SERI/STR--217-3476

DE89 000888

Yaw Dynamics of Horizontal Axis Wind Turbines

Second Annual Report

A Subcontract Report

A. Hansen
C. Xudong
Department of Mechanical Engineering
University of Utah
Salt Lake City, Utah

March 1989

SERI Technical Monitor: A. Wright

Prepared under Subcontract No. XL-6-05078-2

Solar Energy Research Institute
A Division of Midwest Research Institute

1617 Cole Boulevard
Golden, Colorado 80401-3393

Prepared for the
U.S. Department of Energy
Contract No. DE-AC02-83CH10093

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.

SUMMARY

Objectives

The proposal for this work stated the following objectives:

> Short term (first year): *Provide a fundamental understanding of free-yaw mechanics and the design concepts most effective at eliminating yaw problems.*
>
> Longer term (three year): *Provide tested design tools and guidelines for use by free-yaw wind system manufacturers.*

Events of the past few years have reinforced the need for meeting these objectives. Wind turbines continue to experience a destructive variety of yaw problems. Excessive yaw rates are causing structural damage. Yaw tracking errors (often 180°) are causing structural damage and/or loss of energy production. And automatic yaw control system failures continue to be a source of system downtime and maintenance expense. Many of these yaw-related problems have resulted from our lack of understanding of the yaw mechanism and inability to accurately estimate yaw loads and motions during the design or product improvement processes.

Discussion

The first year of research under this contract concentrated on development of a method and model to assist the prediction and understanding of yaw dynamics and aerodynamics. The second year, the subject of this report, has emphasized the validation of the methods. A determined effort has been made to understand the weaknesses and strengths of the model, with the goal of improving the methods and making available a truly practical and useful yaw design tool.

The validation effort has proceeded along three parallel paths. First, the three computer codes were compared one with another as a test of internal consistency and a simple method for preliminary de-bugging of the programs. Second, predictions of each of the codes were compared with the estimates of blade flap and yaw moments from the linearized, small perturbation model of Stoddard. This verified the governing equations and solution methods, at least for small, linear disturbances. Third and finally, the estimates from the models have been compared with test data. Wind tunnel tests of a rigid two-blade rotor (non-teetering Mod-2) and full-scale field tests of a rigid three-blade rotor (Howden 330 kW) have been used. Of course, these comparisons provide the most rigorous and compelling tests of the models. Unfortunately the data sets are limited in scope and subject to some known and some suspected inaccuracies. New data analysis methods developed and programmed under this contract have been useful in better understanding the test data, but the need for additional data remains.

Conclusions

A simple and direct statement of the progress to date would be that the predictions are more accurate than expected and less accurate than desired. Also, we now have a much clearer understanding of yaw mechanisms. The models are implemented in a practical form for use on a desktop computer and have been quite successful in estimating yaw and flap loads and motions for rigid rotors. Many uncertainties remain and virtually no validation has been attempted for soft rotors.

The three computer programs have been shown to be self-consistent and in excellent agreement with the estimates of Stoddard (in situations where such agreement is expected, i.e. small motions and linear range angles of attack). This implies the equations of motion and the programs are correctly implemented. However, it by no means validates the methods for the cases of most interest--high wind speed operation with large deflections.

Comparisons with test data have been satisfactory in many areas. It has been shown that corrections to the blade element aerodynamics are necessary to account for skewed wake effects. Without such corrections the estimates of yaw moments can have significant errors. Uncertainty remains as to the most accurate form of correction though the form used in the current models is consistent with helicopter industry practice and appears to be adequate.

The model estimates show correct trends in all comparisons with test data though the magnitudes of predicted yaw moments sometimes err as much as 100%.

Future work

Work for the remainder of the contract period will continue to concentrate on validation of the models. Data from the SERI Comprehensive Experiment will be used as it becomes available for the most complete and accurate comparison series to date. Refinement and modification of the models will be performed as required. Teetering will be added to the menu of rotor configurations as time and industry interest permit.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

A = area

a = axial induction factor, $\frac{V_i}{V_z}$

a_f = dry friction moment, $\mu F_b R_b$

a_v = viscosity coefficient of yaw damping system

B = number of rotor blades

C_L = section lift coefficient of blade airfoil

C_D = drag coefficient of blade airfoil

$C_{m\gamma}$ = yaw moment coefficient, $\frac{M_\gamma}{\frac{1}{2}\rho V_\infty^2 \pi R^3}$

c = blade chord length

g = acceleration due to gravity, 32.174 ft/sec^2

h = step size, in units of time, used to obtain numerical solution

I_b = moment of inertia of blade about its flap axis

I_L = moment of inertia of blade about its lag axis

I_θ = moment of inertia of blade about its pitch axis

I_n, I_s, I_y = moment of inertia of nacelle, rotor shaft and rotor hub about yaw axis

I_{yaw} = total yaw moment of inertia less blade contribution, $I_n + I_s + I_y$

K_β = blade hinge spring constant

L_n = nacelle length

L_s = distance from rotor hub to yaw axis, approximately equivalent to the shaft length

$M_{d\gamma}$ = damping moment on yaw column

M_{flap} = flapping moment on blade due to aerodynamic forces

M_n = yaw contribution from aerodynamic forces on nacelle

M_{yaw} = yaw moment contribution from aerodynamic forces on blade

M_γ = total sum of all moments about yaw axis, $\sum_{i=1}^{B} M_{yaw_i} + M_n - M_{d\gamma}$

m_b = blade mass

R = rotor radius

R_H = rotor hub radius

R_n = nacelle radius

t = time

V_∞ = free stream wind velocity

V_i = induced velocity due to air flow over blade

V_n = normal component of relative velocity to blade

V_t = tangential component of relative velocity to blade

$\mathbf{x}, \mathbf{y}, \mathbf{z}$ = inertial reference frame (positioned about tower top)

X′, Y′, Z′ = yawed reference system (positioned about rotor center)

$\hat{X}, \hat{Y}, \hat{Z}$ = spinning reference system (about rotor center)

x, y, z = coned reference system (positioned about blade hinge)

x_c = distance from nacelle nose to centroid of nacelle body planform area

x_n = distance from nacelle nose to yaw axis

α = angle of attack

α_n = nacelle angle of attack, $\delta + \gamma$

β = flapping angle of blade

β_0 = preconing angle of blade

δ = wind direction angle, rotor disk sector angle, blade tip deflection

γ = yaw angle

ψ = blade azimuth angle, relative to vertical downward

ψ_0 = half sector angle of tower shadow region

Ω = blade rotation rate

μ = Coulomb friction coefficient

θ = blade pitch angle, measured from chord line to plane of rotation, positive upwind

ρ = air density, blade material density

ϕ = inflow angle of relative velocity vector, $\tan^{-1}\left(\frac{V_n}{V_t}\right)$

ω_h = natural flapping frequency of hinged blade, $\sqrt{\frac{K_\beta}{I_b}}$

ω_Ω = dimensionless rotating blade frequency, $\sqrt{\omega_n^2 + 1}$

Special Symbols

($\dot{}$) = derivative with respect to time

( )′ = derivative with respect to azimuth angle ψ

Section 1.0
INTRODUCTION

1.1 Background

The University of Utah has recently completed the second year of a planned three-year investigation of the free-yaw and controlled-yaw behavior of horizontal-axis wind turbines. This report presents the progress and results of that second year of effort.

The proposal for this work stated the following objectives:

> Short term (first year): *Provide a fundamental understanding of free-yaw mechanics and the design concepts most effective at eliminating yaw problems.*
>
> Longer term (three year): *Provide tested design tools and guidelines for use by free-yaw wind system manufacturers.*

During the past two years yaw problems have continued to plague wind turbines and reinforce the need for meeting these objectives. Hundreds of machines in wind farms in California are shut down (at this writing) by yaw problems. One free-yaw system has suffered many low-speed shaft fatigue cracks caused by repeated and excessive yaw rates. And controlled-yaw systems continue to experience structural failures in the yaw drives as a result of higher than anticipated yaw moments on the rotors. Many of these yaw-related problems have resulted from our lack of understanding of the yaw mechanism and inability to accurately estimate yaw loads and motions during the design or product improvement processes. Fortunately, some of the results of the research at the University of Utah are beginning to assist engineers with understanding the causes and possible cures of some of these yaw dynamics problems.

During the first year of the research the emphasis was upon developing computer models for yaw dynamics and programs for dynamic data analysis. The results of that effort were three computer programs for dynamics predictions (called UFlap, SDOF and YawDyn) and a family of programs for data analysis. The program called YawDyn is a complete simulation of a wind turbine rotor with yaw and blade flap degrees of freedom. It analyzes the loads and motions of a rotor with an arbitrary number of cantilevered blades under the influence of various wind input profiles, including vertical and horizontal wind shear and tower shadow. It models blade stall using 2-D airfoil tables and includes corrections for skewed wake aerodynamics. Program SDOF is the YawDyn program modified to analyze only the yaw degree of freedom. It is applicable to a very rigid rotor and executes much faster than the YawDyn program. Program UFlap is again the YawDyn program modified to operate with specified yaw motion and a blade flap degree of freedom only. It was used primarily to test the subroutines and methods used in YawDyn. Programs UFlap and SDOF were tested and modified extensively during the first year. Program YawDyn was written but not extensively exercised or tested. Complete details are available in the first annual progress report (Cui, Hansen and Siedschlag, 1988).

These programs have been placed into the public domain and are being used by laboratories and turbine manufacturers. Use of these programs has made it possible to largely satisfy the first objective stated above. We now know that rigid rotors can experience unusual or damaging yaw response primarily as a result of horizontal wind shear. Such wind shear is likely in windfarms and even in smooth, flat terrain in the presence of large-scale atmospheric turbulence. We also know that use of a "soft" rotor or addition of mechanical yaw damping can make yaw response much more benign.

Comparisons of wind-tunnel test results and SDOF predictions and sensitivity studies in the first year of effort lead to the following key conclusions for a rigid rotor:

> Horizontal wind shear, such as might be encountered in complex terrain or turbine arrays, can cause significant and persistent yaw errors. Use of a yaw drive, a soft rotor or re-siting the turbine may be the only cures for this problem.
>
> Effects of the skewed wake must be considered when calculating aerodynamic yaw moments on a yawed rotor. The method used in SDOF appears to be adequate. The need for induced velocity correction (from simple blade element/momentum methods) is clear. The precise method for correction is not verified at this time.
>
> Yaw moments are cyclic in nature. Peak yaw moments will greatly exceed average yaw moments during normal operation. Peak-to-peak fluctuations in the yaw moment which are 20 to 50 times the mean value have been observed in some situations.
>
> Yaw moments result primarily from blade root flap *moments*. The blade root *forces* play a lesser role in determining yaw moments. As a result, the length of the moment arm from the yaw axis to the hub has little influence on the yaw loads. Attempts to improve yaw behavior by lengthening the low-speed shaft will not be likely to succeed. (This comment applies to an operating rotor. The length of the moment arm will have a strong influence on the yaw-tracking behavior of a stopped rotor in light winds. Obviously the system must be designed so that it will align with the wind before startup of the rotor.)
>
> Any imbalance in the blades (such as small differences in pitch angle) can result in substantial 1p yaw moments. If 1p yaw moments are consistently observed on an operating machine it is likely that the blades are not well matched.

Several items of additional work were identified in the first annual report. Foremost among the needs was evaluation of the analysis methods through comparison with field test results. This is a formidable task because atmospheric test data is always difficult to analyze and often incomplete, lacking either desirable measurement channels or desirable test conditions. Data analysis software was developed and validated in the first year of effort to assist with this task. Much of the activity of the second year of effort has focused on a data set taken from tests of a Howden 330 kW system in San Gorgonio Pass, California. A parallel effort has concentrated on

comparing predictions made using YawDyn, SDOF and the analytical solutions of Stoddard (1988). A side benefit of this activity has been extensive progress in making the codes more understandable, efficient, and accurate.

1.2 Approach

Work in the second year of the SERI contract has emphasized testing of SDOF against a variety of field test results and other prediction methods. YawDyn has also been modified and tested, though not to the same degree of completion that SDOF has enjoyed. Comparisons between SDOF predictions and field test results from a Howden 330 kW turbine are the major result that will be presented in this report. One result of the extensive testing of the computer codes has been a number of changes in the codes. While no changes have been made in the basic approach or the governing equations, a number of changes have been made to fix "bugs", improve efficiency or simplify input data preparation or operation of the codes.

The discussion that follows is an extension of the material from the first annual report (Cui, Hansen and Siedschlag, 1988). Thus, many details concerning the basic methods used and the background and rational for the technical approach will not be repeated in this report. Rather, it is assumed the reader is familiar with the first annual report.

In the next section the methods for dynamics predictions and data analysis will be briefly described. Emphasis will be placed upon material which was not included in earlier reports. Section 3 will present results from three different validation efforts. The first validation is a comparison of YawDyn and SDOF results, for cases where the two methods should give similar results (a very stiff rotor) and for cases where the predicted results should differ, i.e. a rotor with soft (1p-2p) blades. The second validation is comparisons of YawDyn results with predictions using the simple linear model of Stoddard. The third is comparison of SDOF predictions with field test data from a Howden 330 kW system in San Gorgonio Pass. Section 4 will summarize conclusions that can be drawn from the second year of effort and Section 5 will briefly discuss future plans. Appendices will provide complete listings of the programs used and a User's Guide for the data analysis software system.

Section 2.0
DYNAMICS PREDICTIONS AND DATA ANALYSIS METHODS

2.1 Predictions of Yaw Dynamics

Modeling of yaw dynamics is made complex by the variety of machine and wind characteristics which are important to yaw response. The goal of the current modeling is to develop the simplest model possible while not neglecting significant effects. Previous research indicated the following features must be included in the model

- Blade root flexibility and flap motion
- Aerodynamic model must include stall effects
- Correction for effects of skewed wake aerodynamics on the induced velocities
- Vertical and horizontal wind shear and tower shadow
- Mechanical yaw damping and friction
- Arbitrarily large yaw angles but small flap angles
- Both free- and fixed-yaw behavior

These effects and features are implemented in the current models. The result is a set of equations for yaw and flap motion in the time domain. Simple blade element/momentum aerodynamics are used but corrections are made that account for the interdependence of induced velocities through various elements when the wake is skewed with respect to the rotor. The contributions to the yaw moment of all blade flap and lead-lag forces and moments are included in the calculations. Thus the "H" force of helicopter terminology is included (though the lead-lag vibratory degree of freedom is not considered). Two-dimensional airfoil tables are used to represent the blade lift and drag coefficients and the NASA synthesization method is used to obtain airfoil characteristics in deep stall.

Some details of the model are presented in the paragraphs that follow. Much of this material is excerpted from the first annual progress report and the reader is referred to that report for more details (Cui, Hansen and Siedschlag, 1988).

HAWT Representation

A simplified model for a two blade HAWT is shown schematically in Figure 2.1. Only yaw motion, γ, and blade flapping motion, β, have been used in the development of the equations of motion. A teetering degree-of-freedom has not been included in this model. Additional degrees-of-freedom, such as blade pitch and lag motion, are not considered to be as important to yaw response and have been ignored.

A rigid tower (no top deflection or rotation) is used in the model. This permits the inertial reference frame **xyz** to be located with its origin (point o in Figure 2.2) at the intersection of the yaw column and the rotor shaft. X′ Y′ Z′, $\hat{X}\,\hat{Y}\,\hat{Z}$, and x y z designate the yawing, spinning and coning reference frames used in developing the equations of motion. The distance from yaw axis to the rotor, the approximate shaft length, is designated by L_s.

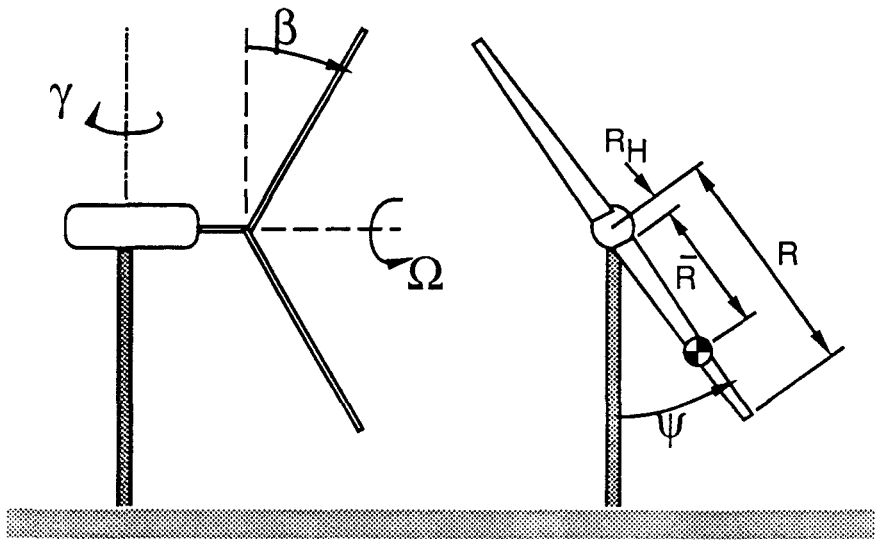

Figure 2.1 Schematic of the rotor showing the primary blade variables.

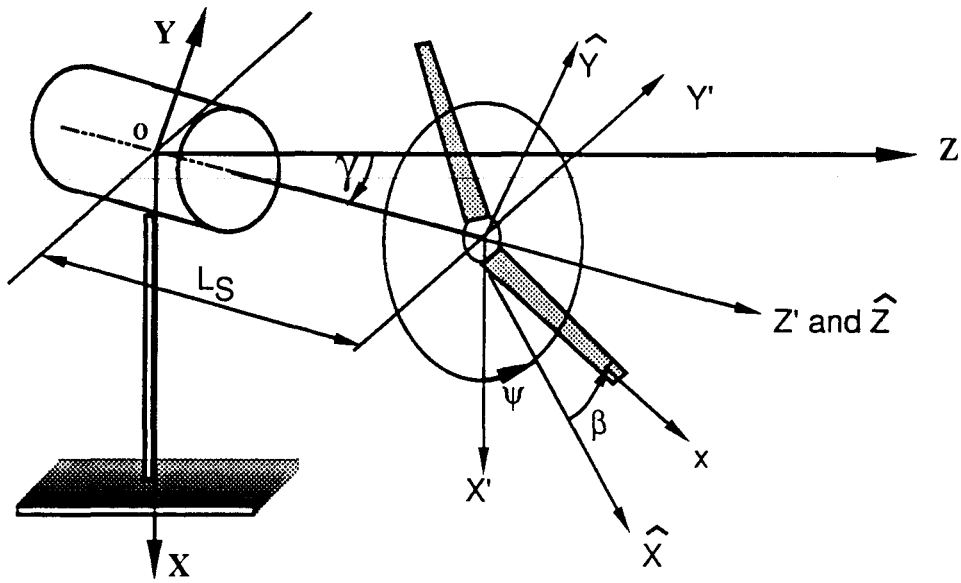

Figure 2.2 HAWT coordinate system

The rotor spin rate, Ω, is considered to be constant for each specified operating condition. The first blade azimuth position, ψ, is defined with respect to the six o'clock position of the rotor disk. This azimuth angle will be used to describe the positions of all the rotor blades.

The nacelle, rotor shaft and rotor blades are treated as rigid bodies. The blades are connected to the rotor hub by frictionless hinges which permit only out-of-plane flapping motion. Torsional springs with stiffness K_β are attached as shown in Figure 2.3. This models the elastic deflections of each wind turbine blade.

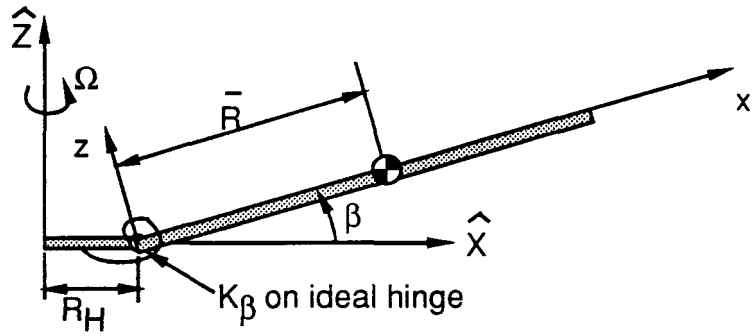

Figure 2.3 Hinged blade with torsional spring

The blade flapping is a function of the instantaneous wind velocity as seen by the blade as it rotates about the shaft. This will cause the i^{th} blade to assume its own motion, described by β_i, independent of each other blade. This gives the model B + 1 degrees of freedom where B is the number of blades.

