11th Australasian Fluid Mechanics Conference
University of Tasmania, Hobart, Australia
14-18 December 1992

APPLICATION OF DUAL STREAM FUNCTIONS TO THE VISUALISATION OF THREE DIMENSIONAL FLUID MOTION

G.D. MALLINSON and D.N. KENWRIGHT

Department of Mechanical Engineering
University of Auckland
Private Bag 92019, Auckland, NEW ZEALAND

ABSTRACT

Mass conservation in a steady compressible or incompressible flow can be represented by expressing the mass flux vector as the cross product of the gradients of two scalar stream functions. Surfaces on which either function is constant are stream surfaces and lines on which both functions are constant are stream lines.

New techniques based on these functions have been developed for constructing stream lines from discretely defined velocity or mass flux fields. The advantages these techniques have over conventional methods are that mass is conserved and that the stream lines can be advanced over a mesh cell in a single step rather than arbitrarily selected time-steps. The new algorithms can be at least an order of magnitude faster than conventional methods.

INTRODUCTION

Although stream lines and particle paths are fundamental to flow visualisation, their use by computational fluid dynamicists is often hindered by excessive computational requirements and inaccuracies which cause unrealistic effects such as false spiralling and stream lines which pass through solid boundaries. These 'consistency' defects can be overcome if the interpolations used to generate continuous velocity fields represent exactly the underlying law of mass conservation (Mallinson 1988). High order integration schemes using small time steps contribute to the computational overheads. Direct construction of a line over large sections, such as mesh cells, of the flow can reduce these overheads.

For a two-dimensional steady flow, the stream function (which doesn't have to be used during the solution process) can be used to generate stream lines, a contour map of the stream function providing a visualisation of the whole field. Recognising that the stream function is an integral or potential of the velocity field, the research reported herein sought to find scalar potential functions which could be applied to three-dimensional flows.

THE MECHANICS OF STREAM LINE CONSTRUCTION

The task of constructing stream lines reduces to finding the lines of a vector field. The most general case considered herein is steady compressible flow where the mass flux vector field is solenoidal,

$$\nabla.(\mathbf{q}) = 0 : \qquad \mathbf{q} = \rho\mathbf{v} \tag{1}$$

where ρ is the density and $\mathbf{v}$ the velocity vector. For incompressible flows ρ is constant and can be removed from (1). For generality, the techniques described in this paper will be applied to the construction of the lines of a general solenoidal vector field, $\mathbf{q}$.

It has been shown (Mallinson 1988) that the essential arguments can be presented in terms of cartesian coordinates without undue loss of generality. Accordingly, $\mathbf{q}$ can be represented by its three cartesian components, q_1, q_2 and q_3 and the lines of $\mathbf{q}$ are the solutions to the equations,

$$\frac{dx}{q_1} = \frac{dy}{q_2} = \frac{dz}{q_3} . \tag{2}$$

By the introduction of an integration variable, s say, (2) can be represented by,

$$\frac{d\mathbf{r}}{ds} = \mathbf{q}(\mathbf{r}) \;:\; \mathbf{r} = x\mathbf{i} + y\mathbf{j} + z\mathbf{k} \tag{3}$$

so that given an initial point, $\mathbf{p}$,

$$\mathbf{r}(s) = \mathbf{p} + \int_0^s \mathbf{q}(\mathbf{r}(s))ds \tag{4}$$

The integration of (4) is usually done numerically. It is now accepted that a 4th order Runge-Kutta scheme is more than adequate provided the integration step sizes are kept suitably small or controlled adaptively using relevant stream line properties such as curvature.

As far as this discussion is concerned, inaccuracies arising from the numerical integration of (4) are conceded to be small. The integration does however incur computational overheads which motivate its avoidance.

