Effects of Physical Coordinate Clustering on Boundary Vorticity Approximations in von Mises Coordinates

M.H. HAMDAN
Department of Mathematics and Statistics
University of New Brunswick, P.O. Box 5050, Saint John, N.B.
CANADA

Abstract:- Forward finite difference expressions of first-order accuracy for boundary vorticity on a solid boundary are evaluated in this work when the physical coordinates are clustered and mapped using von Mises coordinates. Results show that schemes using in-field grid points do not improve solutions obtained. Results also show that the finer the grid used in the physical domain, and the more clustered it is, improves the boundary vorticity values in the computational domain. The "best" expressions forward finite difference expressions are identified when two, three, four and five grid points are used.

Key Words:- boundary vorticity, irregular domains, von Mises transformation

Received: March 22, 2021. Revised: October 7, 2021. Accepted: October 25, 2021. Published: November 20, 2021.

1 Introduction

In the study of two-dimensional, viscous fluid flow, it is customary to solve the governing Navier-Stokes equations in the following vorticity-streamfunction form, [1]:

$$\psi_{xx} + \psi_{yy} = -\omega \tag{1}$$

$$\omega_{xx} + \omega_{yy} = Re\{\psi_y \omega_x - \psi_x \omega_y\} \tag{2}$$

wherein $\psi(x, y)$ is the streamfunction, $\omega(x, y)$ is the vorticity, Re is the Reynolds number and subscript notation denotes partial differentiation.

The tangential and normal velocity components of the flow, $u(x, y)$ and $v(x, y)$, respectively, are related to vorticity and streamfunction, as follows:

$$u = \psi_y \tag{3}$$
$$v = -\psi_x \tag{4}$$

$$\omega = v_x - u_y \tag{5}$$

The vorticity-streamfunction form has advantages over using primitive variables, which include avoiding pressure boundary conditions and the automatic satisfaction of the equation of continuity. However, while the streamfunction enjoys Dirichlet-type conditions, associated with the vorticity are Neumann-type conditions obtained from (5).

There exist an abundance of numerical approaches to this type of flow problem, [2,3]. A method of interest in the current work is the finite differences procedure due to its ease of use and knowledge of its local errors. As is well-known, the finite differences method works well when the flow domain is regular; that is, boundaries of the flow domain coincide with coordinate lines (rectangular) or curves (polar), as examples. If the flow domain is curvilinear, utility of this procedure can be extended by either mapping the flow domain into a regular computational domain, or resorting to grid generation techniques. Many techniques and domain transformations are available in the literature and entail mapping the domain, the governing equations and boundary conditions, then seeking a solution using finite differences. Once a solution is obtained, it is mapped back to the physical domain where results and flow patterns are analyzed (*cf.* [4-9] and the references therein).

A transformation of interest in the current work is the following well-known von Mises transformation, $(x,y) \rightarrow (x,\psi)$, defined by $y = y(x,\psi)$. In the curvilinear net (x,ψ), the curves $\psi = constant$ represent the streamlines of the flow, [10-14]. Jacobian of the von Mises transformation is given by $J = \left|\frac{\partial(x,y)}{\partial(x,\psi)}\right| = y_\psi$. If $0 < J, \infty$, then the inverse transformation exists and the following first partial derivative operators in the two coordinate systems are obtained, [10]:

$$\partial_x = \partial_x - \frac{y_x}{y_\psi}\partial_\psi \tag{6}$$
$$\partial_y = \frac{1}{y_\psi}\partial_\psi \tag{7}$$

Second partial derivative operators can be obtained by applying operators (6) and (7) onto themselves.
Applying the transformation to vorticity-streamfunction equations (1) and (2), respectively, yields:

$$L(y) = \omega(y_\psi)^3 \tag{8}$$
$$L(\omega) = \omega\omega_\psi(y_\psi)^2 + Re\, y_\psi\, \omega_x \tag{9}$$

where

$$L \equiv y_\psi{}^2\partial_{xx} - 2\, y_x\, y_\psi\, \partial_{x\psi} + \; [1 + y_x{}^2]\partial_{\psi\psi} \tag{10}$$

Velocity components and vorticity, defined by equations (3)-(5), take the following forms, respectively, in the new coordinate system:

$$u = \frac{1}{y_\psi} \tag{11}$$
$$v = \frac{y_x}{y_\psi} = u y_x \tag{12}$$
$$\omega = v_x + \left(\frac{v^2}{u} - u\right)u_\psi - \frac{v}{u}u_x - 2vv_\psi \tag{13}$$

The square of the speed of the flow is given by

$$q^2 = u^2 + v^2 = \frac{1+(y_x)^2}{(y_\psi)^2} \tag{14}$$

Vorticity, (13), can thus be written in terms of the square of the speed, $q^2 = u^2 + v^2$, as

$$\omega = v_x - \frac{1}{2}(q^2)_\psi \tag{15}$$

In Section 2 below, it will be shown that on the computational domain boundary, $v_x = 0$ and equation (15) reduces to:

$$\omega = -\frac{1}{2}(q^2)_\psi \tag{16}$$

Equation (16) represents the Neumann vorticity condition that is used to compute boundary vorticity. Finite difference evaluation of the first derivative in (16), and computing the "*best*" value of boundary vorticity (that is, the value closest to the true value) influences the numerical solution to governing equations (8) and (9), and is the main theme of this work. Computations of boundary vorticity using (16) is influenced by factors that include accuracy of the scheme and the choice of grid spacing used.

