

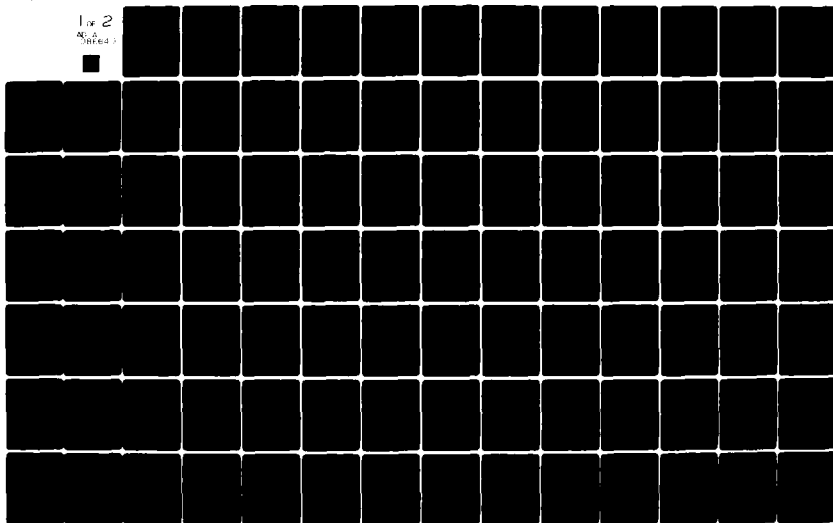
AD-A086 640

NAVAL POSTGRADUATE SCHOOL MONTEREY CA F/G 13/10
COMPUTER AIDED GEOMETRICAL VARIATION AND FAIRING OF SHIP HULL F--ETC(U)
MAY 78 F R HABERLANDT

UNCLASSIFIED

NL

1 of 2
AD-A086 640



ADA 086640

DDC FILE COPY

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

2 p. 5.

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A086640	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) COMPUTER AIDED GEOMETRICAL VARIATION AND FAIRING OF SHIP HULL FORMS		5. TYPE OF REPORT & PERIOD COVERED THESIS
7. AUTHOR(s) HABERLANDT, FREDERICK R		6. PERFORMING ORG. REPORT NUMBER
9. PERFORMING ORGANIZATION NAME AND ADDRESS MASS. INST. OF TECHNOLOGY		8. CONTRACT OR GRANT NUMBER(s)
11. CONTROLLING OFFICE NAME AND ADDRESS CODE 031 NAVAL POSTGRADUATE SCHOOL MONTEREY, CALIFORNIA, 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
13. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) LEVEL		12. REPORT DATE MAY 78
		14. NUMBER OF PAGES 159
		15. SECURITY CLASS. (of this report) UNCLASS
16. DISTRIBUTION STATEMENT (of this Report) APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report) DTIC SELECTE JUL 15 1980		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) GEOMETRICAL VARIATION; FAIRING OF SHIP HULL FORMS; HULL FORMS		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) SEE REVERSE.		

COMPUTER AIDED GEOMETRICAL VARIATION
AND FAIRING OF SHIP HULL FORMS

by

FREDERICK ROBERTSON HABERLANDT

Submitted to the Department of Ocean Engineering in May 1978,
in partial fulfillment of the requirements for the Degree of
Ocean Engineer and the Degree of Master of Science in Naval
Architecture and Marine Engineering.

ABSTRACT

Two distinct aspects of computer aided ship design are addressed in this thesis. First, a geometrical hull form modification technique employing the longitudinal repositioning of sections is developed. The second aspect deals with the mathematical representation of lines and line fairing. Before this is done however, justification is presented for utilizing third degree polynomials as an approximation to the spline curves of the naval architect. The results obtained indicate that a fairing procedure based on a least squares curve fitting criteria and a lines representation procedure based on parametric cubic equations could be adapted to generate faired hull forms from the roughest preliminary hull design. Additionally, the hull form modification technique could be programmed so as to produce designs with desired values of C_p , LCB, C_w and LCF from a basis design of similar type.

Thesis Supervisor: Professor Chrysostomos Chrysostomidis
Title: Associate Professor of Naval Architecture

Approved for public release
distribution unlimited.

COMPUTER AIDED GEOMETRICAL VARIATION
AND FAIRING OF SHIP HULL FORMS

by

FREDERICK ROBERTSON HABERLANDT

B.S., Mechanical Engineering, University of Florida
(1971)

Submitted in Partial Fulfillment
of the Requirements for the
Degree of

OCEAN ENGINEER

and the Degree of

MASTER OF SCIENCE IN NAVAL ARCHITECTURE
AND MARINE ENGINEERING

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May, 1978

© Frederick Robertson Haberlandt 1978

Signature of Author.....*F.R. Haberlandt*.....
Department of Ocean Engineering
May 12, 1978

Certified by.....*Chrysostomos Chrysostomides*.....
Thesis Supervisor

Accepted by.....*A. Douglas Carmichael*.....
Chairman, Department Committee

COMPUTER AIDED GEOMETRICAL VARIATION
AND FAIRING OF SHIP HULL FORMS

by

FREDERICK ROBERTSON HABERLANDT

Submitted to the Department of Ocean Engineering in May 1978,
in partial fulfillment of the requirements for the Degree of
Ocean Engineer and the Degree of Master of Science in Naval
Architecture and Marine Engineering.

ABSTRACT

Two distinct aspects of computer aided ship design are addressed in this thesis. First, a geometrical hull form modification technique employing the longitudinal repositioning of sections is developed. The second aspect deals with the mathematical representation of lines and line fairing. Before this is done however, justification is presented for utilizing third degree polynomials as an approximation to the spline curves of the naval architect. The results obtained indicate that a fairing procedure based on a least squares curve fitting criteria and a lines representation procedure based on parametric cubic equations could be adapted to generate faired hull forms from the roughest preliminary hull design. Additionally, the hull form modification technique could be programmed so as to produce designs with desired values of C_p , LCB, C_w and LCF from a basis design of similar type.

Thesis Supervisor: Professor Chryssostomos Chryssostomidis
Title: Associate Professor of Naval Architecture

ACKNOWLEDGEMENTS

The author is deeply indebted to his advisor, Professor Chrysostomidis for his tireless efforts during the course of this work. A deep debt of gratitude is also owed Professor Heinrich Söding from the University of Hanover, Hanover, West Germany. His contribution of the parametric rotating spline technique added greatly to the capability of the final program.

The author would also like to thank Mrs. Sandy Margeson for her masterful typing of an, all too often, illegible manuscript.

Finally the author wishes to thank his wife and son who, uncomplainingly, sacrificed many weekends and evenings of his companionship through the course of this work.

Accession For	
NTIS	GRA&I
DDC TAB	
Unannounced	
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or special
A	

TABLE OF CONTENTS

	<u>Page</u>
TITLE PAGE.	1
ABSTRACT.	2
ACKNOWLEDGEMENTS.	3
TABLE OF CONTENTS	4
LIST OF FIGURES	7
1. Introduction.	9
1.1 Background.	9
1.2 Thesis Content.	10
2. Method of Hull Form Modification.	13
2.1 Background.	13
2.2 Development	21
2.2.1 The "One Minus Prismatic" Variation	21
2.2.2 Varying the Fullness of an Entrance or Run not Associated with Parallel Middlebody.	24
2.2.3 Modification of Lackenby's Method to Accommodate Hull Forms with Maximum Sections not at Midships.	31
2.2.4 A Method by Which Constraints may be Placed on the Design Waterline.	33
2.2.5 A Method by which Constraints may be Placed on the Ship's Profile.	38
3. Mathematical Representation of the Lines of a Ship.	40
3.1 Background.	40
3.2 Development	42
3.2.1 Derivation of the Spline Cubic Equation Using a Variational Approach	42
3.2.2 Deflections Due to Bending of a Simple Elastic Beam.	49
3.2.3 Piece-wise Continuous Cubic Polynomial Approximation	54
3.2.4 The Rotating Spline	59

4. Mathematical Fairing of Lines.	.62
4.1 Background	.62
4.2 Development.	.64
4.2.1 The Least-Squares Criteria for Defining the Cubic Curve.	.64
4.2.2 The Moving Strip Method.	.67
4.2.2.1 STRIP1	.68
4.2.2.2 STRIP2	.70
4.2.2.3 STRIP3	.72
4.2.2.4 TRANS1	.75
4.2.3 Fairing of Curves with Infinite Slope.	.76
4.2.3.1 Other Transformations.	
5. Computer Algorithms.	.80
5.1 Overview	.80
5.1.1 Specification of Point Type.	.81
5.1.2 Specification of End Conditions.	.83
5.1.3 Storage of Pertinent Line Data	.84
5.2 Description of Subroutines	.87
5.2.1 Lines Fairing.	.87
5.2.1.1 Subroutine PREFAR.	.87
5.2.1.2 Subroutine FARCRV.	.88
5.2.1.3 Subroutine FARLIN.	.88
5.2.1.4 Subroutine FSTPTS.	.89
5.2.1.5 Subroutine TRANS1.	.89
5.2.1.6 Subroutine STRIP1, 2 or 3.	.89
5.2.2 Lines Representation	.89
5.2.2.1 Subroutine PRESPL.	.90
5.2.2.2 Subroutine SPLINE.	.90
5.2.2.3 Subroutine INTERP.	.91
5.2.2.4 Subroutine CALCY	.91
5.2.2.5 Subroutine CALCT	.92
6. Conclusions and Recommendations.	.93
6.1 Hull Form Modification	.93
6.2 Mathematical Representation of Lines and Fairing.	.96
6.2.1 Lines Representation	.96
6.2.2 Fairing.	.97
6.2.3 Recommendations.	.97
REFERENCES	.103

APPENDIX A.	.106
APPENDIX B.	.108
APPENDIX C.	.112
APPENDIX D.	.116
APPENDIX E.	.120
APPENDIX F.	.123
APPENDIX G.	.156

LIST OF FIGURES

2.1	Ship's Lines.	14
2.2	Sectional Area Curve.	17
2.3	Sectional Area Curve (with parallel middlebody) .	21
2.4	Sectional Area Curve.	26
2.5	Sectional Area Curve.	28
2.6	Sectional Area Curve.	31
2.7	Sectional Area and Waterline Curves	35
2.8	Coefficient Ratio Curve	36
3.1	Strained Beam Element	49
3.2	Discretely Loaded Beam.	52
3.3	Spline Curve Fitting Routine.	54
3.4	Rotating Spline Routine	59
4.1	Least Squares Fit	64
4.2	Pinned End.	70
4.3	Clamped End	72
4.4	Axis Rotation	76
5.1	Point Type Examples	82
6.1	Control Lines for a Typical Bulbous Bow Destroyer	99

B.1	Sectional Area Curve.	108
C.1	Section Modification.	112
D.1	Rotating Spline Routine	116
F.1	Flowchart of Fairing Subroutines.	124
F.2	Flowchart of Splinning and Interpolation Subroutines	125
G.1	Bow Section	157

1. Introduction

1.1 Background

Because of the environment in which they operate, the seakeeping characteristics of a ship are of paramount importance when assessing its overall performance. In the past this aspect of a ship's performance had to be judged by the results of model tests conducted at a point in time well into the preliminary design phase. While these tests provide results of good quality, they were not obtained until the pending design was quite firmly established. In fact, the results obtained by model tests had the characteristic of being just that, results, rather than an important input into the design cycle. The obvious desire then would be to have a tool capable of providing accurate predictions of seakeeping performance based on the data available in the conceptual design phase. These predictions could then be used to influence the selection of hull form coefficients, etc. prior to the time when the hull form is actually being generated.

In 1975 Professors T. Loukakis and C. Chryssostomidis published the "Seakeeping Standard Series for Cruiser-Stern Ships".^[1] This paper corrolates the seakeeping behavior, as predicted by computer model, of the Extended Series 60

hull forms and sets forth a method by which the performance of this type of ship may be predicted based on five parameters: Froude number, F ; ratio of significant wave height to ship length, S ; beam/draft ratio, B/T ; length/beam ratio, L/B ; and block coefficient, CB . With this procedure a designer can predict the relative merits of various candidate designs at a very early stage. This represents a significant capability.

As a result of the work represented in reference [1] there is considerable interest in generating a similar sea-keeping series for contemporary cruiser/destroyer type hull forms. In order to do this in the fashion of reference [1], a representative sample of the ship type must be analyzed by computer model and then the results correlated. It was this need for sample hull forms that provided the motivation for this thesis.

1.2 Thesis Content

There are two aspects of hull form generation addressed in this thesis: first, hull form modification and second, mathematical lines representation and fairing. The technique of hull form modification developed in chapter two is based on the work of H. Lackenby reported in reference [2]. The essence of this method is that the sectional area curve

of an existing ship is redrawn in a systematic fashion to produce a curve with the desired values of prismatic coefficient, C_p , and longitudinal center of buoyancy, LCB. The sections are then shifted longitudinally to produce a modified form with these characteristics. In applying this technique to destroyer type ships there were several anomalies encountered which required that the method of Lackenby be further modified. These modifications, with the pertinent background are contained in chapter two.

The other aspect of lines generation which is addressed in chapters three and four is mathematical lines representation and fairing. Although the fairness of a hull form is not critical to the seakeeping analysis it is an unavoidable subject when considering computer aided ship design. In these chapters the use of parametric cubic splines and least squares curve fitting are addressed. While the parametric splines are shown to provide the capability of representing virtually any type of line, the least squares fairing technique is limited to use with curves representable by single valued functions. The algorithms are, however, capable of fairing lines with infinite slopes at the end points.

It is anticipated that the tools developed in this thesis could be readily fused into a single computer program with the capability of modifying an existing ship form to

obtain a faired design with the desired coefficients of form. When this is developed it will be possible to generate rapidly any number of designs for subsequent performance analysis. The implications of this are discussed in chapter six.

2. Method of Hull Form Modification

2.1 Background

During the design of all but the most trivial engineering systems, it is incumbent upon the engineer that he or she formulate a model of that system. Additionally, the designer must continually refine the model with each successive iterative cycle so that the results are of sufficient detail to be meaningful [3]. One such model used during ship design is a geometric description of the ship's hull form. The most traditional manner of providing this information is by way of the lines drawing.

The ship's lines drawing, more frequently referred to as the ship's lines, is a set of three orthogonal views of the ship's hull depicting the lines of intersection of various planes with the hull form. When viewed in conjunction with one another, they provide the capability to spatially locate any point on the moulded surface of the ship. Figure 2.1, taken from reference [4], is an example of a lines drawing for a "Mariner"-class, steel hull cargo vessel.

While the lines drawing, prepared manually by the naval architect and draftsman, has been the older and more traditional means of depicting the geometric properties of a ship,



Figure 2.1 - Ship's Lines [4]

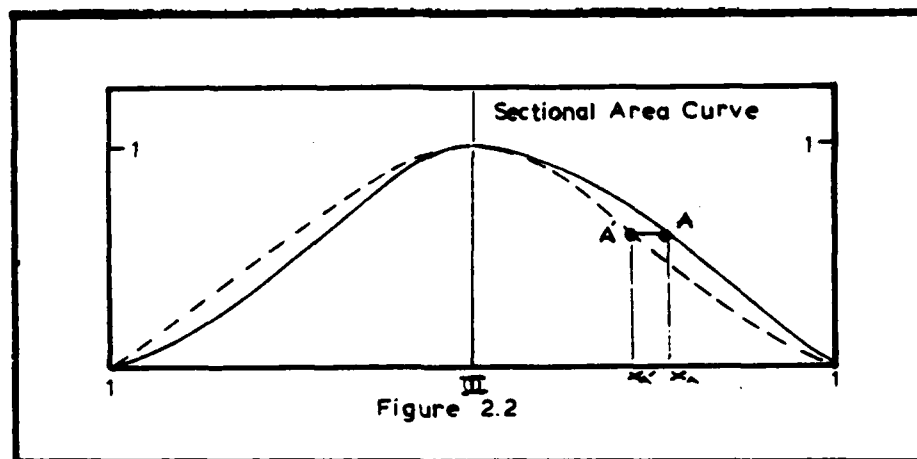
the advent of the high speed computer has provided great impetus for defining the ship's form in a mathematical format [5]. It is interesting however, that a very successful attempt was made at representing ship's lines mathematically by Admiral David Taylor in the early 1900's [6]. This will be expanded upon a bit later.

When first confronted with the job of creating the lines of a new ship, the naval architect seeks a means of quantifying the expected form of the vessel so that he may strive to create an "optimum" design. These optimizing criteria generally take the form of requirements and restrictions placed on the various coefficients of form, i.e., C_p , C_w , LCB, LCF, etc. However, there might also be requirements placed on certain specific regions of the ship. An example of this could be the shape of the midships section for a cargo vessel or the stern configuration dictated by propeller and rudder selections. Nonetheless, when the naval architect completes his candidate design, the important product will be a faired set of ship's lines meeting all the optimizing criteria previously established.

The above procedure is clearly long and involved. For this reason much effort has been expended to develop hull form modification techniques. The objective of these procedures is to utilize an existing, successful hull design, or parent form, as a basis and then to alter this form in a

systematic fashion. This modified form should have the desired characteristics, and, hopefully, require little additional refairing. It is this fairing procedure, described in chapter 4, which requires a large proportion of the designer's effort. The remainder of this chapter addresses the modification techniques themselves.

One of the oldest and most widely used methods of hull form modification is illustrated in Figure 2.2. In essence, the sectional area curve of an existing ship is altered by some arbitrary or systematic method to produce a curve which satisfies some criteria of the designer, usually prismatic or block coefficient and longitudinal center of buoyancy. The offsets for the new design are then obtained by taking the section in the parent whose ordinate in the sectional area curve matches the ordinate of the derived curve. This is represented by the movement of section A at position X_a in the parent to section A' at position X'_a , in the derived hull form. Hence, it is merely a longitudinal repositioning of the existing sections. This method works reasonably well as long as the shifts are of "moderate" amount and the designer is prepared to accept the resulting profile and waterlines without alteration.



The above method lends itself quite well to design without the aid of computers or other automatic computational devices. However, in recent years there has been much work done in the area of hull form modification with the use of high speed digital computers. In virtually all cases where computers are used, an effort is made to represent the ship's contours or surface regions [5] in a mathematical format. It is for this reason that the "Taylor Standard Series" is of interest. It was Admiral David Taylor who, in the early 1900's, generated one of the first successful hull series based on representing the sectional area curve and design waterlines by fifth degree polynomials [6].

Another procedure of hull form modification utilizes specific transformation functions to alter various regions or characteristics of the hull [7]. This form of

modification provides the user with much greater control over the specific ship form than the method of shifting sections longitudinally as previously described. An interesting description of this type of procedure may be found in reference [7].

It is, however, the method of longitudinally shifting sections which was selected for development in this thesis. The reasons for its selection are twofold. First, preliminary work with destroyer-type ships conducted at M.I.T. during the summer of 1977, indicated that the results of the modification were quite realistic and not plagued by gross unfairness. Secondly, the method was tractable and readily adopted to the peculiarities of destroyers. Those peculiarities being principally the fact that this type of ship has no parallel middle body and also that the section of maximum area, in most cases lies at a location other than midship.

The specific method of modification used is that of Lackenby [2] as subsequently modified by Moor [8], and then again by this author. Briefly, the developments presented in reference [2] are highly general, permitting the designer to vary the value of prismatic coefficient, C_p , and the longitudinal center of buoyancy, LCB, of a very wide variety of ships. Included was the capability of altering, or retaining unchanged, the parallel middle bodies of ships so configured. However, one serious drawback was that the method left no

control over the design waterline, and while this line might turn out fair, the longitudinal center of floatation, LCF, merely ended up where it did. It was to this problem that Moor [8] was concerned. By addressing himself to both the sectional area curve and the design waterline in the manner of Lackenby, and then coupling the two procedures, he was able to obtain a derived form having the desired values of C_p , C_w , LCB and LCF.

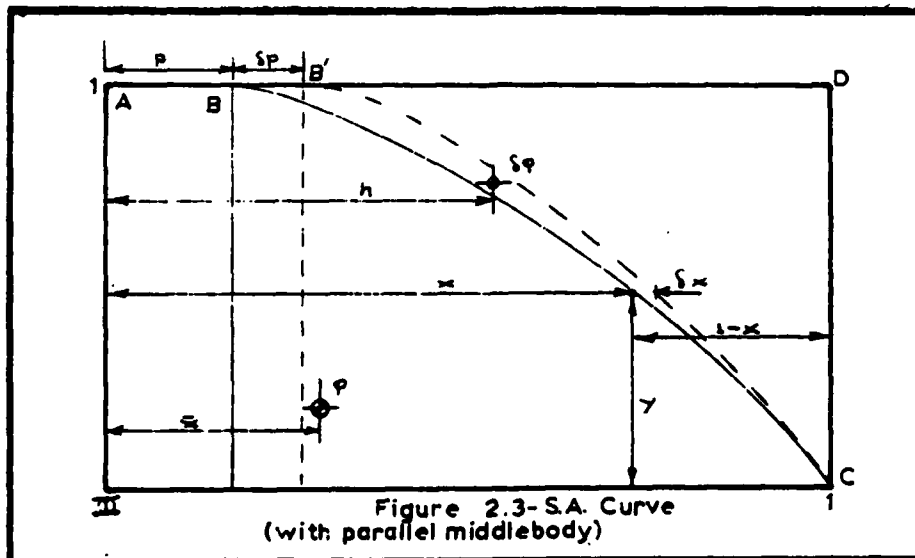
At this point only one minor problem existed with the method as it stood. For ships with keelrise fore or aft it was possible to obtain unwanted oscillations in the ship's centerline profile. To eliminate these oscillations, this author extended the logic of Moor to include the ship's profile. In so doing, the designer may be assured of a derived form having not only the four desired characteristics and coefficients previously mentioned, but also the desired profile. The only restricting requirement, other than the fact that the changes in C_p and C_w be "moderate", is that for the method to be mathematically rigorous the section of maximum beam, sectional area and local draft must be coincident. If this isn't the case a small ($\approx 1\%$) unpredictable error, based on the parent hull design and the desired changes is introduced.

All of these relationships are developed in full in the following section. The only other alteration to the methods of Lackenby and Moor was that the procedure had to be capable of accommodating destroyer-type ships whose maximum sections fill other than at midship. This change is also included in the derivations that follow.

2.2 Development

2.2.1 The "One Minus Prismatic" Variation

As a means of introducing the method of longitudinal repositioning of sections, the traditional "one minus prismatic" is first developed. This procedure enables the designer to modify the fineness of a parent form by expanding (creating in ships without), or reducing the region of parallel middle body. It is convenient for this, and the following derivations to refer to Figure 2.3 and the following definitions. It should also be noted that the sectional area curve is normalized with respect to both the value of maximum area and length of the half body.



For the parent design:

ϕ = the prismatic coefficient of the half body.

$\bar{x}$ = the fractional distance from midships of
the centroids of the half body.

p = the fractional parallel middle of the half body.

x = the fractional distance of any transverse
section from midships.

y = normalized area of any transverse section at
longitudinal position x .

For the derived form:

$\delta\phi$ = the required change in prismatic coefficient
of the half body.

δp = the resulting change in parallel middle body.

δx = the necessary longitudinal shift of the
section at x required to generate the required
change in prismatic coefficient.

h = the fractional distance from midships of the
centroid of the added "sliver" of area
represented by $\delta\phi$.

In Figure 2.3 it should be recognized that AB'C is the
curve of the derived form and curve ABC is that of the
parent. In accordance with the method of the "one minus
prismatic" the new location of the transverse sections is
defined by the following equations.

$$\frac{1-(x+\delta x)}{1-x} = \frac{1-(\phi+\delta\phi)}{1-\phi}$$

$$\frac{\delta x}{1-x} = \frac{\delta\phi}{1-\phi}$$

$$\delta x = \frac{\delta\phi}{1-\phi} (1-x) \quad (2.1)$$

The area BCD is seen to be $1 - \phi$ and the above modification simply reduces it by the factor $\frac{1 - (\phi+\delta\phi)}{1-\phi}$.

The new area B'CD is therefore $1 - (\phi+\delta\phi)$, demonstrating that the method generates the desired prismatic coefficient of $\phi + \delta\phi$.

There is however a concomitant change in a parallel middle body found by solving for x at $x = p$, i.e.,

$$\delta p = \frac{\delta\phi}{1-\phi} (1-p)$$

$$\frac{\delta p}{1-p} = \frac{\delta\phi}{1-\phi} \quad (2.2)$$

Therefore the resulting change in prismatic coefficient is obtained by altering the length of parallel middle body and then proportionally expanding or contracting the entrance and run. Because of this procedure the method has the following disadvantages:

1. There is no control over the length of the parallel middle body, i.e., ϕ and p , cannot be varied independently.
2. The procedure cannot be applied to reduce the fullness of a ship having no parallel middle body.
3. Conversely, a ship cannot be increased in fullness without introducing parallel middle body.
4. The prismatic coefficient of the entrance or run cannot be altered.
5. The region where fullness is added cannot be controlled. That is, the maximum changes in fullness take place at the shoulders of the curve, i.e., point B.

It is because of these numerous, severe restrictions that Lackenby sought to develop a more general technique of modification.

2.2.2 Varying the Fullness of an Entrance or Run not Associated with Parallel Middle Body.

In reference [2], Lackenby concerned himself with providing a means by which to modify both C_p , LCB and the length of parallel middle body in a controllable manner.

While many of these relationships are of importance, only those which apply more specifically to destroyer-type hull forms will be pursued in any detail. However, for the readers' convenience, the most generalized case of Lackenby's formulas is included as equations (2.10) through (2.13) in the last part of this section. For the in depth derivations the reader is referred to the original paper. Nevertheless, the following derivation is the foundation upon which all the subsequent relationships are based.

In referring to figure 2.4 the various quantities have a meaning identical to those of the previous section. The only additional term requiring definition is k , defined mathematically as follows:

$$k^2 = \frac{1}{3\phi} \int_0^1 x^3 dy$$

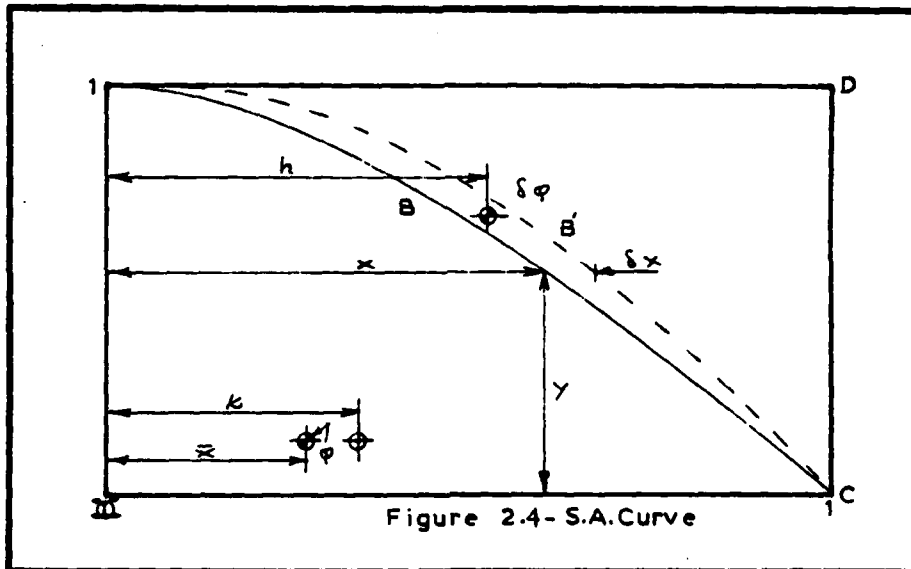
The only other difference between figures 2.3 and 2.4 is that figure 2.4 represents a hull form not having parallel middle body, i.e., $p = 0$, and as a consequence the length of the entrance and run equals that of the half length of the ship.