CRITIQUE OF EXISTING METHODS

Generally, the algorithms in current visualisation packages differ primarily by the interpolations used to generate a continuous $\mathbf{q}$ for equation (4). The most popular interpolation is tri-linear (Eliasson *et al* 1989) and its deficiencies are generally known, (but not necessarily understood). Higher order interpolation schemes, such as cubic splines (Handscome 1983, Mathew and Wilkes 1986) have also been used to ensure that $\mathbf{q}$ is solenoidal and to improve the level of accuracy. The computational overheads are such that the search for good performance low order interpolation schemes is justified.

Tri-linear interpolation of the flux components

Using the notation in Figure 1 tri-linear interpolation can be written as

$$q_i = \{[q_i^0(1-x') + q_i^4 x'](1-y') + [q_i^2(1-x') + q_i^6 x']y'\}(1-z')$$

$$+ \{[q_i^1(1-x') + q_i^5 x'](1-y') + [q_i^3(1-x') + q_i^7 x']y'\}z' \tag{5}$$

$$\text{where} \quad x' = \frac{x - x_c}{\Delta x}; \quad y' = \frac{y - y_c}{\Delta y}; \quad z' = \frac{z - z_c}{\Delta z} . \tag{6}$$

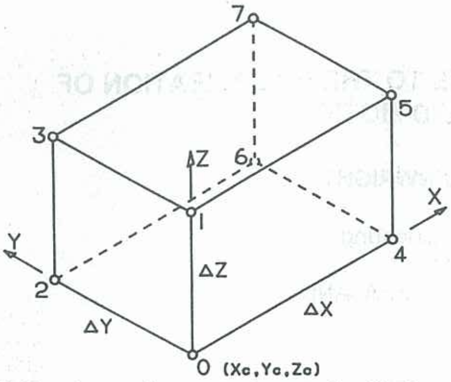

Figure 1. Notation used to represent a mesh cell. The components of **q** (q_i) are located at the corners.

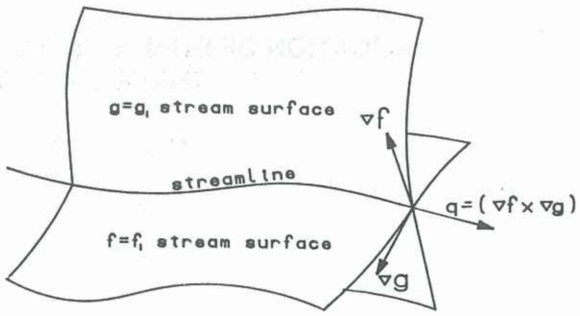

Figure 2. The relationship between the iso-surfaces of two scalar potentials, and a line of vector field.

The superscripts in (5) refer to cell corner locations in Figure 1. An alternative form of (5) is

$$q_i = a_i + b_i x + c_i y + d_i z + e_i xy + f_i xz + g_i yz + h_i xyz . \quad (7)$$

Expanding (5) yields the coefficients in (7).

Tri-linear interpolation provides a 24 degree of freedom interpolation for **q** which has zero order continuity across faces.

<u>Conservation considerations</u>

Flux conservation is implied by the requirement that **q** is solenoidal: calculating the divergence of a **q** represented by (7) produces,

$$(b_1 + c_2 + d_3) + (e_2 + f_3)x + (e_1 + g_3)y$$
$$+ (f_1 + a_2)z + h_1 yz + h_2 xz + h_3 xy = 0 . \quad (8)$$

Equation (8) implies that there are 7 relationships between the corner values of the components of **q** which must be satisfied to ensure that **q** is solenoidal, thereby reducing the degrees of freedom of the interpolation to 17.

A weaker condition is to require only that the total flow of **q** through a cell is conserved which leads to,

$$Q_{1H} - Q_{1L} + Q_{2H} - Q_{2L} + Q_{3H} - Q_{3L} = 0 \quad (9)$$

or $$\Delta Q_1 + \Delta Q_2 + \Delta Q_3 = 0 \quad (10)$$

where $Q_1 = q_1 dydz$; $Q_2 = q_2 dxdz$; $Q_3 = q_3 dxdy$ (11)

and L and H denote low and high faces respectively.

