NREL/TP-257-4099 • UC Category: 261 • DE92010574

A Surface Definition Code for Turbine Blade Surfaces

S. L. Yang
Michigan Technological University
Houghton, Michigan

D. Oryang
M. J. Ho
Tuskegee University
Tuskegee, Alabama

NREL technical monitor: P. K. C. Tu

National Renewable Energy Laboratory
(formerly the Solar Energy Research Institute)
1617 Cole Boulevard
Golden, Colorado 80401-3393
A Division of Midwest Research Institute
Operated for the U.S. Department of Energy
under Contract No. DE-AC02-83CH10093

Prepared under Subcontract No. XX-7-06092-2

May 1992

On September 16, 1991, the Solar Energy Research Institute was designated a national laboratory, and its name was changed to the National Renewable Energy Laboratory.

NOTICE

Printed in the United States of America
Available from:
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road
Springfield, VA 22161

Price: Microfiche A01
Printed Copy A10

Codes are used for pricing all publications. The code is determined by the number of pages in the publication. Information pertaining to the pricing codes can be found in the current issue of the following publications which are generally available in most libraries: *Energy Research Abstracts (ERA)*; *Government Reports Announcements* and Index (*GRA* and I); *Scientific and Technical Abstract Reports (STAR)*; and publication NTIS-PR-360 available from NTIS at the above address.

ACKNOWLEDGMENT

This research is supported by the National Renewable Energy Laboratory (NREL, formerly Solar Energy Research Institute) under subcontract No. XX-7-06092-2 and has been monitored by NREL Project Manager Jim L. Tangler and Dr. Peter K. C. Tu. The authors would like to express their sincere gratitude to George N. Scott of NREL for many helpful discussions and suggestions.

ABSTRACT

A SURFACE GENERATION CODE FOR TURBINE BLADE SURFACES

A numerical interpolation scheme has been developed for generating the three-dimensional geometry of wind turbine blades. The numerical scheme consists of (1) creating the frame of the blade through the input of two or more airfoils at some specific spanwise stations and then scaling and twisting them according to the prescribed distributions of chord, thickness, and twist along the span of the blade; (2) transforming the physical coordinates of the blade frame into a computational domain that complies with the interpolation requirements; and finally (3) applying the bi-tension spline interpolation method, in the computational domain, to determine the coordinates of any point on the blade surface.

Detailed descriptions of the overall approach to and philosophy of the code development are given along with the operation of the code.

To show the usefulness of the bi-tension spline interpolation code developed, two examples are given, namely CARTER and MICON blade surface generation. Numerical results are presented in both graphic and tabular data forms. The solutions obtained in this work show that the computer code developed can be a powerful tool for generating the surface coordinates for any three-dimensional blade.

TABLE OF CONTENTS

page

LIST OF FIGURES vi

LIST OF TABLES x

NOMENCLATURE xi

CHAPTER

1. INTRODUCTION 1

2. TENSION SPLINE INTERPOLATION 5

2.1 Tension Spline for Curves 6

2.2 Tension Spline for Surfaces 11

3. CREATING THE BLADE FRAME 16

3.1 Choice of Span Stations of the Blade at which to Place Airfoils 16

3.2 Choice of Airfoils 17

3.3 Scaling the Airfoils 18

3.3.1 Scaling the Airfoils to the Required Maximum Thickness-to-Chord Ratio 18

3.3.2 Scaling the Airfoils to the Required Chord Length 18

3.4 Twisting the Airfoils 29

4. TRANSFORMATION OF THE BLADE FRAME 24

4.1 Calculation of the Normalized Z-Coordinate 25

4.2 Computation and Normalization of the Polygonal Arc length for Each Input Airfoil 25

4.3 Interpolation of New X and Y Coordinates for Each Airfoil 26

4.4 Defining and Using the Computational Domain Coordinates 27

TABLE OF CONTENTS (Continued)

page

CHAPTER

5. PROGRAM DESCRIPTION 32

5.1 Program Structure 32

5.1.1 Input Module 33

5.1.2 Blade Frame Creation Module 34

5.1.3 Transformation Module 34

5.1.4 Partial Derivatives Module 35

5.1.5 Interpolation Module 36

5.1.6 Output Module 37

5.2 Subroutine Functions 37

5.3 Critical Variable Definitions 41

6. HOW TO RUN THE CODE 45

6.1 Program Installation 45

6.2 Input Data Files 47

6.2.1 Airfoil Data File 47

6.2.2 Chord/Twist/Thickness Distribution Data File 50

6.3 Running the Program 52

6.3.1 Input Options 52

6.3.2 Output Options 58

6.4 Sample Run of the Program 60

TABLE OF CONTENTS (Concluded)

page

CHAPTER

7. RESULTS AND DISCUSSIONS 72

7.1 CARTER Blade 73

7.2 MICON Blade 93

8. CONCLUSIONS 112

REFERENCES 114

APPENDIX

A PROGRAM LISTING A-1

B DERIVATION OF EQUATIONS (2.2) and (2.5) B-1

C DERIVATION OF EQUATION (2.11) C-1

LIST OF FIGURES

Figure		Page
1	Expanded Thin-Airfoil Family	2
2	Data Points in the (x,y) Plane Representing a Curve on Which a 1-D Tension Spline is to be Applied	7
3	Chord-Normalized Input Airfoils at Respective Span Stations	21
4	Input Airfoils That Have Been Scaled and Offset, and Then Placed at Their Respective Span Stations	22
5	Input Airfoils That Have Been Scaled, Offset, and Twisted, and Then Placed at Their Respective Span Stations	23
6	Variation in Grid Spacing with Three Different Stretching Parameters for An Airfoil Represented by 65 Points	28
7	Computational Domain with X as the Dependent Variable	29
8	Computational Domain with Y as the Dependent Variable	31
9	Order of Access of Modules in the BLADE Code	32
10	Structure of the Input Module	33
11	Structure of the Transformation Module	34
12	Structure of the Partial Derivatives Module	35
13	Structure of the Interpolation Module	36
14	Structure of the Output Module	38
15	Sample Input Airfoil Data File (Example of S805A7.DAT)	48
16	Sample Chord/Twist/Thickness Distribution Data File (Example of CTE2.DAT)	51

Figure		Page
17	Sample Input Data File (Example of JAYIN.DAT)	54
18	Data Used to Make the Sample Run	60
19	Concentric Plot of the Input Airfoils that Form the CARTER Blade Frame	76
20	Concentric Plot of the Airfoils Forming the Generated CARTER Blade	77
21	Perspective View of the Generated CARTER Blade	78
22	Top View of the Generated CARTER Blade	79
23	Front View of the Generated CARTER Blade	80
24	Twist Distribution Along the Span of the CARTER Blade	81
25	Chord Distribution Along the Span of the CARTER Blade	83
26	Maximum Thickness Distribution Along the Span of the CARTER Blade	84
27	Span-Normalized Airfoil at a Z/span Coordinate of 0.20 on the Generated CARTER Blade	85
28	Span-Normalized Airfoil at a Z/span Coordinate of 0.55 on the Generated CARTER Blade	86
29	Span-Normalized Airfoil at a Z/span Coordinate of 0.90 on the Generated CARTER Blade	87
30	Span-Normalized Airfoil Data for Airfoil at a Z/span Coordinate of 0.55 on the Generated CARTER Blade	88
31	Chord-Normalized Airfoil at a Z/span Coordinate of 0.20 on the Generated CARTER Blade	89
32	Chord-Normalized Airfoil at a Z/span Coordinate of 0.55 on the Generated CARTER Blade	90

Figure		Page
33	Chord-Normalized Airfoil at a Z/span Coordinate of 0.90 on the Generated CARTER Blade	91
34	Chord-Normalized Airfoil Data for Airfoil at a Z/span Coordinate of 0.55 on the Generated CARTER Blade, with the Twist and Offset Removed	92
35	Concentric Plot of the Input Airfoils that Form the MICON Blade Frame	96
36	Concentric Plot of the Airfoils Forming the Generated MICON Blade	97
37	Perspective View of the Generated MICON Blade	98
38	Top View of the Generated MICON Blade	99
39	Front View of the Generated MICON Blade	100
40	Twist Distribution Along the Span of the MICON Blade	101
41	Chord Distribution Along the Span of the MICON Blade	102
42	Maximum Thickness Distribution Along the Span of the MICON Blade	103
43	Span-Normalized Airfoil at a Z/span Coordinate of 0.20 on the Generated MICON Blade	104
44	Span-Normalized Airfoil at a Z/span Coordinate of 0.55 on the Generated MICON Blade	105
45	Span-Normalized Airfoil at a Z/span Coordinate of 0.90 on the Generated MICON Blade	106
46	Span-Normalized Airfoil Data for Airfoil at a Z/span Coordinate of 0.55 on the Generated MICON Blade	107
47	Chord-Normalized Airfoil at a Z/span Coordinate of 0.20 on the Generated MICON Blade	108

LIST OF FIGURES (Concluded)

Figure | Page

48 Chord-Normalized Airfoil at a Z/span Coordinate of 0.55 on the Generated MICON Blade 109

49 Chord-Normalized Airfoil at a Z/span Coordinate of 0.90 on the Generated MICON Blade 110

50 Chord-Normalized Airfoil Data for Airfoil at a Z/span Coordinate of 0.55 on the Generated MICON Blade, with the Twist and Offset Removed 111

LIST OF TABLES

Table		Page
1	Input Data to Form the CARTER Blade Frame	74
2	Input Chord/Twist/Thickness Data for the CARTER Blade	75
3	Input Data to Form the MICON Blade Frame	94
4	Input Chord/Twist/Thickness Data for the MICON Blade	95

NOMENCLATURE

$A_{i,k}$	vector containing the four 1-D spline coefficients defined by equation (2.5)
$A_{i,j,k,l}$	matrix containing the sixteen 2-D spline coefficients defined by equation (2.19)
B	vector appearing in equation (2.14)
C	matrix defined in equations (2.6) and (2.20)
CL	chord length of an airfoil
CM	coefficients matrix appearing in equation (2.14)
$f_{i,j}$	surface tension spline function defined by equation (2.18)
g_i	curve tension spline function defined by equation (2.2)
K_i	degrees of freedom characteristic vector for a linear element of the 1-D computational domain, as defined by equation (2.7)
$K_{i,j}$	degrees of freedom characteristic matrix for a rectangular element of the 2-D computational domain, as defined by equation (2.21)
m	total number of airfoils representing the blade frame
n	total number of points representing the airfoil surface
OS_X, OS_Y	airfoil twist center
P	vector appearing in equation (2.14)
$p_{i,j}$	partial derivative defined in equation (2.24)
$q_{i,j}$	partial derivative defined in equation (2.24)
$r_{i,j}$	partial derivative defined in equation (2.24)
s_j	term defined in equation (2.36)
S_i	polygonal arc length as defined in equation (4.3)

$\overline{S}_i$	normalized polygonal arc length as defined in equation (4.4)
t_i	term defined in equation (2.35)
$u_{i,j}$	term defined in equation (2.24)
v_i	term defined in equations (2.8) and (2.22)
w_j	term defined in equation (2.23)
X_i, x_i	physical domain X-coordinate
Y_i, y_i	physical domain Y-coordinate
Z_j, z_j	physical domain Z-coordinate

Greek Symbols

α_i	tension parameter in the ξ direction
β_j	tension parameter in the η direction
η_j	computational domain coordinate defined by equation (4.1)
τ	stretching parameter
Θ_j	twist angle of airfoil
ξ_i	stretching function and computational domain coordinate defined by equation (4.8)
$\overline{\xi}_i$	equally spaced polygonal arc lengths, defined by equation (4.7)
Φ_k, Φ_l	interpolation functions in the ξ – and η – directions respectively, as represented by equation (2.3)

Subscripts

i,j	denote the grid indices in (1) the $\xi-$ and $\eta-$ directions respectively for the computational domain; and (2) the chordwise- and spanwise- directions respectively for the physical domain
k,l	denote the position placement of coefficients in the interpolation functions vectors, the spline coefficients matrices and the degrees of freedom vectors and matrices
imax,jmax	denote the total number of points representing a particular input airfoil, and the total number of airfoils input to form the blade frame, respectively
mid	denotes the grid index in the $\xi-$ direction which represents the leading edge point on a particular airfoil

CHAPTER 1

INTRODUCTION

In 1983, the Solar Energy Research Institute (SERI), in conjunction with Airfoils, Incorporated, began the development of some special-purpose airfoil families for horizontal-axis wind turbines (HAWTs) [1]. Specific objectives for this effort were to

(1) Restrain peak power and reduce sensitivity to insect accumulation

(2) Lower blade oscillatory loads for better rotor fatigue life

(3) Improve annual energy output at wind sites having mean annual wind speeds in the 10- to 14-mile-per-hour range.

Three special-purpose airfoil families were developed using the Eppler airfoil design and analysis code [2]. Each family consists of (1) a primary airfoil designed for the 75% radial position and (2) two secondary airfoils, designed to complement the primary airfoil that are located inboard and outboard at radial stations of 30% to 40%, and 95% [1]. Additional airfoils were derived, through interpolation and extrapolation, to aid the manufacturer in designing a blade using these airfoils. Figure 1 shows the expanded thin-airfoil family that is one outcome of this effort.

However, because of the highly twisted and tapered nature of the wind turbine blades, it is not always possible to generate a smooth surface for representing the desired blade through an interpolation based on the above limited available airfoil data. Therefore, an accurate and reliable interpolation technique becomes a very important tool in fabrication processes. Through this interpolation, it is possible to define the whole surface of the blade, and in so doing generate any number of airfoils to represent the blade. The coordinates of the generated airfoils can then be used to make templates for fabricating the blade.

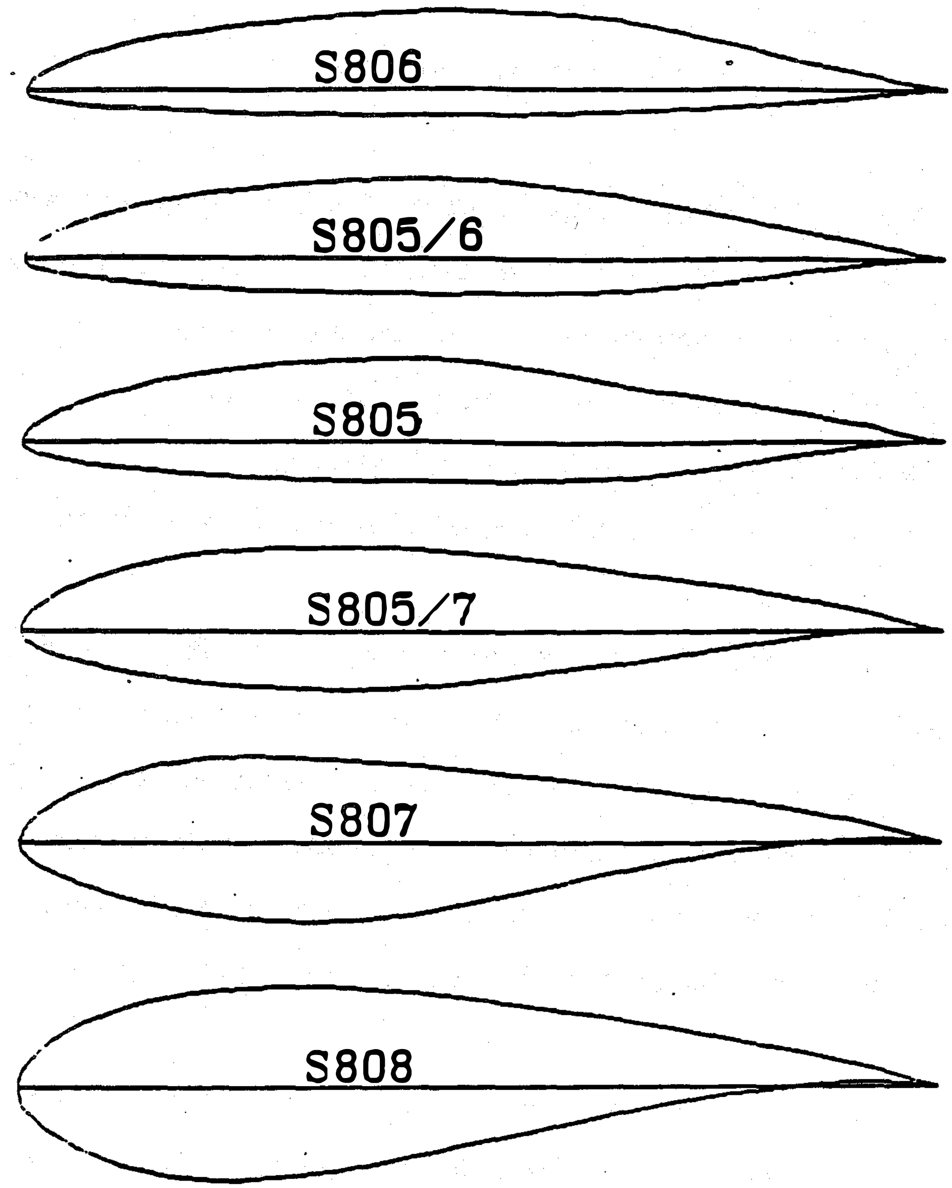


Figure 1 Expanded Thin-Airfoil Family

An early attempt at this task reported that, because of the tapered and twisted nature of the blade, the data generated by interpolation could produce wavy surfaces toward the root of the blade [3]. Another approach, using the naval architecture versions of interpolation for ships and water-planes, avoids double concave/convex curvature, but the data generated are difficult to mold [4]. Most recently, Andrews and Van Doren [5] used one-dimensional tension spline interpolation in both the chordwise and spanwise directions, to generate a blade surface. They reported that the generated geometry is highly dependent on the tension parameter.

The objective of this study is to develop a FORTRAN computer code for generating a three-dimensional (3D) wind turbine blade through the input of two or more airfoils at some specific spanwise stations as the basis for interpolation. In addition, the generated 3D blade geometry should also satisfy the prescribed distributions of chord, thickness, and twist along the span of the blade. In this work, an algorithm used to define the smooth surface of a twisted and tapered wind turbine blade by using the bi-tension spline interpolation method [6, 7] is presented. The numerical algorithm consists of (1) creating the frame of a blade, (2) transforming the physical coordinates of the blade frame into computational coordinates, and finally (3) applying spline-under-tension interpolation to determine the coordinates of any point on the blade surface.

To show the versatility and usefulness of the tension spline interpolation code, two examples are given. Numerical results are presented in both graphic and tabular data forms. The solutions obtained in this work show that the computer code can be a powerful tool for generating the surface coordinates for any 3D blade.

In Chapter 2, the tension spline interpolation method for curves and surfaces is given. Chapter 3 discusses the blade frame creation process, which is necessary as the basis for the interpolation. To comply with the surface interpolation requirements and to facilitate the interpolation, a transformation scheme is developed and is presented in Chapter 4. In Chapter

5, the overall organization of the code and the program modules (subroutines) are given. Chapter 6 describes the operation of the code, including installation and execution of the code, input requirements, and output options. The results of applying the code to two sample problems and the numerical solutions are given in Chapter 7. Finally, Chapter 8 presents the conclusions of this research. The listing of the code is given in Appendix A.

CHAPTER 2

TENSION SPLINE INTERPOLATION

In general, there are two approaches to fitting a curve or surface to a given set of data [8]. One, called least-square fit, uses an approximating function to generate the curve or surface that does not necessarily pass through the given data points. Another, called exact fit, requires that the approximating function pass exactly through the given data points. In this work, the latter approach is used.

One available technique for solving interpolation problems is the cubic spline function [9]. The curve (or surface) generated by the cubic spline function passes through the given data points; has continuous slope or curvature; and, in general, has the local properties of smoothness. It seems that the cubic spline function is a very useful interpolation tool. However, because of the nature of the cubic spline function (a cubic polynomial), it occasionally produces some unwanted inflection points in the curve, especially at regions where the curvatures are negative [10]. It is therefore necessary to provide some way to remove these unwanted inflection points in the interpolated curve while still retaining the merits of cubic spline interpolation, i.e., smoothness and continuity. One promising method for overcoming this flaw is the "tension spline" function developed by Schweikert [10] for curves and extended by Späth [11] for surfaces.

Schweikert proposed that in order to remove the unwanted inflection points in the interpolated curve, the hyperbolic functions (exponential function) should be used instead of the cubic polynomial. Physically, the interpolating function used in his paper represents the solution of a simply supported beam in tension. It is from this that the hyperbolic interpolation function derived the name of "tension" spline [10]. In this chapter, the equations used to describe the tension spline function for curves are given first. These are then extended for surface interpolation.

2.1 Tension Spline for Curves

Consider n data points (x_i,y_i) in the (x,y) plane, where $1 \leq i \leq n$ and $a \leq x_i \leq b$; see Figure 2. The tension spline $g(x)$ used to interpolate the given data on the partition

$$a = x_1 < x_2 < \ldots\ldots < x_i < \ldots\ldots < x_{n-1} < x_n = b \tag{2.1}$$

of the interval {a, b} is expressed as (see Appendix B for derivation)

$$g_i(x) = \sum_{k=1}^{4} A_{i,k}\Phi_k(x) \tag{2.2}$$

for each subinterval i, where $x_i \leq x \leq x_{i+1}$. The functions Φ_k in (2.2) are defined by

$$\Phi_1(x,\alpha_i,x_i,x_{i+1}) = x - x_i$$

$$\Phi_2(x,\alpha_i,x_i,x_{i+1}) = x_{i+1} - x$$

$$\Phi_3(x,\alpha_i,x_i,x_{i+1}) = \psi(x - x_i,\alpha_i,x_i,x_{i+1})$$

$$\Phi_4(x,\alpha_i,x_i,x_{i+1}) = \psi(x_{i+1} - x,\alpha_i,x_i,x_{i+1}) \tag{2.3}$$

where

$$\psi(x,\alpha_i,x_i,x_{i+1}) = \frac{\Delta x_i \sinh(\alpha_i x) - x \sinh(\alpha_i \Delta x_i)}{\sinh(\alpha_i \Delta x_i) - \alpha_i \Delta x_i} \tag{2.4}$$

and

$$\Delta x_i = x_{i+1} - x_i$$

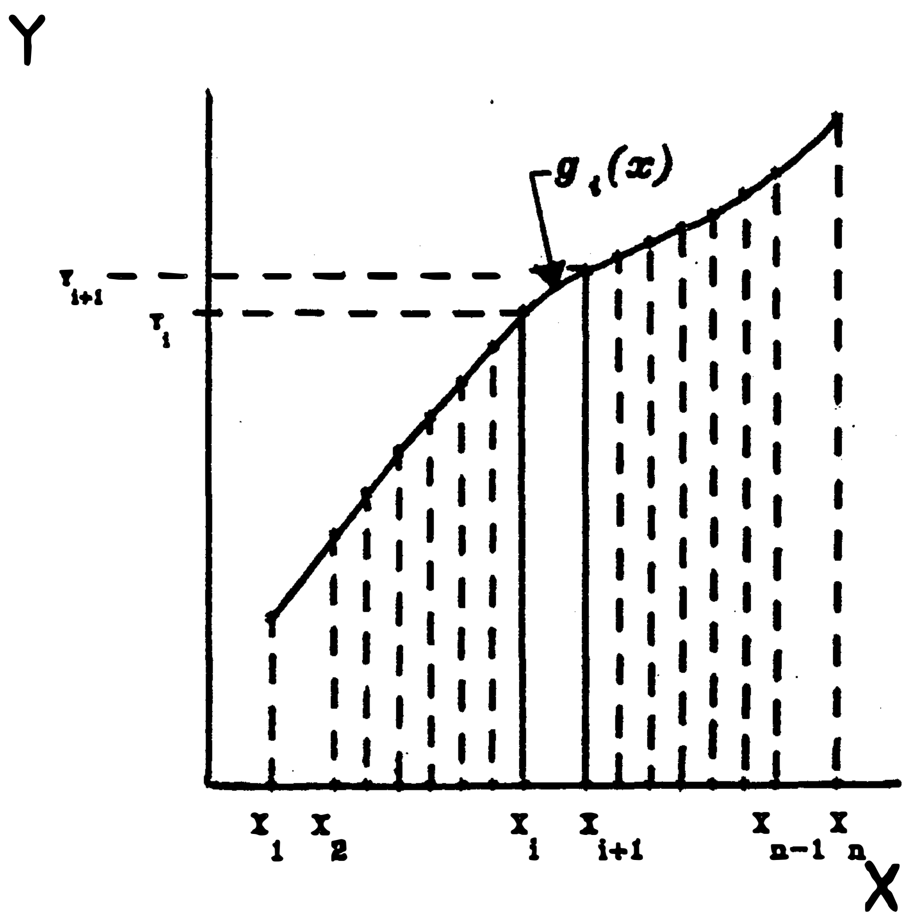


Figure 2 Data Points in the (x,y) Plane Representing a Curve on Which a 1-D Tension Spline is to be Applied

The four coefficients $A_{i,k}$ in (2.2) represent the spline coefficients for the interval $x_i \le x \le x_{i+1}$ and are expressed as (see Appendix B)

$$A_{i,k} = C(\Delta x_i, v_i) K_i \tag{2.5}$$

where

$$C(\Delta x_i, v_i) = \begin{bmatrix} 0 & 0 & \frac{1}{\Delta x_i} & 0 \\ \frac{1}{\Delta x_i} & 0 & 0 & 0 \\ -\frac{1}{\Delta x_i}\frac{1}{1-v_i} & \frac{-1}{1-v_i^2} & \frac{1}{\Delta x_i}\frac{1}{1-v_i} & \frac{-v_i}{1-v_i^2} \\ \frac{1}{\Delta x_i}\frac{1}{1-v_i} & \frac{v_i}{1-v_i^2} & -\frac{1}{\Delta x_i}\frac{1}{1-v_i} & \frac{1}{1-v_i^2} \end{bmatrix} \tag{2.6}$$

$$K_i = \begin{bmatrix} y_i \\ y'_i \\ y_{i+1} \\ y'_{i+1} \end{bmatrix} \tag{2.7}$$

and

$$v_i = \left[\frac{d}{dx} \psi(x - x_i, \alpha_i, x_i, x_{i+1}) \right]_{x = x_{i+1}} \tag{2.8}$$

The coefficient α appearing in the above equations is called the tension factor and is a constant in each subinterval i. Superscript ' in (2.7) denotes the derivative of y with respect to x. It can be shown [5] that

$$\psi(x, 0, x_i, x_{i+1}) = \lim_{\alpha_i \to 0} \psi(x, \alpha_i, x_i, x_{i+1}) = \frac{x^3 - x}{\Delta x_i^2} \tag{2.9}$$

corresponds to the cubic spline interpolation. Also, notice that if the tension factor becomes very large, the interpolating curve is degraded to a piecewise linear interpolation.

The values of y'_i, appearing in (2.7), are determined by the following procedures:

(1) For i = 1, the boundary value y'_1 is determined by the forward difference as

$$y'_1 = \frac{y_2 - y_1}{x_2 - x_1} \tag{2.10}$$

(2) For $2 \le i \le n-1$, derivatives y'_i are determined by using the curvature continuity condition to set up a system consisting of the following n-2 equations (see Appendix C for derivation):

$$t_{i-1} y'_{i-1} + (t_{i-1} v_{i-1} + t_i v_i) y'_i + t_i y'_{i+1} = t_{i-1}(1 + v_{i-1}) \frac{\Delta y_{i-1}}{\Delta x_{i-1}} + t_i (1 + v_i) \frac{\Delta y_i}{\Delta x_i} \tag{2.11}$$

where

$$t_i = \frac{\alpha_i^2 \Delta x_i \sinh(\alpha_i \Delta x_i)}{(v_i^2 - 1)(\sinh(\alpha_i \Delta x_i) - \alpha_i \Delta x_i)} \tag{2.12}$$

for $1 \le i \le n-1$ and v_i as in (2.8).

(3) For i = n, the boundary value y'_n is solved for by the backward difference as

$$y'_n = \frac{y_n - y_{n-1}}{x_n - x_{n-1}} \tag{2.13}$$

If the curvature at the boundary points are specified as zero, the corresponding spline is called a natural spline [12].

The system of equations (2.11) in matrix form (with the boundary derivative values included as known parameters) is

$$[CM][P] = [B] \tag{2.14}$$

here

$$CM = \begin{bmatrix} b_2 & c_2 & 0 & 0 & 0 & \dots & 0 & 0 & 0 & 0 \\ a_3 & b_3 & c_3 & 0 & 0 & \dots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & a_i & b_i & c_i & \dots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \dots & 0 & a_{n-2} & b_{n-2} & c_{n-2} \\ 0 & 0 & 0 & 0 & 0 & \dots & 0 & 0 & a_{n-1} & b_{n-1} \end{bmatrix}$$

$$P = \begin{bmatrix} y'_2 \\ y'_3 \\ \vdots \\ y'_i \\ \vdots \\ y'_{n-2} \\ y'_{n-1} \end{bmatrix}; \qquad B = \begin{bmatrix} B_2 \\ B_3 \\ \vdots \\ B_i \\ \vdots \\ B_{n-2} \\ B_{n-1} \end{bmatrix}$$

the coefficients a, b, and c in matrix CM are

$$a_i = t_{i+1}$$

$$b_i = t_i v_i + t_{i+1} v_{i+1}$$

$$c_i = t_{i+1}$$

and the coefficients in vector B are

$$B_2 = t_1(v_1 + 1)\frac{y_2 - y_1}{x_2 - x_1} + t_2(v_2 + 1)\frac{y_3 - y_2}{x_3 - x_2} + y'_1 t_1$$

$$B_i = t_i(v_i + 1)\frac{y_{i+1} - y_i}{x_{i+1} - x_i} + t_{i+1}(v_{i+1} + 1)\frac{y_{i+2} - y_{i+1}}{x_{i+2} - x_{i+1}}; \qquad \textit{for} \quad 3 \le i \le n-2$$

$$B_{n-1} = t_{n-2}(v_{n-2} + 1)\frac{y_{n-1} - y_{n-2}}{x_{n-1} - x_{n-2}} + t_{n-1}(v_{n-1} + 1)\frac{y_n - y_{n-1}}{x_n - x_{n-1}} + y'_n t_{n-1}$$

The coefficient matrix CM of system (2.14) is a tridiagonal, symmetric, and diagonally dominant matrix with positive diagonal elements. It can be solved by the efficient Thomas algorithm [13] for the values of y'_i $(2 \leq i \leq n-1)$ contained in P.

When the derivatives y'_i are known, the four spline coefficients $A_{i,k}$ for each subinterval i can be determined from (2.5). Equation (2.2) is then used to determine the interpolated y-value for any x in any subinterval of {a, b}.

The curve represented by the given n data points is now defined. Depending on the value chosen for the tension factor, α, the interpolating curve can be varied between a cubic polynomial and a piecewise linear polynomial.

2.2 Tension Spline for Surfaces

To extend the tension spline of curves to that for surface interpolation is straightforward. Let $u_{i,j} = u(\xi_i, \eta_j)$, i = 1, 2,....., n, and j = 1, 2,....., m denote the values of a function given at the nodes of a rectangular computational domain grid with

$$\xi_1 < \xi_2 < \ldots\ldots < \xi_n, \qquad n \geq 2 \tag{2.16}$$

and

$$\eta_1 < \eta_2 < \ldots\ldots < \eta_m, \qquad m \geq 2 \tag{2.17}$$

in the (ξ, η) plane. The surface tension spline f(ξ, η) used to interpolate the given data on the region described in (2.16) and (2.17) is expressed as [11,12]

$$f_{i,j}(\xi,\eta) = \sum_{k=1}^{4}\sum_{l=1}^{4} A_{i,j,k,l}\Phi_k(\xi,\alpha_i,\xi_i,\xi_{i+1})\Phi_l(\eta,\beta_j,\eta_j,\eta_{j+1}) \tag{2.18}$$

for each rectangle $\{\xi_i \leq \xi \leq \xi_{i+1}; \eta_j \leq \eta \leq \eta_{j+1}\}$.

The nature of the spline function is such that the computational value of $f_{i,j}$ at the nodes of the rectangular grid is $u_{i,j}$, i.e.,

$$f_{i,j}(\xi_i,\eta_j) = u_{i,j}$$

The interpolation functions Φ in (2.18) take the same form as (2.3) and (2.4). The coefficients α and β in (2.18) are the tension parameters in the ξ and η directions, respectively.

The coefficient matrix $A_{i,j,k,l}$ (for k, l = 1, 2, 3, 4) in (2.18) is a 4-by-4 matrix at each rectangle and is formed by the matrix products

$$A_{i,j,k,l} = C(\Delta\xi_i, v_i) K_{i,j} [C(\Delta\eta_j, w_j)]^T \tag{2.19}$$

where

$$C(g,h) = \begin{bmatrix} 0 & 0 & \frac{1}{g} & 0 \\ \frac{1}{g} & 0 & 0 & 0 \\ -\frac{1}{g}\frac{1}{1-h} & \frac{-1}{1-h^2} & \frac{1}{g}\frac{1}{1-h} & \frac{-h}{1-h^2} \\ \frac{1}{g}\frac{1}{1-h} & \frac{h}{1-h^2} & -\frac{1}{g}\frac{1}{1-h} & \frac{1}{1-h^2} \end{bmatrix} \tag{2.20}$$

$$K_{i,j} = \begin{bmatrix} u_{i,j} & q_{i,j} & u_{i,j+1} & q_{i,j+1} \\ p_{i,j} & r_{i,j} & p_{i,j+1} & r_{i,j+1} \\ u_{i+1,j} & q_{i+1,j} & u_{i+1,j+1} & q_{i+1,j+1} \\ p_{i+1,j} & r_{i+1,j} & p_{i+1,j+1} & r_{i+1,j+1} \end{bmatrix} \tag{2.21}$$

$$v_i = \left[\frac{d}{d\xi} \psi(\xi - \xi_i, \alpha_i, \xi_i, \xi_{i+1}) \right]_{\xi = \xi_{i+1}} \tag{2.22}$$

$$w_j = \left[\frac{d}{d\eta} \psi(\eta - \eta_j, \beta_j, \eta_j, \eta_{j+1}) \right]_{\eta = \eta_{j+1}} \tag{2.23}$$

$$\Delta\xi_i = \xi_{i+1} - \xi_i$$

$$\Delta\eta_j = \eta_{j+1} - \eta_j$$

and the superscript T in (2.19) denotes the transpose. The elements $u_{i,j}$, $p_{i,j}$, $q_{i,j}$, and $r_{i,j}$ in matrix $K_{i,j}$ of (2.21) are the abbreviations of the following function values and its partial derivatives:

$$u_{i,j} = u(\xi_i, \eta_j)$$

$$p_{i,j} = \frac{\partial u(\xi_i, \eta_j)}{\partial \xi}$$

$$q_{i,j} = \frac{\partial u(\xi_i, \eta_j)}{\partial \eta}$$

$$r_{i,j} = \frac{\partial^2 u(\xi_i, \eta_j)}{\partial \xi \partial \eta} \tag{2.24}$$

The partial derivatives $p_{i,j}$, $q_{i,j}$, and $r_{i,j}$ are determined as follows.

For $1 \le j \le m$, the boundary values $p_{1,j}$ and $p_{n,j}$ are solved for by forward and backward difference respectively according to

$$p_{1,j} = \frac{u_{2,j} - u_{1,j}}{\xi_2 - \xi_1} \tag{2.25}$$

$$p_{n,j} = \frac{u_{n,j} - u_{n-1,j}}{\xi_n - \xi_{n-1}} \tag{2.26}$$

For $1 \le i \le n$, the boundary values $q_{i,1}$ and $q_{i,m}$ are solved for by forward and backward difference respectively as

$$q_{i,1} = \frac{u_{i,2} - u_{i,1}}{\eta_2 - \eta_1} \tag{2.27}$$

$$q_{i,m} = \frac{u_{i,m} - u_{i,m-1}}{\eta_m - \eta_{m-1}} \tag{2.28}$$

For $j = 1$ and m, the boundary values $r_{1,j}$ and $r_{n,j}$ are solved for by forward and backward difference respectively as

$$r_{1,j} = \frac{q_{2,j} - q_{1,j}}{\xi_2 - \xi_1} \tag{2.29}$$

$$r_{n,j} = \frac{q_{n,j} - q_{n-1,j}}{\xi_n - \xi_{n-1}} \tag{2.30}$$

This completes the evaluation of all the required partial derivatives at the boundary nodes. In order to determine the partial derivative values of the internal nodes of the rectangular grid, the curvature continuity condition is used to set up the following 2n + m + 2 linear systems of coupling equations:

For j = 1, ..., m

$$t_{i-1}p_{i-1,j} + (t_{i-1}v_{i-1} + t_i v_i)p_{i,j} + t_i p_{i+1,j} =$$

$$t_{i-1}(1 + v_{i-1})\frac{u_{i,j} - u_{i-1,j}}{\Delta\xi_{i-1}} + t_i(1 + v_i)\frac{u_{i+1,j} - u_{i,j}}{\Delta\xi_i} \tag{2.31}$$

where 2 ≤ i ≤ n-1.

For i = 1, ..., n

$$s_{j-1}q_{i,j-1} + (s_{j-1}w_{j-1} + s_j w_j)q_{i,j} + s_j q_{i,j+1} =$$

$$s_{j-1}(1 + w_{j-1})\frac{u_{i,j} - u_{i,j-1}}{\Delta\eta_{j-1}} + s_j(1 + w_j)\frac{u_{i,j+1} - u_{i,j}}{\Delta\eta_j} \tag{2.32}$$

where 2 ≤ j ≤ m-1.

For j = 1 and m

$$t_{i-1}r_{i-1,j} + (t_{i-1}v_{i-1} + t_i v_i)r_{i,j} + t_i r_{i+1,j} =$$

$$t_{i-1}(1 + v_{i-1})\frac{q_{i,j} - q_{i-1,j}}{\Delta\xi_{i-1}} + t_i(1 + v_i)\frac{q_{i+1,j} - q_{i,j}}{\Delta\xi_i} \tag{2.33}$$

where 2 ≤ i ≤ n-1.

For i = 1, ..., n

$$s_{j-1}r_{i,j-1}+(s_{j-1}w_{j-1}+s_jw_j)r_{i,j}+s_jr_{i,j+1}=$$

$$s_{j-1}(1+w_{j-1})\frac{p_{i,j}-p_{i,j-1}}{\Delta\eta_{j-1}}+s_j(1+w_j)\frac{p_{i,j+1}-p_{i,j}}{\Delta\eta_j} \tag{2.34}$$

where $2 \le j \le m-1$.

In equations (2.31) to (2.34), the values of v_i and w_j are determined using (2.22) and (2.23) respectively, and the values of t_i and s_j are determined as

$$t_i=\frac{\alpha_i^2\Delta\xi_i \sinh(\alpha_i\Delta\xi_i)}{(v_i^2-1)(\sinh(\alpha_i\Delta\xi_i)-\alpha_i\Delta\xi_i)} \qquad (i=1,\ldots,n-1) \tag{2.35}$$

$$s_j=\frac{\beta_j^2\Delta\eta_j \sinh(\beta_j\Delta\eta_j)}{(v_j^2-1)(\sinh(\beta_j\Delta\eta_j)-\beta_j\Delta\eta_j)} \qquad (j=1,\ldots,m-1) \tag{2.36}$$

The above tridiagonal systems of equations, (2.31)-(2.34), is also solved by the Thomas algorithm for the interior values of $p_{i,j}$, $q_{i,j}$ and $r_{i,j}$ (for i = 2,..., n-1 ; j = 2,..., m-1).

When all the partial derivative quantities and hence $K_{i,j}$ are known, the coefficient matrix $A_{i,j,k,l}$ for each and every rectangle of the computational domain grid can be found from (2.19). Equation (2.18) is then used to determine the interpolated u-value for any ξ and η in the computational domain, where

$$\xi_1 \le \xi \le \xi_n \quad and \quad \eta_1 \le \eta \le \eta_m$$

In such a way the whole surface can be defined. Depending on the value chosen for the tension factors, α and β, the interpolation can be varied between a bi-cubic spline and a piecewise linear spline.

CHAPTER 3

CREATING THE BLADE FRAME

An aerodynamic blade is characterized not only by the types of airfoils used along its span, but also by their chord, thickness, and twist distributions. The frame of a blade can be created by placing scaled and twisted airfoil data at particular span stations along the blade. This frame, after transformation, may then be used as the basis for the bi-tension spline interpolation presented in Chapter 2.

Described next is the blade frame creation process, which involves the following four main steps:

(1) Selection of span stations of the blade at which airfoils will be placed

(2) Selection of the type of airfoil for placement at these span stations

(3) Scaling the airfoils to the required chord and maximum thickness at their respective span stations

(4) Twisting the airfoils according to the required angle of twist at their respective span stations.

3.1 Choice of Span Stations of the Blade at which to Place Airfoils

The frame of the blade is to be used as the basis for interpolation. In order that this interpolation be as accurate as possible, it is advised that as many airfoils as possible be placed along the span of the blade in order to form the blade frame. The manner in which these airfoils is placed depends on the nature of the chord, thickness, and twist distributions along the span of the respective blade. More airfoils should be placed in regions of the span where the slope and/or curvature of the aforementioned distributions is high. This is so that the generated blades, chord, thickness, and twist distributions more closely match those of the desired blade. Conversely, it is not critical to have a high density of airfoils in regions along the span where all distributions are close to linear.

The availability of the correct airfoils is also a crucial factor in determining what span stations will have airfoils. In this work, results of [1] are used as the basis for placing the airfoils. The chosen span stations are represented by a Z-coordinate.

3.2 Choice of Airfoils

In a well-designed blade the airfoils that form it enhance its structural integrity and maximize its aerodynamic performance.

The structural integrity of the blade is enhanced by using thicker airfoils in the root region and gradually tapering the maximum thickness of the airfoils to a minimum at the tip of the blade. The blade's aerodynamic performance is optimized if its component airfoils are designed to maximize the desired forces, under given environmental conditions, in such a way that the structural integrity of the blade is not undermined.

This work utilizes already designed airfoils and places them at the chosen span stations. Figure 1 shows some existing low-speed airfoils [1] that could be used to form the frame of a blade for wind turbine rotors. Each airfoil has had its surface coordinates normalized with respect to its chord.

3.3 Scaling the Airfoils

For most of the airfoil data catalog, e.g., Abbott and Von Doenhoff [14], the X and Y coordinates are given in a form normalized with respect to the chord of the airfoil. To make the airfoil data conform to the required chord and thickness dimensions at the respective span stations, these data must be scaled. In the scaling process, the nondimensional coordinates of an airfoil are first scaled to the ones with the required maximum thickness, and then translated into the true dimensions with the desired chord-length. The modified Smoothing and Scaling code of Morgan [15] and Tu and Scott [16] was used in this work.

3.3.1 Scaling the Airfoils to the Required Maximum Thickness-to-Chord Ratio

The maximum thickness of an airfoil is the maximum straight-line distance between the upper and lower airfoil surfaces measured perpendicularly to the chord of the airfoil. In the case of a chord-normalized airfoil, this would represent the maximum thickness-to-chord ratio of the airfoil. The required maximum thickness-to-chord ratio at each blade frame span station is obtained from specification, or through interpolating an already designated maximum thickness-to-chord distribution at the particular span station.

The scaling code uses as input the camberline and thickness distributions generated by the smoothing code [16]. It first determines the maximum thickness of the airfoil and its position. The thickness distribution is then multiplied by a scale factor equal to the ratio of the new maximum thickness to the old maximum thickness. The camberline distribution and new thickness distribution are then combined to generate new thickness-scaled airfoil coordinates.

3.3.2 Scaling the Airfoils to the Required Chord Length

The chord length of an airfoil is the straight-line distance between the leading edge point and trailing edge point of the airfoil. In the case of chord-normalized airfoil data, the chord length is unity. For each blade frame span station airfoil, a required chord length can be obtained from specification or from an already designated chord distribution along the span of the blade. To scale the airfoil to the required chord length, the X and Y coordinates of the normalized airfoil are multiplied by the required chord length at the particular span station.

3.4 Twisting the Airfoils

To have better control and to obtain the desired aerodynamic force and moment distributions along the span of the blade, the blade must be twisted about its twist center. The prescribed twist angle for any particular airfoil can be obtained from an already designated twist distribution for the blade.