In referring to figure 2.4, it is recognized that in order to preserve the form of the parent at both the end of the ship, ($x = 1$) and the middle of the ship, ($x = 0$) an equation for δx of the following form would suffice:

$$\delta x = cx(1 - x)$$

where c is an as yet to be determined constant. It may be seen in Appendix A that the relationship for δx is:

$$\delta x = \frac{\delta \phi}{\phi(1-2\bar{x})} x (1 - x) \quad (2.3)$$



Clearly, the equation for δx is of the form of a second degree polynomial (parabola) whose values are 0 at $x = 0$ and $x = 1$, as desired. This relationship also shows that the amount by which any section in the parent is shifted is a function solely of the unchanged longitudinal position x and some as yet unknown value $\delta \phi$.

At this point we must turn our attention and consider both the entrance and run concurrently if we are to lend some significance to the quantity $\delta\phi$. If we desire to specify both C_p and LCB, we are in essence placing a requirement on the area under the sectional area curve and its moment about some axis (say $x = 0$). Since equation (2.3) applies independently to the entrance and run, it should be possible to select the $\delta\phi$'s of these respective regions such that when taken together, the ship has the values of C_p and LCB desired.

At this point we introduce the following quantities:

$\bar{z}$ = the distance of the parent ship's LCB from midships, normalized by the half length, (positive forward).

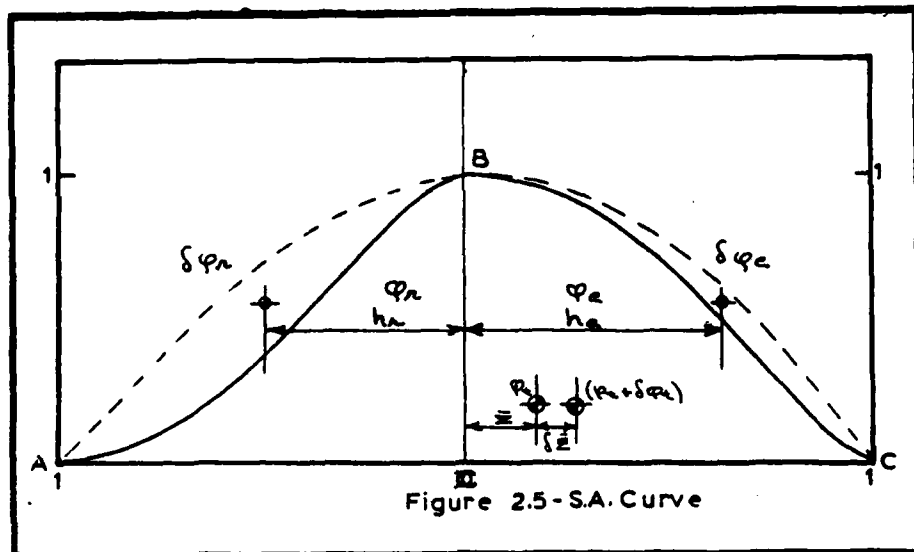
$\delta\bar{z}$ = the required shift in LCB to obtain that required for the daughter hull form (positive forward).

Prime (') - denotes derived forms.

Subscripts - e = entrance or forward half-body.

r = run or after half-body.

t = a property describing the entire ship.



Referring to figure 2.5 above, we may interpret the requirements on C_p and LCB mathematically as follows:

$$C_p = \phi \quad \phi'_t = \phi_t + \delta\phi_t = (\phi_e + \phi_r) + (\delta\phi_e + \delta\phi_r) \quad (2.4)$$

$$LCB = z \quad z' = \bar{z} + \delta\bar{z}$$

Summing moments of areas

$$z'(\phi_t + \delta\phi_t) = \bar{x}_e\phi_e + h_e\delta\phi_e - [\bar{x}_r\phi_r + h_r\delta\phi_r]$$

$$z' = \frac{1}{\phi_t} \{ \bar{x}_e\phi_e + h_e\delta\phi_e - [\bar{x}_r\phi_r + h_r\delta\phi_r] \} \quad (2.5)$$

If equations (2.4) and (2.5) are rearranged, the expressions for $\delta\phi_e$ and $\delta\phi_r$ may be obtained.

$$\delta\phi_e = \frac{2}{(h_e + h_r)} \{ \delta\phi_t (h_r + \bar{z}) + \delta\bar{z}(\phi_t + \delta\phi_t) \} \quad (2.6)$$

$$\delta\phi_r = \frac{2}{(h_e + h_r)} \{ \delta\phi_t (h_e - \bar{z}) - \delta\bar{z}(\phi_t + \delta\phi_t) \} \quad (2.7)$$

At this point the only variables which were not previously defined are h_e and h_r . The exact expression for these variables is, with the appropriate subscript:

$$h = \frac{2\bar{x} - 3k^2}{1 - 2\bar{x}} + \frac{\delta\phi}{\phi} \left\{ \frac{(\bar{x} - 3k^2 + 2r^3)}{(1 - 2\bar{x})^2} \right\} \quad (2.8)$$

While equation (2.8) contains $\delta\phi$, the very thing for which it is being used to calculate, it has been stated [2] that the leading term alone provides a very good approximation to h for "moderate" values of $\delta\phi$, i.e.,

$$h = \frac{2\bar{x} - 3k^2}{1 - 2\bar{x}} \quad (2.9)$$

Should it be desired to calculate $\delta\phi$ using equation (2.8), the solution will prove to be a quadratic which, while unwieldy, is certainly not unsolvable. The derivation of h may be found in Appendix A with the value of r defined as follows:

$$r^3 = \frac{1}{4\phi} \int_0^1 x^4 dy$$

While the above equations with the derivations in the Appendix illustrate the underlying theory, the following expressions represent the most general form of Lackenby's work.

$$\delta\phi_e = \frac{1}{B_e + B_r} \{2[\delta\phi_t(B_r + \bar{z}) + \delta\bar{z}(\phi_t + \delta\phi_t)] + C_e \delta p_e - C_r \delta p_r\} \quad (2.10)$$

$$\delta\phi_r = \frac{1}{B_e + B_r} \{2[\delta\phi_t(B_e - \bar{z}) - \delta\bar{z}(\phi_t - \delta\phi_t)] - C_e \delta p_e - C_r \delta p_r\} \quad (2.11)$$

In the following expressions the items refer to the entrance or run as appropriate:

$$\delta x = (1 - x) \left\{ \frac{\delta p}{1-p} + \frac{(x-p)}{A} [\delta\phi - \delta p \frac{(1-\phi)}{(1-p)}] \right\} \quad (2.12)$$

The practical limits on $\delta\phi_e$ and $\delta\phi_r$ are:

$$\delta\phi = \frac{\delta p(1-\phi) \pm \frac{1}{2}A[1 - \frac{\delta p}{1-p}]}{1-p} \quad (2.13)$$

A, B and C are calculated as follows:

$$A = \phi(1-2\bar{x}) - p(1-\phi) \quad (2.14)$$

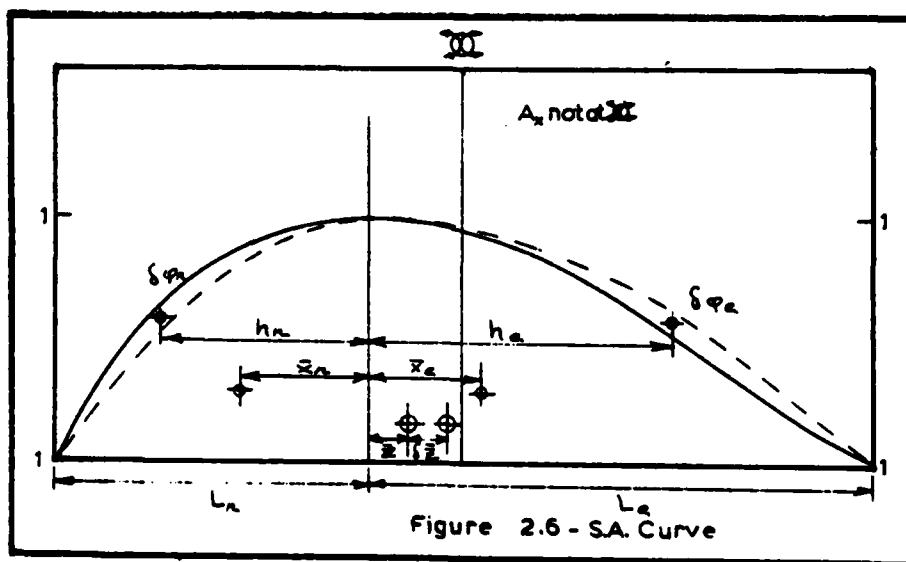
$$B = \frac{\phi}{A} \{2\bar{x} - 3k^2 - p(1-2\bar{x})\} \quad (2.15)$$

$$C = \frac{1}{1-p} \{B(1-\phi) - \phi(1-2\bar{x})\} \quad (2.16)$$

It should be recognized that in the above equations the necessary section shift, δx , is a function of the new values of C_p and LCB, and the properties of the original parent hull form.

2.2.3 Modification of Lackenby's Method to Accommodate Hull Forms with Maximum Sections not at Midships.

It should be realized that in all of the preceding developments it was assumed that the section of maximum area fell at midships. While this is certainly the case for a large class of vessels, it is virtually never true for contemporary cruisers and destroyers. It is for this reason that a new set of equations was sought while still adhering to the basic philosophy of longitudinally shifting sections.



Referring to figure 2.6, the definition of terms is, once again, consistent with the preceding section. There are, however, two very important changes. First, where $\bar{z}$ and $\delta\bar{z}$ were originally normalized by the ship's half length, they are now normalized by twice that length or the length between perpendiculars, L . Second, x , the local longitudinal position of any section is normalized by the appropriate length of entrance or run L_e or L_r respectively. Also all values of x and $\bar{z}$ take as their origin the station of maximum sectional area.

The basic relationship for δx is still of the form $\delta x = cx(1-x)$ or $\delta x = \frac{\delta\phi}{\phi(1-2x)} x(1-x)$, the same as equation (2.3) previously. However, equations (2.4) and (2.5) now become:

$$\phi_t = \frac{1}{L} \{L_e(\phi_e + \delta\phi_e) + L_r(\phi_r + \delta\phi_r)\} \quad (2.17)$$

$$\bar{z}' = \bar{z} + \delta\bar{z} = \frac{1}{L^2\phi_t} \{L_e^2(\phi_e\bar{x}_e + \delta\phi_e h_e) - L_r^2(\phi_r\bar{x}_r + \delta\phi_r h_r)\} \quad (2.18)$$

These two equations are solved simultaneously for $\delta\phi_e$ and $\delta\phi_r$ in Appendix B, the results of which are listed below:

$$\delta\phi_e = \frac{1}{L_e^2 h_e + L_e L_r h_r} \{L_r^2 \phi_r \bar{x}_r - L_e^2 \phi_e \bar{x}_e + \bar{z}' L^2 \phi_t' - L_r h_r (L_e \phi_e + L_r \phi_r - L \phi_t')\} \quad (2.19)$$

$$\delta\phi_r = \frac{1}{L_r^2 h_r + L_r L_e h_e} \{L_e^2 \phi_e \bar{x}_e - L_r^2 \phi_r \bar{x}_r - \bar{z}' L^2 \phi_t' - L_e h_e (L_e \phi_e + L_r \phi_r - L \phi_t')\} \quad (2.20)$$

It was these two equations along with equation (2.3) which proved to give very satisfactory results for several sample calculations.

2.2.4 A Method by which Constraints may be Placed on the Design Waterline

In the preceding development the designer had no control over the shape of the design waterline. Because the longitudinal center of flotation LCF, was felt to be an important parameter in determining a ship's performance in a seaway, Moor [8] further developed the method of Lackenby to include control over the design waterline. It was with his revised method that Moor and his colleagues developed four distinctly different models with only their midships section identical. They were thus able to cut these four models in half to generate sixteen uniquely different hull forms.

An interesting side light of this experiment was that the parent form used was that of a fast twin-screw currently in service whose maximum section was abaft midships. They, therefore, had to first swing the original area curve to place the maximum section at midships and then proceed with the new method of modification. Although having the maximum section at amidship may have proven to be more tractable for

the creation of the models, it was not necessary for the application of Lackenby's method. This fact was demonstrated in the previous section.

The essence of Moor's method is that the sectional area curve and the design waterline are both altered in the sense of Lackenby to produce the desired values of C_p , LCB, C_w and LCF. At this point a new factor is introduced; the ratio between the sectional area ordinate and the design waterline ordinate is calculated for both the parent and derived hull forms. These ratios are then plotted as a function of ship length and it is from this curve that the longitudinal shift of sections is determined. Figures 2.7 and 2.8 illustrate the sectional area and design waterline curves and the area/waterline ratio curve respectively.

Referring to figure 2.8, to obtain the offsets for a particular section R_d in the derived form, section R_p in the parent is used as a basis. The reason for selecting station R_p in the parent is that it is the closest section to R_d with the same value of area/waterline ratio. The offsets of section R_p are then multiplied by the ratio of the beam coefficient in the derived form at section R_d to that of the parent at section R_p , i.e., B_d/B_p . These values may be seen in figure 2.7. Additionally, if the maximum beam of the derived form is different from that of the parent, the offsets of section R_p also have to be multiplied by the ratio

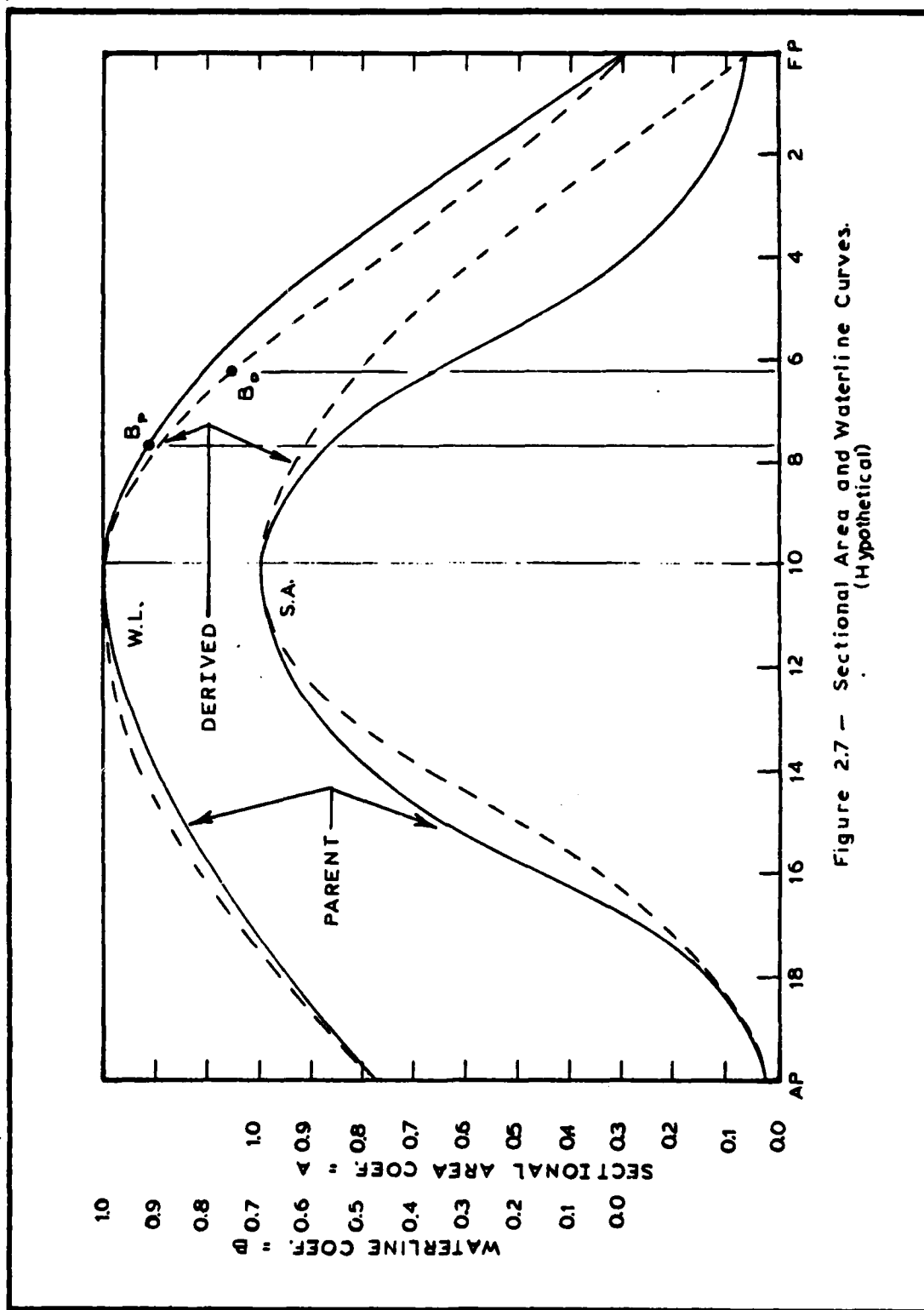
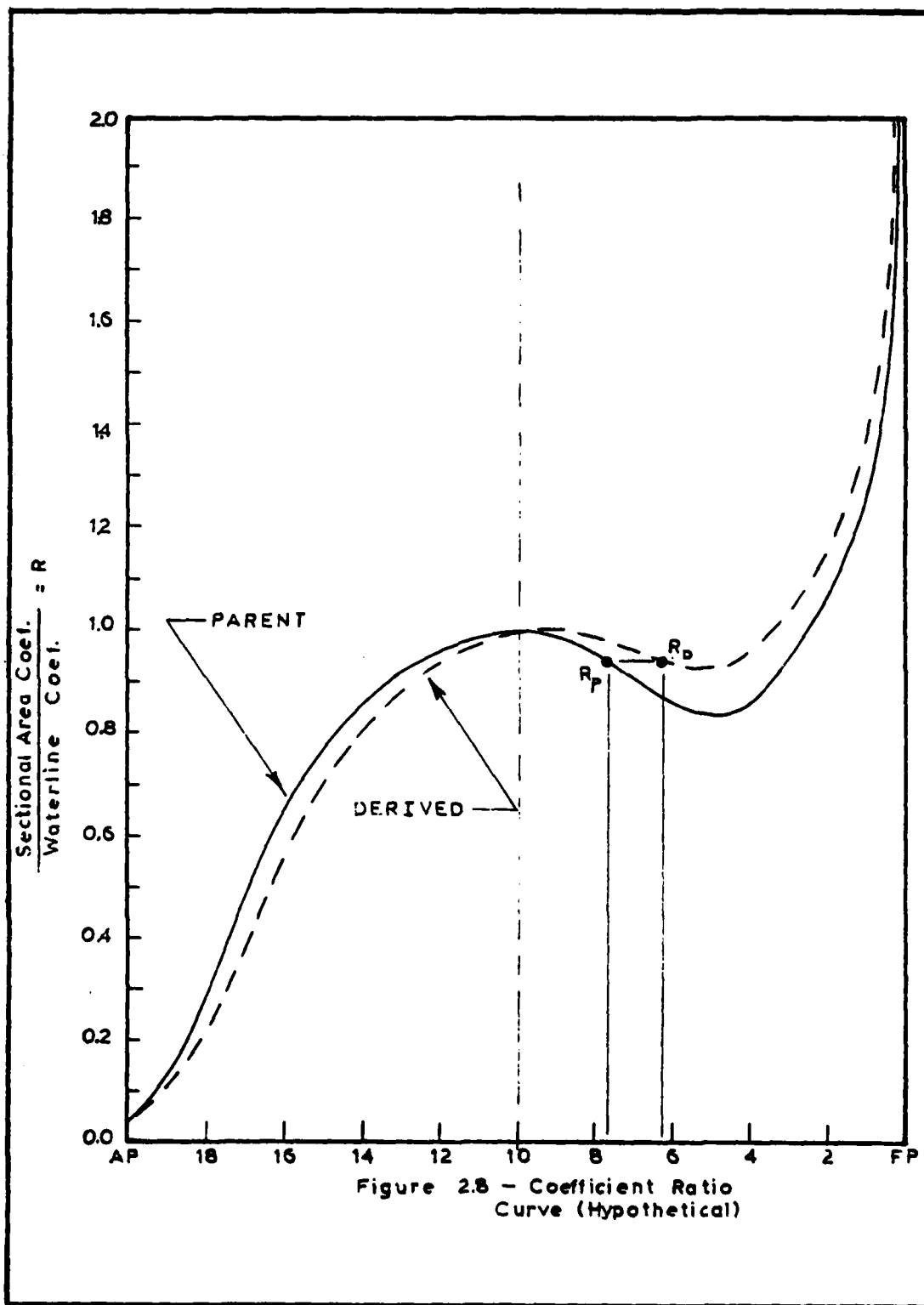


Figure 2.7 — Sectional Area and Waterline Curves.
(Hypothetical)



of the maximum beam in the derived form to the maximum beam in the parent, i.e., b_{dmax}/b_{pmax} . Therefore, the equation for any offset b_d in the derived form is:

$$b_d = b_p \frac{B_d}{B_p} \frac{b_{dmax}}{b_{pmax}} \quad (2.21)$$

It can be seen in figure 2.8 that there are regions of ambiguity. Such is the case where the derived curve lies below a minimum in the parent curve. It has been this author's experience confirming that in reference [8], that the regions which cannot be explicitly be defined may be faired in after defining the sections on either side.

The one remaining undesirable characteristic occurs in regions where there is some form of keel rise, i.e., the fore foot or skeg region. In these areas, if the draft of the parent is proportionally altered to equal that of the derived form, there is a concomitant and undesirable change in the area of the section. It is to this matter which the next section is addressed.

2.2.5 A Method by which Constraints may be Placed on the Ship's Profile

Since Moor's method proved capable of constraining both the sectional area curve and the design waterline, it was decided to extend the method to include the centerline profile of the ship. The actual mechanics require only the introduction of a local draft coefficient, (local draft/maximum draft) into the denominator of the area/beam ratio. This new ratio, (Area/Beam/Draft), is graphed and the sectional shifts determined from this graph. In determining the new offsets not only are the offsets of the parent modified transversely as described in the previous section, they are also altered in the vertical sense. This alteration is accomplished by using the water plane as a reference and moving the waterlines below a distance proportional to the ratio of the derived form draft coefficient/parent draft coefficient. Also if there is a difference in the maximum draft of the derived form and parent, the waterlines are altered by this ratio as appropriate.

As was mentioned in the background section of this chapter, section 2.1, for this modification technique to be mathematically rigorous the sections of maximum area, maximum design waterline breadth and maximum draft must be coincident. If this is not the case, the actual areas of

the sections generated will be consistantly different by a very small amount from what is desired. From the few examples this author has worked, it is estimated the difference in the value of C_p obtained and that desired is on the order of 1%. The explanation of this is seen in Appendix C.

3. Mathematical Representation of the Lines of a Ship

3.1 Background

It was established in section 2.1 that before the naval architect attempts to actually draw the lines of a new ship, he must have a "firm" description of the new design as represented by the various coefficients and curves of form. Examples of these, as cited previously, are: the sectional area curve, and hence C_p and LCB, the design waterline curve, (C_w and LCF), the principle dimensions and perhaps specific information about the geometry of the midship section or stern region. These characteristics should represent what the designer feels is the "optimum" solution to his set of requirements. The naval architect now has to create one or more, of a possible infinity of, design candidates which fulfill his descriptive coefficients.

The traditional method for drawing the various lines of the ship is with the use of long, continuous strips of wood, metal, or more recently plastic, held in the desired position by weights. These tools are called splines and ducks respectively. The curves produced by this method were continuous but often times contained unwanted waviness. Removing these unwanted undulations, while preserving the

desired character of the line, is a process referred to as fairing. This topic, and the implications of placing a mathematical interpretation on it, are discussed in the next chapter. Not only did the naval architect have to generate smooth curves which pleased him visually, there also had to be a consistency in location of the surface points when observed from the different views. This is sometimes referred to as cross-fairing and is also addressed in the final chapter. It is the fairing, and cross-fairing, which represents a very large part of the manual design effort.

It was recognized long ago, that if the ship design process was to be automated to any degree, a technique to represent the lines of the ship mathematically would have to be developed. This is especially true today where much of the work is to be done by high speed digital computers. Not only must the designer/programmer provide the mathematical algorithms for representing the ship's lines, he must also provide the logic necessary for the computer to duplicate the heretofore trial and error methods of the draftsman. The alternative to programming the logic however, would be to give the system an interactive man-machine interface at the decision points. It is, however, the mathematical representation of these ships' lines to which this chapter is devoted.

3.2 Development

3.2.1 Derivation of the Spline Cubic Equation using a Variational Approach

There are essentially two different methods by which one may arrive at a mathematical representation for ships lines.

1. Select some mathematical function with several unspecified parameters whose values may be determined by some accuracy criteria and boundary conditions. Typical of this approach is the use of a polynomial and a least squares fit criterion.
2. Choose some smoothness and closeness of fit criteria such that, when taken together with the boundary condition, the function and parameters are determined.

It is this second method, based on a variational smoothness criterion, that will be developed in this chapter.

In general these variational methods involve the minimization of the integral of some linear combination of the squares of the various derivatives of the function sought. In the case where the equation of the flexible spline is sought, the "smoothness" criterion is taken to be the minimization of the strain energy in the spline.

