS} \tag{45}$$

where $\mathscr{L}$ is a linear operator. The linear system can be solved using an iterative method. In the present study, the discretized Poisson equation (27) is solved using the additive correction multigrid (ACM) [15] method. The main advantage of this method is that the coefficient matrix of the Poisson equation on the coarser level can be directly derived from the corresponding coefficient matrix on the finer level. This method is often used in an unstructured grid system. We apply the ACM method using a classical 'V-cycle' on the correction form of the linear system: namely,

$$\mathscr{L}(\phi + \delta\phi) = \mathrm{RHS} \Leftrightarrow L(\delta\phi) = R \quad \text{with } R = \mathrm{RHS} - \mathscr{L}(\phi) \tag{46}$$

With this method, the form of the coefficient matrix on the coarser level in the irregular Cartesian grid system is the same as that in the regular Cartesian grid system. For the grid hierarchy, the coarse grid spacing in each direction is twice that of the fine grid spacing.

As shown in Figure 8, P is a cell on the coarser level, and $f1$, $f2$, $f3$ and $f4$ are the four cells on the next finer level, which lie in the coarser cell P. Hence, the coefficient matrix on the coarser cell P can be derived as

$$A_P\delta\phi_P + A_E\delta\phi_E + A_W\delta\phi_W + A_N\delta\phi_N + A_S\delta\phi_S = \mathrm{RHS} \tag{47}$$

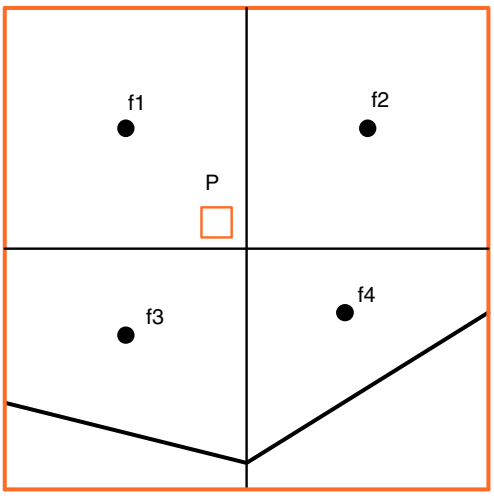

Figure 8. Construction of the coefficient matrix of the Poisson equation on the coarser grid level.

where

$$
\begin{aligned}
A_E &= (A_E)_{f2} + (A_E)_{f4} \\
A_W &= (A_W)_{f1} + (A_W)_{f3} \\
A_N &= (A_N)_{f1} + (A_N)_{f2} \\
A_S &= (A_S)_{f3} + (A_S)_{f4} \\
A_P &= (A_P)_{f1} + (A_P)_{f2} + (A_P)_{f3} + (A_P)_{f4} \\
&\quad + (A_E)_{f3} + (A_W)_{f4} + (A_N)_{f4} + (A_S)_{f2} \\
&\quad + (A_W)_{f2} + (A_E)_{f1} + (A_S)_{f1} + (A_N)_{f3} \\
\mathrm{RHS} &= \sum_{i=1}^{4} [\mathrm{RHS} - (A_P \delta\phi_P + A_E \delta\phi_E + A_W \delta\phi_W + A_N \delta\phi_N + A_S \delta\phi_S)]_{fi}
\end{aligned}
\tag{48}
$$

The values of ϕ in Equation (45) on the finest grid level are first approximated by performing a few iterations with the pointwise Gauss–Seidel method (resulting in a smooth error or residual at this grid level). The residual is then calculated on the finest grid and transferred to the next coarser level using simple averaging as the restriction operation, so

$$
R_l = \sum_{i=1}^{4} (R_{l+1})_i \tag{49}
$$

where the summation is over the four cells (i.e. daughter cells) at the finer level into which the coarser level has been divided (see Figure 8). The correction $\delta\phi$ to the solution in Equation (47) at the coarser level is then approximated by performing a few iterations with the pointwise Gauss–Seidel method. This procedure is then applied recursively down to the

coarsest level. The correction $\delta\phi$ to the solution on the coarsest grid level is then transferred back to the four daughter cells on the next finer grid level (prolongation operation). After performing a few iterations at this level, the correction to the solution on this level is transferred to the next finer level until the finest level is finally reached. The whole V-cycle is repeated until the residual on the finest level has been reduced to some desired (small) level.

In the following, we generally stop the V-cycle iterations when the maximum final residual is smaller than 10^{-10} of the maximum initial residual. We apply four iterations of the Gauss–Seidel method at each grid level, except at the coarsest level. On the coarsest grid level, we apply 10 iterations of the Gauss–Seidel method.