The center of twist for an airfoil is the aerodynamic center of the airfoil. This is because the moment coefficient about the aerodynamic center is independent of the angle of attack [17]. For most airfoils, the twist center is located close to the quarter-chord distance from the leading edge. In this work, the aerodynamic center can be set at the quarter-chord point, leading edge point, or any other point specified by the user. Before each airfoil is twisted, it must be offset by a distance equal to the distance between its leading edge and its twist center. The result is that the axis of twist centers for the component airfoils of the blade is the Z-axis. The translation representing the offset is

$$X_{os} = X - OS_X \tag{3.1}$$

$$Y_{os} = Y - OS_Y \tag{3.2}$$

where OS_X and OS_Y are the X and Y coordinates of the twist center, respectively.

The rotation transformation representing the twisting of the offset airfoil about its twist center is

$$X_{tw} = X_{os}\cos\theta - Y_{os}\sin\theta \tag{3.3}$$

$$Y_{tw} = X_{os}\sin\theta + Y_{os}\cos\theta \tag{3.4}$$

where θ is the twist angle.

Figures 3-5 summarize the blade frame creation processes. After placement of the scaled and twisted airfoils at their respective span stations, the resultant blade frame is shown in Figure 5.

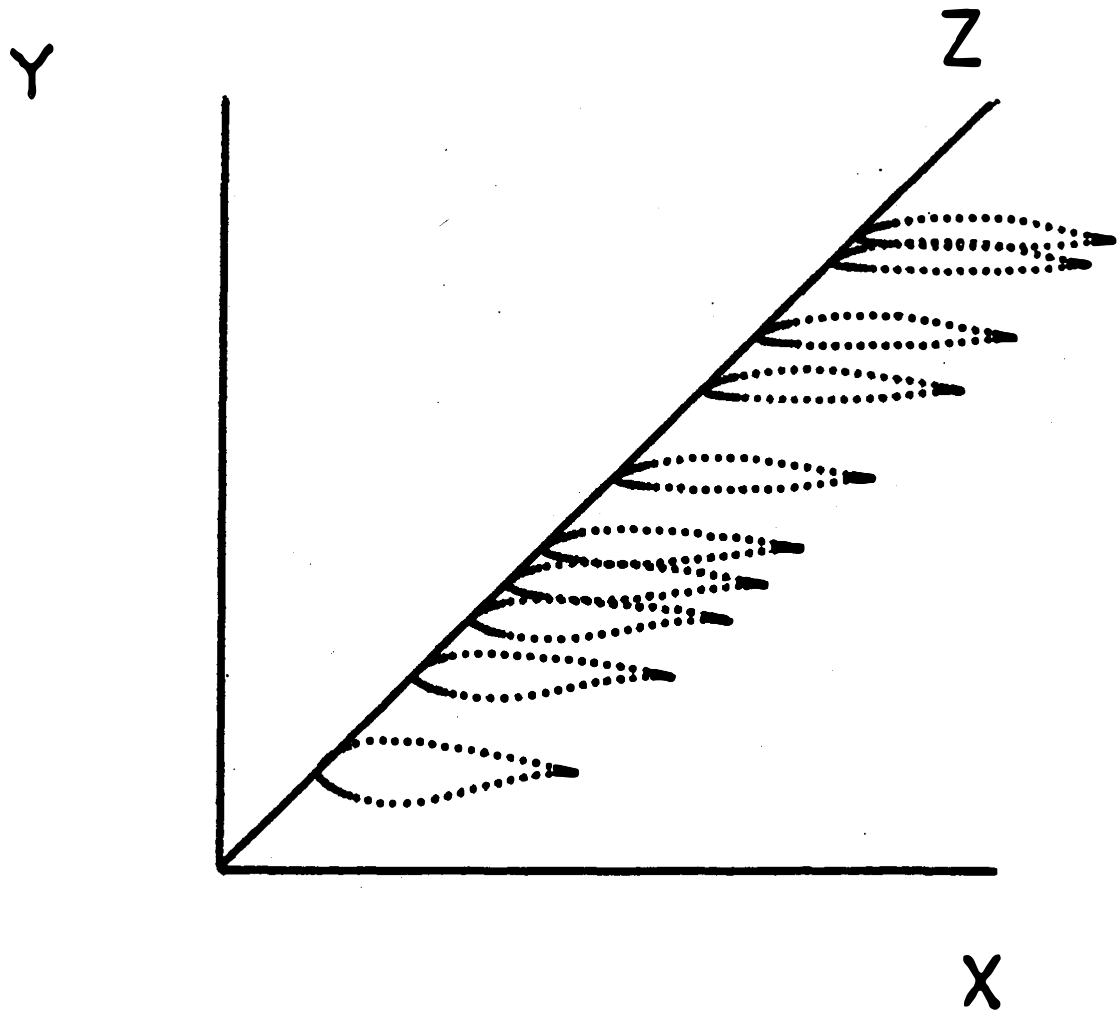


Figure 3 Chord-Normalized Input Airfoils at Respective Span Stations

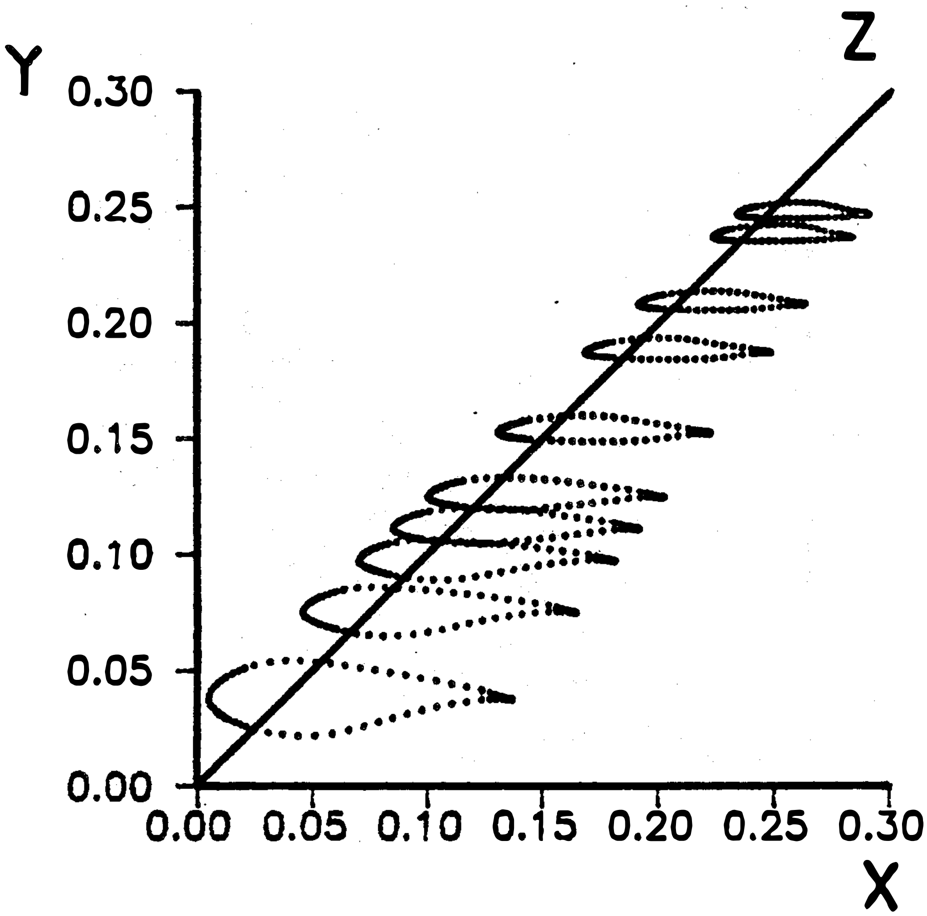


Figure 4 Input Airfoils That Have Been Scaled and Offset, and Then Placed at Their Respective Span Stations

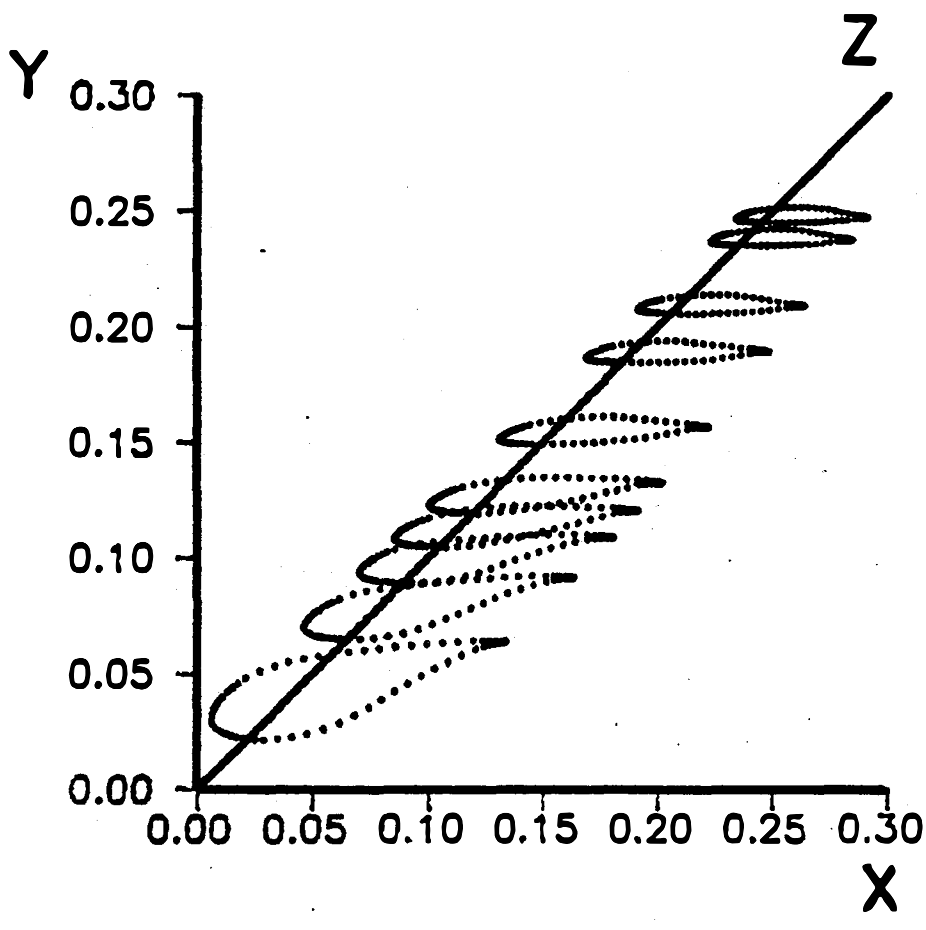


Figure 5 Input Airfoils That Have Been Scaled, Offset, and Twisted, and Then Placed at Their Respective Span Stations

CHAPTER 4

TRANSFORMATION OF THE BLADE FRAME

Now that the blade frame has been formed, the tension spline method is applied to generate the whole blade surface. As shown in equations (2.16) and (2.17), the interpolation requires that the independent variables increase monotonically and that the computational domain be a rectangular mesh. However, the X, Y, and Z coordinates of the blade frame described in Chapter 3 do not conform to these requirements. It is therefore necessary to transform the coordinates into a computational domain coordinate system that is compatible with the interpolation scheme. In addition, to facilitate the surface interpolation scheme and the ease of programming, it is also desired that the computational domain be confined to a unit square region. The bi-tension spline interpolation will be carried out in the computational domain.

In this work the transformation from the physical domain to the computational domain consists of mathematical relationships mapping one domain to another, such that changes to the grid are direct and rapidly obtained and transformation data are readily available for use. To set up and use the computational-domain coordinates as the basis for surface interpolation, the following sequence of manipulations is performed:

(1) The normalized Z-coordinate, η, of each airfoil's span station is computed.

(2) The polygonal arc length of each airfoil is computed and normalized.

(3) New X and Y coordinates for each airfoil are obtained by interpolation to form a regular grid net.

(4) Finally, the computational domain is defined and is ready to use as a basis for interpolation.

These manipulations are now described.

4.1 Calculation of the Normalized Z-Coordinate

The normalized Z-coordinate is represented by

$$\eta_j = \frac{(Z_j - Z_1)}{(Z_{jmax} - Z_1)} \tag{4.1}$$

where $1 \le j \le jmax$, and jmax is the total number of input airfoils used to define the blade frame. The transformation equation (4.1) transforms the Z-coordinate into a unit domain

$$0 \le \eta_j \le 1 \quad . \tag{4.2}$$

4.2 Computation and Normalization of the Polygonal Arc length for Each Input Airfoil

The polygonal arc length is the polygonal distance between the trailing edge of the airfoil and any point on the airfoil's surface. This distance is measured clockwise beginning at the lower surface trailing edge through all points on the surface prior to the point in question. The polygonal arc length, S, for each airfoil is determined as follows:

$$S_1 = 0$$

and

$$S_i = \sqrt{[(X_i - X_{i-1})^2 + (Y_i - Y_{i-1})^2]} + S_{i-1} \tag{4.3}$$

for $2 \le i \le imax$, where imax is the number of points used to define the airfoil surface.

To normalize the polygonal arc length, the following equation is used:

$$\overline{S}_i = \frac{S_i}{S_{imax}} \tag{4.4}$$

where $1 \le i \le imax$. Equation (4.4) transforms the S-coordinate into a unit domain

$$0 \le \overline{S}_i \le 1 \tag{4.5}$$

4.3 Interpolation of New X and Y Coordinates for Each Airfoil

The transformation relations stated above map the computational domain onto a unit square, but the grid net is not rectangular. It is therefore necessary to define a new set of data for each airfoil through interpolation such that a rectangular grid net is obtained. In the early stages of this algorithm's development, the use of an equally spaced set of normalized polygonal arc lengths was adequate for the interpolation. Because of the high gradient of the slope and curvature near the nose of the airfoils, that area could not be well defined. To overcome this discrepancy, a stretching function, which automatically concentrates the data points about the leading edge, is introduced.

Accordingly, the approach adopted is first to generate a set of equally spaced normalized polygonal arc lengths, $\bar{\xi}_i$; then to refine the equally spaced polygonal arc lengths about the nose of the airfoils through a stretching function; and ultimately to use the resulting set of polygonal arc lengths, ξ_i, to interpolate for new values of X and Y, for each airfoil, as functions of their respective normalized polygonal arc lengths computed by equation (4.4). Following are the relations representing the approach.

Let imax be the number of desired computational grid points on each airfoil. Then for an equally spaced grid, the increment $\Delta\bar{\xi}$ would be

$$\Delta\bar{\xi} = \frac{1.0}{imax - 1} \tag{4.6}$$

and the new set of equally spaced normalized polygonal arc lengths would be

$$\bar{\xi}_i = \Delta\bar{\xi} \times (i - 1) \tag{4.7}$$

for $1 \leq i \leq imax$. To concentrate the data points about the leading edge automatically, the following stretching function is used [12]:

$$\xi_i = \bar{\xi}_{mid}\left\{1 + \frac{\sinh[\tau(\bar{\xi}_i - B)]}{\sinh(\tau B)}\right\} \tag{4.8}$$

where

$$B = \frac{1}{2\tau} \ln \left[\frac{1+(e^{\tau}-1)\xi_{mid}}{1+(e^{-\tau}-1)\xi_{mid}} \right] \tag{4.9}$$

and τ is the stretching parameter, which varies from zero (no stretching) to infinity (most refinement) about ξ_{mid}. For each airfoil, ξ_{mid} represents the leading edge of the airfoil. This value varies among airfoils; experience shows that a value of 0.5 is close enough to the leading edge. Therefore, ξ_{mid} is set equal to 0.5 for all calculations. Figure 6 shows how the refinement (grid spacing in ξ) varies with different values of τ about $\xi_{mid} = 0.5$. The same number of grid points is used for three different τ values. Experience shows that for the best result, τ should be set to a value between 4.0 and 6.0, depending on the value of imax chosen.

New values of X and Y for each airfoil are then interpolated at the new set of the normalized polygonal arc lengths, represented by equation (4.8), using the 1-D tension spline described in Chapter 2.

4.4 Defining and Using the Computational Domain Coordinates.

As the result of the data processing steps presented above, the X and Y values become a function of ξ and η, i.e.,

$$X = X(\xi, \eta)$$

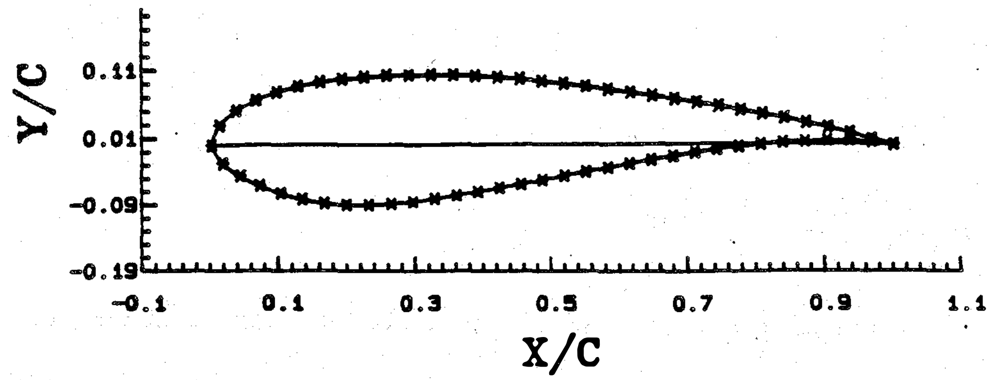
and

$$Y = Y(\xi, \eta)$$

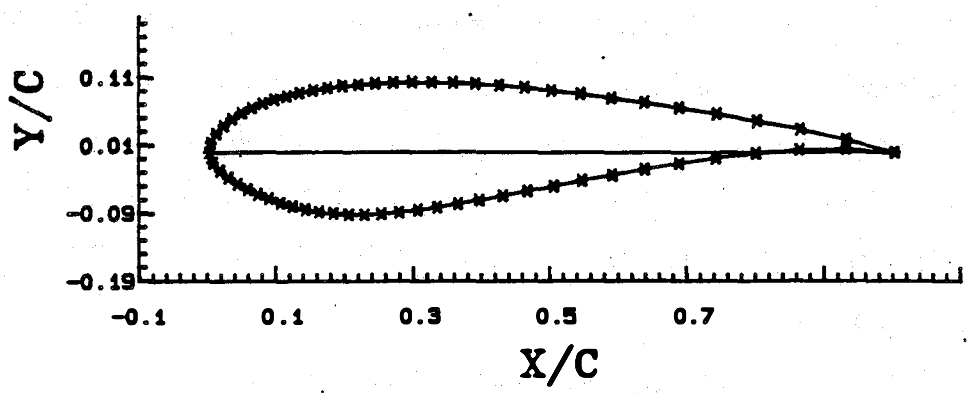
The blade frame is now defined by (X_{ij}, ξ_i, η_j) and (Y_{ij}, ξ_i, η_j).

By performing two separate 2-D interpolations on the computational domain of the blade frame, the physical coordinates (X,Y,Z) of any point on the surface of the blade can be determined.

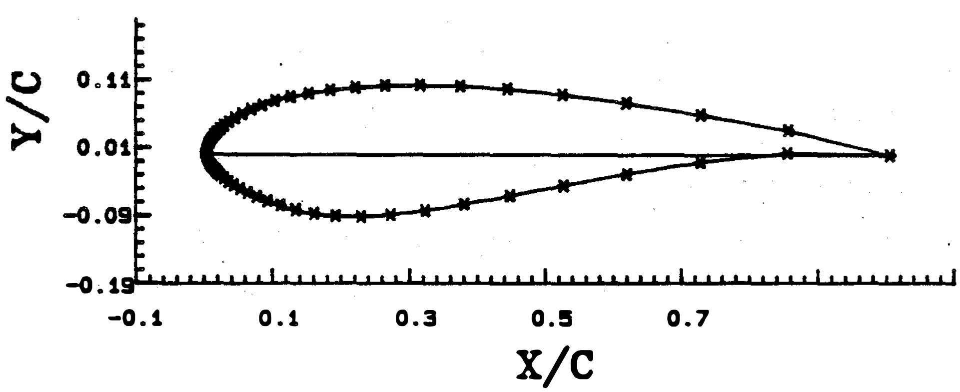
Specifically, to find the X coordinate of a point on the surface of the blade, the surface tension spline method described in Chapter 2 is applied to the (X_{ij}, ξ_i, η_j) data set; see Figure 7.



(a) $\tau = 0.0001$



(b) $\tau = 4.5$



(c) $\tau = 10$

Figure 6 Variation in Grid Spacing with Three Differe for an Airfoil Represented by 65 Points

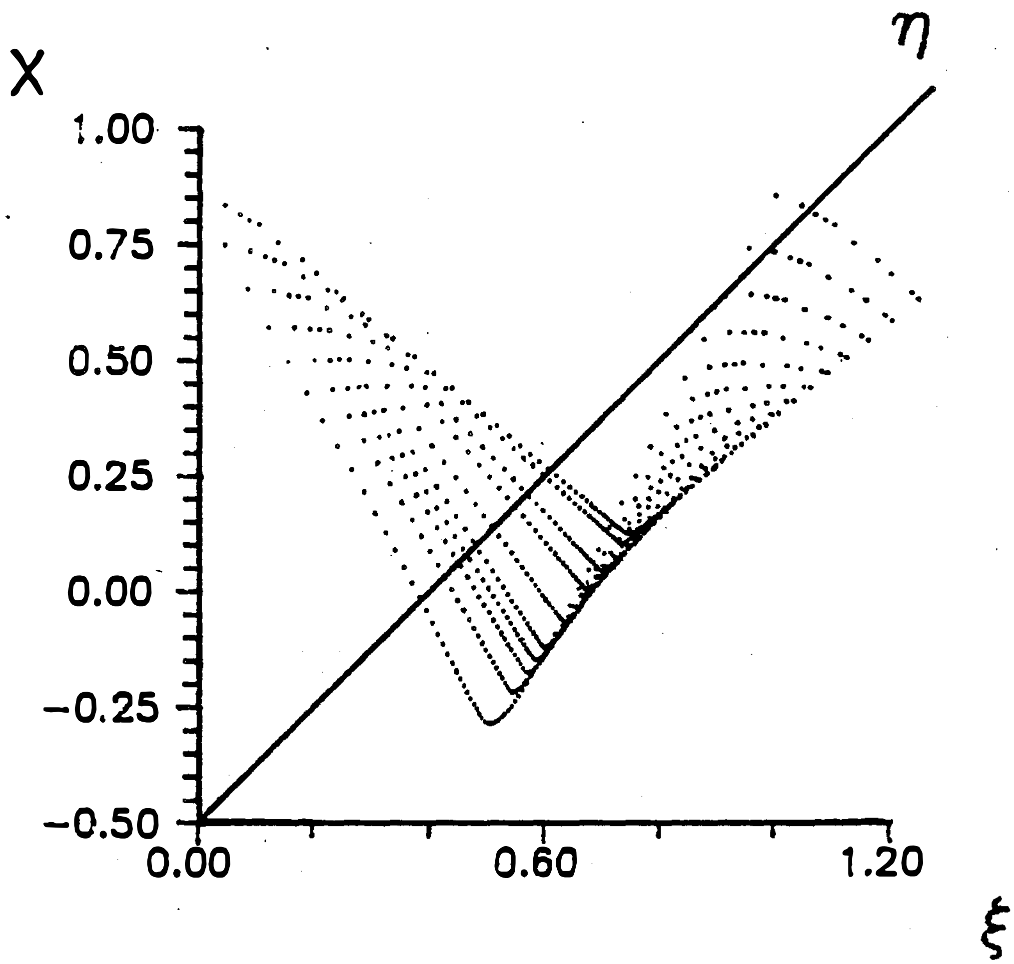


Figure 7 Computational Domain with X as the Dependent Variable

Similarly, to find the corresponding Y coordinate at the same point, the (Y_{ij}, ξ_i, η_j) data set is used; see Figure 8. The Z coordinate of the point is determined by the inverse transformation of equation (4.1).

In such a way, the physical coordinates of any point in the computational domain of the blade frame can be determined. The whole blade surface therefore is defined.

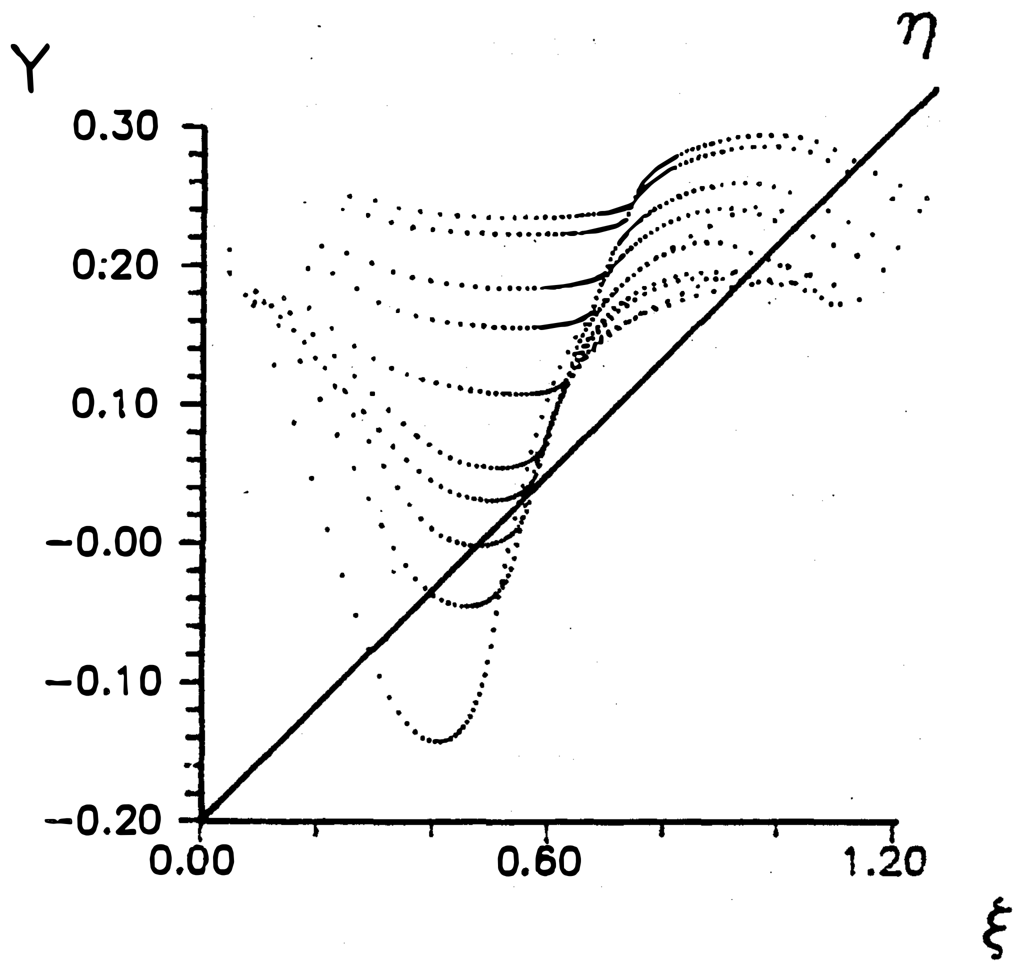


Figure 8 Computational Domain with Y as the Dependent Variable

CHAPTER 5

PROGRAM DESCRIPTION

Presented in this chapter are the program structure, descriptions of the function of each subroutine, and a definition of all common block variables used.

5.1 Program Structure

In order that the whole program structure be appropriately described, the program is broken down into six modules: Input, Blade frame creation, Transformation, Interpolation, Partial derivatives (called by the Interpolation module), and Output. Figure 9 shows the order in which these modules are accessed by the main program.

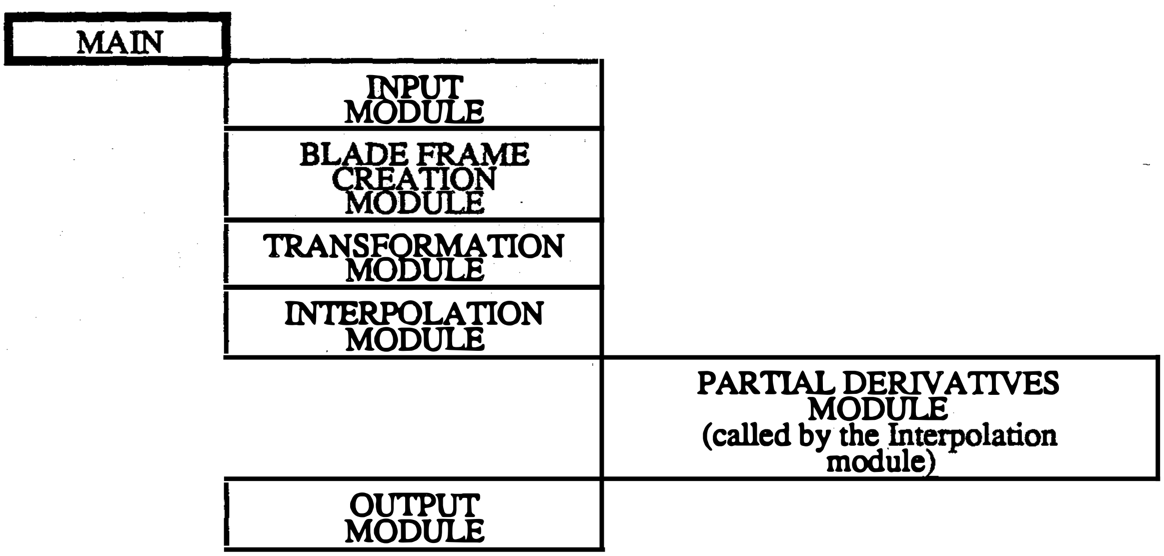


Figure 9 Order of Access of Modules in the BLADE Code

5.1.1 Input Module

The input module's purpose is primarily to accept all necessary input data either from the terminal (subroutine TINPUT) or from the data file (subroutine DINPUT). This module also controls the scaling process of the input airfoils.

Subroutine INPUT is the driver of the input module from which either subroutine TINPUT or DINPUT is called. Both TINPUT and DINPUT call subroutines SPT11D and CURVE1 and function CURVE2. The smoothing and scaling codes, used for the scaling process, are run in batch mode from subroutines DINPUT and TINPUT. Figure 10 shows the structure of the input module.

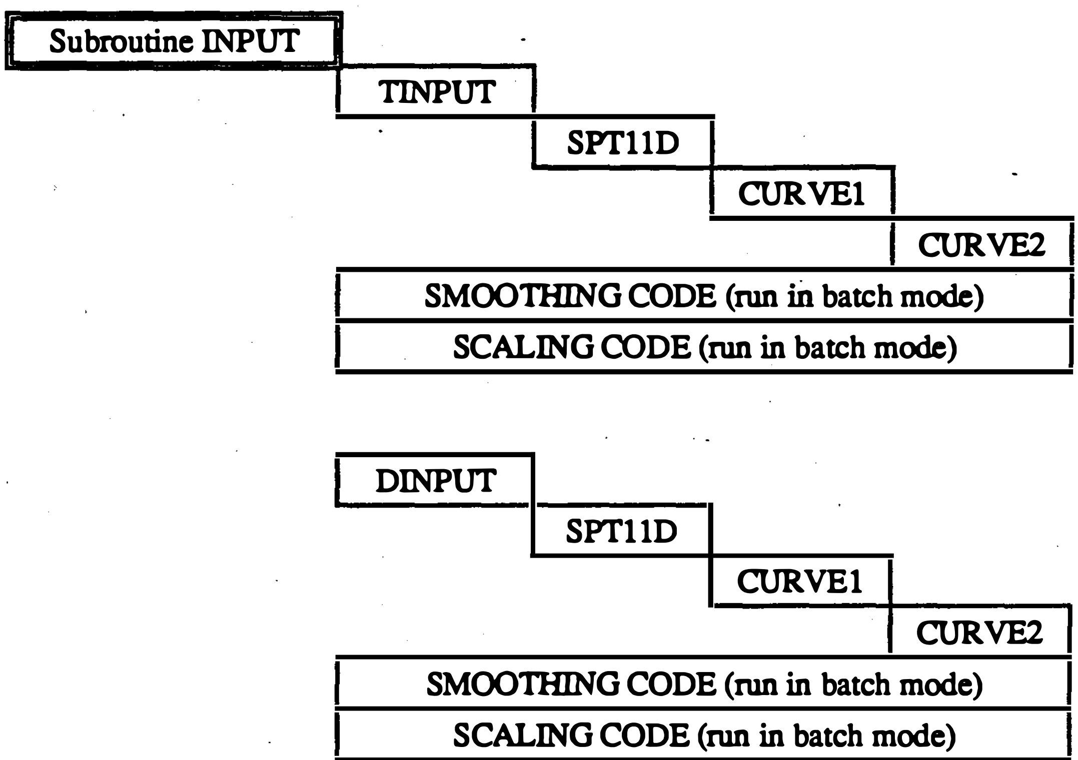


Figure 10 Structure of the Input Module

5.1.2 Blade Frame Creation Module

The blade frame creation module's purpose is to create the frame of the blade from the scaled input data. Subroutine SCOSTW is the sole module's subroutine. In this module, each maximum-thickness-scaled input airfoil is scaled to the prescribed chord, offset to the desired twist center, twisted by the prescribed twist angle, and then placed at the desired span station.

5.1.3 Transformation Module

The transformation module's purpose is to transform the physical coordinates of the blade frame into computational coordinates. Subroutine TRANS is the driver of the transformation module. Subroutines SPT11D and CURVE1 and function CURVE2 are called by subroutine TRANS. Figure 11 shows the structure of the transformation module.

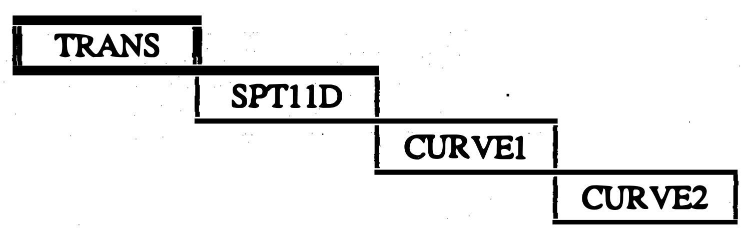


Figure 11 Structure of the Transformation Module

5.1.4 Partial Derivatives Module

The purpose of the partial derivatives module is to control and determine the partial derivatives of the dependent variable (X or Y) with respect to ξ, η, and ξ and η respectively. The partial derivatives module is called by the interpolation module. Subroutine PDERIV is the driver of the partial derivatives module whose structure is shown in Figure 12.

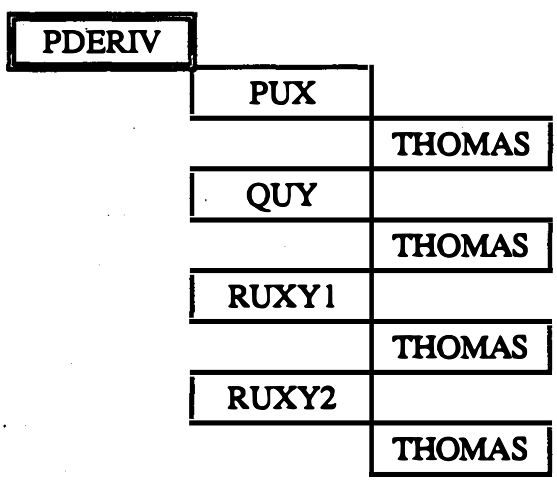


Figure 12 Structure of the Partial Derivatives Module

5.1.5 Interpolation Module

The purpose of the interpolation module is to control and perform the surface interpolation of the transformed blade frame, in the computational domain, so that the physical coordinates representing an airfoil at any span station of the blade can be determined. Subroutine CHOICE is the first routine called by the interpolation module. Subsequently, subroutine SECT is the driver of the interpolation procedure. Each time the coordinates of an airfoil are being determined in subroutine SECT, subroutines GETX and GETY are called. Both GETX and GETY call the partial derivatives module to provide required information. Figure 13 shows the structure of the interpolation module.

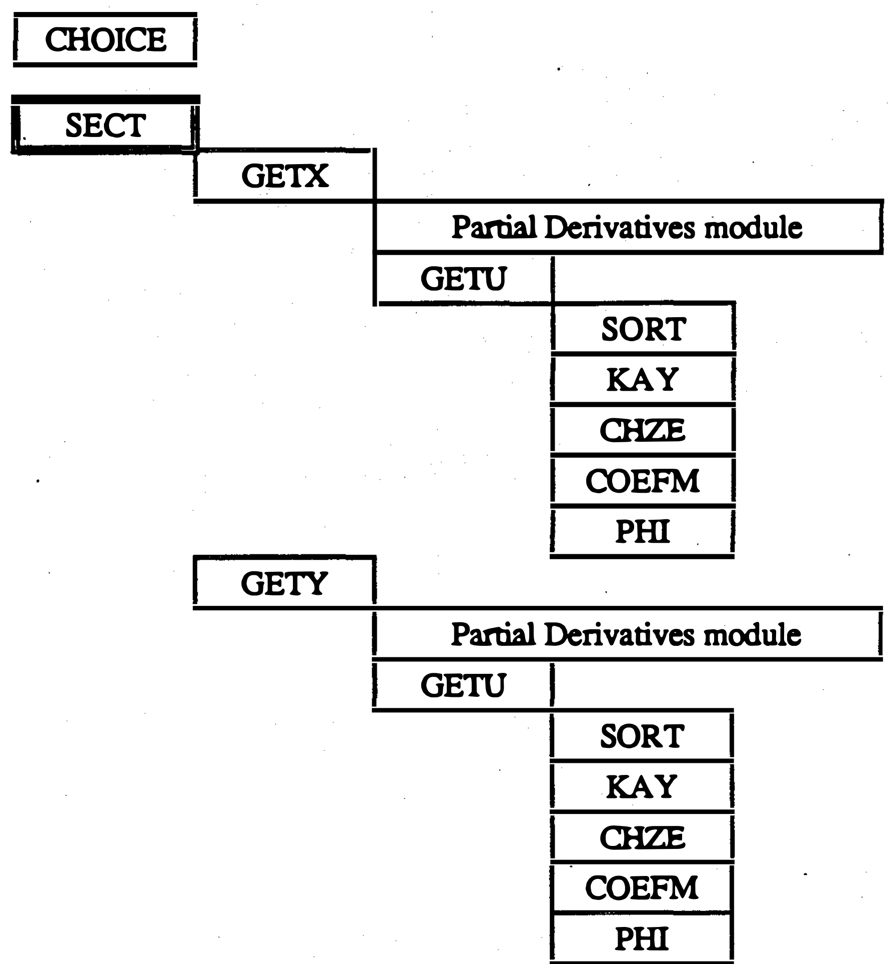


Figure 13 Structure of the Interpolation Module

5.1.6 Output Module

The purpose of the output module is to control and perform the viewing, plotting, and printing of input and generated graphs, tables, or data. The driver of the output module is subroutine PLOTPR from which subroutine HRDOUT is called. Subroutine HRDOUT controls the graphic output device options. Twenty-four programs were developed to facilitate the viewing, plotting, and printing of all the graphics in the program. These programs, which are run in batch mode from subroutine HRDOUT, call upon various GRAFMATIC, PLOTMATIC, and PRINTMATIC[1] subroutines to view, plot, and print the graphs. Figure 14 shows the structure of the output module.

5.2 Subroutine Functions

Following is a description of the function of each subroutine in the FORTRAN program developed:

BLADE	Main program that controls overall program execution.
CHOICE	Allows the user to make a choice of generating either a spanwise station airfoil or a three-dimensional blade defined by at least three airfoils.
CURVE1	Determines the parameters necessary to compute an interpolatory 1-D tension spline through a series of points.
CURVE2	Interpolates a curve at a given point using a spline-under-tension.
CHZE	Determines the two 4-by-4 matrices, CHZY and CHZX equation (2.20) necessary for computations in subroutine COEFM.
CKFILE	Checks and deletes an existing file.

1 GRAFMATIC, PLOTMATIC, AND PRINTMATIC are trademarks of Microcompatibles, Inc. For those who do not have this graphic package, programs listed in Figure 14 should be replaced.

Subroutine PLOTPR

HRDOUT

VICPINP[2] (run in batch)	PLCPINP[3] (run in batch)	PRCPINP[4] (run in batch)
VICPGEN (run in batch)	PLCPGEN (run in batch)	PRCPGEN (run in batch)
VICHNPL (run in batch)	PLCHNPL (run in batch)	PRCHNPL (run in batch)
VITOP (run in batch)	PLTOP (run in batch)	PRTOP (run in batch)
VIFRONT (run in batch)	PLFRONT (run in batch)	PRFRONT (run in batch)
VICHORD (run in batch)	PLCHORD (run in batch)	PRCHORD (run in batch)
VITWIST (run in batch)	PLTWIST (run in batch)	PRTWIST (run in batch)
VITHICK (run in batch)	PLTHICK (run in batch)	PRTHICK (run in batch)

Figure 14 Structure of the Output Module

2 Programs listed in this column are for viewing the graphic output on the screen.

3 Programs listed in this column are for sending the graphic output to the plotter.

4 Programs listed in this column are for sending the graphic output to the printer.

COEFM Determines the 4-by-4 coefficients matrix equation (2.19) for each sector of the computational domain grid.

DINPUT : Called by INPUT to perform the "data file" input mode option. It also prepares all airfoil data for thickness scaling. Runs the smoothing and scaling codes in batch mode.

GETX Determines the X-coordinate of all points on the surface of an airfoil at a particular span station of the blade.

GETY Determines the Y-coordinate of all points on the surface of an airfoil at a particular span station of the blade.

GETU Called by GETX and GETY to determine each interpolated dependent variable.

HRDOUT Called by PLOTPR for controlling the graphic output device options.

INPUT Accepts all major input necessary to successfully run the program.

KAY Creates a 4-by-4 matrix, equation (2.21), containing all the values of u and its partial derivatives, at the four nodes of each sector in the computational domain grid.

PDERIV Determines the boundary values of the partial derivatives of u (the dependent variable), with respect to ξ, η, and the cross derivative of u with respect to ξ and η according to equations (2.25)-(2.30). It then calls PUX, QUY, RUXY1, and RUXY2 to determine those partial derivatives it has not yet determined according to equations (2.13)-(2.34).

PLOTPR Plots and prints input and generated data.

PHI Determines the two interpolation function vectors, in the ξ and η directions, for each sector of the computational domain grid, by using equations (2.3) and (2.4).

PUX	Determines the partial derivative of u with respect to ξ at every internal node of the computational domain grid. These derivatives are contained in an array P. Uses equations (2.25) and (2.26) as boundary conditions to solve systems, equation (2.31), for $P_{i,j}$.
QUY	Determines the partial derivative of u with respect to η at every internal node of the computational domain grid. These derivatives are contained in an array Q. Uses equations (2.27) and (2.28) as boundary conditions to solve systems, equation (2.32), for $Q_{i,j}$.
RUXY1	Determines the cross derivative of u with respect to ξ and η at each boundary node of the computational domain grid where η has a value of 0 or 1 and ξ does not have values of 0 or 1. These derivatives are contained in an array R. Uses equations (2.29) and (2.30) as boundary conditions to solve systems, equations (2.33), for $R_{i,j}$.
RUXY2	Determines the cross derivative of u with respect to ξ and η at each node of the computational domain grid where, η does not have values of 0 and 1. Uses equations (2.29) and (2.30) as boundary conditions to solve system, equation (2.34), for $R_{i,j}$.
SCOSTW	Generates new X and Y coordinates of airfoils caused by chord scaling, offsetting, and then twisting the airfoils. See Chapter 3.
SECT	Controls the 2-D interpolation process of the points on the blade surface.
SORT	Finds the computational domain grid sector in which the point (ξ, η) lies. It returns the (ξ, η) coordinate of the bottom left corner node of the sector.
SPT11D	The main routine that calls CURVE1 and CURVE2 to perform a one-dimensional tension spline interpolation of a given curve at a given point.

THOMAS Solves a tridiagonal system of equations using the Thomas algorithm.

TINPUT Called by INPUT to perform the "terminal" input mode option. It also prepares all airfoil data for thickness scaling. Runs the smoothing and scaling codes in batch mode.

TRANS Transforms the physical domain coordinates (X,Y,Z) into the computational coordinates (X, ξ, η) and (Y, ξ, η) by the method in Chapter 4.

TRANSPO : Finds the transpose of a matrix (See equation [2.19]).

5.3 Critical Variable Definitions

The critical variables used in the computer code are contained in twenty-nine named common blocks. In defining the variables we will therefore list them by common block as is done below:

C1 TH1 - Vector containing twist angles of input airfoils.

OSX - Vector containing the chord-normalized twist center X coordinates.