Mathematically this may be represented by [9]:

$$\int_a^b L(y, y', y'', \dots y^{(n)}, x) dx = \min \quad (3.1)$$

where, for the spline equation (3.1) becomes:

$$\int_s c k^2 ds \quad (3.2)$$

s = the total path length

c = flexural rigidity of the beam

k = curvature defined mathematically as:

$$= \frac{y''}{(1+y'^2)^{3/2}}$$

ds = elemental arc length

$$= \sqrt{1 + y'^2} dx$$

If these values are substituted into equation (3.2) the smoothness criteria becomes:

$$I = \int_{x_1}^{x_m} \frac{y''^2}{(1+y'^2)^{5/2}} dx = \min \quad (3.3)$$

To complete the variational problem one must also consider a "closeness of fit" criterion which takes the following form:

$$N = \int_a^b F(y, y', y'', \dots, y^{(n)}, f, x) dx \quad (3.4)$$

It may be seen in most any text on the calculus of variations, e.g., [10] that the criteria for smoothness and closeness of fit may be combined by the introduction of the unknown Lagrange multiplier λ [10]. The results of combining equations (3.1) and (3.4) into a single variational problem is:

$$\delta \int_a^b \{L(y, y', y'', \dots, y^{(n)}, x) + \lambda F(y, y', y'', \dots, y^{(n)}, f, x)\} dx = 0 \quad (3.5)$$

A necessary condition for the integral in (3.5) to be stationary is:

$$\begin{aligned} & \frac{\partial(L+\lambda F)}{\partial y} - \frac{d}{dx} \left[\frac{\partial(L+\lambda F)}{\partial y'} \right] + \frac{d^2}{dx^2} \left[\frac{\partial(L+\lambda F)}{\partial y''} \right] - \\ & \dots (-1)^n \frac{d^n}{dx^n} \left[\frac{\partial(L+\lambda F)}{\partial y^{(n)}} \right] = 0 \end{aligned} \quad (3.6)$$

with boundary conditions:

$$\begin{aligned} & \left\{ \left(\frac{\partial L}{\partial y'} - \frac{d}{dx} \frac{\partial L}{\partial y''} + \frac{d^2}{dx^2} \frac{\partial L}{\partial y'''} - \dots \right) \delta y + \left(\frac{\partial L}{\partial y''} - \frac{d}{dx} \frac{\partial L}{\partial y'''} + \frac{d^2}{dx^2} \frac{\partial L}{\partial y^{(4)}} - \right. \right. \\ & \left. \left. \dots \right) \delta y' + \left(\frac{\partial L}{\partial y'''} - \frac{d}{dx} \frac{\partial L}{\partial y^{(4)}} + \dots \right) \delta y'' + \dots + \frac{\partial L}{\partial y^{(n)}} \delta y^{(n-1)} \right\}_a^b = 0 \end{aligned} \quad (3.7)$$

Equation (3.6) is known as the Euler differential equation for the variational problem presented in equation (3.5). The usual procedure for solving this system of equations is to first solve the Euler equation (3.6) in conjunction with the boundary conditions, expressed in equation (3.7). This solution results in an equation of the form $y = f(\lambda, x)$. This equation is then substituted into the "closeness of fit", or accuracy criterion of equation (3.4), to determine the value, or values of λ .

Another aspect of the mechanics of this procedure is revealed when the accuracy criterion requires that the resulting relationship for $y = f(x)$ pass through a discrete set of data points. Such is the case for a "colocative spline", or a spline made to pass exactly through discrete data points. In this instance the integral in equation (3.4) would become a summation. However, in order to preserve the consistency and similarity of working with integrals in both portions of equation (3.5), the Dirac delta function may be introduced into F .

Redirecting attention to equation (3.3), it will be noted that due to the complexity of the integrand, the differential element of strain energy, the result of equation (3.5) will not be closed form. In order to simplify the above integrand it is assumed that the demonimator, $(1+y'^2)^{5/2}$, is

approximately 1. Or y'^2 is very small. While this may not be the actual case, if the value of y' is linearized as being the value of the slope of the chord between two successive data points, the integral in (3.3) might be thought of as follows:

$$I^* = \int_{x_1}^{x_m} W(x) y''^2 dx = \min \quad (3.8)$$

If we were to take some mean value for $W(x)$ for the entire domain of x , the minimization would be similar to the minimization of:

$$\int_{x_1}^{x_m} y''^2 dx \quad (3.9)$$

It is also this simplification which will allow us to calculate a closed form solution to $y = f(x)$.

One other simplification which will be made, without harm to generality, is to assume that $x \in [0,1]$. That is $0 < x_1 < x_2 < \dots < x_{m-1} < x_m < 1$. At this point we must actually define the accuracy criterion. Assuming that the curve passes exactly through the data points, we may say:

$$y(x_i) = f_i \quad i = 1 \dots m \quad (3.10)$$

Translating the conditions of (3.9) and (3.10) into the single variational problem of the form of (3.5) we have:

$$\delta \int_0^1 \{y'^2 + 2 \sum_{j=1}^m \lambda_j \delta(x-x_j)(f-y)\} = 0 \quad (3.11)$$

It should be recognized that the first delta (to the left of the integral sign) is the symbol for the variation, while the second is the Dirac delta function. f is any "candidate" function passing through the data points $(x_j, y(x_j))$ and the λ_j 's are to be determined from the accuracy criteria of equation (3.10).

The resulting Euler equation (3.6) is:

$$y^{iv} - \sum_{j=1}^m \lambda_j \delta(x-x_j) = 0 \quad (3.12)$$

The solution of this equation generates a function of the following form:

$$y(\lambda, x) = \frac{1}{12} \sum_{j=1}^m \lambda_j |x-x_j|^3 + Ax^3 + Bx^2 + Cx + D \quad (3.13)$$

Therefore the value of y and the integration constants A , B , C , and D are all linear functions of the λ_j 's.

The boundary equation (3.7) for this specific problem turns out to be of the following form:

$$[-y'' \delta y + y'' \delta y']_0^1 = 0 \quad (3.14)$$

It must also be noted that (3.13) is valid for $0 \leq x \leq 1$.

While it is not intended to determine equation (3.13) for every possible situation, suffice it to say that the values of the λ_j 's and the constants A, B, C, and D are uniquely determined by the coordinate points and the end conditions of the curve [9]. The essential points to be made are:

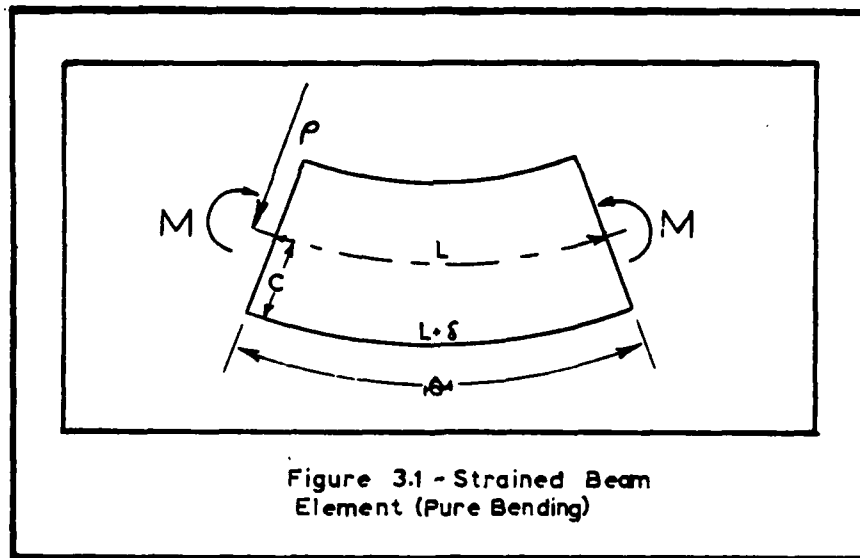
1. For the criteria used, the equation for the spline as obtained by the variational approach is of the form of a multi-coefficient third degree polynomial.
2. Where the curve is defined over the region (0, 1) by m data points, there is also a need for four additional pieces of information to satisfy, and fully solve equation (3.13).

It will now be demonstrated that the form of equation (3.13) is also supported by the theory developed for the small deflection of elastic beams. As it turns out, the simplifying assumptions made in the preceding derivation are exactly that

which will be made in the small deflection theory. Nevertheless, the consistency of results lends much reassurance.

3.2.2. Deflections due to Bending of a Simple Elastic Beam

In virtually any undergraduate strength of materials course the subject of beam deflections may be presented in several different ways [11]. However, it will be the method of multiple integration which will be developed here.



Referring to figure 3.1 above, it may be said that the element of the beam deforms about the neutral axis and that as a result, transverse plane sections remain plane after

deformation. This results in an elongation of those fibers outside of the neutral axis and a compression of those fibers inside. Also, the amount of distortion is proportional to the distance from the neutral axis. This being the case, the following equations hold:

$$\theta = \frac{L}{\rho} = \frac{L+\delta}{\rho+C}$$

or, by rearranging,

$$\frac{C}{\rho} = \frac{\rho}{L} = \epsilon = \frac{\delta}{E} = \frac{M}{I}$$

therefore:

$$\frac{1}{\rho} = \frac{M}{EI} \tag{3.15}$$

In the above equations the following definitions apply:

M = bending moment

E = the stiffness or Young's modulus

I = the moment of inertia of the cross section

ρ = the radius of curvature of the beam, measured to the neutral axis

EI = the "flexural stiffness" or "rigidity"

The mathematical expression for the radius of cruvature, or more traditionally the curvature, K, is defined as follows [12]:

$$\frac{1}{\rho} = \kappa = \frac{d^2y/dx^2}{[1 + (\frac{dy}{dx})^2]^{3/2}} \quad (3.16)$$

It is to this equation which the simplification is applied. For actual beams it is assumed that the value of $\frac{dy}{dx}$ is very small, hence, $(\frac{dy}{dx})^2 \ll 1$. If this is the case, the expression for curvature becomes:

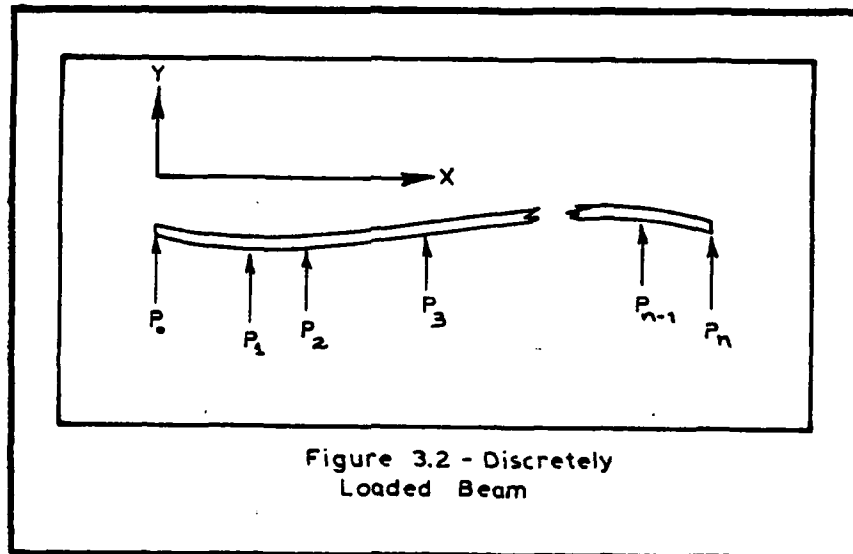
$$\frac{1}{\rho} = \kappa = \frac{d^2y}{dx^2} \quad (3.17)$$

Therefore, when equations (3.15) and (3.17) are combined, the results are as follows:

$$EI \frac{d^2y}{dx^2} = M(x) \quad (3.18)$$

Here, $M(x)$ is meant to indicate that the bending moment is a function of x .

At this point we must address ourselves to the equation for the bending moment in a beam. Specifically, we will look at the results of loading a uniform elastic beam with concentrated point loads; remembering that this case most closely approximates the naval architect's ducks and splines.



It is advantageous, at this time, to introduce the concept of the singularity function defined as follows:

$$\langle x - x_i \rangle^n = \begin{cases} 0, & x \leq x_i \\ (x - x_i)^n & x > x_i \end{cases} \quad (3.19)$$

With the aid of the singularity function, and in reference to figure 3.2, the bending moment equation obtained from the application of concentrated point loads is:

$$M(x) = P_0 x + P_1 \langle x - x_1 \rangle + P_2 \langle x - x_2 \rangle + \dots$$

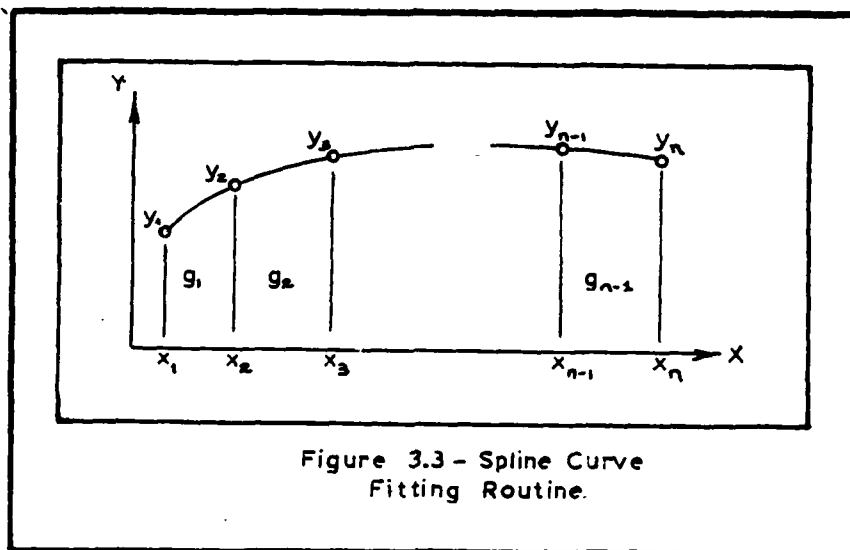
$$P_n \langle x - x_n \rangle \quad (3.20)$$

It should be obvious that when equation (3.20) is substituted into equation (3.18) the result may be twice integrated to obtain an equation for $y = f(x)$ which is of the following form:

$$EIy(x) = A + Bx + \frac{1}{6} \{ P_0 x^3 + P_1 \langle x - x_1 \rangle^3 + P_2 \langle x - x_2 \rangle^3 \\ \dots P_n \langle x - x_n \rangle^3 \} \quad (3.21)$$

The above equation is of a form very much the same as equation (3.13). The essential difference is that in equation (3.13), the values of the end forces P_0 and P_n were still unknowns requiring the statement of two conditions at each end of the beam. It should also be apparent that the resulting values of the λ_i 's are nothing more than $2EI$ times the forces required to keep the beam in equilibrium. Therefore, based on the results of this and the previous section, it will be accepted without further discussion that the third degree polynomial, or cubic, is an adequate model of the ducks and splines of the naval architect. Hence the term "spline cubic".

3.2.3 Piece-wise Continuous Cubic Polynomial Approximation



Referring to figure 3.3 above, it is desired to approximate the curve $y(x)$ by some series of cubic functions of the form:

$$g_j(x) = a_j(x-x_j)^3 + b_j(x-x_j)^2 + c_j(x-x_j) + d_j \quad (3.22)$$

where j represents the j^{th} interval bounded by x_j and x_{j+1} , and $1 \leq j \leq n - 1$, n being the number of data points. In order for these segments to be continuous we impose the following conditions:

$$(1) \begin{cases} g_j(x_j) = y_j \\ g_{n-1}(x_n) = y_n \end{cases} \quad j = 1, 2, \dots, n-1$$

$$(2) \quad g_j(x_{j+1}) = g_{j+1}(x_{j+1}) \quad j=1, 2, \dots, n-2$$

$$(3) \quad g'_j(x_{j+1}) = g'_{j+1}(x_{j+1}) \quad j=1, 2, \dots, n-2$$

$$(4) \quad g''_j(x_{j+1}) = g''_{j+1}(x_{j+1}) \quad j=1, 2, \dots, n-2$$

From equation (3.22) it should be recognized that to fully describe the curve requires $4(n-1)$ unknown coefficients. However, the above equations provide $4n-2$ conditions. We therefore require two more conditions. The obvious choice for these two additional constraints would be to specify the end conditions for the beam. Specifically, you would specify either g' or g'' at the ends.

For the sake of brevity the remainder of this derivation will be abridged to include only the essential equations. For a complete and detailed description of this procedure the reader is referred to references [13] and [14].

Continuing with the derivation, the following definitions will prove useful.

$$h_j = x_{j+1} - x_j \quad (3.23)$$

$$D_j = (y_{j+1} - y_j)/h_j \quad (3.24)$$

We may now relate the unknown coefficients in the following manner:

$$a_j = \frac{1}{6h_j} (s_{j+1} - s_j) \quad (3.25)$$

$$b_j = \frac{s_j}{2} \quad (3.26)$$

$$c_j = D_j - \frac{h_j}{6} (2s_j + s_{j+1}) \quad (3.27)$$

$$d_j = y_j \quad (3.28)$$

Substituting these equations into condition (3) will generate a relationship between successive values of s_j of the following form:

$$s_j h_j + 2(h_j + h_{j+1}) s_{j+1} + s_{j+2} h_{j+1} = 6(D_{j+1} - D_j) \quad (3.29)$$

$$j = 1, 2, \dots, n-2$$

Equation (3.29) will thus generate the following system of equations:

$$\begin{bmatrix}
 2(h_1+h_2) & h_2 & 0 & \dots & \\
 & h_2 & 2(h_2+h_3) & h_3 & \\
 & \cdot & & & \\
 & \cdot & & & \\
 & \cdot & & & \\
 & & h_{n-3} & 2(h_{n-3}+h_{n-2}) & h_{n-2} \\
 & & & h_{n-2} & 2(h_{n-2}+h_{n-1})
 \end{bmatrix}
 \begin{bmatrix}
 s_2 \\
 s_3 \\
 \cdot \\
 \cdot \\
 \cdot \\
 s_{n-1}
 \end{bmatrix}
 =
 \begin{bmatrix}
 6(D_2-D_1)-s_1h_1 \\
 6(D_3-D_2) \\
 \cdot \\
 \cdot \\
 \cdot \\
 6(D_{n-1}-D_{n-2})-h_{n-1}s_n
 \end{bmatrix}$$

(3.30)

It can be seen that, for any point x_j , s_j is the curvature at that point. For the above system of equations if the curvature is known at the end points, i.e., s_1 and s_n , the curve will be completely defined.

If instead of curvature the slope is specified at the end point, the above matrix will be modified slightly. In this case, the value of the curvature at the end point will be unknown. For the situation where the beginning slope is specified the following changes will occur. From equation (3.27):

$$D_1 - \frac{h_1}{6} (2s_1 + s_2) = c_1 = t_1$$

$$\text{or} \quad 2s_1h_1 + s_2h_1 = 6(D_1 - t_1) \quad (3.31)$$

The effect on the matrix system will be to change the currently existing top row and then add another top row and left column as follows:

$$\begin{bmatrix} 2h_1 & h_1 & \dots \\ h_1 & 2(h_1+h_2) & h_3 & \dots \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} = \begin{bmatrix} 6(D_1-t_1) \\ 6(D_2-D_1) \end{bmatrix}$$

If the end slope is specified a change of similar form takes place only adding a row to the bottom and a column to the right side as follows:

$$\begin{bmatrix} \dots & h_{n-2} & 2(h_{n-2}+h_{n-1}) & h_{n-1} \\ \dots & h_{n-1} & 2h_{n-1} \end{bmatrix} \begin{bmatrix} s_{n-1} \\ s_n \end{bmatrix} = \begin{bmatrix} 6(D_{n-1}-D_{n-2}) \\ 6(-D_{n-1}+t_n) \end{bmatrix}$$

The form of the above matrix is tridiagonal and lends itself to rapid solution by a recursive relationship [14]. This fact will save a significant amount of computational time when reduced to a computer algorithm.

3.2.4 The Rotating Spline [15]

One particular disadvantage of the "piece-wise cubic" method developed in section 3.2.3 is that it will not provide a solution for curves having infinite slopes. It is for this reason that method of the "rotating spline" was developed. Referring to figure 3.4 it may be related that this procedure is merely a modification of the "piece-wise cubic" technique.

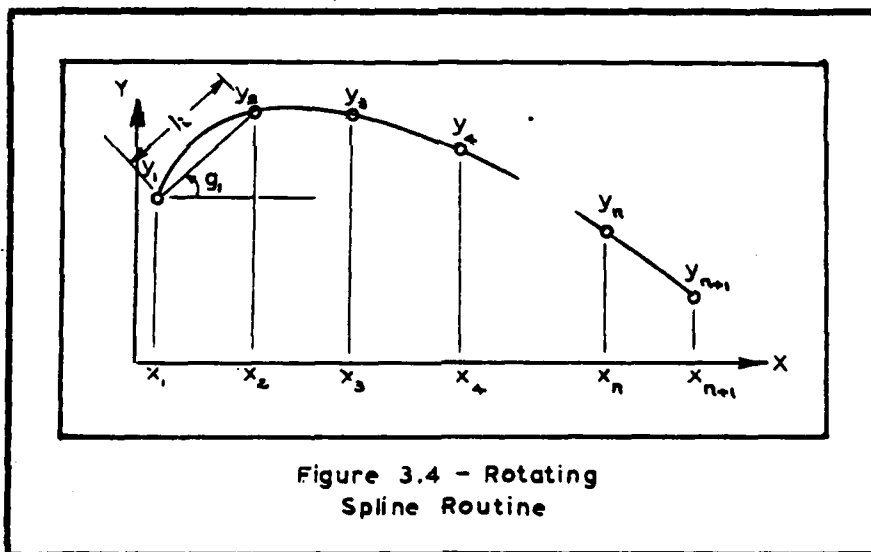


Figure 3.4 - Rotating
Spline Routine

For this method of curve approximation a cubic polynomial is generated for each interval as before. However, in this case the coordinate system is redefined for each

interval, and the cubic equation is generated with respect to this local coordinate system.

Appendix E contains the steps necessary to generate the computer algorithm. There is, however, one definite disadvantage to using a rotated coordinate system. In order to obtain an interpolated ordinate on the curve the following two parametric equations must be used.

$$x = x_i + (x_{i+1} - x_i)t - (y_{i+1} - y_i)t(1-t)[a_i(1-t) - b_it] \quad (3.32)$$

and

$$y = y_i + (y_{i+1} - y_i)t + (x_{i+1} - x_i)t(1-t)[a_i(1-t) - b_it] \quad (3.33)$$

Both of these equations are third degree polynomials in the parameter t . To solve for some value y of the point (x,y) in the unrotated coordinate system, x is used in equation (3.32) to solve for t such that $0 \leq t \leq 1$ and t is also real. This value of t is then used in equation (3.33) to calculate y . It should be pointed out that the quantities a_i and b_i were determined previously as described in Appendix E.

In summary, we have shown by two methods, variational calculus and simple beam theory, that the third degree or cubic polynomial provides a good representation of the thin elastic spline used in drawing ships lines. There was,

however, the disadvantage that the equations were not capable of representing curves having infinite slopes. For this reason the method of the rotating spline was introduced. This parametric method permits the representation of virtually any continuous curve, including those which are non singular.

4. Mathematical Fairing of Lines

4.1 Background

It has been stated previously that the lines fairing process can be the most time consuming aspect of the ship design cycle. For this very reason fairing becomes a prime candidate for automation. The difficulty, however, lies in the fact that obtaining a universally accepted mathematical definition of a faired line, or the fairing process itself, is a virtual impossibility. Perhaps the most general definition, and one which would prove the least restrictive, is the following:

A faired line is one which retains the desired "character" but eliminates any undesired waviness or fluctuations.

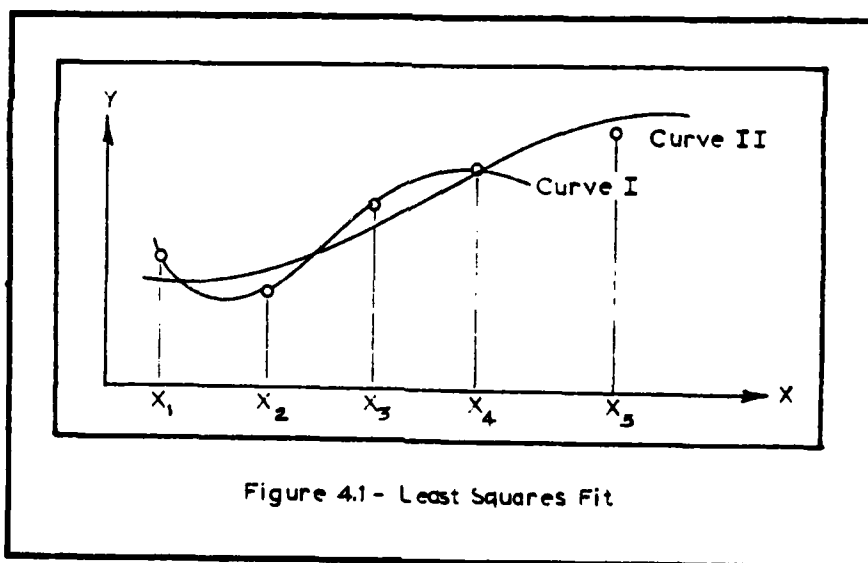
It will be shown in the following sections that this result may be achieved by fitting, in a least squares sense, a third degree polynomial to a set of four or five data points. The number being dependent upon the desired boundary conditions. In addition to the similarity of the third degree polynomial to the form of an elastic spline, the polynomial also provides the capability of introducing a desired inflection point into a series of data points. It

also prohibits the introduction of multiple inflection points and undesired waviness, also an asset. Because of these characteristics and the excellent results demonstrated in reference [16], this "least squares" criteria was employed as the foundation of the fairing process.

4.2 Development

4.2.1 The Least-Squares Criteria for Defining the Cubic Curve

It was established in chapter three that the cubic polynomial would provide a "good" approximation of the shape of a spline used to construct the lines of a ship. What remains to be shown is how these polynomials are applied in order to generate the faired position of a set of data points.



$$P(x) = a_0 + a_1x + a_2x^2 + a_3x^3 \quad (4.1)$$

Referring to equation (4.1) above it should be recognized that, in order to uniquely specify the cubic polynomial in its general form, four independent pieces of information are required. The result of applying this information is to determine the values of a_0 , a_1 , a_2 and a_3 . This will produce a curve which exactly conforms to the given requirements. This is illustrated as curve I in figure 4.1 where the curve is required to pass exactly through the first four data points. The disadvantage of using this type of curve is that, since it is required to pass exactly through the given data points, it is unable to modify their position. It is this alteration however that is necessary if the curve is to be "faired".

As stated above, four pieces of information are required to uniquely specify the cubic polynomial. If, however, we were to over specify the requirements of the curve and then demand that the solution satisfy these requirements in some "best possible" but not exact manner, we begin to get a feel for how fairing can be produced. Mathematically this can be stated as follows.

Referring to curve II in figure 4.1 we will require that our resulting curve pass as close as possible to the five given data points. Usually this translates into a

mathematical form by requiring that the sum of the squares of the distances between the curve and the given data points should be minimized.

$$s = \sum_{i=1}^5 [y_i - (a_0 + a_1x_i + a_2x_i^2 + a_3x_i^3)]^2 \quad (4.2)$$

or

$$\frac{\partial s}{\partial a_0} = \frac{\partial s}{\partial a_1} \dots = \frac{\partial s}{\partial a_3} = 0$$

These derivatives generate the following system of normal equations which can be solved for $a_0 \dots a_3$.

$$\begin{bmatrix} s_0 & s_1 & s_2 & s_3 \\ s_1 & s_2 & s_3 & s_4 \\ s_2 & s_3 & s_4 & s_5 \\ s_3 & s_4 & s_5 & s_6 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} t_0 \\ t_1 \\ t_2 \\ t_3 \end{bmatrix} \quad (4.3)$$

Where s_k and t_k are defined as follows:

$$s_k = \sum_{i=1}^5 x_i^k \quad (4.4)$$

$$t_k = \sum_{i=1}^5 y_i x_i^k \quad (4.5)$$

It can be seen that curve II in figure 4.1, while not passing exactly through the data points, passes "fairly close" and also displays a smooth and continuous character. It is this closeness of proximity, or minimization of the least squares difference, procedure that is the essence of the fairing criteria used in this thesis.