The governing equations are derived using Lagrange's Equation. The resulting equations for the flap motion are:

$$\beta_i'' + \left[\frac{I_L - I_\theta}{I_b} + \frac{m_b\bar{R}R_H}{I_b} + \frac{K_\beta}{I_b\Omega^2} + \frac{m_b g\bar{R}}{I_b\Omega^2}\cos\psi_i\right]\beta_i + (\gamma')^2\left\{\left[\frac{m_b\bar{R}R_H}{I_b}\sin^2\psi_i - \frac{I_L - I_\theta}{I_b}\cos^2\psi_i\right]\beta_i - \frac{m_b\bar{R}L_s}{I_b}\right\} + \gamma'\left[1 + \frac{I_L - I_\theta}{I_b} + \frac{2m_b\bar{R}R_H}{I_b}\right]\cos\psi_i + \gamma''\left(1 + \frac{m_b\bar{R}R_H}{I_b} + \frac{m_b\bar{R}L_s}{I_b}\beta_i\right)\sin\psi_i - \frac{K_\beta}{I_b\Omega^2}\beta_0 = \frac{M_{flap,i}}{I_b\Omega^2} \qquad (2.1)$$

and for yaw angle is

$$\left[I_{yaw} + B(m_b L_s^2 + I_\theta) + (I_b - I_\theta + m_b R_H^2 + 2m_b \bar{R} R_H)\sum_{i=1}^{B}\sin^2\psi_i + I_L\sum_{i=1}^{B}\beta_i^2\cos^2\psi_i + \right.$$

$$2m_b\bar{R}L_s\sum_{i=1}^{B}\beta_i\Big]\gamma'' + 2\gamma'\Big[(I_L - I_\theta)\sum_{i=1}^{B}\beta_i'\beta_i\cos^2\psi_i - I_L\sum_{i=1}^{B}\beta_i^2\sin\psi_i\cos\psi_i + m_b\bar{R}L_s\sum_{i=1}^{B}\beta_i' -$$

$$m_b\bar{R}R_H\sum_{i=1}^{B}\beta_i'\beta_i\sin^2\psi_i + \frac{a_v}{2\Omega}\Big] + (I_b - I_L + I_\theta)\sum_{i=1}^{B}\beta_i'\cos\psi_i + (I_L - I_\theta + m_b\bar{R}R_H)\sum_{i=1}^{B}\beta_i\sin\psi_i +$$

$$m_b\bar{R}(2L_s\sum_{i=1}^{B}\beta_i'\beta_i\cos\psi_i + L_s\sum_{i=1}^{B}\beta_i'^2\sin\psi_i + L_s\sum_{i=1}^{B}\beta_i''\beta_i\sin\psi_i - R_H\sum_{i=1}^{B}\beta_i'^2\beta_i\sin\psi_i +$$

$$(m_b\bar{R}R_H + I_b)\sum_{i=1}^{B}\beta_i''\sin\psi_i + \frac{a_f}{\Omega^2}\mathrm{sgn}(\gamma') = \sum_{i=1}^{B}\frac{M_{yaw_i}}{\Omega^2} + \frac{M_n}{\Omega^2} \qquad (2.2)$$

For a rigid rotor ($\beta_i' = \beta_i'' = 0$, $\beta_i = \beta_0$), the HAWT model possesses only a single degree-of-freedom with the equation of motion becoming

$$\left[I_{yaw} + Bm_b(L_s^2 + 2\bar{R}L_s\beta_0) + (I_b + m_b R_H^2 + 2m_b\bar{R}R_H)\sum_{i=1}^{B}\sin^2\psi_i + \right.$$

$$(I_\theta + I_L\beta_0^2)\sum_{i=1}^{B}\cos^2\psi_i\Big]\gamma'' + \frac{a_v}{\Omega}\gamma' + \frac{a_f}{\Omega^2}\mathrm{sgn}(\gamma') = \sum_{i=1}^{B}\frac{M_{yaw_i}}{\Omega^2} + \frac{M_n}{\Omega^2} \qquad (2.3)$$

Equations 2.1 and 2.2 are the complete set of equations implemented in the program YawDyn. The validation of this model will be the subject of future work.

Equation (2.3) is the governing equation solved in the simpler program SDOF. SDOF is used to test many of the same subroutines used in YawDyn. It also requires fewer computations to obtain a yaw response time-history. This allows it to serve as a preliminary aid to determining the yaw behavior of a HAWT before using YawDyn to perform more detailed analysis. It also permits testing of many aspects of the modelling on a simpler, more straightforward model. Results that will be shown in later sections of this report are primarily from model SDOF (equation 2.3). Other results will be from equations 2.1 with specified yaw motion (model UFlap).

Initial Conditions

All initial positions and velocities of the degrees-of-freedom must be prescribed to obtain a numerical solution. In the present work this is done in a rather arbitrary and expedient manner. Initial conditions for yaw and flap angles and rates are input by the program user. Any values can be input and often zero rates are used. Note however

that if initial conditions are specified which do not satisfy the homogeneous solution of the governing equations, then a transient solution will be generated which will initially have no physical meaning. For this reason it is generally necessary to disregard the first several rotor revolutions while the transients associated with startup of the solution damp out.

In future work this arbitrary selection of initial values, particularly of the flap angles and rates, will be replaced by a scheme which calculates some values. The "trim" solution will be calculated and used as the initial conditions. This will make it possible to generate meaningful solutions more quickly.

Fortran Programs

The computer programs YawDyn, SDOF and UFlap were written in Fortran 77 for use on the University of Utah Industrial and Mechanical Engineering Department VAX 11/750. Listings of SDOF and UFlap are contained in the Appendices. Program YawDyn computes the yaw error angle, yaw rate and yaw moment versus time for a two or three blade downwind HAWT operating in free-yaw with freely flapping blades. As an additional output, YawDyn provides the flapping deflection, flapping rate and flapping moment versus time for the B^{th} blade. SDOF uses many of the same subroutines as YawDyn to solve the single degree-of-freedom (rigid rotor) yaw model. It provides the same yaw outputs as YawDyn. UFlap uses some of the same subroutines as YawDyn to solve the flapping equation-of-motion for a single hinged blade operating under prescribed yaw conditions. It allows the effect of yaw on blade flap response to be analyzed without requiring the additional computations performed by YawDyn. The use of various yaw states with UFlap have indicated that the yaw angle and yaw rate do contribute significantly to the overall flap behavior but the yaw acceleration term in (2.1) does not and can be safely ignored in YawDyn.

2.2 Data Analysis Methods

The analysis of yaw and flap behavior data from atmospheric tests is complicated by the random fluctuations of the wind and the wide range of time scales of interest in the problem. A typical yaw moment will be composed of fluctuations at the rotor frequency of one cycle per revolution of the rotor (1p), and at higher frequencies including at least 2p and 3p. In addition there will be slow variations in yaw moment due to change in yaw angle, wind speed and wind shear. These changes can occur over periods ranging from fractions of a second to several minutes. Extraction of useful information from "noisy" test data is the purpose of a large software package developed in the first year of this effort and extensively applied in the second year.

The data analysis software is capable of performing two fundamental operations plus a number of minor utility operations. The fundamental operations are 1) Method-of-Bins using any data channels as the independent and dependent variables. 2) Harmonic analysis which creates a time-series of the harmonic content of a signal such as the yaw or flap moment. Utility operations include calculation of parameters such as disc-averaged wind speed, wind shear, yaw rate and yaw error; low-pass

filtering of any data channel; data transfer; graphics; and selective sampling of data files.

Appendix D contains a user's guide to the software package and listing of the programs. A recent paper by Hansen (1988) provides details on the harmonic analysis method. This report will contain many analysis results obtained using this software package.

Section 3.0
RESULTS FROM THE DYNAMICS AND DATA ANALYSES

This section will present a series of predictions from the yaw dynamics models and comparisons between predictions and field test results. In the previous annual report the predictions of program SDOF were compared with wind-tunnel tests and predictions using the linearized method of Stoddard. The predictions were generally favorable and demonstrated that the basic method is sound. Yaw moments were estimated accurately and flap loads and motions were in agreement with Stoddard's predictions when perturbations were small (hence when Stoddard's predictions were reliable).

In the next section some comparisons will be drawn between the predictions of SDOF and YawDyn. Cases with stiff blades (where the two methods should agree) and with soft blades (where the two methods should differ) will be shown. In section 3.2 YawDyn will be compared with predictions of Stoddard. In section 3.3 the most useful and rigorous test will be applied. Comparisons with test data from a full-scale Howden 330 kW system in San Gorgonio Pass will be shown. The data reported have all been analyzed using the dynamic data analysis software described in Section 2.

3.1 Comparisons of YawDyn and SDOF Results

A series of comparisons have been made to verify the operation of program YawDyn and to examine the effects of blade natural frequency on the yaw response of a rotor. An Enertech 44/60 wind turbine was used as the baseline system for the comparisons. The Enertech 44/60 is found in a number of California wind farms. It is a downwind, free-yaw, three-blade system with fixed-pitch stall control. Details of the rotor are shown in Table 3.1. The airfoil characteristics used in the calculations are shown in Table 3.2.

The first comparisons show SDOF and YawDyn predictions for the baseline (actual) Enertech machine. The programs calculate the free-yaw motion of the rotor after it is released from rest at an initial yaw angle of 30°. The blade is quite stiff so the blade flap motions should be very small and the two programs should predict similar results. Figure 3.1 shows that the predictions of the two programs are very similar for stiff blades. Three different wind speeds are used for the comparisons to show that the YawDyn and SDOF predictions are similar for unstalled, partially stalled and deeply stalled operation. Unfortunately, little data is available from the Enertechs but it is interesting to note that field reports from the windfarms indicate the systems occasionally oscillate in yaw with a period of approximately eight seconds (Johnson, 1987). This is the same period exhibited by the predicted yaw responses.

Table 3.1
Properties of the Enertech 44/60 Wind Turbine

Radius	22. ft
Number of blades	3
Rotor speed	67. rpm
Precone angle	6°
Hub height	82 ft
Equivalent root stiffness (k_β)	8.35×10^5 ft-lb/radian
Nacelle moment of inertia about yaw axis	500 slug-ft^2
Blade flap moment of inertia about root	1000. slug-ft^2
Blade mass	7.45 slugs
Blade center of gravity	r=8.0 ft
Shaft length (L_S)	4.25 ft
Pitch angle at tip of blade	3.5°

Twist and chord distribution:

r/R	Twist (deg)	Chord (ft)
0.05	5.50	2.0
0.15	4.89	1.99
0.25	4.28	1.96
0.35	3.67	1.91
0.45	3.06	1.86
0.55	2.44	1.82
0.65	1.83	1.77
0.75	1.22	1.73
0.85	0.61	1.70
0.95	0.00	1.67

Table 3.2
Airfoil Characteristics used in the Enertech 44/60 Modeling

Angle of Attack (deg)	C_L	C_D
-7.0	-.23	.0097
-4.0	.085	.0085
0.	.51	.00785
4.	.91	.00845
8.	1.21	.0115
10.	1.32	.0147
12.	1.37	.020
14.	1.37	.024

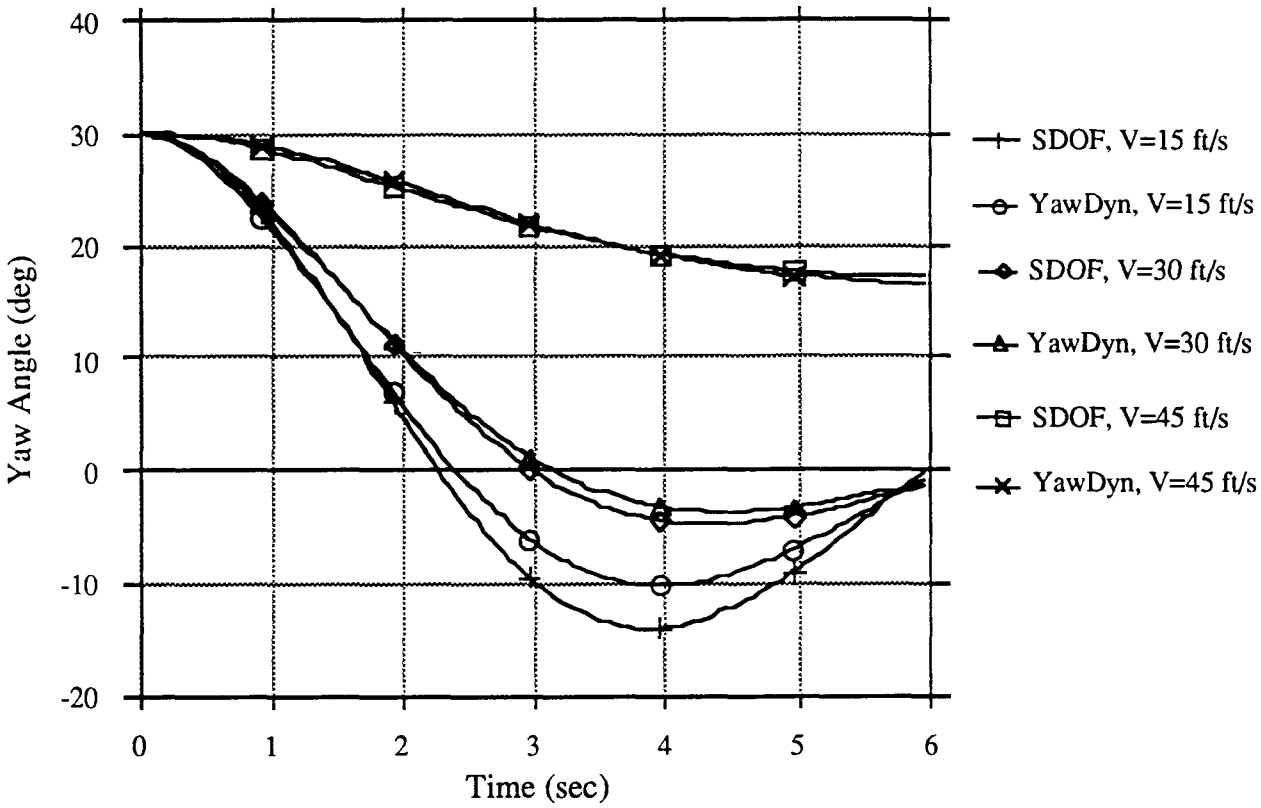

Figure 3.1 Comparison of YawDyn and SDOF Predictions, Baseline Enertech 44/60 at three different wind speeds.

Figure 3.2 shows the importance of the blade stiffness (expressed in terms of the natural frequency) to the yaw response. The low frequency blade exhibits high yaw rates and poor yaw tracking. Figure 3.3 shows similar predictions for the modified Enertech in higher winds. The predictions are run for a longer time to show the yaw behavior of the 1.14 p blade does not stabilize. (In fact, the yaw oscillations continue to grow as time continues.) To see if the yaw oscillations were due to inadequate damping, the Lock number of the blade was doubled by decreasing the mass and moment of inertia by 50%. Indeed, the yaw behavior of the "soft" blades is greatly improved by increasing the aerodynamic damping forces relative to the inertial forces. In Figures 3.2 and 3.3 the models used 1/7 power-law vertical wind shear and no horizontal wind shear. The blade root spring stiffnesses corresponding to the 4.3p, 1.14p and 1.06p frequencies were k_β=8.35x10^5, 8.35x10^3, and 8.35 ft-lb/rad, respectively.

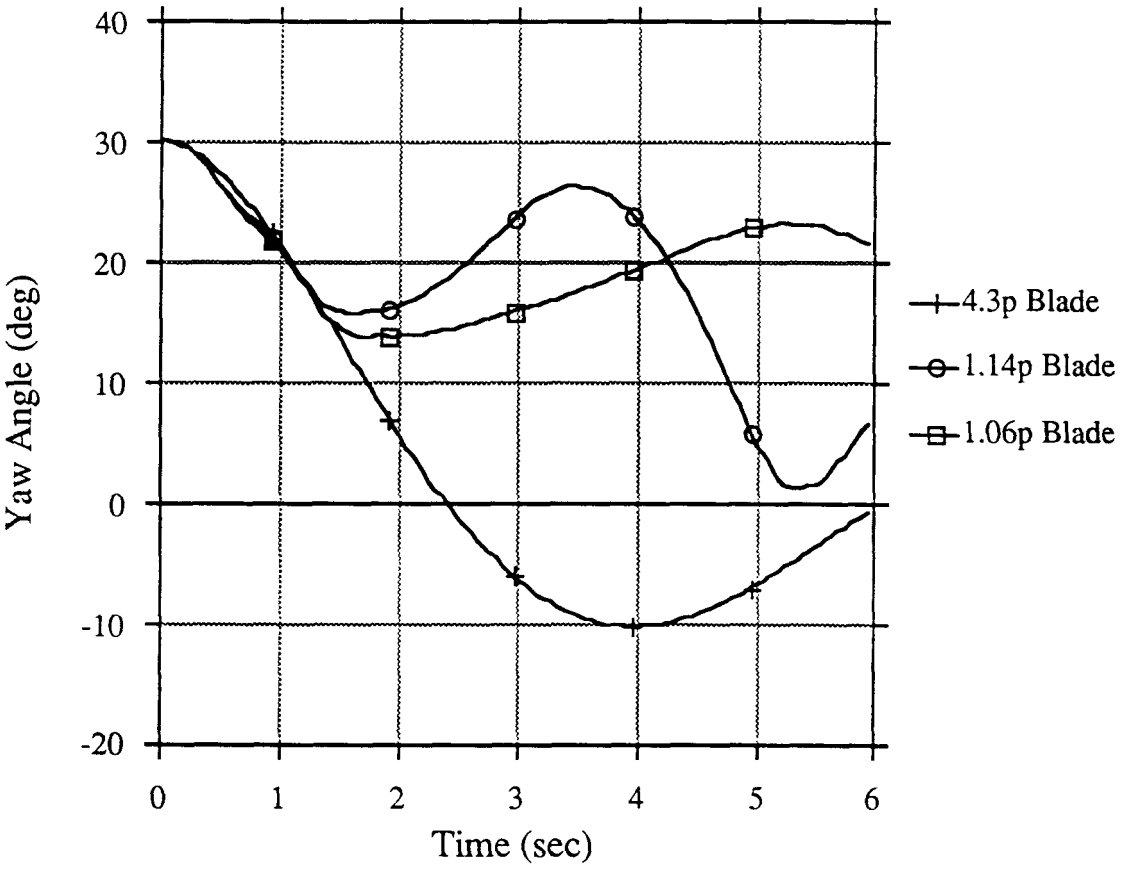

Figure 3.2 YawDyn Predictions showing the effect of blade stiffness on the predicted response. Modified Enertech 44/60 at V=15 ft/s

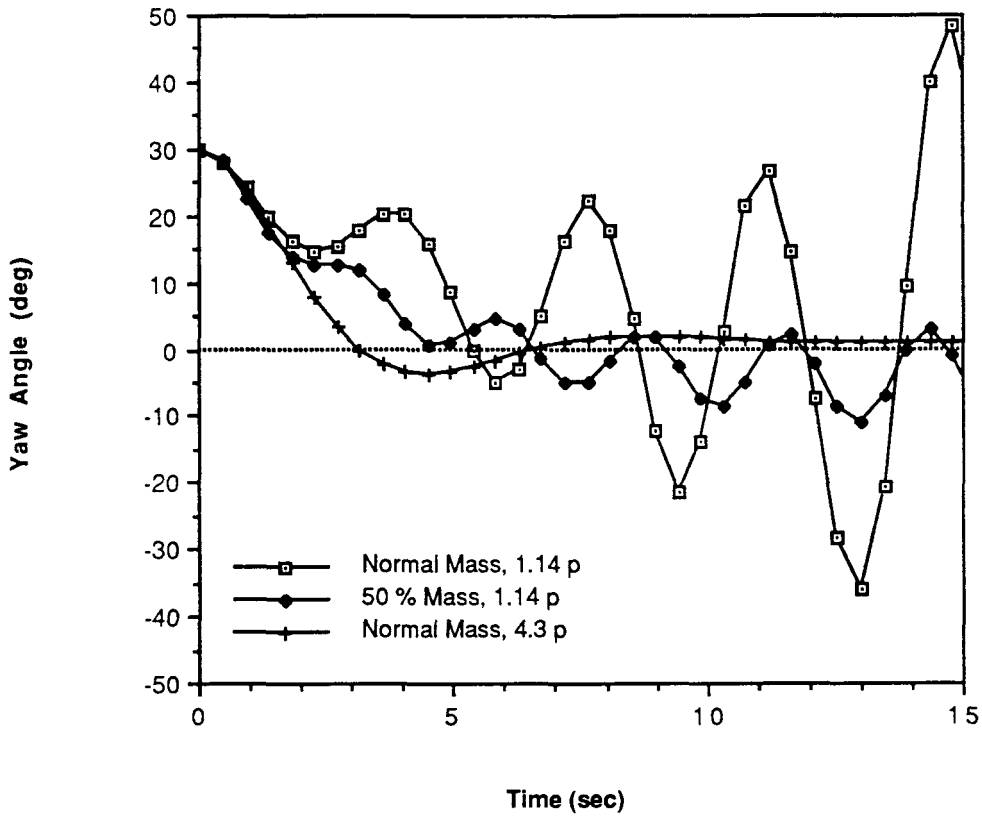

Figure 3.3 YawDyn Predictions showing the effect of blade stiffness and mass on the predicted response. Modified Enertech 44/60 at V=30 ft/s.

