JOURNAL OF COMPUTATIONAL PHYSICS 26, 307–338 (1978)

A Coordinate System for a Viscous Transonic Cascade Analysis

PETER R. EISEMAN*

Theoretical Gas Dynamics Group, United Technologies Research Center, East Hartford, Connecticut 06108

Received August 17, 1976; revised March 22, 1977

A coordinate system suitable for the numerical computation of viscous transonic cascade flows is constructed. The system consists of coordinate loops surrounding the airfoil and radial coordinate lines normal to the airfoil surface. The outermost loop is constructed so that the cascade periodicity conditions can be applied without interpolation between grid points. The coordinates are orthogonal on the airfoil surface but gradually become nonorthogonal away from the airfoil. The coordinate distribution of mesh points is simple and direct; this is a useful property for the resolution of large solution gradients. In addition to the above, the coordinates are generated from discrete input data, little restriction is placed on airfoil camber or spacing, and the entire analysis is easily extended to three dimensions. Moreover, the method of coordinate generation can be readily applied to a wide variety of other problems.

1. INTRODUCTION

An important problem faced by the designer of gas turbine engines is the prediction of the flow field through a cascade of airfoils. The prediction of the flow field depends upon an accurate representation of the cascade geometry. Because of the wide variety of practical cascade geometries, the need for a robust, accurate, and efficient coordinate generation procedure is evident. The coordinate generation procedure developed herein fulfills this need: It can be used as a mesh generator for a finite-element method [1–4], as a mesh generator for a finite-volume method [5], or for the numerical solution of the Navier–Stokes equations in transformed space.

To date, coordinate generation procedures for airfoils or cascades of airfoils have been based upon complex variables [6–9], solutions to elliptic partial differential equations [10–11], and direct geometric constructions [12–17]. Procedures based upon complex variables are usually variants of Theodorsen's method [18], a well-known generalization from the classical Joukowski or Kármán–Treffetz airfoils [19]. Methods based on complex variables are inherently limited in the control of grid spacing and in the restriction to two dimensions. To overcome these limitations, more-general methods based on solutions to elliptic partial differential equations were developed [10–11]. These methods are comparable in efficiency to the methods based upon complex variables. Specifically, coordinate systems were generated from numerical solutions of Poisson equations. The source term, referred to as a

* Present address: Scientific Research Associates, P. O. Box 498, Glastonbury, Connecticut 06033.
Partially supported by Naval Air Systems Command under contract N00019-75-C-0463.

0021-9991/78/0263-0307$02.00/0

forcing function, was used to control grid spacing. On application, however, the coordinates usually become nonorthogonal. Because of this deviation from rigid orthogonality problems, direct geometric constructions have become attractive alternatives. Fundamental advantages are a direct control over grid spacing, an emphasis on the representation of geometric boundaries, and an absence of a complex solution procedure for the entire computational region. This latter point is easily translated into a savings in both computer time and storage.

The simplest and most obvious type of direct geometric construction is the generation of sheared coordinate systems [13–17]. For the cascade problem, however, the shearing process leads to severe mesh distortion, improper mesh distribution, and coordinate singularities in the critical leading and trailing edge regions [13–16]. Unless special treatment were given to these regions, the results from numerical calculations would suffer. The method developed in this paper retains the advantages of a direct construction but removes the above problems of leading and trailing edges.

In addition to the treatment of leading and trailing edges, computational advantages are achieved from a careful parameterization of the bounding surfaces. Through the parameterization, the surface geometry and the distribution of points along the surface are specified. This dual specification leads to a savings in computer time and storage. Since the surface geometry is an invariant of the particular parameterization, the careful choice of parameterization is only for the distribution of points along the surface. The surface distribution is used for the alignment of points between surfaces or between parts of the same surface. For the cascade, the bounding surfaces are the airfoil and the surrounding outer boundary. The entire coordinate system is then generated from a properly aligned boundary specification.

Included in the boundary alignment is the imposition of cascade periodicity conditions between the upper and lower parts of the outer boundary. This removes the need to interpolate boundary conditions. For implicit methods, the periodic alignment is essential since an implicit interpolation would effectively disallow the use of an ADI splitting [20]; hence there would be a severe loss of efficiency. The periodic alignment, however, does not extend to a periodic alignment of the coordinate derivatives. This lack of extension also occurs with the cascade coordinates generated by solutions to elliptic partial differential equations [11]. However, this apparent problem is easily resolved. For finite-difference procedures one-sided differences can be used effectively. For procedures based upon piecewise polynomial spaces (usually via collocation or Galerkin methods), only a pointwise evaluation is needed.

2. A Coordinate System for Cascade Geometries

Overview

A coordinate generation procedure which can be successfully applied to the wide variety of practical cascade problems must accept a set of control parameters and a discrete rendition of the cascade geometry as input data. Controls on coordinate

stretches and distributions are among the needed parameters. Stretching parameters are needed for coordinate extensions in the upstream and downstream directions; distributional parameters, for the resolution of large solution gradients.

The coordinate generation procedure developed herein uses the above input. The discrete rendition of the cascade geometry is given in Cartesian coordinates (x, y) with a vertical y-axis in the direction determined by cascade periodicity. The periodic spacing between airfoils is taken as an increment along the y-axis; the airfoil contour, as a sequence of vertical slices. Each vertical slice consists of an x-coordinate which determines a vertical line $x = x_i$ and two y-coordinates $y = y_i$ and $y = z_i$ to denote points of intersection with the airfoil contour (Fig. 1). The distribution of vertical slices must provide discrete data which can be used to represent the airfoil contour and its curvature accurately. Data fluctuations caused by inaccuracies in measurement will usually occur when graphical data are used. To obtain an accurate rendition of the airfoil contour and associated curvature, any data fluctuations must be removed. For this purpose, it is best to apply a curve-fitting algorithm based upon a paramteric least-squares procedure [21–26]. In the development presented herein, it will be assumed that a least-squares routine is always available to convert discrete descriptions of curves into smooth and differentiable representations of the same curves. If, however, the discrete data are taken from an analytic formulation of an airfoil contour (e.g., [27]), then ordinary splines would suffice [28]. The basic control on coordinate extensions is given by a specification of vectors pointing in the upstream and downstream directions, respectively. Controls on mesh point distributions consist of direct applications of boundary layer resolutions near the airfoil surface and resolutions of the airfoil leading and trailing edges. The latter controls arise from the process of coordinate extension and the distribution of mesh points around the outer computational boundary. The total number of mesh points along the outer computational boundary is broken down into three parts. Specifically, one must specify half the number of periodically aligned points, the number of points for the inflow boundary, and the number of points for the outflow boundary.

With the above input, a system of coordinates is generated in a manner which is a generalization of both polar coordinates and the classical boundary layer coordinates [29, p. 312]. The circles in polar coordinates are now replaced by a family of loops

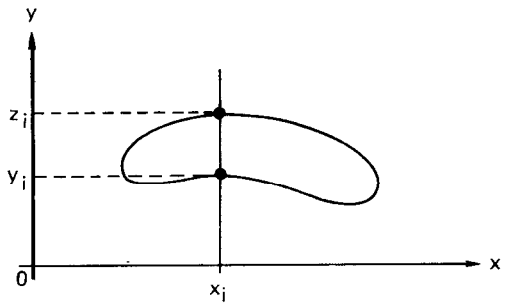

FIG. 1. Input data for the airfoil contour.

about the airfoil which start with the airfoil itself and smoothly deform into the outer boundary of the computational region. The polar radii are replaced by straight lines which emanate from the airfoil surface and end on the outer boundary. As in boundary layer coordinates, the straight lines are taken to be orthogonal to the airfoil surface. However, since the intermediate loops are chosen by an interpolation between the airfoil surface and the outer loop the resultant system of coordinates is generally nonorthogonal. The nonorthogonality of the coordinate system as a whole is of no great concern since the coordinates are nearly orthogonal in the regions where the viscous flow is undergoing its greatest rate of change. Specifically, the coordinates, by construction, are precisely orthogonal along the airfoil surface and gradually deviate from orthogonality as one leaves the airfoil surface. The greatest degree of nonorthogonality occurs in the upstream and downstream regions where free-stream conditions are being approached, and where, therefore, the gradients in the viscous flow field are very small. If the outer loop could be taken as a uniform expansion of the airfoil along its outward normal lines, then the resultant coordinate system would be precisely a set of boundary layer coordinates for the airfoil and accordingly would be orthogonal everywhere. If in addition, the airfoil contour is a circle, then the coordinate system becomes a set of polar coordinates. Since the coordinate system has been constructed for a cascade of airfoils, the outer loop generally cannot be taken as an outward and uniform expansion of the airfoil itself. Instead, the outer loop must be generated from a curve which can conveniently be used for the application of the necessary periodicity conditions in the cascade problem. The basic shape of this curve should be reasonably close to the camber of the airfoil [27]. It will be referred to as a camber curve, hopefully without too much confusion. The generation of the camber curve is accomplished on a discrete level within the airfoil contour and is extended by lines outside of the airfoil. The discrete data can be conveniently generated from the y-values y_i, z_i of each vertical slice $x = x_i$ in the discrete specification of the airfoil contour (depicted in Fig. 1). The discrete data are then made into a differentiable curve which is extended by lines in front of and in back of the airfoil. The camber curve is shown in Fig. 2 as line AB. After a smooth camber curve is created, the domain of the calculation is bounded by two curves each parallel to the camber curve. One curve, line CD, is above the airfoil and the second curve, line EF, is below the airfoil. The curves are separated by a distance equal to the airfoil spacing and are capped off at the upstream and downstream ends by curves which are smoothly joined to form the differentiable outer loop depicted by $CDFE$ in Fig. 2. The outer loop is then reparameterized in a manner which yields a periodic alignment for the mesh points where a periodicity condition must be applied. The next step is to impose the parameterization of the outer loop upon the inner loop by dropping normals onto the airfoil surface. The reparameterization is accomplished by the assignment of the parameter value of each outer loop point to the point on the airfoil contour which has an airfoil normal line passing through the given outer loop point. This is computationally executed on a discrete level and is then made into a smooth curve by a least-squares routine. The inner loop with the imposed parameterization from the outer loop is now properly aligned

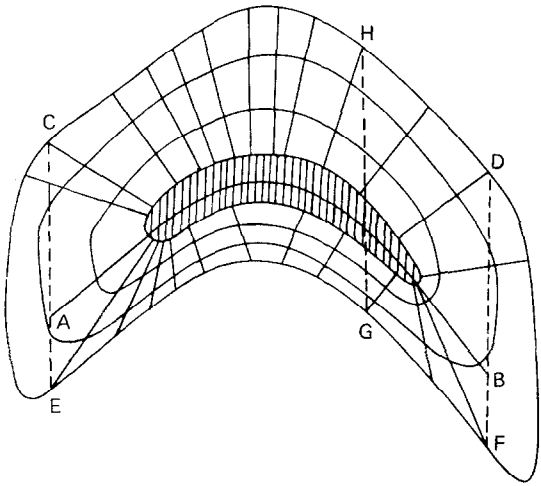

FIG. 2. Schematic of airfoil coordinate system.

so that any line joining inner and outer loop points of the same parametric value results in a line that leaves the airfoil as a normal line. The normal lines form the family of the coordinate curves which correspond to the radii of a polar system. These are illustrated in Fig. 2. The other coordinate curves consist of the loops that are obtained by an interpolation along the above normal lines. The periodic alignment of the resulting coordinate system is illustrated by the line GH, which is represented by a dashed line since it is not a coordinate curve.