OSY - Vector containing the chord-normalized twist center Y coordinates.

CL - Vector containing chord lengths of input airfoils.

C2 X - Array containing the X coordinates of thickness-scaled input airfoils.

Y - Array containing the Y coordinates of thickness-scaled input airfoils.

C3 X1 - Array containing the X coordinates of scaled, offset, and twisted input airfoils for all input span stations.

Y1 - Array containing the Y coordinates of scaled, offset, and twisted input airfoils for all input span stations.

C4 ZI - Vector containing the normalized polygonal arc length coordinates (ξ) for the input span stations.

ETA- Vector containing the normalized Z coordinates (η) for the input span stations.

C5 XMIN - Vector containing the minimum X coordinates at each span station of X1.

XMAX - Vector containing the maximum X coordinates at each span station of X1.

C6 DZI - Vector containing magnitudes of increment of ξ.

DETA - Vector containing magnitudes of increment of η.

C7 P - Array containing partial derivatives of u with respect to ξ at each η.

Q - Array containing partial derivatives of u with respect to η at each ξ.

R - Array containing cross derivatives of u with respect to ξ and η at each ξ and η.

C8 IC - Logic variable indicating user's choice of blade generation.

ZIT - Z coordinate of span station at which user wants interpolation carried out.

C9 AL - Vector of normalized tension factors for each of the nodal intervals in the ξ direction.

BE - Vector of normalized tension factors for each of the nodal intervals in the η direction.

C10 V, W - Vectors containing necessary coefficients in the tridiagonal systems of equations (equations [2.31]-[2.34]) that are solved to find the required partial derivatives.

C11 ALPHA, BETA - Input values of tension factors in the ξ and η directions, respectively.

C12 Z - Vector containing the Z coordinates of the input span stations.

C13 U - Matrix containing the dependent variable, i.e., Y or X coordinates of the scaled, offset, and twisted airfoils at all input span stations.

C14A XU, XL - Matrices containing the X coordinates of the upper and lower surfaces, respectively, of all the input airfoils.

C14B YU, YL - Matrices containing the Y coordinates of the upper and lower surfaces, respectively, of all the input airfoils.

C15 S, T - Vectors containing necessary coefficients in the tridiagonal systems of equations (equations [2.31]-[2.34]) that are solved to find required partial derivatives.

C16 XIT, YIT - Internodal values of ξ and η, respectively, that are currently being used to interpolate for the Y- or X-coordinate.

C17 X3, Y3, Z3 - Rewound coordinates of the scaled, offset, twisted, and resampled input airfoils.

C18A XU1, YU1 - Matrices containing the X and Y coordinates, respectively, of the upper surfaces of the scaled, offset, and twisted input airfoils.

C18B XL1, YL1 - Matrices containing the X and Y coordinates, respectively, of the lower surfaces of the scaled, offset, and twisted input airfoils.

C20 U1, U2 - Matrices containing the values of X and Y, respectively, as functions of ξ and η.

C22 XZ, YZ - Vectors containing the X and Y values as interpolated by the surface tension spline at the current η station. XZ is computed in GETX and YZ in GETY.

C25 SLE - Vector containing the polygonal arc lengths to each airfoil's leading edge.
TAU - Refinement parameter.

C30 NOW - Number of rows of data in the chord/twist/thickness distribution file.
ROR - Z/span coordinate of the airfoil.

COR - Chord/span ratio of the airfoil.

TWI - Twist of the airfoil.

THK - Maximum thickness-to-chord ratio of the airfoil.

C31 SIGMACH, SIGMATW, SIGMATH - Tension parameters needed for the one-dimensional interpolation of the chord, twist, and thickness distributions at the required span stations of the blade frame.

TRIDM CL, CM, CN - Vectors containing lower, leading, and upper diagonal coefficients, respectively, of a tridiagonal matrix.

PQ - Vector that originally contains the right-hand-side values of the tridiagonal system, and that, after solution of the system, contains the solution.

AUKU AK - A 4-by-4 matrix that originally contains the function values of u and its partial derivatives for all four nodes of a sector; see equation (2.21). After manipulations, contains the spline coefficients for the particular sector concerned, see equation (2.19).

CHZXY CHZX, CHZY - Two 4-by-4 coefficient matrices that contain values necessary in computing the spline coefficients for each sector (see equation [2.19]).

PHIXY PHIX, PHIY - Vectors of length 4 that contain interpolation function values in the ξ and η directions, respectively (see equation [2.3]).

CHAPTER 6

HOW TO RUN THE CODE

Included in this chapter are the descriptions of

(1) How to install the code

(2) The necessary input data files

(3) Options for running the program

(4) A sample run of the program.

6.1 Program Installation

Before the code can be executed, the following must be done:

(1) Make sure that the F77L[1] software is loaded on the system.

(2) Make sure that the GRAFMATIC, PLOTMATIC, and PRINTMATIC[2] software libraries are loaded on the system.

(3) Write down the paths to, and the directories into which, the F77L, GRAFMATIC, PLOTMATIC, and PRINTMATIC libraries have been loaded.

(4) Make a directory called BLADE (or any other name), and make this directory the current directory, for example:

MD \BLADE

CD \BLADE

1 F77L is the trademark of Lahey Computer Systems, Inc. For those who do not have the Lahey FORTRAN 77 compiler, all the "CALL SYSTEM (comnd)" statements in the BLADE source code must be modified. The SYSTEM subroutine is a F77L special user-callable system subroutine provided as an extension to the FORTRAN language. This subroutine passes a character expression "comnd" to DOS to be executed as if it had been typed at the console.

2 See the footnote on page 37 for detail.

(5) Copy all the source files from the BLADE diskette into the BLADE directory, for example:

COPY A:*.* \BLADE*.*

(6) Compile and link all the source files in the BLADE directory by typing the following at the DOS prompt:

COMPILE_PD1_PD2_PD3_PD4_PD5_PD6

where

PD1 is the path and directory containing the F77L execution and library files (F77L.EXE and F77L.LIB), e.g., C:\F77L\

PD2 is the path and directory containing the GRAFMATIC library for Lahey FORTRAN (GRAFEXLY.LIB), e.g., C:\GRAFMATIC\

PD3 is the path and directory containing the GRAFMATIC screen font library (QFONTLY.LIB), e.g., C:\GRAFMATIC\

PD4 is the path and directory containing the PLOTMATIC library for Lahey FORTRAN (PLOTLHP.LIB), e.g., C:\PLOTMATIC\

PD5 is the path and directory containing the PLOTMATIC plot font library for HP plotters (HPFONTLY.LIB), e.g., C:\PLOTMATIC\

PD6 is the path and directory containing the PRINTMATIC library (PRINTLY.LIB), e.g., C:\PRINTMATIC\

_ signifies a blank space.

(7) Set up the system for sending graphics to the screen, plotter, and printer. This is done by typing SETUP at the DOS prompt.

NOTE: The set up step is especially important if one has just installed the code on the system, or if one intends to change the terminal, plotter, or printer hardware.

(8) Make sure that all the necessary data files are present. If not, create them (see Sections 6.2 and 6.3).

(9) Begin the execution of the Blade code by typing BLADE at the DOS prompt.

6.2 Input Data Files

Before the code is executed the user needs to prepare files for airfoil data, chord/twist/thickness distribution, and general input data (optional). Note that all data are read in free format by the program. Therefore, the data files do not have to be formatted in any specific way.

6.2.1 Airfoil Data File

The airfoil data file contains X and Y coordinates of the upper and lower surfaces of an airfoil. All data should already be normalized with respect to the chord length of the airfoil. Data are read from this file using free format.

Following is a line-by-line description of the data contained in the airfoil data file.

Line 1: NU

NU: Number of points on upper surface of airfoil.

Line 2: NL

NL: Number of points on lower surface of airfoil.

Lines 3 to line (2+NU): XU, YU

XU: X-coordinate of upper airfoil surface.

YU: Y-coordinate of upper airfoil surface.

Line (3+NU) to line (2+NU+NL): XL, YL

XL: X-coordinate of lower airfoil surface.

YL: Y-coordinate of lower airfoil surface.

Line (3+NU+NL), i.e., the last line:

Blank

An example airfoil data file, S805A7.DAT, is shown in Figure 15. The user should prepare one airfoil data file for each input airfoil.

Line 1	33	Number of points on upper airfoil surface.
Line 2	33	Number of points on lower airfoil surface.
Line 3	0.00000 0.00000	Lines 3 through 35 are the data of X and Y coordinates of upper airfoil surface.
Line 4	0.00270 0.00855	
Line 5	0.00997 0.01836	
Line 6	0.01992 0.02714	
Line 7	0.02985 0.03395	
Line 8	0.03996 0.03982	
Line 9	0.04990 0.04485	
Line 10	0.05996 0.04940	
Line 11	0.07998 0.05722	
Line 12	0.09994 0.06378	
Line 13	0.11995 0.06938	
Line 14	0.14996 0.07627	
Line 15	0.19998 0.08417	
Line 16	0.25000 0.08840	
Line 17	0.30000 0.09019	
Line 18	0.35001 0.09027	
Line 19	0.40001 0.08904	
Line 20	0.45001 0.08658	
Line 21	0.50002 0.08266	
Line 22	0.55001 0.07706	
Line 23	0.60001 0.07036	
Line 24	0.65000 0.06326	
Line 25	0.70000 0.05594	
Line 26	0.75001 0.04843	
Line 27	0.80001 0.04069	
Line 28	0.85001 0.03255	
Line 29	0.90001 0.02351	
Line 30	0.93001 0.01705	
Line 31	0.95000 0.01215	
Line 32	0.97000 0.00700	
Line 33	0.97999 0.00445	
Line 34	0.98999 0.00206	
Line 35	1.00000 0.00000	

Note: This data file is continued on the next page.

Figure 15 Sample Input Airfoil Data File (Example of S805A7.DAT)

Line 36	0.00000	0.00000	Lines 36 through 68 are the data of X and Y coordinates of lower airfoil surface.
Line 37	0.00297	-0.00772	
Line 38	0.01003	-0.01375	
Line 39	0.01997	-0.01916	
Line 40	0.02996	-0.02361	
Line 41	0.03994	-0.02727	
Line 42	0.04998	-0.03046	
Line 43	0.05996	-0.03333	
Line 44	0.07997	-0.03838	
Line 45	0.10000	-0.04268	
Line 46	0.11998	-0.04642	
Line 47	0.14997	-0.05120	
Line 48	0.20000	-0.05739	
Line 49	0.25000	-0.06166	
Line 50	0.30000	-0.06413	
Line 51	0.35001	-0.06455	
Line 52	0.40002	-0.06282	
Line 53	0.45002	-0.05920	
Line 54	0.50001	-0.05415	
Line 55	0.55000	-0.04826	
Line 56	0.60000	-0.04193	
Line 57	0.65000	-0.03531	
Line 58	0.70000	-0.02845	
Line 59	0.74999	-0.02130	
Line 60	0.79999	-0.01389	
Line 61	0.84998	-0.00679	
Line 62	0.89999	-0.00117	
Line 63	0.92999	0.00109	
Line 64	0.95000	0.00193	
Line 65	0.97000	0.00202	
Line 66	0.98001	0.00169	
Line 67	0.99000	0.00103	
Line 68	1.00000	0.00000	
Line 69			The last line should be blank.

Figure 15 (concluded) Sample Input Airfoil Data File (Example of S805A7.DAT)

6.2.2 Chord/Twist/Thickness Distribution Data File

This data file is read in free format and contains data that define the chord, twist, and maximum thickness/chord ratio of an airfoil at specific span stations. The chord should be normalized with respect to the span of the blade.

Following is a line-by-line description of the data contained in the chord/twist/thickness distribution file.

<u>Line 1</u>: NOW

NOW: Number of span stations defining chord/twist/thickness distribution curve.

<u>Line 2 to line (1+NOW)</u>: ROR, COR, TWI, THKN

ROR: The Z coordinate of the airfoil at the span station, normalized with respect to span.

COR: The chord of the airfoil at the span station, normalized with respect to span.

TWI: The twist angle of the airfoil at the span station.

THKN: The maximum thickness/chord ratio of the airfoil at the span station.

<u>Line (2+NOW), i.e., the last line</u>:

Blank

An example chord/twist/thickness distribution file, CTE2.DAT, is shown in Figure 16. At least one chord/twist/thickness distribution data file must be on hand before execution of the program, if the user intends to use a specific distribution to generate his/her own blade surface.

Line 1	16	Number of span stations representing the distributions.
Line 2 Line 3 Line 4 Line 5 Line 6 Line 7 Line 8 Line 9 Line 10 Line 11 Line 12 Line 13 Line 14 Line 15 Line 16 Line 17	0.05 0.225 33.5 0.220 0.10 0.210 26.8 0.210 0.15 0.188 21.0 0.200 0.20 0.167 15.2 0.190 0.30 0.133 7.80 0.180 0.35 0.118 5.00 0.178 0.45 0.092 1.00 0.172 0.50 0.080 0.00 0.168 0.55 0.072 -0.7 0.162 0.65 0.065 -1.0 0.150 0.70 0.065 -1.0 0.143 0.75 0.065 -1.0 0.135 0.80 0.065 -1.0 0.125 0.85 0.065 -1.0 0.115 0.90 0.065 -1.0 0.101 0.95 0.065 -1.0 0.088	Lines 2 through 17 contain values of span coordinates (first column), chord at the current span station (second column), the twist angle in degrees at the current span station (third column), and the maximum thickness/chord ratio at the current span station (fourth column).
Line 18		The last line should be blank.

Figure 16 Sample Chord/Twist/Thickness Distribution Data File (Example of CTE2.DAT)

6.3 Running the Program

The code can accept input either from the terminal or from data files. Following are descriptions of the input and output options a user will encounter during the execution of the code.

6.3.1 Input Options

It is advisable for beginning users to enter input at the terminal, in order to acquaint themselves with the code. However, a tedious amount of data are requested, especially if many input airfoils are being used. For this reason the user is provided an option for entering data by use of an input data file. Most of the input data needed during the execution of the code are requested by subroutine INPUT.

When the program is executing, the user is first introduced to the package. After that, the user is asked for the way of input data as following:

The computer: You have two alternative ways of entering data:

Enter T for terminal input
Enter D for data file input

Enter your choice here ----->

The user's response: T
or
D: If and only if an input data File has been created beforehand.

If the user's response was D then:

The computer: Please enter the name of the data input file --->

The user's response: For example: JAYIN.DAT

The remaining data required for the blade surface generation are either input at the terminal or from the data file specified in the step above. An example input data file, JAYIN.DAT, is shown in Figure 17. Whether input is from the terminal or from an input data file, it is read sequentially using free format.

For simplicity, and yet in order to fully define the remaining of the input data requested during execution of the code, a step-by-step description will now be adhered to. This step-by-step description applies to both the terminal input option and the data file input option.

Step 1 (Input data requested by subroutine INPUT.)

Line 1: IDC

IDC: the desired choice of chord/twist/thickness distribution along the span of the blade.

Value of IDC is:
1. if self generated.
2. if example 1 is used.
3. if example 2 is used.
4. if distribution is from a data file supplied by the user.

If and only if IDC = 4, then enter line 2.

Line 2: DNAME

DNAME: the name of the data file containing the desired chord/twist/thickness distribution data.

Step 2 (Input data, necessary for creating the blade frame, requested by subroutine INPUT.)

Line 1: NAF

Line 1	3	An integer that indicates the method of input of the chord/twist/thickness distribution data. (In this case data from a data file, are being used)
Line 2	9	Number of input airfoils.
Line 3	30.5	Rotor radius, or span.
Line 4	Y	Yes, scale the input airfoils.
Line 5	6	Lines 5 through 9 respectively, represent (1) the desired type of input airfoil, (2) Z coordinate of the span station where this airfoil will be located, (3) the desired choice of specifying twist center X and Y coordinates, (4) the twist center X-coordinate, and (5) the twist center Y-coordinate. This is repeated nine times up to line 49, since the number of input airfoils is equal to nine.
Line 6	4.574	
Line 7	1	
Line 8	0.25	
Line 9	0.	
Line 10	6	
Line 11	7.625	
Line 12	1	
Line 13	0.25	
Line 14	0.	
Line 15	5	
Line 16	9.15	
Line 17	1	
Line 18	0.25	
Line 19	0.	
Line 20	5	
Line 21	13.75	
Line 22	1	
Line 23	0.25	
Line 24	0.	
Line 25	4	
Line 26	16.775	
Line 27	1	
Line 28	0.25	
Line 29	0.	
Line 30	4	
Line 31	19.825	
Line 32	1	
Line 33	0.25	
Line 34	0.	

Note: This data file is continued on the next page.

Figure 17 Sample Input Data File (Example of JAYIN.DAT)

Line 35 Line 36 Line 37 Line 38 Line 39 Line 40 Line 41 Line 42 Line 43 Line 44 Line 45 Line 46 Line 47 Line 48 Line 49	3 22.875 1 0.25 0. 2 25.925 1 0.25 0. 1 28.975 1 0.25 0.	
Line 50	119	The desired number of points to define the airfoil.
Line 51	1.	Tension factor in ξ direction.
Line 52	1.	Tension factor in η direction.
Line 53		Last line should be blank.

Figure 17 (concluded) Sample Input Data File (Example of JAYIN.DAT)

NAF: the number of airfoil data sets to be input and interpolated in order to generate the blade surface.
Note: 1 < NAF < 11

Line 2: SPAN

SPAN: the radius of rotor blade to be generated.

Line 3: SCALE

SCALE: A character specifying whether the scaling of the input airfoils is to be done.

The character of SCALE is: N for no scaling
Y for scaling

Step 3

(Input data, necessary for creating the blade frame, requested by subroutine INPUT. Note that this step is repeated sequentially NAF times.)

Line 1: NDF(K)

NDF(K): the desired airfoil type at station K.

Value of NDF(K) is:
1. for the S806A airfoil.
2. for the S805A/6A airfoil.
3. for the S805A airfoil.
4. for the S805A/7A airfoil.
5. for the S807 airfoil.
6. for the S808 airfoil.
7. for own entry of airfoil.

If and only if NDF(K) = 7, then enter line 2.

Line 2: NAME(K)

NAME(K): the name of the data file containing normalized coordinate data of desired airfoil at station K.

Line 3: ZZ(K)

ZZ(K): the Z coordinate at span station K.

If and only if IDC = 1, then enter lines 4, 5, and 6.

Line 4: TH1(K)

TH1(K): the twist angle, in degrees, of airfoil at station K.

Line 5: CL(K)

CL(K): the chord length of the airfoil at station K.

Line 6: THKN(K)

THKN(K): the maximum thickness/chord ratio of the airfoil at station K.

Line 7: TWCTR

TWCTR: the desired choice of specifying twist center X and Y coordinates.

Value of TWCTR is
1. to be input by the user.
2. to be determined by the program. NOTE: Option 2 will place the twist center at the intersection of X = 1/3 and the airfoil meanline.

If and only if TWCTR = 1, then enter lines 8 and 9.

Line 8: OSX(K)

OSX(K): the X-coordinate of the twist center.

Line 9: OSY(K)

OSY(K): the Y-coordinate of the twist center.

Step 4

(Input data, necessary for the tension spline interpolation, as requested by subroutine INPUT.)

Line 1: N1

N1: the desired number of points to define airfoil. The value entered must be an odd integer number.

Line 2: ALPHA

ALPHA: the interpolation tension factor in the ξ direction. The standard value is 1.

Line 3: BETA

BETA: the interpolation tension factor in the η direction. The standard value is 1.

Step 5	(Stretching parameter requested by subroutine TRANS for concentrating data points about leading edge.)

Line 1: TAU

TAU: the stretching parameter used for concentrating the data points about the leading edge of the airfoil. The standard value is between 4.0 and 6.0, depending upon how many points (N1) are used to define the airfoil. Note: This parameter must be input at the terminal.

After step 4 of input, the program creates the frame of the blade according to the method of Chapter 3. On completion of the blade frame creation, the transformation module is entered. Step 5 reads the data needed to perform the transformations. There is no more required input, and after the program has performed the transformation (according to the methodology of Chapter 4), control passes to the interpolation module.

6.3.2 Output Options

In the interpolation module, the code gives the user two interpolation options:

(1) Generating an airfoil at any user-prescribed station along the span of the blade.

(2) Generating a three-dimensional blade, represented by a user-specified number of airfoils place equally spaced along the span of the blade.

The output that the user can view on the screen, plot, and print is dependent upon which interpolation option the user selects.

Option 1 will furnish the user with

(a) A concentric plot of the input airfoils forming the blade frame (normalized with respect to the span of the blade).

(b) A span-normalized plot of the airfoil at the prescribed span station.

(c) A chord-normalized plot of the airfoil at the prescribed span station, with the airfoil's twist and offset removed.

(d) A printout of tables summarizing the input and generated airfoil data.

Option 2 will furnish the user with

(a) A concentric plot of the input airfoils forming the blade frame (normalized with respect to the span of the blade).

(b) A concentric plot of the user-specified number of airfoils forming the generated blade (normalized with respect to the span of the blade).

(c) A perspective view of the span-normalized generated blade.

(d) A top view of the span-normalized generated blade.

(e) A front view of the span-normalized generated blade.

(f) A plot comparing the chord distribution of the airfoils along the span of the generated blade with that along the span of the input data blade frame.

(g) A plot comparing the twist distribution of the airfoils along the span of the generated blade with that along the span of the input data blade frame.

(h) A plot comparing the maximum thickness/chord ratio distribution of the airfoils along the span of the generated blade with that along the span of the input data blade frame.

(i) A printout of tables summarizing the input and generated airfoil data.

For each plot, the user is given the option of viewing the plot on the screen. The user is also offered hard-copy options, in which they can (1) turn down the offer; (2) send a hard-copy of the plot to the plotter; or (3) send a hard-copy of the plot to the printer.

The BLADE code then gives the user the option to interpolate the same input airfoils again. If so, the user is returned to the interpolation options menu. If not, the program asks whether the user wants to interpolate a new set of input airfoils.

An affirmative answer will loop the program back to the INPUT module, where the whole sequence of interaction described in Section 6.3 is re-initiated. A negative reply will stop the execution of the BLADE code and cause the program to exit to DOS.

6.4 Sample Run of the Program

A sample run of the program using the INPUT DATA option 1 (see Section 6.3.1) is now given below. In this example, three airfoils (S807, S805A, and S806A) were used as input to form a 60-meter-long blade. The airfoils and their respective Z-coordinates, twist angles, chord lengths, maximum thickness/chord ratios, and twist center X and Y coordinates, are shown in Figure 18.

SPAN STATION NUMBER	AIRFOIL TYPE	Z-Coord in meters	TWIST in degrees	CHORD in meters	MAX Th/Ch RATIO	TWIST CENTER (X,Y)
1	S807	0.0	25.0	6	.25	(0.25,0.0)
2	S805A	20.0	10.0	5	.2	(0.25,0.0)
3	S806A	60.0	0.0	2	.15	(0.25,0.0)

Figure 18 Data Used to Make the Sample Run

Here is the sample run of the program. The numbers in **boldface and underline** *are the input values.*

This program uses certain cross-sectional airfoil data as input, and proceeds to interpolate these input data by use of a bi-tension spline method.

As a result of this interpolation, the whole surface of the blade is definable.

This program will then generate airfoil data for
any particular span stations of the blade.

READ AND FOLLOW INSTRUCTIONS CAREFULLY

Press Enter to Continue.

You have two alternative ways of entering data :

Enter T for terminal input

Enter D for data file input

Enter your choice here ---> T

Since this is a free nation , you are given a choice of
how you would like to generate your blade.
Following are your options:

1.....GENERATE YOUR OWN TWISTED AND TAPERED BLADE.
(i.e., The user will input the chord/twist/thickness distribution at the terminal.)

2.....USE Example 1 chord/twist/thickness distribution.

3.....USE Example 2 chord/twist/thickness distribution.

4.....USE A chord/twist/thickness distribution that is not listed here.

NOTE: Option 4 can only be used if you have already
created a chord/twist/thickness distribution data file.
(Please refer to the user's manual on how to generate
this data file.)

Enter your choice here ---> 1

How many airfoils do you want to input to form the blade?
***Enter a number between 2 and 10 ***
---> 3

Enter the span of the desired blade (i.e., the radius)
---> 60.0

Do you want to scale the input airfoils ?

Enter : N for NO
Y for YES

Enter your choice here ---> Y

Below are airfoil types to choose from. Data are contained in data files. They are already normalized with respect to chord.
If you do not desire any of the stated airfoil types, you can choose option 7.

1.....S806A Tangler Somers thin airfoil
2.....S805A/6A " " " "
3.....S805A " " " "
4.....S805A/7A " " " "
5.....S807 " " " "
6.....S808 " " " "
7....None of the above. I will enter my own.

Enter desired airfoil number for position 1 here ---> 5

Reading data from data file S807.DAT
Data from file S807.DAT for airfoil 1 have been read

Press Enter to Continue.

The Z coordinate must be between 0.00 and 60.0

Enter Z coordinate for airfoil 1 here ---> 0.0

Enter the twist angle (in degrees) ---> 25.0

Enter the chord length of airfoil 1 ---> 6.0

Enter the maximum thickness/chord ratio.
The value should be a decimal less than 1.0. ---> 0.25

Enter the twist center X- and Y-coordinates.

Enter 1 if you want to input by yourself

Enter 2 if you want to determine by the program

NOTE: Option 2 only applies to the case where the
twist center is located at the intersection
of X = 1/3 and the meanline.

PLEASE ENTER YOUR CHOICE NUMBER HERE ---> 1

Enter X-coordinate here ---> 0.25

Enter Y-coordinate here ---> 0.0

Please wait, scaling program is running!

Below are airfoil types to choose from. Data are contained in data files. They are already normalized with respect to chord.
If you do not desire any of the stated airfoil types, you can choose option 7.

1.....S806A Tangler Somers thin airfoil
2.....S805A/6A " " " "
3.....S805A " " " "
4.....S805A/7A " " " "
5.....S807 " " " "
6.....S808 " " " "
7....None of the above. I will enter my own.

Enter desired airfoil number for position 2 here ---> 3

Reading data from data file S805A.DAT
Data from file S805A.DAT for airfoil 2 have been read

Press Enter to Continue.

The Z coordinate must be between 0.00 and 60.0

Enter Z coordinate for airfoil 2 here ---> 20.0

Enter the twist angle (in degrees) ---> 10.0

Enter the chord length of airfoil 2 ---> 5.0

Enter the maximum thickness/chord ratio.
The value should be a decimal less than 1.0. ---> 0.2

Enter the twist center X- and Y-coordinates.

Enter 1 if you want to input by yourself

Enter 2 if you want to determine by the program

NOTE: Option 2 only applies to the case where the twist center is located at the intersection of X = 1/3 and the meanline.

PLEASE ENTER YOUR CHOICE NUMBER HERE ---> 1

Enter X-coordinate here ---> 0.25

Enter Y-coordinate here ---> 0.0

Please wait, scaling program is running!

Below are airfoil types to choose from. Data are contained in data files. They are already normalized with respect to chord.
If you do not desire any of the stated airfoil types, you can choose option 7:

1.....S806A Tangler Somers thin airfoil
2.....S805A/6A " " " "
3.....S805A " " " "
4.....S805A/7A " " " "
5.....S807 " " " "
6.....S808 " " " "
7....None of the above. I will enter my own.

Enter desired airfoil number for position 3 here ---> 1

Reading data from data file S806A.DAT
Data from file S806A.DAT for airfoil 3 have been read

Press Enter to Continue.

The Z coordinate must be between 20.00 and 60.0

Enter Z coordinate for airfoil 3 here ---> <u>60.0</u>

Enter the twist angle (in degrees) ---> <u>0.0</u>

Enter the chord length of airfoil 3 ---> <u>2.0</u>

Enter the maximum thickness/chord ratio.
The value should be a decimal less than 1.0. ---> <u>0.15</u>

Enter the twist center X- and Y-coordinates.

Enter 1 if you want to input by yourself

Enter 2 if you want to determine by the program

PLEASE ENTER YOUR CHOICE NUMBER HERE ---> <u>1</u>

Enter X-coordinate here ---> <u>0.25</u>

Enter Y-coordinate here ---> <u>0.0</u>

Please wait, scaling program is running!

ENTER the number of points you want per airfoil:
An ODD integer number between 50 and 120
===> <u>61</u>

WE NOW NEED INPUT FOR BLADE SURFACE INTERPOLATION.
Enter the tension factors in X and Z directions,

respectively.
The numbers must be greater than 0 (zero)
and less than 1000.

ENTER X TENSION FACTOR (standard value is 1) ---> **1.0**

ENTER Z TENSION FACTOR (standard value is 1) ---> **1.0**

PLEASE WAIT.......
CALCULATIONS ARE TAKING PLACE ....

ENTER THE STRETCHING PARAMETER FOR CONCENTRATING
THE AIRFOIL DATA POINTS ABOUT THE LEADING EDGE.

THE STANDARD VALUE RANGES BETWEEN 4.0 AND 6.0
THE FEWER POINTS YOU USED TO REPRESENT THE
AIRFOIL CROSS SECTION, THE SMALLER
VALUE YOU SHOULD USE.

If no stretching is desired, enter 0.0001

Enter your choice here ---> **4.0**

PLEASE WAIT.......
CALCULATIONS IN PROGRESS..............

Beyond this point are the graphic output options, which the user can choose interactively at the terminal.

This now follows on the next page.

YOU ARE NOW GIVEN THE FOLLOWING OPTIONS :
1.....GENERATE AN AIRFOIL AT ANY SPAN STATION.
2.....GENERATE A THREE-DIMENSIONAL BLADE.

NOTE :
Option 1 will furnish the user with an airfoil profile at the span station desired, as well as an airfoil normalized w.r.t. chord

Option 2 will interpolate over the whole blade and generate the coordinates for any specified number of span station airfoils.

Output plots you can create using option 2 are

a) Concentric plots of generated airfoils.
b) Perspective view of the generated blade.
c) Top view of the generated blade.
d) Front view of the generated blade.
e) Chord distribution along the span of the blade
f) Twist distribution along the span of the blade
g) Maximum thickness distribution along the span of blade.

Enter your choice number here ---> **2**

You are now given the following three options on how to specify the output X-coordinates of the generated airfoil, which has its twist and offset removed, at the chosen span station.

Option 1....	The X-coordinates will be specified such that there are more points around the leading edge. This feature depends on the stretching parameter previously used.
Option 2....	The X-coordinates will be specified such that they correspond to the STANDARD 57 X-coordinates; see manual.
Option 3....	The X-coordinates will be specified such that they are regularly spaced. The spacing depends on the number of points (to be input next) used to label the X-coordinates.

Enter your choice number here ---> **1**

HOW MANY span stations do you want your blade represented by ?
ENTER a number greater than 3
(less than 40 for viewing 3D plot) ---> **15**

PLEASE WAIT..............

CALCULATIONS UNDER WAY ....

The standard deviations of
Chord = 0.189267E+00
Twist angle = 0.815737E+00
Thickness = 0.621089E-01
Press Enter to Continue.

CONCENTRIC PLOT OF THE INPUT AIRFOILS.
WOULD YOU LIKE TO VIEW THIS ?

ENTER ---> 0 NO.
1 YES. ---> **1**

DO YOU WANT TO SEND THIS VIEW TO A HARD-COPY DEVICE ?

ENTER ---> 0 NO.
1 YES, SEND TO PLOTTER.
2 YES, SEND TO PRINTER.

ENTER YOUR CHOICE NUMBER HERE ---> **1**

PLOTTING............

DO YOU WANT TO SEND THIS VIEW TO A HARD-COPY DEVICE ?

ENTER ---> 0 NO.
1 YES, SEND TO PLOTTER.
2 YES, SEND TO PRINTER.

ENTER YOUR CHOICE NUMBER HERE ---> **0**

A....Concentric plot of generated span station airfoils.
WOULD YOU LIKE TO VIEW THIS ?

ENTER ---> 0 NO.
1 YES. ---> **1**

DO YOU WANT TO SEND THIS VIEW TO A HARD-COPY DEVICE ?

ENTER ---> 0 NO.
1 YES, SEND TO PLOTTER.
2 YES, SEND TO PRINTER.

ENTER YOUR CHOICE NUMBER HERE ---> **0**

B....Normalized top view plot of the generated blade.
WOULD YOU LIKE TO VIEW THIS ?

ENTER ---> 0 NO.
1 YES. ---> **1**

DO YOU WANT TO SEND THIS VIEW TO A HARD-COPY DEVICE ?

ENTER ---> 0 NO.
1 YES, SEND TO PLOTTER.
2 YES, SEND TO PRINTER.

ENTER YOUR CHOICE NUMBER HERE ---> **0**

C....Normalized front view plot of the generated blade.
WOULD YOU LIKE TO VIEW THIS ?

ENTER ---> 0 NO.
1 YES. ---> **1**

DO YOU WANT TO SEND THIS VIEW TO A HARD-COPY DEVICE ?

ENTER ---> 0 NO.
1 YES, SEND TO PLOTTER.
2 YES, SEND TO PRINTER.

ENTER YOUR CHOICE NUMBER HERE ---> **0**

D....The CHORD distribution of the generated blade.
WOULD YOU LIKE TO VIEW THIS ?

ENTER ---> 0 NO.
1 YES. ---> **1**

DO YOU WANT TO SEND THIS VIEW TO A HARD-COPY DEVICE ?

ENTER ---> 0 NO.
1 YES, SEND TO PLOTTER.
2 YES, SEND TO PRINTER.

ENTER YOUR CHOICE NUMBER HERE ---> 0

E....The TWIST distribution of the generated blade.
WOULD YOU LIKE TO VIEW THIS ?

ENTER ---> 0 NO.
1 YES. ---> 1

DO YOU WANT TO SEND THIS VIEW TO A HARD-COPY DEVICE ?

ENTER ---> 0 NO.
1 YES, SEND TO PLOTTER.
2 YES, SEND TO PRINTER.

ENTER YOUR CHOICE NUMBER HERE ---> 0

F....The MAXIMUM THICKNESS distribution of the generated blade.
WOULD YOU LIKE TO VIEW THIS ?

ENTER ---> 0 NO.
1 YES. ---> 1

DO YOU WANT TO SEND THIS VIEW TO A HARD-COPY DEVICE ?

ENTER ---> 0 NO.
1 YES, SEND TO PLOTTER.
2 YES, SEND TO PRINTER.

ENTER YOUR CHOICE NUMBER HERE ---> 0

Would you like a printout of the input and/or output data ?
If you want a printout, you need to have plenty of paper and time, because the printout is several pages long.

ENTER ---> 0 NO.
1 YES.

ENTER YOUR CHOICE NUMBER HERE ---> 1

Which data set would you like to have a printout?

1. Input data.

2. Output of the generated airfoil at the chosen span stations with twist and taper.

3. Output of the generated airfoil at the chosen span stations without twist and taper.

Enter your choice number here ---> 2

MAKE SURE THAT THE PRINTER IS READY.

Press Enter to Continue.

Tabular data output from each run of the code are found in the data files called INPUT.DAT, WTOUT.DAT, and NTOUT.DAT. The INPUT.DAT file contains most of the input data information. The WTOUT.DAT file contains the generated airfoil data with twist and taper at the chosen span stations. The generated airfoil data with twist and offset removed at the chosen span stations are given in the NTOUT.DAT file.

CHAPTER 7

RESULTS AND DISCUSSIONS

The computer code, as described in the previous chapter, generates output consisting of

(1) Concentric plots of (a) the input airfoils forming the blade frame and (b) the generated airfoils for any number of span stations to define the blade.

(2) Three-dimensional plot of the generated blade.

(3) Top and front views of the generated blade.

(4) Plots of the generated and the input twist, chord, and maximum thickness distributions of the airfoils along the span of the blade.

(5) Chord-normalized and span-normalized plots of the profile of an airfoil at any span station of the generated blade.

(6) Tabulated data for any span station of the generated blade, consisting of (a) the upper and lower surface coordinates of the chord-normalized and span-normalized airfoils, in standard airfoil data format and (b) the twist, chord-to-span ratio, maximum thickness/chord ratio, and twist center of the respective airfoil.

To demonstrate the feasibility and the usefulness of the computer code developed, the results of generating two wind turbine blade profiles are presented. The two examples are the CARTER and MICON wind turbine blades. For each example, the numerical results are presented in graphic form along with one set of generated airfoil data.

7.1 CARTER Blade

This example is an attempt to generate the profile of a twisted and tapered blade with a span of 30.5 ft. Nine airfoils were used to form the frame of the blade. The types of airfoils used and the Z-coordinates along the span of the blade at which they were placed are shown in Table 1.

Contained in Table 2 are the data that were used as input to the code to represent the chord, twist, and thickness distributions of the airfoils along the span of the blade. These data are used as basis for a one-dimensional tension spline interpolation to determine the chord, twist, and maximum thickness of each airfoil on the blade frame stated in Table 1.

Figure 19 shows a concentric plot of the scaled and twisted input airfoils forming the CARTER blade frame. The "R" in the axis labels of Figure 19 and all subsequent figures represents the span of the blade. Figure 20 shows a concentric plot of airfoils that were generated by applying the surface tension spline on the transformed blade frame. The three-dimensional plot of the generated blade is shown in Figure 21. Figures 22 and 23 present the top and front views of the generated blade. The vertical lines represent the locations of the generated airfoils. The surface of the generated blade appears smooth in the three-dimensional plot, as well as in both the top and front views.

Figure 24 compares the twist distribution along the span of the generated CARTER blade and that along the span of the desired blade. The generated twist distribution, like the desired one, varies between 33 deg at the base of the blade to -1 deg at the tip of the blade.

Table 1 Input Data to Form the CARTER Blade Frame

Span Station Number	Airfoil Type	Z-Coordinate in Feet	Z/Span Coordinate
1.	S808	4.575	0.15
2.	S808	7.625	0.25
3.	S807	9.150	0.30
4.	S807	13.750	0.45
5.	S805A/7A	16.775	0.55
6.	S805A/7A	19.825	0.65
7.	S805A	22.875	0.75
8.	S805A/6A	25.925	0.85
9.	S806A	28.975	0.95

Table 2 Input Chord/Twist/Thickness Data for the CARTER Blade

Z/Span Coordinate	Chord/Span Ratio	Twist Angle in Degrees	Maximum Thickness/ Chord Ratio
.05	.225	33.5	0.22
.10	.210	26.8	0.21
.15	.188	21.0	0.20
.20	.167	15.2	0.19
.30	.133	7.8	0.18
.35	.118	5.0	0.178
.45	.092	1.0	0.172
.50	.08	0.0	0.168
.55	.072	-0.7	0.162
.65	.065	-1.0	0.150
.70	.065	-1.0	0.143
.75	.065	-1.0	0.135
.80	.065	-1.0	0.125
.85	.065	-1.0	0.115
.90	.065	-1.0	0.101
.95	.065	-1.0	0.088

