PROC. N.S. INST. SCI (1986)
Volume 36, pp. 115-126

ERROR BOUND STRATEGIES FOR A NEW HULL PLATE EXPANSION PROCEDURE

JOHN C. CLEMENTS
Department of Mathematics
Statistics and Computing Science
Dalhousie University
Halifax, N.S.

and

JULES D. GRIBBLE
Maritime Life Assurance Company
Halifax, N.S.

An improved version of a recently developed (Clements and Leon, 1986) isometric mapping procedure is presented. This algorithm employs a variable step differential equation solver (ODES) in conjunction with an adaptive quadrature routine (AQR). Criteria for choosing appropriate error tolerances for the numerical implementation of the algorithm are presented. An application is briefly discussed.

Une version améliorée d'un procédé d'application isométrique développé récemment est presentée. Cet algorithme utilise une routine de résolution d'équations différentielles a étape variable avec une routine adaptive de quadrature. Les critères à considérer en choisissant les tolérances admissibles pour l'application numérique de cet algorithme sont discutés. Une application est brièvement discutée.

Introduction

The availability of fast, economical computing capability provides small manufacturing operations with the potential for substantially automating and improving existing design procedures and production techniques. This development has permitted the direct application of advanced mathematical methods and results to the solution of problems which hitherto have been avoided because of the computational complexity involved. The implementation of this mathematical analysis in the form of computer programs yields what is often referred to as scientific or engineering software.

In the construction of steel ship hulls (Fig 2), for example, the standard procedure is to loft (from preliminary design plans) the vessel at full scale often using an oak stick called a spline to manually fair or "smooth" each hull line. The frames are constructed from the offset data obtained, set vertically in place and tack-welded together. The steel plate is then measured, cut and welded to the frames in sections. Sometimes the hull plating is difficult and time consuming since one cannot predict if the plate can be shaped to conform exactly to each frame member with simple unidirectional bending. This is particularly true in the areas of large curvature such as the bow where the accepted practice is to patch with small triangular plate sections.

Clearly there is a requirement for a computer system which could use preliminary hull design data to mathematically define and fair all hull lines; determine if the resulting hull surfaces are developable and, if not, automatically modify the existing hull lines to ensure developability; generate graphic displays of waterlines, buttock lines and frame stations and use these to compute elementary hydrostatics, and to generate detailed tables of offsets for both the required frame members and the developable hull surfaces mapped (expanded) onto a flat plane surface (See Fig 3).

This last step eliminates the full scale lofting and manual fairing and permits the precutting from a plane pattern of single hull plate sections of 50 feet or more in length and ensures that the plate will fit at all points of each frame member with unidirectional bending. The use of developable hull forms in shipbuilding offers the advantages of lower cost and faster and simpler construction techniques (Clements and Leon (1986) and Clements (1981)). Indeed, the isometric mapping of developable surfaces or "plate expansion" is a basic problem in many other manufacturing and construction programs. The engineering system specifications listed were met by SYSTEM DEVHULL (Clements 1981) which consists of nineteen component programs and approximately 10,000 lines of FORTRAN source code. In this paper the mathematical analysis and its application will be discussed for a specific improvement to the plate expansion component of the system.

In Differential Geometry a surface **S** in three dimensional Euclidean space $\mathbf{R}^3$ is called a ruled surface if it contains a one parameter family of straight lines called generators or ruling lines **r** which can be chosen as coordinate curves on the surface. A developable surface is a ruled surface defined by nonintersecting generators which has the same tangent plane at all points of each generator (Fig 1). We shall be concerned here with developable surfaces which can be represented in the form

$$\mathbf{S}(s, t) = \mathbf{f}(s) + t\mathbf{r}(a), \qquad a \leq s \leq b, 0 \leq t \leq 1.$$

where **f** and **r** are twice continuously differentiable vector functions in $\mathbf{R}^3$. The term "developable" refers to the property that by a succession of small rotations about each of the generating lines the surface can be laid flat or developed onto a plane without stretching or tearing. That is, it can be mapped isometrically (and isogonally) onto a subset of $\mathbf{R}^2$. Conversely, a plane surface material can be shaped into a developable surface with only simple unidirectional bending along the generating lines.

In Clements and Leon (1986) and Clements (1984) a fast and accurate algorithm was derived to accomplish isometric mapping based on the relationship between the ruling lines $\mathbf{r}(s)$ generating the developable surface **S** and one additional geodesic $\mathbf{g}(s)$ constructed within the surface as the solution of the nonlinear second order ordinary differential equation.

$$\ddot{\mathbf{g}}(s) \cdot (\mathbf{g}(s) x \mathbf{n}) = 0, a \leq s \leq b, \tag{1}$$

where s is the real parameter, $\cdot$ and x are the usual scalar and vector products respectively, $\dot{} \equiv d/ds$, and **n** is the unit normal to **S** at $\mathbf{g}(s)$. The mapping procedure m: r $\rightarrow$ **R** is defined in terms of the ruling line lengths $\|\mathbf{r}(s)\|$, the arclengths along $\mathbf{g}(s)$ and the angles of intersection of $\mathbf{r}(s)$ and $\mathbf{g}(s)$. Since $\mathbf{g}(s)$ is a geodesic in the developable surface **S**, its image under the isometric mapping must be a straight line in the plane and, for simplicity, is taken to be the positive real x—axis (Fig 1). Each ruling line is then mapped into the corresponding plane coordinate line **R** in the developed surface.