The Effect of Airfoil Curvature on the Placement of a Lower Coordinate Boundary

As a first step in the coordinate generation procedure, the discrete airfoil data are converted into a smooth curve by an application of a least-squares algorithm. The curve parameterization is obtained from the cumulative arc length between data points. The result is an accurate fit to data with an (almost) arc-length parameterized curve $\gamma(t) = (\gamma^1(t), \gamma^2(t))$ which has at least two continuous derivatives and accurately reflects the curvature of the contour from which the raw data were taken. The two continuous derivatives are needed to perform the calculation of the curvature, which is given by the formula

$$K = (1/\dot{S}^2)(\ddot{\gamma}^m\ddot{\gamma}^m - \ddot{S}^2)^{1/2}, \tag{1}$$

where a dot indicates a t-derivative, S is the actual arc length along the curve, $\dot{S} = (\dot{\gamma}^m\dot{\gamma}^m)^{1/2}$, $\ddot{S} = \dot{\gamma}^m\ddot{\gamma}^m/\dot{S}$, and the summation convention of summing like indices has been invoked. The analytic arc length, S, and the polygonal arc length, t, are nearly equal since t is an approximation of S. Thus $\dot{S} \simeq 1$, $\ddot{S} \simeq 0$, and as a result $K \simeq (\ddot{\gamma}^m\ddot{\gamma}^m)^{1/2}$. The curvature here is needed to determine the extent of the computational boundary below the airfoil. Since the bottom side of the airfoil is usually

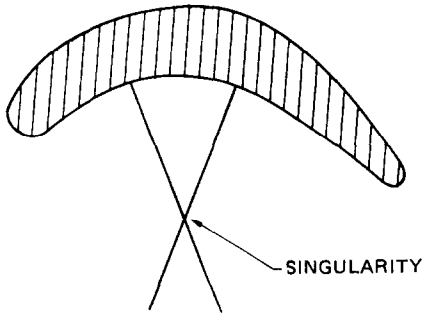

FIG. 3. Singularity from intersecting normals.

concave it is clear that there is a restriction on the distance that the coordinates can extend below the airfoil. Otherwise, the proposed coordinate normal lines would have intersections among themselves when the domain is dropped beyond a certain point. This would cause coordinate singularities as illustrated in Fig. 3. To prevent these singularities, the cascade coordinates are required to lie above all points of possible intersections. The necessary restriction is analytically specified by a knowledge of the centers of the osculating spheres along the concave side of the airfoil. The osculating sphere in two dimensions is the circle which is tangent to the airfoil bottom and is determined by matching its derivatives with the airfoil surface until all of the parameters of the circle are determined. The center is located at a distance of $1/K$ along the airfoil unit normal vector $\hat{n}$ which is given by

$$\hat{n} = [(\dot{\gamma}^m \dot{S} - \dot{\gamma}^m \dot{S})/K\dot{S}^3]\, \hat{u}_m\,, \tag{2}$$

where $\hat{u}_1$ and $\hat{u}_2$ are the standard Cartesian unit vectors along the x- and y-axes, respectively. Thus, the vector position of the center is given by the quantity

$$\gamma + \hat{n}/K, \tag{3}$$

which will trace out a curve below the airfoil as the concave part of the airfoil bottom is traversed. To avoid the singularity, the coordinates must terminate within the region bounded below by this curve (Fig. 4). The bottom of the coordinate system is,

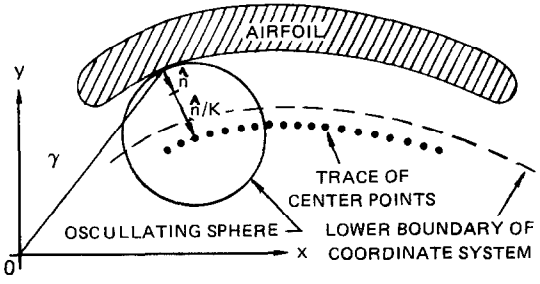

FIG. 4. Determination of lower coordinate boundary.

however, just a lowered version of the camber curve (arc *EF* of Fig. 2). To obtain a safe lower bound on the amount that the camber curve can be lowered, we shall compute half the vertical distance from the airfoil bottom to the centers of the osculating spheres. This should provide enough distance from both the airfoil bottom and the osculating sphere centers. A reasonable distance from the centers is needed to avoid an underresolution of computational mesh points along the airfoil bottom (Fig. 5).

The actual computation shall be accomplished on a discrete level. The analytic curve for the airfoil is discretized by a uniform mesh over its parameterization, and at each of these mesh points, a unit normal vector (Eq. (2)) is computed when possible. At inflection points the curvature vanishes, and the unit normal vector given in Eq. (2) does not exist. Otherwise, the unit normals always exist and point in the direction of curve concavity. This is easily seen from Fig. 4 and the observation that the center of an osculating sphere is in the positive normal direction. From a sequential computation of projections between successive normal vectors (Eq. (2)), an algorithm can be easily obtained to detect airfoil concavity. At points of concavity the centers of the osculating spheres are calculated (Eq. (3)), and their x-coordinates are each determined to lie in some interval resulting from the partition imposed by the x-coordinates of airfoil mesh points. If the interval is directly below the airfoil, then the vertical distance is computed as the average of the y-values at the interval endpoints minus the y-value of the center of the osculating sphere. For an illustration

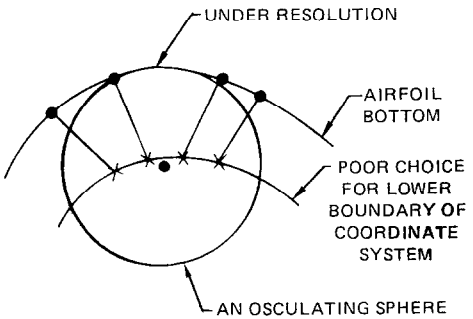

FIG. 5. A coordinate boundary near the center of an osculating sphere.

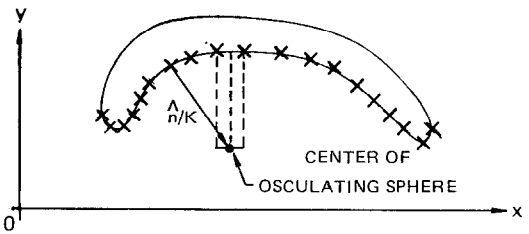

FIG. 6. The vertical distance between the airfoil bottom and the centers of the osculating spheres.

see Fig. 6, where the mesh along the bottom of the airfoil is denoted by a sequence of x's.

As this process continues throughout the concave part of the airfoil bottom, the successive vertical distances are monitored and a minimum is taken. Next, the maximum vertical thickness of the airfoil is computed. With this information, a criterion can easily be constructed to determine the amount which the camber curve is lowered to form the bottom of the computational domain. The difference between the periodic spacing and the maximum thickness of the airfoil is just the smallest vertical distance between consecutive airfoils as their chords are traversed. If one-half of this distance is less than the allowable vertical distance due the concavity restriction above, then the bottom of the computational domain can safely be set at one-half of the periodic spacing distance below the camber curve. The top of the computational region is then one-half of the periodic spacing distance above the airfoil, and the computational domain is bounded from above and below by well-centered curves which are parallel to the camber curve. By contrast, if the inequality is in the other direction, then the camber curve is lowered by one-half of the maximum airfoil thickness plus the distance due to the concavity restriction. This results in a computational domain which is not as well centered about the airfoil as that in the previous case. In either case, however, the coordinates are well defined and properly spaced.

The Construction of the Camber Curve

The above criteria for the vertical displacements of the camber curve can all be generated before the camber curve is constructed. For its application, the camber curve must obviously be in existence, and therefore, shall now be constructed. Since the airfoil data are specified as a sequence of vertical slices $x = x_i$ with y-values $y = y_i$ and $y = z_i$ (which are assumed to be lower and upper surface points, respectively), airfoil camber data are generated by a criterion of the form $x = x_i$, $y = (1 - \alpha_i)\, y_i + \alpha_i z_i$ for $0 < \alpha_i < 1$, as i runs through the vertical slices. For a smooth set of camber data, the sequence α_i must be generated from a continuous function of limited total variation (see Royden [30]). The result of any such choice of function will be a sequence of data points which roughly follow the camber of the airfoil since the data points must lie within the interior of the airfoil. The resulting sequence of data points is first parametrized by polygonal arc length and then fit with a least-squares curve. In this case, accuracy is less important than it was with the airfoil contour. Under the assumption that the curve remains reasonably near the data, the only constraint on accuracy is that the polygonal arc length parameterization provide a reasonable approximation to the analytic arc length of the resultant curve. This part of the camber curve is used to form upper and lower computational

with respect to arc length since it is this mesh distribution which will be used to impose a parameterization over most of the airfoil surface as normal lines are dropped.

A uniform subdivision of the parameter values will then result in a uniform distribution of mesh points over the segments in question on both the airfoil and the surrounding outer loops. An additional bonus is that the linear extensions of the camber curve in upstream and downstream directions are simplified. Since the parameter is almost an arc-length parameter, it has an arc-length derivative which is nearly unity. On the linear extensions, a continuous rate of expansion relative to arc length is desired so that the number of mesh points is conserved as the computational boundaries are stretched. Otherwise, the numerical computation of the viscous flow field would overly resolve the stretched regions, and thus waste a considerable amount of computation time on parts of the flow where no substantial changes are occurring.

Since the linear expansion is to occur smoothly from an existing arc-length parameterization, the arc-length derivative of the parameter must be unity where the extensions are joined to the curve. The direction of each extension is given by the specification of a unit vector in the desired direction. Typical choices of direction may, for example, be selected from the flow conditions or from the global airfoil geometry. More specifically, one may stretch the coordinate system in the free-stream directions of the far-field velocity vectors upstream and downstream of the cascade. Or one may stretch the upstream and downstream extensions in the direction of the airfoil camber curve as it emerges from each end of the airfoil.