The following sections will develop the equations for line segments whose end position and slope or just end position are fixed. First, however, a brief explanation of how these segments are applied to fair a complete line or set of data points.

4.2.2 The Moving Strip Method

In the procedure described above it was seen that a least squares spline was passed through five data points and the points on that line were then considered to be fair. In order to fair a complete set of initially unfair points consider only five points at a time, i.e., P_k to P_{k+4} . After passing a least squares spline through these five points obtain the faired position of point P_{k+2} . Then move

the strip one unit ($k = k+1$) and consider the next five points, P_k to P_{k+4} , fairing P_{k+2} . For each step we could use previously faired values for P_k and P_{k+1} expecting our final solution to be obtained more rapidly. By walking this strip of five points through the entire set of data the faired position of each point may be obtained. This procedure may also be found in references [16, 17].

The following three sections develop the equations needed when considering data points whose boundary conditions are of the following type:

1. Free end--the position and slope of the end is unspecified.
2. Pinned end--the end position is fixed but free to rotate.
3. Clamped end--end position and slope are fixed.

4.2.2.1 STRIP1: Fairing an interval with free ends.

This procedure is the same as that developed in section 4.2.1. It is this routine which is used to fair the center data point (P_{k+2}) of five interior data points, i.e., $P_k \neq P_1$ and $P_{k+4} \neq P_n$, where P_1 and P_n are the first and last points respectively in the set of given data. This routine is also used to fair P_1 , P_2 , P_{n-1} and P_n for the case where the ends are free to both rotate and translate.

Repeating equations (4.3) to (4.5) for convenience.

$$\begin{bmatrix} s_0 & s_1 & s_2 & s_3 \\ s_1 & s_2 & s_3 & s_4 \\ s_2 & s_3 & s_4 & s_5 \\ s_3 & s_4 & s_5 & s_6 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} t_1 \\ t_1 \\ t_2 \\ t_3 \end{bmatrix} \quad (4.3)$$

and

$$s_k = \sum_{i=1}^5 x_i^k \quad (4.4)$$

$$t_k = \sum_{i=1}^5 y_i x_i^k \quad (4.5)$$

also

$$P(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 \quad (4.1)$$

Therefore the faired position of the second and third points in the five point strip are:

$$\bar{y}_2 = P(x_2) \text{ and } \bar{y}_3 = P(x_3)$$

For the case of a free end point the first point becomes:

$$\bar{y}_1 = P(x_1)$$

It would be fair to expect that the equation $P(x)$ would be most representative of the actual curve at its interior regions where uncertainty about end conditions would have less effect. For this reason only the center point of a strip is recalculated as being faired, i.e., P_{k+2} as opposed to recalculating both P_k and P_{k+1} .

4.2.2.2 STRIP2: Fairing an interval with pinned end.

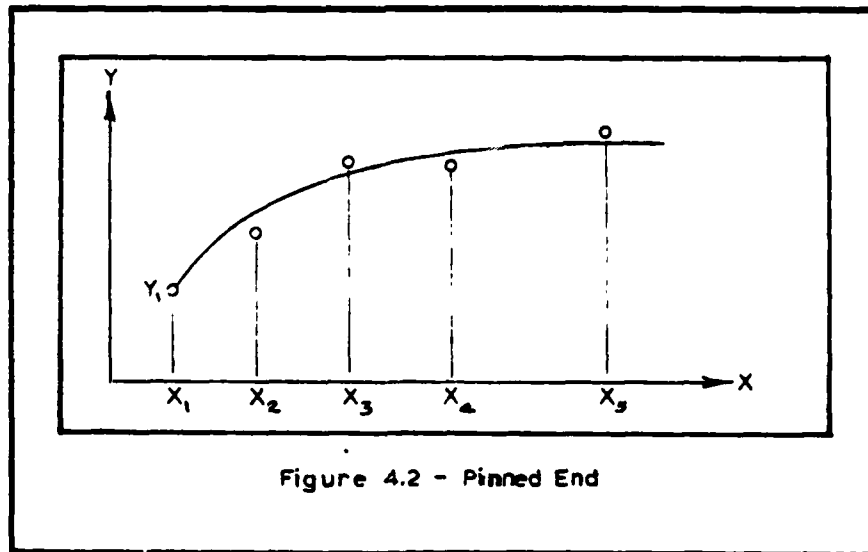


Figure 4.2 - Pinned End

In fitting the curve of equation (4.1) to the five data points in figure 4.2 above, we require that $P(x_1) = y_1$ exactly. If in our calculations we adjust the abscissas such that $x_1 = 0$ we may simplify this equation to:

$$P(x) = y_1 + a_1x + a_2x^2 + a_3x^3 \quad (4.6)$$

Applying our least squares criteria to this we obtain the following normal equations:

$$\begin{bmatrix} s_2 & s_3 & s_4 \\ s_3 & s_4 & s_5 \\ s_4 & s_5 & s_6 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix} \quad (4.7)$$

where

$$s_k = \sum_{i=2}^5 x_i'^k \quad (4.8)$$

$$t_k = \sum_{i=2}^5 (y_i - y_1) x_i'^k \quad (4.9)$$

and

$$x_i = x_i - x_1$$

With the values of a_1 , a_2 and a_3 computed from equation (4.7) we can calculate the faired position of points P_2 and P_3 :

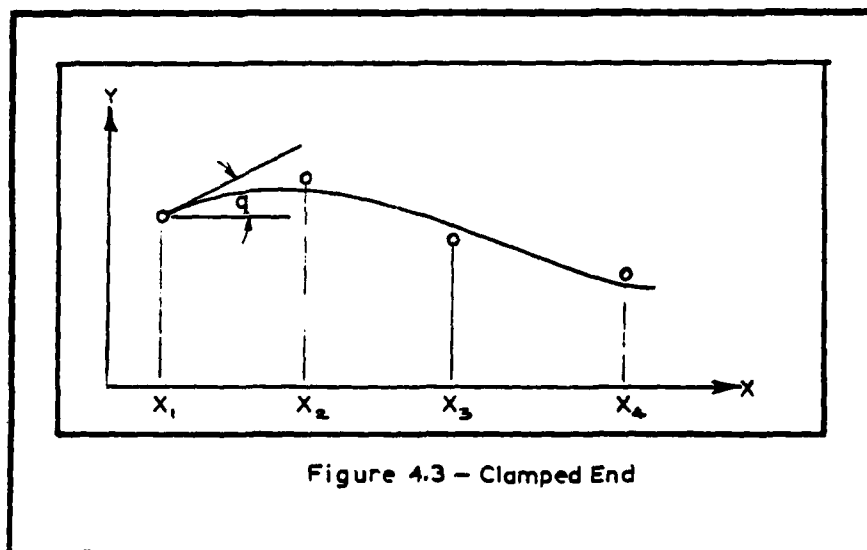
$$\bar{y}_2 = P(x'_2), \bar{y}_3 = P(x'_3)$$

where

$$x'_2 = x_2 - x_1, x'_3 = x_3 - x_1$$

After these two points are determined STRIP1 can be applied to continue the fairing process

4.2.2.3 STRIP3: Fairing an interval with clamped ends.



In order to fair an interval with both position and slope of the first point fixed we will consider only the first four points as shown in figure 4.3 above. In order to simplify the derivation we once again adjust the abscissas such that $x_1 = 0$. For this case equation (4.1) becomes:

$$P(x) = y_1 + qx + a_2x^2 + a_3x^3 \quad (4.10)$$

After applying the least squares fit criteria to the four data points the normal equations obtained are:

$$\begin{bmatrix} s_4 & s_5 \\ s_5 & s_6 \end{bmatrix} \begin{bmatrix} a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} t_2 \\ t_3 \end{bmatrix} \quad (4.11)$$

where

$$s_k = \sum_{i=2}^4 x_i'^k \quad (4.12)$$

$$t_k = \sum_{i=2}^4 (y_i - qx_i' - y_2) x_i'^k \quad (4.13)$$

and

$$x_i' = x_i - x_1$$

For the simple 2 X 2 system above a_2 and a_3 may be written as follows:

$$a_2 = \frac{t_2 s_6 - t_3 s_5}{s_4 s_6 - s_5^2} \quad (4.14)$$

$$a_3 = \frac{t_3 s_4 - t_2 s_5}{s_4 s_6 - s_5^2} \quad (4.15)$$

where

$$\Delta = \begin{vmatrix} s_4 & s_5 \\ s_5 & s_6 \end{vmatrix} \neq 0$$

Using the values of a_2 and a_3 calculated the faired position of the second point may be readily determined as:

$$\bar{y}_2 = P(x_2')$$

where

$$x_2' = x_2 - x_1$$

After fairing the second point STRIP1 can be applied to continue the fairing process, fairing point three and four the first time it is applied.

4.2.2.4 TRANS1: Fairing the last points in a given sequence of data.

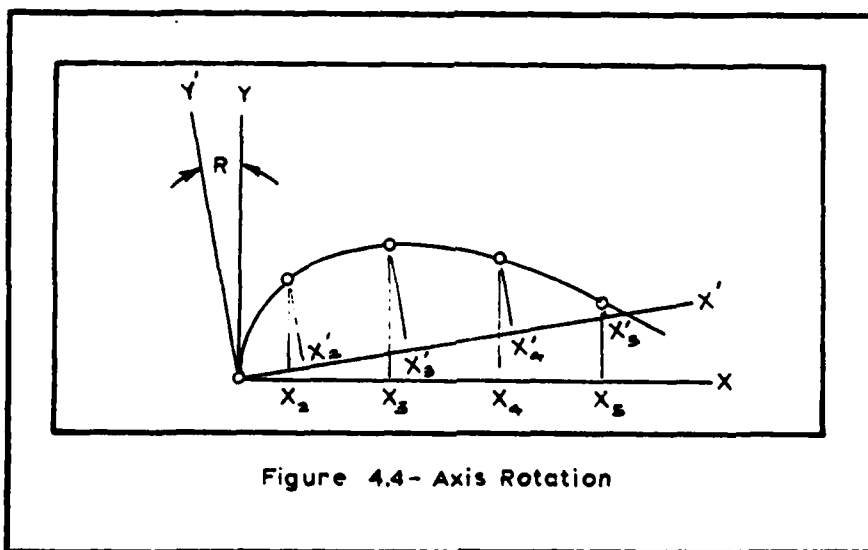
As can be seen in the previous sections, the fairing procedures operate on a series of points with monotonically increasing abscissa and end conditions specified at x_1 ; where $x_1 < x_2 < x_3 \dots$. At the time when the five point fairing interval reaches the other end of the curve, i.e., $P_k = P_{n-4}$, the following transformation must take place:

$$\begin{array}{ll}
 x'_1 = x_n - x_n = 0 & y'_1 = y_n \\
 x'_2 = x_n - x_{n-1} & y'_2 = y_{n-1} \\
 x'_3 = x_n - x_{n-2} & y'_3 = y_{n-2} \\
 x'_4 = x_n - x_{n-3} & y'_4 = y_{n-3} \\
 x'_5 = x_n - x_{n-4} & y'_5 = y_{n-4}
 \end{array} \tag{4.16}$$

This allows the fairing of the last three points as if they were the first three points. Once these three points are faired the reverse transformation is employed to place y'_1 into y_n , etc.

4.2.3 Fairing of Curves with Infinite Slopes

In ship design it is not infrequent that lines are encountered which possess infinite slopes at one or both end points. Such is the case of a section through the bow of a ship equipped with a bulbous bow. Here if the offsets y are expressed as a function of z , an infinite slope will occur at the bottom of the bulb. While the parametric rotating spline of section 3.2.4 will accommodate such a form, the simple cubic polynomial of equation (4.1) will prove indeterminate for an interval containing an end point with infinite slope.



Referring to figure 4.4 above, it can be seen that if the axis are rotated by some small angle θ , and the data points are redefined in this new coordinate system the fairing process may be carried out as normal. The following equations relate the coordinates in the two coordinate systems.

$$\left. \begin{aligned} x' &= x \cos\theta + y \sin\theta \\ y' &= -x \sin\theta + y \cos\theta \end{aligned} \right\} \quad (4.17)$$

$$\left. \begin{aligned} x &= x' \cos\theta - y' \sin\theta \\ y &= x' \sin\theta + y' \cos\theta \end{aligned} \right\} \quad (4.18)$$

It is also assumed that for small values of θ , e.g., 10° :

$$\Delta y \approx \Delta y', \quad \Delta x \approx \Delta x' = 0 \quad (4.19)$$

In order to continue the fairing process the faired position of the first three points and the slope at the third point could be calculated in the rotated coordinate

system and then transformed back into the unrotated coordinate system. The faired position and slope of the third point could be used to continue the fairing in the unrotated plane. The following slope transformation is also helpful.

$$\frac{dy}{dx} = \tan \left\{ \arctan \left(\frac{dy'}{dx'} \right) + \theta \right\} \quad (4.20)$$

4.2.3.1 Other transformations.

There are any number of transformations one could use to accommodate the problem of the infinite slope. One tried by this author was that of letting [18]:

$$\left. \begin{array}{l} x = \frac{1}{2}(1 - \cos \theta) \\ \text{or} \quad \theta = \arccos (1 - 2x) \end{array} \right\} \quad 0 \leq x \leq 1 \quad (4.21)$$

The curve is then plotted as a cubic in θ . This has the advantage of eliminating an infinite slope at $x = 0$; in fact $dy/d\theta \approx \sqrt{r/2}$, where r is the radius of curvature of $y = f(x)$ at $x = 0$. The disadvantage, as seen by this author, is that fairing will take place in the distorted y, θ plane. Additionally, even though the resulting curves appear to be aesthetically pleasing, some apprehension exists regarding the use

of lines faired in the two coordinate planes. For this reason the author opted for fairing in a rotated coordinate system as opposed to one which was distorted.

5. Computer Algorithms

5.1 Overview

The system of subroutines developed in this thesis were designed to provide two distinct capabilities: (1) to provide a means of fairing a series of data points not previously considered fair, and (2) to provide the capability of representing a series of data points by an analytical mathematical expression. This second feature would also provide a means by which slopes, curvatures, etc. could be determined by interpolation. The theory of these two procedures was developed in chapters four and three respectively.

The ultimate objective of these subroutines would be their utilization in a program to fair and draw an entire ship form. Because of this and the virtually infinite nature of the lines existing in a hull form, the program has to be capable of handling many line types. As an example of this, see figure 5.1, the programs require the following information as input.

1. Independent variable coordinates.
2. Dependent variable coordinates.
3. Data point type.
4. Number of data points.

5. Type of end conditions.
6. End slopes if required.
7. An indication as to whether the input data is fair as submitted.

The only other pieces of information required for fairing are:

1. TOL: Is a tolerance representing a limiting distance which any data point may be moved in the fairing process.
2. ACC: This number represents an accuracy which, if during the fairing process a point is not moved by more than this amount, it is considered to be in a faired position.
3. LIMIT: This number sets a limit on the number of iterative cycles permitted in the fairing process.

5.1.1 Specification of Point type

The designation of point type is designed to be as consistant as possible with reference [19].

<u>POINT TYPE</u>	:	<u>DEFINITION</u>
0	:	Normal point at the beginning or in the interior of a continuous curved line segment.
1	:	Break point at the end of a continuous curved segment. At present this point is treated as if it were pinned. The slope, while unspecified, is discontinuous.

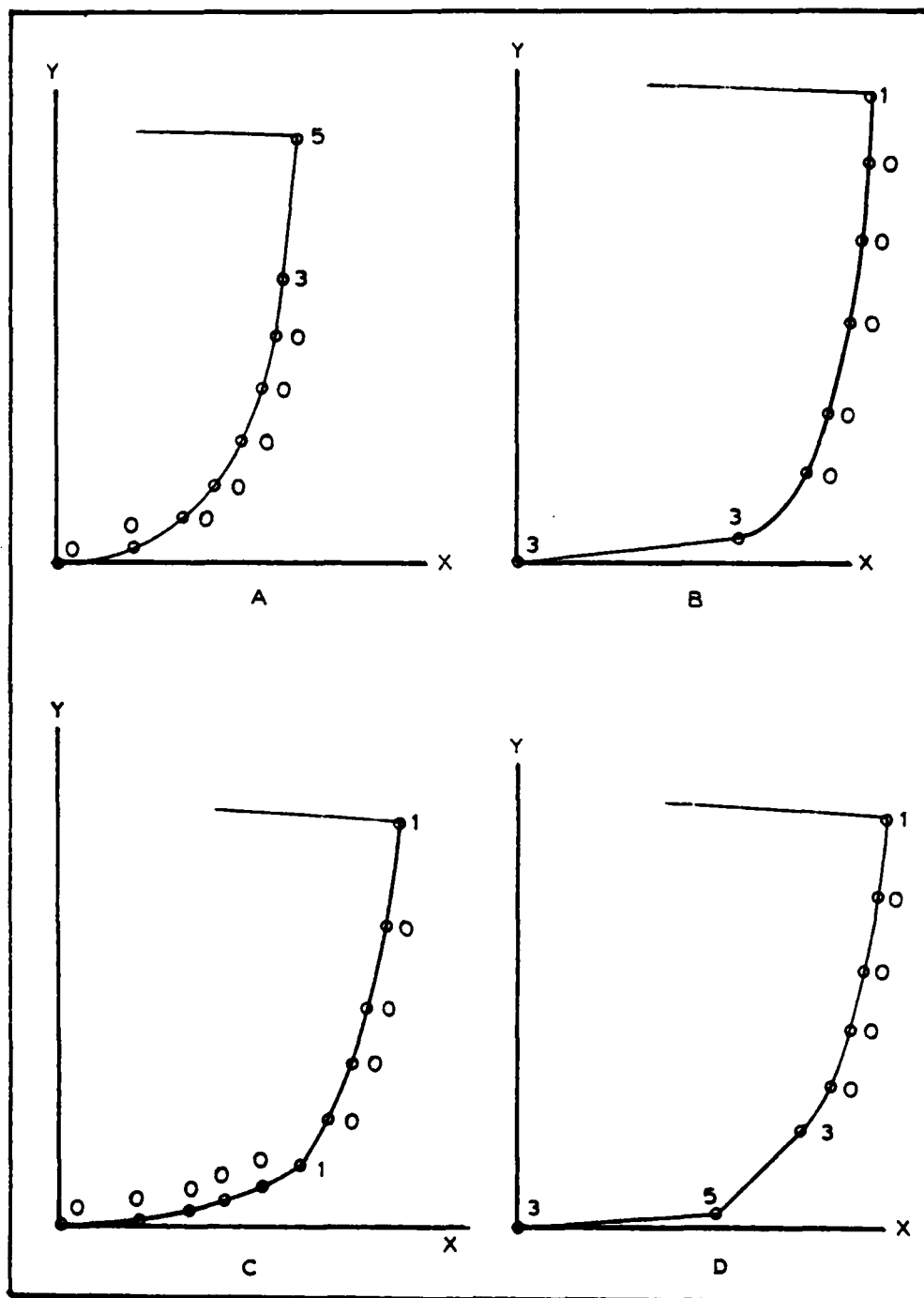


Figure 5.1- Point Type Examples

- 3 : This point can be at the beginning, middle or end of a straight line segment. The slope is continuous at this point. This point must be specified where a curved segment joins a straight line segment since the point is considered to be a clamped end condition for the curved segment.
- 5 : Break point at the end of a straight line. The slope is discontinuous at this point.

5.1.2 Specification of End Conditions

The end condition designation is made with a two digit real number of the following format, "B.E". Here B corresponds to the end conditions at the beginning of the line and E the end condition at the end of the line.

<u>END TYPE</u>	:	<u>DEFINITION</u>
1	:	Free end. The end point is free to both rotate and translate.
2	:	Pinned end. The end point is free to rotate only, the position is fixed.
3	:	Clamped end. The end is totally constrained. It is free to <u>neither</u> rotate or translate. The slope must also be defined.
4	:	Clamped end with infinite slope. The slope, whether $\pm \infty$ is determined by the second data point, i.e., $+\infty$ if $y_2 > y_1$ or $-\infty$ if $y_2 < y_1$.

5.1.3 Storage of Pertinent Line Data

The information necessary to fully describe any line is stored in a 32 x 7 two-dimensional array. This array is labeled CRV in the subroutines and its elements have the following significance. At present the first thirty rows are for data point or interval information and the last two rows are for overall curve characteristics. This could be easily expanded to allow more input data.

1. Column 1, CRV(1,1) to CRV(30,1):
Abscissa of the input data points.
2. Column 2, CRV(1,2) to CRV(30,2):
Ordinates of the original data points.
3. Column 3, CRV(1,3) to CRV(30,3):
The faired ordinates of the data points.
4. Column 4, CRV(1,4) to CRV(30,4):
The point type, see section 5.1.1.
5. Column 5, CRV(1,5) to CRV(29,5):
The values of a_i as defined in Appendix D.
6. Column 6, CRV(1,6) to CRV(29,6):
The values of b_i as defined in Appendix D.
7. Column 7, CRV(1,7) to CRV(30,7):
The slope of the curve at the data points as defined in Appendix D.

It should be noted that the elements of columns 5, 6 and 7 are obtained as a result of the splinning option. The data in column 3 are obtained as a result of exercising the fairing option.

<u>ELEMENT</u>	:	<u>DEFINITION</u>
CRV(31,2)	:	The number of data points. At present, $6 \leq \text{CRV}(31,2) \leq 30$.
CRV(31,3)	:	The slope at the beginning of the curve. Left blank if not specified.
CRV(32,1)	:	An indication of the fairness of the curve. 1.=the data submitted is fair. 2.=the data submitted is not fair.
CRV(32,2)	:	End condition specification, see section 5.1.2.
CRV(32,3)	:	The slope at the end of the curve. Left blank if not specified.

The other elements of the CRV matrix are reserved for future use, e.g., in a full ship fairing program.

There are three other aspects of the program which are of the utmost importance. As was seen in the development of chapter three, the mathematical curve representation, or splinning procedure is fully capable of accommodating multi-valued curves. The fairing option, however, requires that a curve be single valued over the domain of the independent variable. Therefore, while it is possible to fit a cubic curve to virtually any series of data points, care must be observed when exercising the fairing option. The second inviolable characteristic of the program is that the data points must be submitted in a monotonically sequential fashion, i.e., the points must not be submitted in a random fashion,

but rather as they are encountered while following the path of the curve. The last consideration is that in order to fair any curved line segment there must be at least six data points in the continuous curved region. This is true regardless of the end conditions of the line as a whole or the end conditions for a line segment.

5.2 Description of Subroutines

A flow chart and subroutine listing may be found in Appendix F.

5.2.1 Lines Pairing

The subroutines included in this section are utilized to calculate the faired position of the given data points, i.e., column 3 of the CRV matrix.

5.2.1.1 Subroutine PREFAR

This subroutine takes the data in the CRV matrix and loads all the points on a continuous curve segment into three linear arrays; X(), Y() and YORIG () representing the abscissa, faired ordinate (the original ordinate for the first iteration) and the original ordinate respectively. This process is governed by the value of point type, CRV(I,4). With these arrays established PREFAR calls either FARCRV or FARLIN, depending on whether the curve segment begins with an infinite slope.

Upon final return to this subroutine the values of the faired position of the data points will have been calculated and placed in column three of the CRV matrix.

5.2.1.2 Subroutine FARCRV

This subroutine takes the data in the X, Y, YORIG array, for those line segments which have infinite slopes, rotates the coordinate axis $10^\circ (\pi/18 \text{ radians})$ and then places the transformed points, equation (4.17), in an XPRIM, YPRIM and YOPRIM array. The subroutine then calls subroutine FARLIN to fair the first six data points in the rotated system. At this time subroutine SPLINE is called to determine the slope at the third point, also in the rotated system. The subroutine then completes fairing the remaining data points by matching the position and slope at the third point, in the unrotated system. That is, assuming point three to be clamped and beginning with STRIP3.

5.2.1.3 Subroutine FARLIN

This subroutine takes the points of a continuous curve segment and calls the various STRIP_ subroutines which actually compute the faired position of the points. FARLIN also calls subroutines FSTPTS and TRANS1 to fair the first and last points in the sequence.

5.2.1.4 Subroutine FSTPTS

This subroutine fairs the first three data points in a sequence of data points based on the end condition. STRIP1, 2 or 3 are called as appropriate.

5.2.1.5 Subroutine TRANS1

This subroutine fairs the last three data points based on the end condition specified. Specifically it transforms the abscissa in accordance with equation (4.16).

5.2.1.6 Subroutine STRIP1, 2 or 3

These subroutines are described in detail in sections 4.2.2.1 to 4.2.2.3. They use, as arguments, the variables in the X1, Y1 and Y0 arrays. Additionally they require values of TOL and ACC which place limits on the amount which a point may be moved and the amount of movement which is considered to be negligible. For the case where the point would move by more than TOL from its original (unfaired) position, its movement is limited by the value of TOL.

5.2.2 Lines Representation

The methodology of representing a line by a parametric cubic equation was developed in chapter three. The actual

sequence in which the process is executed is described in the following sections.

5.2.2.1 Subroutine PRESPL

This subroutine examines the input data in the CRV matrix and places elements of continuous curved line segments into the X, Y and YORIG arrays. This assignment is based on data point type found in column four of the CRV matrix. Referring to figure 5.1A, the program would load the first eight points into X, Y and YORIG. Point nine, point type 5, would be used in conjunction with point eight to determine the slope of the curved segment ending at point eight. The program then calls SPLINE to carry out the actual curve fitting algorithm.

5.2.2.2 Subroutine SPLINE

This subroutine uses the data in X, Y and YORIG obtained from PRESPL and carries out the curve fitting algorithm presented in Appendix D. Although many intermediate terms are calculated, the only terms which are retained are a_i , b_i and d_i , these quantities are subsequently used to calculate interpolated values of the independent variable and slope at a point specified by the user.

5.2.2.3 Subroutine INTERP

This subroutine determines the interval in which a desired value of a dependent variable is located. It then passes the coordinates of the surrounding points and the values of a_i and b_i for the interval to subroutine CALCY which calculates the value of the dependent variable and slope at the desired point.

5.2.2.4 Subroutine CALCY

This subroutine calls CALCT to obtain the value of the parametric variable T. With the value of T the interpolated value of the independent variable is determined. Since the dependent and independent variables are represented parametrically, the slope of the curve is calculated by the chain rule as follows:

$$\frac{dy}{dx} = \frac{dy}{dt} / \frac{dx}{dt} \quad (5.1)$$

Since the value of T is determined as being the root of a third degree polynomial, CALCY is designed to calculate the interpolated value of the independent variable and slope for up to three unique and real values of T. However, the subroutine is designed to print a warning that additional points are needed to specify the curve if T has more than one real value.

5.2.2.5 Subroutine CALCT

This subroutine calculates the roots of the third degree parametric polynomial in T using the algorithm in reference [20]. This procedure is also presented in Appendix E for the readers' convenience. It should be realized, however, that only the real roots are calculated in the subroutine, the imaginary roots lack physical significance for the purpose of lines plotting.