3.2 Comparisons of YawDyn, UFlap and Stoddard Results

The linear analysis of Stoddard (1978 and 1988) can be used to estimate flap motion and yaw moments for a rigid blade with an equivalent root hinge/spring. The Stoddard predictions differ from the University of Utah models only in their assumptions of linear aerodynamics, small deflections and no skewed wake effects. Thus the Stoddard predictions provide an excellent opportunity for testing the Utah models. Such tests have been performed using the Enertech 44/60 system model.

The first comparisons are of blade flap motions for the baseline Enertech 44/60. The characteristics of this rotor are given in Tables 3.1 and 3.2. Of particular note is the blade root stiffness, $k_\beta = 8.35 \times 10^5$ ft-lb/rad. With the hinge offset of 2 ft, this makes the rotating system frequency 4.25 cycles per revolution (a "4.25p" blade) The predictions of Stoddard and UFlap are shown in Figure 3.4. A wind speed of 22 ft/s was selected for this and subsequent comparisons to ensure low angles of attack along the outer sections of the blade (the angle is approximately 1° at the 75% span). Clearly the flap deflections are small and the two methods predict the same results. Linear horizontal

and vertical wind shear were applied to the rotor. Each shear results in a change of wind speed from tip to hub which is 13% of the free stream wind speed. The yaw angle and yaw rate are zero.

Figure 3.5 shows the effect of reducing the blade root stiffness one order of magnitude (k_β = 8.35x10^4 ft-lb/rad or a 1.68p blade). The deflections are much larger, as expected, and the two models predict similar though not identical results. Figure 3.6 shows the effect of reducing the blade root stiffness by one more order of magnitude (k_β = 8.35x10^3 ft-lb/rad or a 1.14p blade). In this case the predictions differ significantly, presumably due to the large deflections.

When the rotor is set at a fixed yaw angle the results are as shown in Figures 3.7 and 3.8. Both figures are for the 1.68p blade. The figures show two YawDyn predictions, one with and one without the skewed wake corrections. The figures exhibit differences in the phase, amplitude and mean flap angle among the three estimates. This is not surprising since the yaw angles are large. UFlap predictions are not shown because YawDyn and UFlap produce identical results when the programs are run for fixed yaw situations.

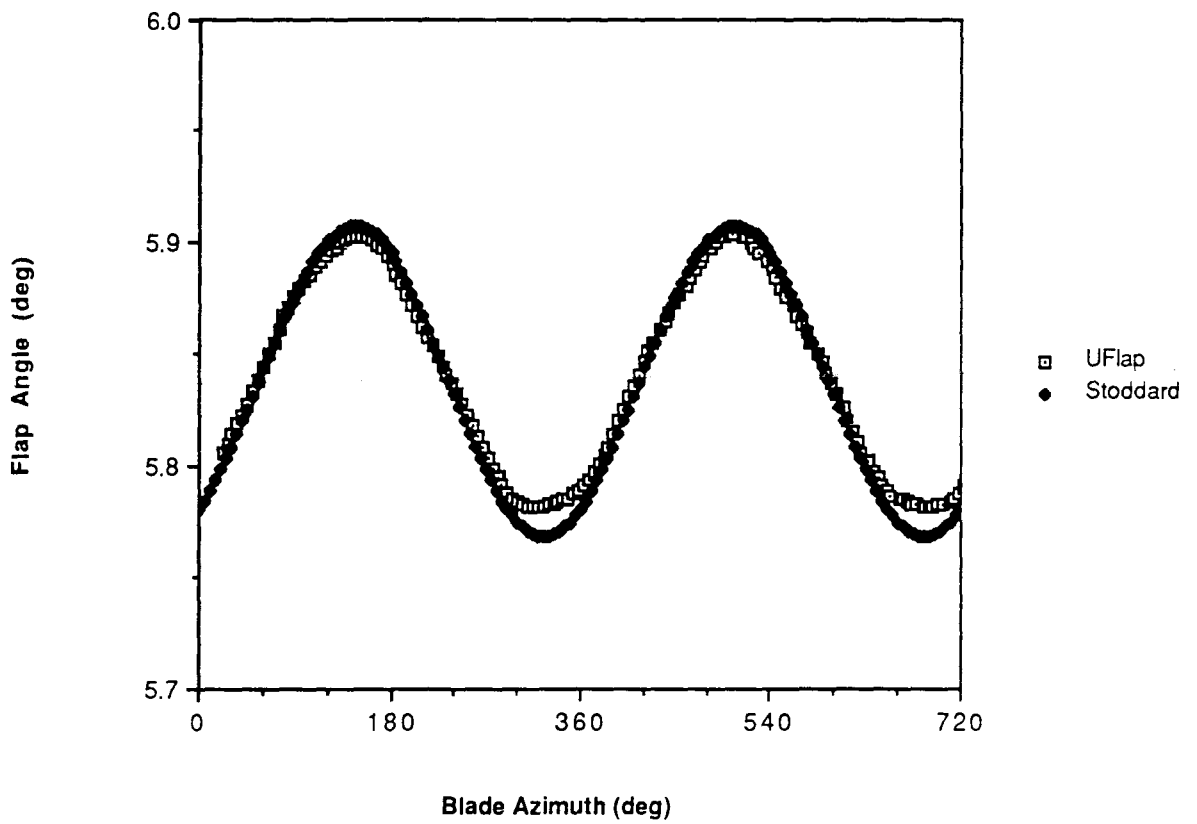

Figure 3.4 Comparison of UFlap and Stoddard Model Predictions for the Baseline Enertech 44/60 rotor (V=22 ft/sec, k_β=8.35x10^5 ft-lb/rad).

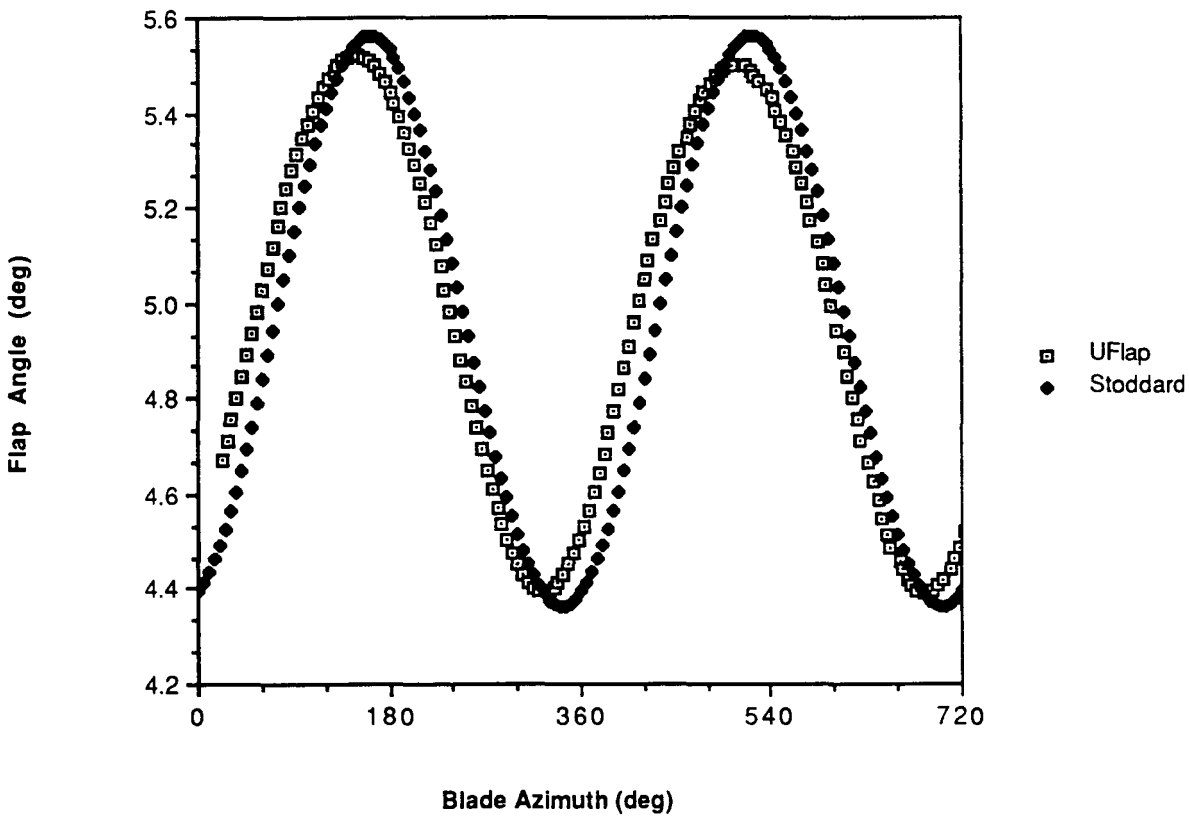

Figure 3.5 Comparison of UFlap and Stoddard Model Predictions for the Modified Enertech 44/60 rotor (V=22 ft/sec, k_β=8.35x10^4 ft-lb/rad).

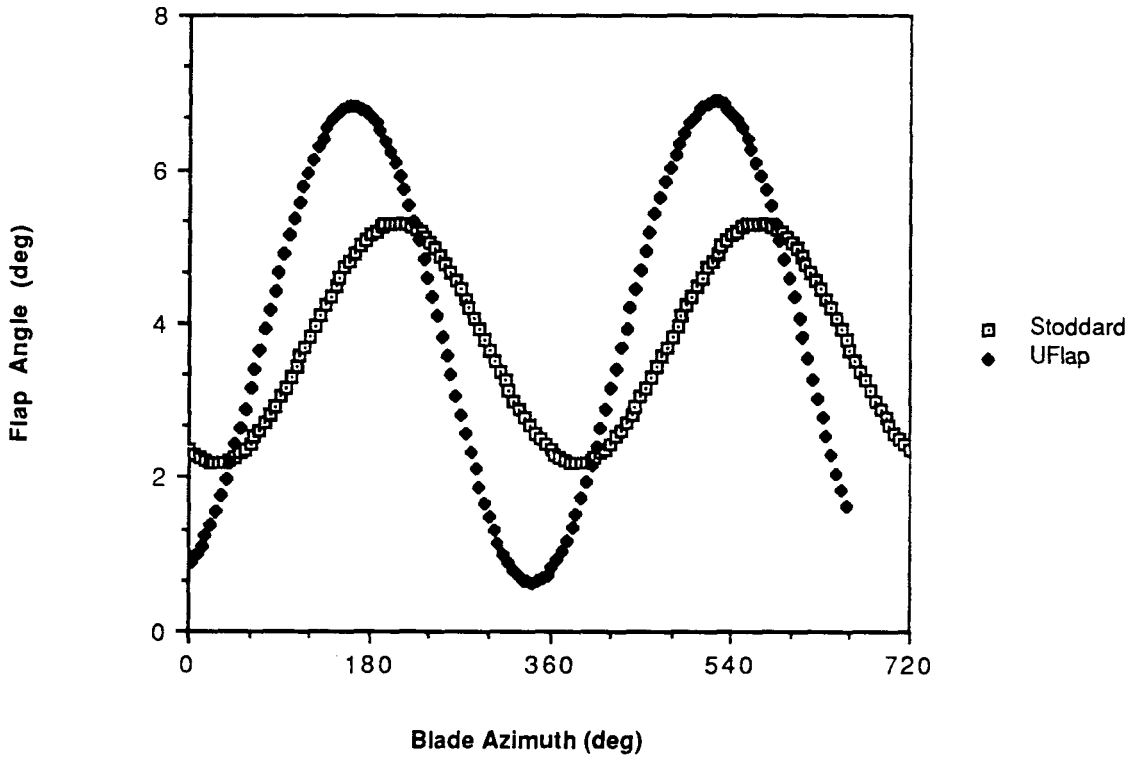

Figure 3.6 Comparison of UFlap and Stoddard Model Predictions for the Modified Enertech 44/60 rotor (V=22 ft/sec, k_β=8.35x10^3 ft-lb/rad).

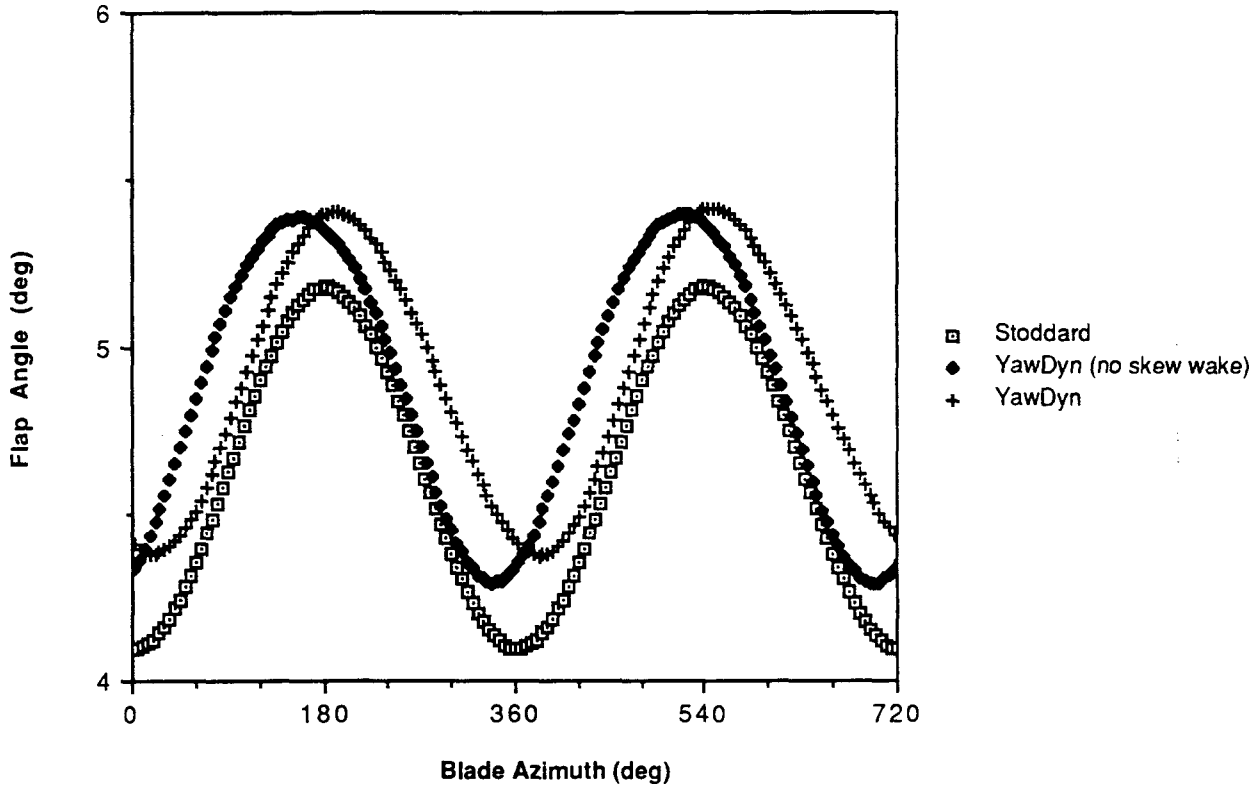

Figure 3.7 Comparison of Stoddard Predictions with YawDyn (both with and without the skewed wake correction). Yaw angle = 30°, Wind speed = 22 ft/sec
$k_\beta = 8.35 \times 10^4$ ft-lb/rad

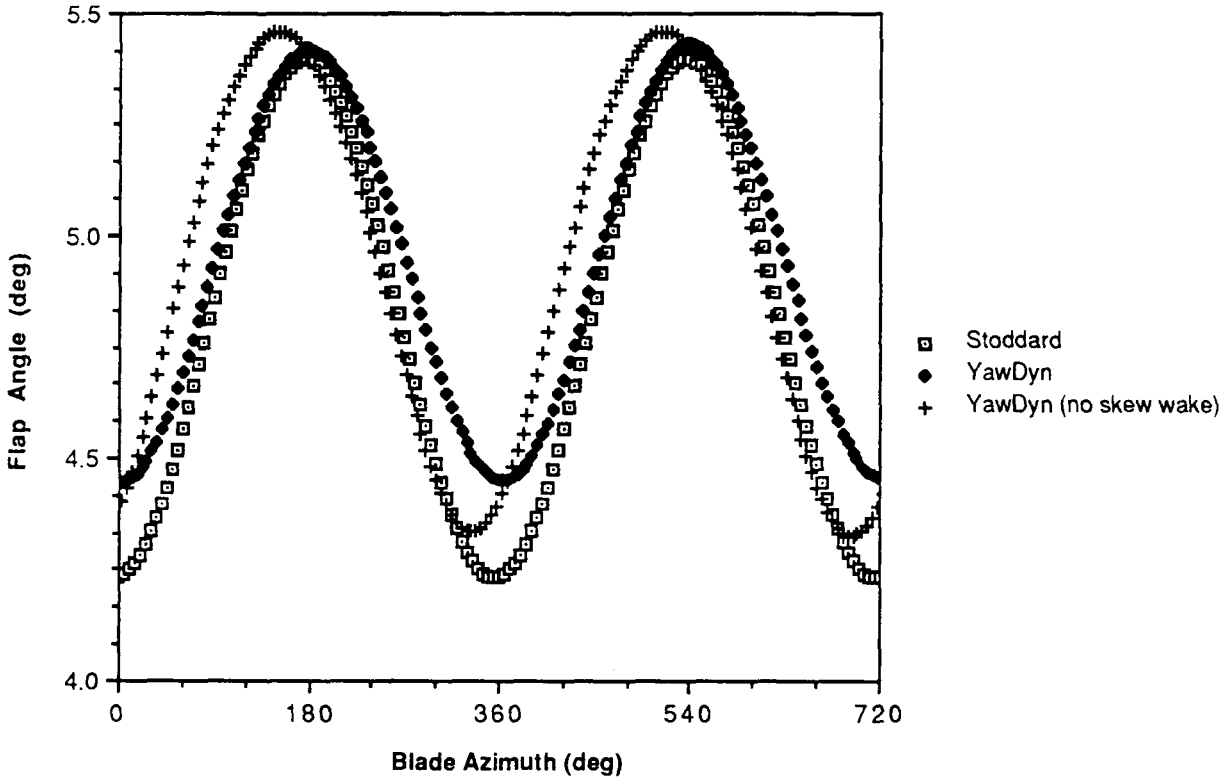

Figure 3.8 Comparison of Stoddard Predictions with YawDyn (both with and without the skewed wake correction). Yaw angle = 20°, Wind speed = 22 ft/sec k_β=8.35x10^4 ft-lb/rad.

Finally, the method of Stoddard can be used to estimate mean yaw moments for the modified Enertech rotor. Since only the first harmonics of the motion are calculated, the method will not predict the 3p yaw moment (which results from 2p and 4p motions). Figure 3.9 shows the mean moments estimated by three methods. The Stoddard method is documented in Stoddard's 1988 paper. The YawDyn model was used both with and without the skewed wake correction for comparison. As noted earlier the YawDyn method with the skewed wake correction is felt to be the most accurate method. It is the method which gave reasonable agreement with wind tunnel test results as discussed in the first annual report completed under this contract.