For the latter choice, the directions in question are obtained from the unit tangent vectors to the camber curve at leading and trailing edges. Let $\alpha(t)$ for $0 \leqslant t \leqslant t_1$ denote the camber curve between leading and trailing edges. Then the vector field consisting of unit tangent vectors to $\alpha(t)$ is given by

$$\mathbf{v}(t) = d\alpha/dS \simeq d\alpha/dt, \tag{4}$$

where S is the arc length starting from $\alpha(0)$. The approximate equality is a result of the polygonal approximation of t to S. Since the vector field $\mathbf{v}$ points in the direction of increasing arc length along α, the extension in front of the leading edge is in the negative $\mathbf{v}(0)$ direction. Thus, an extension in front of length F is of the form

$$\alpha(0) + S_F(t)\,\mathbf{v}(0), \tag{5}$$

where S_F is the arc length measured from $-F$ to 0 as the parameter t varies from 0 to some T. The value of T will determine the proportionate number of points in front of α relative to the number of points on α. If $\alpha(t)$ is discretized into k points uniformly distributed with respect to t, then the parameter spacing is given by $\Delta t = t_1/(k-1)$. For the extension in front, the greatest integer part of $F/\Delta t$ (denoted $[F/\Delta t]$) is a measure of the number of whole Δt increments that could be fit into the extension if the parameter t of the extension were to approximate arc length. For small extensions, it is desirable to continue the parametric approximation to arc length by a discretization of the extension into the number of parametric intervals just given. However, if the extension is large, a coordinate stretching relative to arc length is

best. Large extensions are often needed to approximate free-stream conditions. Thus, the extension is cut into

$$n_F = \min([F/\Delta t], m_F) \tag{6}$$

units of length Δt where the positive integer m_F is a specified cutoff value. The arc-length function $S_F(t)$ is then to be parameterized from 0 to $T = n_F \Delta t$. At the value T the derivative of S_F is taken to be unity since the extension is to be joined at the resultant point with the nearly-arc-length-parameterized curve α. The desired stretching is readily given by the quadratic arc-length function

$$S_F(t) = (T - t)\left[\left(\frac{1}{T} - \frac{F}{T^2}\right)(T - t) - 1\right], \tag{7}$$

which monotonically increases from $S_F(0) = -F$ to $S_F(T) = 0$ and ends with a slope of $S_F'(T) = 1$. If F is large, the function leaves $-F$ fairly rapidly and on approaching 0 gradually decreases its rate of climb until it is equivalent to arc length. Upon substitution into the expression for the linear extension in front (Eq. (5)), a discretization for $t = 0, \Delta t, \ldots, n_F \Delta t$ yields data points which start with large separations, continually decrease, and end with a separation equivalent to arc length. When the parameters of the discretization of α are each increased by the addition of T units, the result is a discretization of the extension in front continued by the discretization of α with the new parameterization which varies smoothly through the juncture point. Note that the juncture point is produced by both curves but is only counted once in this process. Thus, the discretization consists of $k + n_F$ points, counting endpoints. The last point has a parameter value of $t_2 = t_1 + T$, and a rate of change that is equivalent to arc length. In the same manner as before, an extension of B units in length is added on to form a linear continuation in back of the airfoil trailing edge. The extension is of the form

$$\alpha(t_1) + S_B(t)\, \mathbf{v}(t_1), \tag{8}$$

where S_B is the arc length measured from t_2 to $t_2 + B$, as the parameter varies from t_2 to $t_2 + R$ for some length R. The value of t_2 was chosen instead of t_1 since the eventual discretization will be a continuation of the previous discretizations, and this choice will immediately lead to a smooth continuation of the parameterization. The extension in back is cut into

$$n_B = \min([B/\Delta t], m_B) \tag{9}$$

units of length Δt with an integer cut off value of m_B for a total parameter change of $R = n_B \Delta t$. Unlike the extension in front, however, the extension in back expands from an arc-length parameterization and not into one. This change in direction

results in the requirement that $S_B'(t_2) = 1$. As before, a quadratic stretching function is sufficient, and the result is given by

$$S_B(t) = (t - t_2)\left[\left(\frac{B}{R^2} - \frac{1}{R}\right)(t - t_2) + 1\right], \tag{10}$$

which monotonically increases from $S_B(t_2) = 0$ to $S_B(t_2 + R) = B$ and starts with a slope $S_B'(t_2) = 1$.

The discretization for $t = t_2 + \Delta t, \ldots, t_2 + n_B \Delta t$ yields data points on the linear extension in back which at the start are separated by distances that are equivalent to arc length and then continually increase from there in an opposite fashion to the frontal extension. When this discretization is added onto the end of the prior discretization, a properly parameterized discretization of the entire camber curve with extensions is obtained. The parameterization starts from 0 and ends with a value $t_3 = t_2 + R$. The total number of points in this discretization is given by the sum $n_F + k + n_B$. At this stage, the data points could be fit with smooth curve that is parameterized in correspondence with the given discrete parameterization. Instead, however, it is best to use the above discretization directly to form a properly parameterized discretization of the entire outer loop which encircles the airfoil and forms the outer boundary of the coordinate system. Then the outer-loop discretization will be fit with just one curve-fitting process as opposed to separate curve fits which must be smoothly joined between the upstream and downstream endcaps and the vertically translated camber curves. For an illustration, see Fig. 7, where the constituent parts of the outer loop have been displayed. The camber curve appears as the curve which linearly emerges from the airfoil through its extensions. The vertical translates are then displayed along with the adjoining endcaps. At the expense of a small amount of storage, the discretization of the camber curve is vertically translated above and below the airfoil. This is accomplished in the manner prescribed by the algorithm of the previous section which yields a determination of the vertical distance of translation based upon the airfoil thickness, spacing, and underside curvature.

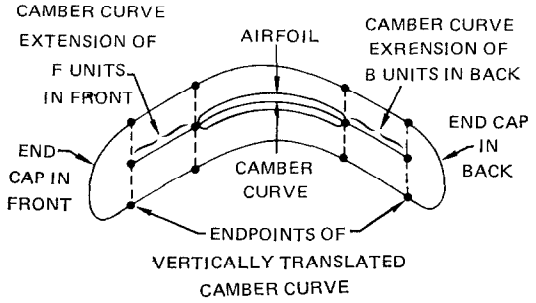

FIG. 7. The constituent parts of the outer loop.

The Construction of Endcaps for the Outer Loop

Let the vertically translated camber curves of the previous section be denoted by $\mathbf{p}(t)$ and $\mathbf{q}(t)$ for the lower and upper parts of the outer loop. As with the camber curve itself, the range of the parameter t will be from 0 to t_3 as the curves are traversed from front to back. To complete the specification of the outer loop, endcaps must be constructed to join the translated camber curves together, at both the front and back ends. For the construction of endcaps, it is sufficient to use a bicubic curve at each end with the stipulation that function values and tangent vectors are matched at the joins. It will be assumed that the parameter values are taken from 0 to some number T. A small value of T will cause the bicubic end to approach a vertical line; large values, a bulge away from the line. The choice, however, is arbitrary and as a matter of convenience T can safely be taken as twice the periodic spacing distance. Each cubic polynomial here is of the form

$$e_0 + e_2 t + \left[3\left(\frac{e_1 - e_0}{T^2}\right) - \frac{e_2}{T}\right] t^2 - 2\left(\frac{e_1 - e_0}{T^3}\right) t^3, \tag{11}$$

where e_0 and e_1 are polynomial values at the respective endpoints 0 and T; e_2 is the slope at 0. For the x-coordinates, the endpoint evaluations are equal due to the vertical alignment of the camber lines. Consequently, $e_0 = e_1$ and the polynomial becomes a quadratic which starts with a slope of e_2 and ends with a slope of $-e_2$, (see Fig. 8a). In the front $e_0 = p_1(0)$ and in the back $e_0 = p_1(t_3)$ where the decomposition $\mathbf{p} = (p_1, p_2)$ has been used. The slope e_2 is given by the x-component of the direction of camber line extension which is $-\mathbf{v}(0)$ for the front and $\mathbf{v}(t_1)$ for the back where $\mathbf{v}(t)$ is given by Eq. (4). The negative slope of $-e_2$ is needed at T since the parameter values are increasing in a direction opposite to the unit vector which points in the direction of camber line extensions. The y-components of these slope conditions are used to evaluate the quantity e_0 for the calculation of the y-coordinates. The

FIG. 8. (a) Quadratic x-coordinate. (b) Cubic y-coordinate.

are then discretized by a sufficiently fine mesh and parameterized by polygonal arc length. When taken with the translated camber lines, the result is a discretized version of the entire outer loop. But one is still not finished, since the parameters must be suitably adjusted to interface with the desired mesh point specification for the fluid dynamic calculation. Suppose that the computational mesh is to have k periodic points along the translated camber lines $\mathbf{p}(t)$ and $\mathbf{q}(t)$; n points along the front endcap, not counting juncture points; and m points along the back endcap, also not counting juncture points. The total computational mesh along the outer loop would then consist of $n + 2k + m$ points, and therefore, that same number of normal lines to the airfoil surface. The parameterization of the translated camber lines $\mathbf{p}(t)$ and $\mathbf{q}(t)$ is the same and varies from 0 to t_3. Since these parameterizations will be preserved up to rigid translations, the interval 0 to t_3 is cut into a uniform mesh with $k - 1$ intervals of length $\Delta t = t_3/(k - 1)$. At the front endcap, $n + 1$ such intervals are needed. Thus, the arc-length parameterization of the front endcap must be replaced by a parameterization from 0 to $(n + 1)\,\Delta t$ which smoothly passes through the juncture points. This is accomplished by a blend of straight lines from each endpoint with slope determined by the existing arc length derivative $\dot{S}$ at those points. This process is illustrated in Fig. 9.

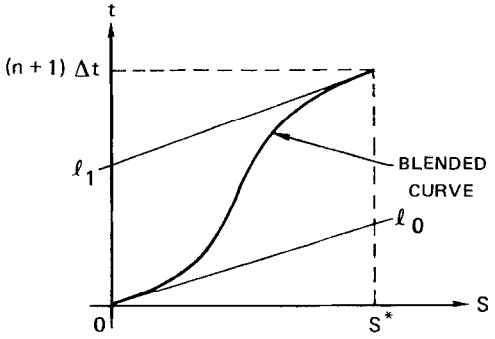

FIG. 9. A blend of lines.