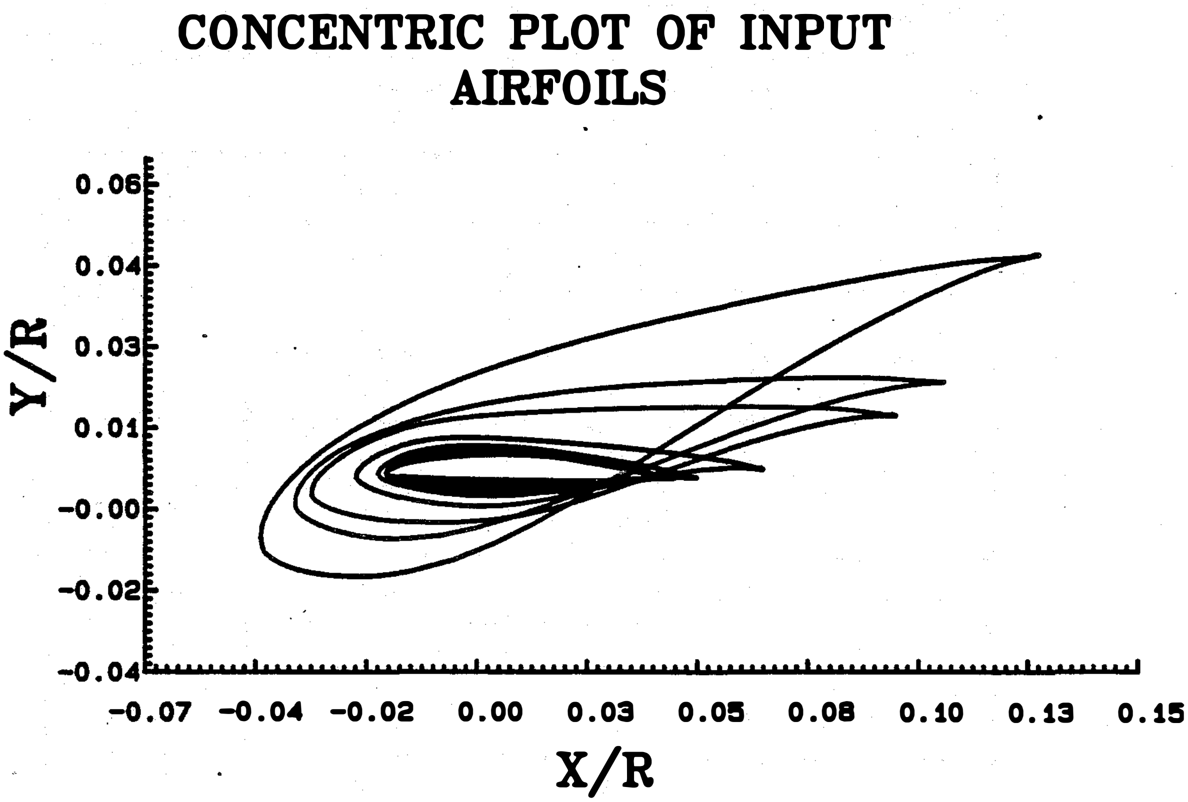


Figure 19 Concentric Plot of the Input Airfoils that Form the CARTER Blade Frame

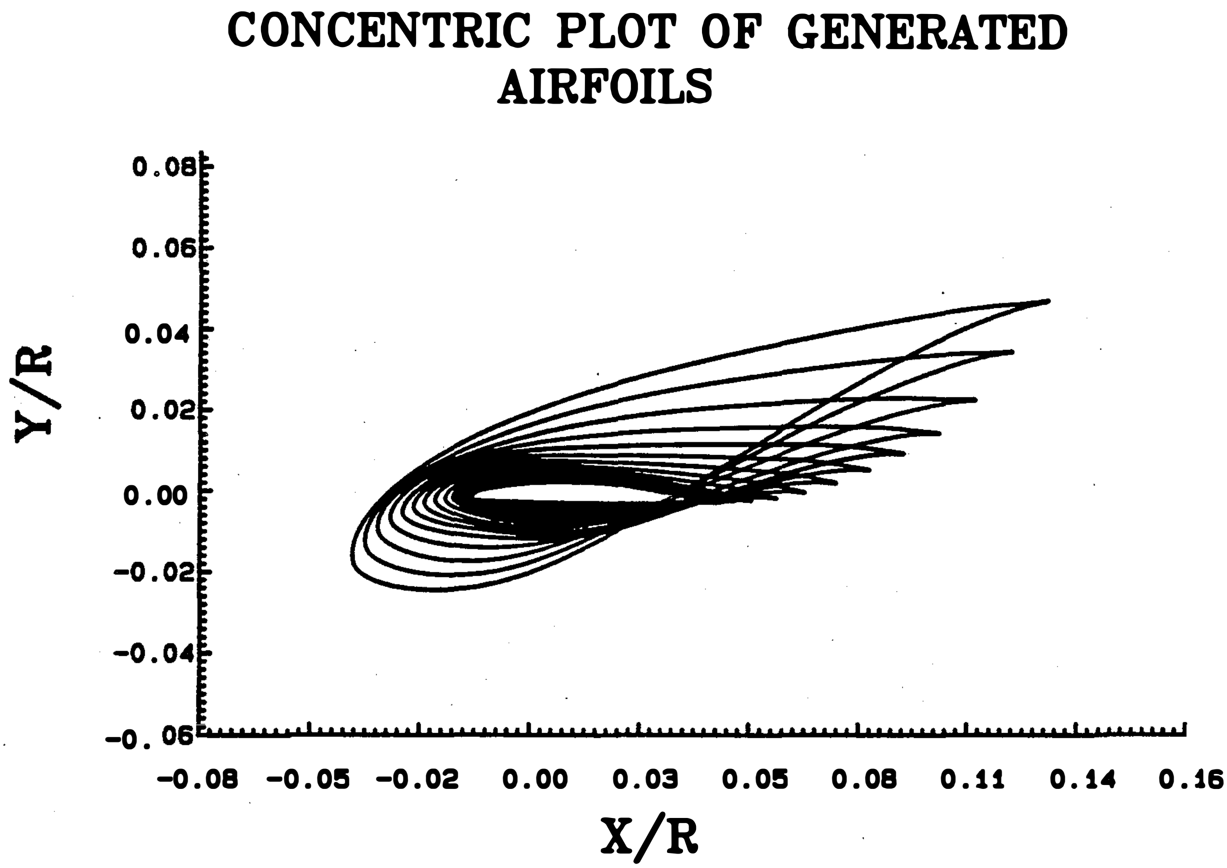


Figure 20 Concentric Plot of the Airfoils Forming the Generated CARTER Blade

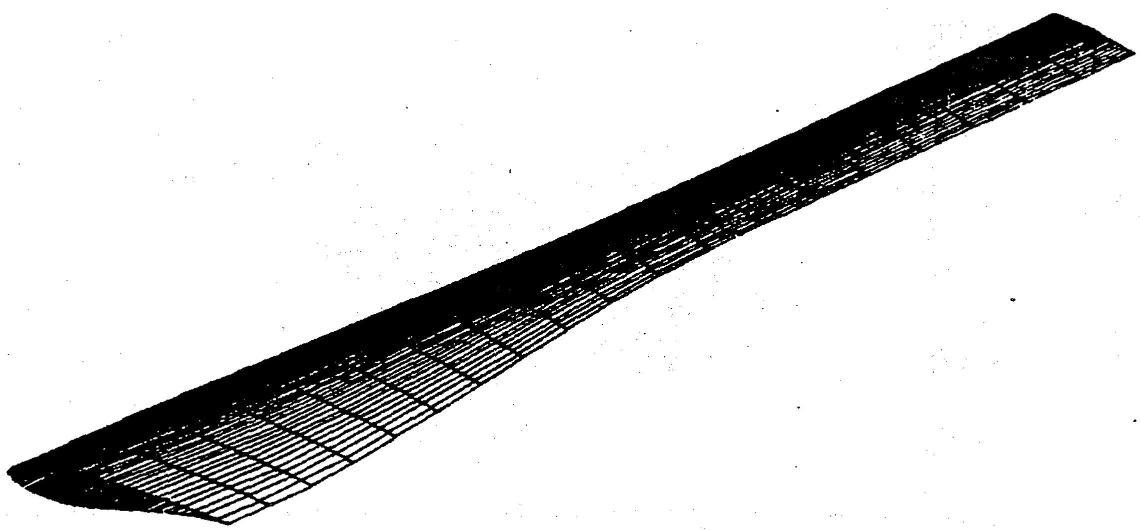


Figure 21   Perspective View of the Generated CARTER Blade

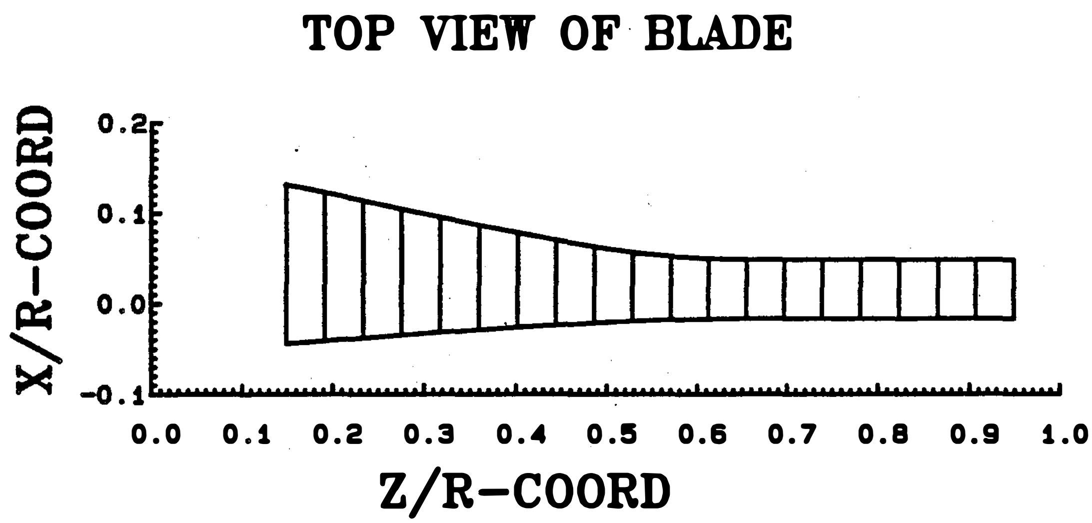


Figure 22 Top View of the Generated CARTER Blade

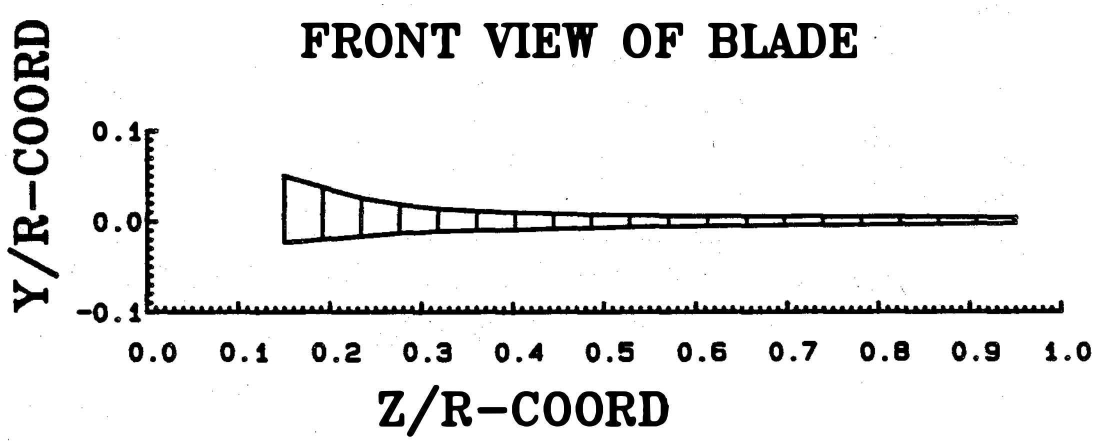


Figure 23 Front View of the Generated CARTER Blade

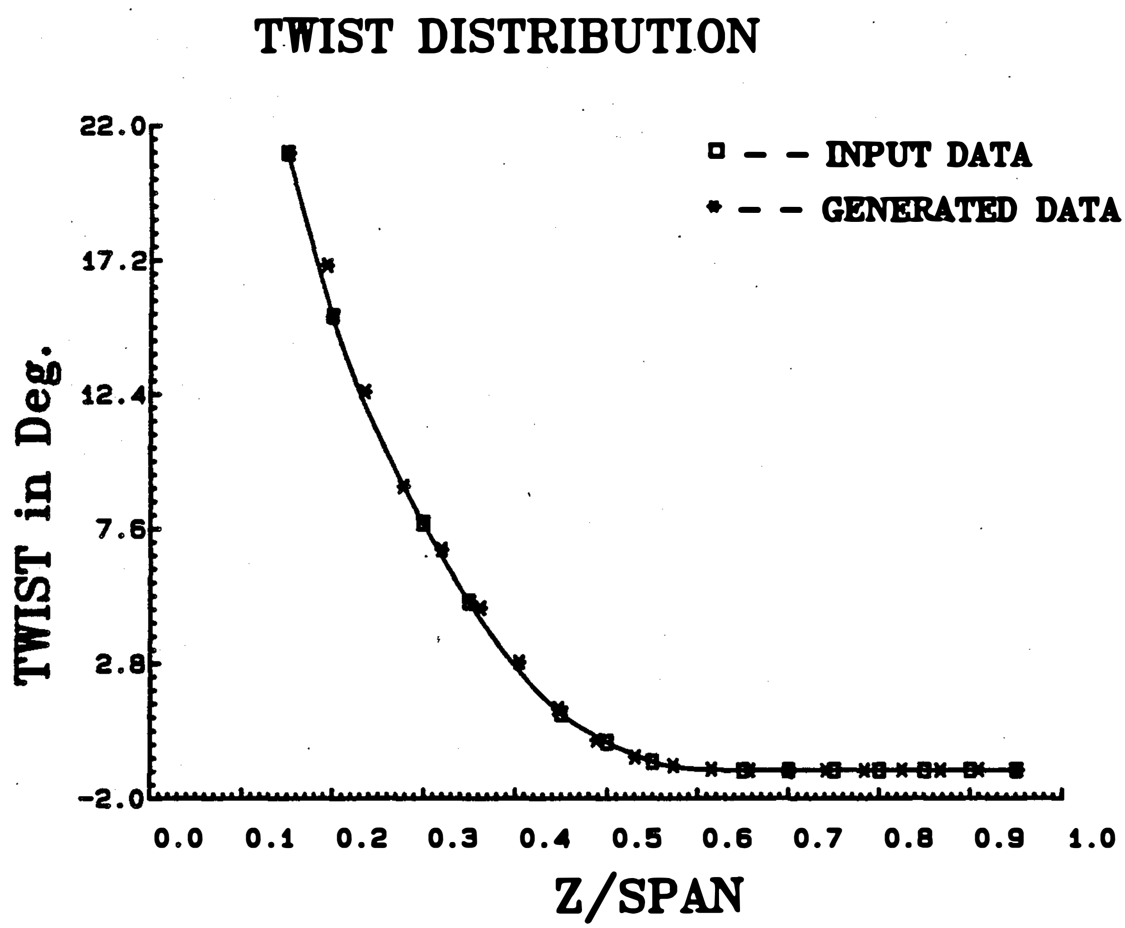


Figure 24 Twist Distribution Along the Span of the CARTER Blade

In Figure 25 the distribution of the chord-to-span ratio along the span of the generated blade is compared to that along the span of the desired blade. The chord-to-span ratio of the airfoil decreases along the span of the blade from root to tip. This taper of the chord is shown in the top view of the blade (Figure 22).

In Figure 26 the distribution of the maximum thickness/chord ratio of the generated blade is compared to that of the desired blade. Like the chord-to-span ratio, the maximum thickness/chord ratio of the airfoils gradually decreases from the root to the tip of the blade. The taper of the blade due to the decrease in maximum thickness of the airfoil is illustrated in the front view of the blade (Figure 23).

The twist, chord, and maximum thickness distributions obtained show a close match between the characteristics of the generated CARTER blade and those of the desired blade. Different distribution characteristics are obtained when the tension parameters of the surface interpolating tension spline are varied. By appropriately varying the tension parameters, generated characteristics that more closely match the desired characteristics can be obtained.

Figures 27-29 represent span-normalized plots of airfoils at Z/span coordinates of 0.20, 0.55, and 0.90, respectively on the span of the generated CARTER blade. As an example of the data output by the code, the actual surface coordinate data of the airfoil with a Z/span coordinate of 0.55 are presented in Figure 30. Figures 31-33 represent chord-normalized plots of the airfoils in Figures 27-29, but without the twist and offset. Because of the removal of the twist and offset, the surface coordinate data of these airfoils can be presented in a form similar to that of standard airfoil data. As an example of this, in Figure 34, the data representing the surface coordinates of the airfoil with a Z/span coordinate of 0.55 are presented in standard form. Any number of airfoils at any span station of the CARTER blade can be generated and have their profiles and data displayed by the code in the manner of Figures 20-34.

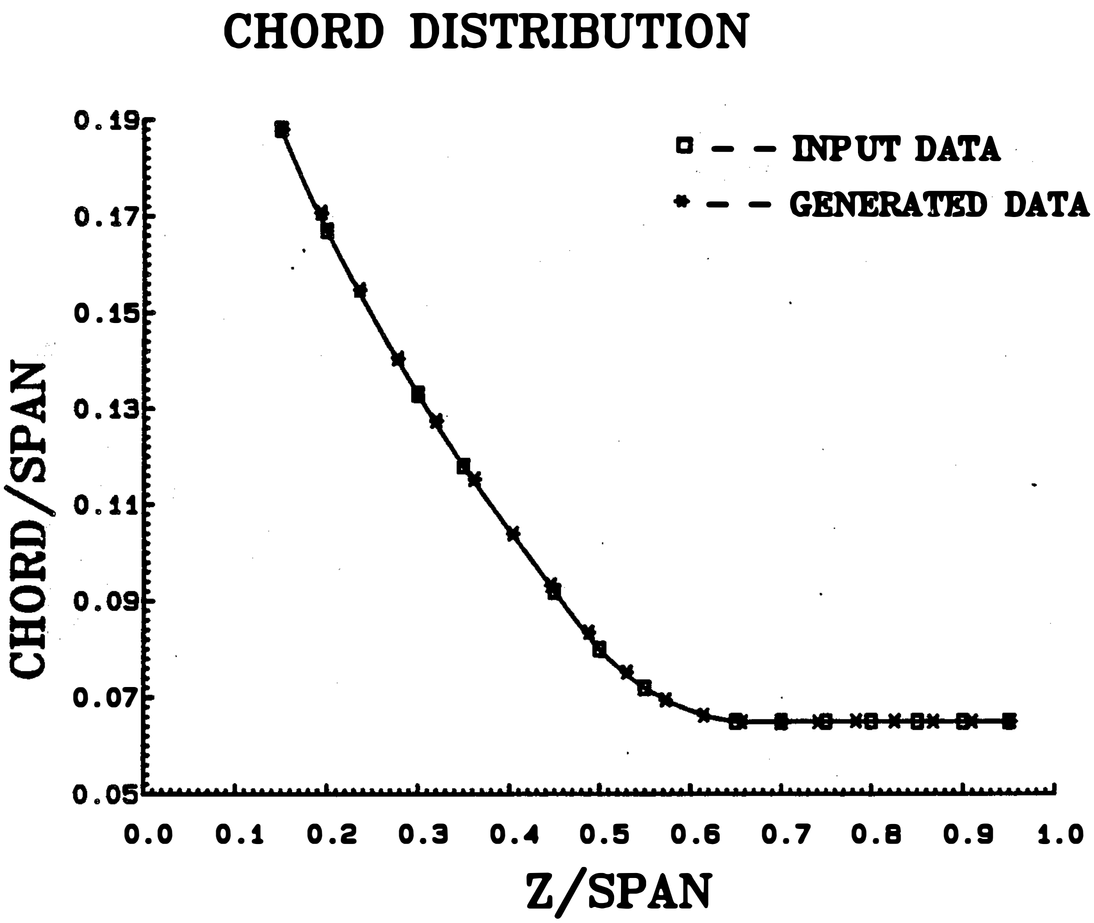


Figure 25 Chord Distribution Along the Span of the CARTER Blade

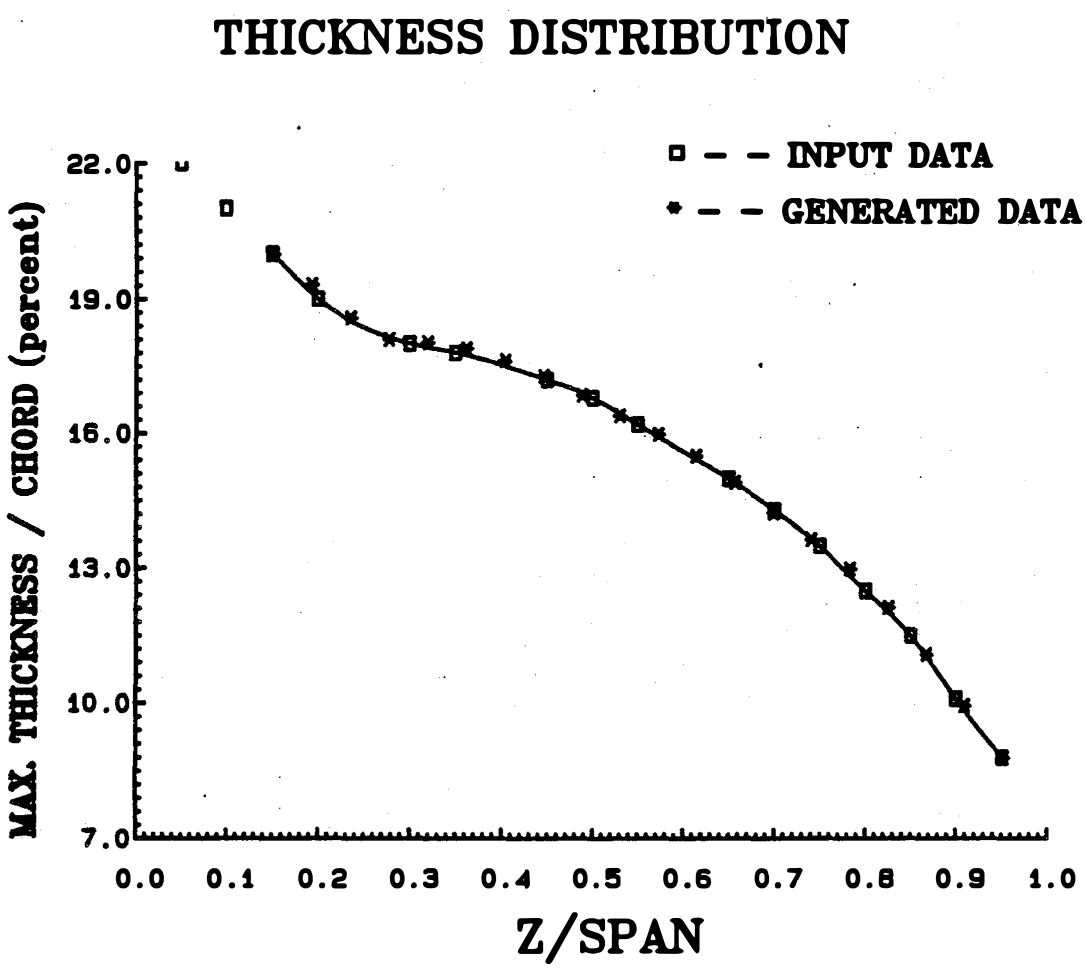


Figure 26 Maximum Thickness Distribution Along the Span of the CARTER Blade

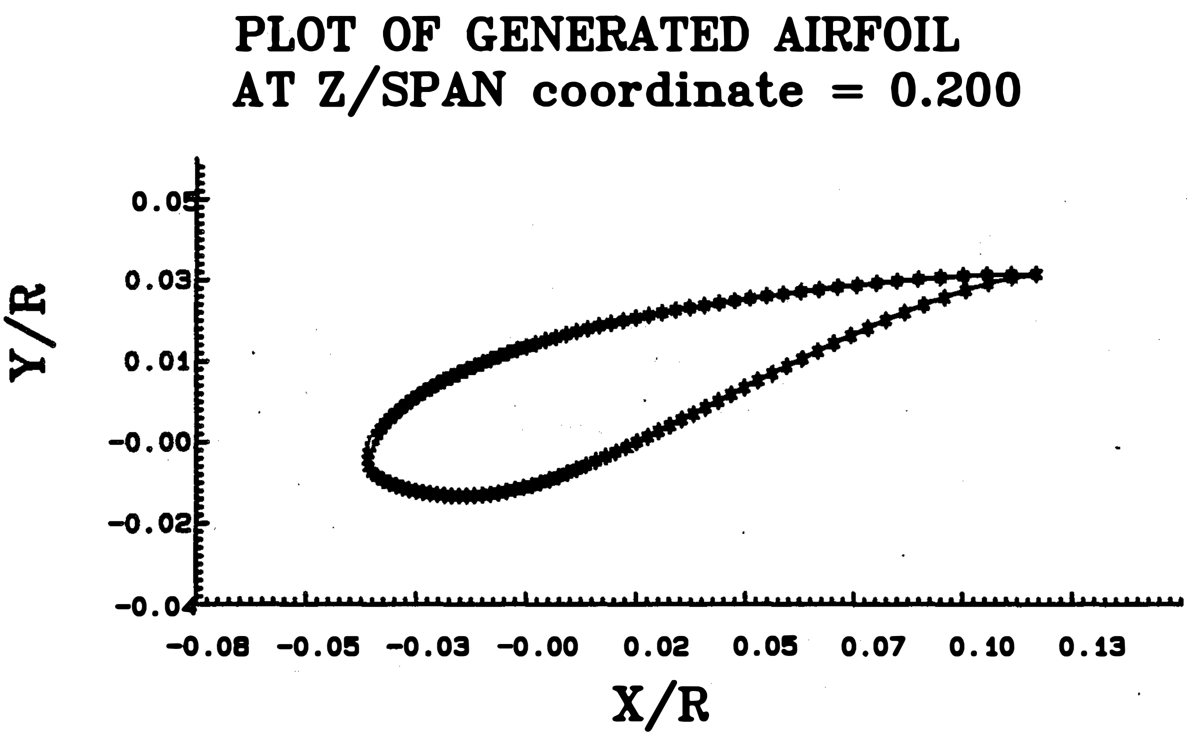


Figure 27 Span-Normalized Airfoil at a Z/span Coordinate of 0.20 on the Generated CARTER Blade

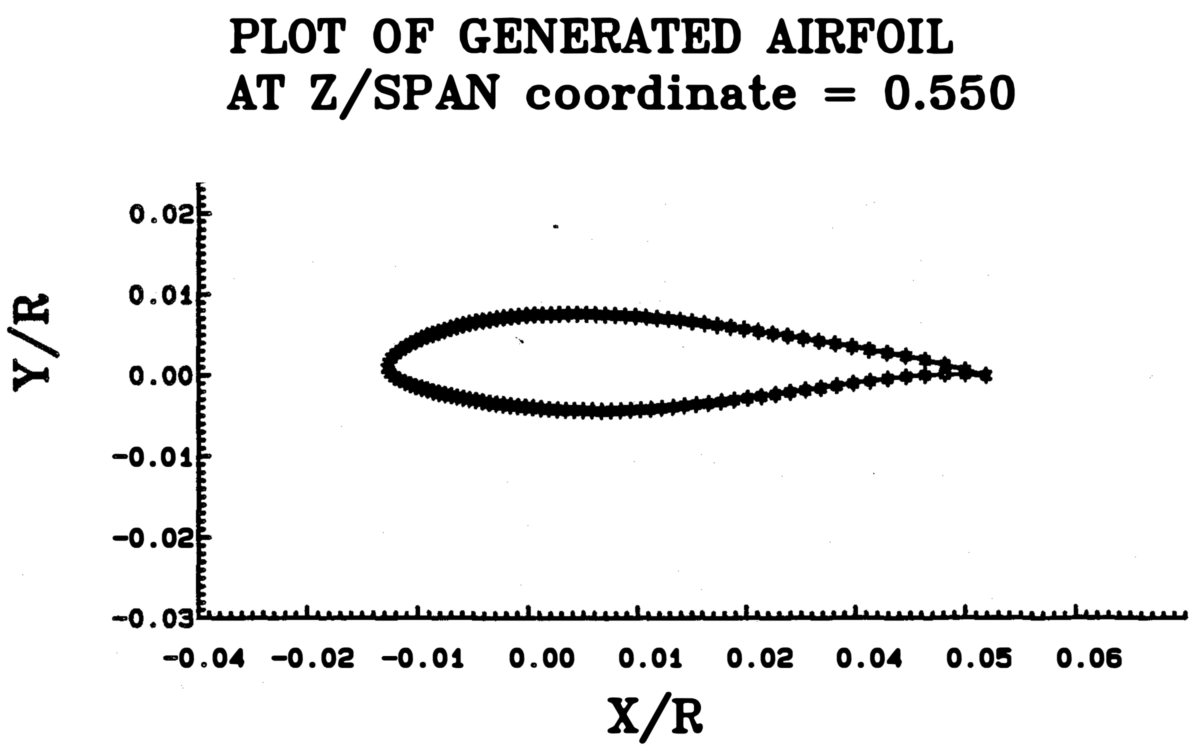


Figure 28 Span-Normalized Airfoil at a Z/span Coordinate of 0.55 on the Generated CARTER Blade

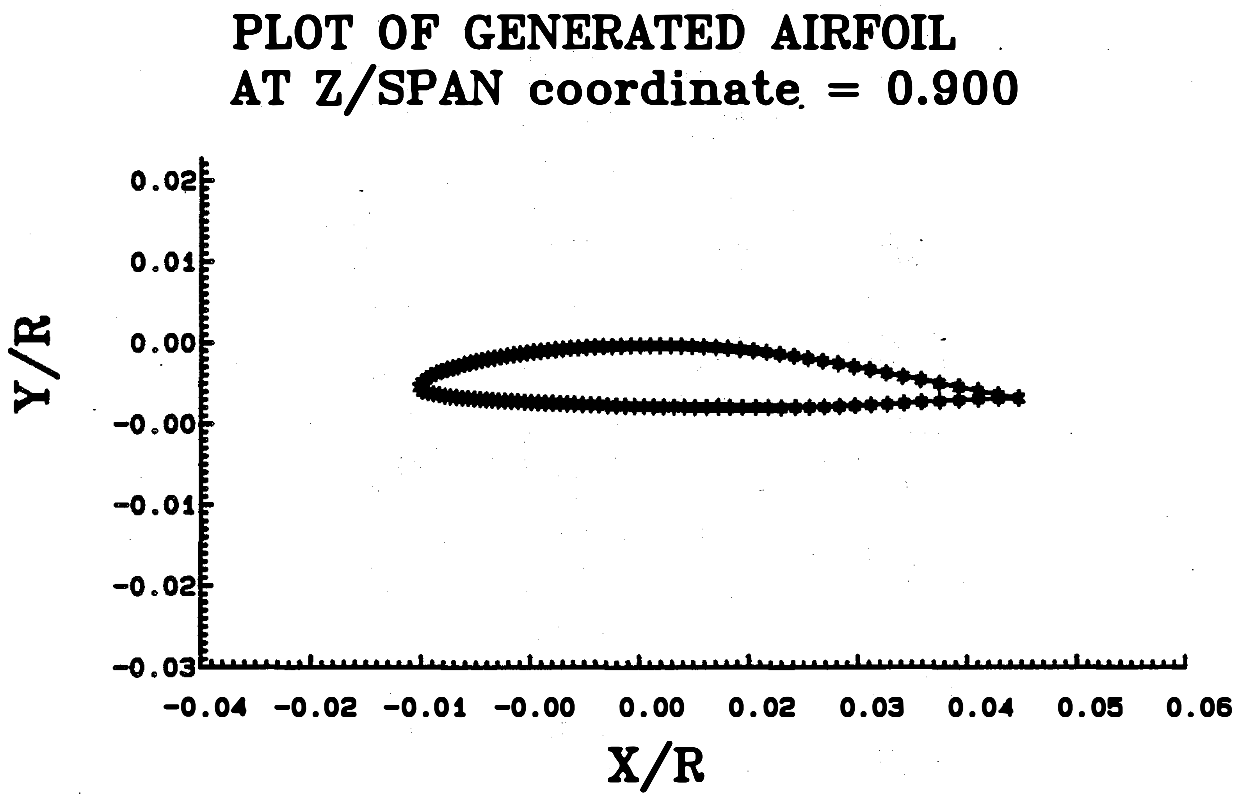


Figure 29 Span-Normalized Airfoil at a Z/span Coordinate of 0.90 on the Generated CARTER Blade

These are the actual coordinates of the airfoil
at span station 16.7750

WITH

Chord-Length	Twist angle	Twist Center X	Twist Center Y	Z/R Coord.	Span R
0.0720	-0.6995	0.5490	0.0000	0.5500	30.5000

The upper and lower airfoil coordinates shown below have been normalized w.r.t. span, R.

XU	YU	XL	YL
-0.01800	0.00022	-0.01800	0.00022
-0.01745	0.00147	-0.01716	-0.00086
-0.01640	0.00239	-0.01593	-0.00152
-0.01521	0.00314	-0.01463	-0.00204
-0.01393	0.00377	-0.01328	-0.00247
-0.01259	0.00431	-0.01188	-0.00285
-0.01119	0.00479	-0.01044	-0.00319
-0.00973	0.00521	-0.00896	-0.00349
-0.00820	0.00558	-0.00742	-0.00378
-0.00661	0.00590	-0.00583	-0.00403
-0.00495	0.00616	-0.00417	-0.00426
-0.00321	0.00637	-0.00244	-0.00447
-0.00138	0.00652	-0.00064	-0.00465
0.00053	0.00662	0.00126	-0.00480
0.00255	0.00667	0.00325	-0.00492
0.00466	0.00668	0.00534	-0.00499
0.00689	0.00664	0.00755	-0.00501
0.00925	0.00655	0.00988	-0.00496
0.01173	0.00641	0.01233	-0.00484
0.01436	0.00621	0.01493	-0.00464
0.01715	0.00593	0.01767	-0.00438
0.02010	0.00558	0.02057	-0.00405
0.02323	0.00515	0.02365	-0.00367
0.02655	0.00466	0.02692	-0.00323
0.03009	0.00410	0.03039	-0.00273
0.03385	0.00348	0.03408	-0.00219
0.03786	0.00280	0.03801	-0.00164
0.04213	0.00203	0.04220	-0.00110
0.04667	0.00114	0.04668	-0.00067
0.05400	-0.00066	0.05400	-0.00066

Figure 30 Span-Normalized Airfoil Data for Airfoil at a Z/span Coordinate of 0.55 on the Generated CARTER Blade

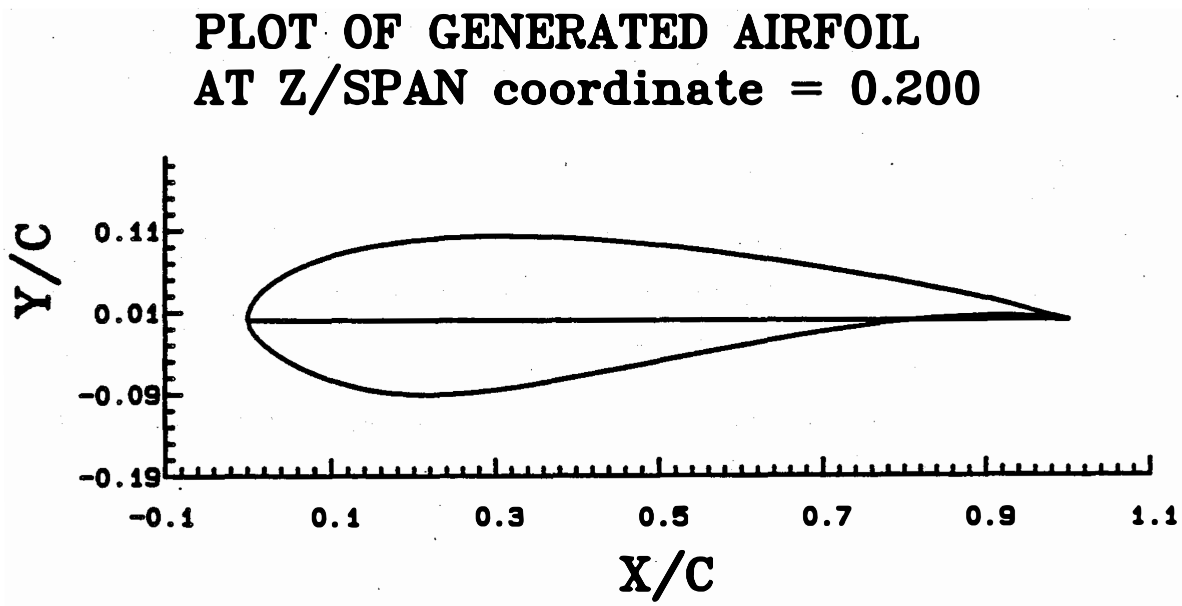


Figure 31 Chord-Normalized Airfoil at a Z/span Coordinate of 0.20 on the Generated CARTER Blade

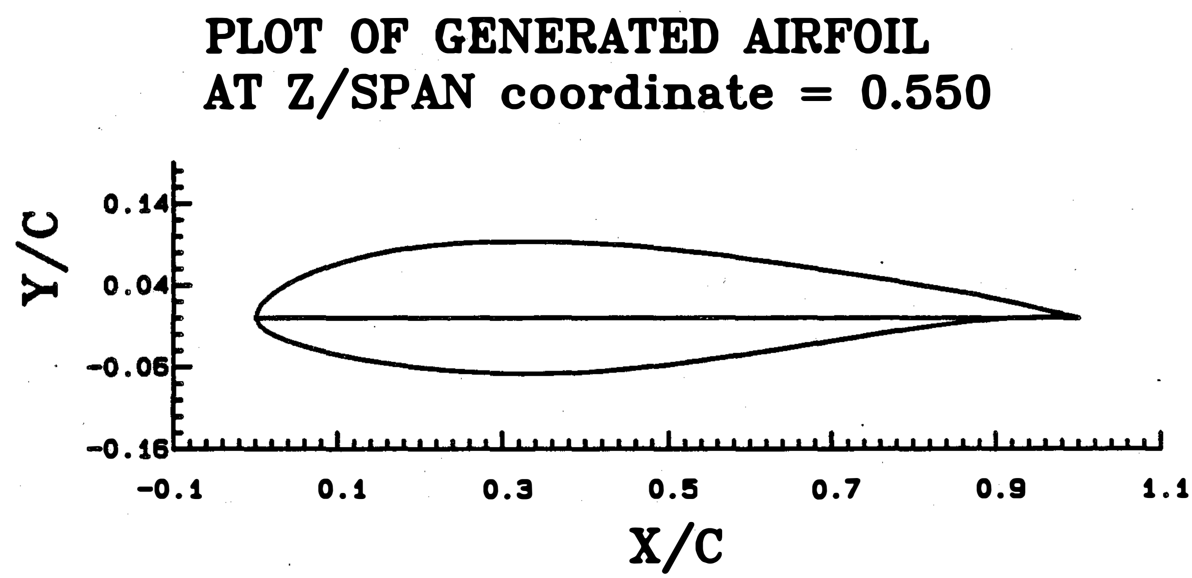


Figure 32 Chord-Normalized Airfoil at a Z/span Coordinate of 0.55 on the Generated CARTER Blade

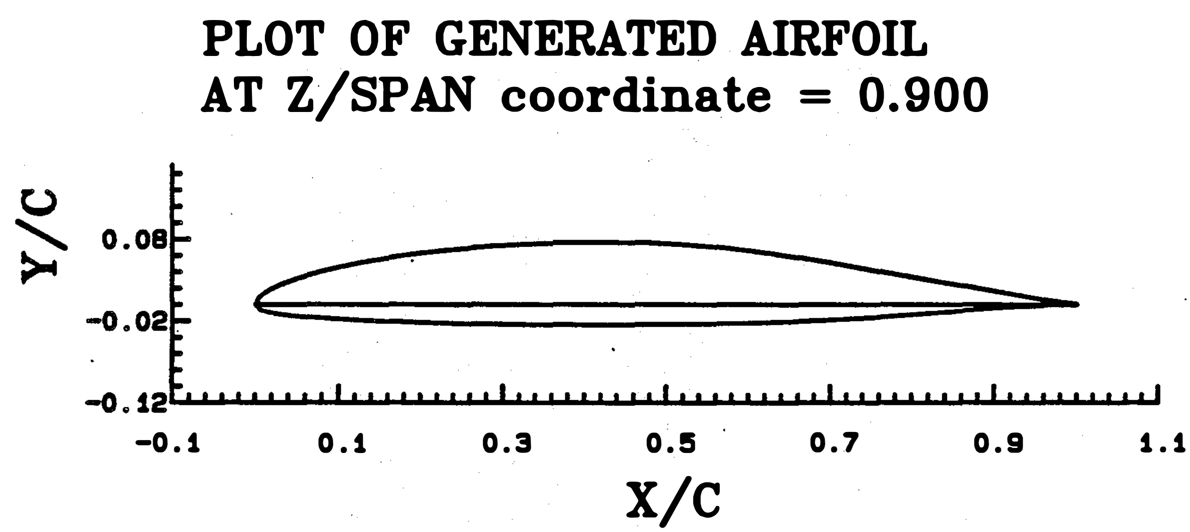


Figure 33 Chord-Normalized Airfoil at a Z/span Coordinate of 0.90 on the Generated CARTER Blade

Chord-Length	Twist angle	Thickness	Twist Center X	Twist Center Y	Z/R Coord.	Span R
0.0720	-0.6995	0.1620	0.2500	0.0000	0.5500	30.5000

The following upper and lower airfoil coordinates are for the airfoil, with no twist, no offset, and are normalized w.r.t. chord = 2.196000

XU	YU	XL	YL
0.00000	0.00000	0.00000	0.00000
0.00050	0.00216	0.00050	-0.00105
0.00100	0.00418	0.00100	-0.00206
0.00200	0.00778	0.00200	-0.00398
0.00500	0.01461	0.00500	-0.00879
0.01000	0.01996	0.01000	-0.01363
0.02000	0.02894	0.02000	-0.01952
0.03000	0.03601	0.03000	-0.02429
0.04000	0.04198	0.04000	-0.02825
0.05000	0.04718	0.05000	-0.03167
0.06000	0.05183	0.06000	-0.03477
0.07000	0.05603	0.07000	-0.03761
0.08000	0.05986	0.08000	-0.04022
0.09000	0.06337	0.09000	-0.04263
0.10000	0.06660	0.10000	-0.04487
0.15000	0.07940	0.15000	-0.05408
0.20000	0.08744	0.20000	-0.06073
0.25000	0.09170	0.25000	-0.06530
0.30000	0.09340	0.30000	-0.06793
0.35000	0.09328	0.35000	-0.06839
0.40000	0.09165	0.40000	-0.06656
0.45000	0.08854	0.45000	-0.06273
0.50000	0.08398	0.50000	-0.05735
0.55000	0.07823	0.55000	-0.05094
0.60000	0.07166	0.60000	-0.04389
0.65000	0.06456	0.65000	-0.03630
0.70000	0.05710	0.70000	-0.02842
0.75000	0.04932	0.75000	-0.02058
0.80000	0.04124	0.80000	-0.01308
0.85000	0.03272	0.85000	-0.00643
0.90000	0.02333	0.90000	-0.00130
0.95000	0.01162	0.95000	0.00142
0.98000	0.00432	0.98000	0.00116
0.99000	0.00212	0.99000	0.00067
1.00000	0.00000	1.00000	0.00000

Figure 34 Chord-Normalized Airfoil Data for Airfoil at a Z/span Coordinate of 0.55 on the Generated CARTER Blade, with the Twist and Offset removed

7.2 MICON Blade

This example is an attempt to generate the profile of another twisted and tapered blade. Ten airfoils were used to form the frame of the 9 meter MICON blade. The types of airfoils and the Z-coordinates along the span of the blade at which they were placed are shown in Table 3.

Input data consisting of the chord, twist, and thickness distributions along the span of the MICON blade are presented in Table 4.

Figure 35 shows a concentric plot of the scaled and twisted input airfoils forming the MICON blade frame. Figure 36 shows a concentric plot of airfoils that were generated by applying the surface tension spline on the transformed blade frame. The three-dimensional plot of the generated blade is shown in Figure 37. Figures 38 and 39 present the top and front views of the generated blade. The surface of the generated blade appears smooth in both the top and front views, as well as in the three-dimensional plot. Figures 40, 41, and 42 compare the twist, chord, and maximum thickness distributions respectively, along the span of the generated blade with those along the span of the desired blade. Again the agreement is excellent.

The generated twist distribution, like the desired one, varies from 16 deg at the base of the blade to 0 deg at the tip of the blade. The chord-to-span ratio of the airfoils decreases in an almost linear manner along the span of the blade from root to tip. This taper of the chord is illustrated well in the top view of the blade (Figure 38). Like the chord-to-span ratio, the maximum thickness/chord ratio of the airfoils gradually decreases from the root to tip of the blade. The taper of the blade due to the decrease in maximum thickness of the airfoils is illustrated in the front view of the blade (Figure 39).

Figures 43-45 represent span-normalized plots of airfoils at Z/span coordinates of 0.20, 0.55, and 0.90, respectively, on the span of the generated MICON blade. The actual surface coordinate data of the airfoil in Figure 44 are presented in Figure 46. Figures 47-49 represent chord-normalized plots of the airfoils in Figures 33-35, but with the twist and offset removed. The chord-normalized surface coordinates of the airfoil in Figure 48 are presented in Figure 50 in a form similar to that of standard airfoil data. Any number of airfoils at any span station of the generated MICON blade can have their profiles and data displayed in the manner of Figures 43-50.