3. NUMERICAL RESULTS

To evaluate the numerical accuracy of the proposed method, we define the volume-weighted norm of a variable e as

$$\|e\|_p = \left[\frac{\sum_i |e_i|^p V_i}{\sum_i V_i}\right]^{1/P} \tag{50}$$

where $\sum_i$ denotes a summation over all the cells in the solution domain. Note that the L_∞-norm for e, $\|e\|_\infty$, is simply the maximum absolute value of e over all the cells in the solution domain. We can now define the rate of convergence between two norms, e_1 and e_2, of the solution obtained on two different grid spacings h_1 and h_2, as

$$r_p = \frac{\log(\|e_1\|_p/\|e_2\|_p)}{\log(h_1/h_2)} \tag{51}$$

In view of Equation (51), a value of $r_p = n$ then indicates that the underlying solution methodology possesses an nth order of accuracy.

3.1. Test case 1

The first test case considers the solution of the following Poisson equation without jump conditions:

$$\nabla^2\phi = 7r^2\cos 3\theta \tag{52}$$

with a domain of definition in the plane consisting of the region inside a unit square and outside a circle of radius $r = 0.215$ with centre at $(0.5, 0.5)$. In this case, we only consider the solution of the Poisson equation in the phase 1 region. Dirichlet boundary conditions are imposed on all the boundaries of the domain, including the interior circular boundary. For this test case, these boundary conditions are specified using the exact solution

$$\phi(r,\theta) = r^4\cos 3\theta \tag{53}$$

Figure 9 shows the contour plot of the numerical solution ϕ obtained on a 128×128 uniform grid.

To estimate the order of accuracy of our proposed solution methodology, we will solve this problem over the computational domain shown in Figure 9 for a series of uniform grids with

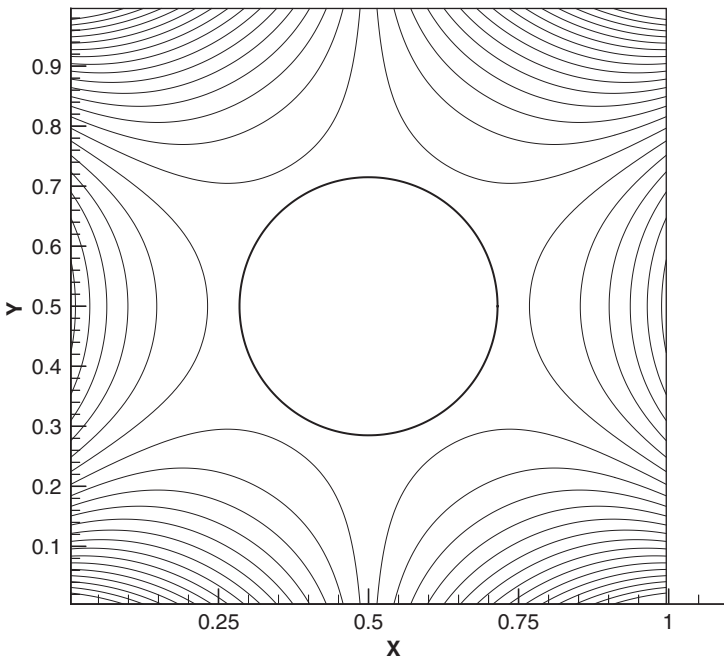

Figure 9. A contour plot of the numerical solution ϕ obtained on a 128×128 uniform grid.

Table I. Numerical accuracy tests for Poisson problem with the Dirichlet boundary conditions applied along the domain boundary.

Grids	$\|\varepsilon\|_1$	r_1	$\|\varepsilon\|_2$	r_2	$\|\varepsilon\|_\infty$	r_∞
32×32	1.75×10^{-5}		2.09×10^{-5}		5.63×10^{-5}	
64×64	4.15×10^{-6}	2.08	5.00×10^{-6}	2.06	1.53×10^{-5}	1.88
128×128	9.91×10^{-7}	2.07	1.20×10^{-6}	2.06	3.98×10^{-6}	1.94
256×256	2.37×10^{-7}	2.06	2.90×10^{-7}	2.05	1.04×10^{-6}	1.94

increasing resolution. For each grid size, the norm of the error $\|\varepsilon\|$ in the solution is calculated using the difference between the numerical solution and the exact (or, known) solution given by Equation (53). Table I summarizes the variation of the error with increasing grid resolution, together with the order of accuracy of the solution determined using Equation (51). From Table I, we can see that the numerical accuracy is the second order for all the norms shown (i.e. L_1-norm, L_2-norm, and L_∞-norm).