The lines are generated with arc length S as the independent variable which varies from 0 to S^*, the arc length of the endcap on the front. The slope of each line is given by the rate at which the camber curve parameter varies with respect to arc length at the beginning of the camber curve. Thus, one has the two parallel lines

$$l_0(S) = \frac{S}{|S_F'(0)|} \tag{12a}$$

and

$$l_1(S) = \frac{S - S^*}{|S_F'(0)|} + (n + 1)\,\Delta t, \tag{12b}$$

where $S_F(t)$ is given by Eq. (7). The desired blend must start along l_0, gradually

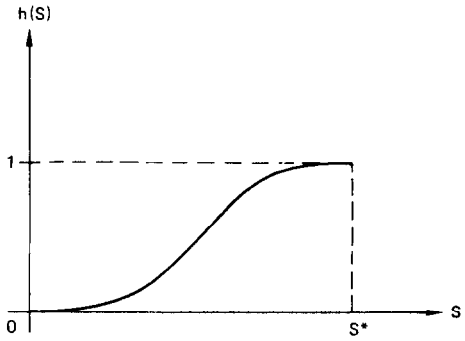

FIG. 10. Homotopy parameter.

leave l_0, and smoothly merge into l_1 to end at $(S^*, (n+1)\,\Delta t)$. This is accomplished with a linear homotopy [31] between l_0 and l_1, a homotopy-deforming parameter given by the function

$$h(S) = \frac{1}{2}\left\{1 + \frac{\tanh(2D/S^*)(S - S^*/2)}{\tanh D}\right\}, \tag{13}$$

and a damping factor D which controls the rate of ascension between the lines (Fig. 10). Altogether, the parameterization for the front endcap is given by the linear homotopy

$$t(S) = [1 - h(S)]\, l_0(S) + h(S)\, l_1(S). \tag{14}$$

Along the upper curve $\mathbf{q}(t)$, the parameterization is shifted by the addition of $(n+1)\,\Delta t$ to each parameter value. The result is a discretized curve with a smooth parameterization covering the front endcap with parameter values from 0 to $(n+1)\Delta t$ and continuing along the top with $(k-1)\,\Delta t$ units to end at a parameter value of $(n+k)\,\Delta t$. At this stage, the endcap at the back is adjoined; in the same manner as above, it is reparameterized to vary smoothly from $(n+k)\,\Delta t$ to $(n+k+m+1)\,\Delta t$, where a juncture occurs with the lower curve $\mathbf{p}(t)$. Then the orientation of the lower curve is reversed by the relabeling of points so that one has the curve $\mathbf{p}((k-1)\,\Delta t - t)$. The resulting parameterization for $\mathbf{p}$ is next, shifted by $(n+k+m+1)\,\Delta t$ units so that a smooth parameterization is properly specified for the entire outer loop. The outer loop is given by a discrete set of points which are parameterized from 0 to $(n+2k+m)\,\Delta t$ as the loop is traversed in a clockwise direction. If desired, the parameterization can be renormalized in preparation for a curve-fitting process. The curve fit here is accomplished with an application of a least-squares procedure. In this process, the outer-loop data are transformed into a smooth curve with three continuous derivatives and the prescribed parameterization. Note that the least-squares procedure will effectively filter out the small smoothness errors that occurred when arc length was approximated with the arc length of a polygonal curve. The small smoothness errors in question appeared as slope information at the juncture

points on each end of the camber curve extensions. Parametric accuracy within the camber curve is not very important, since the periodic alignment of mesh points is not affected by a slight loss of accuracy within that region. On the endcaps, such questions of accuracy would seem more important. However, the construction above was performed in such a manner that the accuracy did not enter into the assignment of parameter values at the endpoints of any endcap. The only effect, then, would be in the specification of slopes for the lines l_0 and l_1 (for each endcap), which would undergo the smoothing of least squares in any event. Consequently, the alignment of periodic points will be very accurate.

The Reparametrization of the Airfoil Surface

After the properly parameterized outer loop is constructed, the airfoil surface must be reparameterized to align airfoil parameter values with outer-loop parameter values so that corresponding points lie on the same airfoil normal line. Once the reparameterization has been accomplished, the coordinate transformation will be given by the equation

$$\mathbf{x} = R(y^2)\,\boldsymbol{\alpha}(y^1) + [1 - R(y^2)]\,\boldsymbol{\beta}(y^1), \tag{15}$$

where $\mathbf{x} = (x^1, x^2)$ are Cartesian coordinates, $\boldsymbol{\beta}$ is the airfoil contour, $\boldsymbol{\alpha}$ is the outer loop, R is a coordinate distribution function along the normals, and $\mathbf{y} = (y^1, y^2)$ are curvilinear coordinates. The coordinate, y^2, is the position along normal lines, and y^1 is the position around the outer loop which is to be imposed upon the airfoil surface, and hence, upon all intermediate coordinate loops. The reparameterization is accomplished in a discrete manner. A sufficiently dense uniform mesh is used to discretize the outer-loop parameter, and hence, to create a smooth sequence of outer-loop points with their smoothed parameter values. From each of these points, a normal line must be dropped to the surface of the airfoil The simplest way to find the desired airfoil normal line is to locate the point on the airfoil surface which is closest to the outer-loop point in question. For each outer-loop point, this distance minimization problem is always solvable since the airfoil surface is either locally convex or is locally concave with the centers of the osculating spheres placed (by construction) a sufficient distance beyond the outer loop. When the airfoil point of minimum distance is located, it is assigned the parameter value of the outer-loop point. The process is then continued to the next point on the outer loop until all data points on the outer loop have been used. The result is a discrete reparameterization of the airfoil surface which can be turned into a smooth curve in either of two ways. First, the airfoil may be recreated by treating the given airfoil data as raw data and directly applying the curve-fitting routine. If the reparameterization should cause enough distortion relative to arc length, then a curve fit to the change of parameter relationship should be considered as a second and alternative method. Specifically, new and old parameter values must be paired off and then subjected to a monotone curve fit [32–33]. The reparameterized airfoil is then given by the composition with the old parameterization expressed as a function of the new parameter-

ization. Consequently, the airfoil geometry remains invariant; the accuracy and rendition of the airfoil geometry is then preserved.

An integral part of the reparameterization is the method used to drop the normals. For reasons of simplicity and stability, the algorithm is based on the minimization of distance, as indicated above. The outer-loop mesh points are consecutively taken in the clockwise ordering of the parameterization. The lower juncture point between the front endcap and the lower camber curve is the starting point, $\alpha(0) = (\alpha_1(0), \alpha_2(0))$. From this point the distance, d, to a selected airfoil data point within the leading edge region is computed. The search for minimum distance is performed over the existing airfoil data points with Cartesian x-coordinates less than $\alpha_1(0) + d$. This criterion limits the search to a region around the leading edge (Fig. 11). The location of the airfoil data point of minimum distance is then used to start the search algorithm used on the remaining points. The algorithm starts with a known previous position. For the first point, this position is the location determined above. For other outer-loop points, the previous position is the existing airfoil data point just before the point on the airfoil determined by the normal line dropped from the previous outer-loop point. Since outer-loop points are taken in a clockwise order, the previous point on the airfoil is simply the existing airfoil data point which is nearest to the point in question when distance is measured only in the counterclockwise direction. From here, a distance is computed and the search over existing data is continued until the measured distance exceeds the starting distance. This process limits the search of existing data points to a small region on the surface of the airfoil, and thus, saves computer time. For an illustration, see Fig. 12. The mesh on the outer loop is denoted by a sequence of dots; the existing airfoil data, a sequence of x's. The distance, d_1, to the previous airfoil data point is measured along a line (dashed in the figure) which generally intersects the previous normal line unless the normal line emanates precisely from an existing airfoil data point. The search is continued

[redacted]

illustration, the search would result in the selection of the point of minimum distance from the first three points pictured. After the point of minimum distance has been found from the existing airfoil data, the analytic formulation of the airfoil contour is used to create new data for a refined search in a small neighborhood of the point.

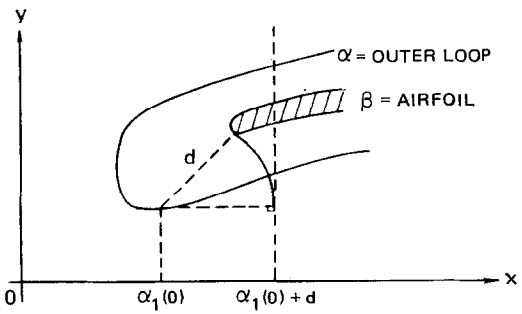

FIG. 11. Region of search for first point.

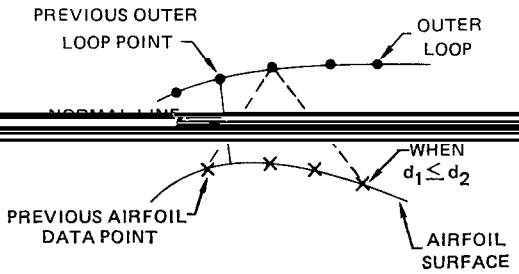

FIG. 12. Criterion to limit the search of existing airfoil data.

The simplest procedure is to search for a point of minimum distance over a nest of discretizations. If the previous airfoil data point is the point of minimum distance, then the initial region of the nest need only cover the surface from that point to the nearest point in the clockwise direction. This is because a move in the counter-clockwise direction would only increase the distance. In the other cases, the initial region is determined by the existing data points on either side of the minimum point. In any case, the initial region is determined by two points which correspond to the end points of a parametric interval. A uniform discretization of this interval is used to start the nest. New data points corresponding to the uniform parameter values are then created, and a search for minimum distance is made. At the point of minimum distance, this process is repeated with the mesh points viewed in the same capacity as the existing airfoil data points of the first step. For succeeding intervals, however, the corresponding surface patch must contain the minimum points in its interior. The continuation of this process results in a nesting of surface patches that contain the analytically defined point of minimum distance. This point is, however, a member of a sequence of points which are ordered in a clockwise direction in correspondance with the clockwise ordering of the outer loop of data points. Consequently, the nesting above should be continued until, at the very least, the parametric interval contains only the parametric value of the analytically defined minimum point in question. This process will ensure that the clockwise ordering of the outer loop sequence will be preserved by the new sequence of airfoil data points determined by the approximate normal lines. The outer-loop parametrization is now discretely given as a monotone function of airfoil arc length; hence, a discrete parametrization of the airfoil surface has been obtained. As a matter of practice, the nesting above is usually continued only slightly beyond the level which ensures monotonicity of the parameter function. Further, nesting beyond this point would certainly result in more accuracy. However, when the accuracy is of the same order as the curve-fitting accuracy, there is little need to continue the search to greater perfection. An intrinsic advantage of this procedure is that accuracy is automatically increased in regions where airfoil normals are heavily concentrated, while other regions are adequately resolved with a lesser level of accuracy. Specifically, the density of normals tends to be greatest on leading and trailing edges; as a result, those regions are more highly resolved without incurring the cost of resolving the other regions which do not require the same high level of

accuracy. In the above procedure, the interval nesting is easily replaced by virtually any sufficiently robust method of optimization (e.g., the method of Hooke and Jeeves [34]). However, for the one-dimensional airfoil surface considered in this application, the use of another optimization technique is not warranted. On the other hand, if the natural three-dimensional extension of the coordinate construction presented herein is to be done, then the method of search becomes more important since the search is over a two-dimensional surface with the result that the potential number of searching directions has increased. In that case, a different optimization technique should be considered.