The three methods clearly give markedly different results, though the Stoddard and YawDyn predictions all show a decrease of mean yaw moment with increasing yaw angle (over this limited range of angles).

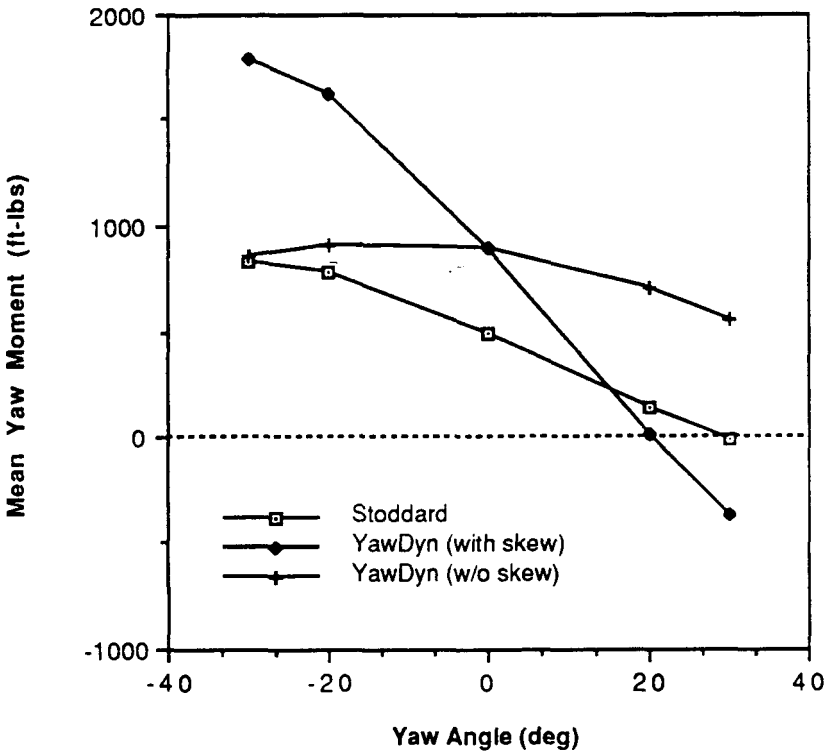

Figure 3.9 Comparison of modified Enertech Mean Yaw Moments Predicted using Three Methods. V=22 ft/sec, Yaw rate=0, k_β=8.35x10^4 ft-lb/rad.

3.3 Comparisons of Predicted and Measured Howden 330 kW Results

Ultimately a thorough comparison of predicted and measured performance of a variety of full-scale wind turbines is required to prove or disprove the validity of the prediction method. Little data is available for this purpose at present, but efforts are currently underway in the wind energy community to obtain reliable and complete data sets for model validation. One such effort involved field testing of a Howden 330 kW prototype in the San Gorgonio Pass. The tests and the turbine are described in a report by Wehrey, et al, 1988. One series of tests ran the system at various fixed yaw angles while measuring yaw and blade loads. The data tapes from these tests were provided to the University of Utah by SERI and the data were analyzed using the methods described earlier in this report.

The figures that follow show results of the predictions of the SDOF program and the test data. First the flap moment data are compared and then the yaw moment data. Unfortunately, much of the data gathered in the yaw tests was invalid and could not be used. Though data were taken over a wide range of wind speeds, the blade flap data was invalid in light winds and the yaw data was invalid in higher winds. To compound

the problem, there was no test run completed with both valid yaw and flap data. Nevertheless the valid data that are available are extremely useful to the present purpose of model validation.

Figure 3.10 shows the mean flap moment as a function of disc-averaged wind speed. Data are shown for both positive and negative yaw angles and it can be seen that the yaw angle has a slight effect on the mean yaw moment. Predicted values are shown for wind speeds of 20 and 30 mph. The bars represent the range of predicted values for yaw angles between +30° and -30°. The slight dependence of mean moment upon yaw angle, the strong dependence upon wind speed and the magnitudes of the moments are all predicted reasonably well. The predictions are somewhat higher than the measured values. The reason for this discrepancy is unknown though it is common in predictions of this type. The most likely explanation for the discrepancy is a small discrepancy in the steady, operating coning angle of the rotor. Zero coning is built into the rotor and the analysis assumed zero coning as well. SDOF does not compute or account for the "live" steady coning.

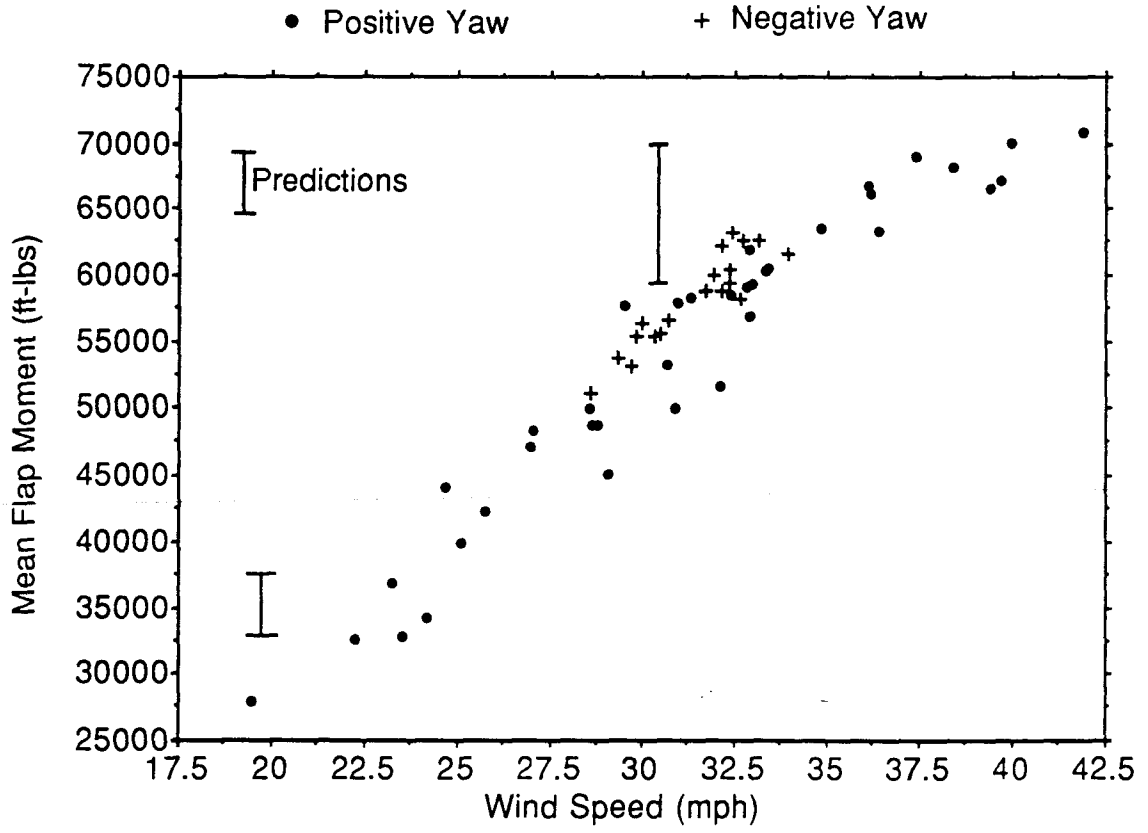

Figure 3.10 Mean Flap Moment for the Howden 330-kW Prototype as a Function of Disc-Averaged Wind Speed.

The one-per-revolution (1p) flap moment is shown in Figure 3.11. Again a range of predictions is shown for various amounts of wind shear, for yaw angles of ±30°. Note the 1p flap has a strong dependence upon yaw angle. Essentially the yaw error (and the resulting advancing and retreating blade effect) can compensate for wind shear and reduce the cyclic moments for negative yaw or increase the moments for positive yaw. The SDOF program predicts the trend and magnitudes more accurately at 30 mph than it does at 20 mph. Future work will use YawDyn predictions to see if blade motion is important in explaining this observed behavior.

It can be seen that SDOF makes reasonable predictions of mean and 1p flap moments, though the accuracy of some of the predictions is not as good as desired. However, the reader must be aware that the test data contain a large amount of scatter and many variables that are not shown on these curves do have a role in causing much of the scatter. The instantaneous wind shears (horizontal and vertical) and the reduced correlation between measured winds and the wind experienced by the rotor are probably the dominant causes of scatter. This is the reason a range of predictions is shown rather than single points. It is interesting to note that the prediction range is often of the same order of magnitude as the data scatter. The predictions were made for values of wind shears the same as were observed in the test data.

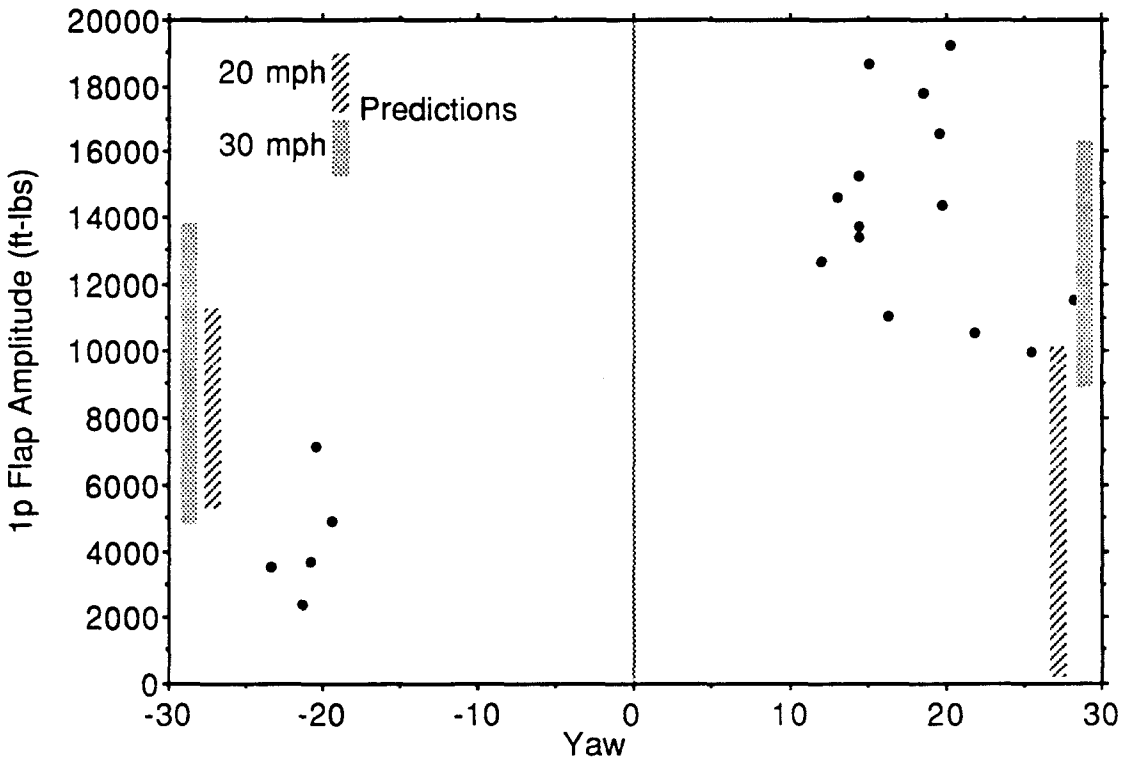

Figure 3.11 Howden 1p Flap Moment vs Yaw Angle for wind speeds between 20 and 30 mph

Next examine measured and predicted yaw moments. Recall these data were taken with the rotor locked in yaw. Figure 3.12 shows the mean yaw moment as a function of disc-averaged wind speed. Values for positive and negative yaw angles are shown. Yaw angles cover the range ±30°. It is surprising to see the weak (though definite) dependence of the yaw moment upon the yaw angle and enormous amount of scatter in the measured values. Predicted mean yaw moments ranged from -5000 ft-lbs for negative yaw angles to +7500 ft-lbs for positive yaw angles. The predictions are approximately correct for positive yaw and show significant error for negative yaw. The cause for this error is unknown, but the data base is small and contrary to both intuition and predictions. It is very encouraging that, in spite of the scatter and uncertainty of the data the predictions appear valid for positive yaw angles.

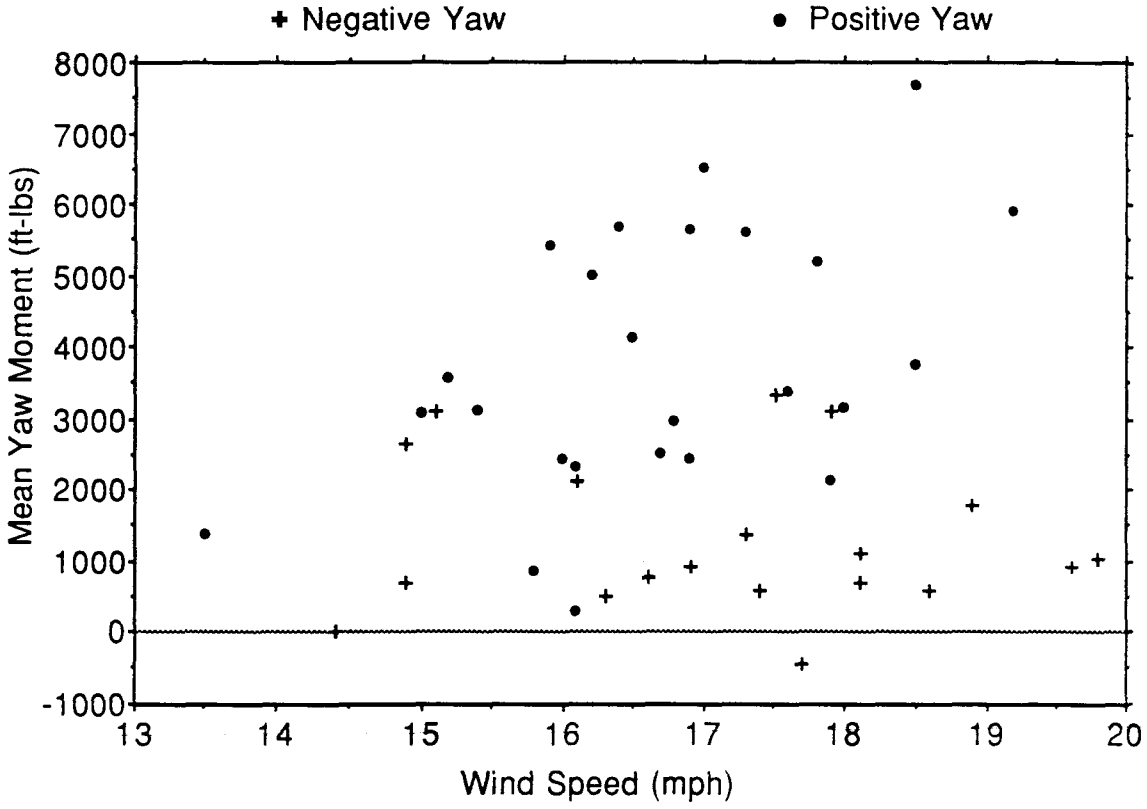

Figure 3.12 Variation of Mean Yaw Moment with Disc-Averaged Wind Speed

The cyclic yaw moment of a three-blade rotor is dominated by 3p terms in the absence of rotor imbalance that can cause high 1p moments. Figure 3.13 shows the 3p yaw moment as a function of four variables: yaw angle, wind speed, horizontal shear of the free stream wind and vertical shear of the wind. The correlation coefficient with each of the variables is shown on the figures as well. It can be seen that the 3p yaw moment correlates weakly with each of the four variables. Predictions are not shown on the curves because there is no single variable which dominates the behavior. However, predicted values ranged from 400 to 9000 ft-lbs for the range of independent variables shown in these figures. The measured values span from 500 to 4000 ft-lbs. The larger predicted values are most likely the result of a combination of independent variables which did not occur in the testing.

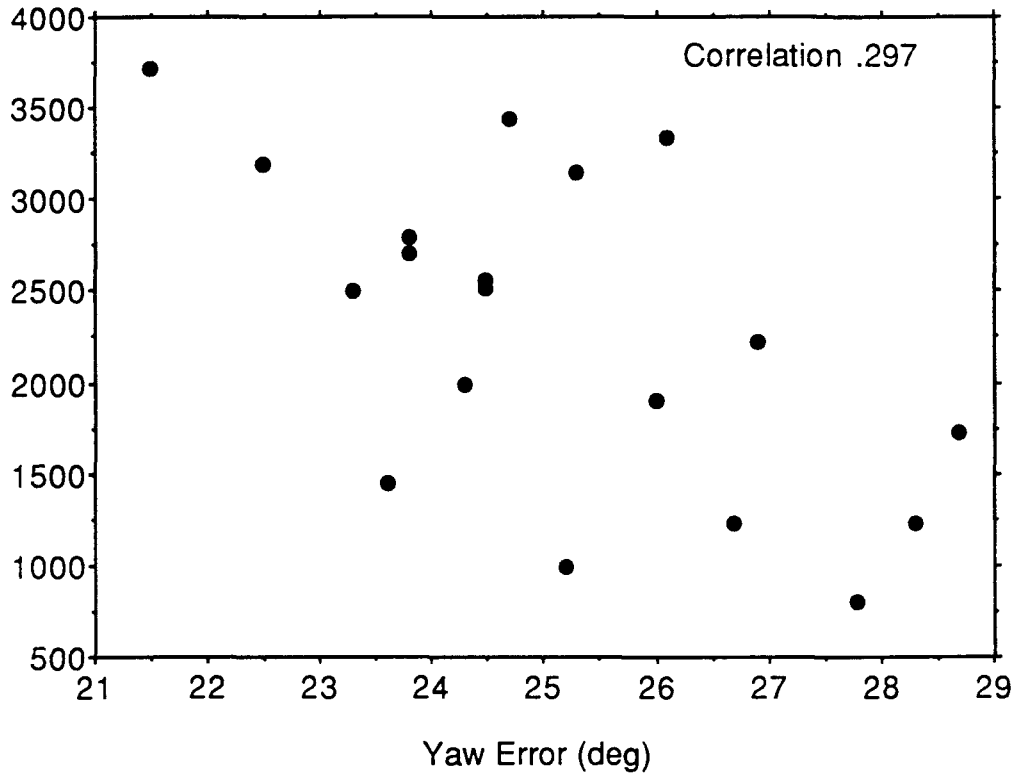

Figure 3.13a Dependence of 3p Yaw Moment upon Yaw Angle.

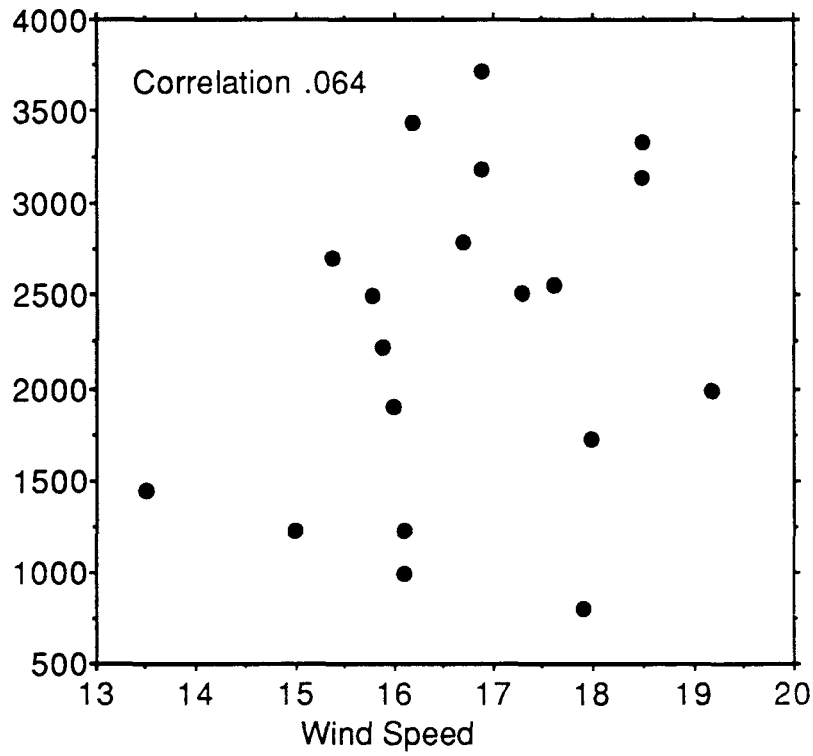

Figure 3.13b Dependence of 3p Yaw Moment upon Wind Speed.