We can also analyse the effectiveness of the ACM algorithm in reducing the residual for this problem. Figure 10 plots the L_∞-norm of the residual,

$$\|\mathscr{L}(\phi^m) - \mathrm{RHS}\|_\infty$$

against the iteration number, m, for a calculation on a series of uniform grids of different sizes. In each case, the solution ϕ is initialized to zero. The ACM algorithm reduces the

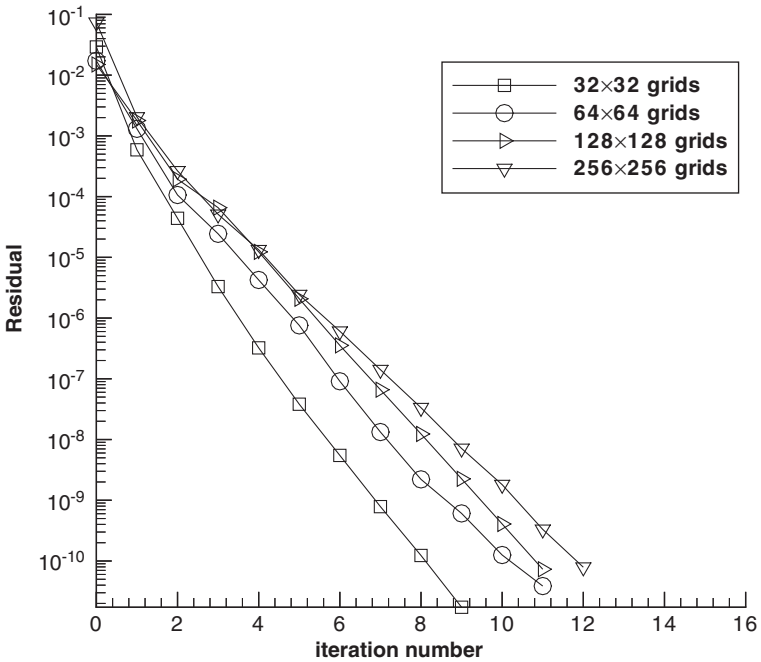

Figure 10. Variation of the residual for Poisson problem with the Dirichlet boundary conditions applied to the domain shown in Figure 9.

residual by about an order of magnitude per iteration. There is a slight decrease in performance as the grid spacing is reduced, but the reduction rates are still around 5 per ACM iteration even for the finest grids shown.

3.2. Test case 2

Test cases 2–6 are taken from Reference [11], for which GFM was used. The first reason for selecting these cases is to demonstrate that our proposed second-order numerical methodology can be applied to the Poisson equation with different interface shapes and solution functions (for example, exponential, logarithmic and parabolic functions). The second reason is to compare the numerical errors from our proposed method with those from GFM [11].

Consider $\nabla\cdot(\beta\nabla\phi)=f(x,y)$ defined within the unit square $[0,1]\times[0,1]$ with the interface defined by the circle $(x-0.5)^2+(y-0.5)^2=0.215^2$, where an outward pointing normal vector is $(n_x\mathbf{i}+n_y\mathbf{j})$. Here, $\beta=2$ and $f(x,y)=8(x^2+y^2-1)\mathrm{e}^{-x^2-y^2}$ inside the circular region; $\beta=1$ and $f(x,y)=0$ outside the circular region. The jump conditions are $[\phi]=-\mathrm{e}^{-x^2-y^2}$ and $[\beta\phi_n]=4(n_x x+n_y y)\mathrm{e}^{-x^2-y^2}$ along the circular boundary. The corresponding exact solution for this problem is $\phi(x,y)=\mathrm{e}^{-x^2-y^2}$ in the region inside the circular boundary and $\phi(x,y)=0$ in the region outside the circular boundary. Dirichlet boundary conditions are imposed on the square domain boundary. Figure 11 shows the numerical solution with 64 grid points in each direction and Table II summarizes the results of the numerical accuracy tests. In Table II,