Appendix G contains an example of a data set that was first faired then splinned and then interpolated at points equal to one-twentieth of the domain of the independent variable. Once again it should be emphasized that, in order to fair a curved segment, at least six data points must be defined in that segment, including the end points of the segment.

6. Conclusions and Recommendations

6.1 Hull Form Modification

When work on this thesis began, the initial goal was to develop a series of destroyer-like hull forms for future use in seakeeping analysis. Preliminary efforts, using the method of longitudinally shifting sections [2,8], while showing promise, indicated that additional work would be required if the procedure was to apply accurately to destroyer type ships. Specifically, the method had to be adapted to ships whose maximum beam and section of maximum area did not lie at midships. These necessary changes were made successfully and the method was also extended to provide control over the ship's centerline profile. The primary motivation for this extension was to gain control over the hull form in the region of a sonar dome. While there was some apprehension about the criticality of changes to the geometry of the sonar dome, a telephone call to the Naval Sea Systems Command in Washington, D.C. [21], indicated that because of acoustic and hydrodynamic considerations the dome design should be maintained unchanged.

The resulting procedure for modifying hull forms does provide good results for that portion of the ship below the design waterline. However, as outlined in chapter two and

Appendix C, there are situations where the method does not provide exact results, e.g., when the station of maximum beam and section of maximum area do not coincide. Another unresolved weakness of the modification scheme is that it still does not provide the degree of control over specific hull regions often desired, e.g., an attempt to preserve the configuration of a sonar dome will result in preservation of the centerline profile only, the three dimensional geometry of the dome will be uncontrollably altered.

In summary it has been concluded that the modification technique holds a great deal of promise for use with automated methods. In particular, the procedure as it currently exists, will provide excellent results when dealing with ships for which there is no rigid requirement to keep a specific region fixed. Not only are the desired coefficients and characteristics obtained, the resulting hull forms appear to be acceptably fair.

As with virtually all work of this type there is still need for additional development. Specifically, it is felt that those aspects worthy of attention are:

1. Investigate a means of controlling the resulting hull form above the design waterline. At present, excessive flair or tumblehome frequently occurs.

AD-A086 640

NAVAL POSTGRADUATE SCHOOL MONTEREY CA
COMPUTER AIDED GEOMETRICAL VARIATION AND FAIRING OF SHIP HULL F--ETC(U)
MAY 78 F R HABERLANDT

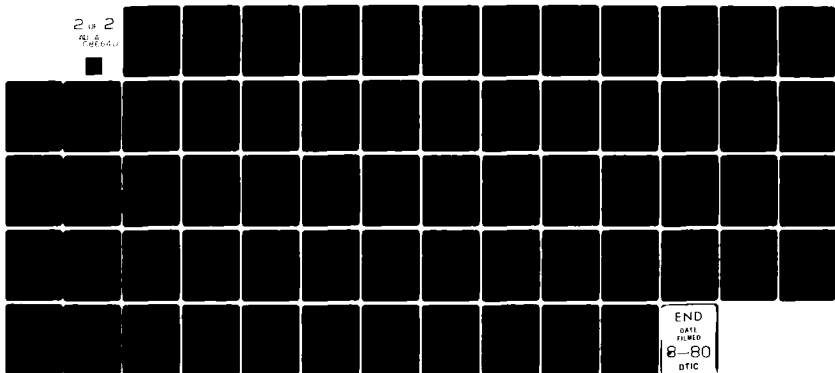
F/G 13/10

UNCLASSIFIED

ML

2 of 2

AD-A086 640



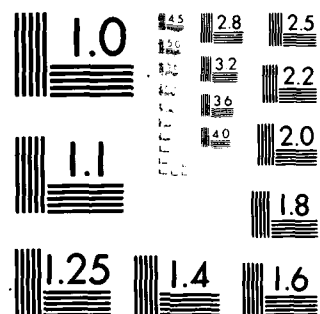
END

DATE

FILED

8-80

DTIC



MICROCOPY RESOLUTION TEST CHART

NATIONAL BUREAU OF STANDARDS-1963-A

2. Investigate a means of rigidly controlling the geometry of a specific region of the hull. This would provide a solution to the problem of keeping the sonar dome unaltered.
3. Develop a computer program to carry out the extensive mathematical and graphical calculations required by the method.

6.2 Mathematical Representation of Lines and Fairing

6.2.1 Lines Representation

In chapter three it was demonstrated, by variational calculus and by simple beam theory, that a third degree or cubic polynomial could be used to approximate the shape taken by the draftsman's spline. However, it was also pointed out that the simple cubic polynomial became indeterminate if the curves contained infinite slopes. For this reason, and also because they are capable of representing multivalued functions, the parametric cubic equations of reference [15] were incorporated into this thesis. The results obtained using this method have proven to be excellent. Not only does the technique lend itself readily to being programmed, the parametric form of the curve allows the user to define either variable as being the independent variable for the purposes of interpolation. The benefit of this capability will become apparent in the discussion of cross fairing.

The only disadvantage, as seen by this author, to using the parametric equations is that they require the user to calculate the, up to three real roots, of the polynomials each time an interpolated value is sought. However, this is not seen as being restrictive since a closed form solution exists for calculating these roots and is in fact utilized in

subroutine CALCT. Therefore, because of its great flexibility, the parametric, or rotating spline technique of chapter three, is highly recommended for use with a lines fairing scheme involving the manipulation of specific waterlines and sections.

6.2.2 Fairing

The least-squares fairing criteria, as presented in chapter four, has shown to provide an effective means of altering the position of data points in order to obtain the desired "fairing" effect. That is, if provided with an adequate tolerance interval, the cubic spline passed through the resulting points will be void of extraneous oscillations and generally pleasing to the eye. When addressing the lines of a ship in the preliminary design phase, the fact that the lines satisfy a visual inspection is likely to be sufficient. For this reason, and also the excellent results obtained by this method in reference [16], this author has concluded that this scheme would be a candidate for a complete lines fairing program for destroyer-type ships.

6.2.3 Recommendations

It is obvious that, given the capability of representing lines mathematically and also a means to fair the points on a line, the next step would be to generate a method

which would fair, in the three-dimensional sense, and display an entire ship. This author has spent a great deal of time attempting to extend the methods of reference [16] in a more general form to accommodate the peculiarities which arise in addressing displacement-type ships. The difficulty arose from two sources. First, an attempt was made to treat the entire ship, i.e., the bow and stern were not truncated as was the case with other methods examined. Second, in trying to treat a large variety of ships, conveniently called displacement-type, the author was confronted with the problem of attempting to describe the myriad of lines of discontinuity which one may encounter. These lines are most frequently termed control lines and may consist of the ship's profile, in an obvious sense, to the locust of points, longitudinally, where rise of floor and bilge radius meet, in a more subtle sense. Figure 6.1, for a typical bulbous bow destroyer illustrates a few of the possibilities.

If we were to ignore the fairing algorithm itself for a moment, it can be seen that if a waterline A-A is taken in figure 6.1A there must be some means of communicating the effect of control line #6 on the waterline; where the explanation of the control lines is contained in table 6.1. For this case, the effect is to create a straight line region in A-A as projected in figure 6.1B. A tentative solution to this

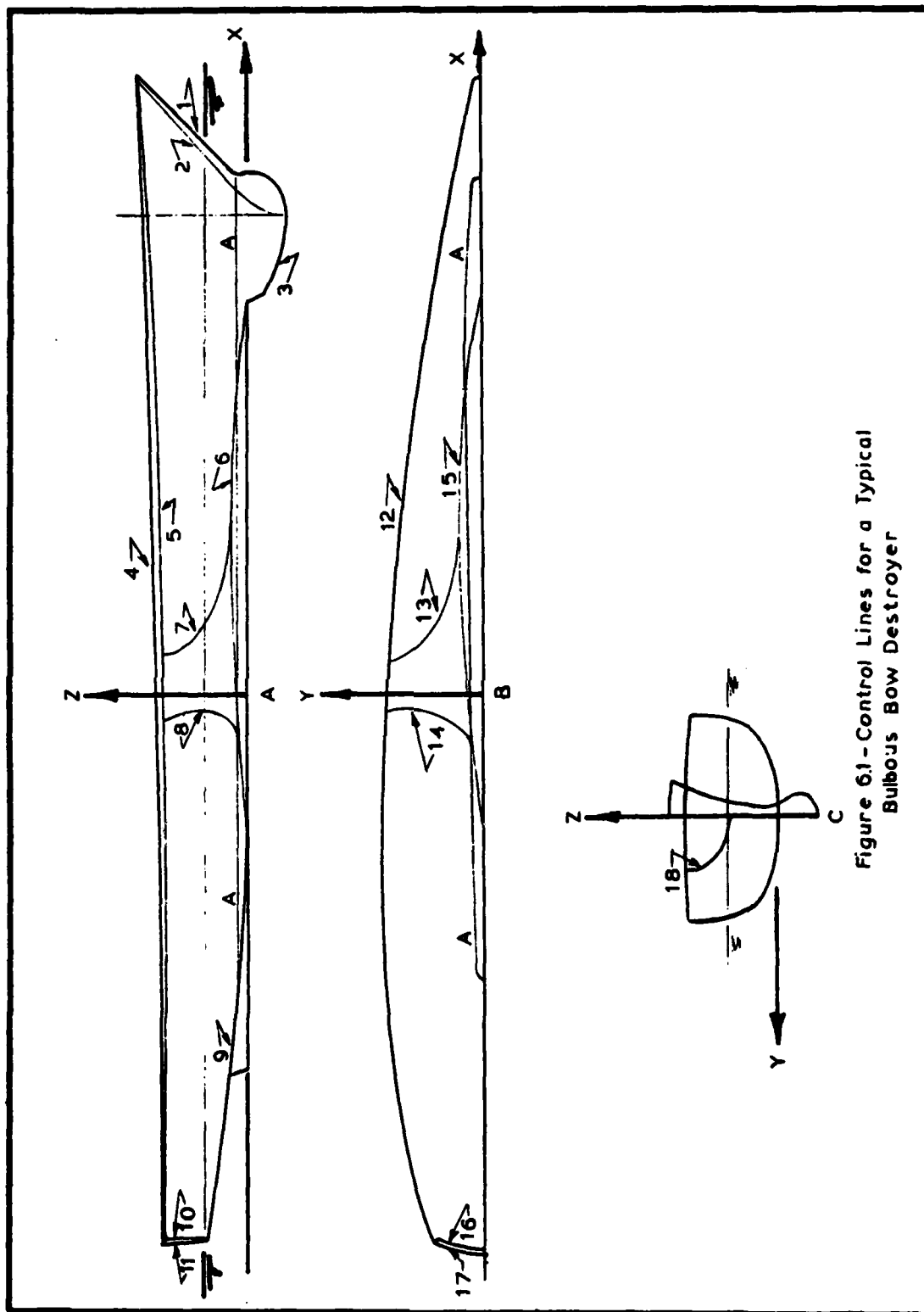


Figure 6.1 - Control Lines for a Typical Bulbous Bow Destroyer

TABLE 6.1
Explanation of Control Lines

1. Bow profile
2. Locus of stern radius centers
3. Sonar dome profile
4. Main deck centerline profile
5. Deck edge profile
6. Extent of deadrise
7. Forward extent of parallel middlebody
8. After extent of parallel middlebody
9. Keelrise aft
10. Outboard transom profile
11. Transom centerline profile
12. Deck edge waterline
13. Forward extent of parallel middlebody
14. After extent of parallel middlebody
15. Extent of deadrise
16. Outboard transom profile
17. Deckedge transom profile
18. Section view of transom

problem would be to ascribe to each control line, over a region where applicable, a code designating the effect of the control line on waterlines or sections at the point of intersection. This could be easily done by assigning another column to the CRV matrix description of the line, see section 5.1.3.

Another complication which must be resolved is: when attempting to establish the offsets for, say an arbitrary waterline, how do you seek out where this waterline intersects which control lines. In the most general case, where control lines could occur at random through a hull form this problem could prove to be formidable at least. As seen by this author, the only solution to this problem is to have only certain control lines admissible for a particular class of ship. This would necessarily limit the possible intersection combinations. The control lines shown in figure 6.1 represent, what this author feels, are typical of a contemporary destroyer.

The final aspect to be addressed is that of the cross fairing algorithm itself. Reference [16] showed that by utilizing a preassigned grid in the X-Z plane the offsets (y-coordinates) at these points could be repeatedly be faired and splined by both lines of section and waterlines. The new, or faired value of each point was taken to be the mean of that obtained by fairing the two lines. These mean values were then used as unfair data points on the lines once again

and the fairing process was repeated. This iterative procedure was continued until the movement of the points on successive iterations was less than some predefined limit. The results of this cross fairing algorithm [16] proved to be quite good. Because of this, it is felt that this procedure would also prove satisfactory for the more general method of lines fairing and representation presented in this thesis.

As a final note, this author can envision where the two independent aspects of this thesis could be combined into one program of significant value. If both the fairing procedure and lines modification techniques were automated, it would provide the designer with the capability to sketch out a rough design on the back of an envelope, specifying its fundamental coefficients and dimensions, and then by passing this information through the fairing and modification routines a faired form could be obtained. The implications of this, as a savings of time and resources, are quite astounding. If the method were further extended to permit an interactive modification of the design, an individual could literally sit down and design a faired vessel in a matter of hours instead of days.

REFERENCES

1. Loukakis, T., Chryssostomidis, C., "Seakeeping Standard Series for Cruiser-Stern Ships", Transactions, Society of Naval Architects and Marine Engineers, Vol. 83, 1975, pp. 67-127.
2. Lackenby, H., "On the Systematic Geometrical Variation of Ship Forms", Transactions, Royal Institute of Naval Architecture, Vol. 92, 1950, pp. 289-315.
3. Chryssostomidis, C., "Computer Aided Ship Design", Paper, New England Section Society of Naval Architects and Marine Engineers, May 1978.
4. Comstock, J.P., "Principles of Naval Architecture", Revised, The Society of Naval Architects and Marine Engineers, 1967, New York, New York.
5. Coons, S.A., "Surfaces for Computer-Aided Design of Space Figures", M.I.T., ESL Memorandum 9442-M-139, July 1965.
6. Gertler, M., "A Reanalysis of the Original Test Data for the Taylor Standard Series", TMB Report 806, March 1954.
7. Söding, H. and Rabein, U., "Hull Surface Design by Modifying an Existing Hull", Paper presented at the First International Symposium on Computer Aided Hull-Surface Definition, Annapolis, MD., September 1977.
8. Moor, D.I., "Effects on Performance in Still Water and Waves of Some Geometric Changes to the Form of a Large Twin-Screw Ship", Transactions, Society of Naval Architects and Marine Engineers, Vol. 78, 1970, pp. 88-150.

9. Mehlum, E., "Variational Criteria for Smoothness", Paper for Central Institute for Industrial Research, Oslo, Norway, December, 1969.
10. Hildebrand, F.B., "Advanced Calculus for Applications", Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1962.
11. Higdon, A., Ohlsen, E.H., Stiles, W.B., Weese, J.A., "Mechanics of Materials", Second Edition, John Wiley and Sons, Inc., New York, July, 1968.
12. Thomas, G.B., "Calculus and Analytic Geometry", Fourth-Edition, Addison-Wesley Publishing Co., Reading, Massachusetts, June, 1972.
13. Yeung, R.W., Class Notes, MIT Ocean Engineering Department Course 13.50, Spring, 1977.
14. Carnahan, B., Wilkes, J.E., "Digital Computing and Numerical Methods", John Wiley and Sons, Inc., New York, 1973.
15. Söding, H., "Numerical Ship Lofting and Hull Form Design", Unpublished paper, Circa 1962.
16. Kyrkos, B., "The Fairing and Mathematical Representation of the Surface of a Ship Using a Small Computer", (Greek) Translations from National Technical University of Athens, Diploma Thesis, 1976.
17. Corin, T., "Recent Developments in Ship Lines Fairing at the David Taylor Model Basin", DTMB Report, Applied Mathematics Laboratory.
18. Kerwin, J., "Fitting Curves with Infinite Slopes at $x=0$ ", Unpublished paper used for Course 13.50, Spring, 1977.

19. Viega, J.P.C., "Hydrostatic Considerations in the Design of Ships with Unusual Shapes", Massachusetts Institute of Technology, Department of Ocean Engineering, Thesis, January, 1975.
20. Baumeister, T. and Marks, L.S., "Standard Handbook for Mechanical Engineers", Seventh Edition, McGraw-Hill Book Company, New York, 1967.
21. Silverstein, S., Telephone Conversation, Shipboard Sonar Group, Naval Sea Systems Command, Washington, D.C., Circa, November, 1977.

APPENDIX A

Calculation of coefficient c and centroid of the sliver of added area. Referring to figure 2.4.

Recall: $\delta x = cx(1-x)$

$$\begin{aligned}\delta\phi &= \int_0^1 \delta x \, dy = c \int_0^1 x(1-x) \, dy \\ &= c \left\{ \int_0^1 x \, dy - \int_0^1 x^2 \, dy \right\} = c[\phi - 2\phi\bar{x}]\end{aligned}$$

$$c = \frac{\delta\phi}{\phi - 2\phi\bar{x}} = \frac{\delta\phi}{\phi(1-2\bar{x})}$$

$$\delta x = \frac{\delta\phi}{\phi(1-2\bar{x})} x(1-x) \quad (\text{A.1})$$

solving for centroid, h

$$\begin{aligned}\delta\phi \cdot h &= \int_0^1 \delta x \left(x + \frac{\delta x}{2}\right) dy \\ &= \int_0^1 x\delta x \, dy + \frac{1}{2} \int_0^1 \delta x^2 \, dy\end{aligned}$$

substituting for δx

$$\begin{aligned}\delta\phi \cdot h &= \frac{\delta\phi}{\phi(1-2\bar{x})} \int_0^1 (x^2 - x^3) \, dy + \frac{\delta x^2}{2(1-2\bar{x})^2} \int_0^1 (x^2 - 2x^3 + x^4) \, dy \\ h &= \frac{2\bar{x} - 3k^2}{1 - 2\bar{x}} + \frac{\delta\phi}{\phi} \left\{ \frac{\bar{x} - 3k^2 + 2x^3}{(1 - 2\bar{x})^2} \right\} \quad (\text{A.2})\end{aligned}$$

For $\delta\phi \ll \phi$

$$h \sim \frac{2\bar{x} - 3k^2}{1-2\bar{x}}$$

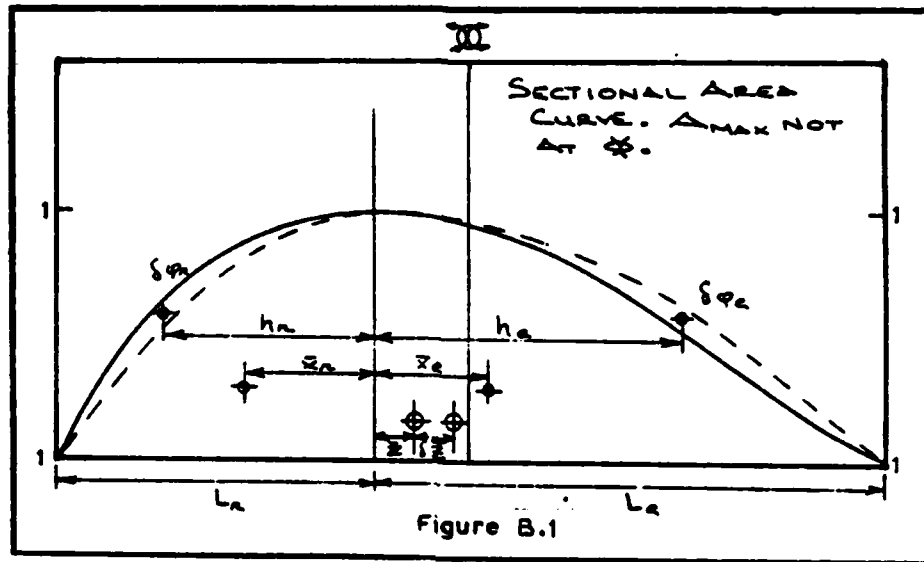
(A.3)

where:

$$k^2 = \frac{1}{3\phi} \int_0^1 x^3 dy$$

$$r^3 = \frac{1}{4\phi} \int_0^1 x^4 dy$$

APPENDIX B



Referring to figure B.1 above, all longitudinal dimensions are measured with respect to the point of maximum sectional area. Assume:

$$\begin{aligned}\delta x &= cx(1-x) \\ &= \frac{\delta \phi}{\phi(1-2\bar{x})} x(1-x)\end{aligned}$$

For the new hull form

$$\phi'_t = \frac{1}{L} \{L_e(\phi_e + \delta\phi_e) + L_r(\phi_r + \delta\phi_r)\} \quad (B.1)$$

$$\bar{z}' = \frac{1}{L^2 \phi_t'} \{ L_e^2 (\phi_e \bar{x}_e + \delta \phi_e h_e) - L_r^2 (\phi_r \bar{x}_r + \delta \phi_r h_r) \} \quad (B.2)$$

Also for the original hull form

$$\phi_t = \frac{1}{L} \{ L_e \phi_e + L_r \phi_r \} \quad (B.3)$$

We now have to apply equations (B.1) and (B.2) to obtain values of $\delta \phi_e$ and $\delta \phi_r$ in terms of the known quantities $\bar{z}$ and ϕ_t' . These quantities representing the desired values LCB and C_p for the derived form.

Solving equation (B.1) for $\delta \phi_e$.

$$\begin{aligned} \phi_t' L &= L_e \phi_e + L_e \delta \phi_e + L_r (\phi_r + \delta \phi_r) \\ \delta \phi_e &= \frac{1}{L_e} \{ L \phi_t' - L_e \phi_e - L_r (\phi_r + \delta \phi_r) \} \end{aligned} \quad (B.4)$$

Substituting equation (B.4) into equation (B.2) and solving for $\delta \phi_r$.

$$\bar{z}' L^2 \phi_t' = L_e^2 (\phi_e \bar{x}_e + \delta \phi_e h_e) - L_r^2 (\phi_r \bar{x}_r + \delta \phi_r h_r)$$

expanding the right hand side, R.H.S.

$$\text{L.H.S.} = L_e^2 \phi_e \bar{x}_e + L_e^2 \delta \phi_e h_e - L_r^2 \phi_r \bar{x}_r - L_r^2 \delta \phi_r h_r$$

rearranging terms

$$L_r^2 \delta \phi_r h_r - L_e^2 \delta \phi_e h_e = L_e^2 \phi_e \bar{x}_e - L_r^2 \phi_r \bar{x}_r - \bar{z}' L^2 \phi_t'$$

substituting in L.H.S.

$$L_r^2 \delta \phi_r h_r - L_e^2 h_e \frac{1}{L_e} \{L \phi_t' - L_e \phi_e - L_r (\phi_r + \delta \phi_r)\} = \text{R.H.S.}$$

$$L_r^2 \delta \phi_r h_r - L_e h_e L \phi_t' + L_e^2 h_e \phi_e + L_r L_e h_e \phi_r + L_r L_e h_e \delta \phi_r =$$

$$\delta \phi_r (L_r^2 h_r + L_r L_e h_e) + L_e (L_e h_e \phi_e + L_r h_e \phi_r - h_e L \phi_t') =$$

$$\delta \phi_r = \frac{1}{L_r^2 h_r + L_r L_e h_e} \{L_e^2 \phi_e \bar{x}_e + L_r^2 \phi_r \bar{x}_r - \bar{z}' L^2 \phi_t' - L_e h_e (L_e \phi_e + L_r \phi_r - L \phi_t')\} \quad (\text{B.5})$$

Equation (B.5) may be substituted back into equation (B.4) to obtain a value for $\delta \phi_e$.

However, if the simplified form for h is not used

$$h = f(\delta\phi_r, \delta\phi_e)$$

Derivation of $\delta\phi_e$ from equation (1)

$$\delta\phi_r = \frac{1}{L_r} \{L\phi'_t - L_e\phi_e - L_e\delta\phi_e - L_r\phi_r\} \quad (B.6)$$

substituting into equation (B.2) and rearranging

$$\begin{aligned} \bar{z}'L^2\phi'_t - L_e^2\phi_e\bar{x}_e + L_r^2\phi_r\bar{x}_r &= \text{R.H.S.} \\ &= L_e^2h_e\delta\phi_e - L_rh_r\{L\phi'_t - L_e\phi_e - L_e\delta\phi_e - L_r\phi_r\} \\ &= \delta\phi_e\{L_e^2h_e + L_rL_eh_r\} - L_rh_r\{L\phi'_t - L_e\phi_e - L_r\phi_r\} \\ \delta\phi_e &= \frac{1}{L_e^2h_e + L_rL_eh_r} \{L_r^2\phi_r\bar{x}_r - L_e\phi_e\bar{x}_e + \bar{z}'L^2\phi'_t - L_rh_r(L_e\phi_e + \\ &\quad L_r\phi_r - L\phi'_t)\} \quad (B.7) \end{aligned}$$

The form of this equation is merely the transposition of subscripts by $r \rightarrow e \rightarrow r$ of equation (B.5) for $\delta\phi_r$.

APPENDIX C

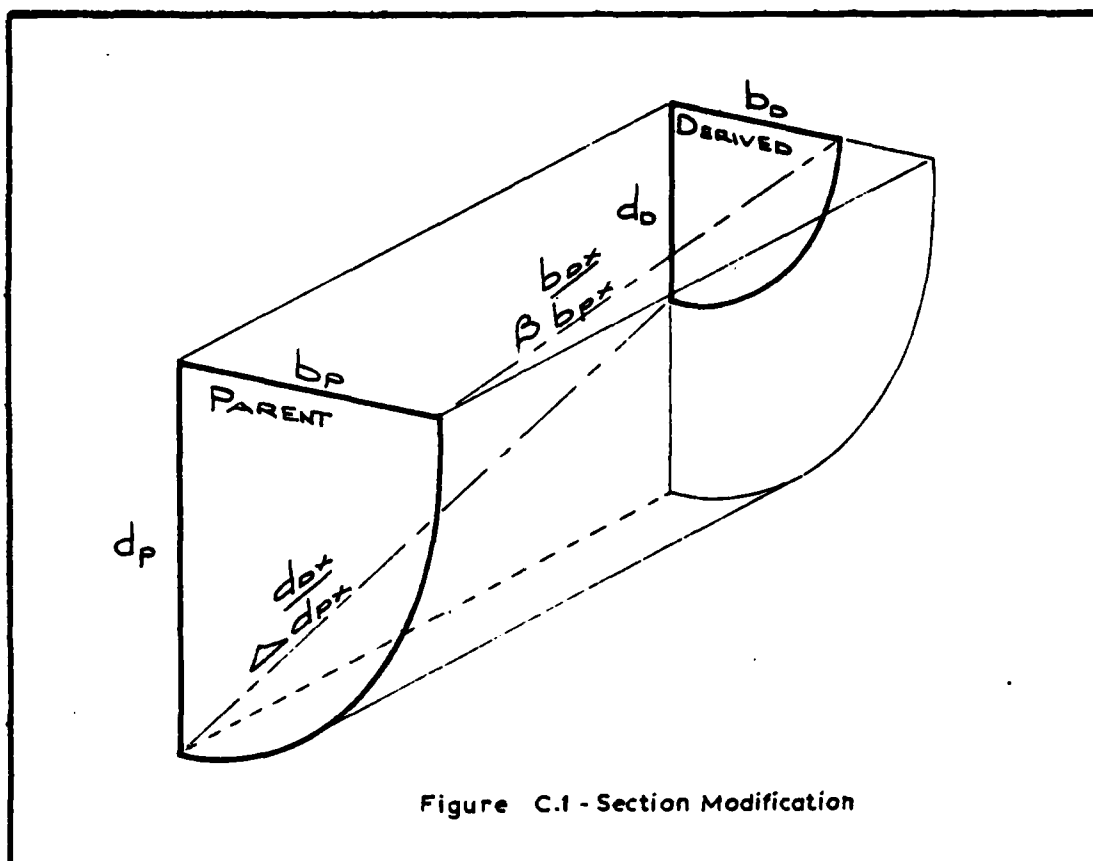


Figure C.1 - Section Modification

Definitions:

a = area of section

b = beam of section

d = draft of section

Subscripts:

p = parent hull form

d = derived hull form

x = a maximum value, e.g., b_{dx} is the maximum value of the beam in the derived hull form

Superscript:

bar, (—): refers to the sections in the parent and derived form which interact to create the derived section of maximum area.

star, (*): refers to the sections which are used to create the section of maximum beam in the derived form.