Table 3   Input Data to Form the MICON Blade Frame

Span Station Number	Airfoil Type	Z-Coordinate in Meters	Z/Span Coordinate
1.	S808	1.35	0.15000
2.	S807	2.7	0.30000
3.	S807	3.5	0.38889
4.	S807	4.0	0.44444
5.	S805A/7A	4.5	0.50000
6.	S805A/7A	5.5	0.61111
7.	S805A	6.75	0.75000
8.	S805A/6A	7.5	0.83333
9.	S806A	8.55	0.95000
10.	S806A	8.9	0.98889

Table 4 Input Chord/Twist/Thickness Data for the MICON Blade

Z/Span Coordinate	Chord/Span Ratio	Twist Angle in Degrees	Maximum Thickness/ Chord Ratio
0.144444	0.132	15.9	0.25
0.233333	0.125	12.7	0.20
0.322222	0.117	10.1	0.171
0.414444	0.109	7.5	0.152
0.505556	0.102	5.6	0.137
0.594444	0.093	3.9	0.127
0.683333	0.085	2.7	0.122
0.772222	0.078	1.8	0.118
0.861111	0.069	0.9	0.118
0.925556	0.063	0.45	0.118
0.988889	0.057	0.0	0.118

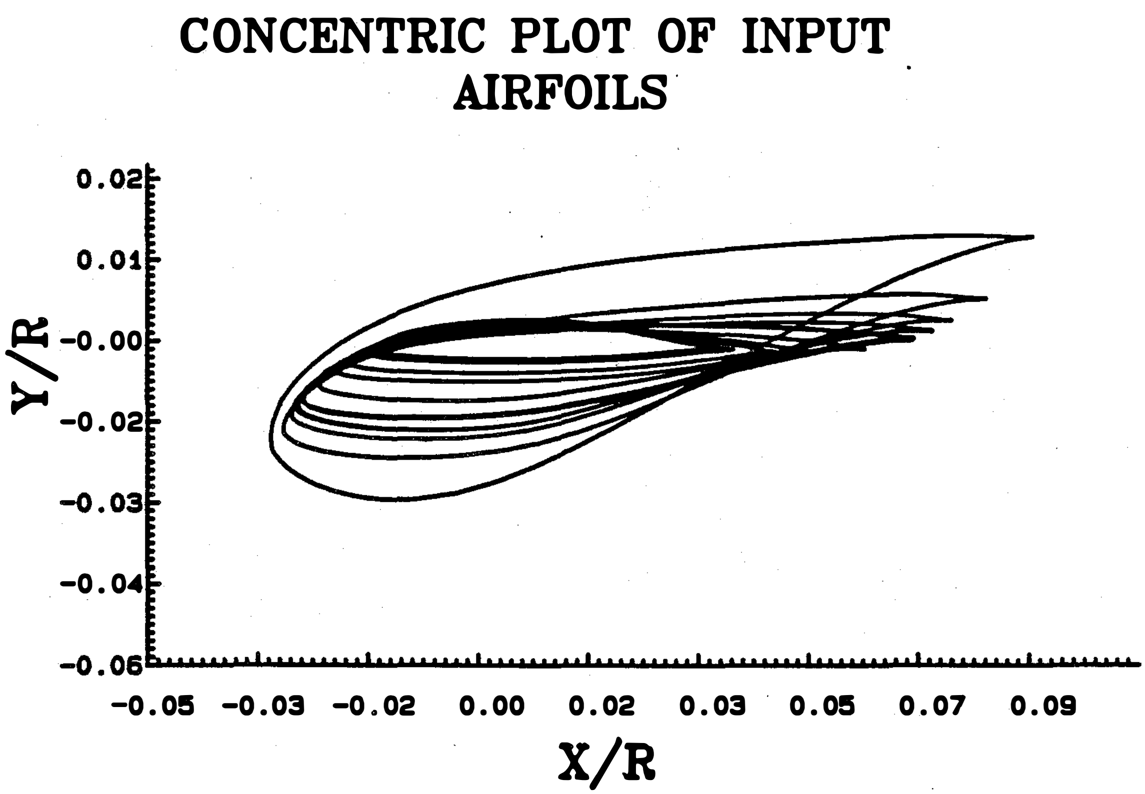


Figure 35 Concentric Plot of the Input Airfoils that Form the MICON Blade Frame

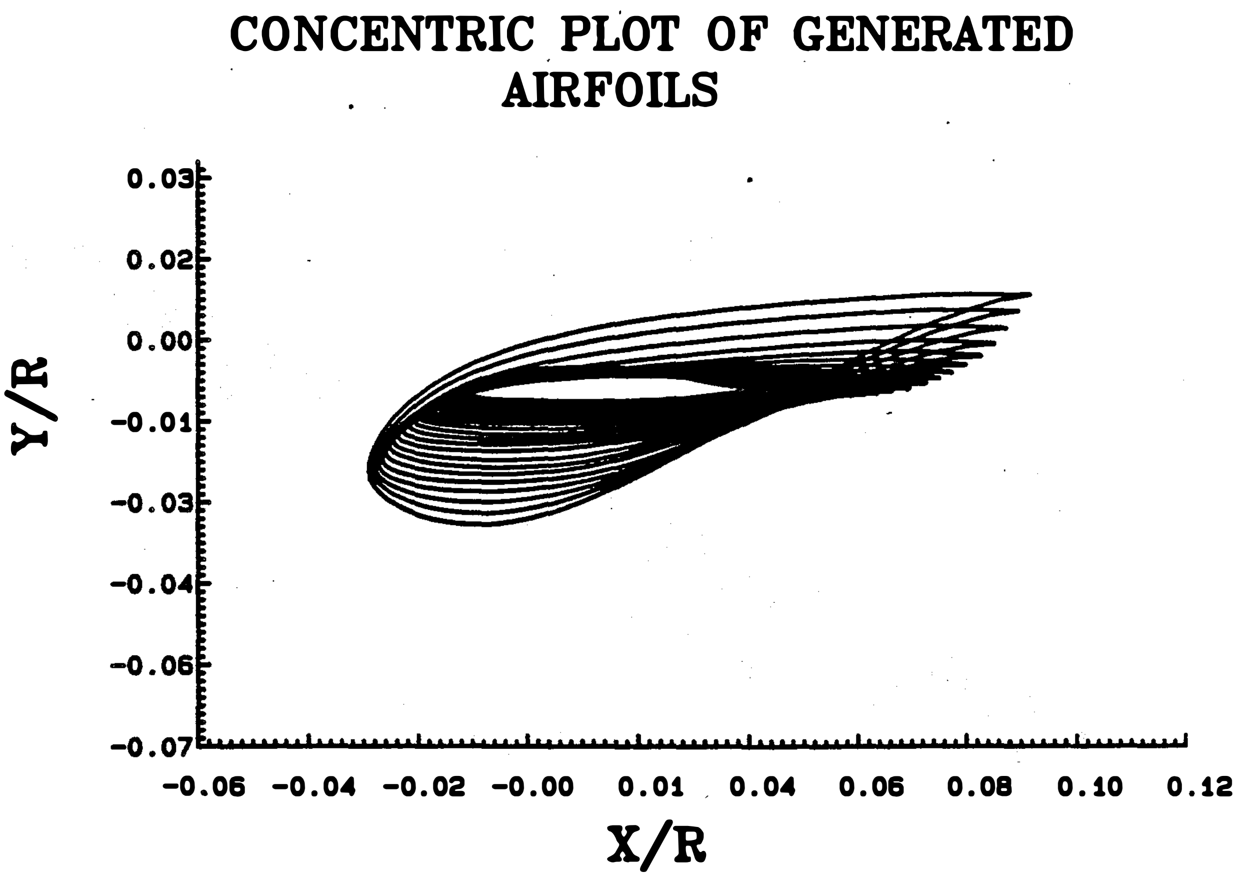


Figure 36 Concentric Plot of the Airfoils Forming the Generated MICON Blade

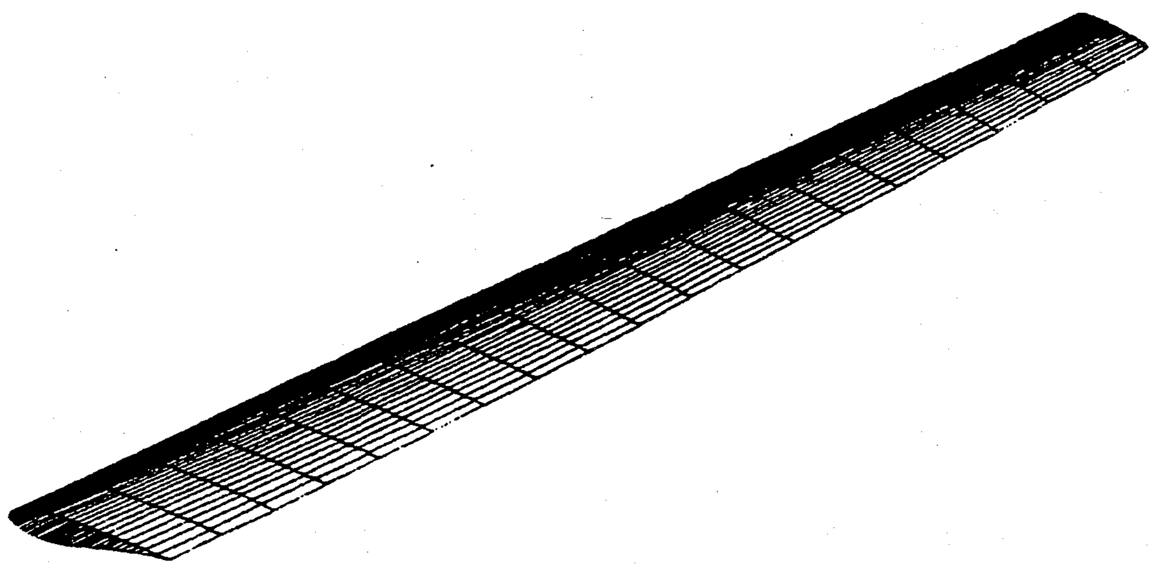


Figure 37 Perspective View of the Generated MICON Blade

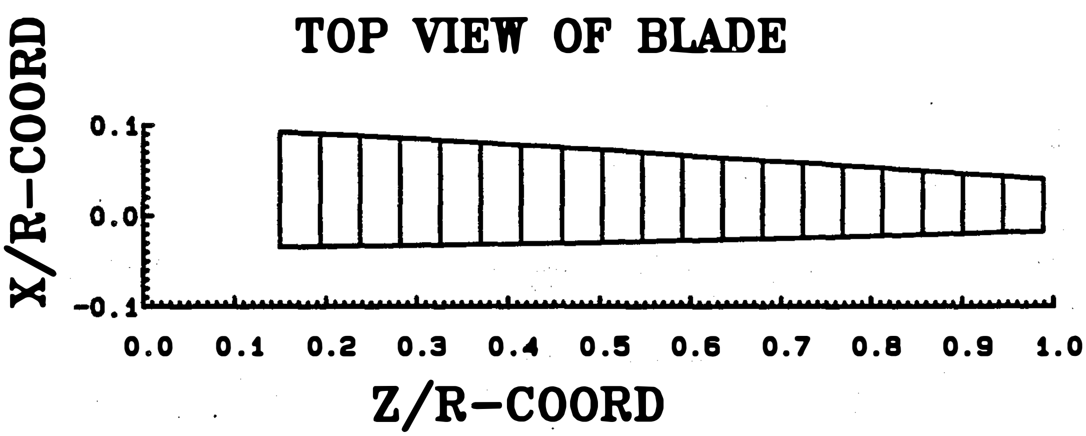


Figure 38 Top View of the Generated MICON Blade

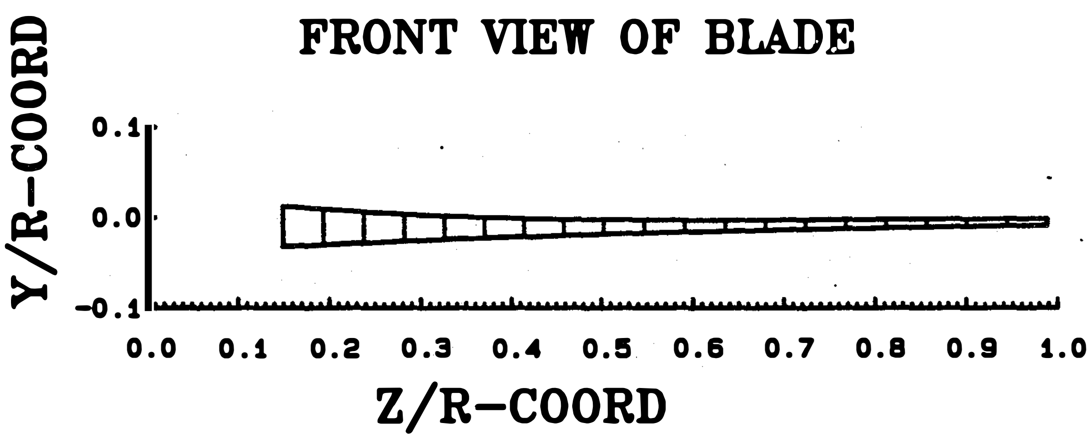


Figure 39 Front View of the Generated MICON Blade

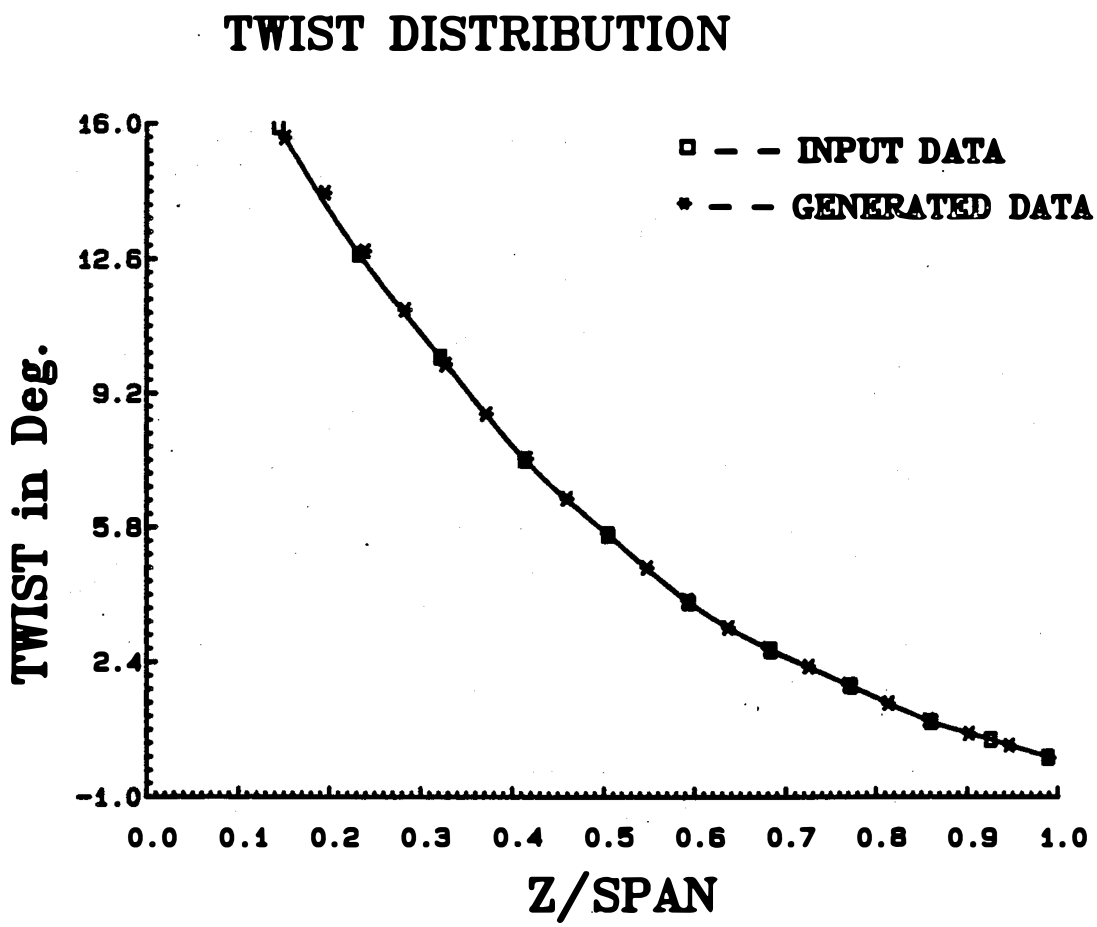


Figure 40 Twist Distribution Along the Span of the MICON Blade

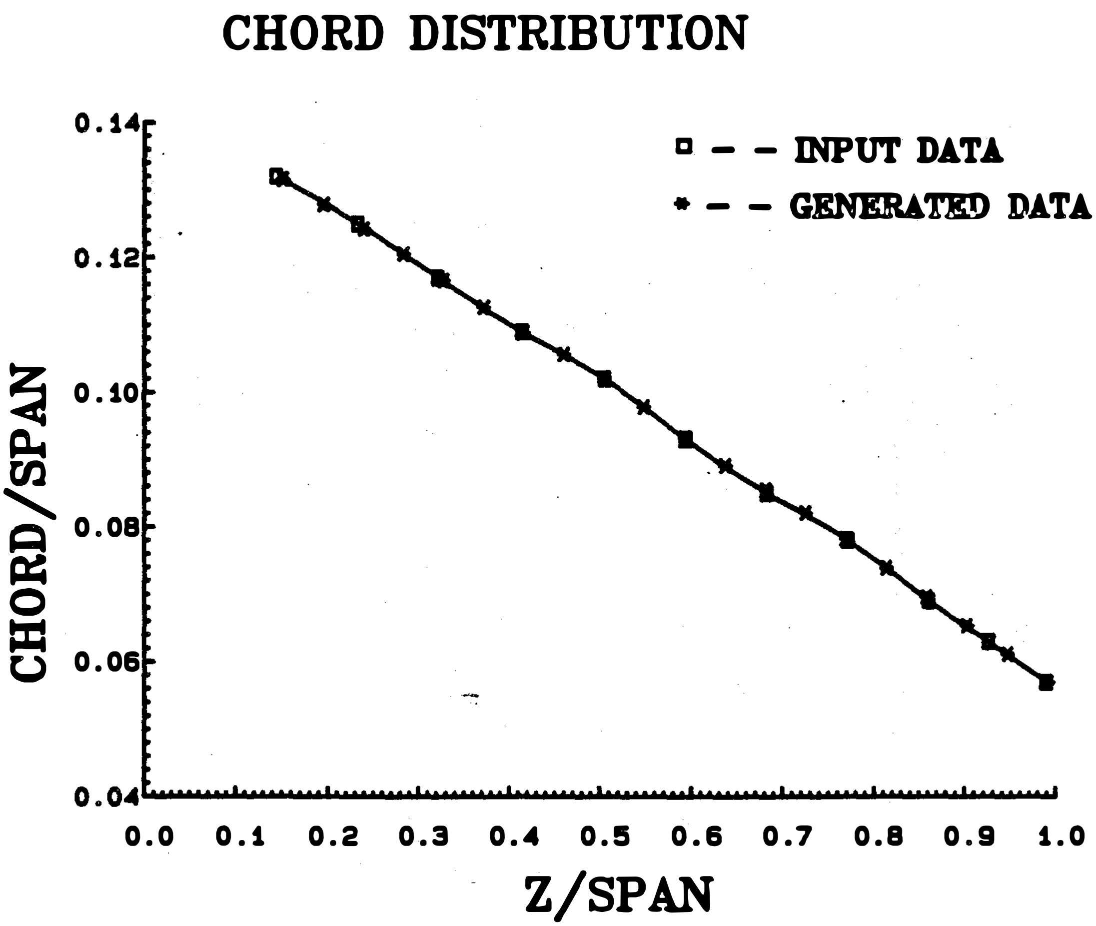


Figure 41 Chord Distribution Along the Span of the MICON Blade

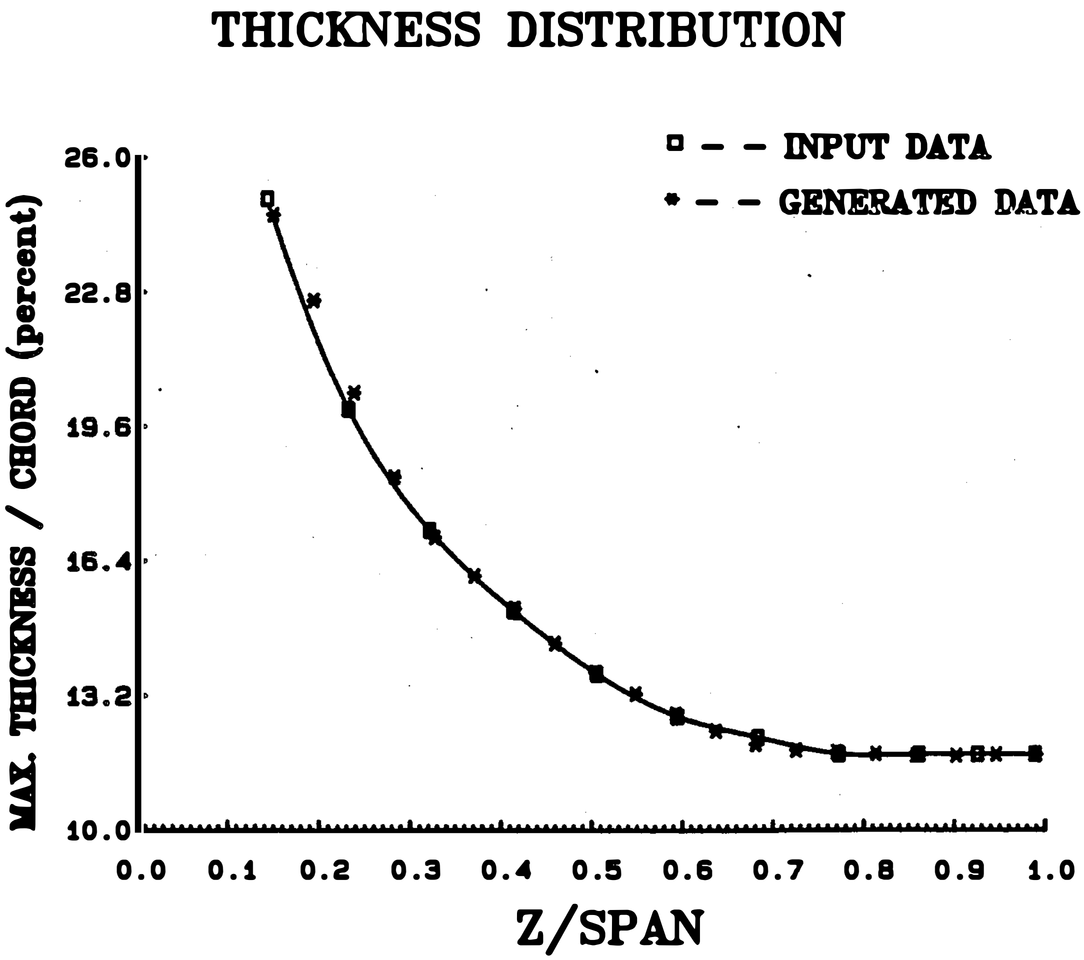


Figure 42 Maximum Thickness Distribution Along the Span of the MICON Blade

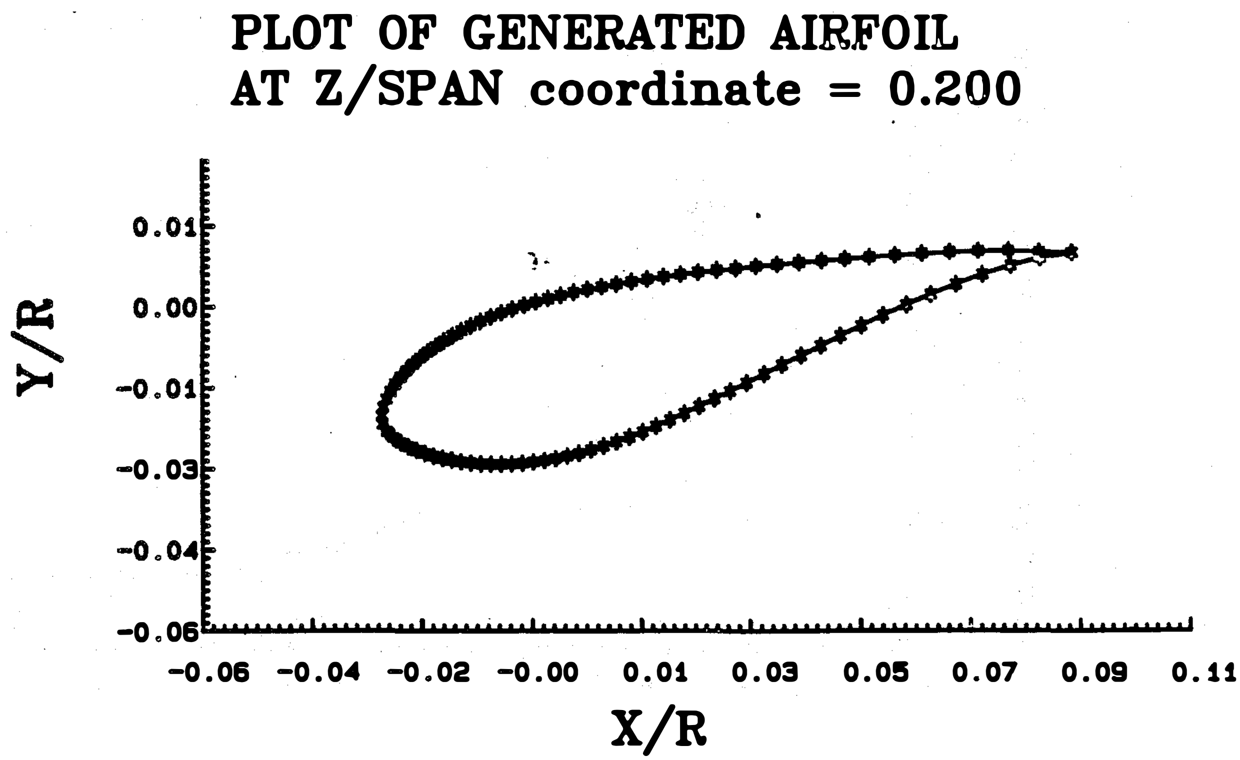


Figure 43 Span-Normalized Airfoil at a Z/span Coordinate of 0.20 on the Generated MICON Blade

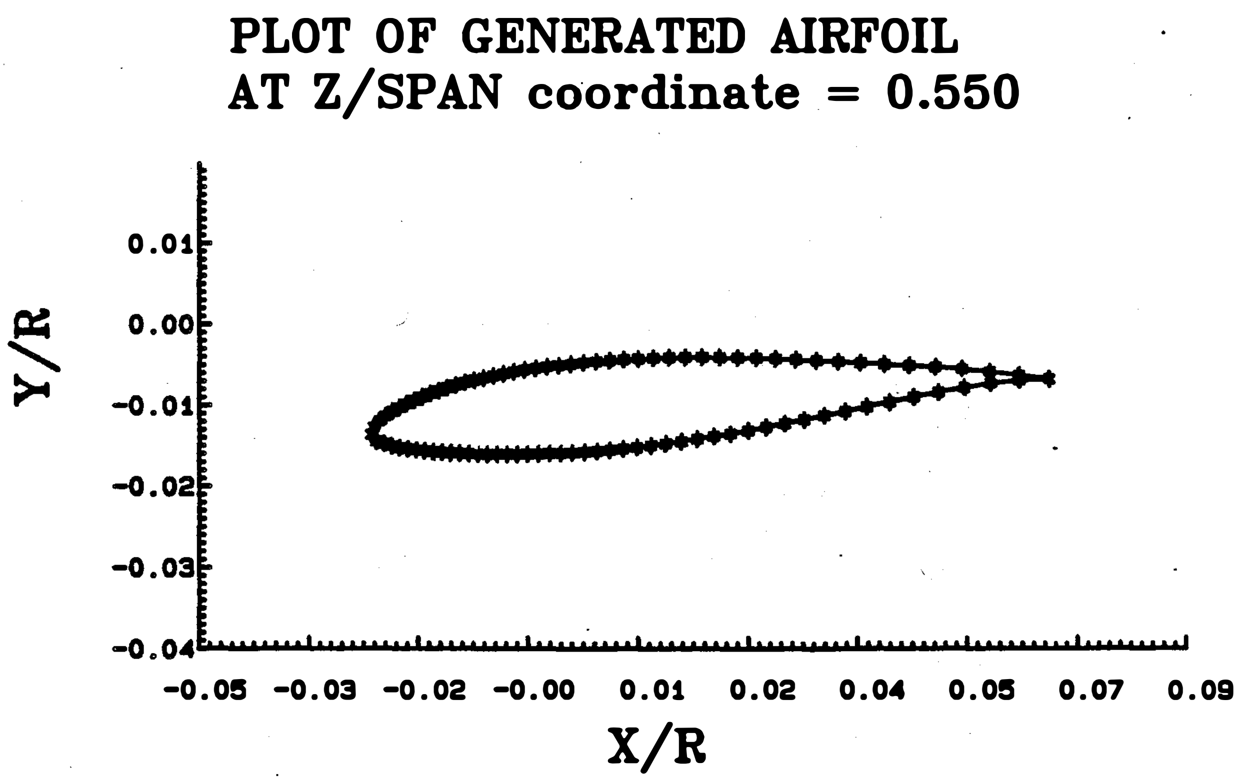


Figure 44 Span-Normalized Airfoil at a Z/span Coordinate of 0.55 on the Generated MICON Blade

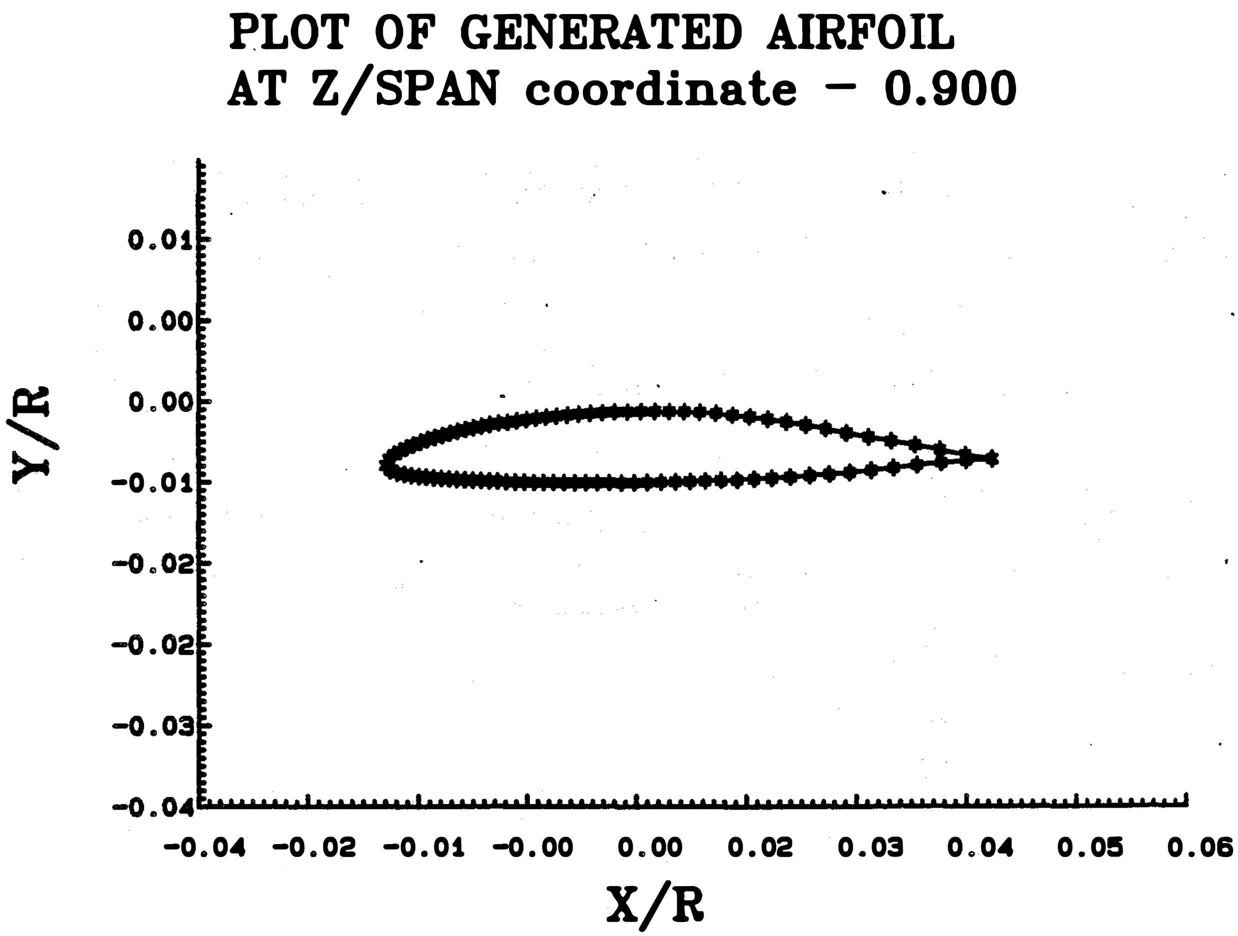


Figure 45 Span-Normalized Airfoil at a Z/span Coordinate of 0.90 on the Generated MICON Blade

These are the actual coordinates of the airfoil
at span station 4.9500

WITH

Chord-Length	Twist angle	Twist Center X	Twist Center Y	Z/R Coord.	Span R
0.0976	4.7321	0.2635	0.1065	0.5500	9.0000

The upper and lower airfoil coordinates shown below have been normalized w.r.t. span, R.

XU	YU	XL	YL
-0.02820	-0.01421	-0.02820	-0.01421
-0.02782	-0.01311	-0.02741	-0.01514
-0.02611	-0.01149	-0.02519	-0.01581
-0.02410	-0.01023	-0.02286	-0.01622
-0.02303	-0.00968	-0.02167	-0.01637
-0.02080	-0.00869	-0.01925	-0.01662
-0.01964	-0.00824	-0.01801	-0.01671
-0.01723	-0.00740	-0.01547	-0.01686
-0.01598	-0.00702	-0.01417	-0.01691
-0.01337	-0.00630	-0.01147	-0.01698
-0.01201	-0.00597	-0.01008	-0.01700
-0.00917	-0.00537	-0.00719	-0.01701
-0.00769	-0.00510	-0.00569	-0.01699
-0.00458	-0.00461	-0.00257	-0.01692
-0.00295	-0.00439	-0.00094	-0.01686
0.00048	-0.00399	0.00248	-0.01668
0.00228	-0.00381	0.00426	-0.01656
0.00607	-0.00350	0.00801	-0.01623
0.01013	-0.00325	0.01202	-0.01578
0.01450	-0.00309	0.01631	-0.01519
0.01681	-0.00305	0.01856	-0.01484
0.02170	-0.00306	0.02332	-0.01405
0.02697	-0.00318	0.02844	-0.01312
0.03267	-0.00340	0.03395	-0.01205
0.03569	-0.00353	0.03685	-0.01146
0.04211	-0.00384	0.04301	-0.01017
0.04906	-0.00423	0.04968	-0.00877
0.05275	-0.00447	0.05322	-0.00807
0.06059	-0.00511	0.06078	-0.00680
0.06906	-0.00616	0.06906	-0.00616

Figure 46 Span-Normalized Airfoil Data for Airfoil at a Z/span Coordinate of 0.55 on the Generated MICON Blade

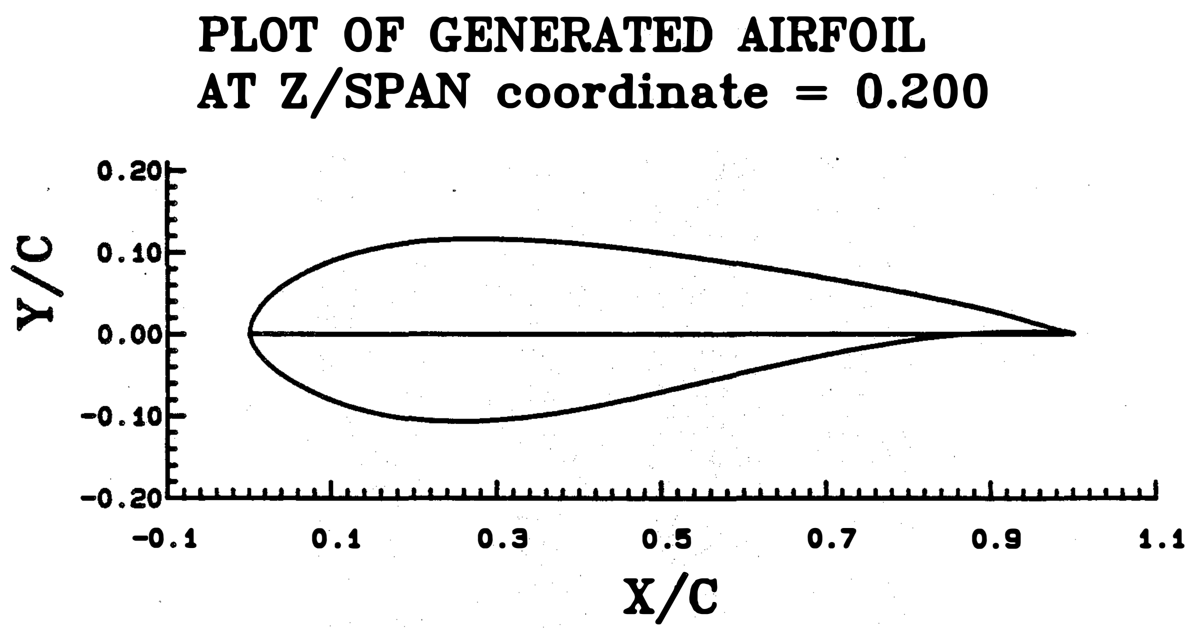


Figure 47 Chord-Normalized Airfoil at a Z/span Coordinate of 0.20 on the Generated MICON Blade

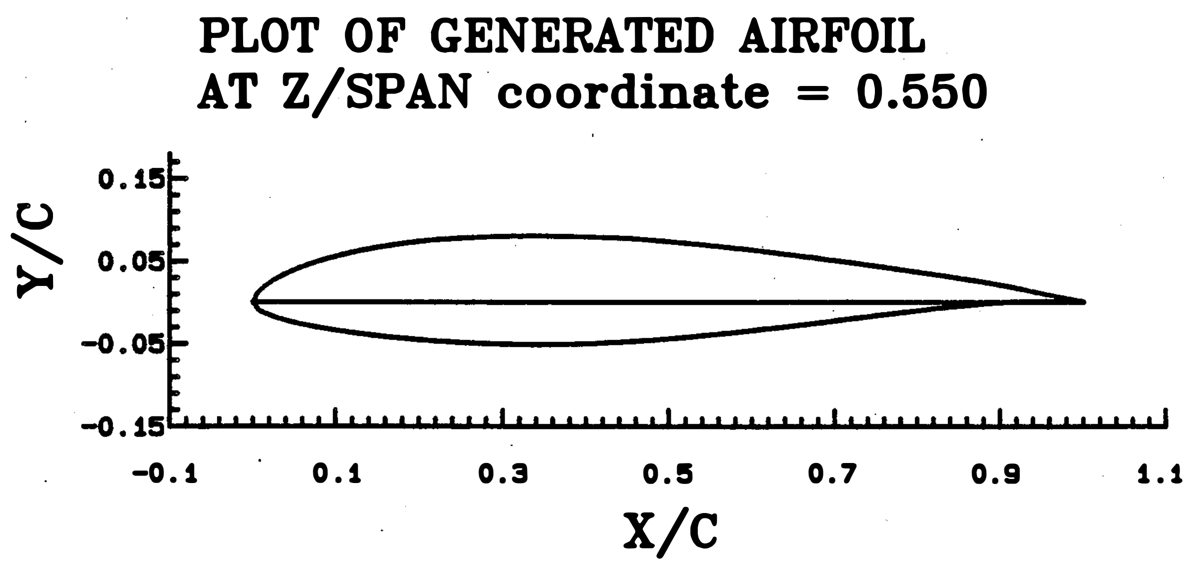


Figure 48 Chord-Normalized Airfoil at a Z/span Coordinate of 0.55 on the Generated MICON Blade

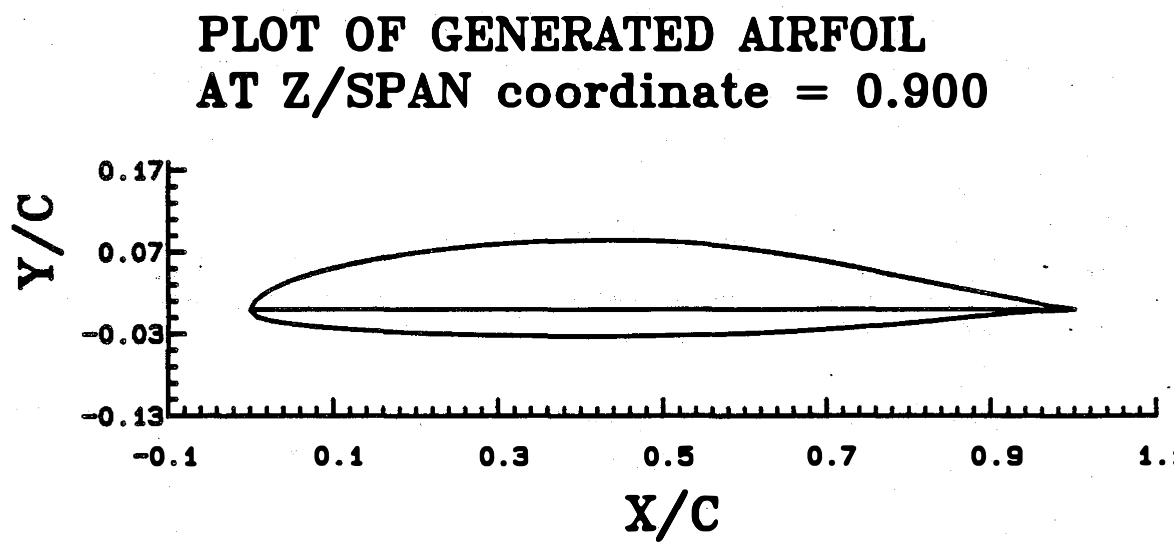


Figure 49 Chord-Normalized Airfoil at a Z/span Coordinate of 0.90 on the Generated MICON Blade

Chord-Length	Twist angle	Thickness	Twist Center X	Twist Center Y	Z/R Coord.	Span R
0.0976	4.7321	0.1320	0.3000	0.1213	0.5500	9.0000

The following upper and lower airfoil coordinates are for the airfoil, with no twist, no offset, and are normalized w.r.t. chord = 0.878370

XU	YU	XL	YL
0.00000	0.00000	0.00000	0.00000
0.00050	0.00135	0.00050	-0.00088
0.00100	0.00265	0.00100	-0.00173
0.00200	0.00512	0.00200	-0.00337
0.00500	0.01120	0.00500	-0.00762
0.01000	0.01668	0.01000	-0.01203
0.02000	0.02337	0.02000	-0.01516
0.03000	0.02973	0.03000	-0.01912
0.04000	0.03469	0.04000	-0.02191
0.05000	0.03920	0.05000	-0.02433
0.06000	0.04317	0.06000	-0.02658
0.07000	0.04675	0.07000	-0.02861
0.08000	0.05003	0.08000	-0.03049
0.09000	0.05304	0.09000	-0.03221
0.10000	0.05581	0.10000	-0.03382
0.15000	0.06687	0.15000	-0.04047
0.20000	0.07404	0.20000	-0.04534
0.25000	0.07816	0.25000	-0.04879
0.30000	0.08014	0.30000	-0.05090
0.35000	0.08053	0.35000	-0.05145
0.40000	0.07958	0.40000	-0.05036
0.45000	0.07723	0.45000	-0.04783
0.50000	0.07345	0.50000	-0.04417
0.55000	0.06850	0.55000	-0.03970
0.60000	0.06276	0.60000	-0.03460
0.65000	0.05654	0.65000	-0.02889
0.70000	0.05002	0.70000	-0.02276
0.75000	0.04324	0.75000	-0.01646
0.80000	0.03620	0.80000	-0.01028
0.85000	0.02878	0.85000	-0.00472
0.90000	0.02058	0.90000	-0.00049
0.95000	0.01029	0.95000	0.00158
0.98000	0.00402	0.98000	0.00108
0.99000	0.00200	0.99000	0.00062
1.00000	0.00000	1.00000	0.00000

Figure 50 Chord-Normalized Airfoil Data for Airfoil at a Z/span Coordinate of 0.55 on the Generated MICON Blade, with the Twist and Offset removed

CHAPTER 8

CONCLUSIONS

A numerical interpolation scheme has been developed that generates the three-dimensional surface of a twisted and tapered wind turbine blade. The scheme involves creating the frame of the blade (Chapter 3); transforming the blade frame from its physical domain to a computational domain (Chapter 4); and performing a bi-tension spline interpolation in the computational domain to determine the physical coordinates of any point on the blade surface (Chapter 2). A FORTRAN computer program has been written to implement the scheme. The program is able to

(1) Input two or more normalized airfoils at some specific spanwise stations as the basis for interpolation.

(2) Scale the input airfoil data according to the prescribed chord and thickness.

(3) Generate the desired three-dimensional blade geometry for linear or nonlinear spanwise chord, thickness, and/or twist distributions.

(4) Graphically output the generated three-dimensional blade profiles, which include the concentric plots of any number of spanwise station airfoils; and a perspective view, top view, and front view, respectively, of the generated blade.

(5) Graphically output plots of the chord, twist, and thickness distribution curves of the generated and input airfoils.

(6) Output in tabular form of all the interpolated airfoil sections specified by the user.

To demonstrate the feasibility, usefulness, and versatility of the numerical scheme, two different aerodynamic blade profiles were generated and presented in Chapter 7. The results of this demonstration indicate that the numerical interpolation scheme can be a powerful tool for generating the surface coordinates for any highly twisted and tapered three-dimensional blade.

REFERENCES

1. Tangler, J. L. and Somers, D. M., "Advanced Airfoils for HAWTs," Wind Power '85 Proceedings, San Francisco, CA, August 1985.
2. Eppler, R., and Somers, D. M., "A Computer Program for the Design and Analysis of Low-Speed Airfoils," NASA TM-80210, 1980.
3. Tangler, J. L., Private Communication, Solar Energy Research Institute, Wind Energy Research Branch, Golden, Colorado.
4. Stoddard, F. S., Private Communication, Pioneer Wind Power, Amherst, Massachusetts.
5. Andrews, B. C., and Van Doren, K. R., "A Method for Geometric Modeling of Complex Wind Turbine Blades," Seventh ASME Wind Energy Symposium, SED-Vol. 5, p. 115, 1988.
6. Cline, A. K., "Scalar- and Planar-Valued Curve Fitting Using Splines Under Tension," Communications of the ACM, Vol. 17, pp. 218-220, 1974.
7. Pruess, S., "Properties of Splines in Tension," Journal of Approximating Theory, Vol. 17, pp. 86-96, 1976.
8. James, M. L., Smith, G. M. and Walford, J. C., Applied Numerical Methods for Digital Computation, 2nd edition, Harper and Row Publishers, New York, 1977.
9. Conte, S. D. and Boor, C., Elementary Numerical Analysis, An Algorithmic Approach, 3rd edition, McGraw Hill Book Company, New York, 1980.
10. Schweikert, D. G., "An Interpolation Curve Using A Spline in Tension," Journal of Mathematics and Physics, Vol. 45, pp. 312-317, 1966.

11. Späth, H., "Algorithm 16: Two-Dimensional Exponential Splines," Computing, Vol. 7, pp. 364-369, 1971.

12. Späth, H., Spline Algorithm for Curves and Surfaces, Utilitas Mathematica Publishing Inc., Winnipeg, Canada, 1974.

13. Anderson, D. A., Tannehill, J. C., and Pletcher, R. H., Computational Fluid Mechanics and Heat Transfer, Hemisphere Publishing Corporation, Washington, 1984.

14. Abbott, I. H., and Von Doenhoff, A. E., Theory of Wing Sections - Including a Summary of Airfoil Data, Dover Publications, Inc., New York, 1959.

15. Morgan, H. L., Jr., "Computer Programs for Smoothing and Scaling Airfoil Coordinates," NASA TM-84666, 1983.

16. Tu, P. K. C., and Scott, G. N., "Smoothing and Scaling Airfoil Coordinates on a Personal Computer," SERI/TR-257-3372, Solar Energy Research Institute, Wind Research Branch, Golden, Colorado, 1989.