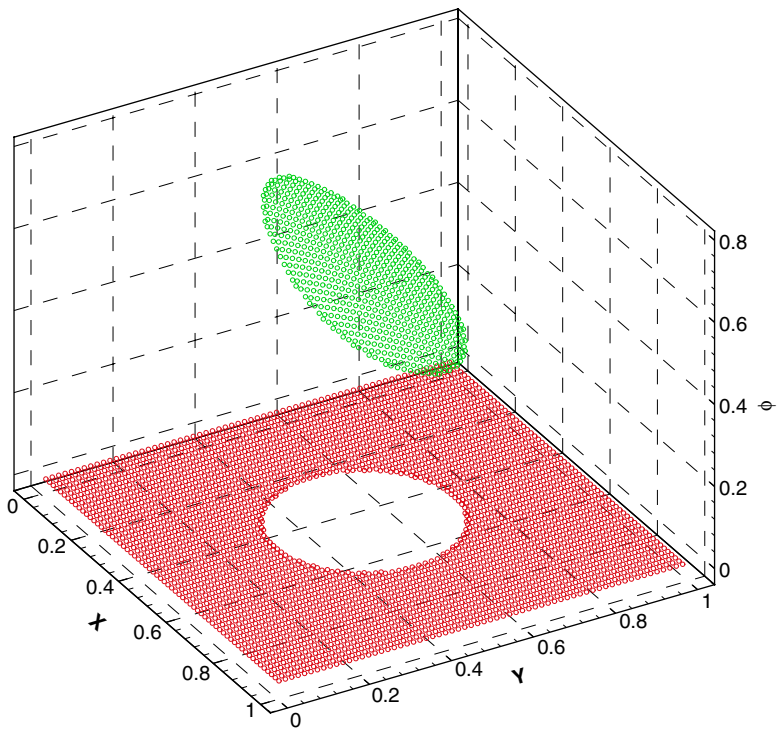

Figure 11. The numerical solution of the test case 2 with 64 grid points in each direction.

Table II. Numerical accuracy test for the case 2.

Grids	$\|\varepsilon\|_1$	r_1	$\|\varepsilon\|_2$	r_2	$\|\varepsilon\|_\infty$	r_∞
32×32	6.35×10^{-5}		7.85×10^{-5}		1.48×10^{-4}	
64×64	1.66×10^{-5}	1.98	2.00×10^{-5}	1.97	3.83×10^{-5}	1.95
128×128	4.23×10^{-6}	1.97	5.16×10^{-6}	1.95	1.08×10^{-5}	1.83
256×256	1.07×10^{-6}	1.98	1.28×10^{-6}	2.01	2.34×10^{-6}	2.20

the various norms of the error (e.g. $\|\varepsilon\|_1$, $\|\varepsilon\|_2$ and $\|\varepsilon\|_\infty$) in the solution are calculated based on the obtained numerical solution and the exact (known) solution. The results of Table II demonstrate clearly that the proposed numerical method maintains a second-order accuracy as claimed earlier.

Table III compares $\|\varepsilon\|_2$ and $\|\varepsilon\|_\infty$ obtained on a 64×64 grid using our proposed numerical methodology with those obtained on an 80×80 grid using the methodology described in Reference [11]. It can be seen that our numerical errors are generally two orders of magnitude lower than those from Reference [11].

Table III. Numerical accuracy test for the case 2 in comparison with Reference [11].

Grids	$\|\varepsilon\|_2$	$\|\varepsilon\|_\infty$
64×64	2.00×10^{-5}	3.83×10^{-5}
80×80 [11]	3.0×10^{-3}	2.0×10^{-3}

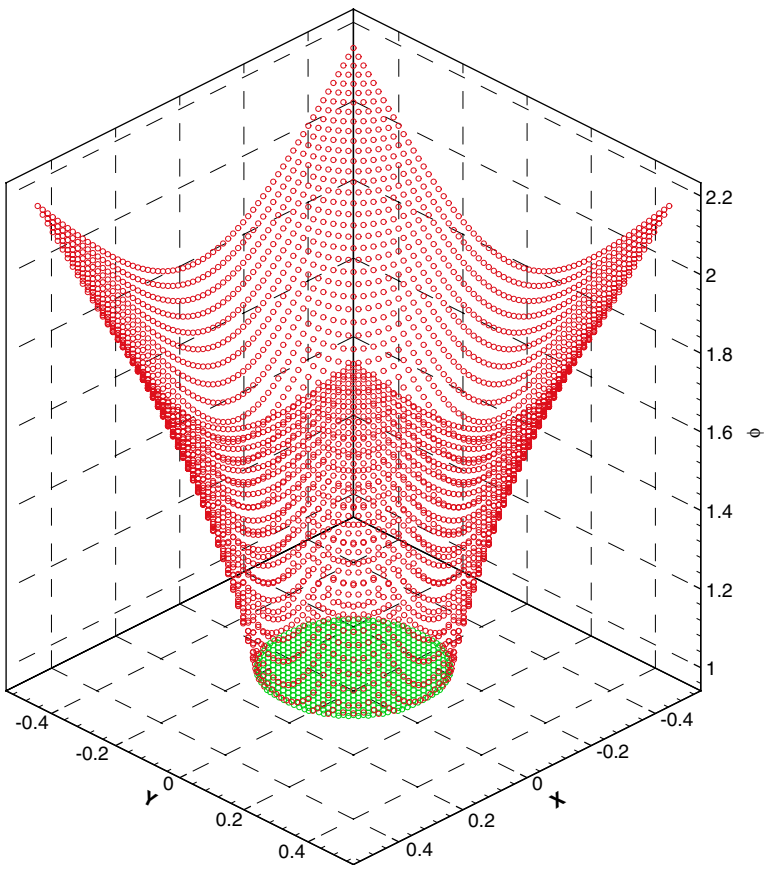

Figure 12. The numerical solution of the test case 3 with 64 grid points in each direction.