Further define the following ratios:

$$A = \frac{a}{a_x}, \quad B = \frac{b}{b_x}, \quad D = \frac{d}{d_x}$$

$$\alpha = \frac{A_d}{A_p}, \quad \beta = \frac{B_d}{B_p}, \quad \Delta = \frac{D_d}{D_p}$$

also

$$\frac{A}{B \cdot D} = R$$

It is by selecting a section in the parent, whose value of R is equal to that of the derived section being sought, that the new sections are created. It should also be evident by referring to figure C.1, that the area of the derived section will be as follows:

$$a_D = a_p \beta \frac{b_{dx}}{b_{px}} \Delta \frac{d_{dx}}{d_{px}}$$

It remains to be shown that in some cases the resulting area ratio is not always what is desired.

The following expression will also be useful.

$$a_{dx} = \bar{a}_p \bar{\beta} \frac{b_{dx}}{b_{px}} \bar{\Delta} \frac{d_{dx}}{d_{px}}$$

For any derived section, the area ratio obtained is:

$$\begin{aligned} \frac{a_d}{a_{dx}} &= a_p \beta \frac{b_{dx}}{b_{px}} \Delta \frac{d_{dx}}{d_{px}} \left\{ \frac{1}{\bar{a}_p \bar{\beta} \frac{b_{dx}}{b_{px}} \bar{\Delta} \frac{d_{dx}}{d_{px}}} \right\} \\ &= \frac{a_p}{\bar{a}_p} \frac{\beta}{\bar{\beta}} \frac{\Delta}{\bar{\Delta}} \end{aligned} \quad (C.1)$$

However, the area ratio desired is:

$$R_{pB_dD_d} = \frac{\frac{a_p}{a_{px}}}{\frac{b_p}{b_{px}} \cdot \frac{d_p}{d_{px}}} \left\{ \frac{b_d}{b_{dx}} \frac{d_d}{d_{dx}} \right\} = \frac{a_p}{a_{px}} \beta \Delta \quad (C.2)$$

Therefore, the ratio of A_d desired to A_d obtained is:

$$\begin{aligned} \frac{\frac{a_p}{a_{px}} \beta \Delta}{\frac{a_p}{\bar{a}_p} \frac{\beta}{\bar{\beta}} \frac{\Delta}{\bar{\Delta}}} &= \frac{\frac{a_p}{a_{px}} \beta \Delta}{\frac{a_p}{\bar{a}_p} \beta \bar{\Delta}} = \frac{\frac{a_p}{a_{px}} \bar{\beta} \bar{\Delta}}{\frac{a_p}{\bar{a}_p} \bar{\beta} \bar{\Delta}} = \frac{\bar{a}_p}{a_{px}} \frac{\bar{\beta}_d}{\bar{\beta}_p} \frac{\bar{\Delta}_d}{\bar{\Delta}_p} \\ &= R_{p\bar{B}_d\bar{D}_d} \end{aligned} \quad (C.3)$$

Define: $S = \frac{1}{\bar{R}_p \bar{B}_p \bar{D}_d}$

It can be readily seen that, if the sectional area curve and the design waterline have their maximum values at the same longitudinal position, the value of S will be 1, i.e., $S = 1$. Hence, the area curve obtained will be equal to that desired.

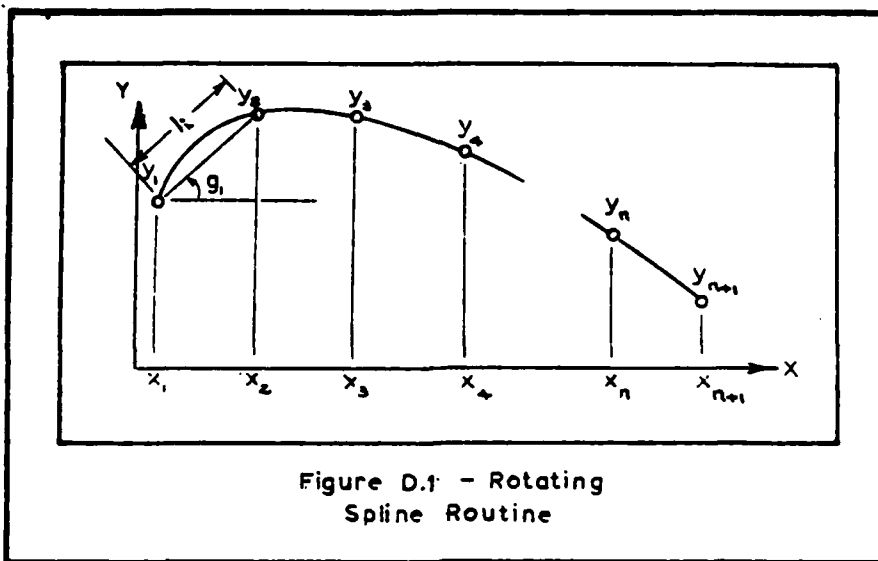
If this is not the case, the designer has one option which will permit him to create a ship with the desired values of C_p , LCB, C_w and LCF.

- The designer must select a common longitudinal position about which to alter both curves. This will permit him to freeze either the point of maximum section or the point of maximum beam. Not both.

- The only other alternative is to carry out the original procedure and accept slightly different values of C_p and C_w . LCB and LCF will be as desired. The factor by which C_w will differ is:

$$W = \frac{R_p^* D_d^*}{A_d^*} \quad (C.5)$$

APPENDIX D, [15]



For n intervals bounded by $n+1$ points the curve between points P_i and P_{i+1} may be computed as follows:

1. Compute initially:

$$a. \quad g_1 = \begin{cases} \arctan [(y_2 - y_1) / (x_2 - x_1)], & \text{if } x_1 \neq x_2 \\ \pi/2, & \text{if } x_1 = x_2 \end{cases}$$

$$b. \quad p_1 = g_1$$

2. Compute n times for $i = 1(1)n$

$$a. \ell_i = \{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2\}^{1/2}$$

b.

$$g_i = g_{i-1} + \arctan \left\{ \frac{(y_{i+1} - y_i)(x_i - x_{i-1}) - (y_i - y_{i-1})(x_{i+1} - x_i)}{(x_{i+1} - x_i)(x_i - x_{i-1}) + (y_{i+1} - y_i)(y_i - y_{i-1})} \right\}$$

only if $i > 1$

$$c. k_i = \begin{cases} -1/2, & \text{if } i = 1 \\ -1 \\ 2 + (k_{i-1} + 2) \ell_i / \ell_{i-1} \end{cases}, \quad \text{if } i > 1$$

$$d. r_i = \frac{3k_i (p_i - g_i)}{k_i + 2}$$

$$e. p_{i+1} = \begin{cases} p_i - r_i(1 + 1/k_i), & \text{if } k_i \neq 0 \\ \frac{3g_i - p_i}{2}, & \text{if } k = 0 \end{cases}$$

3. Compute once:

$$a. q_{n+1} = 0$$

$$b. d_{n+1} = p_{n+1} + q_{n+1}$$

4. Compute n times for $i = 1(-1)n$

a. $q_i = r_i + q_{i+1} k_i$

b. $d_i = p_i + q_i$

c. $a_i = \tan(d_i - g_i)$

d. $b_i = \tan(d_{i+1} - g_i)$

The above procedure applies to the case where the ends are pinned, i.e., $d^2y/dx^2 = 0$ at $x = x_1$ and $x = x_{n+1}$. If, however, it is desired to have the beginning slope equal to t_1 , the following changes must be made:

eqn. 1.b. $p_1 = t_1$

2.c. $k_1 = 0$

If it is desired to specify the end slope as t_{n+1} :

eqn. 3.a. $q_{n+1} = t_{n+1} - p_{n+1}$

To interpolate any point on the curve the following parametric equations are used:

$$x = x_i + (x_{i+1} - x_i)t - (y_{i+1} - y_i)t(1-t)[a_i(1-t) - b_it]$$

$$y = y_i + (y_{i+1} - y_i)t - (x_{i+1} - x_i)t(1-t)[a_i(1-t) - b_it]$$

$$\text{for } 0 \leq t \leq 1$$

Section 3.2.4 describes the actual interpolation procedure.

APPENDIX E

The following procedure was taken from reference [20] and is that used to obtain the roots of the parametric equations for x and y shown in Appendix D.

Given the general form of the cubic polynomial:

$$x^3 + ax^2 + bx + c = 0 \quad (\text{E.1})$$

this may be reduced to the following by dividing by $x = x_1 - a/3$:

$$x_1^3 = Ax_1 + B \quad (\text{E.2})$$

where

$$A = 3(a/3)^2 - b$$

$$B = -2(a/3)^3 + b(a/3) - c \quad (\text{E.3})$$

Defining

$$p = A/3 \text{ and } q = B/2 \quad (\text{E.4})$$

The roots of equation (E.2) are as follows:

Case I: $q^2 - p^3 > 0$, there is one real root

$$x_1 = \{q + \sqrt{q^2 - p^3}\}^{1/3} + \{q - \sqrt{q^2 - p^3}\}^{1/3} \quad (\text{E.5})$$

There are also two complex conjugate roots.

Case II: $q^2 - p^3 = 0$, there are three real roots of which two are repeated, i.e., only two roots are unique.

$$x_1 = 2(q)^{1/3}; x_2 = -(q)^{1/3}; x_3 = x_2 \quad (\text{E.6})$$

Case III: $q^2 - p^3 < 0$, there are three real and distinct roots.

$$x_1 = 2\sqrt{p} \cos (U/3)$$

$$x_2 = 2\sqrt{p} \cos (U/3 + 2\pi/3) \quad (\text{E.7})$$

$$x_3 = 2\sqrt{p} \cos (U/3 + 4\pi/3)$$

where

$$\cos U = q/p\sqrt{p}$$

$$0 \leq U \leq \pi \quad (\text{E.8})$$

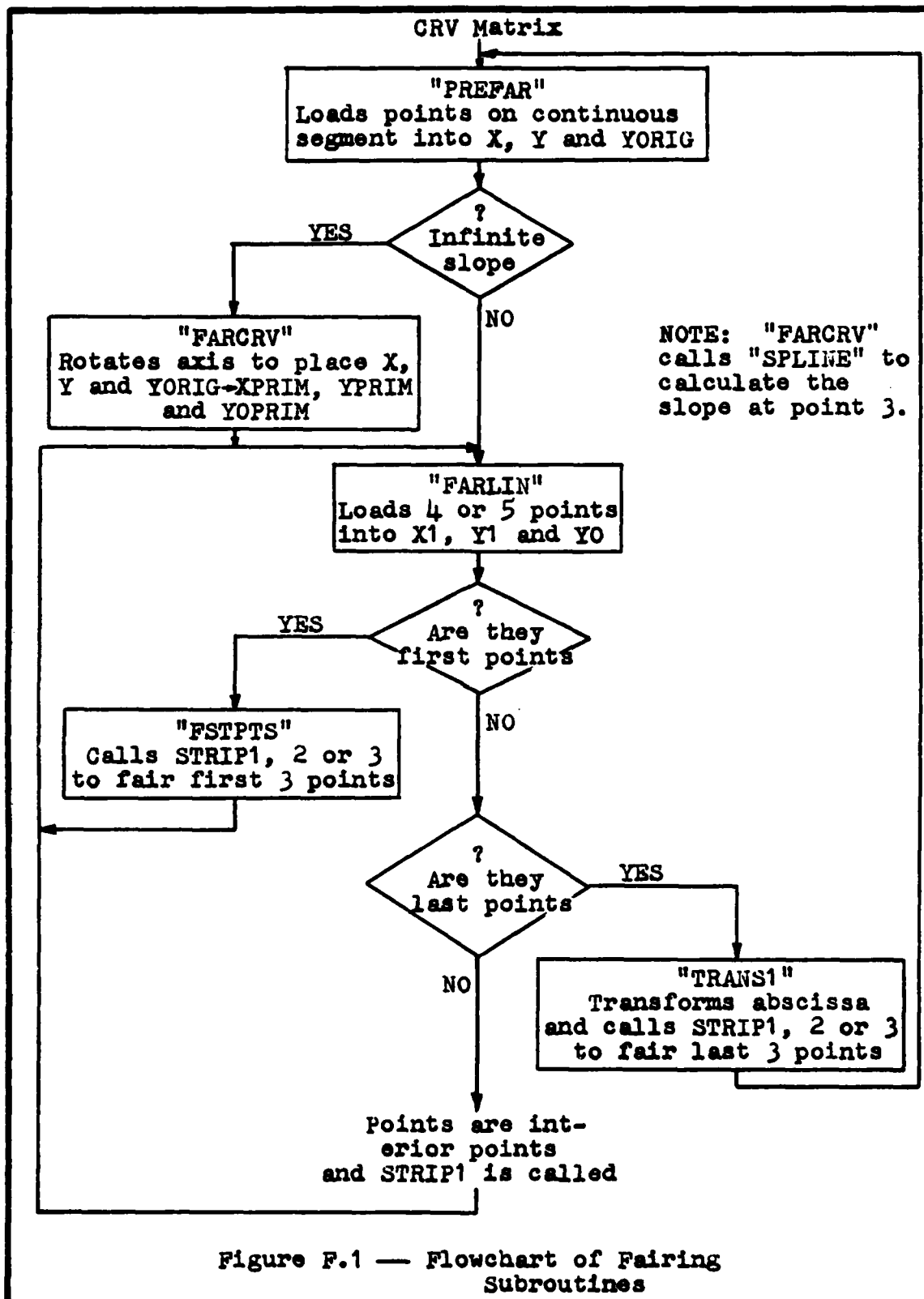
NOTE: These are roots of equation (E.2). To obtain the roots of equation (E.1) $-p/3$ must be added to the above solutions.

APPENDIX F

Figures F.1 and F.2 are conceptual flowcharts of the fairing and the splinning and interpolation procedures respectively. In the program listing that follows there is a short MAIN segment that requests the input data for a specific line and generates twenty-one (including end points) interpolated data points. While this program segment might prove of some value, it was designed primarily to test the various subroutines.

NOTE: The program as listed requires the use of the LEQTLF Subroutine from the IMSL library. This subroutine is used in STRIP1 and STRIP2 to solve a 4x4 and a 3x3 system of simultaneous linear equations. For these SMAT is the coefficient matrix and T is the resultant column vector. If this library is not available any equivalent procedure could be substituted.

NOTE: Due to time constraints at the time of publication, the program, as listed, will not accommodate curves with point types three or five. It will, however, handle curves without straight line segments and infinite slopes at end points.



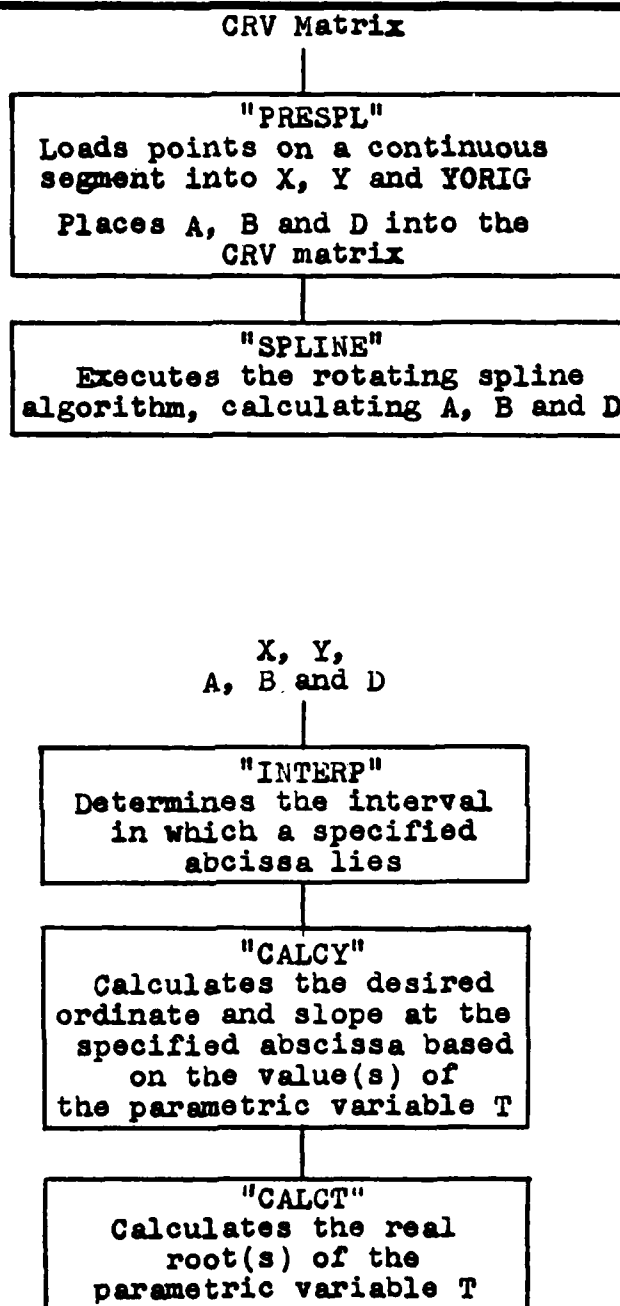


Figure F.2 — Flowchart of Splinning and Interpolation Subroutines

```

COMMON /CON1/SHAT(4,4),T(4),WKAREA(4)
COMMON /CON2/X(30),Y(30),YORIG(30)
COMMON /CON3/X1(5),Y1(5),YO(5)
COMMON /CON4/IPAIR,ACC,LIMIT
COMMON /CON5/XPRIM(30),YPRIM(30),YOPRIM(30)
COMMON /CON6/XCRV(30),YCRV(30),YOCR(30)
COMMON /CON7/A(30),B(30),D(30)
COMMON /CON8/Q
COMMON /CON9/CRV(32,7)
COMMON /CON10/G(30),S(30),C(30),QS(30),R(30),P(30)
COMMON /CON11/YINT(10),DYDX(10)
COMMON /CON12/IPT(30)
COMMON /CON13/KCRV
COMMON /CON14/RT(3),KPLG2
WRITE(6,1000)
1000 FORMAT(' ','PREPARE TO ENTER LINES 31 & 32 OF CRV ARRAY.')
```

C
1C
2C
6

```

DO 1 N=1,4
DO 2 NH=1,4
SHAT(N,NH)=0.
```

```
1 CONTINUE
```

```

READ(5,*) (CRV(31,I), I=1,4)
READ(5,*) (CRV(32,I), I=1,3)
NPTS=INT(CRV(31,2)+.1)
```

```
DO 10 I=1,NPTS
```

```
IF (I.EQ.1) WRITE(6,99) NPTS
```

```
99 FORMAT('0','BEGIN ENTERING DATA FOR THE ',I2,' DATA POINTS.')
```

```
10 READ(5,*) CRV(I,1),CRV(I,2),CRV(I,4)
```

```
CRV(I,3)=CRV(I,2)
```

```
WRITE(6,101)
```

```
101 FORMAT(' ','ENTER ACC, TOL AND LIMIT.')
```

```
READ(5,*) ACC,TOL,LIMIT
```

```
CALL PREPAR(TOL)
```

```
DO 102 I=1,NPTS
```

```

00000010 HAIN0001
00000020 HAIN0002
00000030 HAIN0003
00000040 HAIN0004
00000050 HAIN0005
00000060 HAIN0006
00000070 HAIN0007
00000080 HAIN0008
00000090 HAIN0009
00000100 HAIN0010
00000110 HAIN0011
00000120 HAIN0012
00000130 HAIN0013
00000140 HAIN0014
00000150 HAIN0015
00000160 HAIN0016
00000170 HAIN0017
00000180 HAIN0018
00000190 HAIN0019
00000200 HAIN0020
00000210 HAIN0021
00000220 HAIN0022
00000230 HAIN0023
00000240 HAIN0024
00000250 HAIN0025
00000260 HAIN0026
00000270 HAIN0027
00000280 HAIN0028
00000290 HAIN0029
00000300 HAIN0030
00000310 HAIN0031
00000320 HAIN0032
00000330 HAIN0033
00000340 HAIN0034
00000350 HAIN0035
00000360 HAIN0036
```

```

104 IF (I.EQ.1) WRITE (6,104)
    FORMAT(' ', ORIGINAL ',10X,' PAIRED ')
103 WRITE (6,103) CRV(I,2), CRV(I,3)
102 FORMAT(' ', F10.4, 10X, F10.4)
    CONTINUE
301 WRITE (6,301)
    FORMAT ('0', PRESPL AND SPLINE CHECK')
    DO 302 I=1,NPTS
        X(I)=CRV(I,1)
        Y(I)=CRV(I,3)
        IPT(I)=INT(CRV(I,4)+.1)
        YORIG(I)=CRV(I,2)
        CALL PRESPL
302 WRITE (6,201)
201 FORMAT(' ', INTERPOLATION CHECK')
    NP1=NPTS+1
    WRITE (6,303)
303 FORMAT ('0', XINT ',10X,' YINT ',10X,' DYDX')
    DX=(X(NPTS)-X(1))/20.
    DO 105 I=1,21
        XINT=X(1)+FLOAT(I-1)*DX
        CALL INTERP(NP1,XINT,KFLG1)
        IF (KFLG1.EQ.0) GO TO 105
        DO 106 J=1,KFLG1
            WRITE (6,107) XINT,YINT(J),DYDX(J)
107 FORMAT(' ', F10.4, 10X, F10.4, 10X, F10.4)
106 CONTINUE
105 CONTINUE
    STOP
    END

```

```

00000370 HAIN0037
00000380 HAIN0038
00000390 HAIN0039
00000400 HAIN0040
00000410 HAIN0041
00000420 HAIN0042
00000430 HAIN0043
00000440 HAIN0044
00000450 HAIN0045
00000460 HAIN0046
00000470 HAIN0047
00000480 HAIN0048
00000490 HAIN0049
00000500 HAIN0050
00000510 HAIN0051
00000520 HAIN0052
00000530 HAIN0053
00000540 HAIN0054
00000550 HAIN0055
00000560 HAIN0056
00000570 HAIN0057
00000580 HAIN0058
00000590 HAIN0059
00000600 HAIN0060
00000610 HAIN0061
00000620 HAIN0062
00000630 HAIN0063
00000640 HAIN0064
00000650 HAIN0065
00000660 HAIN0066

```

```

C THIS SUBROUTINE TAKES CURVE DATA IN A 2-D ARRAY AND SPLINES IT .O
C DETERMINE COEFFICIENTS A,B AND ANGLES D. IE. CRV(XX,5 6 OR7).
SUBROUTINE PRESPL
COMMON /CON2/X(30),Y(30),YORIG(30)
COMMON /CON7/A(30),B(30),D(30)
COMMON /CON9/CRV(32,7)
NPTS=INT(CRV(31,2)+.1)
MARK=0
DO 101 I=1,NPTS
  IF (I.EQ.1.OR.INT(CRV(I,4)+.1).EQ.0.OR.INT(CRV(I,4)+.1).EQ.1) GO
  10 201
  IF (INT(CRV(I,4)+.1).EQ.3.OR.INT(CRV(I,4)+.1).EQ.5) GO TO 401
102 KP1=K+1
  IF (K.EQ.I) KP1=I
  X(KP1)=CRV(I,1)
  Y(KP1)=CRV(I,3)
  K=KP1
  GO TO 401
101 CONTINUE
999 RETURN
601 EC=EC1+EC2
  CALL SPLINE(K,EC,ES1P,ES2P)
  IF (I.EQ.NPTS) GO TO 702
  DO 701 L=1,K
    LC=I-L
    LK=K-L+1
    CRV(LC,5)=A(LK)
    CRV(LC,6)=B(LK)
    CRV(LC,7)=D(LK)
701 GO TO 101
702 DO 703 L=1,K
    LC=I+1-L
    LK=K+1-L
    CRV(LC,5)=A(LK)
    CRV(LC,6)=B(LK)
    CRV(LC,7)=D(LK)
703
00000670 PRSP0001
00000680 PRSP0002
00000690 PRSP0003
00000700 PRSP0004
00000710 PRSP0005
00000720 PRSP0006
00000730 PRSP0007
00000740 PRSP0008
00000750 PRSP0009
00000760 PRSP0010
00000770 PRSP0011
00000780 PRSP0012
00000790 PRSP0013
00000800 PRSP0014
00000810 PRSP0015
00000820 PRSP0016
00000830 PRSP0017
00000840 PRSP0018
00000850 PRSP0019
00000860 PRSP0020
00000870 PRSP0021
00000880 PRSP0022
00000890 PRSP0023
00000900 PRSP0024
00000910 PRSP0025
00000920 PRSP0026
00000930 PRSP0027
00000940 PRSP0028
00000950 PRSP0029
00000960 PRSP0030
00000970 PRSP0031
00000980 PRSP0032
00000990 PRSP0033
00001000 PRSP0034
00001010 PRSP0035
00001020 PRSP0036

```

```

GO TO 101
C
C
201 K=I-MARK
    IF (K.EQ.2.AND.MARK.NE.0) GO TO 501
202 X(K)=CRV(I,1)
    Y(K)=CRV(I,3)
    IF (I.EQ.NPTS) GO TO 102
    GO TO 101
501 IN1=I-1
    X(1)=CRV(IN1,1)
    Y(1)=CRV(IN1,3)
    GO TO 202
401 MARK=I-1
C CALCULATE BEGINNING END CONDITION.
    KM1=K-1
    IF (I-K.EQ.1.OR.I-K.EQ.0) GO TO 301
    INK=I-K
    IF (INT(CRV(INK,4)+.1).EQ.0.OR.INT(CRV(INK,4)+.1).EQ.1) GO TO 302
    GO TO 303
C CALCULATE END CONDITION.
402 IF (I.EQ.NPTS) GO TO 304
    IF (INT(CRV(I,4)+.1).EQ.3.OR.INT(CRV(I,4)+.1).EQ.5) GO TO 305
    IF (INT(CRV(I,4)+.1).EQ.0.OR.INT(CRV(I,4)+.1).EQ.1) GO TO 306
    GO TO 999
C
301 ES1P=CRV(31,3)
    EC1=FLOAT(INT(CRV(32,2)))
    GO TO 402
C
302 ES1P=CRV(INK,7)
    EC1=2.
    IF (INT(CRV(INK,4)+.1).EQ.0) EC1=3.
    GO TO 402
C
303 INKM1=INK-1