Accuracy of the scheme affects the local truncation error [18-21]. To this end, standard and non-standard schemes of first, second, third and fourth order of accuracy have been developed and tested using both boundary grid points and up to four in-field grid points, [18,19]. Equally important is the choice of grid spacing. It has been shown, [9,18,19], that if a uniform grid is fitted over the computational domain, the corresponding gridlines in the physical domain would be clustered away from the boundary, and if a uniform grid is chosen in the physical domain, the von Mises transformation naturally produces a desired clustered grid lines near the boundary in the computational domain. In order to capture the effects of boundary conditions more accurately, one needs to have more clustering near the boundary. To this end, Alharbi *et.al.* [19] considered higher order schemes with both uniform and clustered physical domain and showed that a clustered grid in the physical domain produces more clustering in the computational domain, and more accurate results.

Siyyam *et.al.* [18] considered standard and non-standard first-order accurate schemes but only employed a uniform grid in the physical domain, which produced a natural clustering in the computational domain. Their results showed that the best schemes were the schemes that used the natural order of grid lines, as follows. In two-point schemes, the (1,2)-scheme uses the boundary grid line (say *j=1*) and the closes internal grid line to it (*j=2*). In a three-point scheme, the (1,2,3)-scheme uses the boundary grid line (*j=1*) and the two adjacent and closest gridlines (*j=2* and *j=3*), and so on. This leaves a gap in the literature, and the following question unanswered: *What are the effects of a clustered grid in the physical domain on first-order accurate schemes that use the natural order of grid lines?*

Results obtained from these schemes are compared with the exact value of vorticity. In order to accomplish this objective, this work is organized as follows. In Section 2, problem formulation is presented together with domain discretization data. In Section 3, finite difference schemes are presented together with their first omitted terms. In Section 4, results and discussion of scheme evaluations are presented, followed by conclusions to this work and suggestions for future direction.

In order to answer this question, this work employs and tests four first-order schemes, namely, the (1,2)-scheme, the (1,2,3)-schemes, the (1,2,3,4)-scheme and the (1,2,3,4,5)-scheme together with four different physical domain grids, one uniform and three clustered with different levels of clustering.

2 Problem Formulation

Consider the steady flow of a viscous, incompressible fluid in a two-dimensional, dimensionless channel bounded below and above by solid, curvilinear boundaries, as shown in **Fig. 1**. The channel is described by:

$\{(x,y)|\ f_1(x) \le y \le f_2(x); a \le x \le b\}$, where $f_1(x)$ and $f_2(x)$ are known smooth functions, and the flow is governed by equations (1) and (2), above. The streamfunction enjoys Dirichlet conditions when the solid boundaries are taken as streamlines of the flow, say $\psi = \psi_{\min}$ on the lower boundary and $\psi = \psi_{\max}$ on the upper boundary. At the inlet and exit of the channel, the Dirichlet conditions on ψ are determined from the inlet and exit tangential velocity profile for parallel flow, by integrating equation (3). An example of a typical inlet and exit profiles is given as follows. Assuming that the channel is long-enough for the normal component of velocity to vanish there and that at $x = a$ and b, the value of y ranges on the interval $[-1,1]$, we can assume the parallel and parabolic inlet velocity profile given by:

$$v = 0 \text{ and } u = 1 - y^2;\ -1 \le y \le 1 \tag{17}$$

Integrating (3) and using (17), we obtain

$$\psi = y - \frac{y^3}{3};\ -1 \le y \le 1 \tag{18}$$

The absence of explicit vorticity boundary conditions necessitates imposing appropriate inlet, exit and solid boundary conditions on the vorticity. Assuming that equation (5) is valid on all boundaries then the conditions:

$$v_x = 0 \text{ and } \omega = -u_y \tag{19}$$

can be used on all boundaries of the configuration at hand (even at the exit of the channel, since the profile of u is assumed to be known there). Alternatively, ω_x can be taken to be zero at the exit of the channel.

Using (17) and (19), we obtain $\omega = 2y$ at the inlet to the channel. In particular, we are interested in the value of vorticity at $y = -1$, namely $\omega(y = -1) = -2$, in the development of forward differencing schemes.

In using the von Mises this transformation, the problem of solving equations (1) and (2) for $\psi(x,y)$ and $\omega(x,y)$ in the physical curvilinear $xy - plane$ subject to boundary conditions on ψ and ω, has been replaced by the problem of solving equations (8) and (9) for $y(x,\psi)$ and $\omega(x,\psi)$ in the rectangular $x\psi - plane$, shown in **Fig. 2**, described by $\{(x,\psi)|\ \psi_{min} \le \psi \le \psi_{max};\ a \le x \le b\}$, and subject to transformed boundary conditions on the variables y and ω.