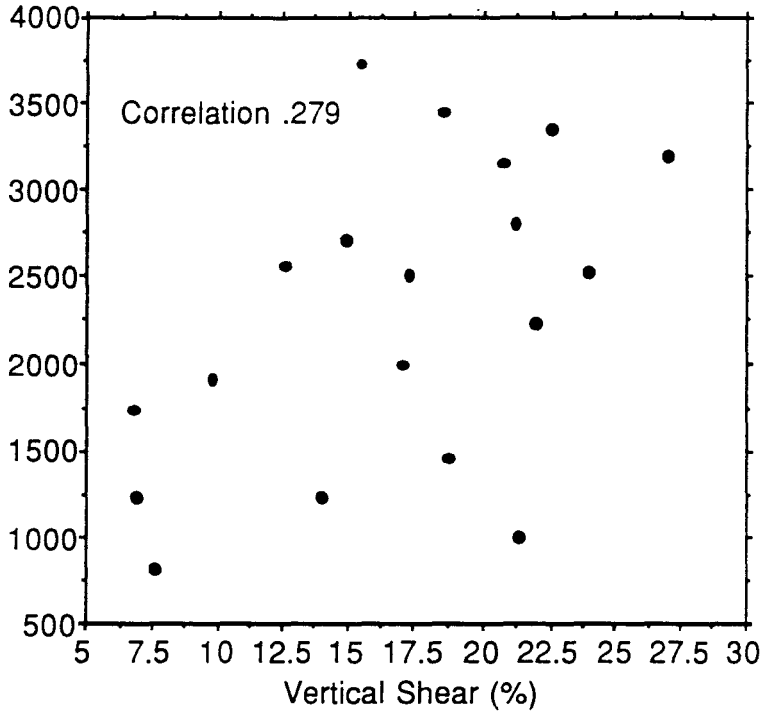

Figure 3.13c Dependence of 3p Yaw Moment upon Vertical Wind Shear.

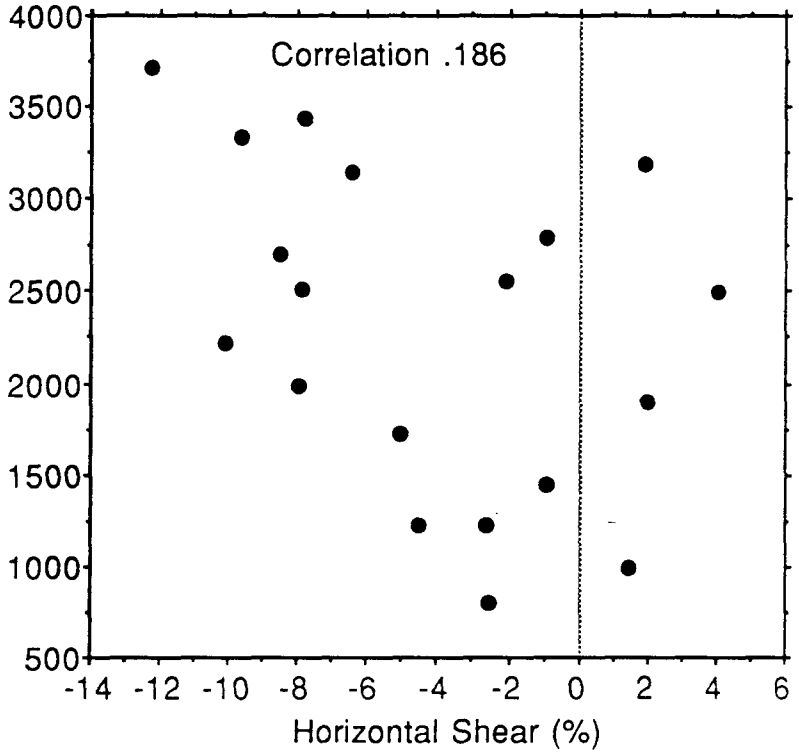

Figure 3.13d Dependence of 3p Yaw Moment upon Horizontal Wind Shear.

In summary, the program SDOF is reasonably accurate in its prediction of blade flap moments. The predictions of yaw moments are, not surprisingly, less accurate. The yaw moments are certainly of the correct order of magnitude and show many of the correct trends. They correctly show a dependence upon wind speed, yaw angle, and wind shears. The validation is made difficult by the scatter in the test data, the limited size of the data base (which makes it challenging to discern trends from the scatter), and by the large number of important dependent variables and the absence of one dominant independent variable. However, the validation effort thus far is very encouraging and provides strong incentive for continuing the work as new data become available.

Section 4.0
CONCLUSIONS

A relatively simple numerical model of yaw behavior has been developed and the first phase of validation has been completed with encouraging results. The model is called YawDyn and features detailed calculation of yaw and blade motions and loads. The model incorporates stall aerodynamics, skewed wake corrections to the basic blade element/momentum method, wind shears, tower shadow and a detailed description of the system and blade geometry. Two simpler derivative models (one for blade flap behavior called UFlap and one for rigid rotor yaw behavior called SDOF) have also been developed and subjected to extensive testing.

Validation of the models has proceeded along three parallel paths. First the models were compared one with another to test the internal consistency and accuracy of the integration algorithms. These tests concluded that the models produce identical results when applied to a rotor where results should be identical. For instance, YawDyn and SDOF both predict the same free-yaw behavior for a rotor with very stiff blades.

The models were then compared with estimates of blade flap motion using the method of Stoddard. Again, when the modeled rotor was appropriate the numerical and algebraic methods produced the same results. From this one can conclude that the governing equations and solution method for the numerical models are correct, at least in the linear terms. Comparisons of estimates of mean yaw moments did not produce identical results. The reason for this is not known, but the Stoddard method is unproven and suspect for two reasons. First, the yaw moments are strongly influenced by skewed wake effects, which are not considered in the current Stoddard model. Second, the yaw moments result from small, instantaneous differences in the root forces and moments on all of the rotor blades. Thus, small errors in the blade loads can result in much larger errors (on a percentage basis) in the yaw moments. It is therefore not surprising that prediction of yaw moments is inherently more difficult than prediction of blade moments.

Finally, the models have been compared with test data. Although the predictions do not identically match the test data the results are quite encouraging. The correct trends are always predicted, even when those trends are counter-intuitive and complex. And the yaw loads are estimated with errors as low as 10% and as high as approximately 100%. This accuracy, though not as good as desired, is apparently better than any past efforts.

Several items of interest to yaw system designers have been learned as a result of this research. For a rigid rotor, horizontal wind shear is the most likely cause of poor yaw tracking of free-yaw systems and high yaw moments on driven-yaw gears (see Cui et al, 1988). Such shear can be persistent or of short duration, but either way it will have serious consequences. The mean yaw moment of a system which has good yaw tracking will often be much less than the cyclic yaw moment. This means the yaw drive

will be subjected to fully reversing load cycles at a frequency of three cycles per rotor revolution (for a three-blade rotor). Consideration of only the mean yaw moments in the design process will virtually ensure yaw drive fatigue failures. A lightly damped, rigid, free-yaw system will have great potential for large yaw errors and high yaw rates. The high yaw rates will expose the entire wind system to potentially destructive gyroscopic moments. Means of coping with this problem include addition of mechanical yaw damping or friction, or softening of the blade-hub attachment. Attempts to shorten the distance from the hub to the yaw axis or to change the rotor coning will not be likely to succeed in solving a yaw problem.

Design questions and tradeoffs such as raised in the preceding paragraph can now be addressed in a quantitative manner using the yaw behavior models being developed and tested under this contract.

Section 5.0
FUTURE WORK

The tasks remaining are concerned with completion of the YawDyn model and, more important, validation of the methods through comparisons with additional test data. A summary list of tasks includes the following:

Comparison of YawDyn and SDOF predictions with data from the SERI Comprehensive Experiment tests of a rigid three-blade rotor.

Provide a better means of estimating initial conditions and determining a trim solution to start the YawDyn model for free-yaw motion.

Investigate the need to include effects of the vertical component of the wind speed approaching the rotor and addition of the angular induction factor.

Explore alternate methods of correcting for skewed wake effects.

Add the teetering degree of freedom to YawDyn and SDOF.

Conduct comparisons with teetering rotor data (the turbine has not yet been identified).

LIST OF REFERENCES

Cui, X., Hansen, A.C., and Siedschlag, 1988, "Yaw Dynamics of Horizontal Axis Wind Turbines, First Annual Report". SERI Report SERI/STR-217-3309.

Hansen, A. C., 1988, "A Method for Analyzing Wind Turbine Dynamic Response Test Data". ASME Journal of Solar Energy Engineering, Vol. 110, No. 4, pp335-339.

Johnson, B. 1987, Personal communication.

Swift, Andrew H.P., 1981. "The Effects of Yawed Flow on Wind Turbine Rotors". PhD. Dissertation, Washington University, St. Louis, MO.

Stoddard, F.S., 1978, "Structural Dynamics, Stability and Control of High Aspect Ratio Wind Turbines". Rocky Flats Plant Report RFP-3027/67025/3533/79-9, available through NTIS. Also PhD. Dissertation, University of Massachusetts at Amherst.

Stoddard, F.S., 1988, "An Analytical Method for Calculating the Yaw Moment and Evaluating the Yaw Stability of Horizontal Axis Wind Turbines". Presented at the 7th ASME Wind Energy Symposium, New Orleans, LA.

Wehrey, M., Redmond, I., Anderson, C., Jamieson, P., 1988, "Dynamic Response of a 330-kW Horizontal Axis Wind Turbine Generator". SERI Report SERI/STR-217-3203.