```

```

00001030 PRSP0037
00001040 PRSP0038
00001050 PRSP0039
00001060 PRSP0040
00001070 PRSP0041
00001080 PRSP0042
00001090 PRSP0043
00001100 PRSP0044
00001110 PRSP0045
00001120 PRSP0046
00001130 PRSP0047
00001140 PRSP0048
00001150 PRSP0049
00001160 PRSP0050
00001170 PRSP0051
00001180 PRSP0052
00001190 PRSP0053
00001200 PRSP0054
00001210 PRSP0055
00001220 PRSP0056
00001230 PRSP0057
00001240 PRSP0058
00001250 PRSP0059
00001260 PRSP0060
00001270 PRSP0061
00001280 PRSP0062
00001290 PRSP0063
00001300 PRSP0064
00001310 PRSP0065
00001320 PRSP0066
00001330 PRSP0067
00001340 PRSP0068
00001350 PRSP0069
00001360 PRSP0070
00001370 PRSP0071
00001380 PRSP0072

```

00001390 PRSP0073
 00001400 PRSP0074
 00001410 PRSP0075
 00001420 PRSP0076
 00001430 PRSP0077
 00001440 PRSP0078
 00001450 PRSP0079
 00001460 PRSP0080
 00001470 PRSP0081
 00001480 PRSP0082
 00001490 PRSP0083
 00001500 PRSP0084
 00001510 PRSP0085
 00001520 PRSP0086
 00001530 PRSP0087
 00001540 PRSP0088

```

ES1P=ATAN((Y (IMK) -Y (IMKH1)) / (X (IMK) -X (IMKH1)))
EC1=3.
GO TO 402

C 304 ES2P=CRV (32,3)
    EC2=CRV (32,2) -FLOAT (INT (CRV (32,2)))
    GO TO 601

C 305 IM1=I-1
    ES2P=ATAN ((CRV (I,3) -CRV (IM1,3)) / (CRV (I,1) -CRV (IM1,1)))
    EC2=3.
    GO TO 601

C 306 EC2=2.
    GO TO 601
    END
  
```

```

C THIS SUBROUTINE CALCULATES THE DATA NECESSARY TO GENERATE A CUBIC
C POLYNOMIAL THROUGH A SERIES OF GIVEN POINTS. THE ALGORITHM USED
C DEVELOPED BY PROFESSOR H. SODING AND INVOLVES THE INCREMENTAL
C ROTATION OF THE AXES TO ACCOMMODATE INFINITE SLOPES.
      SUBROUTINE SPLINE(NP1,EC,ES1,ES2)
      COMMON /COM2/X(30),Y(30),YORIG(30)
      COMMON /COM7/A(30),B(30),D(30)
      COMMON /COM10/G(30),S(30),C(30),QS(30),R(30),P(30)
      INTEGER DEC
      IND=INT(EC)
      DEC=INT((EC-FLOAT(IND)+.01)*10.)
      PI=ACOS(-1.)
      G(1)=PI/2.
      IF (X(2).NE.X(1)) G(1)=ATAN((Y(2)-Y(1))/(X(2)-X(1)))
      P(1)=G(1)
      N=NP1-1
      IF (IND.EQ.3) P(1)=ES1
      IF (IND.EQ.4.AND.Y(1).GT.Y(2)) P(1)=-PI/2.
      IF (IND.EQ.4.AND.Y(1).LE.Y(2)) P(1)=PI/2.
      DO 101 I=1,N
      IP1=I+1
      IM1=I-1
      S(I)=SQRT((X(IP1)-X(I))**2+(Y(IP1)-Y(I))**2)
      A2=X(I)-X(IM1)
      A3=Y(I)-Y(IM1)
      A1=Y(IP1)-Y(I)
      A4=X(IP1)-X(I)
      IF (I.GT.1) G(I)=G(IM1)+ATAN((A1*A2-A3*A4)/(A4*A2+A1*A3))
      C(I)=-.5
      IF (IND.EQ.3.OR.IND.EQ.4) C(1)=0.
      IF (I.GT.1) C(I)=-1./(2.+(C(IM1)+2.)*S(I)/S(IM1))
      R(I)=3.*C(I)*(P(I)-G(I))/(C(I)+2.)
      P(IP1)=(3.*G(I)-P(I))/2.
      IF (C(I).NE.0.) P(IP1)=P(I)-R(I)*(1.+1./C(I))
      QS(NP1)=0.
      IF (DEC.EQ.3.OR.DEC.EQ.4) QS(NP1)=ES2-P(NP1)
101

```

```

00001550 SPLN0001
0000001560 SPLN0002
00001570 SPLN0003
00001580 SPLN0004
00001590 SPLN0005
00001600 SPLN0006
00001610 SPLN0007
00001620 SPLN0008
00001630 SPLN0009
00001640 SPLN0010
00001650 SPLN0011
00001660 SPLN0012
00001670 SPLN0013
00001680 SPLN0014
00001690 SPLN0015
00001700 SPLN0016
00001710 SPLN0017
00001720 SPLN0018
00001730 SPLN0019
00001740 SPLN0020
00001750 SPLN0021
00001760 SPLN0022
00001770 SPLN0023
00001780 SPLN0024
00001790 SPLN0025
00001800 SPLN0026
00001810 SPLN0027
00001820 SPLN0028
00001830 SPLN0029
00001840 SPLN0030
00001850 SPLN0031
00001860 SPLN0032
00001870 SPLN0033
00001880 SPLN0034
00001890 SPLN0035
00001900 SPLN0036

```

```

D(NP1)=P(NP1)+QS(NP1)
DO 102 I=1,N
K=N+1-I
KP1=K+1
QS(K)=R(K)+QS(KP1)*C(K)
D(K)=P(K)+QS(K)
A(K)=TAN(D(K)-G(K))
102 B(K)=TAN(D(KP1)-G(K))
RETURN
END

```

```

00001910 SPLN0037
00001920 SPLN0038
00001930 SPLN0039
00001940 SPLN0040
00001950 SPLN0041
00001960 SPLN0042
00001970 SPLN0043
00001980 SPLN0044
00001990 SPLN0045
00002000 SPLN0046

```

```

C THIS SUBROUTINE CALCULATES INTERPOLATED VALUES OF POSITION AND SLOPE 00002010 INTP0001
C OF A CURVE THAT HAS BEEN SPLINED PREVIOUSLY. IT WILL GIVE MULTIPLE 00002020 INTP0002
C VALUES IF THE CURVE IS NOT SINGLE VALUED. 00002030 INTP0003
SUBROUTINE INTERP(NP1,XINT,KPLG1) 00002040 INTP0004
COMMON /COM2/X(30),Y(30),YORIG(30) 00002050 INTP0005
COMMON /COM7/A(30),B(30),D(30) 00002060 INTP0006
COMMON /COM9/CRV(32,7) 00002070 INTP0007
COMMON /COM11/YINT(10),DYDX(10) 00002080 INTP0008
COMMON /COM12/IPT(30) 00002090 INTP0009
N=NP1-1 00002100 INTP0010
NM1=N-1 00002110 INTP0011
KPLG1=0 00002120 INTP0012
DO 99 I=1,N 00002130 INTP0013
IPT(I)=INT(CRV(I,4)+.1) 00002140 INTP0014
DO 101 I=1,NM1 00002150 INTP0015
IP1=I+1 00002160 INTP0016
IF (XINT.EQ.X(I).AND.I.EQ.1) GO TO 201 00002170 INTP0017
IF (XINT.EQ.X(IP1)) GO TO 202 00002180 INTP0018
IF (X(IP1)-GT.X(I).AND.XINT.GT.X(I).AND.XINT.LT.X(IP1)) CALL CALCY00002190 INTP0019
1(KPLG1,X(I),X(IP1),Y(I),Y(IP1),A(I),B(I),XINT,IPT(IP1)) 00002200 INTP0020
IF (X(IP1)-LT.X(I).AND.XINT.LT.X(I).AND.XINT.GT.X(IP1)) CALL CALCY00002210 INTP0021
1(KPLG1,X(I),X(IP1),Y(I),Y(IP1),A(I),B(I),XINT,IPT(IP1)) 00002220 INTP0022
101 CONTINUE 00002230 INTP0023
RETURN 00002240 INTP0024
C 201 KPLG1=KPLG1+1 00002250 INTP0025
YINT(KPLG1)=Y(I) 00002260 INTP0026
IF (D(I).NE.ACOS(-1.)/2.) DYDX(KPLG1)=TAN(D(I)) 00002270 INTP0027
IF (D(I).EQ.ACOS(-1.)/2.) DYDX(KPLG1)=1.E50 00002280 INTP0028
GO TO 101 00002290 INTP0029
202 KPLG1=KPLG1+1 00002300 INTP0030
YINT(KPLG1)=Y(IP1) 00002310 INTP0031
IF (D(IP1).NE.ACOS(-1.)/2.) DYDX(KPLG1)=TAN(D(IP1)) 00002320 INTP0032
IF (D(IP1).EQ.ACOS(-1.)/2.) DYDX(KPLG1)=1.E50 00002330 INTP0033
GO TO 101 00002340 INTP0034
END 00002350 INTP0035
00002360 INTP0036

```

```

SUBROUTINE CALCY(KPLG1,XI,XIP1,YI,YIP1,AI,BI,XINT,IPTIP1)
COMMON /COM11/YINT(10),DYDX(10)
COMMON /COM14/RT(3),KPLG2
IP1=I+1
IF (IPTIP1.EQ.3..OR.IPTIP1.EQ.5.) GO TO 101
CALL CALCT(XI,XIP1,YI,YIP1,AI,BI,XINT)
IF (KPLG2.EQ.0) GO TO 301
A1=YIP1-YI
A2=XIP1-XI
DO 102 K=1,KPLG2
A3=RT(K)*(1.-RT(K))
A4=1.-2.*RT(K)
A5=1.-RT(K)
A6=AI+A5-BI*RT(K)
KPLG1=KPLG1+1
YINT (KPLG1)=YI+A1*RT(K)+A2*A3*A6
A7=AI+BI
A8=4.*AI+2.*BI
A9=3.*RT(K)**2
A10=A9+A7-RT(K)*A8+AI
DYDT=A1+A2*A10
DXDT=A2-A1*A10
DYDX (KPLG1)=1.E50
102 IF (DXDT.NE.0.) DYDX (KPLG1)=DYDT/DXDT
999 RETURN
C
101 KPLG1=KPLG1+1
DYDX (KPLG1)=(YIP1-YI)/(XIP1-XI)
YINT (KPLG1)=YI+DYDX (KPLG1)*(XINT-XI)
GO TO 999
301 WRITE (6,302)
302 FORMAT(' ','THERE IS NO VALUE OF T WITHIN THE LIMITS.')
```

```

00002370 CLCY0001
00002380 CLCY0002
00002390 CLCY0003
00002400 CLCY0004
00002410 CLCY0005
00002420 CLCY0006
00002430 CLCY0007
00002440 CLCY0008
00002450 CLCY0009
00002460 CLCY0010
00002470 CLCY0011
00002480 CLCY0012
00002490 CLCY0013
00002500 CLCY0014
00002510 CLCY0015
00002520 CLCY0016
00002530 CLCY0017
00002540 CLCY0018
00002550 CLCY0019
00002560 CLCY0020
00002570 CLCY0021
00002580 CLCY0022
00002590 CLCY0023
00002600 CLCY0024
00002610 CLCY0025
00002620 CLCY0026
00002630 CLCY0027
00002640 CLCY0028
00002650 CLCY0029
00002660 CLCY0030
00002670 CLCY0031
00002680 CLCY0032
00002690 CLCY0033
00002700 CLCY0034
```

```

C THIS FUNCTION CALCULATES THE VALUE OF T USING THE CLOSED FORM CUBE
C ROOT EQUATION FOUND IN THE CRC MATH TABLES. THE VALUE OF T SELECTED
C IS THE ROOT THAT IS BOTH REAL AND LYING BETWEEN 0 AND 1.
SUBROUTINE CALCT(XI,XIP1,YI,YIP1,AI,BI,XINT)
COMMON /COM14/RT(3),KPLG2
601 FORMAT(' ','THERE ARE MORE THAN 2 REAL ROOTS OF T. YOU NEED MORE
KPLG2=0
A1=XIP1-XI
A2=YIP1-YI
A3=XI-XINT
C0=A3
C1=A1-A2*AI
C2=A2*(2.*AI+BI)
C3=-A2*(AI+BI)
AA=C2/C3
BB=C1/C3
CC=C0/C3
A=3.*(AA/3.)*2-BB
B=-2.*(AA/3.)*3+BB*(AA/3.)*-CC
PI=ACOS(-1.)
P=A/3.
Q=B/2.
D=Q**2-P**3
IF (D.GT.0.) SQD=SQRT(D)
E3=1./3.
IF (D.GT.0.) GO TO 101
IF (D.EQ.0.) GO TO 102
IF (D.LT.0.) GO TO 103
GO TO 999
101 IF (Q+SQD.GE.0.) XA=(Q+SQD)**E3
IF (Q+SQD.LT.0.) XA=- (ABS(Q+SQD)) **E3
IF (Q-SQD.GE.0.) XB=(Q-SQD)**E3
IF (Q-SQD.LT.0.) XB=- (ABS(Q-SQD)) **E3
X1=XA+XB-AA/3.
IF (X1.GT.0..AND.X1.LT.1.) GO TO 201
00002710 CLCT0001
00002720 CLCT0002
00002730 CLCT0003
00002740 CLCT0004
00002750 CLCT0005
00002760 CLCT0006
00002770 CLCT0007
00002780 CLCT0008
00002790 CLCT0009
00002800 CLCT0010
00002810 CLCT0011
00002820 CLCT0012
00002830 CLCT0013
00002840 CLCT0014
00002850 CLCT0015
00002860 CLCT0016
00002870 CLCT0017
00002880 CLCT0018
00002890 CLCT0019
00002900 CLCT0020
00002910 CLCT0021
00002920 CLCT0022
00002930 CLCT0023
00002940 CLCT0024
00002950 CLCT0025
00002960 CLCT0026
00002970 CLCT0027
00002980 CLCT0028
00002990 CLCT0029
00003000 CLCT0030
00003010 CLCT0031
00003020 CLCT0032
00003030 CLCT0033
00003040 CLCT0034
00003050 CLCT0035
00003060 CLCT0036

```

00003070 CLCT0037
 00003080 CLCT0038
 00003090 CLCT0039
 00003100 CLCT0040
 00003110 CLCT0041
 00003120 CLCT0042
 00003130 CLCT0043
 00003140 CLCT0044
 00003150 CLCT0045
 00003160 CLCT0046
 00003170 CLCT0047
 00003180 CLCT0048
 00003190 CLCT0049
 00003200 CLCT0050
 00003210 CLCT0051
 00003220 CLCT0052
 00003230 CLCT0053
 00003240 CLCT0054
 00003250 CLCT0055
 00003260 CLCT0056
 00003270 CLCT0057
 00003280 CLCT0058
 00003290 CLCT0059
 00003300 CLCT0060
 00003310 CLCT0061
 00003320 CLCT0062
 00003330 CLCT0063
 00003340 CLCT0064
 00003350 CLCT0065
 00003360 CLCT0066
 00003370 CLCT0067
 00003380 CLCT0068
 00003390 CLCT0069
 00003400 CLCT0070
 00003410 CLCT0071
 00003420 CLCT0072

GO TO 999
 102 IF (Q-GE.0.) GO TO 502
 X1=-2.*(ABS(Q))*E3-AA/3.
 X2=-X1/2.-AA/3.
 GO TO 501
 502 X1=2.*Q*E3-AA/3.
 X2=-X1/2.-AA/3.
 501 IF (X1-GT.0.-AND.X1-LT.1.) GO TO 202
 203 IF (X2-GT.0.-AND.X2-LT.1.) GO TO 204
 GO TO 999
 103 U=ACOS(Q/(P**(1.5)))
 SQP=SQRT(P)
 X1=2.*SQP*COS(U/3.)-AA/3.
 X2=2.*SQP*COS(U/3.+PI*2./3.)-AA/3.
 X3=2.*SQP*COS(U/3.+PI*4./3.)-AA/3.
 IF (X1-GT.0.-AND.X1-LT.1.) GO TO 205
 206 IF (X2-GT.0.-AND.X2-LT.1.) GO TO 207
 208 IF (X3-GT.0.-AND.X3-LT.1.) GO TO 209
 GO TO 999
 201 KPLG2=1
 RT(1)=X1
 GO TO 999
 202 KPLG2=1
 RT(1)=X1
 GO TO 203
 204 KPLG2=KPLG2+1
 RT(KPLG2)=X2
 GO TO 999
 205 KPLG2=1
 RT(1)=X1
 GO TO 206
 207 KPLG2=KPLG2+1
 RT(KPLG2)=X2
 GO TO 208
 209 KPLG2=KPLG2+1
 RT(KPLG2)=X3

999 IF (KPLG2.EQ.2.OR.KPLG2.EQ.3) WRITE (6,601)
RETURN
END

00003430 CLCT0073
00003440 CLCT0074
00003450 CLCT0075

```

C THIS SUBROUTINE TAKES A SERIES OF POINTS WITH END CONDITIONS AND
C CALLS THE APPROPRIATE PAIRING SUBROUTINE.
SUBROUTINE PREPAR(TOL)
COMMON /COM2/I(30),Y(30),YORIG(30)
COMMON /COM6/ICRV(30),YCRV(30),YOCRV(30)
COMMON /COM9/CRV(32,7)
COMMON /COM12/KCRV
INTEGER BGN,END
NPTS=INT(CRV(31,2)+.1)
1001 CONTINUE
MARK=0
BGN=INT(CRV(32,2))
END=INT((CRV(32,2)-FLOAT(BGN)+.01)*10.)
DO 101 I=1,NPTS
KCRV=0
IF (I.EQ.1.OR.INT(CRV(I,4)+.1).EQ.0.OR.INT(CRV(I,4)+.1).EQ.1) GO
10 201
IF (INT(CRV(I,4)+.1).EQ.3.OR.INT(CRV(I,4)+.1).EQ.5) GO TO 401
102 KP1=K+1
IF (K.EQ.1.OR.K.EQ.I) KP1=I
X(KP1)=CRV(I,1)
Y(KP1)=CRV(I,3)
YORIG(KP1)=CRV(I,2)
K=KP1
GO TO 401
101 CONTINUE
999 RETURN
601 EC=EC1+EC2
IEC1=INT(EC1+.1)
IEC2=INT((EC2+.01)*10.)
IF (IEC1.EQ.5.OR.IEC2.EQ.4) KCRV=1
IF (IEC1.EQ.4.OR.IEC2.EQ.4) CALL PARCRV(K,EC,TOL,ES1P,ES2P)
IF (KCRV.NE.1) GO TO 12
DO 11 JJ=1,K
X(JJ)=ICRV(JJ)
YORIG(JJ)=YOCRV(JJ)

```

```

00003460 PRPR0001
00003470 PRPR0002
00003480 PRPR0003
00003490 PRPR0004
00003500 PRPR0005
00003510 PRPR0006
00003520 PRPR0007
00003530 PRPR0008
00003540 PRPR0009
00003550 PRPR0010
00003560 PRPR0011
00003570 PRPR0012
00003580 PRPR0013
00003590 PRPR0014
00003600 PRPR0015
00003610 PRPR0016
00003620 PRPR0017
00003630 PRPR0018
00003640 PRPR0019
00003650 PRPR0020
00003660 PRPR0021
00003670 PRPR0022
00003680 PRPR0023
00003690 PRPR0024
00003700 PRPR0025
00003710 PRPR0026
00003720 PRPR0027
00003730 PRPR0028
00003740 PRPR0029
00003750 PRPR0030
00003760 PRPR0031
00003770 PRPR0032
00003780 PRPR0033
00003790 PRPR0034
00003800 PRPR0035
00003810 PRPR0036

```

```

11 Y(JJ)=YCRV(JJ)
12 IF (IEC1.NE.4.AND.IEC2.NE.4) CALL PABLIN(K,EC,TOL,ES1P,ES2P)
   IF (I.EQ.NPTS) GO TO 702
   DO 701 L=1,K
     LC=I-L
     LK=K-L+1
     CRV(LC,1)=X(LK)
     CRV(LC,2)=YORIG(LK)
701  CRV(LC,3)=Y(LK)
     GO TO 101
702  DO 703 L=1,K
     LC=I+1-L
     LK=K+1-L
     CRV(LC,1)=X(LK)
     CRV(LC,2)=YORIG(LK)
703  CRV(LC,3)=Y(LK)
     GO TO 101
801  WRITE (6,802)
802  FORMAT(' ','NOT ENOUGH DATA POINTS TO PAIR. PAIRING BY-PASSED.')
201  K=I-MARK
202  IF (K.EQ.2.AND.MARK.NE.0) GO TO 501
     X(K)=CRV(I,1)
     Y(K)=CRV(I,3)
     YORIG(K)=CRV(I,2)
     IF (I.EQ.NPTS) GO TO 102
     GO TO 101
501  IH1=I-1
     X(1)=CRV(IH1,1)
     Y(1)=CRV(IH1,3)
     YORIG(1)=CRV(IH1,2)
     GO TO 202
C  CALCULATE THE BEGINNING END CONDITIONS.
401  MARK=I-1
     IF (K.LT.6) GO TO 801
     KH1=K-1

```

```

00003820 PRPR0037
00003830 PRPR0038
00003840 PRPR0039
00003850 PRPR0040
00003860 PRPR0041
00003870 PRPR0042
00003880 PRPR0043
00003890 PRPR0044
00003900 PRPR0045
00003910 PRPR0046
00003920 PRPR0047
00003930 PRPR0048
00003940 PRPR0049
00003950 PRPR0050
00003960 PRPR0051
00003970 PRPR0052
00003980 PRPR0053
00003990 PRPR0054
00004000 PRPR0055
00004010 PRPR0056
00004020 PRPR0057
00004030 PRPR0058
00004040 PRPR0059
00004050 PRPR0060
00004060 PRPR0061
00004070 PRPR0062
00004080 PRPR0063
00004090 PRPR0064
00004100 PRPR0065
00004110 PRPR0066
00004120 PRPR0067
00004130 PRPR0068
00004140 PRPR0069
00004150 PRPR0070
00004160 PRPR0071
00004170 PRPR0072

```

```

IP (I-K.EQ.1.OR.I-K.EQ.0) GO TO 301
INK=I-K
IP (INT(CRV(INK,4)+.1).EQ.0.OR.INT(CRV(INK,4)+.1).EQ.1) GO TO 302
GO TO 303
C CALCULATE END CONDITIONS
402 IF (I.EQ.NPTS) GO TO 304
IF (INT(CRV(I,4)+.1).EQ.3.OR.INT(CRV(I,4)+.1).EQ.5) GO TO 305
IF (INT(CRV(I,4)+.1).EQ.0.OR.INT(CRV(I,4)+.1).EQ.1) GO TO 306
GO TO 999
301 ES1P=CRV(31,3)
EC1=FLOAT(BGN)
GO TO 402
302 EC1=2.
ES1P=CRV(INK,7)
IP (INT(CRV(INK,4)+.1).EQ.0) EC1=3.
GO TO 402
303 INKM1=INK-1
ES1P=ATAN((Y(INK)-Y(INKM1))/(X(INK)-X(INKM1)))
EC1=3.
GO TO 402
C
304 ES2P=CRV(32,3)
EC2=FLOAT(END)/10.
GO TO 601
305 IN1=I-1
ES2P=ATAN((CRV(I,3)-CRV(IN1,3))/(CRV(I,1)-CRV(IN1,1)))
EC2=.3
GO TO 601
306 EC2=.2
GO TO 601
END

```

```

00004180 PRPR0073
00004190 PRPR0074
00004200 PRPR0075
00004210 PRPR0076
00004220 PRPR0077
00004230 PRPR0078
00004240 PRPR0079
00004250 PRPR0080
00004260 PRPR0081
00004270 PRPR0082
00004280 PRPR0083
00004290 PRPR0084
00004300 PRPR0085
00004310 PRPR0086
00004320 PRPR0087
00004330 PRPR0088
00004340 PRPR0089
00004350 PRPR0090
00004360 PRPR0091
00004370 PRPR0092
00004380 PRPR0093
00004390 PRPR0094
00004400 PRPR0095
00004410 PRPR0096
00004420 PRPR0097
00004430 PRPR0098
00004440 PRPR0099
00004450 PRPR0100
00004460 PRPR0101
00004470 PRPR0102
00004480 PRPR0103

```

```

SUBROUTINE PARCRV(N,EC,TOL1,ES1,ES2)
COMMON /CON2/X(30),Y(30),YORIG(30)
COMMON /CON5/XPRIN(30),YPRIN(30),YOPRIN(30)
COMMON /CON6/XCRV(30),YCRV(30),YOCR(30)
COMMON /CON7/A(30),B(30),D(30)
COMMON /CON13/KCRV
INTEGER BGN,END
1001 CONTINUE
DO 10 JJ=1,N
XCRV(JJ)=X(JJ)
YCRV(JJ)=Y(JJ)
10 YOCR(JJ)=YORIG(JJ)
C R IS THE ROTATION ANGLE OF 10 DEGREES.
PI=ACOS(-1.)
R=PI/18.
BGN=INT(EC)
END=INT((EC-FLOAT(BGN)+.01)*10.)

C
C
ITRNS=0
IPAIR=0
IF (BGN.EQ.4.AND.END.NE.4) GO TO 201
IF (END.EQ.4.AND.BGN.NE.4) GO TO 301
GO TO 401
C PAIRS CURVE FOR ONE INFINITE SLOPE AT THE BEGINNING.
201 DO 202 I=1,6
XPRIN(I)=XCRV(I)*COS(R)+YCRV(I)*SIN(R)
YPRIN(I)=-XCRV(I)*SIN(R)+YCRV(I)*COS(R)
202 YOPRIN(I)=-XCRV(I)*SIN(R)+YOCR(I)*COS(R)
ES1PRN=(4./9.)*PI
TOLPRN=TOL1/COS(R)
ECPRN=3.1
KCRV=1
CALL PARLIN(6,ECPRN,TOLPRN,ES1PRN,ES2)
DO 11 JJ=1,6
11 YPRIN(JJ)=Y(JJ)