The computational conditions on y are as follows:
On the lower boundary

$$y = f_1(x) \tag{20}$$

On the upper boundary

$$y = f_2(x) \tag{21}$$

At inlet and exit to the channel

$$y - \frac{y^3}{3} = \psi \text{ for } -\frac{2}{3} \le \psi \le \frac{2}{3} \tag{22}$$

The computational condition on ω at inlet and exit, and on the lower and upper boundaries is given by (16).

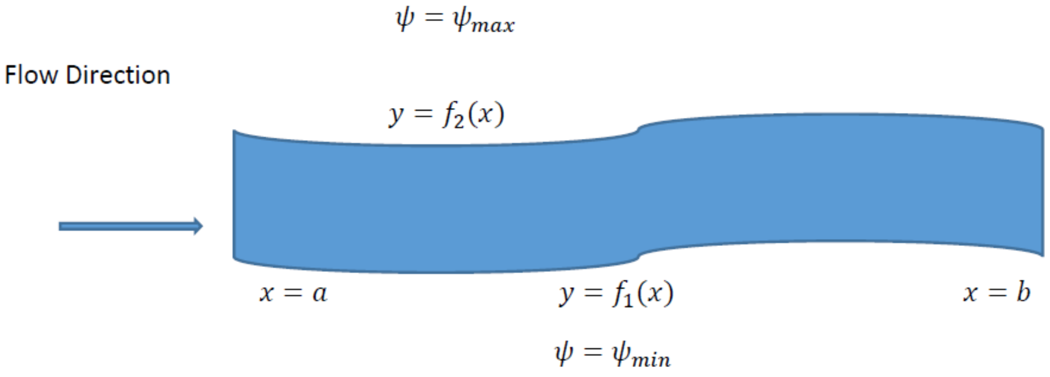
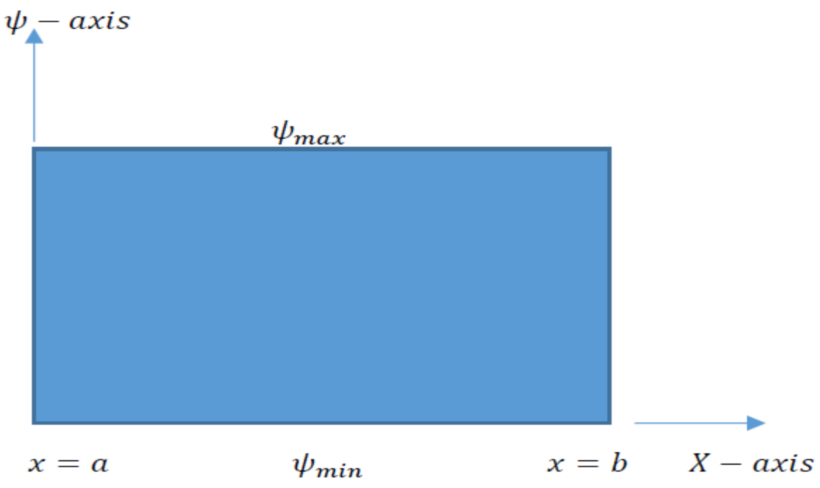


Fig. 1 Representative Sketch (Physical Domain)



2.1 Discretizing the Flow Domain

The rectangular computational domain of **Fig. 2** is discretized using either a uniform grid or a clustered grid, with the vertical grid lines ranging from *i=1* at *x=a* to *i=Imax* at *x=b*, and the horizontal grid lines ranging from *j=1* at the lower computational boundary $(\psi = -\frac{2}{3})$ to *j=Jmax* at the upper computational boundary $(\psi = \frac{2}{3})$.

In order to capture the effects of boundary conditions more accurately, grid needs to have clustering near the boundary. Clustering of grid lines near the computational boundary when the von Mises variables are used can be accomplished in two ways:

1) Selecting a uniform grid in the physical domain and calculating the grid spacings in the computational domain. This results in a "natural" clustering near the computational boundary. For the given parabolic inlet velocity profile, and the indicated range of streamlines, we can choose y to vary over the interval *[-1,1]*, and calculate the step size $\Delta\psi$ between two adjacent grid lines.

Suppose that we wish to produce a uniform grid of step size $\Delta y = 0.01$ in the physical domain, then clustered grid is computed using $y-\frac{y^3}{3}=\psi$ for $-\frac{2}{3}\le\psi\le\frac{2}{3}$. The variable grid spacings in the computational domain, corresponding to a uniform grid in the physical domain with $\Delta y=0.01$, are shown in **Table** 1, wherein $\Delta\psi_j=\psi_{j+1}-\psi_j$ for *j=1,2,3,4.* Clearly, the computational grid lines are clustered near the computational boundary, even though the physical grid lines were equally spaced (uniform grid). This points out the fact that the use of the von Mises variables has the added advantage of naturally producing clustered grid near the computational boundaries without resorting to an algebraic method to cluster the grid.

2) Selecting a grid in the physical domain that is non-uniform and clustered near the physical boundary produces a computational grid that is also clustered near the computational boundary. This is the heart of the current work in which we develop appropriate forward finite difference expressions and study the effects of clustering the grid in the physical domain on local truncation errors.