The solution process requires the numerical solution of the coupled integro-differential equation system consisting of (1) and

$$I_k = {}^{s_k}\int_{s_{k-1}} \|\dot{g}(s)\| ds, \; k = 1, \ldots, N, \tag{2}$$

where $P_N = \{a = s_0 < s_1 < \ldots < s_N = b\}$ is a partition of [a, b] chosen initially by the user to conform to display, cutting or other requirements. Equations (1) and (2) are coupled since each time a quadrature routine is called to approximate the integral in (2), the ordinary differential equation (1) must be solved for $\dot{\mathbf{g}}(s)$ at those points in $[s_{K-1}, s_K]$ required by the quadrature routine.

The practical difficulty lies in choosing the error tolerances, ε_{DK} and ε_{DK}, to be assigned to the numerical routines used to compute the solutions of (1) and (2) respectively so that it is assured that $|\|\dot{\mathbf{g}}(s)\| - J_N| < \varepsilon_T$ for some specified total absolute error tolerance ε_T. That is, which will ensure that the maximum error in the determination of the total length of $\mathbf{g}(s)$ is less than some user specified tolerance ε_T.

The object here is to derive a rational and efficient strategy for the choice of ε_{QK} and ε_{DK} in the solution of (1) and (2), to describe an improved version of the mapping algorithm (Clements 1984) incorporating both a variable stepsize differential equation solver (ODES) and an adaptive quadrature routine (AQR) and to discuss briefly an application.

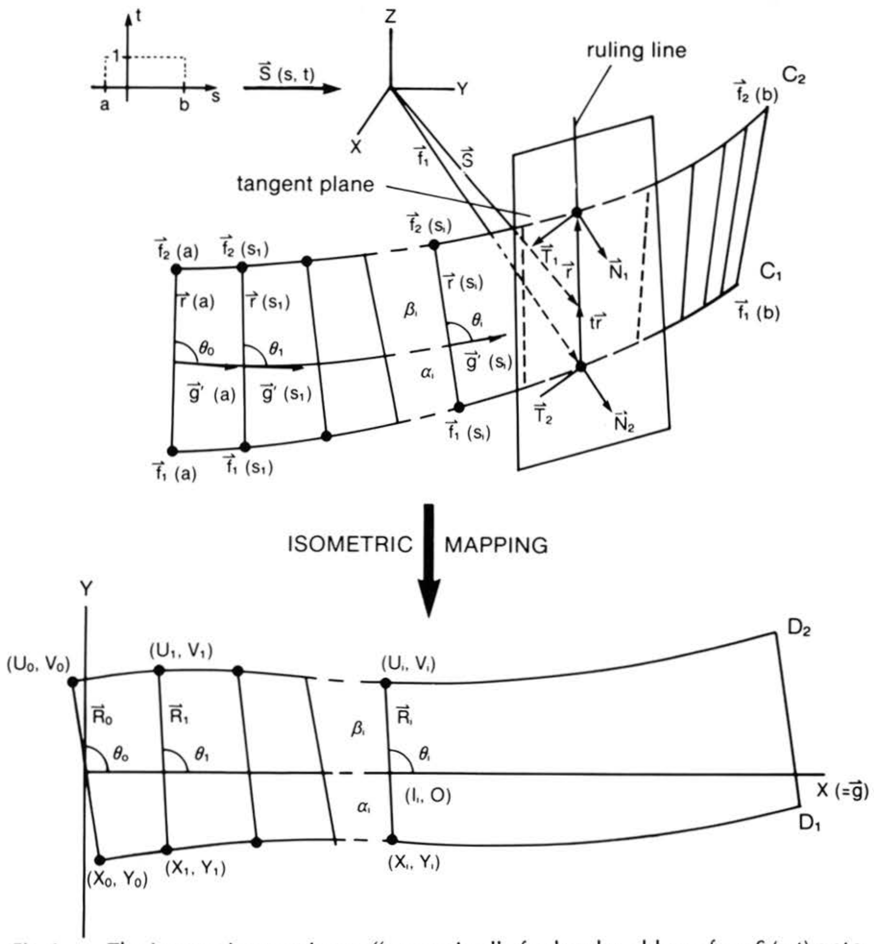

Fig 1 The isometric mapping or "expansion" of a developable surface S (s, t) onto the x, y plane. In the figure vectors are denoted by a superscript arrow rather than bold-face as in the text.