Equation (10) is a weaker requirement than (8) and leads to a single condition which is the same as ensuring that **q** is solenoidal at the centre of the cell only. An interpolation scheme for which q is solenoidal throughout the cell can be said to be *strongly solenoidal*. One satisfying only (9) is *weakly solenoidal*.

It has been suggested before, (e.g. Mallinson 1988), that a non-solenoidal interpolation can lead to consistency defects in stream line tracing. Many of the the test flows used to evaluate stream line tracking algorithms (Mathew and Wilkes 1986) are so simple that the discrete field satisfies equation (8) and the tri-linear interpolation is strongly solenoidal. In fact for one of the most common test problems, that of pure rotation represented by a rectangular mesh, the tri-linear interpolation is an exact solution for the velocity in a cell and the test is too simple to give reliable extrapolations to more general flows.

The issue of whether simply ensuring that an interpolation is weakly solenoidal is sufficient to ensure that stream lines behave consistently will not be resolved by this discussion which explores the roles that potential functions can have in generating strongly solenoidal interpolations.

THE ROLE OF SCALAR POTENTIALS

A solenoidal vector field can be expressed as the cross product of the gradients of two scalar potentials,

$$\mathbf{q} = \nabla f \times \nabla g \ . \quad (12)$$

Since the divergence of the right hand side of (12) is identically zero, the scalar functions can be chosen arbitrarily and **q** will remain solenoidal. The representation has the useful property that **q** is tangential to surfaces on which either f or g are constant. For a mass flux field these surfaces are stream surfaces.

Intersections of f and g iso-surfaces are curves which are tangential to **q** and are thus the lines of **q**. These properties are illustrated in Figure 2 and by the visualisation in Figures 3 and 4 which were generated during the evaluation and testing of the tri-linear f-g algorithm described later in this paper. The scalar functions in these Figures are,

$$f = 4(y - 1)^2 + (x - z)^2 - 2(x + z - 3)^2 + 1 \quad (13)$$

$$g = 4y + 4(x - z)^2 - 5 . \quad (14)$$

The two surfaces, f = 7.5 and g = 6.0 in Figure 3 intersect to form the closed vector line in Figure 4.

The scalar potentials have another very useful interpretation which can be visualised by constructing an f-g diagram which is a plot of one function against the other. Equation (13) can then be interpreted a transformation from three-dimensional space into the two-dimensional space of the f-g diagram. Consider, for example, the flow (Q) of **q** through the z = constant plane in Figure 5.

$$Q = \int_R \mathbf{q}.d\mathbf{a} = \iint_R q_3 dxdy = \iint_R \frac{\partial(f,g)}{\partial(x,y)} dxdy = \iint_{R'} dfdg \ . \quad (15)$$

The flux through the region R on the surface z=constant is numerically equal to the area of the transformation of R in the f-g, diagram. This result can be extended to any closed region on an arbitrary surface in real space.

The lines of **q**, being curves in real space on which both f and g are constant, are points in the f-g diagram. The diagram is therefore a representation of the flow in which all the stream lines have been straightened and aligned so that they are pointing into the page. Stream lines can be tracked by locating real space points which are transformed to the same point in the f-g diagram.

For recirculating flows at least one of the stream functions must be multi-valued, severely limiting the f-g diagram's usefulness as a "whole field" visualisation technique. The diagram can, however, be usefully employed to generate efficient procedures for constructing vector lines through individual cells in a computational mesh.

If **q** represents mass flux, the scalar potentials are the the three dimensional equivalents of the two dimensional stream function and are called dual stream functions.