Physical grid clustering can be accomplished in various ways, including the use of elementary functions. For the current work, we employ the square root, the cubic root and the square functions to define the clustering in the physical y-direction. These are illustrated in **Tables 2** to **4**, wherein we provide a comparison with the case of uniform grid in the physical domain (**Table 1**). It should be noted that five grid points are chosen in this work (one boundary point, *j=1*, and four internal points, *j=2,3,4,5*).

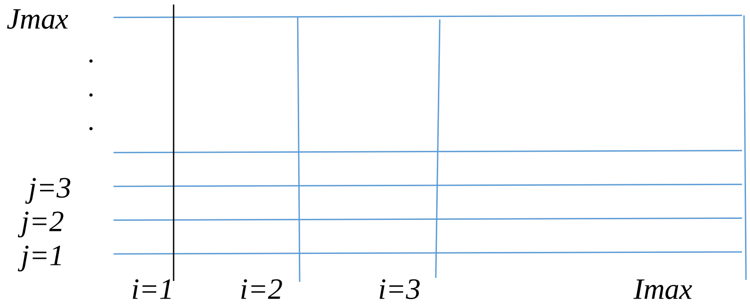


Fig. 3. Grid Lines. *(i,j)=(1,1)* Corresponds to *(x,y)=(a,-1)*

j	y_j	Δy_j	ψ_j	$\Delta\psi_j$
1	$y_1=-1$	0.01	$\psi_1=-0.666666666$	$\Delta\psi_1=0.000099666$
2	$y_2=-0.99$	0.01	$\psi_2=-0.666567000$	$\Delta\psi_2=0.000297667$
3	$y_3=-0.98$	0.01	$\psi_3=-0.666269333$	$\Delta\psi_3=0.000493667$
4	$y_4=-0.97$	0.01	$\psi_4=-0.665775666$	$\Delta\psi_4=0.000687666$
5	$y_5=-0.96$	0.01	$\psi_5=-0.665088000$	

Table 1:**Grid 1.** Uniform Grid in the Physical Domain
$y_j=-y_j^*$ for y_j^* =1,0.99,0.98, 0.97,0.96

j	y_j	Δy_j	ψ_j	$\Delta\psi_j$
1	$y_1 = -1$	0.005012567	$\psi_1 = -0.666666666$	$\Delta\psi_1 = 0.000025083$
2	$y_2 = -0.994987437$	0.005037944	$\psi_2 = -0.666641582$	$\Delta\psi_2 = 0.00007559$
3	$y_3 = -0.989949493$	0.005063713	$\psi_3 = -0.666565991$	$\Delta\psi_3 = 0.000126613$
4	$y_4 = -0.98488578$	0.005089883	$\psi_4 = -0.666439377$	$\Delta\psi_4 = 0.000178167$
5	$y_5 = -0.979795897$		$\psi_5 = -0.666261209$	

Table 2: **Grid 2.** Non-uniform Grid in the Physical Domain.

$y_j = -\sqrt{y_j^{*}}$ for y_j^{*}=1, 0.99, 0.98, 0.97, 0.96

j	y_j	Δy_j	ψ_j	$\Delta\psi_j$
1	$y_1 = -1$	0.003344507	$\psi_1 = -0.666666666$	$\Delta\psi_1 = 0.000011173$
2	$y_2 = -0.996655493$	0.003367105	$\psi_2 = -0.666655493$	$\Delta\psi_2 = 0.000033772$
3	$y_3 = -0.993288388$	0.003390089	$\psi_3 = -0.666621721$	$\Delta\psi_3 = 0.000056756$
4	$y_4 = -0.989898299$	0.003413469	$\psi_4 = -0.666564965$	$\Delta\psi_4 = 0.000080136$
5	$y_5 = -0.986484829$		$\psi_5 = -0.666484829$	

Table 3: **Grid 3.** Non-uniform Grid in the Physical Domain

$y_j = -\sqrt[3]{y_j^{*}}$ for y_j^{*}=1,0.99,0.98, 0.97, 0.96

j	y_j	Δy_j	ψ_j	$\Delta\psi_j$
1	$y_1 = -1$	0.0199	$\psi_1 = -0.666666666$	$\Delta\psi_1 = 0.000393383$
2	$y_2 = -0.9801$	0.0197	$\psi_2 = -0.666273283$	$\Delta\psi_2 = 0.001154077$
3	$y_3 = -0.9604$	0.0195	$\psi_3 = -0.665119206$	$\Delta\psi_3 = 0.001876541$
4	$y_4 = -0.9409$	0.0193	$\psi_4 = -0.663242665$	$\Delta\psi_4 = 0.002561929$
5	$y_5 = -0.9216$		$\psi_5 = -0.660680736$	

Table 4: **Grid 4.** Non-uniform Grid in the Physical Domain

$y_j = -(y_j^{*})^2$ for y_j^{*}= 1,0.99, 0.98, 0.97, 0.96

Clearly, these variable step sizes are fine near the computational boundary, and are expected to better capture the boundary effects and render more accurate computations. Moreover, they “correspond” to a coarse-enough uniform grid in the physical domain to "avoid" regions of viscous separation, recirculation, and flow reversal, that hinders the use of the von Mises transformation without domain decomposition, due to the change of sign of the Jacobian of transformation. These effects need special considerations that are not discussed in this work.