APPENDIX A

LISTING OF THE YawDyn PROGRAM

```
C
C     ************************YAWDYN*****************************
C           YAWDYN CALCULATES YAW ANGLE, YAW RATE, YAW MOMENT
C           FLAP ANGLE, FLAP RATE AND FLAP MOMENT FOR EACH BLADE OF
C           A 2 OR 3-BLADED HAWT.
C
C           FILE YAWDYN.IPT CONTAINS THE HAWT DATA AND STARTING
C           CONDITIONS. FILE YAWDYNPLT.OPT IS THE OUTPUT FILE.
C           ALL UNITS ARE FEET,SLUGS, SECONDS, POUNDS FORCE AND DEGREES.
C
C           YAW MOMENT AND FLAP MOMENT VS. AZIMUTH FOR ONE SELECTED
C           REVOLUTION OF THE ROTOR ARE STORED IN FILE YAWDYN.OPT
C
C           FILE YHARMON.IPT IS CREATED CONTAINING YAW AND FLAP MOMENT
C           RESULTS FOR ONE SELECTED REVOLUTION OF THE ROTOR.
C           THIS FILE IS USED BY PROGRAM YHARMON.FOR TO CALCULATE
C           HARMONIC CONTENT OF THE YAW AND FLAP LOADS.
C
C           WRITTEN BY CUI XUDONG, N. SIEDSCHLAG AND C. HANSEN,
C           UNIVERSITY OF UTAH,   LAST MODIFIED 4/15/88
C
C     ************************************************************
C
C       LIST OF VARIABLES:
C
C           A1     =MOMENT OF INERTIA OF NACELLE
C           A2     =MOMENT OF INERTIA OF ROTOR SHAFT
C           A3     =MOMENT OF INERTIA OF HUB
C           BM     =BLADE MASS
C           FS     =BLADE HINGE SPRING STIFFNESS
C           B1     =BLADE PITCH MOMENT OF INERTIA
C           B2     =BLADE FLAP MOMENT OF INERTIA
C           B3     =BLADE LAG MOMENT OF INERTIA
C           R      =ROTOR RADIUS
C           RH     =BLADE HINGE OFFSET
C           HH     =HUB HEIGHT
C           RB     =DISTANCE FROM BLADE HINGE TO BLADE CENTER OF MASS
C           B      =NUMBER OF BLADES
C           PC     =PRECONING ANGLE
C           VB     =MEAN AXIAL WIND SPEED
C           RPM    =ROTOR ROTATIONAL SPEED (IN REVOLUTIONS/MINUTE)
C           HSHR   =HORIZONTAL WIND SHEAR
C           VSHR   =VERTICAL WIND SHEAR
C           VELDEF=TOWER SHADOW VELOCITY DEFICIT FRACTION
C           SL     =DISTANCE FROM YAW COLUMN TO ROTOR HUB
C           AV     =VISCOUS DAMPING COEFICIENT (FT-LBS-SECS/RADIAN)
C           AF     =DRY FRICTION MOMENT
C           C1N    =NACELLE AERODYNAMIC MOMENT COEFFICIENT
C           C2N    =NACELLE AERODYNAMIC MOMENT COEFFICIENT
C           AA     =BLADE GEOMETRIC ANGLE OF ATTACK VECTOR
C           CL     =BLADE AERODYNAMIC LIFT COEFFICIENTS
C           CD     =BLADE AERODYNAMIC DRAG COEFFICIENTS
C           THETA =BLADE TWIST ANGLES (AT EACH OF TEN BLADE STATIONS)
C           C      =BLADE CHORD LENGTHS (AT TEN BLADE STATIONS)
C           RHO    =AIR DENSITY
C           N      =NUMBER OF DATA POINTS TO BE COMPUTED
C           SECTOR=NUMBER OF SECTORS
C           PITCH =PITCH ANGLE OF THE BLADE
```

```
C           PITNOW=PITCH OF BLADE IN CURRENT CALCULATION
C           SKEW   =SKEWED WAKE FACTOR
C           Q      =INSTANTANEOUS VALUES OF iTH BLADE DEGREES-OF-FREEDOM:
C                  FLAP ANGLE, FLAP RATES, YAW ANGLE AND YAW RATE
C           MYAW   = YAW MOMENT
C           MFLAP  = FLAP MOMENT

C
C       ************************************************************
C                 MAIN  PROGRAM
C       ************************************************************
C
      COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
     *       V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *       B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *       VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
      COMMON/DENSE/ RHO
      DIMENSION QP(8),F(8,4),Q1(8),Q2(8),Q3(8),Q4(8)
      CHARACTER*80 TITLE
      EQUIVALENCE (F(1,1),Q1(1)),(F(1,2),Q2(1)),
     *            (F(1,3),Q3(1)),(F(1,4),Q4(1))
C
C
C       ************************************************************
C                 OPEN INPUT AND OUTPUT FILES
C       ************************************************************
C
      OPEN (UNIT=10, FILE= 'YAWDYN.IPT',STATUS='OLD',IOSTAT=IERR)
      IF(IERR.NE.0)  THEN
            WRITE(*,*) 'ERROR OPENING YAWDYN.IPT'
            WRITE(*,*) 'IOSTAT=',IERR
            STOP
      ENDIF
C
      OPEN (UNIT=12, FILE= 'YAWDYNPLT.OPT',STATUS='NEW',IOSTAT=IERR)
      IF(IERR.NE.0)  THEN
            WRITE(*,*) 'ERROR OPENING YAWDYNPLT.OPT'
            WRITE(*,*) 'IOSTAT=',IERR
            STOP
      ENDIF
C
      OPEN (UNIT=14, FILE= 'YHARMON.IPT',STATUS='NEW',IOSTAT=IERR)
      IF(IERR.NE.0)  THEN
            WRITE(*,*) 'ERROR OPENING YHARMON.IPT'
            WRITE(*,*) 'IOSTAT=',IERR
            STOP
      ENDIF
C
      OPEN (UNIT=100, FILE='YAWDYN.OPT',STATUS='NEW',IOSTAT=IERR)
      IF(IERR.NE.0)  THEN
            WRITE(*,*) 'ERROR OPENING YAWDYN.OPT'
            WRITE(*,*) 'IOSTAT=',IERR
            STOP
      ENDIF
C
C       ************************************************************
C          READ THE INPUT FILE
C       ************************************************************
```

```
C
      DELTA=0.
      READ(10,2100) TITLE
 2100 FORMAT(A)
      READ(10,*) A1,A2,A3,BM,FS
      YI=A1+A2+A3
      READ(10,*) B1, B2, B3
      READ(10,*) R,RB,RH,HH,B,PC
      NB=B
      READ(10,*) VB,RPM,HSHR,VSHR,VELDEF
        PI=4.*ATAN(1.)
        TWOPI=2.*PI
        DEG=180./PI
        REVS=RPM*PI/30.
      READ(10,*) SL,AV,AF,C1N,C2N
      READ(10,*) (PITCH(I),I=1,NB)
      READ(10,*) Q(1),Q(2),Q(3),Q(4)
      READ(10,*) N
      READ(10,*) SECTOR,RHO
C      WRITE(*,2100) TITLE
      WRITE(*,*) ' '
      WRITE(*,*) ' SELECTED DATA FROM ONE REVOLUTION OF THE'
      WRITE(*,*) ' ROTOR WILL BE PRINTED TO A DISC FILE'
      WRITE(*,*) ' ENTER THE REVOLUTION NUMBER YOU WISH TO SEE'
      READ(*,*)  MC
C
      WRITE(*,*) ' THE ANGLE OF ATTACK, CL AND CD FOR ONE'
      WRITE(*,*) ' BLADE ELEMENT WILL BE WRITTEN TO THE SAME'
      WRITE(*,*) ' FILE.  ENTER THE BLADE ELEMENT NUMBER YOU'
      WRITE(*,*) ' WISH TO SEE'
      READ(*,*)  NS
C
      WRITE(*,*) ' CHOOSE LINEAR OR POWER LAW VERTICAL WIND SHEAR '
      WRITE(*,*) ' ENTER 1 FOR LINEAR SHEAR'
      WRITE(*,*) ' ENTER 2 POWER LAW VERTICAL SHEAR'
      READ  (*,*)  VLP
C
      WRITE (*,*) ' CHOOSE FREE YAW OR FIXED YAW  '
      WRITE (*,*) ' ENTER 1 FOR FREE YAW. 0 FOR FIXED YAW  '
      READ (*,*)  IYAC
C
      WRITE(*,*) ' THE RESULTS OF EVERY Nth CALCULATION WILL'
      WRITE(*,*) ' BE WRITTEN TO THE SCREEN AND FILE YAWDYNPLT.OPT'
      WRITE(*,*) ' ENTER N ( >=1)'
      READ (*,*) IPRINT
C
      WRITE(100,2100) TITLE
      WRITE(100,*) '  WIND SPEED AT HUB (FT/SEC) =' ,VB
      WRITE(100,*) '  ROTOR SPEED (RPM) =' ,RPM
      WRITE(100,*) ' '
      WRITE(100,*) '  ROTOR RADIUS (FT) =' ,R
      WRITE(100,*) '  HUB RADIUS (FT) ='   ,RH
      WRITE(100,*) '  HUB HEIGHT (FT) ='   ,HH
      WRITE(100,*) '  PITCH ANGLES (DEG) =',(PITCH(I),I=1,NB)
      WRITE(100,*) '  BLADE CENTER OF GRAVITY (FT) =' ,RB
      WRITE(100,*) '  YAW AXIS-TO-HUB DISTANCE (FT) =',SL
      WRITE(100,*) '  NUMBER OF BLADES =' ,B
      WRITE(100,*) '  PRE-CONING ANGLE (DEG) =' ,PC
```

```
      WRITE(100,*) ' '
      WRITE(100,*) '  MASS OF BLADE (SLUG) =' ,BM
      WRITE(100,1006) B1, B2, B3
 1006 FORMAT('  BLADE MOMENTS OF INERTIA (SLUG*FT^2) =',3(F10.1,1X))
      WRITE(100,1007) A1, A2, A3
 1007 FORMAT('  NACELLE MOMENTS OF INERTIA (SLUG*FT^2) =',3(F10.1,1X))
      WRITE(100,*) '  BLADE STIFFNESS COEF. (LB-FT/RAD) =' ,FS
      FREQ=SQRT(FS/B2)/TWOPI
      WRITE(100,*) '  BLADE NATURAL FREQUENCY (HZ) =',FREQ
      P=FREQ*60./RPM
      WRITE(100,*) '  BLADE NATURAL FREQUENCY (P#) =',P
      WRITE(100,*) '  YAW AXIS FRICTION (FT-LB) = ',AF
      WRITE(100,*) '  YAW AXIS DAMPING (FT-LB-SEC) = ',AV
      WRITE(100,*) ' '
      WRITE(100,*) '  LINEAR HORIZONTAL WIND SHEAR'
      WRITE(100,*) '  SHEAR COEFFICIENT = ' , HSHR
      IF (VLP .EQ. 1)  THEN
         WRITE(100,*) '  LINEAR VERTICAL WIND SHEAR'
         WRITE(100,*) '  SHEAR COEFFICIENT = ' , VSHR
      ELSE
         WRITE(100,*) '  POWER LAW VERTICAL WIND SHEAR '
         WRITE(100,*) '  POWER LAW EXPONENT = ' , VSHR
      ENDIF
      WRITE(100,*) ' '
C
C     READ SECTION CL AND CD CHARACTERISTICS
C     NLIFT=NUMBER OF POINTS DEFINING CL VS. ALPHA CURVE
C     NDRAG=NUMBER OF POINTS DEFINING CD VS. ALPHA CURVE
C
      WRITE (100,1002)
1002  FORMAT(/,'  AIRFOIL CHARACTERISTICS:',/,' ANGLE OF ATTACK(DEG)
     *         LIFT COEF.         ')
      READ(10,*) NLIFT,NDRAG
      DO 10 I=1,NLIFT
            READ(10,*) AL(I),CL(I)
            WRITE(100,1003) AL(I),CL(I)
            AL(I)=AL(I)/DEG
  10  CONTINUE
C
1003  FORMAT(4X, F10.4, 8X, F10.4 )
      WRITE(100,*) ' '
      WRITE(100,1012)
1012  FORMAT(/,'  AIRFOIL CHARACTERISTICS:',/,' ANGLE OF ATTACK(DEG)
     *         DRAG COEF.         ')
      DO 12 I=1,NDRAG
            READ(10,*) AD(I),CD(I)
            WRITE(100,1013) AD(I),CD(I)
            AD(I)=AD(I)/DEG
12    CONTINUE
1013  FORMAT(4X, F10.4, 8X, F10.4 )
      WRITE(100,*) ' '
C
C     READ TWIST ANGLE AND CHORD AT EACH SECTION
C
      WRITE(100,1005)
      WRITE(100,*) ' '
      DO 20 I=1,10
            READ(10,*) THETA(I),C(I)
```

```
            WRITE(100,1004) THETA(I),C(I)
            THETA(I)=THETA(I)/DEG
  20    CONTINUE
1005    FORMAT(/' TWIST ANGLE(DEG)    CHORD(FT)   ')
1004    FORMAT(1X,F10.4 ,4X , F7.4 )
        WRITE(100,*) ' '
C
        WRITE(100,*) '  INITIAL YAW ANGLE (DEG) = ' , Q(3)
        WRITE(100,*) ' '
C
C       ************************************************************
C            SET UP THE INITIAL VALUES
C       ************************************************************
C
        PC=PC/DEG
        V=VB
        PSI=0.
        H=2.*PI/SECTOR
        Q(1)=Q(1)/DEG+PC
        Q(2)=Q(2)/DEG/REVS
        Q(3)=Q(3)/DEG
        Q(4)=Q(4)/DEG/REVS
        NB=IFIX(B)
        DO 40 I=1,NB
            J=I*2-1
            K=J+1
            QP(J)=Q(1)
            QP(K)=Q(2)
   40   CONTINUE
        QP(7)=Q(3)
        QP(8)=Q(4)
C
C
        WRITE(14,4000) IFIX(SECTOR)
 4000   FORMAT(I5)
        WRITE(*,*) 'RUNNING ',N,' POINTS'
        WRITE(*,*) 'WITH ',SECTOR,' POINTS PER REVOLUTION'
C
C       READ(11,*) VENT,V,DEL
C       DELTA=DEL/DEG
C       FILE 11 IS FOR STUDY GUST.
C
C
C          ************************************************************
C              START RUNGE-KUTTA CALCULATION
C          ************************************************************
C
        CALL FK(PSI,QP,Q1,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
        CALL RK(PSI,H,QP,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
             TIME=PSI/REVS
             REM=AMOD(PSI,TWOPI)
             REMM=REM*DEG
             YE=QP(7)*DEG
             YR=QP(8)*DEG*REVS
             FE=QP(1)*DEG
             FR=QP(2)*DEG*REVS
        WRITE (*,2000) TIME,YE,YR,FE,FR
 2000   FORMAT(1X,'T= ',F5.1,'  YE= ',F5.1,'  YR= ',F5.1,
```

```
     *            '  FE= ',E10.2, '  FR= ',E10.2  )
      WRITE (12,130) TIME,CHAR(9),YE,CHAR(9),YR,CHAR(9),
     *          YAWM,CHAR(9),FE,CHAR(9),FR,CHAR(9),AMFP
C
C       SECOND R-K PASS
C
      CALL FK(PSI,QP,Q2,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
      CALL RK(PSI,H,QP,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
            TIME=PSI/REVS
            REM=AMOD(PSI,TWOPI)
            REMM=REM*DEG
            YE=QP(7)*DEG
            YR=QP(8)*DEG*REVS
            FE=QP(1)*DEG
            FR=QP(2)*DEG*REVS
      WRITE (*,2000) TIME,YE,YR,FE,FR
      WRITE (12,130) TIME,CHAR(9),YE,CHAR(9),YR,CHAR(9),
     *          YAWM,CHAR(9),FE,CHAR(9),FR,CHAR(9),AMFP
C
C       THIRD R-K PASS
C
      CALL FK(PSI,QP,Q3,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
      CALL RK(PSI,H,QP,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
            TIME=PSI/REVS
            REM=AMOD(PSI,TWOPI)
            REMM=REM*DEG
            YE=QP(7)*DEG
            YR=QP(8)*DEG*REVS
            FE=QP(1)*DEG
            FR=QP(2)*DEG*REVS
      WRITE (*,2000) TIME,YE,YR,FE,FR
      WRITE (12,130) TIME,CHAR(9),YE,CHAR(9),YR,CHAR(9),
     *          YAWM,CHAR(9),FE,CHAR(9),FR,CHAR(9),AMFP
C
C       SWITCH TO PREDICTOR-CORRECTOR CALCULATION
C
      CALL FK(PSI,QP,Q4,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
      NN=N-4
      WRITE(100,*) '  FLAP MOMENT INCLUDES AERODYNAMIC,
     *  GRAVITY AND CENTRIFUGAL MOMENTS. '
      WRITE(100,1000) MC,NS
 1000   FORMAT(/'   DATA FOR CYCLE NUMBER',I3,', BLADE ELEMENT ',I2,
     *      //'   PSI       YAWMOMENT
     *  FLAPMOMENT    ALPHA      CL        CD        A'/'   DEG
     *     FT-LB          FT-LB          DEG                ')
      WRITE(100,*) ' '
C
C     LOOP THROUGH PREDICTOR CORRECTOR
C
      DO 50 I=1,NN
      CALL AB(PSI,H,QP,F,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
            TIME=TIME+H/REVS
            REM=AMOD(PSI,TWOPI)
            REMM=REM*DEG
            PM1=MC*TWOPI
            PM2=(MC+1)*TWOPI
            IF( PSI.LE.PM2 .AND. PSI .GE. PM1 ) THEN
                WRITE (100,1001) REMM, YAWM, AMFP,ALF,CLL,CDD,AAA
```

```
               WRITE (14,4010) REMM, YAWM, AMFP
 4010          FORMAT(3E15.6)
            ENDIF
C
 1001   FORMAT( 1X,F7.2 ,2(4X,E10.4),2X,F8.4,2X,F7.4,2X,
     *                F7.4,2X,F7.4)
            YE=QP(7)*DEG
            YR=QP(8)*DEG*REVS
            FE=QP(1)*DEG
            FR=QP(2)*DEG*REVS
         IF(I/IPRINT .EQ. FLOAT(I)/FLOAT(IPRINT) ) THEN
            WRITE (*,2000) TIME,YE,YR,FE,FR
            WRITE (12,130) TIME,CHAR(9),YE,CHAR(9),YR,CHAR(9),
     *           YAWM,CHAR(9),FE,CHAR(9),FR,CHAR(9),AMFP
         ENDIF
 130        FORMAT( ' ',G10.4, 6(A1,E10.4) )
  50  CONTINUE
C
      CLOSE(10)
      CLOSE(12)
      CLOSE(14)
      CLOSE(100)
      STOP
      END
C
C     ************************************************************
C     SUBROUTINES
C     ************************************************************
C     AB SOLVES THE FIRST ORDER EQUATION USING ADAMS-BASHFORTH
C     PREDICTOR-CORRECTOR SCHEME.
C     ************************************************************
C
C     THE MATRIX F CONTAINS THE PREVIOUS FOUR SETS OF DERIVATIVE
C     FUNCTIONS EVALUATED BY SUBROUTINE FK.  THE DEGREES-OF-FREEDOM
C     CONTAINED IN VECTOR QP ARE ADVANCED
C     ONE STEP SIZE, H, AND THE MATRIX F IS SHIFTED TO STORE
C     THE NEW SET OF DERIVATIVE FUNCTION VALUES DURING EACH RUN OF AB.
C
      SUBROUTINE AB(PSI,H,QP,F,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
      COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
     *      V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *      B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *      VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
      COMMON/DENSE/ RHO
      DIMENSION QP(8),F(8,4),PK(8),CK(8),FK1(8)
      DO 10 I=1,8
            PK(I)=QP(I)+(55.*F(I,4)-59.*F(I,3)+37.*F(I,2)-
     *      9.*F(I,1))*H/24.
  10  CONTINUE
C
C       FILE 11 IS FOR STUDY GUST.
C       READ(11,*) vent,V,DEL
C       DELTA=DEL/DEG
C
      PSI=PSI+H
      CALL FK(PSI,PK,FK1,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
      DO 20 I=1,8
            CK(I)=QP(I)+(9.*FK1(I) +19.*F(I,4)-5.*F(I,3)
```

```
     *          +F(I,2))*H/24.
                QP(I)=(251.*CK(I)+19.*PK(I))/270.
                F(I,1)=F(I,2)
                F(I,2)=F(I,3)
                F(I,3)=F(I,4)
                F(I,4)=FK1(I)
  20  CONTINUE
      RETURN
      END
C
C       ********************************************************
C            RK SOLVES THE EQUATION USING RUNGE-KUTTA METHOD.
C       ********************************************************
C
C     THE NEW VALUES ARE RETURNED IN QP. SUBROUTINE
C     FK IS USED TO EVALUATE THE DERIVATIVE FUNCTIONS.
C
      SUBROUTINE RK(PSI,H,QP,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
      COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
     *      V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *      B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *      VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
      COMMON/DENSE/ RHO
      DIMENSION QP(8),PK(8),QK(8),RR(8),SK(8),TK(8)
      CALL FK(PSI,QP,TK,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
      DO 10 I=1,8
            PK(I)=TK(I)*H
  10  TK(I)=QP(I)+PK(I)/2.
C
C
      PSI=PSI+H/2.
      CALL FK(PSI,TK,QK,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
      DO 20 I=1,8
                  QK(I)=QK(I)*H
                  TK(I)=QP(I)+QK(I)/2.
  20  CONTINUE
      CALL FK(PSI,TK,RR,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