Definitions and Results

Let C^k [a, b] $\equiv$ {x(s): x is k times continuously differentiable for all s in [a, b]}. Curves in $\mathbf{R}^3$ will be denoted by

$$C: f = x(s)i + y(s)j + z(s)k, s \varepsilon [a, b],$$

with components x(s), y(s), and z(s), and Euclidean norm $\|f\| = (x^2(s) + y^2(s) + z^2(s))^{1\ 2}$. In Fig 1 vector functions are denoted by f rather than by **f** as in the text. In what follows, it is assumed that the developable surface **S** is given by ([4])

$$\mathbf{S}(s, t) = \mathbf{f}_1(s) + tr(s), \dot{\mathbf{f}}_1(s) \times \mathbf{r}(s) \neq 0, s \varepsilon [a, b], t \varepsilon [0, 1], \quad (3)$$

where

(i) $\mathbf{f}_1(s) = x_1 i + y_1 j + z_1 k$, with x, y, z ε C^2 [a, b],

(ii) $r(s) = r_1(s)i + r_2(s)j + r_3(s)k$, with r_i ε C^2 [a, b], i = 1, 2, 3,

(iii) each point of **S** corresponds to only one ordered pair (s, t).

Thus **S** is bounded by the curves C_1: $\mathbf{f}_1(s)$ and C_2: $\mathbf{f}_2(s) = \mathbf{f}_1(s) + \mathbf{r}(s)$ and the ruling lines **r**(a) and **r**(b). In the actual application of the mapping procedure to hull plate expansion, C_1 and C_2 represent the chine curves defining the developable surfaces as in Fig 2.

Remark

Each ruling line **r**(s) in **S** can be computed as $\mathbf{r}(s) = \mathbf{f}_2(s^*) - \mathbf{f}^1(s)$ for fixed s ε [a, b] where s* ε [a, b] is the solution of the fixed point equation (Clements 1984)

$$G\ (s^*) = \det(\dot{\mathbf{f}}_1(s), [\mathbf{f}_2(s^*) - \mathbf{f}_1(s)], \dot{\mathbf{f}}_2(s^*)) = 0, \quad (4)$$

Equation (4) is used because of the result that a ruled surface given by (3) is developable if and only if det $(\dot{\mathbf{f}}_1(s), \mathbf{r}(s), \dot{\mathbf{r}}(s)) = 0$ (Kreyszig 1959). It is assumed here that this calculation has already been completed and that **r**(s) is known for every fixed s in [a, b].

The geodesic curvature κ_g of a curve C: **g**(s) in a surface **S** is given by

$$\kappa_g(s_0) = \det\ (\dot{\mathbf{g}}, \ddot{\mathbf{g}}, n) = -\ddot{\mathbf{g}} \cdot (\dot{\mathbf{g}} \times \mathbf{n}),$$

where **n** is the unit normal to the surface **S** at $\mathbf{g}(s_0)$. We also note that **g**(s) on **S** is a geodesic if and only if $\kappa_g = 0$ and is either a straight line or its principal normal coincides with the surface normal at the point s. Since a geodesic **g**(s) in **S** joining any two points of **S** not on the same generator can be represented in the form

$$\mathbf{g}(s) = \mathbf{f}_1\ (s) + t^*\ (s)\ \mathbf{r}(s) \quad (5)$$

for some twice continuously differentiable function (Clements & Leon 1984) **g**(s) and $\dot{\mathbf{g}}(s)$ can be obtained by solving (1) for t*(s) and $\dot{t}^*$ (s), or equivalently by solving the first order system

$$\dot{u}_1(s) = u_2(s), \qquad a \leq s \leq b,$$

$$\dot{u}_2(s) = F(u_1(s), u_2(s)), \qquad a \leq s \leq b, \quad (6)$$

$$= - \frac{\ddot{\mathbf{f}} \cdot (\dot{\mathbf{g}} \times \mathbf{n}) + u_1 \ddot{\mathbf{r}} \cdot (\dot{\mathbf{g}} \times \mathbf{n}) + 2u_2 \dot{\mathbf{r}}(\dot{\mathbf{g}} \times \mathbf{n})}{\mathbf{r} \cdot (\mathbf{g} \times \mathbf{n})},$$

for $u_1(s) = t^*(s)$ and $u_2(s) = \dot{t}^*(s)$ at each s in [a, b] where $\mathbf{n}(s) = \mathbf{r}(s) \times \mathbf{T}_1(s)$ is a normal to the developable surface **S** at $\mathbf{g}(s)$ and $\mathbf{T}_1(s) = \dot{\mathbf{f}}_1(s)/\|\dot{\mathbf{f}}_1\|$. Calculation of the Lipschitz constant for F in (6) shows that the system is solvable by standard numerical methods (Dahlquist and Bjorck 1969) provided $\mathbf{r} \cdot (\dot{\mathbf{g}} \times \mathbf{n})$ stays bounded away from zero. The numerical solution of equation (6) requires some starting values at s = a and involves the computation of $\mathbf{f}_1, \dot{\mathbf{f}}_1, \ddot{\mathbf{f}}_1, \mathbf{r}, \dot{\mathbf{r}}$, and $\ddot{\mathbf{r}}$. For simplicity the starting values are taken to be

$$u_1(a) = .5, \quad u_2(a) = 0 \quad (7)$$

or the midpoint of $\mathbf{r}(a)$ as indicated in Fig 1.