2.2 Calculating Inlet Square of Speed

At the inlet to the channel, the computational velocity conditions are: $v(x,\psi)=0$, and $u(x,\psi)=1-y^2$ for $-1\leq y\leq 1$. The square of the speed at inlet to the computational domain is given by

$$q^2(x,\psi)=u^2+v^2=\ (1-y^2)^2 \tag{23}$$

Using (23), we compute $(q^2)_j$ for each of the four types of grid (**Tables 1-4**), at the first five grid points. These are illustrated in **Table 5**, below.

j	${q^2}_j$ Grid1	${q^2}_j$ Grid2	${q^2}_j$ Grid3	${q^2}_j$ Grid4
1	0	0	0	0
2	0.00039601	0.0001	0.000044593	0.001552674
3	0.00156816	0.0004	0.000178975	0.006026702
4	0.00349281	0.0009	0.000404064	0.013157739
5	0.00614656	0.0016	0.000720798	0.022696458

Table 5: Square of the Speed at Inlet to Computational Domain for the Four Grids Used.

2.3 Finite Difference Schemes

The definition of vorticity is expressed at the lower boundary of the computational domain, in von Mises coordinates as the following derivative of the square of the speed of the flow:

$$\omega_{i,1}=-\tfrac{1}{2}\left(q_\psi^2\right)_{i,1} \tag{24}$$

In order to obtain finite difference expressions for $\omega_{i,1}$, Siyyam *et.al.*, [18], developed the following forward finite difference schemes for the first partial derivative on the right-hand –side of (24), using up to four internal grid points. The schemes are based on the natural order or points. For instance, a two-point scheme uses grid points on the grid lines *j=1* and *j=2*. A five-point scheme uses grid points on *j=1,2,3,4,5*. The following is used in the schemes below.

$$\Delta\psi_j=\psi_{j+1}-\psi_j;j=1,2,3,4 \tag{25}$$

$$\left.\begin{aligned} a_1&=\Delta\psi_1\\ a_2&=\Delta\psi_1+\Delta\psi_2\\ a_3&=\Delta\psi_1+\Delta\psi_2+\Delta\psi_3\\ a_4&=\Delta\psi_1+\Delta\psi_2+\Delta\psi_3+\Delta\psi_4\end{aligned}\right\} \tag{26}$$

2.3.1 Two-point Scheme: (1,2)-Scheme

(1,2) Scheme	$\left(q_\psi^2\right)_{i,1}$	Leading error term

Non-uniform Grid	$\frac{(q^2)_{i,2}-(q^2)_{i,1}}{\Delta\psi_1}$	$-\frac{\Delta\psi_1}{2}\left(q^2_{\psi\psi}\right)_{i,1}$

Table 6. First-order Accurate, (1,2) Scheme for $(q^2{}_{\psi})_{i,1}$

2.3.2 Three-Point Scheme (1,2,3)-Scheme

(1,2,3) Scheme	$\left(q^2_{\psi}\right)_{i,1}$	Leading error term
Non-uniform Grid	$\frac{(q^2)_{i,3}+(q^2)_{i,2}-2(q^2)_{i,1}}{2\Delta\psi_1+\Delta\psi_2}$	$-\frac{(2\Delta\psi_1+\Delta\psi_2)^2+(\Delta\psi_2)^2}{4(2\Delta\psi_1+\Delta\psi_2)}\left(q^2_{\psi\psi}\right)_{i,1}$

Table 7. First-order Accurate, (1,2,3) Scheme for $(q^2{}_{\psi})_{i,1}$

2.3.3 Four-Point Schemes: (1,2,3,4) Schemes:

Four-point Schemes	$(q^2{}_{\psi})_{i,1}$
Scheme 1	$\frac{(q^2)_{i,4}+(q^2)_{i,3}+(q^2)_{i,2}-3(q^2)_{i,1}}{3\Delta\psi_1+2\Delta\psi_2+\Delta\psi_3}$
Scheme 2	$\frac{(q^2)_{i,4}+(q^2)_{i,3}-(q^2)_{i,2}-(q^2)_{i,1}}{\Delta\psi_1+2\Delta\psi_2+\Delta\psi_3}$
Scheme 3	$\frac{(q^2)_{i,4}-(q^2)_{i,3}+(q^2)_{i,2}-(q^2)_{i,1}}{\Delta\psi_1+\Delta\psi_3}$
Scheme 4	$\frac{(q^2)_{i,4}-(q^2)_{i,3}-(q^2)_{i,2}+(q^2)_{i,1}}{\Delta\psi_3-\Delta\psi_1}$

Table 8a. First-order Accurate, (1,2,3,4) Schemes for $(q^2{}_{\psi})_{i,1}$

Four-point Schemes	***Leading Error Term***
Scheme 1	$-\frac{1}{2}\left(\frac{a_1{}^2+a_2{}^2+a_3{}^2}{a_1+a_2+a_3}\right)(q^2{}_{\psi\psi})_{i,1}$
Scheme 2	$-\frac{1}{2}\left(\frac{-a_1{}^2+a_2{}^2+a_3{}^2}{-a_1+a_2+a_3}\right)(q^2{}_{\psi\psi})_{i,1}$
Scheme 3	$-\frac{1}{2}\left(\frac{a_1{}^2-a_2{}^2+a_3{}^2}{a_1-a_2+a_3}\right)(q^2{}_{\psi\psi})_{i,1}$