17. Kuethe, A. M. and Chow, C. Y., Foundations of Aerodynamics: Bases of Aerodynamic Design, 4th edition, John Wiley & Sons Inc., New York, 1986.

APPENDIX A

PROGRAM LISTING

```
      PROGRAM BLADE
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER(IX = 120 , IY = 10)
C
      DIMENSION NDF(IY),NNN(IY),NMID(IY)
      COMMON /C1/TH1(IY),OSX(IY),OSY(IY),CL(IY)/C2/X(IX,IY),Y(IX,IY)
      COMMON /C3/X1(IX,IY),Y1(IX,IY)/C4/ZI(IX),ETA(IY)
      COMMON /C5/XMIN(IY),XMAX(IY)/C6/DZI(IX),DETA(IY)
      COMMON /C7/P(IX,IY),Q(IX,IY),R(IX,IY)/C8/IC,ZIT
      COMMON /C9/AL(IX),BE(IY)/C10/V(IX),W(IY)
      COMMON /C11/ALPHA,BETA/C12/Z(IY)/C13/U(IX,IY)
      COMMON /C14A/XU(IX,IY),XL(IX,IY)
      COMMON /C14B/YU(IX,IY),YL(IX,IY)
      COMMON /C15/S(IY),T(IX)/C16/XIT,YIT
      COMMON /C18A/XU1(IX,IY),YU1(IX,IY)
      COMMON /C18B/XL1(IX,IY),YL1(IX,IY)
      COMMON /C20/U1(IX,IY),U2(IX,IY)
      COMMON /C30/NOW,ROR(IX),COR(IX),TWI(IX),THK(IX)
      COMMON /C31/SIGMACH,SIGMATW,SIGMATH
C
 1    CALL INPUT(NAF,NDF,NNN,NMID,SPAN,IDC,N1,SIGMA)
      CALL SCOSTW(NAF,NNN,NMID,SPAN)
      CALL TRANS(N1,NNN,NMID,NAF,SIGMA)
C
      IAN = 0
C
 5    CALL CHOICE(NAF)
      CALL SECT(N1,NAF,ICT,SPAN)
      CALL PLOTPR(ICT,IAN)
      CALL SYSTEM('CLS')
      WRITE(6,100)
      READ(5,*)IAN
C
      IF (IAN .EQ. 1) GOTO 5
C
      CALL SYSTEM('CLS')
      WRITE(6,200)
      READ(5,*)IAN
C
      IF (IAN .EQ. 1) GOTO 1
C
      STOP
 100  FORMAT(1X,///,
     1      " Do you want to interpolate the 'SAME' input airfoils"
     2   ,/,' again ? ENTER --->  0     No '
     3   ,/,'                     1     Yes  ---> ')
```

```
 200  FORMAT(1X,///,
     1                   Do you want to interpolate a',/,
     2     "                     'NEW SET' ",/,
     3                        of input airfoils?',//,
     4     ' ENTER --->  0    No ',/,
     5                   1    Yes  ---> ')
      END
```

```
C
C-----------------------------------------------------------------------
C
      SUBROUTINE CHOICE(NAF)
C
C*******************************************************************
C The purpose of this subroutine is to allow the user to choose   *
C plots of desired sections.                                       *
C*******************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER(IX = 120 , IY = 10)
C
      COMMON /C8/IC,ZIT/C12/Z(IY)/C3/X(IX,IY),Y(IX,IY)
C
      CALL SYSTEM('CLS')
      WRITE(6,90)
      READ(5,*)IC
C
      IF (IC .EQ. 1) THEN
 1      WRITE(6,100)Z(1),Z(NAF)
        READ(5,*)ZIT
C
        IF (ZIT .LT. Z(1) .OR. ZIT .GT. Z(NAF)) THEN
          WRITE(6,200)Z(1),Z(NAF)
          GOTO 1
        ENDIF
C
      ENDIF
C
      RETURN
 90   FORMAT(1X,
     1    ' YOU ARE NOW GIVEN THE FOLLOWING OPTIONS :'
     2 ,/,'   1.....GENERATE AN AIRFOIL AT ANY SPAN STATION.'
     3 ,/,'   2.....GENERATE A THREE-DIMENSIONAL BLADE.'
     4,//,' NOTE : '
     5 ,/,' Option 1 will furnish  the  user  with  an  airfoil'
     6 ,/,'          profile at the span  station  desired,  as'
     7 ,/,'          well as an airfoil normalized w.r.t. chord'
     8,//,' Option 2 will  interpolate  over  the  whole  blade'
     9 ,/,'          and   generate  the  coordinates  for  any'
     1 ,/,'          specified number of span station airfoils.'
     2,//,'      Output plots you can create using option 2 are
     3,//,'       a)   Concentric  plots of the generated  airfoils.'
     4 ,/,'       b)   Perspective  view  of  the  generated  blade.'
     5 ,/,'       c)   Top view of the generated blade. '
     6 ,/,'       d)   Front view of the generated blade.'
     7 ,/,'       e)   Chord distribution along the span of the blade.'
     8 ,/,'       f)   Twist distribution along the span of the blade.'
     9 ,/,'       g)   Maximum thickness distribution along the span'
     1                                                 ' of blade.'
     2,//,'  Enter your choice number here ---> ')
```

```
100  FORMAT(' The Z Coordnate must be between ',F7.2,' and ',F7.2,/,
    1       ' Enter the Z coordinate of the desired span station '
    2       ' here ---> ')
200  FORMAT(' The Z Coordnate must be between ',F7.2,' and ',F7.2,/,
    1       ' You just entered Z = ',F7.2,' Please try AGAIN ! ')
     END
```

```
C
C-----------------------------------------------------------------------
C
      SUBROUTINE CURVE1(N,X,Y,SLP1,SLPN,YP,TEMP,SIGMA)
C
C*********************************************************************
C THIS SUBROUTINE DETERMINES THE PARAMETERS NECESSARY TO COMPUTE     *
C AN INTERPOLATORY SPLINE UNDER TENSION THROUGH A SEQUENCE OF        *
C FUNCTIONAL VALUES.  THE SLOPES AT THE TWO ENDS OF THE CURVE MAY    *
C BE SPECIFIED OR OMITTED.  FOR ACTUAL COMPUTATION OF POINTS ON THE *
C CURVE IT IS NECESSARY TO CALL THE SUBROUTINE CURVE2.               *
C                                                                    *
C   N = THE NUMBER OF VALUES TO BE INTERPOLATED (N.GE.2).            *
C   X = AN ARRAY OF THE N INCREASING ABSCISSAE OF THE FUNCTIONAL     *
C       VALUES.                                                      *
C   Y = AN ARRAY OF THE N ORDINATES OF THE VALUES (I.E. Y(K) IS      *
C       THE FUNCTIONAL VALUE CORRESPONDING TO X(K)).                 *
C   SLP1 AND SLPN = CONTAIN THE DESIRED VALUES FOR THE FIRST         *
C                   DERIVATIVE OF THE CURVE AT X(1) AND X(N),        *
C                   RESPECTIVELY.  IF THE QUANTITY SIGMA IS          *
C                   NEGATIVE THESE VALUES WILL BE DETERMINED         *
C                   INTERNALLY AND THE USER NEED ONLY FURNISH        *
C                   PLACE-HOLDING PARAMETERS FOR SLP1 AND SLPN.      *
C                   SUCH PLACE-HOLDING PARAMETERS WILL BE IGNORED    *
C                   BUT NOT DESTROYED                                *
C   YP = AN ARRAY OF LENGTH AT LEAST N.                              *
C   TEMP = AN ARRAY OF LENGTH AT LEAST N WHICH IS USED FOR           *
C          SCRATCH STORAGE, AND                                      *
C   SIGMA = THE TENSION FACTOR.  THIS IS NON-ZERO AND INDICTES THE  *
C           CURVINESS DESIRED.  IF ABS(SIGMA) IS NEARLY ZERO (E.G.  *
C           0.01) THE RESULTING CURVE IS APPROXIMATELY A CUBIC       *
C           SPLINE.  IF ABS(SIGMA) IS LARGE (E.G. 50.) THE           *
C           RESULTING CURVE IS NEARLY A POLYGONAL LINE.  THE SIGN    *
C           OF SIGMA INDICATES WHETHER THE DERIVATIVE INFORMATION    *
C           HAS BEEN INPUT OR NOT.  IF SIGMA IS NEGATIVE THE END-    *
C           POINT DERIVATIVES WILL BE DETERMINED INTERNALLY.  A      *
C           STANDARD VALUES FOR SIGMA IS APPROXIMATELY 1, IN         *
C           ABSOLUTE VALUE.                                          *
C                                                                    *
C  ON OUTPUT ----                                                    *
C                                                                    *
C   YP = VALUES PROPORTIONAL TO THE 2ND DERIVATIVE OF THE CURVE      *
C        AT THE GIVEN NODES.                                         *
C   N, X, Y, SLP1, SLPN, AND SIGMA ARE UNALTERED.                    *
C*********************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER(IX = 120 , IY = 10)
C
      DIMENSION X(N),Y(N),YP(IX),TEMP(IX)
C
      NM1 = N - 1
```

```
      NP1 = N + 1
      DELX1 = X(2) - X(1)
      DX1 = (Y(2) - Y(1)) / DELX1
C
C************************************
C DETERMINE SLOPES IF NECESSARY     *
C************************************
C
      IF (SIGMA .LT. 0.D0) GOTO 50
C
      SLPP1 = SLP1
      SLPPN = SLPN
C
C*********************************
C DENORMALIZE TENSION FACTOR     *
C*********************************
C
 10   SIGMAP = DABS(SIGMA) * FLOAT(N-1) / (X(N)-X(1))
C
C*****************************************************************
C SET UP RIGHT HAND SIDE AND TRIGIAGONAL SYSTEM FOR YP AND       *
C PERFORM FORWARD ELIMINATION                                    *
C*****************************************************************
C
      DELS = SIGMAP * DELX1
      EXPS = DEXP(DELS)
      SINHS = 0.5D0 * (EXPS - 1.D0/EXPS)
      SINHIN = 1.D0 / (DELX1*SINHS)
      DIAG1 = SINHIN * (DELS * 0.5D0 * (EXPS+1/EXPS) - SINHS)
      DIAGIN = 1.D0/DIAG1
      YP(1) =  DIAGIN * (DX1-SLPP1)
      SPDIAG = SINHIN * (SINHS-DELS)
      TEMP(1) = DIAGIN * SPDIAG
      IF (N .EQ. 2) GOTO 30
C
      DO 20 I = 2,NM1
        DELX2  = X(I+1) - X(I)
        DX2    = (Y(I+1)-Y(I)) / DELX2
        DELS   = SIGMAP * DELX2
        EXPS   = DEXP(DELS)
        SINHS  = 0.5D0 * (EXPS-1.D0/EXPS)
        SINHIN = 1.D0 / (DELX2*SINHS)
        DIAG2  = SINHIN * (DELS * (0.5D0 * (EXPS+1.D0/EXPS))-SINHS)
        DIAGIN = 1.D0 / (DIAG1+DIAG2-SPDIAG*TEMP(I-1))
        YP(I)  = DIAGIN * (DX2-DX1-SPDIAG*YP(I-1))
        SPDIAG = SINHIN * (SINHS-DELS)
        TEMP(I) = DIAGIN * SPDIAG
        DX1 = DX2
        DIAG1 = DIAG2
 20   CONTINUE
C
 30   DIAGIN = 1.D0 / (DIAG1-SPDIAG*TEMP(NM1))
```

```
C
      YP(N) = DIAGIN * (SLPPN-DX2-SPDIAG*YP(NM1))
C
C*******************************
C PERFORM BACK SUBSTITUTION    *
C*******************************
C
      DO 40 I = 2,N
        IBAK = NP1-I
        YP(IBAK) = YP(IBAK) - TEMP(IBAK) * YP(IBAK+1)
 40   CONTINUE
C
      RETURN
 50   IF (N .EQ. 2) GOTO 60
C
C**************************************************************
C IF NO DERIVATIVES ARE GIVEN USE SECOND ORDER POLYNOMIAL     *
C INTERPOLATION ON INPUT DATA FOR VALUES AT ENDPOINTS.        *
C**************************************************************
C
      DELX2  = X(3)-X(2)
      DELX12 = X(3)-X(1)
      C1 = -(DELX12+DELX1) / DELX12 / DELX1
      C2 =   DELX12        / DELX1  / DELX2
      C3 = - DELX1          /DELX12 /DELX2
      SLPP1 = C1 * Y(1) + C2 * Y(2) + C3 * Y(3)
      DELN   = X(N)   - X(NM1)
      DELNM1 = X(NM1) - X(N-2)
      DELNN  = X(N)   - X(N-2)
      C1 = (DELNN+DELN) / DELNN / DELN
      C2 = -DELNN       / DELN  / DELNM1
      C3 =  DELN        / DELNN / DELNM1
      SLPPN = C3 * Y(N-2) + C2 * Y(NM1) + C1 * Y(N)
      GOTO 10
C
C*********************************************************************
C IF ONLY TWO POINTS AND NO DERIVATIVES ARE GIVEN, USE STRAIGHT      *
C LINE FOR CURVE                                                     *
C*********************************************************************
C
 60   YP(1) = 0.D0
      YP(2) = 0.D0
C
      RETURN
      END
```

```
C
C-----------------------------------------------------------------------
C
      FUNCTION CURVE2(T,N,X,Y,YP,SIGMA,IT)
C
C**********************************************************************
C THIS SUBROUTINE INTERPOLATES A CURVE AT A GIVEN POINT USING A       *
C SPLINE UNDER TENSION.  THE SUBROUTINE CURVE1 SHOULD BE CALLED       *
C EARLIER TO DETERMINE CERTAIN NECESSARY PARAMETERS.                  *
C                                                                     *
C  ON INPUT ----                                                      *
C                                                                     *
C   T = A REAL VALUE TO BE MAPPED ONTO THE INTERPOLATING CURVE        *
C   N = THE NUMBER OF POINTS WHICH WERE INTERPOLATED TO DETERMINE     *
C       THE CURVE                                                     *
C   X AND Y = ARRAYS CONTAINING THE ORDINATES AND ABCISSAS OF THE     *
C             POINTS                                                  *
C   YP = AN ARRAY WITH VALUES PROPORTIONAL TO THE SECOND              *
C        DERIVATIVE OF THE CURVE AT THE NODES                         *
C   SIGMA = THE TENSION FACTOR (ITS SIGN IS IGNORED).                 *
C   IT = IS AN INTEGER SWITCH.  IF IT IS NOT 1 THIS INDICATES THAT    *
C        THE SUBROUTINE HAS BEEN CALLED PREVIOUSLY (WITH N, X, Y,     *
C        YP, AND SIGMA UNALTERED) AND THAT THIS VALUE OF T EXCEEDS    *
C        THE PREVIOUS VALUE.  WITH SUCH INFORMATION THE SUBROUTINE    *
C        IS ABLE TO PERFORM THE INTERPOLATION MUCH MORE RAPIDLY.      *
C        IF A USER SEEKS TO INTERPOLATE AT A SEQUENCE OF POINTS,      *
C        EFFICIENCY IS GAINED BY ORDERING THE VALUES INCREASING       *
C        AND SETTING IT TO THE INDEX OF THE CALL.  IF IT IS 1 THE     *
C        SEARCH FOR THE INTERVAL (X(K),X(K+1)) CONTAINING T STARTS    *
C        WITH K=1.                                                    *
C                                                                     *
C  THE PARAMETERS N, X, Y, YP, AND SIGMA SHOULD BE INPUT UNALTERED    *
C  FROM THE OUTPUT OF CURVE1.                                         *
C                                                                     *
C  ON OUTPUT ----                                                     *
C                                                                     *
C  YT = THE INTERPOLATED VALUE.  FOR T LESS THAN, X(1) YT=Y(1).       *
C       FOR T GREATER THAN X(N), YT=Y(N).                             *
C  YTDX = DERIVATIVE OF YT WITH RESPECT TO X.                         *
C                                                                     *
C  NONE OF THE INPUT PARAMETERS ARE ALTERED.                          *
C**********************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER(IX = 120 , IY = 10)
C
      DIMENSION X(N),Y(N),YP(IX)
C
      S = X(N)-X(1)
C
C***********************
C DENORMALIZE SIGMA
C***********************
```

```
C
      SIGMAP = DABS(SIGMA) * FLOAT(N-1) / S
C
C*****************************************************************
C IF IT.NE.1 START SEARCH WHERE PREVIOUSLY TERMINATED, OTHERWISE  *
C START FROM BEGINNING                                            *
C*****************************************************************
C
      IF (IT .EQ. 1) I1 = 2
C
C************************
C SERACH FOR INTERVAL   *
C************************
C
 10   DO 20 I = I1,N
        IF (X(I)-T)20,20,30
 20   CONTINUE
C
      I = N
C
C***************************************
C CHECK TO INSURE CORRECT INTERVAL    *
C***************************************
C
 30   IF (X(I-1) .LE. T .OR. T .LE. X(I)) GOTO 40
C
C*********************************
C RESTART SERACH AND RESET I1    *
C (INPUT * IT * WAS INCORRECT)   *
C*********************************
C
      I1 = 2
      GOTO 10
C
C***************************************
C SET UP AND PERFORM INTERPOLATION    *
C***************************************
C
 40   DEL1 = T    - X(I-1)
      DEL2 = X(I) - T
      DEL3 = X(I) - X(I-1)
      EXPS1 = DEXP(SIGMAP*DEL1)
      SINHD1 = 0.5D0 * (EXPS1-1.D0/EXPS1)
      EXPS = DEXP(SIGMAP*DEL2)
      SINHD2 = 0.5D0 * (EXPS-1.D0/EXPS)
      EXPS = EXPS1 * EXPS
      SINHS = 0.5D0 * (EXPS-1.D0/EXPS)
      A1 = (YP(I) * SINHD1 + YP(I-1) * SINHD2) / SINHS
      A2 = ((Y(I)-YP(I)) * DEL1+(Y(I-1)-YP(I-1)) * DEL2) / DEL3
      CURVE2 = A1 + A2
      I1 = 1
C
      RETURN
```

```
      367          END
```

```
C
C-----------------------------------------------------------------------
C
      SUBROUTINE CHZE(I,J)
C
C*********************************************************************
C This subroutine determines the two 4x4 coefficient matrices, i.e. *
C CHZY and CHZX in equation (2.20) of manual.   These are then      *
C input in SUBROUTINE COEFM.                                        *
C*********************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER(IX = 120 , IY = 10)
C
      COMMON /CHZXY/CHZX(4,4),CHZY(4,4)
      COMMON /C10/V(IX),W(IY)/C6/DX(IX),DY(IY)
C
      DIMENSION CHZ(4,4,2)
      DIMENSION Z(2),H(2)
C
      H(1) - DX(I)
      Z(1) = V(I)
      H(2) = DY(J)
      Z(2) = W(J)
C
      DO 100 K = 1,2
        WON = 1.D0 / H(K)
        WOZ = 1.D0 / (1.D0-Z(K))
        WOP = 1.D0 / (1.D0-Z(K)*Z(K))
        CHZ(1,1,K) = 0.D0
        CHZ(1,2,K) = 0.D0
        CHZ(1,3,K) = WON
        CHZ(1,4,K) = 0.D0
        CHZ(2,1,K) = WON
        CHZ(2,2,K) = 0.D0
        CHZ(2,3,K) = 0.D0
        CHZ(2,4,K) = 0.D0
        CHZ(3,1,K) = -1.D0 * WON * WOZ
        CHZ(3,2,K) = -1.D0 * WOP
        CHZ(3,3,K) = WON * WOZ
        CHZ(3,4,K) = -1.D0 * Z(K) * WOP
        CHZ(4,1,K) = WON   * WOZ
        CHZ(4,2,K) = Z(K)  * WOP
        CHZ(4,3,K) = -1.D0 * WON * WOZ
        CHZ(4,4,K) = WOP
  100 CONTINUE
C
      DO 200 K = 1,4
C
        DO 300 L = 1,4
          CHZX(K,L) = CHZ(K,L,1)
          CHZY(K,L) = CHZ(K,L,2)
```

```
  300   CONTINUE
C
 200  CONTINUE
C
      RETURN
      END
```

```
C
C----------------------------------------------------------------------------
C
      SUBROUTINE CKFILE(FNAME)
C
C******************************************************
C  This subroutine checks/deletes an existing file.  *
C******************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      CHARACTER FNAME*(*),DELCMD*44
      LOGICAL*1 FEXIST
C
      INQUIRE (FILE = FNAME , EXIST = FEXIST)
C
      IF (FEXIST) THEN
        WRITE (DELCMD,10) FNAME
        CALL SYSTEM (DELCMD)
      ENDIF
C
      RETURN
 10   FORMAT('DEL ',A40)
      END
```

```
C
C-----------------------------------------------------------------------
C
      SUBROUTINE COEFM
C
C*********************************************************************
C This subroutine determines the 4x4 coefficient matrix, i.e.       *
C equation (2.19) of manual, for each sector of the computational   *
C domain.                                                           *
C*********************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
      COMMON /AIJKIJ/AK(4,4)/CHZXY/CHZX(4,4),CHZY(4,4)
C
      DIMENSION CHZYT(4,4)
C
      CALL TRANSPO(4,4,CHZY,CHZYT)
      CALL MATM(4,4,4,AK,CHZYT,CHZY)
      CALL MATM(4,4,4,CHZX,CHZY,AK)
C
      RETURN
      END
```

```
C
C-----------------------------------------------------------------------
C
      SUBROUTINE DINPUT(NAF,NDF,NNN,NMID,SPAN,IDC,N1,SIGMA,IU)
C
C*****************************************************************
C This subroutine performs the "DATA FILE" input mode option    *
C*****************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER(IX = 120   IY = 10)
C
      CHARACTER*1 SCALE
      CHARACTER*12 NAME(IY),FNAME,DNAME
C
      COMMON /C1/TH1(IY),OSX(IY),OSY(IY),CL(IY)/C2/XX(IX,IY),YY(IX,IY)
      COMMON /C12/ZZ(IY)/C11/ALPHA,BETA
      COMMON /C14A/XU(IX,IY),XL(IX,IY)
      COMMON /C14B/YU(IX,IY),YL(IX,IY)
      COMMON /C30/NOW,ROR(IX),COR(IX),TWI(IX),THK(IX)
      COMMON /C31/SIGMACH,SIGMATW,SIGMATH
C
      DIMENSION NDF(IY),NNN(IY)
      DIMENSION ZE(IX),CHORD(IX),VAR(8),THKN(IX),NMID(IY)
C
      DATA VAR/ 300.0D0,0.0D0,5.0D0,0.0D0,1.0D0,1.0D0,1.0D0,1.0D0/
C
      CALL SYSTEM('CLS')
      WRITE(6,121)
C
      READ(IU,*)IDC
C
      IF (IDC .EQ. 2) DNAME = 'CTE1.DAT'
C
      IF (IDC .EQ .3) DNAME = 'CTE2.DAT'
C
      IF (IDC .EQ. 4) THEN
        READ(IU,*)DNAME
      ENDIF
C
      IF (IDC .GT. 1) THEN
        OPEN(UNIT = 10 , FILE = DNAME , STATUS = 'OLD')
      ENDIF
C
      CALL CKFILE('CTEX1.DAT')
      OPEN(UNIT = 20 , FILE = 'CTEX1.DAT', STATUS = 'NEW')
C
C  Step 2
C
      READ(IU,*)NAF
C
      READ(IU,*)SPAN
```

```
C
      READ(IU,*)SCALE
C
      IF (IDC .GT. 1) THEN
        READ(10,*)NOW
C
        DO 8 I = 1,NOW
          READ(10,*)ROR(I),COR(I),TWI(I),THK(I)
          WRITE(20,205)ROR(I),COR(I),TWI(I),THK(I) * 100.0D0
          ZE(I)    = ROR(I) * SPAN
          CHORD(I) = COR(I) * SPAN
 8      CONTINUE
C
        CLOSE(UNIT = 10)
      ENDIF
C
      IF (IDC.EQ.1) NOW = NAF
      MAX = 0
      OPEN(UNIT = 8 , FILE = 'AI0A.DAT' , STATUS = 'UNKNOWN')
C
C  Step 3
C
      DO 100 K = 1,NAF
C
        READ(IU,*)NDF(K)
C
        IF (NDF(K) .EQ. 1) NAME(K) = 'S806A.DAT'
        IF (NDF(K) .EQ. 2) NAME(K) = 'S805A6.DAT'
        IF (NDF(K) .EQ. 3) NAME(K) = 'S805A.DAT'
        IF (NDF(K) .EQ. 4) NAME(K) = 'S805A7.DAT'
        IF (NDF(K) .EQ. 5) NAME(K) = 'S807.DAT'
        IF (NDF(K) .EQ. 6) NAME(K) = 'S808.DAT'
        IF (NDF(K) .EQ. 7) THEN
          READ(IU,*)NAME(K)
        ENDIF
C
        FNAME = NAME(K)
        LUN = 10
        OPEN(UNIT = LUN,FILE = FNAME , STATUS = 'OLD' )
C
        READ(LUN,*)NU
        READ(LUN,*)NL
        N = NU + NL - 1
        NNN(K) = N
C
        DO 50 I = 1,NU
          READ(LUN,*)XU(I,K),YU(I,K)
 50     CONTINUE
C
        DO 60 I = 1,NL
        READ(LUN,*)XL(I,K),YL(I,K)
 60     CONTINUE
```

```
C
         CLOSE(UNIT = LUN)
C
         READ(IU,*)ZZ(K)
C
         IF (IDC .GT. 1) THEN
           ZIT = ZZ(K)
           SIGMACH = -1.D0
           SIGMATW = -1.D0
           SIGMATH = -1.D0
           CALL SPT11D(NOW   ZE   CHORD   SIGMACH   ZIT   CHL  )
           CALL SPT11D(NOW , ZE , TWI   , SIGMATW , ZIT , TWA  )
           CALL SPT11D(NOW , ZE , THK   , SIGMATH , ZIT , THICK)
           CL(K)   = CHL
           TH1(K)  = TWA
           THKN(K) = THICK
         ELSE
C
           READ(IU,*)TH1(K)
C
           READ(IU,*)CL(K)
C
           READ(IU,*)THKN(K)
C
           THICK = THKN(K)
C
           ROR(K) = ZZ(K)/SPAN
           COR(K) = CL(K)/SPAN
           TWI(K) = TH1(K)
           THK(K) = THKN(K)
C
           WRITE(20,205)ROR(K) , COR(K) , TH1(K) , THKN(K)*100.0D0
C
         ENDIF
C
         READ(IU,*)TWCTR
C
         IF (TWCTR .EQ. 1) THEN
           READ(IU,*)OSX(K)
           READ(IU,*)OSY(k)
         ENDIF
C
         IF (TWCTR .EQ. 2) THEN
           TWCTRX = 0.3D0
           SIG    = -1.D0
           CALL SPT11D(NU,XU,YU,SIG,TWCTRX,TWCTYU)
           CALL SPT11D(NL,XL,YL,SIG,TWCTRX,TWCTYL)
           OSX(K) = TWCTRX
           OSY(K) = 0.5D0 * (TWCTYU-TWCTYL)
         ENDIF
C
C SET UP FOR SCALING
C
```

```
            IF (SCALE .EQ. 'Y' .OR. SCALE .EQ. 'y') THEN
              CALL CKFILE('AFXYIN')
              OPEN (15 , FILE = 'AFXYIN' , STATUS = 'NEW')
              WU  = 1.0D0
              NU1 = NU-1
              VAR(1) = 0.D0
              WRITE(15,400)CHARNB(NAME(K))
              WRITE(15,401)(VAR(I),I = 1,8)
              WRITE(15,402)NU
              YU(NU,K) = (YU(NU,K) + YU(NU1,K)) / 2.0D0
C
              DO 66 I = 1,NU
                WRITE(15,403)XU(I,K),YU(I,K),WU
 66           CONTINUE
C
              WRITE(15,402)NL
C
              DO 67 I = 1,NL
                WRITE(15,403)XL(I,K),YL(I,K),WU
 67           CONTINUE
C
              CLOSE(UNIT = 15)
C
              CALL CKFILE('SCXYIN')
C
              CALL SYSTEM('SMOOTH2')
C
              OPEN (UNIT = 16 , FILE = 'SCALEINP.DAT' , ACCESS = 'APPEND'
     /          , STATUS = 'OLD')
C
              XNT = 1.0D0
              WRITE(16,402)XNT
              WRITE(16,402)THICK
              CLOSE(UNIT = 16)
C
              CALL SYSTEM('RENAME SCALEINP.DAT SCXYIN')
C
              CALL SYSTEM('SCALE1')
C
C CONCLUSION OF SCALING
C
C
C     RETRIEVE  SCALED DATA FROM DATA FILES
C
              OPEN (UNIT = 17 , FILE = 'XUXLSS.DAT' , STATUS = 'OLD')
              READ(17,*)NU
              READ(17,*)(XU(I,K),YU(I,K),I = 1,NU)
              READ(17,*)NL
              READ(17,*)(XL(I,K),YL(I,K),I = 1,NL)
              CLOSE(UNIT = 17)
C
            ENDIF
```

```
C
C  CONTINUE AND FORM THE AIRFOIL
C
          N = NU + NL - 1
          NNN(K)   = N
          NMID(K)  = NL
          YU(NU,K) = 0.0D0
          YL(NL,K) = 0.0D0
C
          DO 70 I = 1,N
            J = NL - I  + 1
            L = I  - NL + 1
C
            IF (I .LT. NL) THEN
C
              XX(I,K) = XL(J,K)
              YY(I,K) = YL(J,K)
C
            ELSE
C
              XX(I,K) = XU(L,K)
              YY(I,K) = YU(L,K)
C
            ENDIF
C
            WRITE(8,*)XX(I,K),YY(I,K),ZZ(K)/SPAN
C
   70     CONTINUE
C
          IF (NNN(K) .GT. MAX) MAX = NNN(K)
C
  100   CONTINUE
C
        CLOSE (UNIT = 8)
C
C  Step 4
C
        SIGMA = -1.D0
        READ(IU,*)N1
C
        READ(IU,*)ALPHA
C
        READ(IU,*)BETA
C
        CLOSE(UNIT = IU)
C
        OPEN(UNIT = 100 , FILE = 'INPUT.DAT' , ACCESS = 'APPEND'
       /    , FORM  = 'FORMATTED',CARRIAGE CONTROL = 'FORTRAN'
       /    , STATUS = 'UNKNOWN')
C
        WRITE(100,105)NAF,ALPHA,BETA
C
        DO 200 K = 1,NAF
```

```
C
             WRITE(100,106)K,NAME(K)
             WRITE(100,203)
             WRITE(100,220)CL(K),TH1(K),OSX(K),OSY(K),ZZ(K),THKN(K)
             WRITE(100,107)
             IA = (NNN(K) + 1) / 2
C
             IF (NMID(K) .GE. IA) THEN
               NREF = NMID(K)
             ELSE
               NREF = NNN(K) - NMID(K) + 1
             ENDIF
C
             DO 120 I = 1,NREF
               WRITE(100,210)XU(I,K),YU(I,K),XL(I,K),YL(I,K)
C
  120        CONTINUE
C
  200      CONTINUE
C
           N = MAX
C
           CLOSE(UNIT = 20)
           CLOSE(UNIT = 100)
C
           RETURN
  105      FORMAT(/,10X,' Number of input airfoils = ',I2,//,10X,
     /           ' Tension parameters: Alpha = ',F7.2,', and Beta = ',F7.2)
  106      FORMAT(1H1,/,10X,' Input data file name for airfoil',I2,' is '
     /           ,A12,' and the input data are:',//)
  107      FORMAT(//,17X,'XU',12X,'YU',13X,'XL',11X,'YL',/)
  121      FORMAT(1X,/,' Please wait, program is running!')
  201      FORMAT(1X,I2)
  202      FORMAT(1X,2F8.2)
  203      FORMAT(7X,'Chord-',7X,'Twist',10X,'Twist Center',9X,'Span',
     /       9X,'max',/,7X,'Length',7X,'Angle',10X,'X',10X,'Y',8X,'Coord.'
     /     ,5X,'thickness')
  205      FORMAT(1X,4(F9.5,1X))
  210      FORMAT(10X,4(2X,E12.5))
  220      FORMAT(2X,6(2X,F10.4))
  400      FORMAT(A)
  401      FORMAT(8F10.5)
  402      FORMAT(F10.5)
  403      FORMAT(3E15.6)
           END
```

```
C
C-----------------------------------------------------------------------
C
      SUBROUTINE GETX(N,M,TAU,SMID)
C
C**********************************************************************
C This subroutine determines the X-coordinate of all points on the    *
C surface of an airfoil at a particular span station of the blade.    *
C**********************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER(IX = 120 , IY = 10)
C
      COMMON /C4/ZI(IX),ETA(IY)/C6/DZI(IX),DETA(IY)
      COMMON /C7/P(IX,IY),Q(IX,IY),R(IX,IY)/C13/U(IX,IY)
      COMMON /C10/V(IX),W(IY)/C9/AL(IX),BE(IY)/C16/XIT,YIT
      COMMON /C20/U1(IX,IY),U2(IX,IY)
      COMMON /C22/XZ(IX),YZ(IX)
C
      DO 10 J = 1,M
C
        DO 20 I = 1,N
          U(I,J) = U1(I,J)
 20     CONTINUE
C
 10   CONTINUE
C
      CALL PDERIV(N,M)
C
      IA = (N + 1) / 2
      DZL = SMID / FLOAT(IA-1)
      DZU = (1.0D0 - SMID) / FLOAT(IA-1)
      B1 = 1.0D0 + (DEXP(TAU)  - 1.0D0) * SMID
      B2 = 1.0D0 + (DEXP(-TAU) - 1.0D0) * SMID
      B = DLOG(B1/B2) * 0.5D0 / TAU
C
      DO 100 I = 1,N
        IF (I .EQ. IA) THEN
          XIT = SMID
        ELSE
          IF (I .LT. IA) ZIT = DZL * FLOAT(I-1)
          IF (I .GT. IA) ZIT = DZU * FLOAT(I-IA) + SMID
          XIT = SMID * (1.0D0 + (DSINH(TAU*(ZIT-B)) / DSINH(TAU*B)))
        ENDIF
        CALL GETU(N,M,UIT)
        XZ(I) = UIT
 100  CONTINUE
C
      RETURN
      END
```

```
C
C------------------------------------------------------------------------
C
      SUBROUTINE GETY(N,M,TAU,SMID)
C
C**********************************************************************
C This subroutine determines the Y-coordinate of all points on the    *
C surface of an airfoil at a particular span station of the blade.    *
C**********************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER(IX = 120 , IY = 10)
C
      COMMON /C4/ZI(IX),ETA(IY)/C6/DZI(IX),DETA(IY)
      COMMON /C7/P(IX,IY),Q(IX,IY),R(IX,IY)/C13/U(IX,IY)
      COMMON /C10/V(IX),W(IY)/C9/AL(IX),BE(IY)/C16/XIT,YIT
      COMMON /C20/U1(IX,IY),U2(IX,IY)
      COMMON /C22/XZ(IX),YZ(IX)
C
      DO 10 J = 1,M
C
        DO 20 I = 1,N
          U(I,J) = U2(I,J)
 20     CONTINUE
C
 10   CONTINUE
C
      CALL PDERIV(N,M)
C
      IA = (N + 1) / 2
      DZL = SMID / FLOAT(IA-1)
      DZU = (1.0D0 - SMID) / FLOAT(IA-1)
      B1 = 1.0D0 + (DEXP(TAU)  - 1.0D0) * SMID
      B2 = 1.0D0 + (DEXP(-TAU) - 1.0D0) * SMID
      B = DLOG(B1/B2) * 0.5D0 / TAU
C
      DO 100 I = 1,N
        IF (I .EQ. IA) THEN
          XIT = SMID
        ELSE
          IF (I.LT.IA) ZIT = DZL * FLOAT(I-1)
          IF (I.GT.IA) ZIT = DZU * FLOAT(I-IA) + SMID
          XIT = SMID * (1.0D0 + (DSINH(TAU*(ZIT-B)) / DSINH(TAU*B)))
        ENDIF
        CALL GETU(N,M,UIT)
        YZ(I) = UIT
  100 CONTINUE
C
      RETURN
      END
```

```
C
C------------------------------------------------------------------------
C
      SUBROUTINE GETU(N,M,UIT)
C
C------------------------------------------------------------------------
C The purpose of this routine is to determine the ORDINATE value at
C the points xit and yit in the computational domain.The value of the
C ordinate is passed out as UIT.
C------------------------------------------------------------------------
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER(IX = 120 , IY = 10)
C
      COMMON /C4/ZI(IX),ETA(IY)/C6/DZI(IX),DETA(IY)
      COMMON /C7/P(IX,IY),Q(IX,IY),R(IX,IY)/C13/U(IX,IY)
      COMMON /C10/V(IX),W(IY)/C9/AL(IX),BE(IY)/C16/XIT,YIT
      COMMON /AIJKIJ/AK(4,4)/CHZXY/CHZX(4,4),CHZY(4,4)
      COMMON /PHIXY/PHIX(4),PHIY(4)
C
      CALL SORT(N,M,IL,JL)
      CALL KAY(IL,JL)
      CALL CHZE(IL,JL)
      CALL COEFM
      CALL PHI(IL,JL)
      SUM = 0.D0
C
      DO 100 K = 1,4
C
        DO 200 L = 1,4
          SUM = SUM + AK(K,L) * PHIX(K) * PHIY(L)
 200    CONTINUE
C
 100  CONTINUE
C
      UIT = SUM
C
      RETURN
      END
```

```
C
C------------------------------------------------------------------
C
      SUBROUTINE HRDOUT(SCREEN,PLOTER,PRINTR)
C
C************************************************************
C This subroutine controls graphic output device options.   *
C************************************************************
C
      CHARACTER*7 SCREEN, PLOTER, PRINTR
C
C IF THE USER WANTS TO VIEW OR PLOT OUTPUT DATA HE/SHE HAS THE OPTI
C
      PRINT,' WOULD YOU LIKE TO VIEW THIS ?'
      PRINT,' ENTER --->     0     NO. '
      PRINT,'                1     YES.  --->
C
      READ(5,*)IANS
C
      IF (IANS .EQ. 1) CALL SYSTEM(SCREEN)
C
10    PRINT,' DO YOU WANT TO SEND THIS VIEW TO A HARD-COPY DEVICE ?
      PRINT,' ENTER --->     0     NO.'
      PRINT,'                1     YES, SEND TO PLOTTER.'
      PRINT,'                2     YES, SEND TO PRINTER.'
      PRINT,' ENTER YOUR CHOICE NUMBER HERE ---> '
C
      READ(5,*)IANS
C
      IF (IANS .EQ. 1) THEN
        PRINT *
        PRINT,' PLOTTING............'
        CALL SYSTEM(PLOTER)
        GO TO 10
      ENDIF
C
      IF (IANS .EQ. 2) THEN
        PRINT *
        PRINT,' PRINTER FILE GENERATING.......PLEASE WAIT......'
        CALL SYSTEM(PRINTR)
        PRINT,' PRINTING.................'
        CALL SYSTEM('DPRINT DUMFIL')
        GO TO 10
      ENDIF
C
      RETURN
      END
```

```
C
C-----------------------------------------------------------------------
C
      SUBROUTINE INPUT(NAF,NDF,NNN,NMID,SPAN,IDC,N1,SIGMA)
C
C*******************************************************************
C This subroutine performs all major input necessary to run the    *
C program.                                                          *
C*******************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER(IX = 120 , IY = 10)
C
      CHARACTER*1  CHOICE
      CHARACTER*12 INAME
C
      COMMON /C1/TH1(IY),OSX(IY),OSY(IY),CL(IY)/C2/XX(IX,IY),YY(IX,IY)
      COMMON /C12/ZZ(IY)/C11/ALPHA,BETA
      COMMON /C14A/XU(IX,IY),XL(IX,IY)
      COMMON /C14B/YU(IX,IY),YL(IX,IY)
      COMMON /C30/NOW,ROR(IX),COR(IX),TWI(IX),THK(IX)
      COMMON /C31/SIGMACH,SIGMATW,SIGMATH
C
      CALL CKFILE('INPUT.DAT')
      CALL CKFILE('WTOUT.DAT')
      CALL CKFILE('NTOUT.DAT')
      CALL SYSTEM('CLS')
C
      WRITE(6,90)
      PAUSE
C
      CALL SYSTEM('CLS')
C
      WRITE(6,91)
      READ(5,*)CHOICE
      CALL SYSTEM('CLS')
C
      IF (CHOICE .EQ. 'D' .OR. CHOICE .EQ. 'd') THEN
        WRITE(6,92)
        READ(5,*)INAME
        IU = 50
        OPEN(UNIT = IU , FILE = INAME , STATUS = 'OLD')
        CALL DINPUT(NAF,NDF,NNN,NMID,SPAN,IDC,N1,SIGMA,IU)
      ELSE
        IU = 5
        CALL TINPUT(NAF,NDF,NNN,NMID,SPAN,IDC,N1,SIGMA,IU)
      ENDIF
C
      RETURN
C
 90   FORMAT(1X,//,
     1          This program uses  certain  cross-sectional  airfoil'
     2   ,/,' data  as  input,  and  proceeds  to interpolate  these'
```

```
     3   ,/,' input data by use of a bi-tension spline method.'
     4  ,//,'   As a result of this interpolation, the whole surface'
     5   ,/,' of the blade is definable.'
     6  ,//,' This  program  will  then  generate  airfoil  data for'
     7   ,/,' any particular span stations of the blade. '
     8 ,///,' ***  READ AND FOLLOW INSTRUCTIONS CAREFULLY ***')
91    FORMAT(1X
     1  ,//,' You have two alternative ways of entering data :'
     2  ,//,'     Enter T for the terminal input'
     3  ,//,'     Enter D for the data file input'
     4  ,//,' Enter your choice here ---> ')
92    FORMAT(1X,//,
     1           Please enter the name of the data input file ---> ')
      END
```

```
C
C-----------------------------------------------------------------------
C
      SUBROUTINE KAY(I,J)
C
C*********************************************************************
C This subroutine creates the 4x4 coefficient matrix, i.e.          *
C equation (2.21) ofthe manual, which contains all the values of    *
C U and its partial derivatives at the four nodes of each sector    *
C in the computational domain.                                      *
C*********************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER(IX = 120 , IY = 10)
C
      COMMON /C13/U(IX,IY)/C7/P(IX,IY),Q(IX,IY),R(IX,IY)
      COMMON /AIJKIJ/AK(4,4)
C
      JP1 = J + 1
      IP1 = I + 1
C
      AK(1,1) = U(I,J)
      AK(1,2) = Q(I,J)
      AK(1,3) = U(I,JP1)
      AK(1,4) = Q(I,JP1)
      AK(2,1) = P(I,J)
      AK(2,2) = R(I,J)
      AK(2,3) = P(I,JP1)
      AK(2,4) = R(I,JP1)
      AK(3,1) = U(IP1,J)
      AK(3,2) = Q(IP1,J)
      AK(3,3) = U(IP1,JP1)
      AK(3,4) = Q(IP1,JP1)
      AK(4,1) = P(IP1,J)
      AK(4,2) = R(IP1,J)
      AK(4,3) = P(IP1,JP1)
      AK(4,4) = R(IP1,JP1)
C
      RETURN
      END
```

```
C
C------------------------------------------------------------------------
C
      SUBROUTINE MATM(N,M,L,A,B,C)
C
C**************************************************************
C This subroutine calculates the product of two matrices,    *
C A(I,J)*B(J,K), and stores the resutls in matrix C(I,J).    *
C**************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
      DIMENSION A(N,M),B(M,L),C(N,L)
C
      DO 60 I = 1,N
C
        DO 50 K = 1,L
          SUM = 0.00
C
            DO 40 J = 1,M
              SUM = SUM + A(I,J) * B(J,K)
 40         CONTINUE
C
          C(I,K) = SUM
 50     CONTINUE
C
 60   CONTINUE
C
      RETURN
      END
```

```
C
C-----------------------------------------------------------------------
C
      SUBROUTINE PDERIV(N,M)
C
C***********************************************************************
C  This routine receives the transformed coordinates as input data.    *
C  It then proceeds to find the partial derivatives of Y and X with
C  respect to ZI and ETA.
C
C  NOTE: X, Y, and U represent ZI, ETA and X or Y respectively.        *
C                                                                      *
C        DX and DY (ie. dzi and deta) are computed first.              *
C***********************************************************************
C  First we will normalize the tension factors AL and BE.              *
C  Simultaneously we find DZI and DETA , represented by DX and DY.
C***********************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER(IX = 120 , IY = 10)
C
      COMMON /C4/X(IX),Y(IY)/C6/DX(IX),DY(IY)/C11/ALPHA,BETA
      COMMON /C7/P(IX,IY),Q(IX,IY),R(IX,IY)/C13/U(IX,IY)
      COMMON /C15/S(IY),T(IX)/C10/V(IX),W(IY)/C9/AL(IX),BE(IY)
      COMMON /TRIDM/CL(IX),CM(IX),CN(IX),PQ(IX)
C
      NM1 = N-1
      MM1 = M-1
C
      DO 10 I = 1,NM1
        IP1 = I + 1
        AL(I) = ALPHA * FLOAT(NM1)
        DX(I) = X(IP1) - X(I)
 10   CONTINUE
C
      DO 20 J = 1,MM1
        JP1 = J + 1
        BE(J) = BETA * FLOAT(MM1)
        DY(J) = Y(JP1) - Y(J)
 20   CONTINUE
C
C Next the coefficients V and T are computed
C
      DO 30 I = 1,NM1
        ADX = AL(I) * DX(I)
C
        IF (ADX .GT. 0.D0 .AND. ADX .LT. 0.1D0) THEN
          ADX2   = ADX * ADX
          XAN    = 1.D0 + ADX2 * (0.1D0  + ADX2 / 280.D0)
          XAD    = 1.D0 + ADX2 * (0.05D0 + ADX2 / 840.D0)
          V(I)   = 2.D0 * XAN / XAD
          SINHDX = DSINH(ADX)
```

```
            SAME   = SINHDX - ADX
          ELSE
            SINHDX = DSINH(ADX)
            SAME   = SINHDX - ADX
            V(I)   = (ADX * DCOSH(ADX) - SINHDX) / SAME
          ENDIF
C
          T(I) = ADX * AL(I) * SINHDX / ((V(I) * V(I) - 1.D0) * SAME)
C
 30     CONTINUE
C
C Here the coefficients W and S are computed
C
        DO 40 J = 1,MM1
          BDY = BE(J) * DY(J)
C
          IF (BDY .GT. 0.D0 .AND. BDY .LT .0.1D0) THEN
            BDY2   = BDY * BDY
            YAN    = 1.D0 + BDY2 * (0.1D0  + BDY2 / 280.D0)
            YAD    = 1.D0 + BDY2 * (0.05D0 + BDY2 / 840.D0)
            W(J)   = 2.D0 * YAN / YAD
            SINHDY = DSINH(BDY)
            SAME   = SINHDY - BDY
          ELSE
            SINHDY = DSINH(BDY)
            SAME   = SINHDY - BDY
            W(J)   = (BDY * DCOSH(BDY) - SINHDY) / SAME
          ENDIF
C
          S(J) = BDY * BE(J) * SINHDY / ((W(J) * W(J) - 1.D0) * SAME)
C
 40     CONTINUE
C
C*********************************************************************
C The following procedure is meant to determine the mesh point slopes *
C P,Q, AND R.                                                          *
C    First the boundary slopes are determined by:                      *
C        1....Forward  difference for the first point, 1.              *
C        2....Backward differemce for the last  point, N/M.            *
C    Next routines to set up the internal mesh point  equations  are  *
C  called (i.e. PUX,QUY).   These   routines   return   the partial  *
C  derivatives.  The whole job is done below for the three necessary  *
C  partial derivatives: P,Q AND R.                                     *
C*********************************************************************
C
        DO 50 J = 1,M
          P(1,J) = (U(2,J) - U(1,J))   / (X(2) - X(1))
          P(N,J) = (U(N,J) - U(NM1,J)) / (X(N) - X(NM1))
 50     CONTINUE
C
        CALL PUX(N,M)
C
        DO 60 I = 1,N
```

```
          Q(I,1) = (U(I,2) - U(I,1))   / (Y(2) - Y(1))
          Q(I,M) = (U(I,M) - U(I,MM1)) / (Y(M) - Y(MM1))
  60    CONTINUE
C
        IF (M .GT. 2) THEN
          CALL QUY(N,M)
        ENDIF
C
        DO 70 J = 1,M,MM1
          R(1,J) = (Q(2,J) - Q(1,J))   / (X(2) - X(1))
          R(N,J) = (Q(N,J) - Q(NM1,J)) / (X(N) - X(NM1))
  70    CONTINUE
C
        CALL RUXY1(N,M)
C
        IF (M .GT. 2) THEN
          CALL RUXY2(N,M)
        ENDIF
C
        RETURN
        END
```

```
C
C-----------------------------------------------------------------------
C
      SUBROUTINE PLOTPR(ICT,IAN)
C
C****************************************************************
C This subroutine plots and prints input and generated data    *
C****************************************************************
C
      CHARACTER*7 SCREEN(9), PLOTER(9), PRINTR(9)
C
      DATA SCREEN / 'VICPINP' , 'VICPGEN' , 'VICHNPL'   'VI3D
     1             'VITOP    , 'VIFRONT' , 'VICHORD'   'VITWIST'
     2            , 'VITHICK' /
C
      DATA PLOTER / 'PLCPINP'   'PLCPGEN'   'PLCHNPL' , 'PL3D
     1             'PLTOP     'PLFRONT'   'PLCHORD' , 'PLTWIST'
     2            , 'PLTHICK' /
C
      DATA PRINTR / 'PRCPINP' , 'PRCPGEN' , 'PRCHNPL' , 'PR3D
     1             'PRTOP     'PRFRONT'   'PRCHORD'   'PRTWIST'
     2            , 'PRTHICK' /
C
      IF (IAN .EQ. 0) THEN
C
        CALL SYSTEM('CLS')
        PRINT,' CONCENTRIC PLOT OF THE INPUT AIRFOILS.'
        CALL HRDOUT (SCREEN(1),PLOTER(1),PRINTR(1))
C
      ENDIF
C
C Plots for ICT = 1
C
      IF (ICT .EQ. 1) THEN
C
        CALL SYSTEM('CLS')
        PRINT,' PLOT OF GENERATED AIRFOIL AT CHOSEN SPAN STATION.'
        CALL HRDOUT (SCREEN(2),PLOTER(2),PRINTR(2))
C
        CALL SYSTEM('CLS')
        PRINT,' PLOT OF GENERATED AIRFOIL AT CHOSEN SPAN STATION.'
        PRINT,' that has had the twist and  offset  removed,  and'
        PRINT,' been normalized w.r.t. its chord.'
        CALL HRDOUT (SCREEN(3),PLOTER(3),PRINTR(3))
C
      ENDIF
C
C Plots for ICT = 2
C
      IF (ICT .EQ. 2) THEN
C
        CALL SYSTEM('CLS')
```

```
           PRINT,' A....Concentric plot of generated span station'
           PRINT,'      airfoils.'
           CALL HRDOUT (SCREEN(2),PLOTER(2),PRINTR(2))
C
           CALL SYSTEM('CLS')
           PRINT,' B....3-D Plot of the generated blade.'
           PRINT,'     ** RESTRICTION : Total generated blade stations '
           PRINT,'                           MUST less than 40'
           CALL HRDOUT (SCREEN(4),PLOTER(4),PRINTR(4))
C
           CALL SYSTEM('CLS')
           PRINT,' C....Normalized TOP view plot of the generated blade.'
           CALL HRDOUT (SCREEN(5),PLOTER(5),PRINTR(5))
C
           CALL SYSTEM('CLS')
           PRINT,' D....Normalized FRONT view plot of the generated blade.'
           CALL HRDOUT (SCREEN(6),PLOTER(6),PRINTR(6))
C
           CALL SYSTEM('CLS')
           PRINT,' E....The CHORD distribution of the generated blade.'
           CALL HRDOUT (SCREEN(7),PLOTER(7),PRINTR(7))
C
           CALL SYSTEM('CLS')
           PRINT,' F....The TWIST distribution of the generated blade.'
           CALL HRDOUT (SCREEN(8),PLOTER(8),PRINTR(8))
C
           CALL SYSTEM('CLS')
           PRINT,' G....The MAXIMUM THICKNESS distribution of the'
           PRINT,'                generated blade.'
           CALL HRDOUT (SCREEN(9),PLOTER(9),PRINTR(9))
C
         ENDIF
C
C Printed Output
C
         WRITE(6,100)
         READ(5,*)IA
C
         IF (IA .EQ. 1) THEN
C
 11        CALL SYSTEM('CLS')
           WRITE(6,101)
           READ(5,*)IWANT
           CALL SYSTEM('CLS')
           WRITE(6,102)
           PAUSE
           IF (IWANT .EQ. 1) CALL SYSTEM('COPY INPUT.DAT PRN')
           IF (IWANT .EQ. 2) CALL SYSTEM('COPY WTOUT.DAT PRN')
           IF (IWANT .EQ. 3) CALL SYSTEM('COPY NTOUT.DAT PRN')
           CALL SYSTEM('CLS')
           WRITE(6,103)
           READ(5,*)IAGAIN
```

```
          IF (IAGAIN .EQ. 1) GOTO 11
C
        ENDIF
C
        RETURN
  100   FORMAT(1X,/,
     1     ' Would you like a printout of the input and/or output data?'
     2 ,/,' If you want a printout, you need to  have  plenty of paper'
     3 ,/,' and time, because the printout is several pages long.'
     4,//,' ENTER  --->     0      NO.'
     5 ,/,'                 1      YES.'
     6,//,' ENTER YOUR CHOICE NUMBER HERE ---> ')
  101   FORMAT(1X,/,
     1        'Which data set you would like to have a printout?'
     2  ,//,' 1.  Input data.'
     3  ,//,' 2.  Output of the generated airfoil at the chosen span'
     4   ,/,'     stations with twist and taper.'
     5  ,//,' 3.  Output of the generated airfoil at the chosen span'
     6   ,/,'     stations without twist and taper.'
     7  ,//,' Enter your choice number here ---> ')
  102   FORMAT(' MAKE SURE THAT THE PRINTER IS READY.')
  103   FORMAT(1X,/,
     1        ' WOULD YOU LIKE TO HAVE ANOTHER SET OF PRINTOUT?'
     2   ,/,' ENTER  --->     0      NO.'
     3   ,/,'                 1      YES.',/)
        END
```

```
C
C-----------------------------------------------------------------------
C
      SUBROUTINE PHI(IL,JL)
C
C*************************************************************
C This subroutine determines the two interpolation function   *
C vectors, in the xi and eta directions, for each sector of   *
C the computational domain, see equations (2.3) and (2.4) of  *
C of the manual.                                              *
C*************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER(IX = 120 ,  IY = 10)
C
      COMMON /C4/X(IX),Y(IY)/C9/AL(IX),BE(IY)/C6/DX(IX),DY(IY)
      COMMON /C16/XIT,YIT
      COMMON /PHIXY/PHIX(4),PHIY(4)
C
      IU = IL + 1
      JU = JL + 1
      PHIX(1) = XIT   - X(IL)
      PHIY(1) = YIT   - Y(JL)
      PHIX(2) = X(IU) - XIT
      PHIY(2) = Y(JU) - YIT
      ADX = AL(IL) * DX(IL)
      BDY = BE(JL) * DY(JL)
      QUX = DSINH(ADX) - ADX
      QUY = DSINH(BDY) - BDY
      PHIX(3) = (DX(IL)*DSINH(AL(IL)*PHIX(1))-PHIX(1)*DSINH(ADX))/QUX
      PHIY(3) = (DY(JL)*DSINH(BE(JL)*PHIY(1))-PHIY(1)*DSINH(BDY))/QUY
      PHIX(4) = (DX(IL)*DSINH(AL(IL)*PHIX(2))-PHIX(2)*DSINH(ADX))/QUX
      PHIY(4) = (DY(JL)*DSINH(BE(JL)*PHIY(2))-PHIY(2)*DSINH(BDY))/QUY
C
      RETURN
      END
```

```
C
C---------------------------------------------------------------------------
C
      SUBROUTINE PUX(N,M)
C
C*****************************************************************
C This subroutine determines the partial derivative of U with  *
C respect to xi at every internal node of the computational    *
C domain.  These derivatives are stored in array P.            *
C*****************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER(IX = 120 , IY = 10)
C
      COMMON /C6/DX(IX),DY(IY)
      COMMON /C7/P(IX,IY),Q(IX,IY),R(IX,IY)/C13/U(IX,IY)
      COMMON /C15/S(IY),T(IX)/C10/V(IX),W(IY)
      COMMON /TRIDM/CL(IX),CM(IX),CN(IX),PQ(IX)
C
      NM1 = N-1
      NM2 = N-2
C
      DO 80 J = 1,M
C
        DO 60 I = 2,NM1
          IM1 = I - 1
          IP1 = I + 1
          CM(IM1) = T(IM1) * V(IM1) + T(I) * V(I)
          CL(IM1) = T(IM1)
          CN(IM1) = T(I)
          BA = T(IM1) * (V(IM1)+1.D0) * (U(I,J)-U(IM1,J)) / DX(IM1)
          BB = T(I)   * (V(I)  +1.D0) * (U(IP1,J)-U(I,J)) / DX(I)
          PQ(IM1) = BA + BB
 60     CONTINUE
C
        PQ(1)   = PQ(1)   - T(1)   * P(1,J)
        PQ(NM2) = PQ(NM2) - T(NM1) * P(N,J)
        CALL THOMAS(NM2)
C
        DO 70 I = 2,NM1
          IM1 = I - 1
          P(I,J) = PQ(IM1)
 70     CONTINUE
C
 80   CONTINUE
C
      RETURN
      END
```

```
C
C-------------------------------------------------------------------------
C
      SUBROUTINE QUY(N,M)
C
C****************************************************************
C This subroutine determines the partial derivative of U with  *
C respect to eta at every internal node of the computational   *
C domain.  These derivatives are stored in array Q.            *
C****************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER(IX = 120 , IY = 10)
C
      COMMON /C6/DX(IX),DY(IY)
      COMMON /C7/P(IX,IY),Q(IX,IY),R(IX,IY)/C13/U(IX,IY)
      COMMON /C15/S(IY),T(IX)/C10/V(IX),W(IY)
      COMMON /TRIDM/CL(IX),CM(IX),CN(IX),PQ(IX)
C
      MM1 = M-1
      MM2 = M-2
C
      DO 80 I = 1,N
C
        DO 60 J = 2,MM1
          JM1 = J - 1
          JP1 = J + 1
          CM(JM1) = S(JM1) * W(JM1) + S(J) * W(J)
          CL(JM1) = S(JM1)
          CN(JM1) = S(J)
          BA = S(JM1) * (W(JM1)+1.D0) * (U(I,J)-U(I,JM1)) / DY(JM1)
          BB = S(J)   * (W(J)+1.D0)   * (U(I,JP1)-U(I,J)) / DY(J)
          PQ(JM1) = BA + BB
 60     CONTINUE
C
        PQ(1)   = PQ(1)   - S(1)   * Q(I,1)
        PQ(MM2) = PQ(MM2) - S(MM1) * Q(I,M)
        CALL THOMAS(MM2)
C
        DO 70 J = 2,MM1
          JM1 = J - 1
          Q(I,J) = PQ(JM1)
 70     CONTINUE