3.3. *Test case 3*

Consider $\Delta\phi = 0$ defined on a domain of definition $[-0.5, 0.5] \times [-0.5, 0.5]$ with the interface defined by the circle $x^2 + y^2 = 0.215^2$ and an outward pointing normal vector denoted by $(n_x\mathbf{i} + n_y\mathbf{j})$. The jump conditions are $[\phi] = 0$ and $[\phi_n] = (xn_x + yn_y)/0.215^2$ on the circular boundary. The corresponding exact solutions are $\phi(x, y) = 1$ in the interior of the circle and $\phi(x, y) = 1 + \ln(\sqrt{x^2 + y^2}/0.215)$ in the region exterior to the circle. Dirichlet boundary conditions are imposed on the square domain boundary. Figure 12 shows the numerical solution with 64 grid points in each direction. From an examination of the behaviour of $\|\varepsilon\|_1$, $\|\varepsilon\|_2$ and $\|\varepsilon\|_\infty$

Table IV. Numerical accuracy test for the case 3.

Grids	$\|\varepsilon\|_1$	r_1	$\|\varepsilon\|_2$	r_2	$\|\varepsilon\|_\infty$	r_∞
32×32	5.00×10^{-4}		5.03×10^{-4}		6.54×10^{-4}	
64×64	1.17×10^{-4}	2.13	1.18×10^{-4}	2.09	1.59×10^{-5}	2.04
128×128	2.77×10^{-5}	2.08	2.77×10^{-5}	2.09	3.69×10^{-5}	2.11
256×256	6.97×10^{-6}	1.99	7.00×10^{-6}	1.98	1.06×10^{-5}	1.80

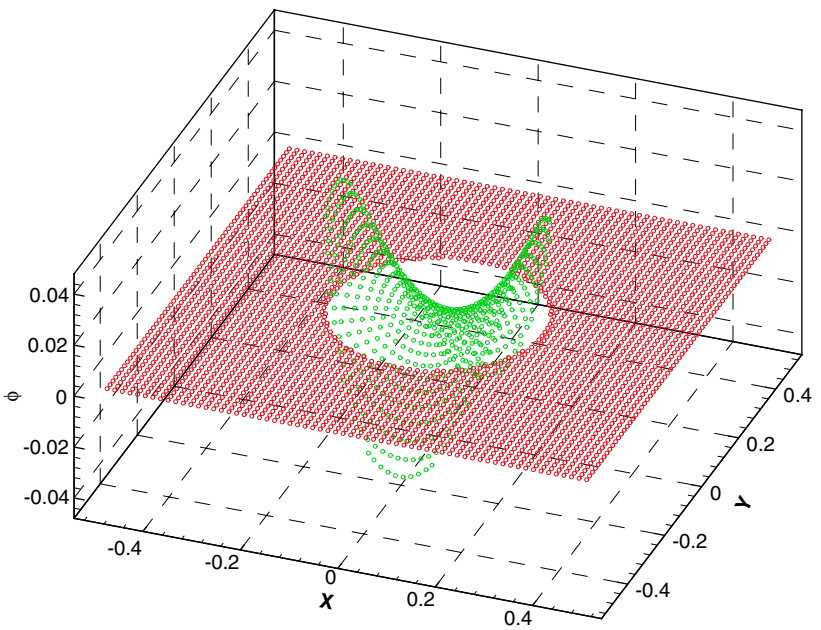

Figure 13. The numerical solution of the test case 4 with 64 grid points in each direction.

as a function of the grid resolution, it can be seen from Table IV that our proposed numerical method is second-order accurate.

3.4. *Test case 4*

Consider $\Delta\phi = 0$ on the domain of definition $[-0.5, 0.5] \times [-0.5, 0.5]$ with the interface defined by the circle $x^2 + y^2 = 0.215^2$ and an outward pointing normal vector denoted by $(n_x\mathbf{i} + n_y\mathbf{j})$. The jump conditions are $[\phi] = y^2 - x^2$ and $[\phi_n] = 2(yn_y - xn_x)$ along the circular boundary. The corresponding exact solutions are $\phi(x, y) = x^2 - y^2$ in the region inside the circular boundary and $\phi(x, y) = 0$ in the region outside the circular boundary. Dirichlet boundary conditions are imposed on the square domain boundary. Figure 13 shows the numerical solution with 64 grid points in each direction. Table V demonstrates that second-order accuracy in the numerical solution has been achieved using our proposed methodology.