Scheme 4	$-\frac{1}{2}\left(\frac{-a_1^{\;2}-a_2^{\;2}+a_3^{\;2}}{-a_1-a_2+a_3}\right)(q^2{}_{\psi\psi})_{i,1}$

Table 8b. Leading Error Terms for First-order Accurate, (1,2,3,4) Schemes

2.3.4 Five-Point Schemes: (1,2,3,4,5) Schemes

Five-point Schemes	$(q^2{}_{\psi})_{i,1}$
(1,2,3,4,5)-Scheme 1	$\frac{(q^2)_{i,5}+(q^2)_{i,4}+(q^2)_{i,3}+(q^2)_{i,2}-4(q^2)_{i,1}}{4\Delta\psi_1+3\Delta\psi_2+2\Delta\psi_3+\Delta\psi_4}$
(1,2,3,4,5)-Scheme 2	$\frac{(q^2)_{i,5}+(q^2)_{i,4}+(q^2)_{i,3}-(q^2)_{i,2}-2(q^2)_{i,1}}{2\Delta\psi_1+3\Delta\psi_2+2\Delta\psi_3+\Delta\psi_4}$
(1,2,3,4,5)-Scheme 3	$\frac{(q^2)_{i,5}+(q^2)_{i,4}-(q^2)_{i,3}+(q^2)_{i,2}-2(q^2)_{i,1}}{2\Delta\psi_1+\Delta\psi_2+2\Delta\psi_3+\Delta\psi_4}$
(1,2,3,4,5)-Scheme 4	$\frac{(q^2)_{i,5}-(q^2)_{i,4}+(q^2)_{i,3}+(q^2)_{i,2}-2(q^2)_{i,1}}{2\Delta\psi_1+\Delta\psi_2+\Delta\psi_4}$
(1,2,3,4,5)-Scheme 5	$\frac{-(q^2)_{i,5}+(q^2)_{i,4}+(q^2)_{i,3}+(q^2)_{i,2}-2(q^2)_{i,1}}{2\Delta\psi_1+\Delta\psi_2-\Delta\psi_4}$

Table 10a. First-order Accurate, (1,2,3,4,5) Schemes for $(q^2{}_{\psi})_{i,1}$

Five-point Schemes	***Leading Error Term***
(1,2,3,4,5)-Scheme 1	$-\frac{1}{2}\left(\frac{a_1^{\;2}+a_2^{\;2}+a_3^{\;2}+a_4^{\;2}}{a_1+a_2+a_3+a_4}\right)(q^2{}_{\psi\psi})_{i,1}$
(1,2,3,4,5)-Scheme 2	$-\frac{1}{2}\left(\frac{-a_1^{\;2}+a_2^{\;2}+a_3^{\;2}+a_4^{\;2}}{-a_1+a_2+a_3+a_4}\right)(q^2{}_{\psi\psi})_{i,1}$
(1,2,3,4,5)-Scheme 3	$-\frac{1}{2}\left(\frac{a_1^{\;2}-a_2^{\;2}+a_3^{\;2}+a_4^{\;2}}{a_1-a_2+a_3+a_4}\right)(q^2{}_{\psi\psi})_{i,1}$
(1,2,3,4,5)-Scheme 4	$-\frac{1}{2}\left(\frac{a_1^{\;2}+a_2^{\;2}-a_3^{\;2}+a_4^{\;2}}{a_1+a_2-a_3+a_4}\right)(q^2{}_{\psi\psi})_{i,1}$
(1,2,3,4,5)-Scheme 5	$-\frac{1}{2}\left(\frac{-a_1^{\;2}-a_2^{\;2}-a_3^{\;2}+a_4^{\;2}}{-a_1-a_2-a_3+a_4}\right)(q^2{}_{\psi\psi})_{i,1}$

Table 10b. Leading Error Terms for First-order Accurate, (1,2,3,4,5) Schemes

3 Results and Discussion

Vorticity on the computational channel boundary is given by equation (24). The lower, left-corner vorticity has a value of $\omega_{1,1} = -2$, obtained from the channel inlet condition $\omega = 2y$ at $y = -1$.

Vorticity $\omega_{1,1}$ is approximated by evaluating all the schemes in **Tables 6-10** using the four grids of **Tables 1-4** in order to study the effects of grid clustering on vorticity. Values of $\omega_{1,1}$ are computed using the square of the speed values of **Table 5**.

Computed values of $\omega_{1,1}$ are tabulated below together with the ***Percentage Relative Error (P.R.E.)*** defined as:

$$P.R.E. = \frac{Exact\ \omega_{1,1} - Computed\ \omega_{1,1}}{Exact\ \omega_{1,1}} \tag{27}$$

The P.R.E. values are listed for each of the schemes in parenthesis in the tables below.