```

```

DO 203 I=1,6
203 YCRV(I)=XPRM(I)*SIN(R)+YPRM(I)*COS(R)
C CALCULATE THE SLOPE AT POINT 3.
CALL SPLINE(6,4,1,ES1PRM,ES2)
ES1PRM=D(3)+R
ES2PRM=ES2
DO 204 I=3,N
IDX=I-2
XPRM(IDX)=XCRV(I)
YPRM(IDX)=YCRV(I)
204 YOPRM(IDX)=YOCRV(I)
ECPRM=3.+FLOAT(END)/10.
NM2=N-2
KCRV=1
CALL FARLIN(NM2,ECPRM,TOL1,ES1PRM,ES2PRM)
DO 12 JJ=1,NM2
JJP2=JJ+2
YCRV(JJP2)=Y(JJ)
12 YPRM(JJ)=Y(JJ)
GO TO 500
C PAIRS CURVE WHERE INFINITE SLOPE IS AT THE END.
301 NM5=N-5
NM1=N-1
ECPRM=FLOAT(BGN)+.1
KCRV=0
CALL FARLIN(NM1,ECPRM,TOL1,ES1,ES2)
DO 13 JJ=1,NM1
YCRV(JJ)=Y(JJ)
13 YPRM(JJ)=Y(JJ)
C CALCULATE SLOPE AT POINT NM5. I.E., P(NM5)
CALL SPLINE(NM1,ECPRM,ES1,ES2)
ES2PRM=(D(NM5)+R)
ES1PRM=(4./9.)*PI
ECPRM=3.3
TOLPRM=TOL1/COS(R)
DO 302 I=1,6

```

```

00004850 PRCV0037
00004860 PRCV0038
00004870 PRCV0039
00004880 PRCV0040
00004890 PRCV0041
00004900 PRCV0042
00004910 PRCV0043
00004920 PRCV0044
00004930 PRCV0045
00004940 PRCV0046
00004950 PRCV0047
00004960 PRCV0048
00004970 PRCV0049
00004980 PRCV0050
00004990 PRCV0051
00005000 PRCV0052
00005010 PRCV0053
00005020 PRCV0054
00005030 PRCV0055
00005040 PRCV0056
00005050 PRCV0057
00005060 PRCV0058
00005070 PRCV0059
00005080 PRCV0060
00005090 PRCV0061
00005100 PRCV0062
00005110 PRCV0063
00005120 PRCV0064
00005130 PRCV0065
00005140 PRCV0066
00005150 PRCV0067
00005160 PRCV0068
00005170 PRCV0069
00005180 PRCV0070
00005190 PRCV0071
00005200 PRCV0072

```

```

143
302 YOPRIM(I)=-XBAR*SIN(R)+YOCRV(IDX)*COS(R)
    KCRV=1
    CALL PARLIN(6,ECPRM,TOLPRM,ES1PRM,ES2PRM)
    DO 14 JJ=1,6
      YPRIN(JJ)=Y(JJ)
    DO 303 I=1,6
      IDX=N+1-I
      303 YCRV(IDX)=XPRIN(I)*SIN(R)+YPRIM(I)*COS(R)
        GO TO 500
    C THIS SEGMENT PAIRS THE CURVE FOR THE CASE OF INFINITE SLOPES AT BOTH
    401 TOLPRM=TOL1/COS(R)
      ECPRM=3.1
    C ROTATE AND PAIR THE FIRST FIVE POINTS.
    DO 402 I=1,6
      XPRIN(I)=XCRV(I)*COS(R)+YCRV(I)*SIN(R)
      YPRIN(I)=-XCRV(I)*SIN(R)+YCRV(I)*COS(R)
      402 YOPRIM(I)=-XCRV(I)*SIN(R)+YOCRV(I)*COS(R)
        ES1PRM=(4./9.)*PI
        KCRV=1
    CALL PARLIN(6,ECPRM,TOLPRM,ES1PRM,ES2)
    DO 15 JJ=1,6
      YPRIN(JJ)=Y(JJ)
    C PLACE PAIRED POINTS IN THE CRV VECTORS.
    DO 403 I=1,6
      403 YCRV(I)=XPRIN(I)*SIN(R)+YPRIM(I)*COS(R)
    C OBTAIN THE SLOPE AT POINT 2.
    CALL SPLINE(6,ECPRM,ES1PRM,ES2)
    ES2PRM=- (D(2)+2.*R)
    ECPRM=3.3
    C NOW PAIR THE REMAINING POINTS ON THE LINE.
    NM1=N-1
    DO 404 I=1,NM1

```

```

00005210 PRCV0073
00005220 PRCV0074
00005230 PRCV0075
00005240 PRCV0076
00005250 PRCV0077
00005260 PRCV0078
00005270 PRCV0079
00005280 PRCV0080
00005290 PRCV0081
00005300 PRCV0082
00005310 PRCV0083
00005320 PRCV0084
00005330 PRCV0085
00005340 PRCV0086
00005350 PRCV0087
00005360 PRCV0088
00005370 PRCV0089
00005380 PRCV0090
00005390 PRCV0091
00005400 PRCV0092
00005410 PRCV0093
00005420 PRCV0094
00005430 PRCV0095
00005440 PRCV0096
00005450 PRCV0097
00005460 PRCV0098
00005470 PRCV0099
00005480 PRCV0100
00005490 PRCV0101
00005500 PRCV0102
00005510 PRCV0103
00005520 PRCV0104
00005530 PRCV0105
00005540 PRCV0106
00005550 PRCV0107
00005560 PRCV0108

```

```

      404  IDX=N+1-I
           XBAR=XCRV(N)-XCRV(IDX)
           XPRM(I)=XBAR*COS(R)+YCRV(IDX)*SIN(R)
           YPRM(I)=-XBAR*SIN(R)+YCRV(IDX)*COS(R)
           YOPRM(I)=-XBAR*SIN(R)+YOCRV(IDX)*COS(R)
           KCRV=1
           CALL PARLIN(NH1,ECPRM,TOLPRM,ES1PRM,ES2PRM)
           DO 16 JJ=1,NH1
      16  YPRM(JJ)=Y(JJ)
           DO 405 I=1,NH1
           IDX=N+1-I
      405  YCRV(IDX)=XPRM(I)*SIN(R)+YPRM(I)*COS(R)
           KCRV=1
      500  RETURN
           END

```

```

00005570 PRCV0109
00005580 PRCV0110
00005590 PRCV0111
00005600 PRCV0112
00005610 PRCV0113
00005620 PRCV0114
00005630 PRCV0115
00005640 PRCV0116
00005650 PRCV0117
00005660 PRCV0118
00005670 PRCV0119
00005680 PRCV0120
00005690 PRCV0121
00005700 PRCV0122
00005710 PRCV0123

```

C THIS SUBROUTINE CALLS THE STRIP SUBROUTINES TO PAIR LINES WITHOUT
C INFINITE SLOPES.

SUBROUTINE PARLIN(N,EC,TOL1,ES1,ES2)

COMMON /CON2/X(30),Y(30),YORIG(30)

COMMON /CON3/X1(5),Y1(5),Y0(5)

COMMON /CON4/IPAIR,ACC,LIMIT

COMMON /CON5/XPRIM(30),YPRIM(30),YOPRIM(30)

COMMON /CON8/Q

COMMON /CON13/KCRV

IF (KCRV.NE.1) GO TO 11

DO 10 JJ=1,N

X(JJ)=XPRIM(JJ)

Y(JJ)=YPRIM(JJ)

10 YORIG(JJ)=YOPRIM(JJ)

11 CONTINUE

ITRNS=0

1001 CONTINUE

100 IPAIR=0

NM5=N-5

DO 101 I1=1,NM5

IF (I1.EQ.1) CALL PSTPTS(ES1,ES2,EC,TOL1)

IF (I1.EQ.1) GO TO 101

DO 102 I2=1,5

INDEX=I1-1+I2

X1(I2)=X(INDEX)

Y1(I2)=Y(INDEX)

1002 CONTINUE

102 Y0(I2)=YORIG(INDEX)

CALL STRIP1(TOL1)

I1P2=I1+2

ICOUNT=0

DO 103 I3=I1,I1P2

ICOUNT=ICOUNT+1

103 Y(I3)=Y1(ICOUNT)

101 CONTINUE

CALL TRANS1(N,EC,TOL1,ES1,ES2)

00005720 PRLN0001
00005730 PRLN0002
00005740 PRLN0003
00005750 PRLN0004
00005760 PRLN0005
00005770 PRLN0006
00005780 PRLN0007
00005790 PRLN0008
00005800 PRLN0009
00005810 PRLN0010
00005820 PRLN0011
00005830 PRLN0012
00005840 PRLN0013
00005850 PRLN0014
00005860 PRLN0015
00005870 PRLN0016
00005880 PRLN0017
00005890 PRLN0018
00005900 PRLN0019
00005910 PRLN0020
00005920 PRLN0021
00005930 PRLN0022
00005940 PRLN0023
00005950 PRLN0024
00005960 PRLN0025
00005970 PRLN0026
00005980 PRLN0027
00005990 PRLN0028
00006000 PRLN0029
00006010 PRLN0030
00006020 PRLN0031
00006030 PRLN0032
00006040 PRLN0033
00006050 PRLN0034
00006060 PRLN0035
00006070 PRLN0036

ITRNS=ITRNS+1
IF (ITRNS.GE.LIMIT) GO TO 500
IF (IPAIB.NE.0) GO TO 100
500 RETURN
END

00006080 FRLN0037
00006090 FRLN0038
00006100 FRLN0039
00006110 FRLN0040
00006120 FRLN0041

```

C THIS SUBROUTINE CALCULATES THE PAIRED POINTS FOR THE BEGINNING OF THE00006130 FSPT00001
C LINES.
SUBROUTINE FSPTS(ES1,ES2,EC,TOL1)
COMMON /CON2/X(30),Y(30),YORIG(30)
COMMON /CON3/X1(5),Y1(5),Y0(5)
COMMON /CON8/Q
IEC=INT(EC)
1001 CONTINUE
IF (IEC.EQ.1) GO TO 301
IF (IEC.EQ.2) GO TO 302
IF (IEC.EQ.3) GO TO 303
C THIS LOOP ESTABLISHES THE FIRST THREE POINTS FOR FREE ENDS.
301 DO 401 I2=1,5
Y1(I2)=Y(I2)
X1(I2)=X(I2)
X1(I1)=X(I2)
401 Y0(I2)=YORIG(I2)
CALL STRIP1(TOL1)
DO 402 I3=1,3
402 Y(I3)=Y1(I3)
GO TO 101
C THIS LOOP ESTABLISHES THE SECOND AND THIRD POINTS FOR FIXED ENDS.
302 DO 403 I4=1,5
X1(I4)=X(I4)
Y1(I4)=Y(I4)
403 Y0(I4)=YORIG(I4)
CALL STRIP2(TOL1)
Y(2)=Y1(2)
Y(3)=Y1(3)
GO TO 101
C THIS LOOP ESTABLISHES THE SECOND AND THIRD POINTS WITH SLOPE GIVEN.
303 DO 405 I6=1,4
X1(I6)=X(I6)
Y1(I6)=Y(I6)
405 Y0(I6)=YORIG(I6)
Q=ES1

```

```

CALL STRIP3(TOL1)
Y(2)=Y1(2)
DO 406 I7=2,6
  I7H1=I7-1
  X1(I7H1)=X(I7)
  Y1(I7H1)=Y(I7)
  406 Y0(I7H1)=YORIG(I7)
CALL STRIP1(TOL1)
Y(3)=Y1(2)
101 RETURN
END

```

```

00006490 PSPT0037
00006500 PSPT0038
00006510 PSPT0039
00006520 PSPT0040
00006530 PSPT0041
00006540 PSPT0042
00006550 PSPT0043
00006560 PSPT0044
00006570 PSPT0045
00006580 PSPT0046
00006590 PSPT0047

```

```

C THIS SUBROUTINE CHANGES THE ABSCISSAS OF THE END POINTS TO ORIENT THE
C POINTS AS IF THEY ARE AT THE ORIGIN.
SUBROUTINE TRANS1(N,EC,TOL1,ES1,ES2)
COMMON /COM1/SMAT(4,4),T(4),WKAREA(4)
COMMON /COM2/X(30),Y(30),YORIG(30)
COMMON /COM3/X1(5),Y1(5),YO(5)
COMMON /COM4/IFAIR,ACC,LIMIT
INTEGER DEC
1001 CONTINUE
NM1=N-1
NM2=N-2
DO 101 I1=1,5
NM1P1=N-I1+1
X1(I1)=X(NM1P1)-X(NM1P1)
Y1(I1)=Y(NM1P1)
101 YO(I1)=YORIG(NM1P1)
DEC=INT((EC-FLOAT(INT(EC))+.01)*10.)
IF (DEC.EQ.1) GO TO 301
IF (DEC.EQ.2) GO TO 302
IF (DEC.EQ.3) GO TO 303
Y(1)=9999.
GO TO 500
C 301 CALL STRIP1(TOL1)
Y(N)=Y1(1)
Y(NM1)=Y1(2)
Y(NM2)=Y1(3)
GO TO 500
C 302 CALL STRIP2(TOL1)
Y(NM1)=Y1(2)
Y(NM2)=Y1(3)
GO TO 500
C 303 Q=-ES2
CALL STRIP3(TOL1)

```

```

Y(NM1)=Y1(2)
DO 304 I2=1,5
  NM12=N-I2
  X1(I2)=X(NM12)
  Y1(I2)=Y(NM12)
  YO(I2)=YORIG(NM12)
  CALL STRIP1(TOL1)
  Y(NM2)=Y1(2)
304 RETURN
500 END

```

```

00006960 TRNS0037
00006970 TRNS0038
00006980 TRNS0039
00006990 TRNS0040
00007000 TRNS0041
00007010 TRNS0042
00007020 TRNS0043
00007030 TRNS0044
00007040 TRNS0045
00007050 TRNS0046

```

```

C THIS SUBROUTINE CALCULATES THE FIRST THREE PAIRED POINTS (OR CENTER
C PCINT) OF A SERIES OF FIVE DATA POINTS. STRIP1 WOULD BE SUBSEQUENT
C APPLIED TO CONTINUE THE PAIRING PROCESS.
SUBROUTINE STRIP1(TOL1)
COMMON /CON1/SHAT(4,4),T(4),WKAREA(4)
COMMON /CON3/X1(5),Y1(5),YO(5)
COMMON /CON4/IPAIR,ACC,LIMIT
1001 CONTINUE
SHAT(1,1)=5.
DO 101 I1=1,6
SI1=0.
DO 102 I2=1,5
102 SI1=SI1+X1(I2)*I1
I1P1=I1+1
IP (I1.LE.3) GO TO 103
I1M2=I1-2
IP (I1.GE.4) GO TO 104
103 CONTINUE
GO TO 105
103 SHAT(1,I1P1)=SI1
GO TO 101
104 SHAT(4,I1M2)=SI1
GO TO 101
105 CONTINUE
SHAT(2,1)=SHAT(1,2)
SHAT(3,1)=SHAT(1,3)
SHAT(2,2)=SHAT(1,3)
SHAT(4,1)=SHAT(1,4)
SHAT(3,2)=SHAT(1,4)
SHAT(2,3)=SHAT(1,4)
SHAT(3,3)=SHAT(4,2)
SHAT(2,4)=SHAT(4,2)
SHAT(3,4)=SHAT(4,3)
1002 CONTINUE
C

```

```

DO 106 I3=1,4
TI3=0.
DO 107 I4=1,5
IF (X1(I4).EQ.0..AND.I3.EQ.1) TI3=TI3+Y1(I4)
107 IF (X1(I4).NE.0..OR.I3.NE.1) TI3=TI3+Y1(I4)*X1(I4)**(I3-1)
106 T(I3)=TI3
C
C NOW CALL LEQTI FROM THE IMSL LIBRARY TO CALCULATE THE CUBIC
C COEFFICIENTS.
CALL LEQTI (SHAT,1,4,T,0,WKAREA,IER)
YP1=T(1)+T(2)*X1(1)+T(3)*X1(1)**2+T(4)*X1(1)**3
YP2=T(1)+T(2)*X1(2)+T(3)*X1(2)**2+T(4)*X1(2)**3
YP3=T(1)+T(2)*X1(3)+T(3)*X1(3)**2+T(4)*X1(3)**3
DELTA1=YP1-YO(1)
DELTA=ABS(DELTA1)
IF (DELTA.GE.TOL1) Y1(1)=YO(1)+SIGN(TOL1,DELTA1)
IF (ABS(Y1(1)-YP1).GT.ACC) IPAIR=IPAIR+1
IF (DELTA.LT.TOL1) Y1(1)=YP1
DELTA1=YP2-YO(2)
DELTA=ABS(DELTA1)
IF (DELTA.GE.TOL1) Y1(2)=YO(2)+SIGN(TOL1,DELTA1)
IF (ABS(Y1(2)-YP2).GT.ACC) IPAIR=IPAIR+1
IF (DELTA.LT.TOL1) Y1(2)=YP2
DELTA1=YP3-YO(3)
DELTA=ABS(DELTA1)
IF (DELTA.GE.TOL1) Y1(3)=YO(3)+SIGN(TOL1,DELTA1)
IF (ABS(Y1(3)-YP3).GT.ACC) IPAIR=IPAIR+1
IF (DELTA.LT.TOL1) Y1(3)=YP3
RETURN
END

```

```

00007420 STP10037
00007430 STP10038
00007440 STP10039
00007450 STP10040
00007460 STP10041
00007470 STP10042
00007480 STP10043
00007490 STP10044
00007500 STP10045
00007510 STP10046
00007520 STP10047
00007530 STP10048
00007540 STP10049
00007550 STP10050
00007560 STP10051
00007570 STP10052
00007580 STP10053
00007590 STP10054
00007600 STP10055
00007610 STP10056
00007620 STP10057
00007630 STP10058
00007640 STP10059
00007650 STP10060
00007660 STP10061
00007670 STP10062
00007680 STP10063
00007690 STP10064
00007700 STP10065
00007710 STP10066

```

```

C THIS SUBROUTINE CALCULATES THE SECOND AND THIRD PAIRED POINTS FROM
C A SERIES OF FIVE DATA POINTS WITH END POINT FIXED. STRIP1 WOULD BE
C SUBSEQUENTLY APPLIED TO CONTINUE THE PAIRING PROCESS.
      SUBROUTINE STRIP2(TOL1)
      COMMON /CON1/SHAT(4,4),T(4),WKAREA(4)
      COMMON /CON3/1(5),Y1(5),Y0(5)
      COMMON /CON4/1PAIR,ACC,LIMIT
      XREP=X1(1)
      DO 91 L1=1,4
      91 X1(L1)=X1(L1)-XREP
      DO 101 I1=2,6
      101 I1M1=I1-1
      SI1=0.
      DO 102 I2=2,5
      102 SI1=SI1+X1(I2)*I1
      IF (I1M1.LE.3) GO TO 103
      I1M2=I1-2
      IF (I1M1-GE.4) GO TO 104
      101 CONTINUE
      GO TO 105
      103 SHAT(1,I1M1)=SI1
      GO TO 101
      C
      104 SHAT(3,I1M2)=SI1
      GO TO 101
      105 CONTINUE
      SHAT(2,1)=SHAT(1,2)
      SHAT(3,1)=SHAT(1,3)
      SHAT(2,2)=SHAT(1,3)
      SHAT(2,3)=SHAT(3,2)
      C
      DO 106 I3=1,3
      106 TI3=0.
      DO 107 I4=2,5
      107 TI3=TI3+(Y1(I4)-Y1(1))*X1(I4)*I3
      106 T(I3)=TI3

```

```

C NOW CALL LEQT1P FROM THE INSL LIBRARY TO CALCULATE THE COEFFICIENTS.
CALL LEQT1P(SHAT,1,3,4,T,0,WKAREA,IER)
YP2=Y1(1)+T(1)*X1(2)+T(2)*X1(2)**2+T(3)*X1(2)**3
YP3=Y1(1)+T(1)*X1(3)+T(2)*X1(3)**2+T(3)*X1(3)**3
DELTA1=YP2-YO(2)
DELTA=ABS(DELTA1)
IF (DELTA-GE.TOL1) Y1(2)=YO(2)+SIGN(TOL1,DELTA1)
IF (ABS(Y1(2)-YP2)-GT.ACC) IPAIR=IPAIR+1
IF (DELTA-LT.TOL1) Y1(2)=YP2
DELTA1=YP3-YO(3)
DELTA=ABS(DELTA1)
IF (DELTA-GE.TOL1) Y1(3)=YO(3)+SIGN(TOL1,DELTA1)
IF (ABS(Y1(3)-YP3)-GT.ACC) IPAIR=IPAIR+1
IF (DELTA-LT.TOL1) Y1(3)=YP3
DO 92 L2=1,4
92 X1(L2)=X1(L2)+XREF
RETURN
END
00008080 STP20037
00008090 STP20038
00008100 STP20039
00008110 STP20040
00008120 STP20041
00008130 STP20042
00008140 STP20043
00008150 STP20044
00008160 STP20045
00008170 STP20046
00008180 STP20047
00008190 STP20048
00008200 STP20049
00008210 STP20050
00008220 STP20051
00008230 STP20052
00008240 STP20053
00008250 STP20054

```

```

C THIS SUBROUTINE CALCULATES THE PAIRED SECOND DATA POINT FOR A SERIES
C OF FOUR DATA POINTS. STRIP1 WOULD BE SUBSEQUENTLY APPLIED TO
C CONTINUE THE PAIRING PROCESS.
  SUBROUTINE STRIP3(TOL1)
    COMMON /CON3/X1(5),Y1(5),YO(5)
    COMMON /CON4/IPAIR,ACC,LIMIT
    COMMON /CON8/Q
    XREF=X1(1)
    DO 91 L1=1,4
      X1(L1)=X1(L1)-XREF
    DO 101 I1=4,6
      SI1=0.
    DO 102 I2=2,4
      SI1=SI1+X1(I2)*I1
      IF (I1.EQ.4) S4=SI1
      IF (I1.EQ.5) S5=SI1
      IF (I1.EQ.6) S6=SI1
    DO 103 I3=2,3
      TI3=0.
    DO 104 I4=2,4
      TI3=TI3+(Y1(I4)-Q*X1(I4)-Y1(1))*X1(I4)**I3
      IF (I3.EQ.2) T2=TI3
      IF (I3.EQ.3) T3=TI3
      A2=(T2*S6-T3*S5)/(S4*S6-S5**2)
      A3=(T3*S4-T2*S5)/(S4*S6-S5**2)
      YP2=Y1(1)+Q*X1(2)+A2*X1(2)**2+A3*X1(2)**3
      DELTA1=YP2-YO(2)
      DELTA=ABS(DELTA1)
      IF (DELTA.GE.TOL1) Y1(2)=YO(2)+SIGN(TOL1,DELTA1)
      IF (ABS(Y1(2)-YP2).GT.ACC) IPAIR=IPAIR+1
      IF (DELTA.LT.TOL1) Y1(2)=YP2
    DO 92 L2=1,4
      X1(L2)=X1(L2)+XREF
    RETURN
  END

```

APPENDIX G

The following is an example of a typical bow section for a ship with a bulbous bow mounted sonar dome. Although the section is for a hypothetical ship it illustrates the capability of the computer program to fair, spline and interpolate a curve having an infinite slope at an end point.

In the CRV matrix shown, the items written in block numbers are input values while those in italics are values calculated by the program. The elements left blank were not used in this example. The values of tolerance, accuracy and limit are:

TOL = 1.00

ACC = 0.05

LIMIT = 10

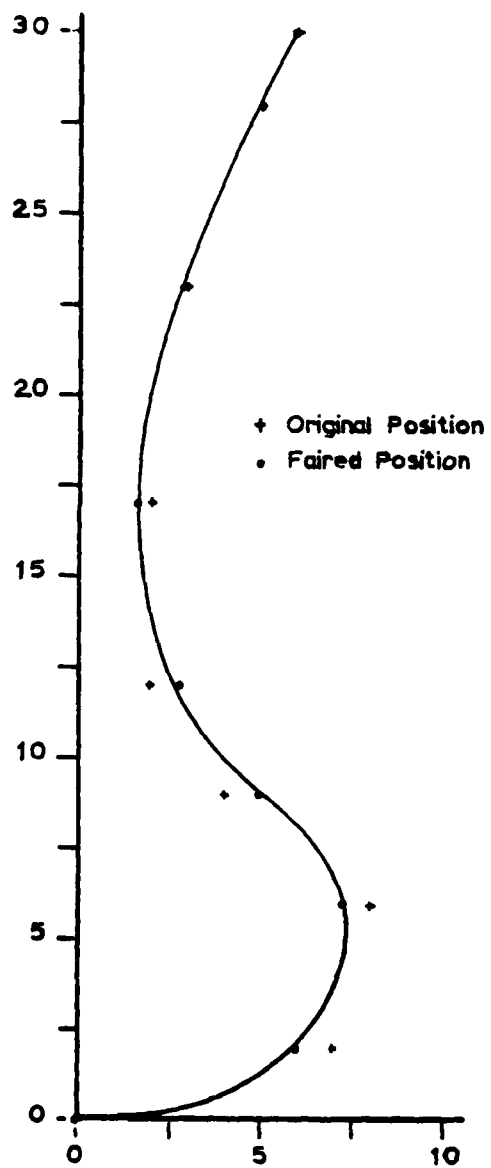


Figure G.1.- Bow Section (Bulbous)

CRV MATRIX

	1	2	3	4	5	6	7
1	0.0	0.0	0.000	0.0	0.333	-0.480	1.570
2	2.0	7.0	6.000	0.0	0.530	-0.675	0.802
3	6.0	8.0	7.301	0.0	0.394	-0.122	-0.279
4	9.0	4.0	4.999	0.0	-0.156	0.160	-0.776
5	12.0	2.0	2.805	0.0	-0.242	0.241	-0.473
6	17.0	2.0	1.609	0.0	-0.218	0.149	0.002
7	23.0	3.0	2.934	0.0	-0.041	0.009	0.366
8	28.0	5.0	5.086	0.0	0.006	-0.003	0.416
9	30.0	6.0	5.954	0.0			0.407
.							
.							
.							
31	1.0	9.0	0.0	1.1			
32	2.0	4.1	0.0				

The resulting interpolated values for increments of $\Delta X = (30-0)/20$ are as follows:

<u>X</u>	<u>Y</u>	<u>DY/DX</u>
0.0	0.0000	*****
1.5	5.4206	1.3048
3.0	6.8443	0.6555
4.5	7.4269	0.1361
6.0	7.3018	-0.2867
7.5	6.4713	-0.8319
9.0	4.9988	-0.9816
10.5	3.7259	-0.7238
12.0	2.8047	-0.5114
13.5	2.1714	-0.3371
15.0	1.7846	-0.1815
16.5	1.6191	-0.0415
18.0	1.6542	0.0867
19.5	1.8721	0.2010
21.0	2.2473	0.2956
22.5	2.7465	0.3656
24.0	3.3316	0.4116
25.5	3.9724	0.4396
27.0	4.6403	0.4476
28.5	5.3051	0.4368
30.0	5.9543	0.4308

Figure G.1 shows the curve generated as a result of fairing the given data points.