Distribution Functions along Normal Lines

In regions where the expected solution gradients are large, the approximate solution may be very poor unless the particular region is dissected into smaller regions. In fluid mechanics, the boundary layer of a viscous flow around or through an object is such a region. The process of dissection is commonly accomplished by the use of a coordinate distribution function.

Within the framework of cascade coordinate systems, boundary layer resolution near the airfoil surface is accomplished with the distribution

$$R(y^2) = my^2 + (1 - m)\left[1 - \frac{\tanh D(1 - y^2)}{\tanh D}\right]. \tag{16}$$

Here, the ratio of hyperbolic tangents is a homotopy parameter [31] in the linear deformation of the line $R = my^2$ into the line $R = m(y^2 - 2) + 2$. The rate of deformation is controlled by a damping factor D; this determines the length of essential adherence to the line $R = my^2$. The slope m is chosen so that the resulting line would yield a uniform mesh which is fine enough to resolve the given boundary layer region. A good choice is to let m be the ratio of the fractional part of the flow occupied by the boundary layer to the percentage of the mesh that is needed there.

The distribution is, in fact, a generalization of the distribution due to Roberts [35]. For n mesh points and a constant $a > 0$, his distribution is given in inverse form as a map from a physical domain $[-a, a]$ into a computational range $[1, n]$. From an inversion and a normalization of the range and domain to the interval $[-1, 1]$, his distribution reduces to a normalized hyperbolic tangent. A rigid translation then corresponds to our special case with $m = 0$. As a result, the damping factor is the only control on the shape of his distribution. For purposes of boundary layer resolution, the shape must depend upon an estimate of boundary layer growth. The estimate is generally a function of distance along the body surface. In his case, the estimate is used in a complex and indirect manner to specify the damping factor. By our introduction of a slope adjustment, a direct use of the estimate is possible and is, indeed, an advantage both geometrically and computationally.

3. The Generation of the Navier–Stokes Equations with the Metric Data for Cascade Coordinates

The efficient generation of metric data is an important part of any solution procedure involving general curvilinear coordinates. Before a solution can be undertaken, the physical problem must be specified. Problem specification, however, involves the creation of boundary and initial data and the generation of the equations of motion with the associated boundary conditions. In addition, the solution may be monitored, examined, or put under physical constraints. In all of these tasks, the metric data are needed. A knowledge of the metric data is enough to specify completely the equations of motion and to analyze the coordinate invariant directions for the specification of boundary and initial conditions. For very complicated geometries, the equations of motion may contain an inordinate number of terms. However, if the equations are taken in tensor form, then the coefficients to terms can be constructed from the metric data, with the construction process performed on a computer. Once a nontrivial term is constructed, its contribution to the desired difference equations is computed before searching for the next nontrivial term. Sequentially, the process continues until all terms in the equations have given their contributions to the system of difference equations. Then, in the same fashion, one cycles through terms in the boundary conditions, sequentially adding in their respective contributions. The result is the desired set of difference equations, and the problem is effectively reduced to linear algebra. Note that with such methods there is no real need to write out the differential equations or complicated boundary conditions in detail. Thus, all one needs to do is to generate the metric data and use them.

The coordinate transformation from cascade coordinates into Cartesian coordinates is given by Eq. (15) in the previous section. By differentiation, one obtains the Jacobian transformation which leads directly to the transformation rules for tensor fields. These rules allow one to input, monitor, or extract basic information from a solution procedure involving transformed variables. The Jacobian transformation is essentially obtained from the chain rule which yields

$$\mathbf{e}_i = \frac{\partial \mathbf{x}}{\partial y^i} = \frac{\partial x^j}{\partial y^i}\frac{\partial \mathbf{x}}{\partial x^j} = \frac{\partial x^j}{\partial y^i}\hat{u}_j , \tag{17}$$

where $\hat{u}_j$ is the standard orthonormal basis of constant vector fields, and $\mathbf{e}_j$ is the natural basis of tangent vectors to coordinate curves. With a slight abuse of notation, $\mathbf{x}$ has been used as a position vector in the definitions of $\mathbf{e}_j$ and $\hat{u}_i$. However, nothing is lost since the covariant derivative of $\mathbf{x} = x^j\hat{u}_j$ is just the partial derivative of the x^j summed on $\hat{u}_j$. In the standard Cartesian basis $\hat{u}_i$ the outer loop and the airfoil contour are expressed in the forms $\boldsymbol{\alpha} = \alpha^i\hat{u}_i$ and $\boldsymbol{\beta} = \beta^i\hat{u}_i$, respectively. In this notation, the transformation for cascade geometries is given in component form by the equations

$$x^i = R(\alpha^i - \beta^i) + \beta^i \tag{18}$$

for $i = 1, 2$. By differentiation the Jacobian transformation is given by

$$\mathbf{e}_1 = \left[R\left(\frac{d\alpha^i}{dy^1} - \frac{d\beta^i}{dy^1}\right) + \frac{d\beta^i}{dy^1}\right]\hat{u}_i \tag{19a}$$

and

$$\mathbf{e}_2 = \left[\frac{dR}{dy^2}(\alpha^i - \beta^i)\right]\hat{u}_i\,. \tag{19b}$$

The metric tensor g_{ij} is obtained from the differential element of arc length $(ds)^2 = g_{ij}\,dy^i\,dy^j$. From the known Cartesian form and an application of the chain rule, the differential element of arc length is expanded through the sequence of equalities.

$$(ds)^2 = d\mathbf{x}\cdot d\mathbf{x} = \left(\frac{\partial \mathbf{x}}{dy^i}\,dy^i\right)\cdot\left(\frac{\partial \mathbf{x}}{\partial y^j}\,dy^j\right) = \frac{\partial \mathbf{x}}{\partial y^i}\cdot\frac{\partial \mathbf{x}}{\partial y^j}\,dy^i\,dy^j = (\mathbf{e}_i\cdot\mathbf{e}_j)\,dy^i\,dy^j \tag{20}$$

and, as a result, the metric is given by $g_{ij} = \mathbf{e}_i\cdot\mathbf{e}_j$.

The $\mathbf{e}_j$-direction covariant derivative D_j of the vector $\mathbf{e}_i$ is again a vector, and hence, is expressible in terms of the same basis $\mathbf{e}_1$, $\mathbf{e}_2$. Specifically,

$$D_i\mathbf{e}_j = \Gamma^m_{ij}\mathbf{e}_m\,, \tag{21}$$

where the coefficients Γ^m_{ij} are known as Christoffel symbols. This covariant derivative measures the rate of change of $\mathbf{e}_j$ along a coordinate curve in the direction of $\mathbf{e}_i$. This coordinate curve is an integral curve of $\mathbf{e}_i$ which is obtained by fixing all except the ith variable in the transformation.

The assumption will be made that the covariant derivative is the natural one derivable from the metric. This is known as the Levi–Civita connection [36]. The Christoffel symbols for this covariant derivative are given by the formula

$$\Gamma^k_{ij} = \frac{g^{km}}{2}\left\{\frac{\partial g_{mj}}{\partial y^i} + \frac{\partial g_{mi}}{\partial y^j} - \frac{\partial g_{ij}}{\partial y^m}\right\}, \tag{22}$$

where the g^{km} are elements of the matrix inverse to the matrix of metrics (g_{ij}). This formula is easily obtained by differentiating $g_{ij} = \mathbf{e}_i\cdot\mathbf{e}_j$ with respect to y^m, permuting all three of these indices, forming the sum in parentheses, applying symmetry to the lower indices of the Christoffel symbols, and then applying the inverse metric. With some calculation, one can obtain the nonzero Christoffel symbols directly from the above formula.

For the automatic computation of the metric data, it is convenient to use forms

$$g_{ij} = \frac{\partial x^l}{\partial y^i}\frac{\partial x^l}{\partial y^j}\,. \tag{23}$$

If A denotes the Jacobian matrix $(\mathbf{e}_1, \mathbf{e}_2)$ with transpose A^t, then it is easy to see that

$$g = \det(g_{ij}) = \det(A^t A) = \det(A^t)\det(A) = (\det A)^2 = J^2, \tag{24}$$

where J is used to denote the Jacobian of the transformation. For nonsingular transformations, J is nonzero; hence, both A and (g_{ij}) are invertible. Thus, the inverse metric is obtained from

$$(g^{km}) = (g_{ij})^{-1} = (A^t A)^{-1} = A^{-1}(A^t)^{-1} = A^{-1}(A^{-1})^t, \tag{25a}$$

which is converted into components to yield

$$g^{km} = \frac{\partial y^k}{\partial x^l}\frac{\partial y^m}{\partial x^l}. \tag{25b}$$

The Christoffel symbols can now be obtained by a direct substitution into Eq. (22); the result is the simple form

$$\Gamma^k_{ij} = \frac{\partial y^k}{\partial x^l}\frac{\partial^2 x^l}{\partial y^i\,\partial y^j}, \tag{26}$$

which is suitable for automatic computation.