C
 80   CONTINUE
C
      RETURN
      END
```

```
C
C-------------------------------------------------------------------
C
      SUBROUTINE RUXY1(N,M)
C
C*****************************************************************
C This subroutine determines the mixed-derivative of U with    *
C respect to xi and eta at each boundary grid node of the      *
C computational domain, where eta has a value of 0 or 1 and    *
C xi does not have values of 0 or 1.  These derivatives are    *
C stored in array R.                                            *
C*****************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER(IX = 120 , IY = 10)
C
      COMMON /C6/DX(IX),DY(IY)
      COMMON /C7/P(IX,IY),Q(IX,IY),R(IX,IY)/C13/U(IX,IY)
      COMMON /C15/S(IY),T(IX)/C10/V(IX),W(IY)
      COMMON /TRIDM/CL(IX),CM(IX),CN(IX),PQ(IX)
C
      NM1 = N-1
      MM1 = M-1
      NM2 = N-2
C
      DO 80 J = 1,M,MM1
C
        DO 60 I = 2,NM1
          IM1 = I - 1
          IP1 = I + 1
          CM(IM1) = T(IM1) * V(IM1) + T(I) * V(I)
          CL(IM1) = T(IM1)
          CN(IM1) = T(I)
          BA = T(IM1) * (V(IM1)+1.D0) * (Q(I,J)-Q(IM1,J)) / DX(IM1)
          BB = T(I)   * (V(I)+1.D0)   * (Q(IP1,J)-Q(I,J)) / DX(I)
          PQ(IM1) = BA + BB
 60     CONTINUE
C
        PQ(1)   = PQ(1)   - T(1)   * R(1,J)
        PQ(NM2) = PQ(NM2) - T(NM1) * R(N,J)
        CALL THOMAS(NM2)
C
        DO 70 I = 2,NM1
          IM1 = I - 1
          R(I,J) = PQ(IM1)
 70     CONTINUE
C
 80   CONTINUE
C
      RETURN
      END
```

```
C
C-----------------------------------------------------------------------
C
      SUBROUTINE RUXY2(N,M)
C
C*****************************************************************
C This subroutine determines the mixed-derivative of U with     *
C respect to xi and eta at each boundary grid node of the       *
C computational domain, where eta does not have the values of  *
C 0 and 1.  These derivatives are stored in array R.            *
C*****************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER(IX = 120 , IY = 10)
C
      COMMON /C6/DX(IX),DY(IY)
      COMMON /C7/P(IX,IY),Q(IX,IY),R(IX,IY)/C13/U(IX,IY)
      COMMON /C15/S(IY),T(IX)/C10/V(IX),W(IY)
      COMMON /TRIDM/CL(IX),CM(IX),CN(IX),PQ(IX)
C
      MM1 = M-1
      MM2 = M-2
C
      DO 80 I = 1,N
C
        DO 60 J = 2,MM1
          JM1 = J - 1
          JP1 = J + 1
          CM(JM1) = S(JM1) * W(JM1) + S(J) * W(J)
          CL(JM1) = S(JM1)
          CN(JM1) = S(J)
          BA = S(JM1) * (W(JM1)+1.D0) * (P(I,J)-P(I,JM1)) / DY(JM1)
          BB = S(J)   * (W(J)+1.D0)   * (P(I,JP1)-P(I,J)) / DY(J)
          PQ(JM1) = BA + BB
 60     CONTINUE
C
        PQ(1)   = PQ(1)   - S(1)   * R(I,1)
        PQ(MM2) = PQ(MM2) - S(MM1) * R(I,M)
        CALL THOMAS(MM2)
C
        DO 70 J = 2,MM1
          JM1 = J - 1
          R(I,J) = PQ(JM1)
 70     CONTINUE
C
 80   CONTINUE
C
      RETURN
      END
```

```
C
C-------------------------------------------------------------------------
C
      SUBROUTINE SCOSTW(NAF,NNN,NMID,SPAN)
C
C************************************************************
C This subroutine generates new X and Y coordinates caused    *
C by scaling, off-setting, and twisting.                      *
C************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER(IX = 120 , IY = 10)
C
      COMMON /C2/X(IX,IY),Y(IX,IY)/C1/TH1(IY),OSX(IY),OSY(IY),CL(IY)
      COMMON /C3/XI(IX,IY),YI(IX,IY)/C12/Z(IY)
      COMMON /C18A/XU1(IX,IY),YU1(IX,IY)
      COMMON /C18B/XL1(IX,IY),YL1(IX,IY)
C
      DIMENSION NNN(NAF),NMID(IY)
C
      OPEN(UNIT = 100,FILE = 'INPUT.DAT',ACCESS = 'APPEND'
     1    ,FORM='FORMATTED',CARRIAGE CONTROL = 'FORTRAN',STATUS = 'OLD')
C
      CALL CKFILE('AI1.DAT')
C
      OPEN(UNIT = 200,FILE = 'AI1.DAT' ,STATUS = 'NEW')
      OPEN(UNIT = 11 ,FILE = 'AI0B.DAT',STATUS = 'UNKNOWN')
C
      PI = DACOS(-1.D0)
C
      DO 40 J = 1,NAF
        NIX = NNN(J)
C
C  Setup the twist center.
C
        CX = OSX(J)
        CY = OSY(J)
C
C  The twist angle in degrees is converted to twist in radians.
C
        TAU  = PI * TH1(J) / 180.D0
        TCOS = DCOS(TAU)
        TSIN = DSIN(TAU)
C
C Offsetting the airfoil AND scaling it by multiplying by C.L.
C
        DO 20 I = 1,NIX
          XI(I,J) = (X(I,J)-CX) * CL(J)
          YI(I,J) = (Y(I,J)-CY) * CL(J)
          WRITE(11,*)XI(I,J)/SPAN,YI(I,J)/SPAN,Z(J)/SPAN
C
C Next we will TWIST the airfoil about the offset center.
C
```

```
                XR = XI(I,J)
                YR = YI(I,J)
C
C Find the new coordinates after twisting.
C
                XI(I,J) = XR * TCOS - YR * TSIN
                YI(I,J) = XR * TSIN + YR * TCOS
C
C Next in case the user wants to view the offsetted and twisted airfoils
C then we must store the airfoil data in data file AI1.DAT
C
                WRITE(200,100)XI(I,J)/SPAN,YI(I,J)/SPAN,Z(J)/SPAN
C
 20           CONTINUE
C
 40         CONTINUE
C
            CLOSE(UNIT = 11)
C
            DO 70 J = 1,NAF
              IM = NMID(J)
              NIX = NNN(J)
C
C Next the airfoil coordinates are put into upper and lower airfoil format.
C
              DO 50 I = 1,NIX
C
                IF (I .LE. IM) THEN
                  K = IM - I + 1
                  XL1(K,J) = XI(I,J)
                  YL1(K,J) = YI(I,J)
                ENDIF
C
                IF (I .GE. IM) THEN
                  K = I - IM + 1
                  XU1(K,J) = XI(I,J)
                  YU1(K,J) = YI(I,J)
                ENDIF
C
 50           CONTINUE
C
C Also the upper and lower airfoil coordinates of the airfoil,
C as well as twist offset and chord data are output to a file
C named INPUT.DAT
C
              WRITE(100,105)J,NNN(J)
              WRITE(100,106)
              WRITE(100,110)CL(J),TH1(J),OSX(J),OSY(J),Z(J)
              WRITE(100,107)
C
              DO 60 I = 1,IM
                WRITE(100,120)XU1(I,J),YU1(I,J),XL1(I,J),YL1(I,J)
```

```
 60      CONTINUE
C
 70    CONTINUE
C
       CLOSE(UNIT = 200)
       CLOSE(UNIT = 100)
C
       RETURN
  100 FORMAT(3(G18.9E3,2X))
  105 FORMAT(1H1,/,10X,'Scaled, offset, and twisted data for airfoil',I2
     1
     2        ,///,10X,'Number of original input data points are:',I3)
  106 FORMAT(//,10X,'Chord-',7X,'Twist',10X,'Twist center',9X,'Span'
     1    ,/,10X,'Length',7X,'Angle',10X,'X',10X,'Y',8X,'Coord.')
  107 FORMAT(//,17X,'XU',12X,'YU',13X,'XL',11X,'YL',/)
  110 FORMAT(5X,5(2X,F10.4))
  120 FORMAT(10X,4(2X,E12.5))
       END
```

```
C
C------------------------------------------------------------------------
C
      SUBROUTINE SECT(N,M,ICT,SPAN)
C
C*****************************************************************
C This subroutine generates the desired cross-sectional airfoil  *
C data for the user.                                              *
C*****************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER(IX = 120 , IY = 10)
C
      CHARACTER*14 ZTEMP
      CHARACTER*16 ZCOORD
C
      COMMON /C4/ZI(IX),ETA(IY)/C6/DZI(IX),DETA(IY)
      COMMON /C7/P(IX,IY),Q(IX,IY),R(IX,IY)/C13/U(IX,IY)
      COMMON /C10/V(IX),W(IY)/C9/AL(IX),BE(IY)/C12/Z(IY)
      COMMON /C5/XMIN(IY),XMAX(IY)/C8/IC,ZIT/C16/XIT,YIT
      COMMON /C20/U1(IX,IY),U2(IX,IY)
      COMMON /C22/XZ(IX),YZ(IX)/C25/SLE(IY),TAU
      COMMON /C30/NOW,RORINP(IX),CORINP(IX),TWIINP(IX),THKINP(IX)
      COMMON /C31/SIGMACH,SIGMATW,SIGMATH
C
      DIMENSION XL2(IX),YL2(IX),XU2(IX),YU2(IX)
      DIMENSION XL3(IX),YL3(IX),XU3(IX),YU3(IX),XSTAND(57)
C
C          Standard coordinate interpolation values
C
      DATA (XSTAND(I),I = 1,57)/
     1    0.D0,    .00025D0, .0005D0, .00075D0, .001D0, .0015D0,
     2     .002D0, .0025D0,  .005D0,  .01D0,    .02D0,  .03D0,
     3     .04D0,  .05D0,    .06D0,   .07D0,    .08D0,  .09D0,
     4     .1D0,   .125D0,   .15D0,   .175D0,   .2D0,   .225D0,
     5     .25D0,  .275D0,   .3D0,    .325D0,   .35D0,  .375D0,
     6     .4D0,   .425D0,   .45D0,   .475D0,   .5D0,   .525D0,
     7     .55D0,  .575D0,   .6D0,    .625D0,   .65D0,  .675D0,
     8     .7D0,   .725D0,   .75D0,   .775D0,   .8D0,   .825D0,
     9     .85D0,  .875D0,   .9D0,    .925D0,   .95D0,  .97D0,
     1     .98D0,  .99D0,    1.D0/
C
      OPEN(UNIT = 700,FILE = 'WTOUT.DAT',ACCESS = 'APPEND'
     1    ,FORM = 'FORMATTED',CARRIAGE CONTROL = 'FORTRAN'
     2    ,STATUS = 'UNKNOWN')
      OPEN(UNIT = 750,FILE = 'NTOUT.DAT',ACCESS = 'APPEND'
     1    ,FORM = 'FORMATTED',CARRIAGE CONTROL = 'FORTRAN'
     2    ,STATUS = 'UNKNOWN')
C
      CALL CKFILE('AIRFOIL.DAT')
      CALL CKFILE('NAIRFOIL.DAT')
      CALL CKFILE('NZCT.DAT')
```

```
      CALL CKFILE('FVBOUND.DAT')
      CALL CKFILE('TVBOUND.DAT')
      CALL CKFILE('ZSPAN.DAT')
      CALL CKFILE('ZNSPAN.DAT')
C
      OPEN(UNIT = 200,FILE = 'AIRFOIL.DAT' ,STATUS = 'NEW')
      OPEN(UNIT = 250,FILE = 'NAIRFOIL.DAT',STATUS = 'NEW')
      OPEN(UNIT = 300,FILE = 'NZCT.DAT'    ,STATUS = 'NEW')
      OPEN(UNIT = 400,FILE = 'FVBOUND.DAT' ,STATUS = 'NEW')
      OPEN(UNIT = 450,FILE = 'TVBOUND.DAT' ,STATUS = 'NEW')
      OPEN(UNIT = 500,FILE = 'ZSPAN.DAT'   ,STATUS = 'NEW')
      OPEN(UNIT = 550,FILE = 'ZNSPAN.DAT'  ,STATUS = 'NEW')
C
      CALL SYSTEM('CLS')
      WRITE(6,401)
      READ(5,*)LABEL
C
      IF (LABEL .EQ. 3) THEN
        IBOUND = IX/2
 246    WRITE(6,402)IBOUND
        READ(5,*)ILABEL
C
C Check to see if the number entered falls within dimensional bounds.'
C
        IF (ILABEL .LT. 24 .OR. ILABEL .GE. IBOUND) THEN
          WRITE(6,403)ILABEL
          GOTO 246
        ENDIF
C
C Check to see if the number entered is an odd number.
C
        IF (MOD(ILABEL,2) .EQ. 0) THEN
          WRITE(6,404)
          GOTO 246
        ENDIF
C
      ENDIF
C
C If the user wants to generate a blade
C
      IB = 1
      ICT = IC
C
C If ICT .EQ. 1 ---> the user wants to generate an airfoil at the
C                    ZIT span station.
C
C    ICT .EQ. 2 ---> the user wants to generate a 3D blade  which
C                    is represented by "NSS" equally spaced  span
C                    stations.
C
      IF (ICT .EQ. 1) THEN
        WRITE(200,*)N,IB
```

```
            XPL = 0.11D0
            YPL = 0.15D0
            DPL = ZIT/SPAN
            WRITE(ZTEMP,'(F8.4)')DPL
            ZCOORD = ''''//ZTEMP//''''
            WRITE(500,*)XPL,YPL,ZCOORD
            XPL = 0.95D0
            YPL = 0.6D0
            WRITE(550,*)XPL,YPL,ZCOORD
C
          ELSE
C
 100        WRITE(6,405)
            READ(5,*)NSS
C
            IF (NSS .LT. 3) THEN
              WRITE(6,406)
              GOTO 100
            ENDIF
C
            ZIT = Z(1)
            DZIT = (Z(M)-Z(1))/FLOAT(NSS-1)
            WRITE(200,*)N,NSS
C
          ENDIF
C
C Generate the airfoil section data at the ZIT span station
C
          WRITE(6,104)
C
C Initialize sums of the standard deviations for the chord, twist,
C and thickness distributions, respectively.
C
          SUMCH = 0.0D0
          SUMTW = 0.0D0
          SUMTH = 0.0D0
C
C First determine ETA at particular ZIT span station
C
 130      YIT = (ZIT-Z(1))/(Z(M)-Z(1))
C
C Also determine the leading edge S value at this ZIT span station
C
          SIGMAL = 20.0D0
          CALL SPT11D(M,ETA,SLE,SIGMAL,YIT,SMID)
C
C Next determine all the X values, N of them, that represent
C the ZIT span station
C
          CALL GETX(N,M,TAU,SMID)
C
C Next determine all the Y values, N of them, that represent
C the ZIT span station
```

```
C
      CALL GETY(N,M,TAU,SMID)
C
C Determine the chord and twist of this ZIT span station.  Also
C determine the chord/span ratio and the r/R  or Z/SPAN  ratio.
C These values are dumped into a data file named NZCT.DAT.
C
C First compute midpoint of trailing-edge base
C
      XTE = 0.5D0 * (XZ(1) + XZ(N))
      YTE = 0.5D0 * (YZ(1) + YZ(N))
C
C Next, find the most forward leading-edge point and the longest
C chord.
C
      CHORD = 0.D0
C
      DO 5 I = 1,N
        DX = XZ(I) - XTE
        DY = YZ(I) - YTE
        DIST = DSQRT(DX * DX + DY * DY)
C
        IF (DIST .GT. CHORD) THEN
          CHORD = DIST
          IA = I
          XLE = XZ(I)
          YLE = YZ(I)
        ENDIF
C
 5    CONTINUE
C
      DX = XTE - XLE
      DY = YTE - YLE
      TRAD = DATAN2 (DY , DX)
      TWIST = TRAD * 180.D0 / DACOS(-1.D0)
      ROR = ZIT / SPAN
      CHORDN = CHORD / SPAN
C
C Determine the twist center (TWISTX , TWISTX) which will make the
C leading-edge coordinate (0,0).
C
      CT = DCOS (TRAD)
      ST = DSIN (TRAD)
      TWISTX = -XZ(IA) * CT - YZ(IA) * ST
      TWISTY =  XZ(IA) * ST - YZ(IA) * CT
C
      XMA  = XZ(IA)
      XMI  = XZ(IA)
      YMIN = YZ(IA)
      YMAX = YZ(IA)
C
C The following calculations are to arrange the upper and lower
C airfoil x and y coordinates into a manner similar to that of
```

```
C airfoil tables.  Also output to data files  NZCT.DATA,
C AIRFOIL.DAT, WTOUT.DAT, and NTOUT.DAT
C
      G = SPAN
C
      DO 210 I = 1,N
C
        IF (I .LE. IA) THEN
          K = IA - I + 1
          XL2(K) = XZ(I)
          YL2(K) = YZ(I)
        ENDIF
C
        IF (I .GE. IA) THEN
          K = I - IA + 1
          XU2(K) = XZ(I)
          YU2(K) = YZ(I)
        ENDIF
C
        IF (XZ(I) .GT. XMA ) XMA = XZ(I)
        IF (XZ(I) .LT. XMI ) XMI = XZ(I)
        IF (YZ(I) .GT. YMAX) YMAX = YZ(I)
        IF (YZ(I) .LT. YMIN) YMIN = YZ(I)
C
        WRITE(200,500)XZ(I)/G, YZ(I)/G, ZIT/G
C
 210  CONTINUE
C
      IF (ICT .EQ. 2) THEN
        WRITE(400,500)ROR, YMAX/G, YMIN/G
        WRITE(450,500)ROR, XMA /G, XMI/G
      ENDIF
C
      WRITE(700,105)ZIT, ZIT
      WRITE(700,510)CHORDN, TWIST, TWISTX, TWISTY, ZIT/SPAN, SPAN
      WRITE(700,106)
C
      DO 220 I = 1,IA
        WRITE(700,520)XU2(I)/G, YU2(I)/G, XL2(I)/G, YL2(I)/G
 220  CONTINUE
C
C Also here the airfoil has its twist and offset removed so that
C the airfoil data without offset and twist can be output.
C
      OFFX = TWISTX/CHORD
      OFFY = TWISTY/CHORD
C
      DO 230 I = 1,N
        XNT = ( XZ(I) * CT + YZ(I) * ST) / CHORD + OFFX
        YNT = (-XZ(I) * ST + YZ(I) * CT) / CHORD + OFFY
        XZ(I)=XNT
        YZ(I)=YNT
```

```
 230  CONTINUE
C
      DO 231 I = 1,IA
        IBACK = IA - I + 1
        XL2(IBACK) = XZ(I)
        YL2(IBACK) = YZ(I)
 231  CONTINUE
C
      NPTL = IA
      IUPPER = 1
C
      DO 232 I = IA,N
        XU2(IUPPER) = XZ(I)
        YU2(IUPPER) = YZ(I)
        IUPPER = IUPPER + 1
 232  CONTINUE
C
      NPTU = N - IA + 1
C
      IF (LABEL .EQ. 1) THEN
        NUPPER = NPTU
C
        DO 248 I = 1,NUPPER
          XU3(I) = XU2(I)
          YU3(I) = YU2(I)
 248    CONTINUE
C
        NLOWER = NPTL
C
        DO 249 I = 1,NLOWER
          XL3(I) = XL2(I)
          YL3(I) = YL2(I)
 249    CONTINUE
C
      ELSE IF (LABEL .EQ. 2) THEN
        NUPPER = 57
        NLOWER = 57
        SIG = -1.D0
C
        DO 245 I = 1,57
          CALL SPT11D(NPTU,XU2,YU2,SIG,XSTAND(I),YSTAND)
          XU3(I) = XSTAND(I)
          YU3(I) = YSTAND
          CALL SPT11D(NPTL,XL2,YL2,SIG,XSTAND(I),YSTAND)
          XL3(I) = XSTAND(I)
          YL3(I) = YSTAND
 245    CONTINUE
C
      ELSE IF (LABEL.EQ.3) THEN
        NUPPER = ILABEL
        NLOWER = ILABEL
        SIG = -1.D0
```

```
            DXOUT = 1.D0/FLOAT(ILABEL-1)
            XXX = 0.D0
C
            DO 247 I = 1,ILABEL
              IF (I .EQ. ILABEL) XXX = 1.D0
              CALL SPT11D(NPTU,XU2,YU2,SIG,XXX,YYY)
              XU3(I) = XXX
              YU3(I) = YYY
              CALL SPT11D(NPTL,XL2,YL2,SIG,XXX,YYY)
              XL3(I) = XXX
              YL3(I) = YYY
              XXX = XXX + DXOUT
 247        CONTINUE
C
          ENDIF
C
C Determine the max. thickness THMAX
C
          THMAX = 0.0D0
          IXDATA = MAX0(NUPPER,NLOWER)
C
          DO 240 I = 1,IXDATA
            GTH = DABS(YU3(I) - YL3(I))
            IF (GTH .GT. THMAX) THMAX = GTH
  240 CONTINUE
C
          WRITE(750,107)ZIT
          WRITE(750,511)CHORDN, TWIST, THMAX, OFFX, OFFY, ZIT/SPAN, SPAN
          WRITE(750,108)CHORD
C
          DO 242 I = 1,IXDATA
            WRITE(750,520)XU3(I), YU3(I), XL3(I), YL3(I)
  242 CONTINUE
C
C Write airfoil data to data file NAIRFOIL.DAT
C
          DO 251 I = 1,NLOWER
            IBACK = NLOWER - I + 1
            XZ(I) = XL3(IBACK)
            YZ(I) = YL3(IBACK)
 251      CONTINUE
C
          ISTART = 1
          IF ( YL3(1) .EQ. YU3(1) ) ISTART = 2
          IP1 = NLOWER
C
          DO 252 J = ISTART,NUPPER
            IP1 = IP1 + 1
            XZ(IP1) = XU3(J)
            YZ(IP1) = YU3(J)
 252      CONTINUE
C
          DO 253 I = 1,IP1
```

```
          WRITE(250,500)XZ(I), YZ(I), ZIT/G
  253   CONTINUE
C
C IF the choice is to generate a blade then
C
        IF (ICT .EQ. 2) THEN
C
C Also here the standard deviations of the chord, twist, and
C thickness are determined.
C
          WRITE(300,500)ROR, CHORDN, TWIST, THMAX * 100.0D0
C
          IF (NOW .GT. 3) THEN
            CALL SPT11D(NOW,RORINP,CORINP,SIGMACH,ROR,CORACT)
            CALL SPT11D(NOW,RORINP,TWIINP,SIGMATW,ROR,TWIACT)
            CALL SPT11D(NOW,RORINP,THKINP,SIGMATH,ROR,THKACT)
C
          ELSE
C
            CALL SPT11D(NOW,RORINP,CORINP,-10.    ,ROR,CORACT)
            CALL SPT11D(NOW,RORINP,TWIINP,-10.    ,ROR,TWIACT)
            CALL SPT11D(NOW,RORINP,THKINP,-10.    ,ROR,THKACT)
C
          ENDIF
C
          DCH = (CHORDN - CORACT) * 100.0D0
          DTW = TWIST-TWIACT
          DTH = (THMAX * CHORDN - THKACT * CORACT) * 100.0D0
          SUMCH = SUMCH + DCH * DCH
          SUMTW = SUMTW + DTW * DTW
          SUMTH = SUMTH + DTH * DTH
C
C Increment ZIT and IB, the span station counter,
C and continue with the loop.
C
          ZIT = ZIT + DZIT
          IB  = IB  + 1
          IF (IB .LE. NSS) GOTO 130
C
C Next interpolate input chord, twist, maximun
C thickness distribution for graphic output
C
          CALL CKFILE('CTTINP.DAT')
          OPEN(UNIT = 900,FILE = 'CTTINP.DAT',STATUS = 'NEW')
          ZIT = Z(1)
          DZIT = (Z(M) - Z(1))/100.D0
C
          DO 300 I = 1 , 101
            ROR = ZIT/SPAN
C
            IF (NOW .GT. 3) THEN
C
              CALL SPT11D(NOW,RORINP,CORINP,SIGMACH,ROR,CORACT)
```

```
                CALL SPT11D(NOW,RORINP,TWIINP,SIGMATW,ROR,TWIACT)
                CALL SPT11D(NOW,RORINP,THKINP,SIGMATH,ROR,THKACT)
C
              ELSE
C
                CALL SPT11D(NOW,RORINP,CORINP,-10.    ,ROR,CORACT)
                CALL SPT11D(NOW,RORINP,TWIINP,-10.    ,ROR,TWIACT)
                CALL SPT11D(NOW,RORINP,THKINP,-10.    ,ROR,THKACT)
C
              ENDIF
C
              WRITE(900,*)ROR, CORACT, TWIACT, THKACT * 100.D0
              ZIT = ZIT + DZIT
 300        CONTINUE
C
            CLOSE(UNIT = 900)
C
            CALL CKFILE('CTTIN.DAT')
            OPEN(UNIT = 900,FILE = 'CTTIN.DAT',STATUS = 'NEW')
C
            DO 301 I = 1,NOW
              WRITE(900,*)RORINP(I),CORINP(I),TWIINP(I),THKINP(I)*100.D0
 301        CONTINUE
C
            CLOSE(UNIT = 900)
C
C Next the standard deviations are determined.
C
            SDCH = DSQRT(SUMCH/NSS)
            SDTW = DSQRT(SUMTW/NSS)
            SDTH = DSQRT(SUMTH/NSS)
            WRITE(6,407)SDCH, SDTW, SDTH
            PAUSE
          ENDIF
C
          CLOSE(UNIT = 200)
          CLOSE(UNIT = 250)
          CLOSE(UNIT = 300)
          CLOSE(UNIT = 400)
          CLOSE(UNIT = 450)
          CLOSE(UNIT = 500)
          CLOSE(UNIT = 550)
          CLOSE(UNIT = 600)
          CLOSE(UNIT = 610)
          CLOSE(UNIT = 620)
          CLOSE(UNIT = 700)
          CLOSE(UNIT = 750)
C
          RETURN
 104      FORMAT(1X,/,
     1        ' PLEASE WAIT..............'
     2  ,//,'   CALCULATIONS UNDER WAY ...................')
```

```
  105  FORMAT(1H1,/,14X,'************** OUTPUT DATA ****************'
      1  ,//,10X,'DATA FOR SPAN STATION  =  ',F7.4,' ARE:'
      2  ,/,10X,'____________________________________________________'
      3  ,//,10X,'These are the actual coordinates of the airfoil'
      4  ,/,10X,'at span station',F7.4
      5  ,//,10X,'                         WITH'
      6 ,//,8X,'Chord-',6X,'Twist',10X,'Twist Center',10X,'Z/R',8X,'Span'
      7 ,/,8X,'Length',6X,'Angle',9X,'X',12X,'Y',8X,'Coord.',7X,'R')
 C
  106  FORMAT(//,10X,'The upper and lower airfoil coordinates shown'
      1        ,' below have been',/,10X,'normalized w.r.t. span, R.'
      2       ,///,20X,'XU',12X,'YU',13X,'XL',11X,'YL',/)
 C
  107  FORMAT(1H1,/,14X,'************** OUTPUT DATA ****************'
      1  ,//,10X,'DATA FOR SPAN STATION',F7.4,' ARE:'
      2  ,/,10X,'____________________________________________________'
      3,///,6X,'Chord-',4X,'Twist',4X,'Thickness',5X,'Twist Center'
      4,6X,'Z/R',6X,'Span'
      5 ,/,6X,'Length',4X,'Angle',16X,'X/C',8X,'Y/C',5X,'Coord.',6X,'R')
 C
  108  FORMAT(//,10X,'The following upper and lower airfoil coordinates'
      1        ,' are for the',/,10X,'airfoil, with no twist, no offset,'
      2        ,' and are normalized w.r.t.',/,10X,'chord =',F10.6
      3       ,///,20X,'XU',12X,'YU',12X,'XL',11X,'YL',/)
 C
  401  FORMAT(
      1      You are now given the following three  options on  how  to'
      2 ,/,' specify the output X-coordinates of the generated airfoil,'
      3 ,/,' which has its twist and offset removed, at the chosen span'
      4 ,/,' station.'
      5 ,//,' Option 1.... The    X-coordinates    will    be    specified'
      6 ,/,'                such that there are more  points  around  the'
      7 ,/,'                leading edge.   This  feature depends on the'
      8 ,/,'                stretching parameter previously used.'
      9 ,//,' Option 2.... The    X-coordinates    will    be    specified'
      1 ,/,'                such   that   they       correspond     to  the'
      2 ,/,'                STANDARD 57 X coordinates; see manual.'
      3 ,//,' Option 3.... The    X-coordinates    will    be    specified'
      4 ,/,'                such that they  are  regularly  spaced.   The'
      5 ,/,'                spacing  depends  on  the  number  of  points'
      6 ,/,'                (to be input  next)  used  to  label  the  X-'
      7 ,/,'                coordinates.'
      8 ,//,' Enter your choice number here ---> ')
 C
  402  FORMAT(
      1     ' Enter the number  of  points  you  want  to  specify  the'
      2  ,/,' X-coordinates.'
      3   ,//,' An ODD integer number between 24 and 'I3,' ---> ')
 C
  403  FORMAT(' The number you just typed is out of bound.',/,
      1        ' Please try AGAIN !')
 C
  404  FORMAT('.............YOU ENTERED AN EVEN NUMBER.............'
```

```
     1    ,/,' **** YOU DID NOT ENTER AN ODD NUMBER. TRY AGAIN ! ****')
C
 405 FORMAT(' HOW MANY  span stations do you want your blade'
     1      ,/,' represented by ?'
     2      ,/,' ENTER a number greater than 3 '
     3      ,/,' (less than 40 for viewing 3D plot) ---> ')
C
 406 FORMAT(' The number of span stations for representing  the'
     /     ,/,' blade must be greater than 3.  Please try AGAIN !')
C
 407 FORMAT(' The standard deviations of ',/,
     /       ' Chord         = ',D15.6,/,
     /       ' Twist  angle = ',D15.6,/,
     /       ' Thickness    = ',D15.6)
C
 500 FORMAT(4(F9.5,2X))
 510 FORMAT(2X,6(2X,F10.4))
 511 FORMAT(2X,7(2X,F8.4))
 520 FORMAT(10X,4(2X,F12.5))
     END
```

```
C
C---------------------------------------------------------------------
C
      SUBROUTINE SORT(N,M,IL,JL)
C
C*******************************************************************
C This subroutine finds the computational domain grid sector in    *
C which the point (xi,eta) lies.  It returns the (xi,eta)          *
C coordinates of the bottom left corner node of the sector.        *
C*******************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER(IX = 120 , IY = 10)
C
      COMMON /C4/X(IX),Y(IY)/C16/XIT,YIT
C
      NM1 = N - 1
      MM1 = M - 1
C
 10   DO 20 I = 2,N
        IM1 = I - 1
        IF (X(I)-XIT) 20,20,30
 20   CONTINUE
C
      IL = NM1
C
 30   IF (X(IM1) .LE. XIT .OR. XIT .LE. X(I)) GOTO 40
C
      WRITE(6,200)
      PAUSE
      GOTO 50
C
 40   IL = IM1
C
 50   DO 60 J = 2,M
        JM1 = J - 1
        IF (Y(J)-YIT) 60,60,70
 60   CONTINUE
C
      JL = MM1
C
 70   IF (Y(JM1) .LE. YIT .OR. YIT .LE. Y(J)) GOTO 80
C
      WRITE(6,201)
      PAUSE
      GOTO 90
C
 80   JL = JM1
C
 90   RETURN
 200  FORMAT(1X,' ZI  COORDINATE OF POINT FALLS OUTSIDE THE BOUNDS.')
 201  FORMAT(1X,' ETA COORDINATE OF POINT FALLS OUTSIDE THE BOUNDS.')
```

```
     2310         END
```

```
C
C----------------------------------------------------------------------
C
      SUBROUTINE SPT11D(M,X,Y,SIGMA,XIT,YIT)
C
C*****************************************************************
C This subroutine calls SUBROUTINE CURVE1 and FUNCTION CURVE2 to  *
C perform a 1D tension spline interpolation of a given curve at a *
C given point.                                                     *
C*****************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER(IX = 120 , IY = 10)
C
      DIMENSION X(M),Y(M),YP(IX),TEMP(IX)
C
      CALL CURVE1(M,X,Y,SLP1,SLPN,YP,TEMP,SIGMA)
      IT = 1
      YIT = CURVE2(XIT,M,X,Y,YP,SIGMA,IT)
C
      RETURN
      END
```

```
C
C------------------------------------------------------------------------
C
      SUBROUTINE THOMAS(N)
C
C*************************************************************
C This subroutine solves tridiagonal system by the THOMAS'   *
C algorithm.                                                 *
C*************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER(IX = 120)
C
      COMMON /TRIDM/CL(IX),CM(IX),CN(IX),PQ(IX)
C
      NM1 = N - 1
C
      IF (N .LE. 1) THEN
        PQ(1) = PQ(1) / CM(1)
        RETURN
      ENDIF
C
C**************************************
C Establish upper triangular matrix   *
C**************************************
C
      DO 20 I = 2,N
        IM1 = I - 1
        CL(I) = CL(I) / CM(IM1)
        CM(I) = CM(I) - CL(I) * CN(IM1)
        PQ(I) = PQ(I) - CL(I) * PQ(IM1)
 20   CONTINUE
C
C***********************
C Back substitution    *
C***********************
C
      PQ(N) = PQ(N) / CM(N)
C
      DO 30 I = NM1,1,-1
        PQ(I) = (PQ(I) - CN(I) * PQ(I+1)) / CM(I)
 30   CONTINUE
C
      RETURN
      END
```

```
C
C-----------------------------------------------------------------------
C
      SUBROUTINE TINPUT(NAF,NDF,NNN,NMID,SPAN,IDC,N1,SIGMA,IU)
C
C*************************************************************
C This subroutine performs the "TERMINAL" input mode option   *
C*************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER(IX = 120 , IY = 10)
C
      CHARACTER*1 SCALE
      CHARACTER*12 NAME(IY),FNAME,DNAME
C
      COMMON /C1/TH1(IY),OSX(IY),OSY(IY),CL(IY)/C2/XX(IX,IY),YY(IX,IY)
      COMMON /C12/ZZ(IY)/C11/ALPHA,BETA
      COMMON /C14A/XU(IX,IY),XL(IX,IY)
      COMMON /C14B/YU(IX,IY),YL(IX,IY)
      COMMON /C30/NOW,ROR(IX),COR(IX),TWI(IX),THK(IX)
      COMMON /C31/SIGMACH,SIGMATW,SIGMATH
C
      DIMENSION NDF(IY),NNN(IY)
      DIMENSION ZE(IX),CHORD(IX),VAR(8),THKN(IX),NMID(IY)
C
      DATA VAR/ 300.0D0,0.0D0,5.0D0,0.0D0,1.0D0,1.0D0,1.0D0,1.0D0/
C
 12   WRITE(6,93)
      READ(IU,*)IDC
C
      IF (IDC .LT. 1 .OR. IDC .GT. 4) THEN
        WRITE(6,103)
        PAUSE
        GOTO 12
      ENDIF
C
      IF (IDC .EQ. 2) DNAME = 'CTE1.DAT'
C
      IF (IDC .EQ .3) DNAME = 'CTE2.DAT'
C
      IF (IDC .EQ. 4) THEN
        WRITE(6,94)
        READ(IU,*)DNAME
      ENDIF
C
      IF (IDC .GT. 1) THEN
        OPEN(UNIT = 10 , FILE = DNAME , STATUS = 'OLD' , IOSTAT = IERR)
        IF (IERR .GT. 0) THEN
          WRITE(6,95)DNAME
          PAUSE
          GO TO 12
        ENDIF
```

```
         ENDIF
C
         CALL CKFILE('CTEX1.DAT')
         OPEN(UNIT = 20 , FILE = 'CTEX1.DAT', STATUS = 'NEW')
C
C  Step 2
C
 2       WRITE(6,96)IY
         READ(IU,*)NAF
C
         IF (NAF .LT. 2 .OR. NAF .GT. IY) THEN
           WRITE(6,97)IY
           PAUSE
           GOTO 2
         ENDIF
C
         WRITE(6,98)
         READ(IU,*)SPAN
C
         CALL SYSTEM('CLS')
         WRITE(6,99)
         READ(IU,*)SCALE
C
         IF (IDC .GT. 1) THEN
           READ(10,*)NOW
C
           DO 8 I = 1,NOW
             READ(10,*)ROR(I),COR(I),TWI(I),THK(I)
             WRITE(20,205)ROR(I),COR(I),TWI(I),THK(I) * 100.0D0
             ZE(I)    = ROR(I) * SPAN
             CHORD(I) = COR(I) * SPAN
 8         CONTINUE
C
           CLOSE(UNIT = 10)
         ENDIF
C
         IF (IDC.EQ.1) NOW = NAF
         MAX = 0
         OPEN(UNIT = 8 , FILE = 'AI0A.DAT' , STATUS = 'UNKNOWN')
C
C  Step 3
C
         DO 100 K = 1,NAF
C
 9         CALL SYSTEM('CLS')
           WRITE(6,302)K
C
           READ(IU,*)NDF(K)
C
           IF (NDF(K) .GT. 7) THEN
             WRITE(6,103)
             PAUSE
```

```
            GOTO 9
          ENDIF
C
          IF (NDF(K) .EQ. 1) NAME(K) = 'S806A.DAT'
          IF (NDF(K) .EQ. 2) NAME(K) = 'S805A6.DAT'
          IF (NDF(K) .EQ. 3) NAME(K) = 'S805A.DAT'
          IF (NDF(K) .EQ. 4) NAME(K) = 'S805A7.DAT'
          IF (NDF(K) .EQ. 5) NAME(K) = 'S807.DAT'
          IF (NDF(K) .EQ. 6) NAME(K) = 'S808.DAT'
          IF (NDF(K) .EQ. 7) THEN
            WRITE(6,108)
            READ(IU,*)NAME(K)
          ENDIF
C
          FNAME = NAME(K)
          LUN = 10
          OPEN(UNIT = LUN,FILE = FNAME , STATUS = 'OLD' , IOSTAT = IERR)
          IF (IERR .GT. 0) THEN
            WRITE(6,109)
            PAUSE
            GOTO 9
          ENDIF
C
          WRITE(6,110)FNAME
C
          READ(LUN,*)NU
          READ(LUN,*)NL
          N = NU + NL - 1
          NNN(K) = N
C
          DO 50 I = 1,NU
            READ(LUN,*)XU(I,K),YU(I,K)
 50       CONTINUE
C
          DO 60 I = 1,NL
          READ(LUN,*)XL(I,K),YL(I,K)
 60       CONTINUE
C
          CLOSE(UNIT = LUN)
C
          WRITE (6,305) FNAME , K
          PAUSE
          CALL SYSTEM('CLS')
C
          IF (IDC  .GT. 1) THEN
            SP1 = SPAN * ROR(1)   - 1.0D-10
            SP2 = SPAN * ROR(NOW) + 1.0D-10
          ENDIF
C
          IF (IDC .EQ. 1) THEN
            SP1 = 0.D0
            SP2 = SPAN
```

```
            ENDIF
C
            IF (K .EQ. 1) THEN
 61           WRITE(6,303)SP1,SP2
              WRITE(6,304)K
              READ(IU,*)ZZ(K)
C
              IF (ZZ(K) .LT. SP1 .OR. ZZ(K) .GT. SP2) GOTO 61
C
              IF (ZZ(K) .EQ. SP2 .AND. K .NE. NAF) THEN
                WRITE(6,111)
                PAUSE
                GOTO 61
              ENDIF
C
            ELSE
C
              KM1 = K-1
 62           WRITE(6,303)ZZ(KM1),SP2
              WRITE(6,304)K
              READ(IU,*)ZZ(K)
C
              IF (ZZ(K) .LE. ZZ(KM1) .OR. ZZ(K) .GT. SP2) GOTO 62
C
              IF (ZZ(K) .EQ. SP2 .AND. K .NE. NAF) THEN
                WRITE(6,111)
                PAUSE
                GOTO 62
              ENDIF
C
            ENDIF
C
            IF (IDC .GT. 1) THEN
              ZIT = ZZ(K)
              SIGMACH = -1.D0
              SIGMATW = -1.D0
              SIGMATH = -1.D0
              CALL SPT11D(NOW , ZE , CHORD , SIGMACH , ZIT , CHL  )
              CALL SPT11D(NOW , ZE , TWI   , SIGMATW , ZIT , TWA  )
              CALL SPT11D(NOW , ZE , THK   , SIGMATH , ZIT , THICK)
              CL(K)   = CHL
              TH1(K)  = TWA
              THKN(K) = THICK
C
            ELSE
C
              WRITE(6,112)
              READ(IU,*)TH1(K)
C
 63           WRITE(6,113)K
              READ(IU,*)CL(K)
C
              IF (CL(K) .LE. 0.D0) THEN
```

```
                  WRITE(6,114)
                  GOTO 63
                ENDIF
C
 64             WRITE(6,115)
                READ(IU,*)THKN(K)
C
                IF (THKN(K) .GT. 1.0D0) THEN
                  WRITE(6,116)
                  GOTO 64
                ENDIF
C
                THICK = THKN(K)
C
                ROR(K) = ZZ(K)/SPAN
                COR(K) = CL(K)/SPAN
                TWI(K) = TH1(K)
                THK(K) = THKN(K)
C
                WRITE(20,205)ROR(K) , COR(K) , TH1(K) , THKN(K)*100.0D0
C
              ENDIF
C
              CALL SYSTEM('CLS')
C
 13           WRITE(6,117)
              READ(IU,*)TWCTR
C
              CALL SYSTEM('CLS')
C
              IF (TWCTR .EQ. 1) THEN
                WRITE(6,118)
                READ(IU,*)OSX(K)
                WRITE(6,119)
                READ(IU,*)OSY(k)
              ENDIF
C
              IF (TWCTR .EQ. 2) THEN
                TWCTRX = 0.3D0
                SIG    = -1.D0
                CALL SPT11D(NU,XU,YU,SIG,TWCTRX,TWCTYU)
                CALL SPT11D(NL,XL,YL,SIG,TWCTRX,TWCTYL)
                OSX(K) = TWCTRX
                OSY(K) = 0.5D0 * (TWCTYU-TWCTYL)
              ENDIF
C
              CALL SYSTEM('CLS')
C
C SET UP FOR SCALING
C
              IF (SCALE .EQ. 'Y' .OR. SCALE .EQ. 'y') THEN
                CALL CKFILE('AFXYIN')
```

```
      OPEN (15 , FILE = 'AFXYIN' , STATUS = 'NEW')
      WU  = 1.0D0
      NU1 = NU-1
      VAR(1) = 0.D0
      WRITE(15,400)CHARNB(NAME(K))
      WRITE(15,401)(VAR(I),I = 1,8)
      WRITE(15,402)NU
      YU(NU,K) = (YU(NU,K) + YU(NU1,K)) / 2.0D0
C
      DO 66 I = 1,NU
        WRITE(15,403)XU(I,K),YU(I,K),WU
 66   CONTINUE
C
      WRITE(15,402)NL
C
      DO 67 I = 1,NL
        WRITE(15,403)XL(I,K),YL(I,K),WU
 67   CONTINUE
C
      CLOSE(UNIT = 15)
C
      CALL CKFILE('SCXYIN')
      CALL SYSTEM('CLS')
C
      WRITE(6,191)
C
      CALL SYSTEM('SMOOTH2')
C
      OPEN (UNIT = 16 , FILE = 'SCALEINP.DAT' , ACCESS = 'APPEND'
     /        , STATUS = 'OLD')
C
      XNT = 1.0D0
      WRITE(16,402)XNT
      WRITE(16,402)THICK
      CLOSE(UNIT = 16)
C
      CALL SYSTEM('RENAME SCALEINP.DAT SCXYIN')
C
      CALL SYSTEM('SCALE1')
C
C CONCLUSION OF SCALING
C
C
C     RETRIEVE  SCALED DATA FROM DATA FILES
C
      OPEN (UNIT = 17 , FILE = 'XUXLSS.DAT' , STATUS = 'OLD')
      READ(17,*)NU
      READ(17,*)(XU(I,K),YU(I,K),I = 1,NU)
      READ(17,*)NL
      READ(17,*)(XL(I,K),YL(I,K),I = 1,NL)
      CLOSE(UNIT = 17)
C
    ENDIF
```

```
C
C  CONTINUE AND FORM THE AIRFOIL
C
           N = NU + NL - 1
           NNN(K)    = N
           NMID(K)   = NL
           YU(NU,K) = 0.0D0
           YL(NL,K) = 0.0D0
C
           DO 70 I = 1,N
             J = NL - I  + 1
             L = I  - NL + 1
C
             IF (I .LT. NL) THEN
C
               XX(I,K) = XL(J,K)
               YY(I,K) = YL(J,K)
C
             ELSE
C
               XX(I,K) = XU(L,K)
               YY(I,K) = YU(L,K)
C
             ENDIF
C
             WRITE(8,*)XX(I,K),YY(I,K),ZZ(K)/SPAN
C
 70        CONTINUE
C
           IF (NNN(K) .GT. MAX) MAX = NNN(K)
C
 100     CONTINUE
C
         CLOSE (UNIT = 8)
C
C  Step 4
C
         CALL SYSTEM('CLS')
         SIGMA = -1.D0
 101     WRITE(6,122)IX
         READ(IU,*)N1
C
C Check to see if the number entered falls within dimensional bounds.
C
         IF (N1 .LT. 50 .OR. N1 .GE. IX) GOTO 101
C
C Check to see if the number entered is an odd number.
C
         IF (MOD(N1,2) .EQ. 0) THEN
           WRITE(6,123)
           GOTO 101
         ENDIF
```

```
C
 102  CALL SYSTEM('CLS')
C
      WRITE(6,124)
      READ(IU,*)ALPHA
C
      WRITE(6,125)
      READ(IU,*)BETA
C
      IF (ALPHA .GT. 10000.D0 .OR. ALPHA .LT. 0.D0 .OR.
     /    BETA  .GT. 10000.D0 .OR. BETA.  LT. 0.D0) THEN
        WRITE(6,126)
        PAUSE
        GOTO 102
      ENDIF
C
      WRITE(6,104)
C
      OPEN(UNIT = 100 , FILE = 'INPUT.DAT' , ACCESS = 'APPEND'
     /    , FORM  = 'FORMATTED',CARRIAGE CONTROL = 'FORTRAN'
     /    , STATUS = 'UNKNOWN')
C
      WRITE(100,105)NAF,ALPHA,BETA
C
      DO 200 K = 1,NAF
C
        WRITE(100,106)K,NAME(K)
        WRITE(100,203)
        WRITE(100,220)CL(K),TH1(K),OSX(K),OSY(K),ZZ(K),THKN(K)
        WRITE(100,107)
        IA = (NNN(K) + 1) / 2
C
        IF (NMID(K) .GE. IA) THEN
          NREF = NMID(K)
        ELSE
          NREF = NNN(K) - NMID(K) + 1
        ENDIF
C
        DO 120 I = 1,NREF
          WRITE(100,210)XU(I,K),YU(I,K),XL(I,K),YL(I,K)
 120    CONTINUE
C
 200  CONTINUE
C
      N = MAX
C
      CLOSE(UNIT = 20)
      CLOSE(UNIT = 100)
C
      RETURN
  93  FORMAT(1X,//,
     1     ' Since this is a free nation , you are given a choice of'
     2  ,/,' how you.would like to generate your blade.'
```

```
     3  ,/,' Following are your options:'
     4 ,//,' 1.....GENERATE   YOUR  OWN  TWISTED  AND  TAPERED  BLADE.'
     5  ,/,'       (i.e., The user will input the chord/twist/thickness'
     6  ,/,'               distribution at the terminal.)'
     7 ,//,' 2.....USE Example 1 chord/twist/thickness  distribution.'
     8 ,//,' 3.....USE Example 2 chord/twist/thickness  distribution.'
     9 ,//,' 4.....USE A chord/twist/thickness  distribution that  is'
     1  ,/,'            not listed here.'
     2 ,//,' NOTE: Option 4 can only  be  used  if  you  have  already'
     3  ,/,' created a chord/twist/thickness distribution data file.'
     4  ,/," (Please refer to the user's manual  on  how  to  generate"
     5  ,/,' this data file.)'
     6 ,//,' Please enter your choice number here ---> ')
94   FORMAT(1X,//,
     1        ' PLEASE ENTER THE NAME OF DATA FILE CONTAINING THE'
     2     ,/,' chord/twist/thickness distribution required.'
     3     ,/,' File name extension must be .DAT '
     4     ,/,'   e.g.  File name   --->  JOSEPH.DAT '
     5    ,//,'         File name   ---> ')
95   FORMAT(1X,/,' FILE NAME ',A12,' DOES NOT EXIST.  TRY AGAIN !')
96   FORMAT(1X,//,
     1       ' How many airfoils do you want to input to form the blade?'
     2   ,/,' *** Enter a number between  2  and',I3,' ***'
     3   ,/,' ---> ')
97   FORMAT(1X,/,' *** ERRROR! Input must be between 2 and',I3,'.')
98   FORMAT(1X,//,
     1       ' Enter the span of the desired blade (i.e., the radius)'
     2   ,/,' ---> ')
99   FORMAT(1X,//,
     1      ' Do you want to scale the input airfoils ?'
     2  ,//,'  Enter          N  for NO'
     3   ,/,'                 Y  for YES'
     4  ,//,' Enter your choice here ---> ')
103  FORMAT(1X,/,
     1      ' You must choose an existing option, please try AGAIN!')
104  FORMAT(1X,//,
     1       ' PLEASE WAIT.......'
     2   ,/,'         CALCULATIONS ARE TAKING PLACE .........')
105  FORMAT(/,10X,' Number of input airfoils = ',I2,//,10X,
     /       ' Tension parameters: Alpha = ',F7.2,', and Beta = ',F7.2)
106  FORMAT(1H1,/,10X,' Input data file name for airfoil',I2,' is '
     /      ,A12,' and the input data are:',//)
107  FORMAT(//,17X,'XU',12X,'YU',13X,'XL',11X,'YL',/)
108  FORMAT(1X,/,
     1' Enter the name of file containing the normalized airfoil data'
     2  ,/,' File name extension must be .DAT'
     3  ,/,'   e.g.  File name   --->  JOSEPH.DAT '
     4 ,//,'   Please enter the file name here  ---> ')
109  FORMAT(1X,//,' File name does not exist !!  Please try AGAIN!')
110  FORMAT(1X,/,' Reading data from data file ',A)
111  FORMAT(1X,/,'  YOU HAVE TO LEAVE ROOM FOR MORE SECTIONS.'
     1          ',     REDUCE Z !')
```

```
  112  FORMAT(1X,/,' Enter the twist angle (in degrees) ---> ')
  113  FORMAT(1X,/,' Enter the chord length of airfoil ',I3,' ---> ')
  114  FORMAT(1X,/,' The chord length cannot be zero or less.'
     1                Try again !')
  115  FORMAT(1X,/,
     1       ' Enter the maximum thickness/chord ratio.'
     2   ,/,' The value should be a decimal less than 1.0. ---> ')
  116  FORMAT(1X,/,
     1         ' The maximum thickness/chord ratio cannot be'
     2     ,/,' greater than 1.0. Try again !')
  117  FORMAT(1X,/,
     1       ' Enter twist center X- and Y-coordinates.'
     3  ,//,' ENTER   1      if you want to  input  by  yourself'
     4  ,//,' ENTER   2      if you want to determine by the program'
     5  ,//,' NOTE:  OPTION 2 only applies to the case  where  the'
     6   ,/,'         twist center is located at  the  intersection'
     7   ,/,'         of  X = 1/3  and  the meanline.'
     8  ,//,' PLEASE ENTER YOUR CHOICE NUMBER HERE  ---> ')
  118  FORMAT(1X,//,' Enter X-coordinate here  ---> ')
  119  FORMAT(1X,//,' Enter Y-coordinate here  ---> ')
  121  FORMAT(1X,/,' Please wait, program is running!')
  122  FORMAT(1X,/,
     1       ' ENTER the number of points you want per airfoil:'
     2  .,/,' An ODD integer number between 50 and ',I4,/,' ---> ')
  123  FORMAT(1X,/,
     1         ' ..............YOU ENTERED AN EVEN NUMBER..............'
     2     ,/,' **** YOU DID NOT ENTER AN ODD NUMBER. TRY AGAIN ! ****')
  124  FORMAT(1X,/,
     1       ' WE NOW NEED INPUT FOR BLADE SURFACE INTERPOLATION.'
     2   ,/,' Enter the tension factors in X and Z directions,'
     3   ,/,' respectively.'
     4   ,/,' The numbers must be greater than 0 (zero)'
     4   ,/,' and less than 1000'
     5   ,//,' ENTER X TENSION FACTOR (standard value is 1) ---> ')
  125  FORMAT(1X,/,
     1         ' ENTER Z TENSION FACTOR (standard value is 1) ---> ')
  126  FORMAT(1X,/,
     1             X and Z tension factors must be greater than zero'
     2     ,/,'  and less than 1000.  Please enter these two'
     3     ,/,'  parameters AGAIN !')
  191  FORMAT(' Please wait, scaling program is running!')
  201  FORMAT(1X,I2)
  202  FORMAT(1X,2F8.2)
  203  FORMAT(7X,'Chord-',7X,'Twist',10X,'Twist Center',9X,'Span',
     /   9X,'max',/,7X,'Length',7X,'Angle',10X,'X',10X,'Y',8X,'Coord.'
     /  ,5X,'thickness')
  205  FORMAT(1X,4(F9.5,1X))
  210  FORMAT(10X,4(2X,E12.5))
  220  FORMAT(2X,6(2X,F10.4))
  301  FORMAT(A1)
  302  FORMAT(
     /   ' Below are airfoil  types  to  choose  from.  Data are',/,
     /   ' contained in data files.  They are already normalized',/,
```

```
     /   ' with respect to chord.',/
     /   ' If you do not desire any of the stated airfoil types,',/,
     /   ' you can choose option 7.',/,
     /   '  ',/,
     /   '   1.....S806A   Tangler  Somers  thin  airfoil',/,
     /   '   2.....S805A/6A    "       "      "      "   ',/,
     /   '   3.....S805A       "       "      "      "   ',/,
     /   '   4.....S805A/7A    "       "      "      "   ',/,
     /   '   5.....S807        "       "      "      "   ',/,
     /   '   6.....S808        "       "      "      "   ',/,
     /   '       ',/,
     /   '   7....None of the above.  I will enter my own.',/,
     /   '      ',/,
     /   ' Enter desired airfoil number for position ',I2,' here ---> ')
303  FORMAT(' The Z coordinate must be between',F7.2,' and ',F7.2)
304  FORMAT(' Enter Z coordinate for airfoil ',I2,' here ---> ')
305  FORMAT(' Data from file ',A,' for airfoil ',I2,' have been read')
400  FORMAT(A)
401  FORMAT(8F10.5)
402  FORMAT(F10.5)
403  FORMAT(3E15.6)
     END
```

```
C
C----------------------------------------------------------------------
C
      SUBROUTINE TRANS(N,NNN,NMID,NAF,SIGMA)
C
C*********************************************************************
C  This subroutine transforms the data from the physical coordinates  *
C  to the computational coordinates.                                  *
C*********************************************************************
C  The sequence of steps is as follows:                               *
C
C    1.    The normalized Z coordinate ETA is computed.
C    2.    The polygonal arclength S is first computed.
C    3.    The polygonal arclenth is normalised and saved as ZI       *
C    4.    New values of Y and X are interpolated for by using the    *
C          normalised polygonal arclength, ZI, as the independent     *
C          variable.                                                  *
C    5.    The new values of X are dumped into array U1               *
C    6.    The new values of Y are dumped into array U2               *
C                                                                     *
C*********************************************************************
C  There are going to be two computational domains.                   *
C  Both computational domains consist of ZI and ETA as independent    *
C  variables.                                                         *
C     In the first, X is the dependent variable. ie. X = X(zi,eta)    *
C     In the second, Y is the dependent variable. ie. Y = Y(zi,eta)   *
C  The maximum S value, SMAX(I), at each span station, is stored for  *
C  later use.                                                         *
C*********************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER(IX = 120 , IY = 10)
C
      COMMON /C3/X(IX,IY),Y(IX,IY)/C12/Z(IY)/C4/ZI(IX),ETA(IY)
      COMMON /C5/XMIN(IY),XMAX(IY)
      COMMON /C20/U1(IX,IY),U2(IX,IY)/C25/SLE(IY),TAU
C
      DIMENSION NNN(IY),X1(IX),Y1(IX),SN(IX),S(IX),NMID(IY)
C
      CALL SYSTEM('CLS')
      WRITE(6,100)
      READ(5,*)TAU
C
      CALL SYSTEM('CLS')
      WRITE(6,101)
C
      DO 40 J = 1,NAF
        NIX = NNN(J)
C
C  Compute the normalized Z coordinate
C
        ETA(J) = (Z(J)-Z(1))/(Z(NAF)-Z(1))
```

```
C
C  Compute the polygonal arclenth. Initialize the first value as zero
C
        S(1) = 0.0D0
C
        DO 10 I = 2,NIX
          XSQ  = X(I,J) - X(I-1,J)
          YSQ  = Y(I,J) - Y(I-1,J)
          S(I) = DSQRT(XSQ*XSQ+YSQ*YSQ) + S(I-1)
 10     CONTINUE
C
C The length of the spanstation polygonal arc is
C
        SMAX = S(NIX)
C
C Compute the normalized polygonal arclengths.
C
        DO 20 I = 1,NIX
          SN(I) = S(I)/SMAX
          X1(I) = X(I,J)
          Y1(I) = Y(I,J)
 20     CONTINUE
C
        IA = NMID(J)
        SLE(J) = SN(IA)
        SAME = SLE(J) * (SLE(J)-1.D0)
        CONST1 = (SLE(J) * SLE(J) - 0.5D0) / SAME
        CONST2 = (0.5D0 - SLE(J)) / SAME
C
        DO 50 I = 1,NIX
          SN(I) = SN(I) * (CONST1 + CONST2 * SN(I) )
 50     CONTINUE
C
        SLE(J) = 0.5D0
        SMID = SLE(J)
C
C Resample X and Y using the normalized polygonal arc length as the
C independent variable.
C
        B1  = 1.0D0 + (DEXP(TAU)  - 1.0D0) * SMID
        B2  = 1.0D0 + (DEXP(-TAU) - 1.0D0) * SMID
        B   = DLOG(B1/B2) / (2.0D0*TAU)
        DZI = 1.0D0 / FLOAT(N-1)
C
        DO 30 I = 1,N
          ZIT = DZI*FLOAT(I-1)
          ZZIT = SMID*(1.0D0+(DSINH(TAU*(ZIT-B) )/DSINH(TAU*B) ) )
          CALL SPT11D(NIX,SN,X1,SIGMA,ZZIT,X1IT)
          CALL SPT11D(NIX,SN,Y1,SIGMA,ZZIT,Y1IT)
          U1(I,J) = X1IT
          U2(I,J) = Y1IT
          ZI(I)   = ZZIT
```

```
 30      CONTINUE
C
 40    CONTINUE
C
       RETURN
 100   FORMAT(1X,//,
     1       ' ENTER THE STRETCHING PARAMETER FOR CONCENTRATING'
     2    ,/,' THE AIRFOIL DATA POINTS ABOUT THE LEADING EDGE.'
     3   ,//,' The STANDARD VALUE RANGES BETWEEN 4.0 AND 6.0'
     4    ,/,'  THE FEWER POINTS YOU USED TO REPRESENT THE'
     5    ,/,'   AIRFOIL  CROSS SECTION,  THE  SMALLER'
     6    ,/,'           VALUE YOU SHOULD USE.'
     7   ,//,' If no stretching desired, enter 0.0001'
     8   ,//,' Enter your choice here ---> ')
 101   FORMAT(1X,
     1       ' PLEASE WAIT.......'
     2    ,/,'      CALCULATIONS IN PROGRESS..............')
       END
```

```
C
C-----------------------------------------------------------------------
C
      SUBROUTINE TRANSPO(N,M,A,AT)
C
C****************************************************************
C This subroutine finds the transpose of matrix A and stores    *
C the result in matrix AT.                                      *
C****************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
      DIMENSION A(N,M),AT(M,N)
C
      DO 20 I = 1,N
C
        DO 10 J = 1,M
          AT(J,I) = A(I,J)
 10     CONTINUE
C
 20   CONTINUE
C
      RETURN
      END
```