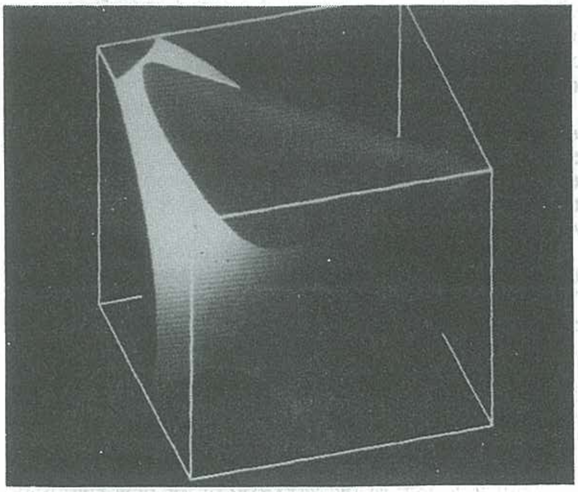

Figure 3. Visualisation of two iso-surfaces for the potentials defined by equations (14) and (15).

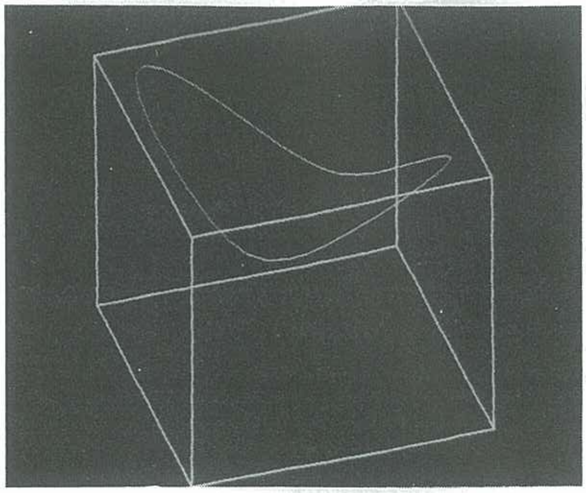

Figure 4. A vector line constructed by finding the intersection of the surfaces shown in Figure 3.

DUAL STREAM FUNCTION METHODS

Mallinson (1988) described a stream line tracing method which relied on the assumption of a simple interpolation for **q** to perform an analytical integration of (3) over a mesh cell. The method was shown to produce consistent results for two-dimensional flows and there was no need to choose integration step sizes. This method is revisited here and compared with an equivalent formulation based on scalar potentials, the purpose being to demonstate analytically the equivalence of the two appraoches.

The interpolation scheme assumed that the flow components for **q** were represented in terms of flow rate, as defined by equation (11), by,

$$Q_1 = Q_{1L} + \Delta Q_1 x'; \quad Q_2 = Q_{2L} + \Delta Q_2 y'; \quad Q_3 = Q_{3L} + \Delta Q_3 z' \quad (16)$$

Equation (2) now separates and can be integrated to yield, for Q_1 say,

$$\ln\left(\frac{Q_1}{Q_1^*}\right) = \frac{\Delta Q_1}{\Delta x}(s - s^*) \quad (17)$$

where Q^*_1 and s^* correspond to an initial point (e.g. entry point into the cell) on the line. The path of a **q** line is governed by,

$$\frac{1}{\Delta Q_1}\ln\left(\frac{Q_1}{Q_1^*}\right) = \frac{1}{\Delta Q_2}\ln\left(\frac{Q_2}{Q_2^*}\right) = \frac{1}{\Delta Q_3}\ln\left(\frac{Q_3}{Q_3^*}\right) \quad (18)$$

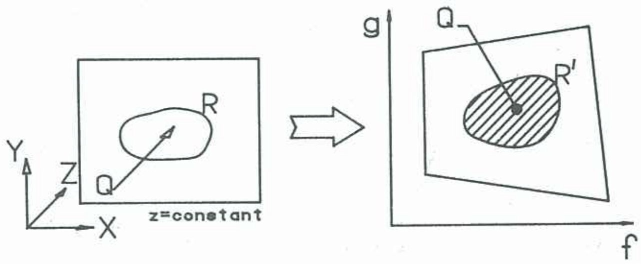

Figure 5. Transformation of the region R on a z=constant plane to R' in the f-g diagram.