In terms of arbitrary metric data, the governing equations are derived from the Navier–Stokes equations. For velocity $\mathbf{v} = v^i\mathbf{e}_i$, density ρ, total energy per unit volume E, temperature $T = T(\rho, E, \mathbf{v})$, thermal conductivity K, and pressure $p = p(\rho, E)$; the metric version of the governing system of equations is given by

$$\frac{\partial}{\partial t}(\rho g^{1/2}) + \frac{\partial}{\partial y^i}(\rho v^i g^{1/2}) = 0, \tag{27a}$$

for continuity;

$$\frac{\partial}{\partial t}(Eg^{1/2}) + \frac{\partial}{\partial y^i}\left[\left\{Ev^i - g^{ij}K\frac{\partial T}{\partial y^j} + g_{rj}\tau^{ij}v^r\right\} g^{1/2}\right] = 0, \tag{27b}$$

for energy; and

$$\left[\frac{\partial}{\partial t}(\rho v^k g^{1/2}) + \frac{\partial \sigma^{jk}}{\partial y^j} + \sigma^{ij}\Gamma^k_{ij}\right]\mathbf{e}_k = 0 \tag{27c}$$

for momentum, where $\sigma^{ij} = (\rho v^i v^j + \tau^{ij})\, g^{1/2}$. The quantities τ^{ij} are the components of the stress tensor in the tensor product basis $\mathbf{e}_i \otimes \mathbf{e}_j$. In expanded form, the stress tensor is given by

$$\tau^{ij} = g^{ij}p + a^{ij}_k v^k + b^{ilj}_k \frac{\partial v^k}{\partial y^l}, \tag{28a}$$

where

$$a^{ij}_k = \mu\left(\frac{2}{3}g^{ij}\Gamma^l_{kl} + \frac{\partial g^{ij}}{\partial y^k}\right) \tag{28b}$$

and

$$b_k^{ilj} = \mu \left(\frac{2}{3} g^{ij} \delta_k{}^l - g^{il} \delta_k{}^j - g^{jl} \delta_k{}^i \right) \tag{28c}$$

with viscosity μ and Kronecker deltas $\delta_j{}^i = \delta^{ij} = \delta_{ij}$ [37–39, 17].

If desired, the momentum equation can easily be put into conservation law form. When the expression for the Christoffel symbols given in Eq. (26) is inserted into the momentum equation (27c), one obtains

$$\left[\frac{\partial}{\partial t} (\rho v^k g^{1/2}) + \frac{\partial \sigma^{jk}}{\partial y^j} + \sigma^{ij} \frac{\partial y^k}{\partial x^l} \frac{\partial^2 x^l}{\partial y^i \, \partial y^j} \right] \mathbf{e}_k = 0. \tag{29}$$

A change of basis from the curvilinear direction $\mathbf{e}_k$ into the Cartesian directions $\hat{u}_m$ can be expected to simplify the momentum equation. This is performed by an application of Eq. (17) which yields

$$\left[\frac{\partial}{\partial t} \left(\rho v^k g^{1/2} \frac{\partial x^m}{\partial y^k} \right) + \frac{\partial x^m}{\partial y^k} \frac{\partial \sigma^{jk}}{\partial y^j} + \sigma^{ij} \delta_l{}^m \frac{\partial^2 x^l}{\partial y^i \, \partial y^j} \right] \hat{u}_m = 0. \tag{30}$$

With the assumption of nonmoving coordinates, the Jacobian transformation was brought through the time derivative. Now the definition of the Kronecker symbol is applied and the dummy indices i in the last term are replaced by k's. The result is given by

$$\left[\frac{\partial}{\partial t} \left(\rho v^k g^{1/2} \frac{\partial x^m}{\partial y^k} \right) + \frac{\partial x^m}{\partial y^k} \frac{\partial \sigma^{jk}}{\partial y^j} + \sigma^{jk} \frac{\partial}{\partial y^j} \left(\frac{\partial x^m}{\partial y^k} \right) \right] \hat{u}_m = 0, \tag{31}$$

which, in component form, reduces to the system of conservation laws

$$\frac{\partial}{\partial t} \left(\rho v^k g^{1/2} \frac{\partial x^m}{\partial y^k} \right) + \frac{\partial}{\partial y^j} \left(\sigma^{jk} \frac{\partial x^m}{\partial y^k} \right) = 0. \tag{32}$$

For more information on this topic see Refs. [39–41].

Although the rather formal development above provides a specification of a problem in the cascade coordinate system, it does not provide much insight into the metric structure which is needed to interpret results and to apply boundary conditions properly. For this reason, the metric will be derived in terms of the basic geometric parameters of the cascade. Once this is done, correlations between the metric structure and the underlying coordinates can be made. It is first observed that the cascade coordinate transformation (15) can be broken down into two basic parts. Since $\boldsymbol{\alpha} - \boldsymbol{\beta}$ is a nontrivial normal vector pointing from the airfoil $\boldsymbol{\beta}$ to the outer loop $\boldsymbol{\alpha}$, its magnitude $d = \| \boldsymbol{\alpha} - \boldsymbol{\beta} \|$ is a measure of the distance across the coordinate system in the direction given by the outward unit normal vector from the airfoil. However, the outward unit normal is given by both

$$\hat{n} = (\boldsymbol{\alpha} - \boldsymbol{\beta}) / \| \boldsymbol{\alpha} - \boldsymbol{\beta} \| \tag{33}$$

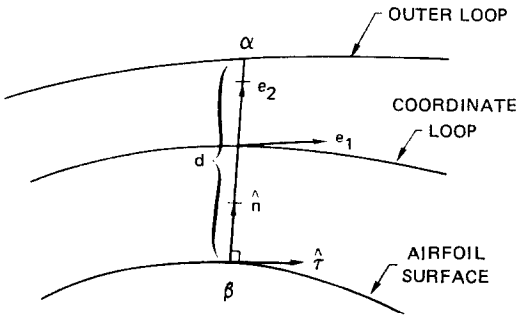

FIG. 13. Intrinsic geometry of the cascade coordinate system.

and the Frenet formulas on the airfoil contour. For the airfoil contour, a unit tangent vector $\hat{\tau}$ is given by $\hat{\tau} = \dot{S}D_1\boldsymbol{\beta}$ where $\dot{S}$ is the y^1-derivative of arc length along the airfoil. Upon successive differentiation, one obtains the Frenet formulas

$$\begin{aligned} D_1\hat{\tau} &= c\dot{S}K\hat{n}, \\ D_1\hat{n} &= -c\dot{S}K\hat{\tau}, \end{aligned} \tag{34}$$

where $c = -1$ on convex parts of the airfoil and $c = 1$ on concave parts. At inflection points, however, the formulas do not exist [37]. Consequently, the coordinate transformation can be written in the form

$$\mathbf{x} = Rd\hat{n} + \boldsymbol{\beta}, \tag{35}$$

with the unit normal vector given by either of the above specifications. At noninflection points, the latter specification shall be used so that the Frenet formulas can be employed to some advantage. Since the coordinate transformation is constructed from functions each of only one variable, derivatives of these functions can be denoted with a dot and result in no ambiguities. In this notation, transformation (35) is differentiated to obtain the natural basis of tangent vectors to coordinate curves. From an application of the Frenet formulas, the result becomes

$$\begin{aligned} \mathbf{e}_1 &= R\dot{d}\hat{n} + \dot{S}(1 - cKRd)\,\hat{\tau}, \\ \mathbf{e}_2 &= \dot{R}d\hat{n}. \end{aligned} \tag{36}$$

For an illustration of the vector relationships, see Fig. 13. Since the vectors $\hat{\tau}$ and $\hat{n}$ are orthonormal, the metric is readily obtained from a direct substitution into the equation $g_{ij} = \mathbf{e}_i \cdot \mathbf{e}_j$. The result is given by

$$\begin{aligned} g_{11} &= (R\dot{d})^2 + [\dot{S}(1 - cKRd)]^2, \\ g_{12} &= g_{21} = R\dot{R}d\dot{d}, \\ g_{22} &= (\dot{R}d)^2, \end{aligned} \tag{37}$$

and from the determinant of this metric one obtains the Jacobian

$$J = g^{1/2} = d\dot{R}\dot{S}(1 - cKRd). \tag{38}$$

The magnitude of the Jacobian, however, is a measure of the relative scaling of coordinate volume elements throughout the domain of the transformation. If the Jacobian is zero at a point, then the differential volume element there is zero and the transformation is singular. Since the Jacobian is a continuous function, one may also examine the coordinates as a singularity is approached. With the cascade coordinates presented herein, a singularity can occur only if one of the factors in the expression of the Jacobian should vanish. However, each of these possibilities will lead to an unreasonable system of coordinates. The factors $\dot{R}$ and $\dot{S}$ can be eliminated from consideration since both R and S must be given by strictly monotone functions, and therefore, cannot vanish. A lack of monotonicity here would cause the coordinates to locally double back upon themselves, and thus, render local regions where the coordinates are not uniquely defined. This leaves one with two possible factors that could vanish. First, if d should vanish, then the airfoil surface and the outer loop would coincide at the point or points in question. As the points of coincidence are approached, the coordinate loops are then smoothly compressed into a region of zero cross section, and hence, smoothly approach the singularity. The second possibility for a singularity would occur if the last factor $(1 - cKRd)$ should vanish. This, however, could only occur in the region of a concave part of the airfoil, since otherwise the factor is the sum of positive quantities. But in the region of a concave part of the airfoil, the centers of the osculating spheres were sent outside of the coordinate system by construction; the analytic implication is that $Kd < 1$; hence the factor cannot vanish.

The rate of change along coordinate curves is measured by the covariant derivatives of the natural coordinate tangent vectors. In this regard, the Christoffel symbols contain the desired information. For example, an application of the covariant derivative D_2 to $\mathbf{e}_2$ yields

$$D_2\mathbf{e}_2 = \ddot{R}d\hat{n} = (\ddot{R}/\dot{R})\,\mathbf{e}_2\,, \tag{39}$$

and hence, the Christoffel symbols

$$\Gamma^1_{22} = 0 \qquad \text{and} \qquad \Gamma^2_{22} = \ddot{R}/\dot{R} \tag{40}$$

by observation from Eq. (21). This result is also partially evident from the basic geometry. The curves of constant y^1 are just the normal lines; hence, any variation of their tangent vectors must be in magnitude only. This conclusion is born out from the analytic fact that Γ^1_{22} vanishes. In the special case of a uniform distribution of loops, the function R is given by $R = y^2$. The second derivative vanishes with the result that Γ^2_{22} also vanishes. Then $D_2\mathbf{e}_2 = 0$ which implies that $\mathbf{e}_2$ is constant along its normal line. The other Christoffel symbols can be computed in the same manner.