Scheme	***(1,2)-scheme***	***(1,2,3)-scheme***
Grid	$\omega_{1,1}$ *(P.R.E.)*	$\omega_{1,1}$ *(P.R.E.)*
Grid 1	-1.9867357 (0.666%)	-1.976030133 (1.198%)
Grid 2	-1.993381972 (0.331%)	-1.987976717 (0.601%)
Grid 3	**-1.995569677** **(0.221%)**	**-1. 991945543** **(0.403%)**
Grid 4	-1.973488941 (1.325%)	-1. 952598948 (2.371%)

Table 11a. Values of $\boldsymbol{\omega_{1,1}}$ and $\boldsymbol{P.R.E.}$ for (1,2)- and (1,2,3)-Schemes

(1,2,3.4) Schemes	***Scheme 1***	***Scheme 2***	***Scheme 3***	***Scheme 4***
Grid	$\omega_{1,1}$ *(P.R.E.)*	$\omega_{1,1}$ *(P.R.E.)*	$\omega_{1,1}$ *(P.R.E.)*	$\omega_{1,1}$ *(P.R.E.)*
Grid 1	**-1.96577231** **(1.711%)**	-1.962265294 (1.887%)	-1.955611481 (2.219%)	-1.939893554 (3.005%)
Grid 2	**-1.982766923** **(0.862%)**	-1.98100873 (0.949%)	-1.97763949 (1.118%)	-1.969861125 (1.506%)
Grid 3	**-1.988455129** **(0.577%)**	-1.987281599 (0.636%)	-1.985028486 (0.748%)	-1.979860913 (1.069%)

Grid 4	**-1.936405457** **(3.179%)**	-1.925673503 (3.716%)	-1.912775714 (4.361%)	-1.880569366 (5.972%)

Table 11b. Values of $\boldsymbol{\omega_{1,1}}$ and $\boldsymbol{P.R.E.}$ for (1,2,3,4)-Schemes

(1,2,3,4,5) Schemes	***Scheme 1***	***Scheme 2***	***Scheme 3***	***Scheme 4***	***Scheme 5***
Grid	$\omega_{1,1}$ *(P.R.E.)*	$\omega_{1,1}$ *(P.R.E.)*	$\omega_{1,1}$ *(P.R.E.)*	$\omega_{1,1}$ *(P.R.E.)*	$\omega_{1,1}$ *(P.R.E.)*
Grid 1	**-1.955653908** **(2.217%)**	-1.953418689 (2.329%)	-1.94917447 (2.541%)	-1.949040446 (2.547%)	-1.808335999 (9.583%)
Grid 2	**-1.97760038** **(1.119%)**	-1.97648268 (1.175%)	-1.974337206 (1.283%)	-1.974184251 (1.291%)	-1. 90799641 (4.600%)
Grid 3	**-1.984993641** **(0.750%)**	**-1.984248842** **(0.787%)**	**-1.982815916** **(0.859%)**	**-1.982701425** **(0.865%)**	**-1.939503706** **(3.025%)**
Grid 4	**-1.913242789** **(4.337%)**	-1.908755891 (4.562%)	-1. 89684611 (5.157%)	-1.900843438 (4.957%)	-1.577352412 (21.132%)

Table 11c. Values of $\boldsymbol{\omega_{1,1}}$ and $\boldsymbol{P.R.E.}$ for (1,2,3,4,5)-Schemes

Tables 11(a,b,c) illustrate the following:

1) Using Grid 3 consistently produces the most accurate computed values of vorticity and least Percentage Relative Errors (for all first-order schemes and grids used). This may be ascribed to the fact that Grid 3 uses grid points that are closest to the boundary. Associated with Grid 3 is the lowest truncation error.
2) Of the (1,2,3,4)-Schemes, Scheme I produces the most accurate corner vorticity and lease P.R.E., for all grids used. This scheme, together with Grid 3, produce the best results.
3) Of the (1,2,3,4,5)-Schemes, Scheme I produces the most accurate corner vorticity and lease P.R.E., for all grids used. This scheme, together with Grid 3, produce the best results.
4) The (1,2)-Scheme produces the most accurate results when used in conjunction with Grid 3.

4 CONCLUSION

In this work, the effects of grid clustering on forward finite difference approximations to the first derivative were considered. Results obtained support the following conclusions:

1- Grids with lowest Percentage Relative Errors for the schemes that use four or five points are, respectively: Grid 3, Grid 2, Grid 1, and Grid 4.

2- Schemes that employ the natural order, (1,2)-scheme, of grid points produce smaller Percentage Relative Errors, for all grids employed. This supports the general understanding that a standard first order scheme uses two points; a second order scheme uses three points; a third order scheme uses four points, and an *nth* order scheme requires *n+1* points.

3- Using clustered Grid 3 produces the lowest Percentage Relative Error, for all standard schemes used. This might be attributed to the lower truncation error associated with the scheme.

4- For first-order accurate schemes, the use of more internal grid points in the scheme did not improve the result.

5- When internal grid points are used, schemes that employ grid points closest to the boundary, in general, produce better results.

6- When using the von Mises transformation in the study of viscous fluid flow through irregular domains (in particular two-dimensional channels), the velocity inlet profile influences the type of grid employed in the computational domain.

7- Schemes with more clustering near the boundary (Grid 3 in this work) produce better results than grids with less clustering.