      DO 30 I=1,8
            RR(I)=RR(I)*H
            TK(I)=QP(I)+RR(I)
  30  CONTINUE
C
C       W AND GAMMA ARE GUST VELOCITY AND DIRECTION
C       V=W
C       DELTA=GAMMA
C
      PSI=PSI+H/2.
      CALL FK(PSI,TK,SK,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
      DO 40 I=1,8
                  SK(I)=SK(I)*H
  40              QP(I)=QP(I)+(PK(I)+2.*(QK(I)+RR(I))+SK(I))/6.
      RETURN
      END
C
C       ********************************************************
C           VIND CALCULATES THE AXIAL INDUCTION FACTOR FOR EACH
C           ANNULAR SEGMENT AND SPECIFIED AZIMUTH ANGLE PSI.
C       ********************************************************
```

```
C
      SUBROUTINE VIND(A,J,X,PSI,VY,VZ,VN,VT)
     COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
     *      V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *      B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *      VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
     SOLID=B*C(J)/(PI*X*COS(PC))
     TS=0.
     TWOPI=2.*PI
     PSIDEG=DEG*AMOD( PSI, TWOPI )
     IF(PSIDEG .LT. 15.  .OR.  PSIDEG .GT. 345.)
     *       TS=VELDEF*(1.+COS(12.*PSI))/2.
     AI=0.1
     DAI1=.1
     STEP=0.5
     ICOUNT=0
     VT=(X*COS(Q(1))+RH)-VY*COS(Q(3))*COS(PSI)/REVS
     *      -VZ*(1.-TS)*SIN(Q(3))*COS(PSI)/REVS-
     *      (X*SIN(Q(1))+SL)*Q(4)*COS(PSI)
 10  ICOUNT=ICOUNT+1
     VN=( (VZ*(1.-TS)*COS(Q(3))-VY*SIN(Q(3)))*COS(Q(1))/REVS
     *          -SIN(PSI)*SIN(Q(1))*SIN(Q(3))*VZ*(1.-TS)/REVS
     *                  -VY*SIN(PSI)*SIN(Q(1))*COS(Q(3))/REVS
     *          -(X+RH*COS(Q(1))+SL*SIN(Q(1)))*Q(4)*SIN(PSI)
     *                  -X*Q(2) )*(1.-AI)
     PHI=ATAN2(VN,VT)
     W2=VN*VN+VT*VT
     ALPHA=PHI-THETA(J)-PITNOW/DEG
     IF( ALPHA .GT. PI ) ALPHA=ALPHA-PI
     IF( ALPHA .LT. -PI/2. ) ALPHA=ALPHA+PI
C
     CLA=GETCL(ALPHA)
        IF( CLA .LT. -900. ) CLA=FPL(ALPHA)
     CDA=GETCD(ALPHA)
        IF( CDA .LT. -900. ) CDA=FPD(ALPHA)
C
     CH=W2*REVS*REVS*SOLID*(CLA*COS(PHI)+CDA*SIN(PHI))/(2.*(VZ*VZ))
       IF(CH.LT.0.96) THEN
          AI1=(1.-SQRT(1.-CH))/2.
       ELSE
          AI1=.143+SQRT(.0203-.6427*(.889-CH))
       ENDIF
     DAI=AI1-AI
C
C    TEST FOR CONVERGENCE, STOP AFTER 100 ITERATIONS
C
     IF(ICOUNT .GT. 100) THEN
          WRITE(*,*) 'EXCESSIVE ITERATIONS TO FIND INDUCTION FACTOR'
          WRITE(*,*) 'ELEMENT= ',J,'    PSI= ',PSI*DEG
          WRITE(*,*) 'VN= ',VN,'    VT= ',VT
          WRITE(*,*) 'ALPHA= ',ALPHA*DEG,'    CL= ',CLA,'  CD= ',CDA
          WRITE(*,*) 'AI= ',AI,'     DAI= ',DAI
          PAUSE 'ENTER CR TO CONTINUE'
          STOP
     ENDIF
C
     IF(ABS(DAI).LE.0.01) GO TO 14
     IF( IFIX(SIGN(1.,DAI)) .NE. IFIX(SIGN(1.,DAI1)))
```

```
     *      STEP=.5*STEP
      AI=AI+STEP*DAI
      DAI1=DAI
      GO TO 10
C
  14  A=AI
C
      RETURN
      END
C
C       ************************************************************
C           FK EVALUATES THE FIRST DERIVATIVES OF THE COMPONENTS
C           OF VECTOR QR AND RETURNS THE VALUES IN VECTOR  QS.
C
C           SUBROUTINES F2 AND F4 ARE USED TO EVALUATE
C           THE DERIVATIVES OF THE VELOCITY COMPONENTS OF QR.
C       ************************************************************
C
      SUBROUTINE FK(PSI,QR,QS,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
      COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
     *      V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *      B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *      VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
      COMMON/DENSE/ RHO
      DIMENSION QR(8),QS(8)
      Q(3)=QR(7)
      Q(4)=QR(8)
      NB=IFIX(B)
C
      DO 10 I=1,NB
         J=2*I-1
         Q(1)=QR(J)
         K=2*I
         Q(2)=QR(K)
         QS(J)=QR(K)
         PSIA=PSI+FLOAT(I-1)*2.*PI/B
         PITNOW=PITCH(I)
         CALL F2(F21,PSIA)
         QS(K)=F21
  10  CONTINUE
      CALL F4(FY4,PSI,QR,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
      QS(7)=QR(8)
      QS(8)=FY4
C
      RETURN
      END
C
C
*****************************************************************
C
C     F2 COMPUTES THE FLAP ACCELERATION FUNCTION AND THE FLAPPING MOMENT
C     FOR THE HAWT BLADE AT AZIMUTH ANGLE PSI.
C
C
*****************************************************************
C
      SUBROUTINE F2(F21,PSI)
      COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
```

```
     *       V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *       B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *       VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
      COMMON/DENSE/ RHO
C
      CPSI=COS(PSI)
      SPSI=SIN(PSI)
      A1=BM*RB*RH/B2
      D1=B2*REVS*REVS
      A2=A1+(B3-B1)/B2+32.174*BM*RB*CPSI/D1
      A3=A1*SPSI*SPSI-(B3-B1)*CPSI*CPSI/B2
      A4=FS/D1
      F21=A4*PC - Q(4)*(1.+(B3-B1)/B2+2*A1)*CPSI
     *    - (A2+A4)*Q(1)
     *    - Q(4)*Q(4)*(A3*Q(1)-BM*RB*SL/B2)
      CALL YAM( YMX,FM,PSI,ALF,CLL,CDD,AAA )
      F21=F21+FM/D1
C     PRINT *, FS, F21
C     F21=0.
      RETURN
      END
C
C       ************************************************************
C          F4 COMPUTES THE YAW ACCELERATION FUNCTION AND YAW MOMENT
C       ************************************************************
C

      SUBROUTINE F4(FY4,PSI,QP,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
      COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
     *       V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *       B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *       VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
      COMMON/DENSE/ RHO
      DIMENSION QP(8),CY(10)
C
C     INITIALIZE
C
      DO 5 I=1,10
    5 CY(I)=0.
      Q(3)=QP(7)
      Q(4)=QP(8)
      NB=IFIX(B)
C
C     SUM MOMENT FOR ALL BLADES
C
      DO 10 I=1,NB
        J=I*2-1
        K=J+1
        Q(1)=QP(J)
        Q(2)=QP(K)
        PSIA=PSI+FLOAT(I-1)*2.*PI/B
        SPSIA=SIN(PSIA)
        CPSIA=COS(PSIA)
        S2=SPSIA*SPSIA
        CY(1)=CY(1)+S2
        CY(2)=CY(2)+Q(1)
        CY(3)=CY(3)+Q(2)
        CY(4)=CY(4)+Q(1)*Q(2)*S2
```

```
          CY(5)=CY(5)+Q(1)*Q(2)*(1.-S2)
          CY(6)=CY(6)+Q(1)*SPSIA
          CY(7)=CY(7)+Q(1)*Q(2)*CPSIA
          CY(8)=CY(8)+Q(2)*CPSIA
          PITNOW=PITCH(I)
          CALL YAM(YMX,FM,PSIA,ALF,CLL,CDD,AAA)
          IF (I .EQ. 1) THEN
            AMFP=FM-(B2*REVS*REVS*PC+32.174*BM*PC*RB*COS(PSIA))
          ENDIF
          CY(9)=CY(9)+YMX
        CY(10)=CY(10)+( Q(1)-PC )*SPSIA
   10   CONTINUE
C
        CALL YAWN(YMN)
        YAWM=CY(9)
        CY(9)=(CY(9)+YMN)/(REVS*REVS)
        IF(ABS(Q(4)).LE.1.E-08) THEN
              SGN=0.
        ELSE
          SGN=SIGN(1.,Q(4))*AF/(REVS*REVS)
        END IF
        IF(IYAC .EQ. 1) THEN
          YM1=CY(9) - SGN - 2.*Q(4)
      *     *(BM*RB*(SL*CY(3)-RH*CY(4))+(B3-B1)*CY(5)+AV/(2.*REVS))
      *     + (B1+B2-B3-BM*RB*SL*PC)*CY(6)
      *     - 2.*BM*RB*SL*CY(7) - (B1+B2-B3)*CY(8)
          CI=YI+B*(BM*SL*SL+B1)+(B2-B1+BM*(RH*RH+2.*RB*RH))*CY(1)+
      *    2.*BM*RB*SL*CY(2)
          FY4=YM1/CI
        ELSE
          FY4=0.
        ENDIF
C
C       YM1 IS THE YAW ACCELERATION
C
          YM1=FY4*REVS*REVS
C
        RETURN
        END
C
C
C**********************************************************************
C             YAWN CALCULATES THE YAW MOMENT DUE TO AERODYNAMIC
C             FORCE ACTING ON THE NACELLE.
C
C**********************************************************************
C
        SUBROUTINE YAWN(YMN)
        COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
      *       V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
      *       B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
      *       VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
        COMMON/DENSE/ RHO
        DP=.5*RHO*V*V
        AN=DELTA+Q(3)
        YMN=DP*(C1N*SIN(2.*AN)*COS(.5*AN)+C2N*(SIN(AN)**2))
        RETURN
        END
```

```
C
C
**********************************************************************
C          YAM CALCULATES THE YAW MOMENT AND FLAP MOMENT ON THE BLADE
C          AT CERTAIN AZIMUTH ANGLE PSI.
C
**********************************************************************
C
      SUBROUTINE YAM(YMX,FM,PSI,ALF,CLL,CDD,AAA)
      COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
     *      V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *      B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *      VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
      COMMON/DENSE/ RHO
      YMX=0.
      FM=0.
C
      DO 10 J=1,10
      X=(J*.1-.05)*R-RH
      IF(X.LE.0.) GO TO 10
      CALL VEL(VY,VZ,X,PSI)
      CALL VIND(A,J,X,PSI,VY,VZ,VN,VT)
      CALL VNMOD(VN,VY,VZ,X,PSI,A,AXY)
      W2=VN*VN+VT*VT
      PHI=ATAN2(VN,VT)
      ALPHA=PHI-THETA(J)-PITNOW/DEG
C
      IF( ALPHA .GT. PI ) ALPHA=ALPHA-PI
      IF( ALPHA .LT. -PI/2. ) ALPHA=ALPHA+PI
C
      CLA=GETCL(ALPHA)
          IF( CLA .LT. -900. ) CLA=FPL(ALPHA)
      CDA=GETCD(ALPHA)
          IF( CDA .LT. -900. ) CDA=FPD(ALPHA)
C
C     SAVE VALUES AT APPROPRIATE STATION
C
      IF ( J .EQ. NS )  THEN
        ALF=ALPHA*DEG
        CLL=CLA
        CDD=CDA
        AAA=AXY
      ENDIF
      SIPSI=SIN(PSI)
      CSPSI=COS(PSI)
      AK=CLA*COS(PHI)+CDA*SIN(PHI)
      BK=CLA*SIN(PHI)-CDA*COS(PHI)
      CK=X+RH*COS(Q(1))+SL*SIN(Q(1))
      DK=X*SIN(Q(1))+SL
      YMX=YMX+(CK*AK*SIPSI-DK*BK*CSPSI)*W2*(.1*R*C(J))
      FM=FM+X*AK*W2*(.1*R*C(J))
C
   10 CONTINUE
C
      YMX=.5*RHO*YMX*REVS*REVS
      FM=FM*0.5*RHO*REVS*REVS
      RETURN
      END
```

```
C
C       ************************************************************
C           VEL COMPUTES THE INERTIAL FRAME OF REFERENCE VELOCITY
C           COMPONENTS.
C       ************************************************************
C
      SUBROUTINE VEL(VY,VZ,XB,PSI)
      COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
     *      V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *      B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *      VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
C
C
      CPSI=COS(PSI)
      SPSI=SIN(PSI)
      CDELTA=COS(DELTA)
      SDELTA=SIN(DELTA)
      SQ1=SIN(Q(1))
      SQ3=SIN(Q(3))
      CQ1=COS(Q(1))
      CQ3=COS(Q(3))
C
      X=XB*CPSI*CQ1 + RH*CPSI
      Y=XB*(SPSI*CQ1*CQ3-SQ1*SQ3) + RH*SPSI*CQ3 - SL*SQ3
      Z=XB*(SPSI*CQ1*SQ3+SQ1*CQ3) + RH*SPSI*SQ3 + SL*CQ3
C
C     CHOOSE LINEAR WIND SHEAR OR POWER WIND SHEAR
C
      IF ( VLP .EQ. 1 )  THEN
          V1=V*( 1.+HSHR*( Y*CDELTA-Z*SDELTA )/(1.5*R)
     *           -VSHR*X/(1.5*R)  )
      ELSE
          V1=V* HSHR*( Y*CDELTA-Z*SDELTA )/(1.5*R)
     *           + V * (1.-X/HH)**(VSHR)
      ENDIF
C
      VY=V1*SDELTA
      VZ=V1*CDELTA
      RETURN
      END
C
C       ************************************************************
C           VNMOD COMPUTES THE NORMAL COMPONENTS
C           OF VELOCITY BY SKEWED WAKE CORRECTION.
C       ************************************************************
C
      SUBROUTINE VNMOD(VN,VY,VZ,XB,PSI,A,AXY)
      COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
     *      V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *      B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *      VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
      COMMON/DENSE/RHO
C
C     CALCULATE SKEWED WAKE FACTOR
C
      VYZ=VY*COS(Q(3))+VZ*SIN(Q(3))+Q(4)*SL*REVS
      SKEW=15.*PI/32.
      SKEW=SKEW*SQRT((1.-COS(Q(3)))/(1.+COS(Q(3))))
```

```
      IF (VYZ .LE. 0.) THEN
           AXY=A*(1.-SKEW*XB/R*SIN(PSI))
      ELSE
           AXY=A*(1.+SKEW*XB/R*SIN(PSI))
      ENDIF
C
C     MODIFY FOR EFFECTS OF TOWER SHADOW
C
      VN=VN*(1.-AXY)/(1.-A)

      RETURN
      END
C
C
C     ***************************************************************
C           FUNCTION FPL CALCULATES PLATE LIFT COEFFICIENT
C             USING THE VITERNA METHOD
C     ***************************************************************
      FUNCTION  FPL(ANG)
      COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
     *      V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *      B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *      VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
C
      CE=.5*( C(10)+C(1) )
      AR=R/CE
      CDMAX=1.11 + 0.018*AR
      A1=CDMAX/2.
      ATN=AL(NLIFT)
      SSI=SIN(ATN)
      SCO=COS(ATN)
      A2=( CL(NLIFT) - CDMAX*SSI*SCO )*SSI/SCO/SCO
C
C     REFLECT CL CURVE ABOUT ALPHA=90 DEG
C
C
      IF( ANG.GT. AL(NLIFT) .AND. ANG .LE. PI/2. ) THEN
         SANG=SIN(ANG)
         COSANG=COS(ANG)
         FPL=2.*A1*SANG*COSANG + A2*COSANG*COSANG/SANG
      ENDIF
      IF( ANG .GT. PI/2. .AND.  ANG .LE. PI-AL(NLIFT) ) THEN
            ANG=PI-ANG
            SANG=SIN(ANG)
            COSANG=COS(ANG)
            FPL=2.*A1*SANG*COSANG + A2*COSANG*COSANG/SANG
            FPL=-FPL
      ENDIF
      IF( ANG .GT. PI-AL(NLIFT) .AND. ANG .LE. PI ) THEN
            ANG=PI-ANG
            FPL=-GETCL(ANG)
      ENDIF
      IF( ANG .GT. -AL(NLIFT).AND. ANG .LE. AL(1) ) THEN
            FPL=CL(1)+(ANG-AL(1))*(CL(NLIFT)+CL(1))/(AL(NLIFT)+AL(1))
      ENDIF
      IF( ANG .LE. -AL(NLIFT) .AND. ANG .GE. -PI/2. )  THEN
            ANG=-ANG
            SANG=SIN(ANG)
```

```
            COSANG=COS(ANG)
            FPL=2.*A1*SANG*COSANG + A2*COSANG*COSANG/SANG
            FPL=-FPL
      ENDIF
C
      RETURN
      END
C
C
C     ***************************************************************
C           FUNCTION FPD CALCULATES DRAG COEFFICIENT FOR HIGH
C           ANGLE OF ATTACK USING THE VITERNA METHOD.
C     ***************************************************************
C
      FUNCTION FPD(ANG)
      COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
     *      V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *      B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *      VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
C
      CE=.5*( C(10)+C(1) )
      AR=R/CE
      CDMAX=1.11 + 0.018*AR
      ATN=AD(NDRAG)
      SSI=SIN(ATN)**2
      SCO=COS(ATN)
      B22=( CD(NDRAG)-CDMAX*SSI )/SCO
C
C     REFLECT CD ABOUT ALPHA=90 DEG
C
      IF( ANG.GT. AD(NDRAG) .AND. ANG .LE. PI/2. ) THEN
            FPD=CDMAX*SIN(ANG)**2+B22*COS(ANG)
      ENDIF
      IF( ANG .GT. PI/2. .AND.  ANG .LE. PI-AD(NDRAG) ) THEN
            ANG=PI-ANG
            FPD=CDMAX*SIN(ANG)**2+B22*COS(ANG)
      ENDIF
      IF( ANG .GT. PI-AD(NDRAG) .AND. ANG .LE. PI ) THEN
            ANG=PI-ANG
            FPD=GETCD(ANG)
      ENDIF
      IF( ANG .GT. -AD(NDRAG).AND. ANG .LE. AD(1) ) THEN
            FPD=CD(1)+(-ANG+AD(1))*(CD(NDRAG)-CD(1))/(AD(NDRAG)+AD(1))
      ENDIF
      IF( ANG .LE. -AD(NDRAG) .AND. ANG .GE. -PI/2. )  THEN
            ANG=-ANG
            FPD=CDMAX*SIN(ANG)**2+B22*COS(ANG)
      ENDIF
C
      RETURN
      END
C
C
C     ***************************************************************
C         FUNCTION GETCL IS INTERPOLATION ROUTINE FOR AIRFOIL CL
C     ***************************************************************
C
      FUNCTION GETCL(ANG)
```

```
      COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
     *       V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *       B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *       VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
C
      GETCL = -999.
      IF( ANG .LT. AL(1) .OR. ANG .GT. AL(NLIFT) ) RETURN
      N=1
50    N=N+1
      IF( AL(N) .LT. ANG ) GO TO 50
      P=( ANG-AL(N-1) )/( AL(N)-AL(N-1) )
      GETCL = CL(N-1)+P*( CL(N)-CL(N-1) )
      RETURN
      END
C
C     ****************************************************************
C         FUNCTION GETCD IS INTERPOLATION ROUTINE FOR AIRFOIL CD
C     ****************************************************************
C
      FUNCTION GETCD(ANG)
      COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
     *       V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *       B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *       VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
C
      GETCD = -999.
      IF( ANG.LT.AD(1) .OR. ANG.GT.AD(NDRAG) ) RETURN
      N=1
50    N=N+1
      IF( AD(N) .LT. ANG ) GO TO 50
      P=( ANG-AD(N-1) )/( AD(N)-AD(N-1) )
      GETCD = CD(N-1)+P*( CD(N)-CD(N-1) )
      RETURN
      END
```