Table V. Numerical accuracy test for the case 4.

Grids	$\|\varepsilon\|_1$	r_1	$\|\varepsilon\|_2$	r_2	$\|\varepsilon\|_\infty$	r_∞
32×32	5.59×10^{-5}		8.78×10^{-5}		3.45×10^{-4}	
64×64	1.47×10^{-5}	1.93	2.34×10^{-5}	1.91	9.06×10^{-5}	1.93
128×128	3.61×10^{-6}	2.02	5.86×10^{-6}	2.00	2.59×10^{-5}	1.81
256×256	8.96×10^{-7}	2.01	1.46×10^{-6}	2.00	7.54×10^{-6}	1.78

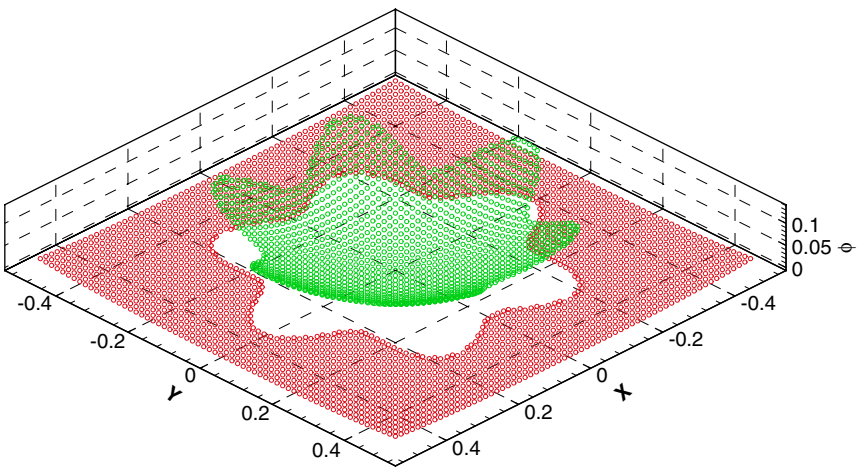

Figure 14. The numerical solution of the test case 5 with 64 grid points in each direction.

3.5. Test case 5

Consider $\nabla \cdot (\beta \nabla \phi) = f(x, y)$ on the domain of definition $[-0.5, 0.5] \times [-0.5, 0.5]$. The star-shaped interface is defined in polar coordinates as $r(\theta) = 0.32 + 0.05\cos(6\theta)$ with $\theta \in [0, 2\pi]$. The unit normal vector $(n_x\mathbf{i} + n_y\mathbf{j})$ is assumed to point from the interior to the exterior region. Here, $\beta = 1$ and $f(x, y) = 4$ in the interior of the star-shaped region; $\beta = 10$ and $f(x, y) = 16(x^2 + y^2)$ in the exterior of the star-shaped region. The jump conditions are $[\phi] = 0.1(x^2+y^2)^2 - 0.01\ln(2\sqrt{x^2 + y^2}) - (x^2+y^2)$ and $[\beta\phi_n] = (4(x^2+y^2) - 0.1(x^2+y^2)^{-1} - 2)(xn_x + yn_y)$ along the star-shaped curve. The corresponding exact solutions are $\phi(x, y) = x^2 + y^2$ in the region inside the star-shaped curve and $\phi(x, y) = 0.1(x^2 + y^2)^2 - 0.01\ln(2\sqrt{x^2 + y^2})$ in the region outside the star-shaped curve. Dirichlet boundary conditions are imposed on the square domain boundary. Figure 14 shows the numerical solution with 64 grid points in each direction. Consistent with the previous examples, a perusal of Table VI shows that our proposed method is second-order accurate.

3.6. Test case 6

Consider $\nabla \cdot (\beta \nabla \phi) = f(x, y)$ with a domain of definition $[-0.5, 0.5] \times [-0.5, 0.5]$. The star-shaped interface is defined in polar coordinates as $r(\theta) = 0.32 + 0.05\cos(6\theta)$ with $\theta \in [0, 2\pi]$. The unit normal vector $(n_x\mathbf{i} + n_y\mathbf{j})$ is assumed to point from the interior to the exterior

Table VI. Numerical accuracy test for the case 5.