Since derivatives along the airfoil surface are involved, it is convenient first to invert system (36), which yields

$$\begin{aligned} \hat{n} &= (1/\dot{R}d)\,\mathbf{e}_2\,, \\ \hat{\tau} &= (\dot{R}d/J)\,\mathbf{e}_1 - (R\dot{d}/J)\,\mathbf{e}_2\,. \end{aligned} \tag{41}$$

By differentiation of the second equation in system (36), one obtains

$$\begin{aligned} \Gamma^m_{12}\mathbf{e}_m = D_1\mathbf{e}_2 &= \dot{R}\dot{d}\hat{n} + \dot{R}d\dot{\hat{n}} \\ &= \dot{R}\dot{d}\hat{n} - c\dot{S}K\dot{R}d\hat{\tau}, \end{aligned} \tag{42}$$

where the last equality follows from Frenet formulas (34). By a substitution of system (41) into the Eq. (42) one obtains

$$\Gamma^m_{12}\mathbf{e}_m = \dot{R}dA\mathbf{e}_1 + (\dot{d}/d + RdA)\,\mathbf{e}_2\,, \tag{43}$$

where $A = c\dot{S}K\dot{R}d/J$. From linear independence, the Christoffel symbols of the left-hand side are just the coefficients displayed on the right-hand side. With some algebraic simplification they are given by formulas

$$\Gamma^1_{21} = \Gamma^1_{12} = \frac{-cK\dot{R}d}{1 - cKRd}$$

and (44)

$$\Gamma^2_{21} = \Gamma^2_{12} = \frac{\dot{d}}{d}\frac{1}{1 - cKRd}\,.$$

By the evaluation of $D_1\mathbf{e}_1$ in the above manner, one obtains

$$\Gamma^1_{11} = \frac{\partial}{\partial y^1}(\log J) - \Gamma^2_{12}$$

and

$$\Gamma^2_{11} = \frac{g_{12}}{g_{22}}\left\{\frac{\partial}{\partial y^1}(\log \dot{d}) - \Gamma^1_{11}\right\} + \frac{c\dot{S}K}{g^{11}J}\,.$$

This completes the evaluation of Christoffel symbols, and hence, the metric data.

The metric data, above, are applicable to a class of coordinate systems. Each member of the class is generated by linear interpolation between two nonintersecting surfaces and is orthogonal along one of the surfaces. This includes cascade coordinates (developed in Section 2), boundary layer coordinates [29], and polar coordinates. The metric data for boundary layer coordinates over a convex body are obtained by setting $c = -1$ and $r = Rd$; the metric data for polar coordinates, by setting $c = -1$, $\dot{S} = b$, $K = b^{-1}$, and $r = Rd + b$, where b is some fixed inner radius.

4. Results

To evaluate the algorithm for the generation of cascade coordinate systems described in the previous sections, several test cases were devised. For the first case, the chief criterion was to obtain a test case which was complicated enough to simulate a real cascade, and yet specialized enough so that comparisons could be made with known geometric parameters. Since most real cascades are known to be composed of highly cambered airfoils, it was required that the first test case be for a cascade with a highly bent airfoil. In addition, since the cascade coordinates are generated from raw data, the test case was constrained to a problem where the airfoil curvature was known. In this way, the geometric representation of the airfoil could be evaluated for accuracy in both location and curvature. Since circles are curves with known

the underlying arc. With the above criteria, the airfoil was constructed from two concentric arcs of slightly different radii which were closed by smaller circular arcs attached to each end. An illustration is given in Fig. 14.

The two concentric arcs were constructed with an inner radius R_1 and an outer radius R_2. The center point was taken at $x_4 = 0$ and the arcs extended through angles from $\pi/4$ to $3\pi/4$ radians. To form a closed loop, smaller arcs of radius $r = (R_2 - R_1)/2$ were attached to either end. These arcs were centered at the Cartesian locations $(\pm x, x)$ with $x = (R_1 + r)/2^{1/2}$. To express the data in terms of vertical slices, the airfoil was subdivided into five regions where a unique analytic description was available. The regions are marked off by the dashed vertical lines in the figure. When the endcap angles α and the angle θ_1 for the inner concentric arc are discretized by a uniform mesh, the result is a collection of vertical slices which discretely define the airfoil contour. For the test case, the central interval $x_3 < x \leqslant x_5$ was partitioned by 29 verticals determined by a uniform partition of $\pi/4 \leqslant \theta_1 < 3\pi/4$. The other intervals $x_l < x \leqslant x_{l+1}$ for $l = 1, 2, 5, 6$ were similarly partitioned with 9 verticals apiece resulting from subdivisions of the angles α on either endcap. In addition,

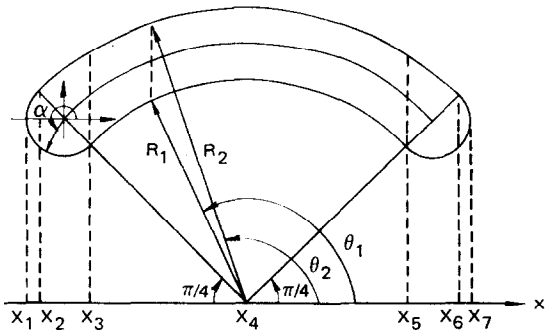

Fig. 14. Cascade airfoil for first test case.

a vertical slice was added at x_1. The inner radius R_1 was given a value of unity; the outer radius, the value of 1.2. Coordinate stretches were specified by setting smooth camber curve extensions of 0.75 in the upstream and downstream directions. The inner part of the camber curve was, for simplicity, given as vertically averaged data with a specification for a biparabolic fit. Next, the periodic spacing of airfoils was given a value of 0.75. The scaling of the coordinates should then be roughly 3 units across and 1 unit high. Consequently, absolute errors should be about twice as large as relative errors. The number of computational mesh points on the outer loop was set by choosing 36 periodic points above and below the airfoil and 15 points on both the upstream and downstream endcaps. The radial distribution was set for a boundary layer region to occupy one-quarter of the distance from the airfoil to the outer loop and to be resolved with one-half of the mesh points. For aesthetice reasons, 10 radial points were chosen so that there would be 8 inner coordinate loops. The generation of the coordinate system and three of its derivatives was executed with a computation time of slightly less than 30 seconds on a UNIVAC 1110. Since most of the computation time went into the construction of the bounding surfaces, one need only store those surfaces and regenerate the coordinates when needed. If, in addition, the radial distribution function is stored, then the regeneration of the coordinate system and three of its derivatives is accomplished with only 48 arithmetic operations per grid point. Regeneration of the coordinate transformation alone would take only 6 operations per point. A computer-generated graph of the results appears in Fig. 15. The airfoil contour was fit with a maximum absolute error of 4×10^{-3} in the location of points. The curvature along the concentric arcs was generally accurate to within two or three digits while the larger curvature regions

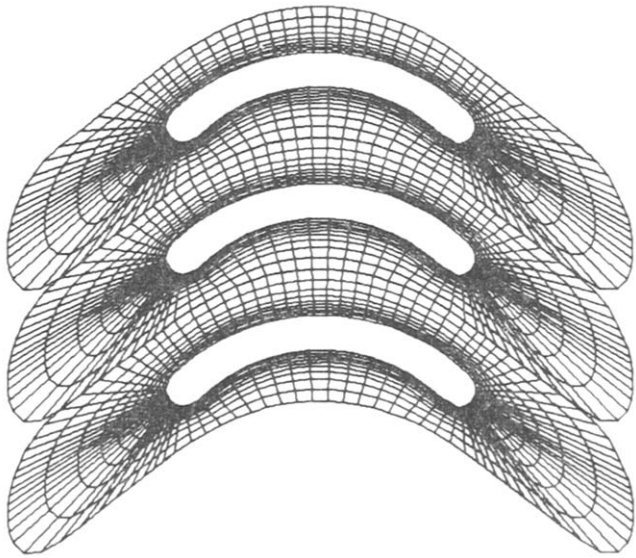

FIG. 15. Three cycles of the coordinate system generated in the first test case.

on the leading and trailing edges were accurate to within only one or two digits. The periodic alignment for periodically matched points was generally accurate to three decimal places and in some places had even greater accuracy. Certainly such excellent results cannot be visually discerned from the graph itself. However, it can be observed that the lines from the airfoil to the outer loop are for all practical purposes normal to the airfoil, and again, the result is in excellent agreement with the theory. Also the radial distribution, as expected, properly distributed the 8 inner loops.

The second test case was constructed to display a coordinate system for a practical cascade of airfoils. For this purpose, measured data were taken from a typical jet engine turbine cascade. The cascade geometry for this case consists of the vertical slices displayed in Table I and the airfoil spacing of 16.5 units. The camber curve was generated within the airfoil by a biparabolic curve joined by linear segments of slope 0.0 in front and 2.4 in back. Smooth extensions outside of the airfoil contour

TABLE I

Vertical Slices of the Airfoil Contour

x_i	y_i	z_i	x_i	y_i	z_i
2.500	2.000	2.000	9.100	0.320	5.200
2.505	1.890	2.110	9.800	0.460	5.500
2.510	1.770	2.230	10.60	0.700	5.920
2.550	1.600	2.400	11.40	0.980	6.400
2.600	1.460	2.540	12.30	1.350	7.010
2.650	1.350	2.650	13.20	1.800	7.710
2.700	1.250	2.750	14.20	2.500	8.650
2.800	1.100	2.900	15.10	3.320	9.600
2.900	0.920	3.020	16.00	4.300	10.70
3.000	0.890	3.110	16.80	5.400	11.80
3.100	0.800	3.200	17.60	6.530	12.92
3.200	0.740	3.260	18.30	7.730	14.00
3.300	0.680	3.320	19.00	9.100	15.20
3.400	0.620	3.380	19.60	10.50	16.30
3.500	0.580	3.400	20.20	11.95	17.20
3.600	0.550	3.430	20.70	13.25	18.30
3.700	0.520	3.470	21.20	14.60	19.25
3.800	0.500	3.500	21.60	15.70	20.02
3.900	0.490	3.520	22.00	16.70	20.80
4.000	0.470	3.540	22.30	17.55	21.35
4.100	0.460	3.580	22.60	18.40	21.95
4.200	0.420	3.600	22.80	19.05	22.33
4.300	0.410	3.620	22.90	19.36	22.55
4.400	0.400	3.640	23.00	19.70	22.75
4.500	0.390	3.680	23.10	20.00	22.95
4.700	0.370	3.720	23.20	20.28	23.08
4.900	0.320	3.780	23.30	20.60	23.13
5.200	0.300	3.850	23.40	29.90	23.18
5.500	0.290	3.930	23.50	21.20	23.20
5.900	0.270	4.050	23.60	21.50	23.18
6.300	0.230	4.180	23.70	21.80	23.13
6.700	0.220	4.300	23.80	22.15	23.08
7.200	0.200	4.450	23.83	22.33	22.95
7.800	0.200	4.670	23.86	22.50	22.85

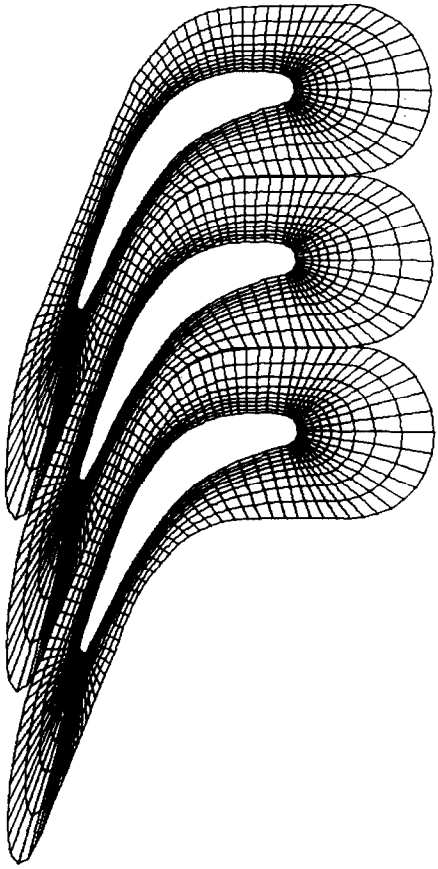

FIG. 16. Three cycles of the coordinate system for a turbine cascade.

were 5.0 units in front and 10.0 units in back. The loopwise computational mesh was set for 40 periodic points on top and bottom, 15 points in front, and 15 points in back. The radial distribution and mesh were the same as in the first case. In about 30 second of UNIVAC 1110 time, the coordinate system and all three of its deviations were generated. The geometric results appear in the computer plot of Fig. 16. As in the first test case, the previous level of accuracy was retained for the curve fits (absolute error of 9.6×10^{-8}), the periodic alignment (to three or more decimal places), and the airfoil curvature (which varied smoothly and reflected deviations from flatness).