Once (1), (2) have been solved, the image of each $\mathbf{r}(s_i)$, i = 0, ..., *N*, is determined by computing as in Fig 1.

$$\alpha_i = t^* (s_i) \|\mathbf{r}(s_i)\|,$$

$$\beta_i = (1 - t^* (s_i)) \|\mathbf{g}(s_i) - \mathbf{f}_2(s_i)\|. \quad (8)$$

$$\theta_i = \cos^{-1} \left(\frac{\dot{\mathbf{g}}(s_i) \cdot \mathbf{r}(s_i)}{\|\dot{\mathbf{g}}(s_i)\| \cdot \|\mathbf{r}(s_i)\|} \right) \text{for } i = 0, ..., N,$$

Then

$$(X_i, Y_i) = (J_i - \alpha_i \cos \theta_i, - \alpha_i \sin \theta_i),$$

$$(U_i, V_i) = (J_i + \beta_i \cos \theta_i, \beta_i \sin \theta_i), \quad (9)$$

$$J_i = \sum_{j=1}^{i} l_j,$$

$$\mathbf{R}_i = (U_i - X_i, V_i - Y_i)$$

and the surface **S** bounded by $\mathbf{r}(s_0)$ and $\mathbf{r}(s_N)$ has been mapped isometrically onto that portion of the plane bounded by $\mathbf{R}_0$, R_N and (X_i, Y_i), (U_i, V_i), i = 0, ..., N.

The calculations θ_i, $\cos \theta_i$ and $\sin \theta_i$ in (8) and (9) are only required at each partition point and can be computed using double precision arithmetic. Consequently, it is assumed that roundoff errors in individual arithmetic operations and evaluations of the elementary special functions are negligible when compared to the errors propagated due the use of inexact function values in the solution of (2). Consequently, it is assumed that the errors in evaluating θ_i, $\cos \theta_i$ and $\sin \theta_i$ do not contribute significantly to the global error in the mapping procedure.

It is assumed that the geodesic $\mathbf{g}(s)$ stays within the boundaries of the developable surface, and that $r \cdot (\dot{\mathbf{g}} \times \mathbf{n})$ stays bounded away from zero. Hence, the differential equations given by (6) are solvable everywhere on [a, b]. One method to ensure this is to restart the procedure with new initial values at that point on **g** where it crosses the boundary curves C_1 or C_2, or where $\mathbf{r} \cdot (\dot{\mathbf{g}} \times \mathbf{n})$ becomes sufficiently small. However, to avoid unnecessary complexity, these details are not included in the statement of the mapping algorithm given here.

The Mapping Algorithm:

1) Choose P_N and ε_T.

2) Choose an adaptive quadrature routine for evaluating I_k in (9) accurate to a specified ε_{Qk} for each $k = 1, ..., N$.

3) Choose an accurate variable step differential equation solving routine where each evaluation of t^* and $\dot{t}^*$ required in the integration routine is computed accurate to a specified ε_{Dk}, for each $k = 1, ..., N$.

4) Compute α_0, β_0 and θ_0 as in (8) to obtain

$$(X_0, Y_0) = (-\alpha_0 \cos\theta_0, -\alpha_0 \sin\theta_0),$$

$$(U_0, V_0) = (\beta_0 \cos\theta_0, \beta_0 \sin\theta_0),$$

and

5) For each $i = 1, ..., N$ compute I_i, α_i, β_i, and θ_i as in (8) and (9) using the differential equation solver to evaluate $\mathbf{g}$ and $\dot{\mathbf{g}}$ at the points required by the quadrature routine, to obtain the coordinates of the ruling line end points of **S** mapped isometrically

$$(X_i, Y_i) = (J_i - \alpha_i \cos\theta_i, -\alpha_i \sin\theta_i),$$

$$(U_i, V_i) = (J_i + \beta_i \cos\theta_i, \beta_i \sin\theta_i),$$

onto the plane.