Grids	$\|\varepsilon\|_1$	r_1	$\|\varepsilon\|_2$	r_2	$\|\varepsilon\|_\infty$	r_∞
32×32	1.85×10^{-4}		1.89×10^{-4}		2.21×10^{-4}	
64×64	5.35×10^{-5}	1.79	5.40×10^{-5}	1.81	6.03×10^{-5}	1.87
128×128	1.24×10^{-5}	2.11	1.25×10^{-5}	2.11	1.46×10^{-5}	2.05
256×256	3.19×10^{-6}	1.96	3.20×10^{-6}	1.97	3.61×10^{-6}	2.01

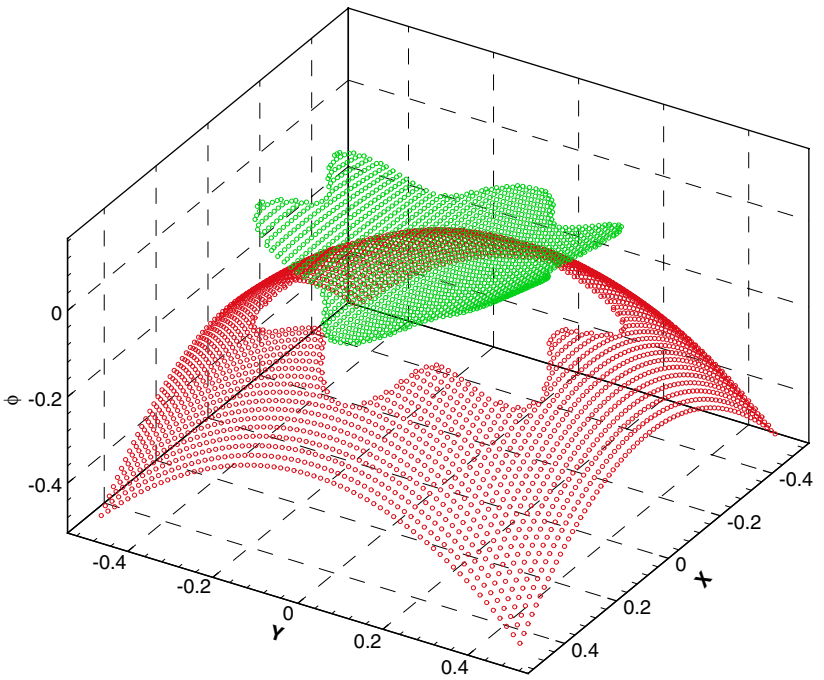

Figure 15. The numerical solution of the test case 6 with 64 grid points in each direction.

region. Here, $\beta = 1$ and $f(x, y) = e^x(2 + y^2 + 2\sin y + 4x\sin y)$ in the region interior to the star-shaped curve; $\beta = 10$ and $f(x, y) = -40$ in the region exterior to the star-shaped curve. The jump conditions are $[\phi] = -(x^2 + y^2) - e^x(x^2 \sin y + y^2)$ and $[\beta\phi_n] = (-20x - e^x(x^2 + 2x)\sin y + y^2)n_x + (-20y - e^x(x^2 \cos y + 2y))n_y$ along the star-shaped boundary. For this test case, the exact solution is $\phi(x, y) = e^x(x^2 \sin y + y^2)$ inside the star-shaped region and $\phi(x, y) = -(x^2 + y^2)$ outside the star-shaped region. Dirichlet boundary conditions are imposed on the square domain boundary. Figure 15 shows the numerical solution with 64 grid points in each direction. Table VII demonstrates that the numerical solution obtained using our proposed numerical methodology is second-order accurate.

Table VII. Numerical accuracy test for the case 6.

Grids	$\|\varepsilon\|_1$	r_1	$\|\varepsilon\|_2$	r_2	$\|\varepsilon\|_\infty$	r_∞
32×32	4.24×10^{-4}		4.25×10^{-4}		5.68×10^{-4}	
64×64	1.36×10^{-4}	1.63	1.37×10^{-4}	1.63	1.88×10^{-4}	1.60
128×128	2.78×10^{-5}	2.29	2.80×10^{-5}	2.29	4.15×10^{-5}	2.18
256×256	7.49×10^{-6}	1.89	7.54×10^{-6}	1.89	1.10×10^{-5}	1.92

4. CONCLUSIONS

A method has been presented for solving the variable coefficient Poisson equation on a two-dimensional domain in the presence of irregular interfaces where both the variable coefficient and the solution itself may be discontinuous. The approach involves using piecewise cubic splines to represent the irregular interface, and applying this representation to calculate the geometrical information on each cut-cell (e.g. cell volume, areas of cell faces) required for the discretization of the Poisson equation using the finite volume method. Although the use of cubic splines allows us to accurately calculate the curvature of an interface and, hence, the associated surface tension effect for free-surface problems [13], the proposed method is currently limited to two-dimensional problems.