By seeking f and g functions of the form

$$f = AQ_1^a Q_2^b Q_3^c; \qquad g = BQ_1^d Q_2^e Q_3^f \quad (19)$$

it can be shown that

$$a + d = 1; \quad b + e = 1; \quad c + f = 1 . \quad (20)$$

Moreover, two of the powers can be chosen arbitrarily. A particularly convenient choice is b=0 and f=0. The solution for this case is,

$$f = \frac{Q_3}{Q_{3L}}\left(\frac{Q_1}{Q_{1L}}\right)^{-\Delta Q_3/\Delta Q_1} \qquad g = -\frac{Q_2 Q_{1L} Q_{2L}}{\Delta Q_2 \Delta Q_3}\left(\frac{Q_1}{Q_{1L}}\right)^{-\Delta Q_2/\Delta Q_1} \quad (21)$$

A line of **q** may be traced by solving,

$$f = f^* ; \qquad g = g^* \quad (22)$$

where f^* and g^* denote f and g values corresponding to an initial point on the line. Equation (21) can be used to evaluate both sides of equation (22) and the result after a little manipulation is equation (19). The two approaches have been demonstrated to be equivalent.

Equation (22) transforms y-z planes through a mesh cell into rectangles in the f-g diagram. Transformations of the x-z and x-y planes have two straight and two curved edges (except for the trivial case where all three components of **q** are constant). This curvature causes some difficulty in the application of (22) in a general purpose stream line tracing algorithm and it is desirable to seek interpolations for the potentials which lead to f-g representations that have straight edges. This led to the tri-linear stream function method described below.

<u>Tri-linear Stream Functions</u>

Starting with tri-linear interpolation formulae (5) for f and g, expressions for the components of **q** can be developed. Matching these expressions to the spatial variation of the components of **q** around a cell is the essential element of the method which is described in detail by Kenwright and Mallinson (1992). The ensuing representation has 11 degrees of freedom, which are adjusted to match the components of **q** and their derivatives at the centres of cells. The number of degrees of freedom is less than the tri-linear velocity interpolation, but the interpolation yields expressions for each

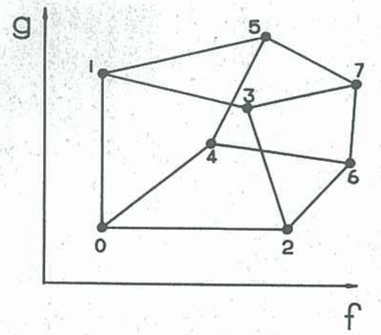

Figure 6. The f-g diagram resulting from application of the tri-linear scalar potential interpolation to the flow through a typical mesh cell.

q_i which are quadratic in the coordinate parallel to the component. The resulting representation of the flow of **q** through a cell has straight edges (Figure 6).

The procedure for constructing **q** lines within computational cells consists of the following steps.

1. For each cell, evaluate the stream functions and construct the representation shown in Figure 6.
2. Given a point within the cell, calculate its f-g image.
3. For that f-g point, determine the faces that surround it. (It is this step which is difficult if the edges of the f-g representation are curved.)
4. Calculate the real space image of the f-g point on the exit face.

Evaluation of the performance tri-linear stream function algorithm

Stream lines produced by the new algorithm, are consistent with the solenoidal condition. It does not produce false spirals for two-dimensional flow and stream lines do not "stick to" or pass through solid boundaries. A typical stream line for a well known flow is shown in Figure 7. Differences between the stream lines produced by this algorithm and those produced by direct integration or tri-linear velocity interpolation - Runge-Kutta integration have been observed, in particular, close to the axes of recirculating flows. Measures for quantifying the differences are being developed.