Choosing the error tolerances

Assume a tolerance ε_T, is given, which is to bound the total error in the numerical process. On the assumption that the calculations are equally difficult in each subinterval $[s_{k-1}, s_k]$ we bound the total error committed in evaluating I_k by

$$\varepsilon_T \frac{(s_k - s_{k-1})}{(b - a)} = \varepsilon_k \tag{10}$$

for each k. It is also assumed that an Adaptive Quadrature Routine (AQR), based on a quadrature formula with degree of precision d, will be used to evaluate (2), and that an appropriate Ordinary Differential Equation Solver (ODES), based on a method of order r, will be used to solve (1) at the points required by the AQR. Given ε_k the problem is to determine the tolerances, ε_{Qk} and ε_{Dk}, to be given to the AQR and the ODES respectively. As in Fritsch, Kahaner and Lyness (1981) it is assumed throughout that both the AQR and the ODES are completely reliable, that is, they will always return results which are accurate to within the tolerances specified, that the AQR will perform its computations in the same order with polluted data as with exact data, and that all calls to the ODES are of the same level of difficulty.

The problem is one of interfacing two software packages in an efficient and reliable manner. This problem has been considered recently in Fritsch, Kahaner and Lyness (1981) (under the same assumptions as above), where the scenario of an AQR using results obtained by a second AQR was considered. The analysis given here is motivated by this work.

Denote by $\phi(f)$ the AQR applied to a function f, by $I(f)$ the correct integral of f, and by f_c the numerical approximation to the function f which is actually computed by the ODES, here denoted by ψ. Given ε_k, it is then required that

$$|I(f) - \phi(f) + \phi(f) - \phi(f_c)| \leq |I(f) - \phi(f)| + |\phi(f) - \phi(f_c)| \leq \varepsilon_k.$$

The error can now be seen to be made up of two distinct components: $|I(f) - \phi(f)|$, due to approximating I by ϕ (assuming that correct function evaluations are available to ϕ), and $|\phi(f) - \phi(f_c)|$, due to incorrect function evaluations being made available to ϕ. Thus divide ε_k, the total error tolerance for the entire numerical procedure of evaluating I_k, into two components, ε_{Qk} and ε_{ODEk} with

$$\varepsilon_k = \varepsilon_{Qk} + \varepsilon_{ODEk},$$

$\varepsilon_{Qk} \geq |I(f) - \phi(f)|$ being the tolerance given to the AQR, and ε_{ODEk} bounding the errors due to obtaining inaccurate values of f. Thus the tolerances given in each call to ψ, namely ε_{Dk}, depend on ε_{ODEk}, although they will not normally be equal to ε_{ODEk}. The interpretation of ε_{ODEk} is that it reflects the total error accummulated due to the calls to ψ made by ϕ.

Assume that the AQR uses a basic Quadrature Formula $Q_B(f) = \Sigma w_j f(x_j)$ and has some strategy for applying Q_B to some sequence of subintervals in the interval of integration. Q_B, when applied over an interval of unit length will have a condition number $C_B = \Sigma|w_j|$. It follows by linearity of the quadrature process that if Q_B is applied over an interval of length H, the condition number of the integration will be HC_B. Now define C_Q to be the condition number of ϕ applied over the interval (s_{k-1}, s_k). By linearity of the quadrature process, we have $C_Q \leq (s_k - s_{k-1})C_B$. In fact C_Q will equal C_BH unless Q_B uses both endpoints of the interval of the integration and the corresponding coefficients are of opposite sign. Note that the method by which ϕ determines the order in which Q_B will be applied to the subintervals does not affect the previous comments.

Now assume that the maximum error in any call to ψ is ε_{Dk}. It then follows that the maximum error in the numerical process due to ϕ using inexact data from ψ is bounded by $C_Q\,\varepsilon_{Dk}$. Thus, from

$$\varepsilon_k = \varepsilon_{Qk} + \varepsilon_{ODEk} = \varepsilon_{Qk} + (s_k - s_{k-1})\,C_B\,\varepsilon_{Dk} \tag{11}$$

it follows that, once ε_{ODEk} is specified, ε_{Dk} should be chosen as

$$\varepsilon_{Dk} = \frac{\varepsilon_{ODEk}}{(S_k - s_{k-1})\,C_B} \tag{12}$$

If the condition number of ϕ is specified in more detail (see, for example, Stoer and Bulirsch 1980) then a similar analysis may be carried out. However, since this leads to different tolerances being specified for each call to ψ, (that is, ψ would have to be solved from the initial point a on every call) and also leads to some restrictions on the type of AQR which may safely be used, we shall not pursue this possibility further here.

There are two generic ways in which an AQR can be constructed (Fritsch, Kahaner and Lyness, 1981), it can either be based on a local error control strategy (LG) or it can be based upon some form of global error control strategy (GE). The essence of a LE method is that the procedure moves through the interval of interest, say from left to right, and locally acts upon a subinterval. It does not progress to the next subinterval until the result obtained for the subinterval under consideration is considered to be sufficiently (locally) accurate. Consequently, when the procedure is complete, since all the local errors are believed to be sufficiently small, the total error of the procedure is believed to be within the specified tolerance. In contrast, a GE method is always able to consider one of many subintervals, and always chooses to work on that subinterval which has the worst error estimate. Thus a GE method examines the subintervals in an order very difficult to predict before the computation is begun. A

GE method exists when the sum of all the local errors is believed to be less than the prescribed tolerance. It is recommended (Fritsch, Kahaner and Lyness 1981) that a global error control strategy AQR be used if it is felt that the AQR may encounter some difficulty since, in contrast to a local error control strategy AQR, a global error AQR may have the opportunity to recover from dealing with a subinterval with a large error estimate by considering other subintervals.