APPENDIX B

DERIVATION OF EQUATIONS (2.2) AND (2.5)

For a given partition

$$a = x_1 < x_2 < \ldots.. < x_i < \ldots.. < x_{n-1} < x_n = b$$

of an interval {a,b}, and a set of tension parameters α_i ($1 \le i \le n$), g(x) is a tension spline if it satisfies the following conditions [1]:

$$g \in C^3[a,b] \tag{B.1}$$

and

$$(D^4 - \alpha_i^2 D^2)g = 0 \tag{B.2}$$

in each subinterval i, where $x_i \le x \le x_{i+1}$. In equation (B.1) C^3 is a differentiable space up to third order and D in equation (B.2) is the differential operator defined as D = d/dx.

Accordingly, as in [2], the general solution of equation (B.2) is

$$g_i(x) = A_{i,1} + A_{i,2}(x - x_i) + A_{i,3}e^{\alpha(x-x_i)} + A_{i,4}e^{-\alpha(x-x_i)} \tag{B.3}$$

where coefficients $A_{i,1}$, $A_{i,2}$, $A_{i,3}$, and $A_{i,4}$ are unknown constants to be determined by the boundary conditions for each subinterval i.

For convenience in applying the boundary conditions for a finite domain, equation (B.3) is rewritten [2] in terms of the hyperbolic functions as

$$g_i(x) = A_{i,1}(x - x_i) + A_{i,2}(x_{i+1} - x) + A_{i,3}\psi_i(x - x_i) + A_{i,4}\psi_i(x_{i+1} - x) \tag{B.4}$$

where

$$\psi_i(x) = \frac{\Delta x_i \sinh(\alpha_i x) - x \sinh(\alpha_i \Delta x_i)}{\sinh} \qquad (B.5)$$

and

$$\Delta x_i = x_{i+1} - x_i \qquad (B.6)$$

with the following conditions:

$$g(x_i) = y_i \qquad (B.7)$$

$$g(x_{i+1}) = y_{i+1} \qquad (B.8)$$

$$g'(x_i) = y'_i \qquad (B.9)$$

$$g'(x_{i+1}) = y'_{i+1} \qquad (B.10)$$

where superscript ' denotes the derivative of y with respect to x.

Equation (B.4) in summation form is actually equation (2.2).

There are four unknown coefficients ($A_{i,k}$, k = 1, 2, 3, and 4 for each i) in equation (B.4) and there are four conditions ([B.7] through [B.10]) to solve them. These coefficients are determined as follows.

By solving equation (B.4) at $x = x_i$, and $x = x_{i+1}$ and using (B.7) and (B.8) as the boundary conditions, one can get the following two relations:

$$A_{i,2} \Delta x_i = y_i \qquad (B.11)$$

for $x = x_i$, and

$$A_{i,1} \Delta x_i = y_{i+1} \qquad (B.12)$$

for $x = x_{i+1}$.

The other two relations are obtained by solving for the derivative of the interpolating function (B.4), at the boundaries of the subdomain, i, and then equating the results to the boundary conditions (B.9) and (B.10). From equation (B.4), the first derivative of g_i with respect to x is

$$g'_i(x) = A_{i,1} - A_{i,2} + A_{i,3}\left[\frac{\Delta x_i \alpha_i \cosh(\alpha_i x - x_i) - \sinh(\alpha_i \Delta x_i)}{\sinh(\alpha_i \Delta x_i) - \alpha_i \Delta x_i}\right]$$

$$-A_{i,4}\left[\frac{\Delta x_i \alpha_i \cosh(\alpha_i x_{i+1} - x) - \sinh(\alpha_i \Delta x_i)}{\sinh(\alpha_i \Delta x_i) - \alpha_i \Delta x_i}\right] \qquad (B.13)$$

By solving equation (B.13) at $x = x_i$, and $x = x_{i+1}$, and equating the result to the true values of slope at these points, we get the following two equations:

$$A_{i,1} - A_{i,2} - A_{i,3} - v_i A_{i,4} = y'_i \qquad (B.14)$$

for $x = x_i$, and

$$A_{i,1} - A_{i,2} + v_i A_{i,3} + A_{i,4} = y'_{i+1} \qquad (B.15)$$

for $x = x_{i+1}$, where

$$v_i = \left[\frac{\Delta x_i \alpha_i \cosh(\alpha_i \Delta x_i) - \sinh(\alpha_i \Delta x_i)}{\sinh(\alpha_i \Delta x_i) - \alpha_i \Delta x_i}\right]$$

Equations (B.11), (B.14), (B.12), and (B.15) can be put into the following matrix form:

$$\begin{bmatrix} 0 & \Delta x_i & 0 & 0 \\ 1 & -1 & -1 & -v_i \\ \Delta x_i & 0 & 0 & 0 \\ 1 & -1 & v_i & 1 \end{bmatrix} \begin{bmatrix} A_{i,1} \\ A_{i,2} \\ A_{i,3} \\ A_{i,4} \end{bmatrix} = \begin{bmatrix} y_i \\ y'_i \\ y_{i+1} \\ y'_{i+1} \end{bmatrix} \qquad (B.16)$$

for $x_i \le x \le x_{i+1}$.

This matrix system can then be used to solve for the spline coefficients $A_{i,k}$, k = 1, 2, 3, 4, for each subinterval i, by expressing it in the form

$$A_{i,k} = C(\Delta x_i, v_i)K_i \tag{B.17}$$

where

$$A_{i,k} = \begin{vmatrix} A_{i,1} \\ A_{i,2} \\ A_{i,3} \\ A_{i,4} \end{vmatrix} \tag{B.18}$$

$$C(\Delta x_i, v_i) = \begin{bmatrix} 0 & 0 & \frac{1}{\Delta x_i} & 0 \\ \frac{1}{\Delta x_i} & 0 & 0 & 0 \\ -\frac{1}{\Delta x_i}\frac{1}{1-v_i} & \frac{-1}{1-v_i^2} & \frac{1}{\Delta x_i}\frac{1}{1-v_i} & \frac{-v_i}{1-v_i^2} \\ \frac{1}{\Delta x_i}\frac{1}{1-v_i} & \frac{v_i}{1-v_i^2} & -\frac{1}{\Delta x_i}\frac{1}{1-v_i} & \frac{1}{1-v_i^2} \end{bmatrix} \tag{B.19}$$

and

$$K_i = \begin{bmatrix} y_i \\ y'_i \\ y_{i+1} \\ y'_{i+1} \end{bmatrix} \tag{B.20}$$

Equation (B.17) is actually equation (2.5).

REFERENCE

1. Pruess, S., "Properties of Splines in Tension," Journal of Approximating Theory, Vol. 17, pp. 86-96, 1976.
2. SpäTh, H., "Exponential Spline Interpolation," Computing, Vol. 4, pp. 225-233, 1969.

APPENDIX C

DERIVATION OF EQUATION (2.11)

To derive equation (2.11), the curvature continuity condition will be used at each node i, $2 \leq i \leq n-1$, such that the unknown quantities y'_i can be determined. The second derivative of g_i for subdomain i is obtained by differentiating equation (B.4) twice with respect to x:

$$g_i''(x) = A_{i,3}\left[\frac{\Delta x_i \alpha_i^2 \sinh(\alpha_i(x - x_i))}{\sinh(\alpha_i \Delta x_i) - \alpha_i \Delta x_i}\right] + A_{i,4}\left[\frac{\Delta x_i \alpha_i^2 \sinh(\alpha_i(x_{i+1} - x))}{\sinh(\alpha_i \Delta x_i) - \alpha_i \Delta x_i}\right] \qquad (C.1)$$

For subdomain i-1, g''_{i-1} is

$$g''_{i-1}(x) = A_{i-1,3}\left[\frac{\Delta x_{i-1} \alpha_{i-1}^2 \sinh(\alpha_{i-1}(x - x_{i-1}))}{\sinh(\alpha_{i-1} \Delta x_{i-1}) - \alpha_{i-1} \Delta x_{i-1}}\right] + A_{i-1,4}\left[\frac{\Delta x_{i-1} \alpha_{i-1}^2 \sinh(\alpha_{i-1}(x_i - x))}{\sinh(\alpha_{i-1} \Delta x_{i-1}) - \alpha_{i-1} \Delta x_{i-1}}\right] \qquad (C.2)$$

Solving these second derivatives at $x = x_i$ we get

$$g''_i(x_i) = A_{i,4}\left[\frac{\Delta x_i \alpha_i^2 \sinh(\alpha_i \Delta x_i)}{\sinh(\alpha_i \Delta x_i) - \alpha_i \Delta x_i}\right] \qquad (C.3)$$

and

$$g''_{i-1}(x_i) = A_{i-1,3}\left[\frac{\Delta x_{i-1} \alpha_{i-1}^2 \sinh(\alpha_{i-1} \Delta x_{i-1})}{\sinh(\alpha_{i-1} \Delta x_{i-1}) - \alpha_{i-1} \Delta x_{i-1}}\right] \qquad (C.4)$$

Equating (C.3) and (C.4) yields

$$A_{i,4}\left[\frac{\Delta x_i \alpha_i^2 \sinh(\alpha_i \Delta x_i)}{\sinh(\alpha_i \Delta x_i) - \alpha_i \Delta x_i}\right] = A_{i-1,3}\left[\frac{\Delta x_{i-1} \alpha_{i-1}^2 \sinh(\alpha_{i-1} \Delta x_{i-1})}{\sinh(\alpha_{i-1} \Delta x_{i-1}) - \alpha_{i-1} \Delta x_{i-1}}\right] \qquad (C.5)$$

where $A_{i,4}$ and $A_{i-1,3}$ are obtained from solving system (B.17), and are represented as

$$A_{i,4} = \frac{-v_i y_i' - y_{i+1}' + (1+v_i)\frac{\Delta y_i}{\Delta x_i}}{(v_i^2 - 1)} \qquad (C.6)$$

and

$$A_{i-1,3} = \frac{y_{i-1}' + v_{i-1} y_i' - (1+v_{i-1})\frac{\Delta y_{i-1}}{\Delta x_{i-1}}}{(v_{i-1}^2 - 1)} \qquad (C.7)$$

After substituting (C.6) and (C.7) into (C.5) and rearranging, we get (2.11), which takes the form

$$t_{i-1} y'_{i-1} + (t_{i-1} v_{i-1} + t_i v_i) y'_i + t_i y'_{i+1} =$$

$$t_{i-1}(1+v_{i-1})\frac{\Delta y_{i-1}}{\Delta x_{i-1}} + t_i(1+v_i)\frac{\Delta y_i}{\Delta x_i} \qquad (C.8)$$

where

$$t_i = \left[\frac{\Delta x_i \alpha_i^2 \sinh(\alpha_i \Delta x_i)}{(v_i^2 - 1)(\sinh(\alpha_i \Delta x_i) - \alpha_i \Delta x_i)}\right] \qquad (C.9)$$

Document Control Page	1. SERI Report No. NREL/TP-257-4099	2. NTIS Accession No. DE92010574	3. Recipient's Accession No.

4. Title and Subtitle A Surface Definition Code for Turbine Blade Surfaces	**5. Publication Date** May 1992 **6.**
7. Author(s) S. L. Yang, D. Oryang, M. J. Ho	**8. Performing Organization Rept. No.**
9. Performing Organization Name and Address Mechanical Engineering Department Michigan Technological University Houghton, Michigan 49931	**10. Project/Task/Work Unit No.** WE21.8104 **11. Contract (C) or Grant (G) No.** (C) XX-7-06092-2 (G)
12. Sponsoring Organization Name and Address National Renewable Energy Laboratory 1617 Cole Blvd. Golden, Colorado 80401-3393	**13. Type of Report & Period Covered** Technical Report **14.**

15. Supplementary Notes
NREL Technical Monitor: Peter K.C. Tu, (303) 231-7000

16. Abstract (Limit: 200 words)
A numerical interpolation scheme has been developed for generating the three-dimensional geometry of wind turbine blades. The numerical scheme consists of (1) creating the frame of the blade through the input of two or more airfoils at some specific spanwise stations and then scaling and twisting them according to the prescribed distributions of chord, thickness, and twist along the span of the blade; (2) transforming the physical coordinates of the blade frame into a computational domain that complies with the interpolation requirements; and finally (3) applying the bi-tension spline interpolation method, in the computational domain, to determine the coordinates of any point on the blade surface. Detailed descriptions of the overall approach to and philosophy of the code development are given along with the operation of the code. To show the usefulness of the bi-tension spline interpolation code developed, two examples are given, namely CARTER and MICON blade surface generation. Numerical results are presented in both graphic and tabular data forms. The solutions obtained in this work show that the computer code developed can be a powerful tool for generating the surface coordinates for any three-dimensional blade.

17. Document Analysis
a. Descriptors
wind turbines, blade surfaces, surface definition code

b. Identifiers/Open-Ended Terms

c. UC Categories
261

18. Availability Statement National Technical Information Service U.S. Department of Commerce 5285 Port Royal Road Springfield, VA 22161	**19. No. of Pages** 207 **20. Price** A10

Form No. 0069E (6-30-87)