References

[1] Fletcher, C.A.J., *Computational Techniques for Fluid Dynamics*. Vol. II, Chapter 17, 1988, Springer-Verlag.

[2] Wang, J., Zhong, W. and Zhang, J., High Order Compact Computation and
Nonuniform Grids for Streamfunction Vorticity Equations, *Applied Math. and Computation*, Vol. 179(1), 2006, pp. 108–20.

[3] Pandit, S.K., Kalita, J.C. and Dalal, D.C., A Fourth Order Accurate Compact Scheme for the Solution of Steady Navier-Stokes Equations on Nonuniform Grids, *Computers and Fluids, Vol.* 37(2), 2008**,** pp. 121-134.

[4] de Rivas, E.K., On the Use of Nonuniform Grids in Finite-Difference Equations, *J. Computational Physics,* Vol. 10(2), 1972, pp. 202-210.

[5] Shukla, R.K. and Zhong, X., Derivation of High-order Compact Finite Difference Schemes for Non-uniform Grid Using Polynomial Interpolation, *J. Computational Physics,* Vol. 204, 2005, pp. 404–429.

[6] Spotz, W.F. and Carey, G.F., Formulation and Experiments with High-order Compact Schemes for Nonuniform Grids, *Int. J. Numerical Methods Heat Fluid Flow,* Vol. 8(3), 1998, pp. 288–303.

[7] Veldman, A.E.P. and Rinzema, K., Playing with Nonuniform Grids, *J. Engineering Math.,* Vol. 26, 1992, pp. 119-130.

[8] Visbal, M.R. and Gaitonde, D.V., On the Use of Higher-order Finite-difference Schemes on Curvilinear and Deforming Meshes, *J. Computational Physics,* Vol. 181, 2002, pp. 155–185.

[9] Awartani, M.M., Ford, R.A. and Hamdan, M.H., Computational Complexities and Streamfunction Coordinates, *Applied Math. and Computation*, Vol. 169(2), 2005, pp. 758-777.

[10] Barron, R.M., Computation of Incompressible Potential Flow Using von Mises Coordinates, *Math. Computers in Simulation*, Vol. 31, 1989, pp. 177-188.

[11] Hamdan, M.H., Natural Coordinate System Approach to Coupled n-Phase Fluid Flow in Curved Domains, *Applied Math. and Computation*, Vol. 85(2-3), 1997, pp. 297-304.

[12] Hamdan, M.H., Recent Developments in the von Mises Transformation and its Applications in the Computational Sciences: Plenary Lecture. *In*: *Mathematical Methods, System Theory and Control, MAMECTIS 09*, Eds. L. Perlovsky *et.al.*, (2009), 180-189.

[13] Barron, R.M. and Hamdan, M.H., The Double von Mises Transformation in the Study of Two-Phase Flow over Curved Boundaries: Theory & Analysis, *Int. J. Numerical Methods in Fluids,* Vol. 14 , 1992, pp. 883-905.

[14] Hamdan, M.H., Barron, R.M. and McCallum, F.M., Flow through Curved Porous Channels, *European Journal of Mechanics/B: Fluids*, Vol. 12(3), 1993, pp. 323-336.

[15] Agarwal, R.K., A Third-order-accurate Upwind Scheme for Navier-Stokes Solutions in Three Dimensions, In: *Computers in flow predictions and fluid dynamics experiments; Proceedings of the Winter Annual Meeting, Washington, DC, November 15-20, 1981, American Society of Mechanical Engineers, , (A82-29001 13-02),* 1981, pp. 73-82.

[16] Castillo, J.E., Hyman, J.M., Shashkov, J.M. and Steinberg, S., The Sensitivity and Accuracy of Fourth Order Finite-Difference Schemes on Nonuniform Grids in One Dimension, *Computers and Mathematics with Applications*, Vol. 30(8), 1995, pp. 41-55.

[17] Gupta, M.M., A Survey of Some Second-order Difference Schemes for the Steady-State Convection-Diffusion Equation. *Int. J. Numerical Methods in Fluids*, Vol. 3(4), 1983, pp. 319-331.

[18] Siyyam, H.I., Ford, R.A. and Hamdan, M.H., First-Order Accurate Finite Difference Schemes for Boundary Vorticity Approximations in Curvilinear Geometries, *Applied Math. and Computation*, Vol. 215, 2009, pp. 2378-2387.

[19] Alharbi, S.O. and Hamdan, M.H., High-Order Finite Difference Schemes for the First Derivative in Von Mises Coordinates, *Journal of Applied Mathematics and Physics*, Vol. 4, 2017, pp. 524-545.

[20] Burova, I.G., Application Local Polynomial and Non-polynomial Splines of the Third-order of Approximation for the Construction of the Numerical Solution of the Volterra Integral Equation of the Second Kind, *WSEAS Transactions on Mathematics*, Vol. 20, 2021, pp. 9-23.

[21] Rubio, D., Tarzia, D.A. and Umbricht, G.F., Heat Transfer Process with Solid-Solid Interface: Analytical and Numerical Solutions, *WSEAS Transactions on Mathematics*, Vol. 20, 2021, pp. 404-414.

Creative Commons Attribution License 4.0 (Attribution 4.0 International, CC BY 4.0)