Now consider the question of choosing ε_{ODEk} (and thus ε_{Dk}) and ε_{Qk} to minimize the computational cost, given the requirement $\varepsilon_k = \varepsilon_{Qk} + \varepsilon_{ODEk}$. The error when Q_B is applied over an interval of length H depends upon H^{d+2}. Thus, if (a, b) is partitioned into subintervals of length H, the overall error when using Q_B in a compounded manner will be $O(H^{d+1})$. Measuring work in terms of the number of subintervals (which is proportional to the number of integrand evaluations required) of equal length to which Q_B should be applied to achieve a given tolerance, it follows that the cost of using ϕ with tolerance ε_{Qk} is proportional to $\varepsilon_{Qk}^{-1/(d+1)}$. Now consider the amount of work involved in a call to ψ. Since ψ is based on a method, either multistep or Runge-Kutta, which has local order r, and thus local error $O(H^{r+1})$ when applied to a subinterval of length H, it follows that, under appropriate conditions (see, for example Stoer and Bulirsch 1980), the global error involved in using the basic method of ψ when the interval of integration is partitioned into subintervals of length H is $O(H^r)$. Thus the cost of using ψ with error tolerance ε_{Dk} is proportional to $\varepsilon_{Dk}^{-1/r}$.

To follow the approach in Fritsch, Kahaner and Lyness (1981), it is assumed that each call to ψ is independent of every other one. In the simplest implementation, this would mean that ψ would always begin from the point a when obtaing a value of $\mathbf{g}(s)$ at an integration point. Since we have assumed that the use of inexact data by ϕ does not change the sequence of computational steps in ϕ, we shall also assume that the number of function evaluations required by ϕ is fixed at, say, N. If the same tolerance ε_{Dk} is specified on each call to ψ, then the total work done will be proportional to $N\,\varepsilon_{Dk}^{1-/r}$ that is, proportional to $\varepsilon_{Qk}^{-1/(d+1)}\,\varepsilon_{ODEk}^{-1/r}$ Using (12) and the constraint $\varepsilon_k = \varepsilon_{Qk} + \varepsilon_{ODEk}$, we minimize this expression obtaining

$$\varepsilon_{Qk} = \frac{r}{d+r+1}\varepsilon_k, \qquad \varepsilon_{ODEk} = \frac{d+1}{d+r+1}\varepsilon_k, \tag{13}$$

as the appropriate choices to minimize the total amount of work required for the computations done in the interval (s_{k-1}, s_k). Note that the specific values of the constants of proportionality will not affect ε_{Qk} and ε_{ODEk} as given in (13).

The choice of ε_{Qk} and ε_{Dk} must also be such that the computation is stable. In particular, ε_{Dk} must not be so large as to preclude the possibility of ϕ being able to converge given the tolerance ε_{Qk}. Since ϕ is a linear process, assume it has a linear method for estimating the error of the computation in a given subinterval, let this process have condition number C_E when applied over a subinterval of unit length (see the discussion of C_B above). Thus, for a subinterval of length $s_k - s_{k-1}$, the bound on the error in the error estimation process of ϕ due to using inexact data from ψ is $C_E\,(s_k - s_{k-1})\,\varepsilon_{Dk}$. Thus it follows from (12) that the inequality

$$\varepsilon_{Qk} > C_E\,(s_k - s_{k-1})\,\varepsilon_{Dk} = C_E\,(s_k - s_{k-1})\,\frac{\varepsilon_{ODEk}}{(s_k - s_{k-1})\,C_B} = \frac{C_E}{C_B}\varepsilon_{ODEk} \tag{14}$$

should be satisfied for all k to ensure stability of the AQR in all subintervals (s_{k-1}, s_k).

From (10), (12) and (13) it follows that

$$\varepsilon_{Dk} = \frac{(d+1)}{(r+d+1)}\,\frac{1}{C_B\,(b-a)}\,\varepsilon_T; \qquad \varepsilon_{Qk} = \frac{r}{(r+d+1)}\,\frac{(s_k - s_{k-1})}{(b-a)}\,\varepsilon_T. \tag{15}$$

Note that ε_{Dk} does not depend upon k. Combining (14) with (15) leads to the global stability requirement

$$r\, C_B > (d + 1)\, C_E. \tag{16}$$

Implementation

It was assumed in the previous section that each call to the ODES ψ was independent of all other calls to ψ. In the simplest implementation this would lead us to call ψ always taking a as the left endpoint of the interval of integration. Since we cannot tell in advance at which points the AQR will need a function value we cannot simply make one pass with ψ and obtain all the function values needed.