The gradient fluxes of ϕ on the cut-cell faces and the interface faces at the face mid-points are evaluated using a second-order accurate scheme. The resulting discretization of the Poisson equation on an irregular (complex) domain is rearranged in such a way that the computational stencil on the *implicit* part of the discretized equation is identical to that obtained in a conventional discretization of the Poisson equation over a regular (simple) domain plus additional *explicit* correction terms for cells involving an immersed interface. Consequently, an efficient iterative solver, such as the multigrid method, for the Poisson equation over a regular domain can be applied to the current discretized equation system for the irregular domain without any major changes.

Several test cases (for which exact solutions of the Poisson equation are known) have been used used to test our new methodology for discretization on an irregular domain. A comparison of the numerical solutions with the exact solutions for these test cases demonstrates that our new solution methodology is second-order accurate and can keep the interface sharp as claimed. It is noted that the major differences between the present cut-cell approach and GFM in solving a variable coefficient Poisson equation with jump conditions are that (1) the present method does not require ghost cells, and (2) the present method imposes jump conditions directly (or explicitly) at the interface to secure the second-order accuracy in the solution.

ACKNOWLEDGEMENTS

This work has been partially supported by Chemical Biological Radiological Nuclear Research and Technology Initiative (CRTI) program under project number CRTI-02-0093RD.

REFERENCES

1. Ferziger JH, Peric M. *Computational Methods for Fluid Dynamics*. Springer: Berlin, 1996.
2. Udaykumar HS, Shyy W, Rao MM. Elafint: a mixed Eulerian–Lagrangian method for fluid flows with complex and moving boundaries. *International Journal for Numerical Methods in Fluids* 1996; **22**:691–712.
3. Udaykumar HS, Mittal R, Rampunggoon P. Interface tracking finite volume method for complex solid–fluid interactions on fixed meshes. *Communications in Numerical Methods in Engineering* 2002; **18**:89–97.
4. Peskin C. Numerical analysis of blood flow in the heart. *Journal of Computational Physics* 1977; **25**:220–235.
5. LeVeque RJ, Li Z. The immersed interface method for elliptic equations with discontinuous coefficients and singular sources. *SIAM Journal on Numerical Analysis* 1994; **31**:1019–1033.
6. Johansen H, Colella P. A Cartesian grid embedded boundary method for Poisson's equation on irregular domains. *Journal of Computational Physics* 1998; **147**:60–85.
7. Fedkiw R, Aslam T, Merriman B, Osher S. A non-oscillatory Eulerian approach to interfaces in multimaterial flows (The Ghost Fluid Method). *Journal of Computational Physics* 1999; **152**:457–492.
8. Tseng YH, Ferziger JH. A ghost-cell immersed boundary method for flow in complex geometry. *Journal of Computational Physics* 2003; **192**:593–623.
9. Li Z. A note on immersed interface method for three dimensional elliptic equations. *Computers and Mathematics with Applications* 1996; **35**:9–31.
10. Youngs DL. Time-dependent multi-material flow with large fluid distortion. In *Numerical Methods for Fluid Dynamics*, Morton KW, Baines MJ (eds). Academic Press: New York, 1982; 27–39.
11. Liu X-D, Fedkiw R, Kang M. A boundary condition capturing method for Poisson's equation on irregular domains. *Journal of Computational Physics* 2000; **160**:151–178.
12. Gibou F, Fedkiw R, Cheng L-T, Kang M. A second order accurate symmetric discretization of the Poisson equation on irregular domains. *Journal of Computational Physics* 2002; **176**:205–227.
13. Ji H. Large eddy simulation of free surface flows. *Ph.D. Thesis*, Department of Mechanical Engineering, University of Waterloo, Canada, 2004.
14. Barth TJ. A 3-D upwind Euler solver for unstructured meshes. *AIAA Paper 91-1548*, 1991.
15. Hutchinson BR, Raithby GD. A multigrid method based on the additive correction strategy. *Numerical Heat Transfer* 1986; **9**:511–537.
16. Raw M. Robustness of coupled algebraic multigrid for the Navier–Stokes equations. *34th Aerospace Sciences Meeting and Exhibit*, *AIAA 96-0297*, Reno, NV, 1996.
17. Press WH, Flannery BP, Teukolsky SA, Vetterling WT. *Numerical Recipes*: *the Art of Scientific Computing* (2nd edn). Cambridge University Press: Cambridge, 1989.
18. Barth TJ, Jersperson DC. The design and application of upwind schemes on unstructured meshes. *AIAA Paper 89-0366*, 1989.