The computational efficiency of the new method was evaluated by applying it to a test velocity field defined by,

$$q_1 = ax - by; \quad q_2 = bx + ay; \quad q_3 = -2az + c . \qquad (23)$$

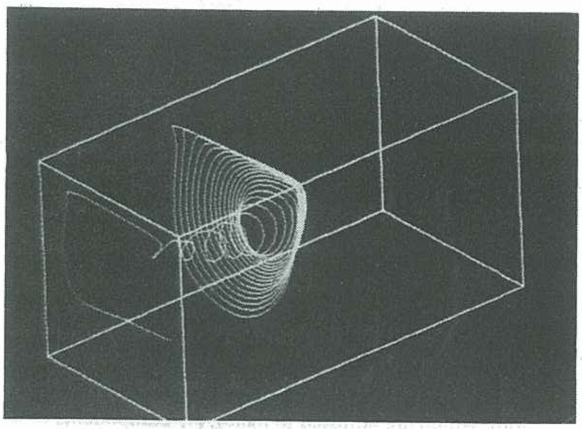

Figure 7. Example stream line constructed using the dual stream function method for flow in a cavity with a sliding top.

Equation (23) was used to produce discrete values throughout a rectangular domain. Results are presented for the case where the stream functions were pre-evaluated and for the case where they where calculated "on the fly" as each cell was encountered.

Usually, the Runge-Kutta integration of the tri-linear velocity interpolation uses adaptive time stepping. The tests reported here used a fixed number of time steps to cross each cell. Typical adaptive procedures use between 1 and 10 time steps per cell with 5 being a representative average. Tests were performed for these three cases.

The times reported in Table 1 are for the calculation of 9 stream lines and were obtained on an IBM 4341 computer. Tabulations are the average of three runs and the relative results are normalised by the time for the new method using "on the fly" evaluation of f and g.

The single time step case indicates that the new algorithm, despite its greater complexity, involves approximately 75% of the work associated with a single step advance through the Runge-Kutta algorithm.

The evaluation of the scalar potentials involves approximately 38% of the workload of the new algorithm. This represents the potential speed gain associated with pre-evaluation which is offset against the storage of at least 11 variables for each cell.

The speed improvements offered by the new algorithm are approximately 4 and 8 when compared with 5 and 10 time steps respectively for the conventional method. The performance of the new algorithm improves slightly as the number of mesh cells increases.

	Stream function		Conventional		
	Stored	"on the fly"	1 step per cell	5 steps per cell	10 step per cell
10x10x10	0.28	0.45	0.58	1.84	3.40
20x20x20	0.86	1.38	1.89	6.01	11.2
20x20x40	1.25	2.05	2.84	9.00	17.4

Table 1. Times taken to produce a test set of stream lines using the dual stream function and conventional methods. Times are given in seconds (IBM4341).

CONCLUSIONS

The use of scalar potentials to represent a vector field within a mesh cell guarantees that the field will be solenoidal. The transformation from real three-dimensional space a two-dimensional scalar potential space leads to the construction of consistent and efficient procedures for constructing the lines of the vector field.

ACKNOWLEDGEMENTS

This research was supported by the New Zealand Vice Chancellors' Committee, the Auckland University Research Grants Committee and by BHP Research and New Technology, Central Research Laboratories, Newcastle.

REFERENCES

ELIASSON, J, OPPLESTRUP, J and RIZZI, A (1989), STREAM 3-D: Computer graphics program for streamline visualization, Advances in Engineering Software, 11, 4, 162

HANDSCOMBE, D C, Spline Representation of Incompressible Flow, (1983), Oxford University Report.

KENWRIGHT, D N and MALLINSON, G D (1992), A 3-D streamline tracking algorithm using dual stream functions, Proc IEEE Visualization '92, Boston, October.

MALLINSON, G D, The calculation of the lines of a three-dimensional vector field, 1988, Computational Fluid Dynamics, Ed. G de Vahl Davis and C A Fletcher, North Holland, 525-535.

MATHEW, M D and WILKES, N S, Particle Tracking for 3-dimensional fluid flow predictions, (1986), AERE (Harwell) Report R12153.