This now suggests the following more efficient two stage strategy. First, call ψ with a tolerance $\varepsilon_{Dk}/2$, and obtain the function values at the points s_k of the partition P_N. Second, in solving (2) in each subinterval (s_{k-1}, s_k), $k = 1, 2, ..., N$ when the AQR ϕ requires a function evaluation, ψ is called with left starting point s_{k-1}, the previously computed function value at s_{k-1} given, and a tolerance of $\varepsilon_{Dk}/2$. Since constants do not affect the analysis leading up to (13), ε_{Qk} and ε_{Dk} are still chosen according to (15), subject to (16) being satisfied. Assuming that the error propagated by ψ due to using an inexact starting value at s_{k-1} is damped by ψ (as is typically assumed when error control strategies of ODES depend upon local error estimation), we expect to obtain function values accurate to the required tolerance of ε_{Dk}.

A more efficient strategy might be to replace the second stage above with the use of an interpolating function of some sort (for example, cubic splines). However, this approach has two difficulties. It leads to a three tiered system for solving (2) (that is, the use of an ODES is followed by the use of an interpolation routine, which in turn is followed by the use of an AQR) in contrast to the two tiered method. Secondly, it appears that when one assigns appropriate tolerances for each of the three routines to be used, an attempt to choose them optimally to reduce computational costs is highly dependent on the particular problem being solved. Thus, while it may well be possible to tune such a styem to be efficient for particular problems, it seems that it is difficult to derive appropriate strategies for the choice of error tolerances in a more general setting. Consequently, we have not used this three tiered approach here.

Using hull curve offset data for a 38 foot developable hull sloop (Fig 2), the mapping algorithm together with criteria (13), (14) and (16) was successfully employed in the complete generation of each hull plate (Fig 3) with only very short CPU times being required on a CYBER 170-730 computer. The AQR used was QUANC8 ($d = 8$, Forsythe, Malcolm and Moler 1977) and the ODES used was RKF45 ($r = 4$, Forsythe, Malcolm and Moler 1977).

Precise results regarding the efficiency of the algorithm and the error bound strategies outlined in this work are currently under investigation using surface patches of regular circular cones for which simple analytic mapping formulas exist.

References

Clements, J.C. 1981. A computer system to derive developable hull surfaces and tables of offsets, Marine Technology, 18: 227-233.

Clements, J.C. and **Leon, L.J.** 1986. A fast, accurate algorithm for the isometric mapping of a developable surface, in press.

Clements, J.C. 1984. Developed plate expansion using geodesics, *Marine technology,* 21: 384-388.

Dahlquist, G. and **Bjorck, A.** 1969. *Numerical Methods,* Prentice-Hall.