An additional point can be made with regard to the tapered trailing edge which in this case is rounded. Since it is a smaller region with roughly the same amount of turning as in the leading edge, the concentration of normal lines is greater. If the size of the trailing edge were to be limited toward a cusped trailing edge, then the concentration of normal lines would be increased with the limit being a local polar-

like set of lines emanating from the cusp. Since the algorithm for generating the normal lines is based on distance minimization from outer loop points, the limit case of a cusp can be stably computed. If the airfoil is to be a coordinate curve (independent of the selection of coordinate system), then a cusp will lead to a coordinate singularity. The only possible removal of this singularity is to round the cusp with a small curve so that the result is a good approximation to the geometry and is sufficiently smooth.

In a wider sense, the cusp is a special case of a discontinuity in the tangent vector field along a coordinate curve. Specifically, the airfoil, as a coordinate curve, had a tangent discontinuity at the cusp. Since it is possible, however, to design an airfoil shape with a similar discontinuity elsewhere, consideration must be given to its treatment. As in the case of a cusp, it can be formally removed by a rounding process. Also, as in the case of a cusp, the inclusion of such a discontinuity will always lead to a coordinate singularity. This fact is not at all peculiar to the present method of coordinate system construction, but is common to all coordinate systems which contain the airfoil contour as a coordinate curve. In particular, the coordinate systems generated from elliptic partial differential equations [10, 11] or from a shearing process [13–17] are also singular in the same location and for the same reason. For the present method of construction and for methods based upon elliptic partial differential equations, the singularity occurs only on the airfoil surface. However, for sheared coordinate systems, the singularity is propagated across the entire coordinate system. To illustrate this matter, we shall consider a simple example. Suppose that the top of the airfoil contour contains a section of the form $(y^1, -|y^1|)$ while the outer coordinate boundary is locally given by $(y^1, 1)$. A uniform sheared coordinate system is then given by $(x^1, x^2) = (y^1, (y^2 - 1)|y^1| + y^2)$ for $0 \leqslant y^2 \leqslant 1$. Since the derivative of the absolute value $|y^1|$ does not exist at $y^1 = 0$, the singularity at (0, 0) is propagated all along the coordinate line $y^1 = 0$. By contrast, if the present method of coordinate construction is applied to the same local bounding curves, then the singularity at (0, 0) is not propagated into the interior of the coordinate system. Instead, a local polar-like coordinate system is smoothly imbedded into the overall system with no interior singularities.

One may now recall that coordinate singularities occur at points where the Jacobian transformation (Eq. (17)) does not have a well-defined inverse. This is equivalent to a Jacobian (Eq. (24)) which either vanishes or is undefined at the given points. For the cases with discontinuous tangent vectors, the Jacobian is undefined. But from Eq. (24), the Jacobian is directly proportional to $g^{1/2}$, which prominently appears in the Navier–Stokes equations (Eq. (27)). In this respect, a coordinate singularity leads to a vanishing or undefined Jacobian, and hence, to a degenerate representation of the Navier–Stokes equations at the singularity.

REFERENCES

1. O. C. ZIENKIEWICZ AND D. V. PHILLIPS, *Int. J. Numer. Meth. Eng.* **3** (1971).
2. B. N. IRONS, *Inst. J. Numer. Meth. Eng.* (1970).
3. I. ERGATOUDIS, B. M. IRONS, AND O. C. ZIENKIEWICZ, *Int. J. Solid Struct* **4** (1968).
4. G. FIX AND G. STRANG, "An Analysis of the Finite Element Method," Prentice–Hall, Englewood Cliffs, N.J., 1973.
5. R. W. MACCORMACK AND A. J. PAULLAY, *Computers and Fluids* **2** (1974), 339–361.
6. A. JAMESON, *Comm. Pure Appl. Math.* **27** (1974).
7. F. BAUER, P. GARABEDIAN, D. KORN, AND A. JAMESON, "Supercritical Wing Sections," Vol. II, Lecture Notes in Economics and Mathematical Systems No. 108, Springer–Verlag, New York, 1975.
8. D. C. IVES AND J. F. LIVTERMOZA, Analysis of transonic cascade flow using conformal mapping and relaxation techniques, AIAA Paper No. 76–370, Ninth Fluid and Plasma Dynamics Conference, 1976.
9. O. L. ANDERSON, "User's Manual for a Finite-Difference Calculation of Turbulent Swirling Compressible Flow in Axisymmetric Ducts with Struts and Slot Cooled Walls," USAAMRDL-TR-74-50, Vol. 1, 1974.
10. J. F. THOMPSON, F. C. THAMES, AND C. W. MASTIN, *J. Computational Physics* **15** (1974), 299.
11. U. GHIA AND K. N. GHIA, "Numerical Generation of a System of Curvilinear Coordinates for Turbine Cascade Flow Analysis," University of Cincinnati Report No. AFL75-4-17, 1975.
12. A. RIZZI AND H. BAILEY, Finite-volume solution of the Euler equations for steady three-dimensional transonic flow, *in* "Proceedings of the Fifth International Conference on Numerical Methods in Fluid Dynamics," Springer–Verlag, New York, 1976.
13. D. A. OLIVER AND P. SPARIS, "Computational Aspects of the Prediction of Multidimensional Transonic Flows in Turbomachinery," NASA SP-347, 1975.
14. R. A. DELANEY AND P. KAVANAGH, "Transonic Flow Analysis in Axial-Flow Turbomachinery Cascades by a Time-Dependent Method of Characteristics," ASME Paper No. 75-GT-8, 1975.
15. S. GOPALAKRISHNA AND R. BOZZOLA, "A Numerical Technique for the Calculation of Transonic Flows in Turbomachinery Cascades," ASME Paper No. 71-GT-42, 1971.
16. J. ERDOS, E. ALZNER, P. KALBEN, W. MCNALLY, ANS S. SLUTSKY, "Time-Dependent Transonic Flow Solutions for Axial Turbomachinery," NASA SP-347, 1975.
17. T. GAL-CHEN AND R. C. J. SOMERVILLE, *J. Computational Physics* **17** (1975), 209–228.
18. T. THEODORSEN, "Theory of Wing Sections of Arbitrary Shape," TR No. 411, NACA, 1931.
19. L. M. MILNE-THOMSON, "Theoretical Hydrodynamics," Macmillan, New York, 1960.
20. J. DOUGLAS AND J. E. GUNN, *Numer Math.* **6** (1964), 428–543.
21. L. L. SCHUMAKER, Fitting surfaces to scattered data, *in* "Approximation Theory" (G. G. Lorentz, C. K. Chui, and L. L. Schumacher, Eds.), Vol. II, pp. 203–268, Academic Press, 1976.
22. P. D. PATENT, *SIAM Numer. Anal.* **13** (1976), 344–361.
23. C. DE BOOR AND J. R. RICE, "Least Squares Cubic Spline Approximation. I. Fixed Knots," CSD TR No. 20, Purdue University, 1968.
24. C. DE BOOR AND J. R. RICE, "Least Squares Cubic Spline Approximation. II. Variable Knots," CSD TR No. 21, Purdue University, 1968.
25. C. L. LAWSON AND R. J. HANSON, "Solving Least Squares Problems," Prentice–Hall, Englewood Cliffs, N.J., 1974.
26. M. P. EPSTEIN, *SIAM Numer. Anal.* **13** (1976), 261–268.
27. I. H. ABBOTT AND A. E. VON DOENHOFF, "Theory of Wing Sections," Dover, New York, 1949.
28. J. J. AHLBERG, E. N. NILSON, AND J. L. WASH, "The Theory of Splines and Their Applications," Academic Press, New York, 1967.
29. H. ROUSE, *et al.*, "Advanced Mechanics of Fluids," Wiley, New York, 1959.
30. H. L. ROYDEN, "Real Analysis," Macmillan, New York, 1963.
31. S. T. HU, "Homotopy Theory," Academic Press, New York, 1959.

32. C. DE BOOR, Splines as linear combinations of *B*-splines: A survey, *in* "Approximation Theory" (G. G. Lorentz, C. K. Chui, and L. L. Schumaker, Eds.), Vol. II, pp. 1–49, Academic Press, 1976.
33. L. RAYMON, Comonotone approximation and piecewise monotone interpolation, *in* "Theory of Approximation With Applications" (A. G. Law and B. N. Sahney, Eds.), pp. 220–237, Academic Press, 1976.
34. J. R. WALSH, "Methods of Optimization," Wiley, New York, 1975.
35. G. O. ROBERTS, Computational meshes for boundary layer problems, *in* "Proceedings of the Second International Conference on Numerical Methods in Fluid Dynamics," p. 171, Springer–Verlag, New York, 1971.
36. D. LAUGWITZ, "Differential and Riemannian Geometry," Academic Press, New York, 1965.
37. I. S. SOKOLNIKOFF, "Tensor Analysis," Wiley, New York, 1964.
38. A. C. ERINGEN, "Nonlinear Theory of Continuous Media," McGraw–Hill, New York, 1962.
39. P. R. EISEMAN, "The Numerical Solution of the Fluid Dynamical Equations in Curvilinear Coordinates," AFWL-TR-73-172, August 1973.
40. H. OSBORN, *Ann. of Math.* **69** (1959).
41. P. R. EISEMAN AND A. P. STONE, *Proc. Amer. Math. Soc.* **53**, No. 1 (1975).