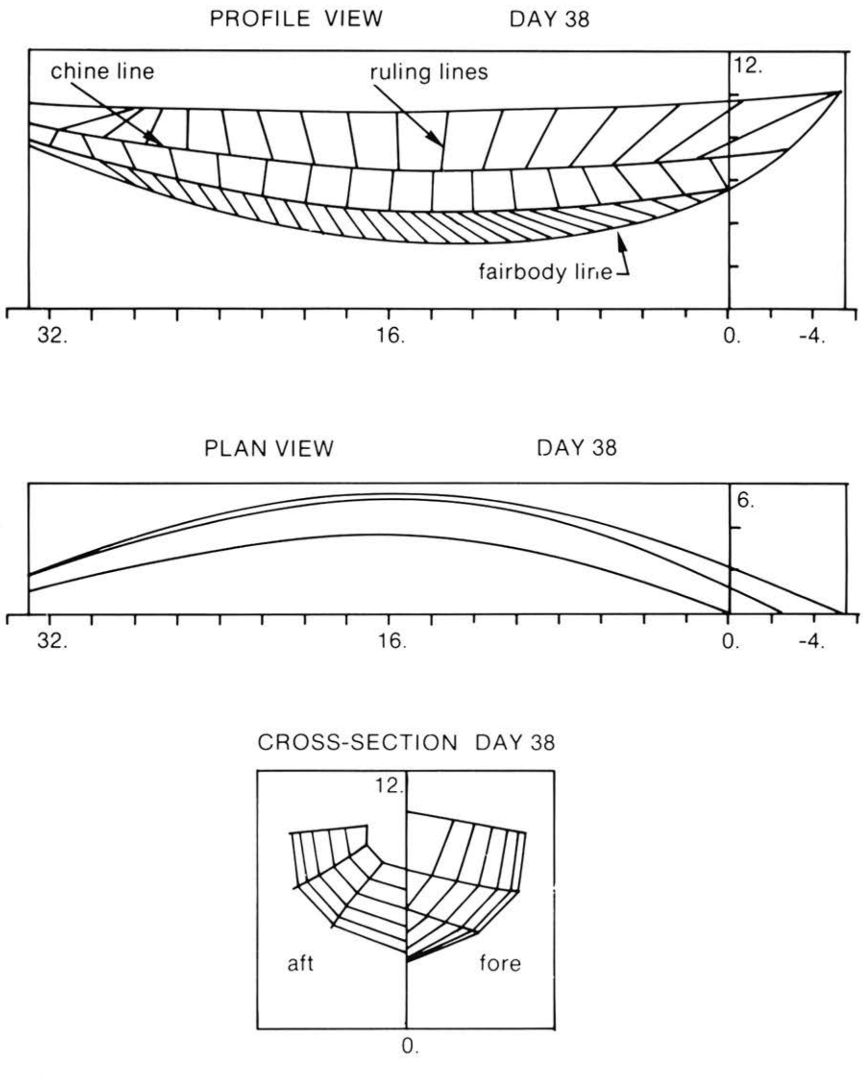

Fig 2 The profile, plan and cross-section views of a 11.583 m developable hull sloop DAY38. The profile view shows the ruling lines computed for each of the three developable surfaces which comprise one side of the hull form. All units are in 0.3048 m.

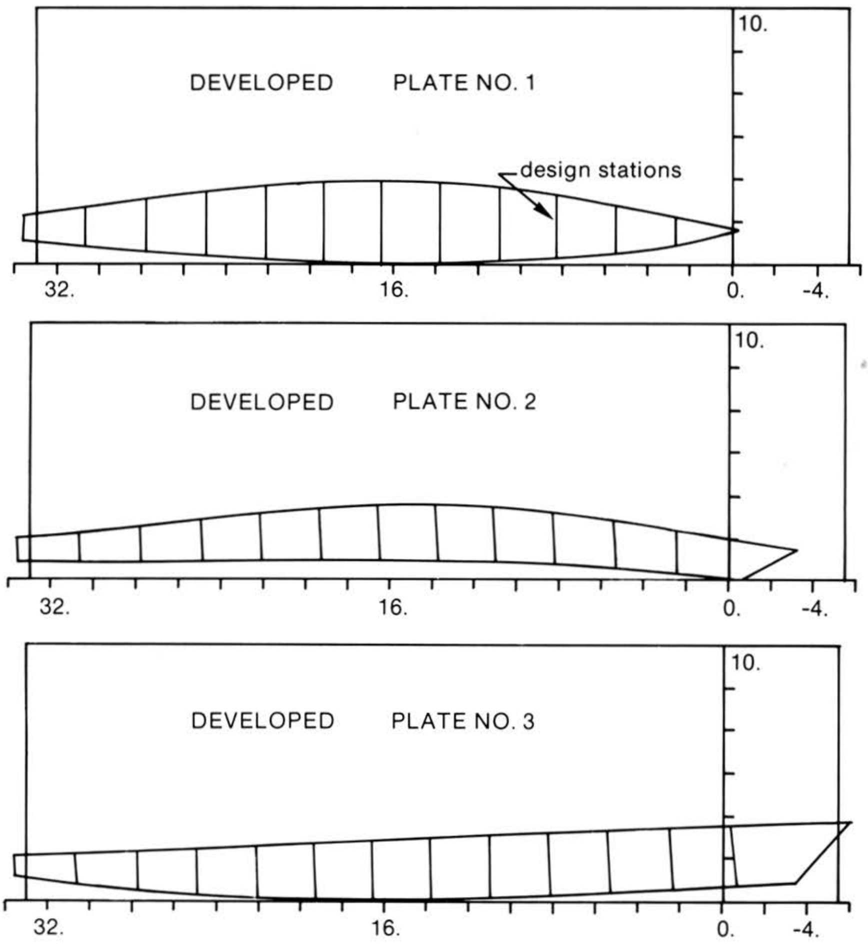

Fig 3 The results obtained for the mapping or "plate expansion" of each of the 3 developable hull surfaces of DAY38. Plate No. 1 corresponds to the surface adjacent to the keel; plate No. 2 to the next surface up and plate No. 3 to the uppermost surface.

Fritsch, F.N., Kahaner, D.K. and **Lyness, J.N.** 1981. Double integration using one-dimensional adaptive quadrature routines: A software interface problem, *ACM T.O.M.S.*, *7*: 46-75.

Forsythe, G.E., Malcolm, M.A. and **Moler, C.B.** 1977. *Computer Methods for Mathematical Computations*, Prentice-Hall.

Kreyszig, E. 1959. *Differential Geometry*, University of Toronto Press.

Stoer, J. and **Bulirsch, R.** 1980. *Introduction to Numerical Analysis*, trans. R. Bartels, W. Gautschi, & C. Witzgall, Springer-Verlag, New York.