APPENDIX B

LISTING OF THE SDOF PROGRAM

```
C
C     ************************S-D-O-F*****************************
C       SDOF CALCULATES YAW ANGLE, YAW RATE AND YAW MOMENT
C       FOR A 2 OR 3-BLADED HAWT.  FILE SDOF.IPT CONTAINS THE
C       HAWT DATA AND STARTING CONDITIONS. FILE SDOFPLT.OPT IS
C       THE OUTPUT FILE.  ALL UNITS ARE IN FEET,SLUGS,
C       SECONDS,POUNDS FORCE AND DEGREES.
C
C       YAW MOMENT AND FLAP MOMENT VS. AZIMUTH FOR ONE SELECTED
C       REVOLUTION OF THE ROTOR ARE STORED IN FILE SDOF.OPT
C
C       WRITTEN BY CUI XUDONG AND C. HANSEN, UNIVERSITY OF UTAH
C       LAST MODIFIED 4/20/88
C
C     ************************************************************
C
C       LIST OF VARIABLES:
C
C       A1     =MOMENT OF INERTIA OF NACELLE
C       A2     =MOMENT OF INERTIA OF ROTOR SHAFT
C       A3     =MOMENT OF INERTIA OF HUB
C       BM     =BLADE MASS
C       FS     =BLADE HINGE SPRING STIFFNESS
C       B1     =BLADE PITCH MOMENT OF INERTIA
C       B2     =BLADE FLAP MOMENT OF INERTIA
C       B3     =BLADE LAG MOMENT OF INERTIA
C       R      =ROTOR RADIUS
C       RH     =BLADE HINGE OFFSET
C           HH     =HUB HEIGHT
C       RB     =DISTANCE FROM BLADE HINGE TO BLADE CENTER OF MASS
C       B      =NUMBER OF BLADES
C       PC     =PRECONING ANGLE
C       VB     =MEAN AXIAL WIND SPEED
C       RPM    =ROTOR ROTATIONAL SPEED (IN REVOLUTIONS/MINUTE)
C       HSHR   =HORIZONTAL WIND SHEAR
C       VSHR   =VERTICAL WIND SHEAR
C       VELDEF=TOWER SHADOW VELOCITY DEFICIT FRACTION
C       SL     =DISTANCE FROM YAW COLUMN TO ROTOR HUB
C       AV     =VISCOUS DAMPING COEFICIENT (FT-LBS-SECS/RADIAN)
C       AF     =DRY FRICTION MOMENT
C       C1N    =NACELLE AERODYNAMIC MOMENT COEFFICIENT
C       C2N    =NACELLE AERODYNAMIC MOMENT COEFFICIENT
C       AA     =BLADE GEOMETRIC ANGLE OF ATTACK VECTOR
C       CL     =BLADE AERODYNAMIC LIFT COEFFICIENTS
C       CD     =BLADE AERODYNAMIC DRAG COEFFICIENTS
C       THETA =BLADE TWIST ANGLES (AT EACH OF TEN BLADE STATIONS)
C       C      =BLADE CHORD LENGTHS (AT TEN BLADE STATIONS)
C       RHO    =AIR DENSITY
C       N      =NUMBER OF DATA POINTS TO BE COMPUTED
C       SECTOR=NUMBER OF SECTORS
C       PITCH =PITCH ANGLE OF THE BLADE
C           PITNOW=PITCH OF BLADE IN CURRENT CALCULATION
C       SKEW   =SKEWED WAKE FACTOR
C       Q      =INSTANTANEOUS VALUES OF iTH BLADE DEGREES-OF-FREEDOM:
C                 FLAP ANGLE, FLAP RATES, YAW ANGLE AND YAW RATE
C       AMFP   =NET FLAP MOMENT
C       YAWM   =YAW MOMENT
C           YM1    =YAW ACCELERATION
```

```
C           S4    =AERODYNAMIC FLAP MOMENT
C
C
C       ************************************************************
C                     MAIN  PROGRAM
C       ************************************************************
C
        COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
     *          V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *          B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *          VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
        COMMON/DENSE/ RHO
        DIMENSION QP(2),F(2,4),Q1(2),Q2(2),Q3(2),Q4(2)
        CHARACTER*80 TITLE
        EQUIVALENCE (F(1,1),Q1(1)),(F(1,2),Q2(1)),
     *           (F(1,3),Q3(1)),(F(1,4),Q4(1))
C
C
C       ************************************************************
C                OPEN INPUT AND OUTPUT FILES
C       ************************************************************
C
        OPEN (UNIT=10, FILE= 'SDOF.IPT',STATUS='OLD',IOSTAT=IERR)
        IF(IERR.NE.0)  THEN
                WRITE(*,*) 'ERROR OPENING SDOF.IPT'
                WRITE(*,*) 'IOSTAT=',IERR
                STOP
        ENDIF
C
        OPEN (UNIT=12, FILE= 'SDOFPLT.OPT',STATUS='NEW',IOSTAT=IERR)
        IF(IERR.NE.0)  THEN
                WRITE(*,*) 'ERROR OPENING SDOFPLT.OPT'
                WRITE(*,*) 'IOSTAT=',IERR
                STOP
        ENDIF
C
        OPEN (UNIT=14, FILE= 'SHARMON.IPT',STATUS='NEW',IOSTAT=IERR)
        IF(IERR.NE.0)  THEN
                WRITE(*,*) 'ERROR OPENING SDOFHARM.OPT'
                WRITE(*,*) 'IOSTAT=',IERR
                STOP
        ENDIF
C
        OPEN (UNIT=100, FILE='SDOF.OPT',STATUS='NEW',IOSTAT=IERR)
        IF(IERR.NE.0)  THEN
                WRITE(*,*) 'ERROR OPENING SDOF.OPT'
                WRITE(*,*) 'IOSTAT=',IERR
                STOP
        ENDIF
C
C
C       ************************************************************
C            FILE 11 IS WIND FILE. IT WAS USED TO STUDY A GUST OR
C                 SUDDEN CHANGE OF WIND DIRECTION.
C       ************************************************************
C       OPEN (UNIT=11, FILE= 'YAWDYN.WND',STATUS='OLD',IOSTAT=IERR)
C               IF(IERR.NE.0)  THEN
C              WRITE(*,*) 'ERROR OPENING YAWDYN.WND'
```

```
C               WRITE(*,*) 'IOSTAT=',IERR
C               STOP
C       ENDIF
C
C
C       ****************************************************************
C          READ THE INPUT FILE
C       ****************************************************************
C
        DELTA=0.
        READ(10,2100) TITLE
 2100   FORMAT(A)
        READ(10,*) A1,A2,A3,BM,FS
        YI=A1+A2+A3
        READ(10,*) B1, B2, B3
        READ(10,*) R,RB,RH,HH,B,PC
        NB=B
        READ(10,*) VB,RPM,HSHR,VSHR,VELDEF
                PI=4.*ATAN(1.)
               TWOPI=2.*PI
                DEG=180./PI
               REVS=RPM*PI/30.
        READ(10,*) SL,AV,AF,C1N,C2N
        READ(10,*) (PITCH(I),I=1,NB)
        READ(10,*) AX4,AX5,Q(3),Q(4)
C
C       AX4 AND AX5 ARE NOT USED IN SDOF
C
        READ(10,*) N
        READ(10,*) SECTOR,RHO
C
        WRITE(*,2100) TITLE
        WRITE(*,*) ' '
        WRITE(*,*) ' SELECTED DATA FROM ONE REVOLUTION OF THE'
        WRITE(*,*) ' ROTOR WILL BE PRINTED TO A DISC FILE'
        WRITE(*,*) ' ENTER THE REVOLUTION NUMBER YOU WISH TO SEE'
        READ(*,*)  MC
C
        WRITE(*,*) ' THE ANGLE OF ATTACK, CL AND CD FOR ONE'
        WRITE(*,*) ' BLADE ELEMENT WILL BE WRITTEN TO THE SAME'
        WRITE(*,*) ' FILE.  ENTER THE BLADE ELEMENT NUMBER YOU'
        WRITE(*,*) ' WISH TO SEE'
        READ(*,*)  NS
C
        WRITE(*,*) ' CHOOSE LINEAR OR POWER LAW VERTICAL WIND SHEAR '
        WRITE(*,*) ' ENTER 1 FOR LINEAR SHEAR'
        WRITE(*,*) ' ENTER 2 POWER LAW VERTICAL SHEAR'
        READ  (*,*)  VLP
C
        WRITE (*,*) ' CHOOSE FREE YAW OR FIXED YAW  '
        WRITE (*,*) ' ENTER 1 FOR FREE YAW. 0 FOR FIXED YAW  '
        READ (*,*)  IYAC
C
        WRITE(*,*) ' THE RESULTS OF EVERY Nth CALCULATION WILL'
        WRITE(*,*) ' BE WRITTEN TO THE SCREEN AND FILE SDOFPLT.OPT'
        WRITE(*,*) ' ENTER N ( >=1)'
        READ (*,*) IPRINT
C
```

```
      WRITE(100,2100) TITLE
      WRITE(100,*) '  WIND SPEED AT HUB (FT/SEC) =' ,VB
      WRITE(100,*) '  ROTOR SPEED (RPM) =' ,RPM
      WRITE(100,*) ' '
      WRITE(100,*) '  ROTOR RADIUS (FT) =' ,R
      WRITE(100,*) '  HUB RADIUS (FT) ='   ,RH
      WRITE(100,*) '  HUB HEIGHT (FT) ='   ,HH
      WRITE(100,*) '  PITCH ANGLES (DEG) =',(PITCH(I),I=1,NB)
      WRITE(100,*) '  BLADE CENTER OF GRAVITY (FT) =' ,RB
      WRITE(100,*) '  YAW AXIS-TO-HUB DISTANCE (FT) =',SL
      WRITE(100,*) '  NUMBER OF BLADES =' ,B
      WRITE(100,*) '  PRE-CONING ANGLE (DEG) =' ,PC
      WRITE(100,*) ' '
      WRITE(100,*) '  MASS OF BLADE (SLUG) =' ,BM
      WRITE(100,1006) B1, B2, B3
 1006 FORMAT('  BLADE MOMENTS OF INERTIA (SLUG*FT^2) =',3(F10.1,1X))
      WRITE(100,1007) A1, A2, A3
 1007 FORMAT('  NACELLE MOMENTS OF INERTIA (SLUG*FT^2) =',3(F10.1,1X))
      WRITE(100,*) '  BLADE STIFFNESS COEF. (LB-FT/RAD) =' ,FS
      WRITE(100,*) '  YAW AXIS FRICTION (FT-LB) = ',AF
      WRITE(100,*) '  YAW AXIS DAMPING (FT-LB-SEC) = ',AV
      WRITE(100,*) ' '
      WRITE(100,*) '  LINEAR HORIZONTAL WIND SHEAR'
      WRITE(100,*) '  SHEAR COEFFICIENT = ' , HSHR
      IF (VLP .EQ. 1)  THEN
         WRITE(100,*) '  LINEAR VERTICAL WIND SHEAR'
         WRITE(100,*) '  SHEAR COEFFICIENT = ' , VSHR
      ELSE
         WRITE(100,*) '  POWER LAW VERTICAL WIND SHEAR '
         WRITE(100,*) '  POWER LAW EXPONENT = ' , VSHR
      ENDIF
      WRITE(100,*) ' '
C
C      READ SECTION CL AND CD CHARACTERISTICS
C      NLIFT=NUMBER OF POINTS DEFINING CL VS. ALPHA CURVE
C      NDRAG=NUMBER OF POINTS DEFINING CD VS. ALPHA CURVE
C
        WRITE (100,1002)
1002    FORMAT(/,'  AIRFOIL CHARACTERISTICS:',/,' ANGLE OF ATTACK(DEG)
     *         LIFT COEF.         ')
        READ(10,*) NLIFT,NDRAG
        DO 10 I=1,NLIFT
                READ(10,*) AL(I),CL(I)
                WRITE(100,1003) AL(I),CL(I)
                AL(I)=AL(I)/DEG
  10    CONTINUE
C
1003    FORMAT(4X, F10.4, 8X, F10.4 )
        WRITE(100,*) ' '
        WRITE(100,1012)
1012    FORMAT(/,'  AIRFOIL CHARACTERISTICS:',/,' ANGLE OF ATTACK(DEG)
     *         DRAG COEF.         ')
        DO 12 I=1,NDRAG
                READ(10,*) AD(I),CD(I)
                WRITE(100,1013) AD(I),CD(I)
                AD(I)=AD(I)/DEG
12      CONTINUE
1013    FORMAT(4X, F10.4, 8X, F10.4 )
```

```
      WRITE(100,*) ' '
C
C     READ TWIST ANGLE AND CHORD AT EACH SECTION
C
      WRITE(100,1005)
      WRITE(100,*) ' '
      DO 20 I=1,10
            READ(10,*) THETA(I),C(I)
            WRITE(100,1004) THETA(I),C(I)
            THETA(I)=THETA(I)/DEG
 20   CONTINUE
1005  FORMAT(/' TWIST ANGLE(DEG)   CHORD(FT)  ')
1004  FORMAT(1X,F10.4 ,4X , F7.4 )
      WRITE(100,*) ' '
C
      WRITE(100,*) '  INITIAL YAW ANGLE (DEG) = ' , Q(3)
      WRITE(100,*) ' '
      WRITE(14,4000) IFIX(SECTOR)
4000  FORMAT(I5)
C
C     READ(11,*) VENT,V,DEL
C     DELTA=DEL/DEG
C     FILE 11 IS FOR STUDY GUST.
C
C     ***************************************************************
C          SET UP THE INITIAL VALUES.
C     ***************************************************************
C
      PC=PC/DEG
      Q(3)=Q(3)/DEG
      Q(4)=Q(4)/REVS/DEG
      QP(1)=Q(3)
      QP(2)=Q(4)
      V=VB
      Q(1)=PC
      Q(2)=0.
      PSI=0.
      H=2.*PI/SECTOR
C
C      ************************************************************
C          START RUNGE-KUTTA CALCULATION
C      ************************************************************
C
      CALL FK(PSI,QP,Q1,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
      CALL RK(PSI,H,QP,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
            TIME=PSI/REVS
            REM=AMOD(PSI,TWOPI)
            REMM=REM*DEG
            YE=(QP(1))*DEG
            YR=QP(2)*DEG*REVS
      WRITE (*,2000) TIME,REMM,YE,YAWM
2000  FORMAT(1X,'T= ',F5.1,'     PSI= ',F5.1,'     YE= ',F5.1,
     *           '     YM= ',E12.4)
      WRITE(12,130) TIME,CHAR(9),YE,CHAR(9),YR,CHAR(9),YAWM
C
C     SECOND R-K PASS
C
      CALL FK(PSI,QP,Q2,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
```

```
      CALL RK(PSI,H,QP,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
               TIME=PSI/REVS
               REM=AMOD(PSI,TWOPI)
               REMM=REM*DEG
               YE=(QP(1))*DEG
               YR=QP(2)*DEG*REVS
      WRITE (*,2000) TIME,REMM,YE,YAWM
      WRITE(12,130) TIME,CHAR(9),YE,CHAR(9),YR,CHAR(9),YAWM
C
C     THIRD R-K PASS
C
      CALL FK(PSI,QP,Q3,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
      CALL RK(PSI,H,QP,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
               TIME=PSI/REVS
               REM=AMOD(PSI,TWOPI)
               REMM=REM*DEG
               YE=(QP(1))*DEG
               YR=QP(2)*DEG*REVS
      WRITE (*,2000) TIME,REMM,YE,YAWM
      WRITE(12,130) TIME,CHAR(9),YE,CHAR(9),YR,CHAR(9),YAWM
C
C     SWITCH TO PREDICTOR-CORRECTOR CALCULATION
C
      CALL FK(PSI,QP,Q4,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
      NN=N-4
      WRITE(100,*) '  FLAP MOMENT INCLUDES AERODYNAMIC,
     *  GRAVITY AND CENTRIFUGAL MOMENTS. '
      WRITE(100,1000) MC,NS
 1000 FORMAT('   DATA FOR CYCLE NUMBER',I3,', BLADE ELEMENT ',I2,
     *      //'   PSI       YAWMOMENT
     *  FLAPMOMENT     ALPHA      CL         CD         A'/'   DEG
     *     FT-LB           FT-LB           DEG                 ')
      WRITE(100,*) ' '
C
C     LOOP THROUGH PREDICTOR CORRECTOR
C
      DO 50 I=1,NN
      CALL AB(PSI,H,QP,F,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
      TIME=TIME+H/REVS
      REM=AMOD(PSI,TWOPI)
      REMM=REM*DEG
      PM1=MC*TWOPI
      PM2=(MC+1)*TWOPI
      IF( PSI.LE.PM2 .AND. PSI .GE. PM1 ) THEN
         WRITE (100,1001) REMM, YAWM, AMFP,ALF,CLL,CDD,AAA
         WRITE (14,4010) REMM, YAWM, AMFP
 4010 FORMAT(3E15.6)
      ENDIF
C
 1001 FORMAT( 1X,F7.2 ,2(4X,E10.4),2X,F8.4,2X,F7.4,2X,
     *                F7.4,2X,F7.4)
      YE=(QP(1))*DEG
      YR=QP(2)*DEG*REVS
      IF(I/IPRINT .EQ. FLOAT(I)/FLOAT(IPRINT) ) THEN
         WRITE (*,2000) TIME,REMM,YE,YAWM
         WRITE (12,130) TIME,CHAR(9),YE,CHAR(9),YR,CHAR(9),YAWM
      ENDIF
 130  FORMAT( ' ',G10.4, 3(A1,E10.4) )
```

```
  50    CONTINUE
C
       CLOSE(10)
       CLOSE(12)
       CLOSE(14)
       CLOSE(100)
       STOP
       END
C
C      ************************************************************
C               SUBROUTINES
C      ************************************************************
C      AB SOLVES THE FIRST ORDER EQUATION USING ADAMS-BASHFORTH
C      PREDICTOR-CORRECTOR SCHEME.
C      ************************************************************
C
       SUBROUTINE AB(PSI,H,QP,F,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
       COMMON/YAW/ AL(30),AD(30),CL(30),CD(30),THETA(10),C(10),
     *          V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *          B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *         VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
       COMMON/DENSE/ RHO
       DIMENSION QP(2),F(2,4),PK(2),CK(2),FK1(2)
       DO 10 I=1,2
          PK(I)=QP(I)+(55.*F(I,4)-59.*F(I,3)+37.*F(I,2)-9.*
     *          F(I,1))*H/24.
  10   CONTINUE
C
C      FILE 11 IS FOR STUDY GUST.
C      READ(11,*) vent,V,DEL
C      DELTA=DEL/DEG
C
       PSI=PSI+H
       CALL FK(PSI,PK,FK1,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
       DO 20 I=1,2
          CK(I)=QP(I)+(9.*FK1(I) +19.*F(I,4)-5.*F(I,3)
     *          +F(I,2))*H/24.
          QP(I)=(251.*CK(I)+19.*PK(I))/270.
          F(I,1)=F(I,2)
          F(I,2)=F(I,3)
          F(I,3)=F(I,4)
          F(I,4)=FK1(I)
  20   CONTINUE
       RETURN
       END
C
C      ************************************************************
C           RK SOLVES THE EQUATION USING RUNGE-KUTTA METHOD.
C      ************************************************************
C
       SUBROUTINE RK(PSI,H,QP,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
       COMMON/YAW/ AL(30),AD(30),CL(30),CD(30),THETA(10),C(10),
     *          V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *          B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *         VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
       COMMON/DENSE/ RHO
       DIMENSION QP(2),PK(2),QK(2),RR(2),SK(2),TK(2)
```

```
      CALL FK(PSI,QP,TK,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
      DO 10 I=1,2
              PK(I)=TK(I)*H
10            TK(I)=QP(I)+PK(I)/2.
C
C     FILE 11 IS FOR STUDY GUST.
C     READ(11,*)vent,W,GAMMA
C     GAMMA=GAMMA/DEG
C     V=V+(W-V)/2.
C     DELTA=DELTA+(GAMMA-DELTA)/2.
C
      PSI=PSI+H/2.
      CALL FK(PSI,TK,QK,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
      DO 20 I=1,2
              QK(I)=QK(I)*H
              TK(I)=QP(I)+QK(I)/2.
20    CONTINUE
      CALL FK(PSI,TK,RR,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
      DO 30 I=1,2
              RR(I)=RR(I)*H
              TK(I)=QP(I)+RR(I)
30    CONTINUE
C
C     W AND GAMMA ARE GUST VELOCITY AND DIRECTION
C     V=W
C     DELTA=GAMMA
C
      PSI=PSI+H/2.
      CALL FK(PSI,TK,SK,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
      DO 40 I=1,2
              SK(I)=SK(I)*H
40            QP(I)=QP(I)+(PK(I)+2.*(QK(I)+RR(I))+SK(I))/6.
      RETURN
      END
C
C     ************************************************************
C         VIND CALCULATES THE INDUCED VELOCITY FACTOR FOR EACH
C         ANNULAR SEGMENT AND SPECIFIED AZIMUTH ANGLE PSI.
C     ************************************************************
C
      SUBROUTINE VIND(A,J,X,PSI,VY,VZ,VN,VT)
      COMMON/YAW/ AL(30),AD(30),CL(30),CD(30),THETA(10),C(10),
     *          V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *          B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *         VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
      TWOPI=2.*PI
      RPSI=AMOD( PSI, TWOPI )
      SOLID=B*C(J)/(PI*X*COS(PC))
      TS=0.
      IF(RPSI*DEG .LT. 15.  .OR.  RPSI*DEG .GT. 345.)
     *       TS=VELDEF*(1.+COS(12.*PSI))/2.
      AI=0.1
      DAI1=.1
      STEP=0.5
      ICOUNT=0
      VT=(X*COS(Q(1))+RH)-VY*COS(Q(3))*COS(PSI)/REVS
     *           -VZ*(1.-TS)*SIN(Q(3))*COS(PSI)/REVS-
     *                       (X*SIN(Q(1))+SL)*Q(4)*COS(PSI)
```

```
 10      ICOUNT=ICOUNT+1
         VN=( (VZ*(1.-TS)*COS(Q(3))-VY*SIN(Q(3)))*COS(Q(1))/REVS
     *          -SIN(PSI)*SIN(Q(1))*SIN(Q(3))*VZ*(1.-TS)/REVS
     *                  -VY*SIN(PSI)*SIN(Q(1))*COS(Q(3))/REVS
     *          -(X+RH*COS(Q(1))+SL*SIN(Q(1)))*Q(4)*SIN(PSI)
     *                  -X*Q(2) )*(1.-AI)
         PHI=ATAN2(VN,VT)
         W2=VN*VN+VT*VT
         ALPHA=PHI-THETA(J)-PITNOW/DEG
         IF( ALPHA .GT. PI ) ALPHA=ALPHA-PI
         IF( ALPHA .LT. -PI/2. ) ALPHA=ALPHA+PI
C
         CLA=GETCL(ALPHA)
          IF( CLA .LT. -900. ) CLA=FPL(ALPHA)
         CDA=GETCD(ALPHA)
          IF( CDA .LT. -900. ) CDA=FPD(ALPHA)
C
         CH=W2*REVS*REVS*SOLID*(CLA*COS(PHI)+CDA*SIN(PHI))/(2.*(VZ*VZ))
         IF(CH.LT.0.96) THEN
            AI1=(1.-SQRT(1.-CH))/2.
         ELSE
            AI1=.143+SQRT(.0203-.6427*(.889-CH))
         ENDIF
         DAI=AI1-AI
C
C        TEST FOR CONVERGENCE, STOP AFTER 100 ITERATIONS
C
         IF(ICOUNT .GT. 100) THEN
            WRITE(*,*) 'EXCESSIVE ITERATIONS TO FIND INDUCTION FACTOR'
            WRITE(*,*) 'ELEMENT= ',J,'    PSI= ',PSI*DEG
            WRITE(*,*) 'VN= ',VN,'    VT= ',VT
            WRITE(*,*) 'ALPHA= ',ALPHA*DEG,'    CL= ',CLA,'  CD= ',CDA
            WRITE(*,*) 'AI= ',AI,'     DAI= ',DAI
            PAUSE
            STOP
         ENDIF
C
         IF(ABS(DAI).LE.0.01) GO TO 14
         IF( IFIX(SIGN(1.,DAI)) .NE. IFIX(SIGN(1.,DAI1)))
     *       STEP=.5*STEP
         AI=AI+STEP*DAI
         DAI1=DAI
         GO TO 10
C
   14    A=AI
C
         RETURN
         END
C
C        ***************************************************************
C            FK EVALUATES THE FIRST DERIVATIVES OF THE COMPONENTS
C            OF VECTOR QR AND RETURNS THE VALUES IN VECTOR  QS.
C        ***************************************************************
C
         SUBROUTINE FK(PSI,QR,QS,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
         COMMON/YAW/ AL(30),AD(30),CL(30),CD(30),THETA(10),C(10),
     *           V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *           B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
```

```
     *           VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
          COMMON/DENSE/ RHO
          DIMENSION QR(2),QS(2)
          Q(3)=QR(1)
          Q(4)=QR(2)
          QS(1)=QR(2)
          CALL F4(FY4,PSI,QR,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
          QS(2)=FY4
          RETURN
          END
C
C         ***************************************************************
C            F4 COMPUTES THE YAW ACCELERATION FUNCTION AND YAW MOMENT
C         ***************************************************************
C

          SUBROUTINE F4(FY4,PSI,QP,YM1,YAWM,AMFP,ALF,CLL,CDD,AAA)
          COMMON/YAW/ AL(30),AD(30),CL(30),CD(30),THETA(10),C(10),
     *           V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *           B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *           VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
          COMMON/DENSE/ RHO
          DIMENSION QP(2)
C
          Q(3)=QP(1)
          Q(4)=QP(2)
          SUM=0.0
          NB=IFIX(B)
          IF (NB .EQ. 2) THEN
             C2=2.*COS(PSI)**2
             S2=2.*SIN(PSI)**2
          ELSE
             C2=B/2.
             S2=B/2.
          ENDIF
C
C         SUM MOMENT FOR ALL BLADES
C
          DO 10 I=1,NB
            PSIA=PSI+FLOAT(I-1)*2.*PI/B
            PITNOW=PITCH(I)
            CALL YAM(YMX,S4,PSIA,ALF,CLL,CDD,AAA)
          IF (I .EQ. 1) THEN
            AMFP=S4-(B2*REVS*REVS*PC+32.174*BM*PC*RB*COS(PSIA))
          ENDIF
          SUM=SUM+YMX
   10     CONTINUE
C
          CALL YAM(YMX,S4,PSI,ALF,CLL,CDD,AAA)
          YAWM=SUM
          CALL YAWN(YMN)
          IF(ABS(Q(4)).LE.1.E-08) THEN
                  SGN=0.
          ELSE
                  SGN=Q(4)/ABS(Q(4))
          END IF
          IF(IYAC .EQ. 1) THEN
             C1=(SUM+YMN-AF*SGN)/(REVS*REVS)-AV*Q(4)/REVS
```

```
            C2=YI+B*BM*SL*SL+2.*BM*B*RB*SL*PC+
     *      (B2-B1+BM*RH*RH+2.*BM*RB*RH)*S2+(B3*PC*PC)*C2+B*B1
            FY4=C1/C2
         ELSE
            FY4=0.
         ENDIF
C
C        YM1 IS THE YAW ACCELERATION
C
         YM1=FY4
         RETURN
         END
C
C        ************************************************************
C            YAMN CALCULATES THE YAW MOMENT DUE TO AERODYNAMIC FORCE
C             ACTING ON THE NACELLE.
C        ************************************************************
C
         SUBROUTINE YAWN(YMN)
         COMMON/YAW/ AL(30),AD(30),CL(30),CD(30),THETA(10),C(10),
     *           V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *           B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *           VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
         COMMON/DENSE/ RHO
         DP=.5*RHO*V*V
         AN=DELTA+Q(3)
         YMN=DP*(C1N*SIN(2.*AN)*COS(.5*AN)+C2N*(SIN(AN)**2))
         RETURN
         END
C
C        ************************************************************
C           YAM CALCULATES THE YAW MOMENT ON THE BLADE AT AZIMUTH
C               ANGLE PSI.
C        ************************************************************
C
         SUBROUTINE YAM(YMX,S4,PSI,ALF,CLL,CDD,AAA)
                  COMMON/YAW/
AL(30),AD(30),CL(30),CD(30),THETA(10),C(10),
     *           V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *           B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *          VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
                  COMMON/DENSE/ RHO
         YMX=0.
         S4=0.
C
C        SUM MOMENTS FOR 10 BLADE ELEMENTS
C
         DO 10 J=1,10
         X=(J*.1-.05)*R-RH
         IF(X.LE.0.) GO TO 10
         CALL VEL(VY,VZ,X,PSI)
         CALL VIND(A,J,X,PSI,VY,VZ,VN,VT)
         CALL VNMOD(VN,VY,VZ,X,PSI,A,AXY)
         W2=VN*VN+VT*VT
         PHI=ATAN2(VN,VT)
         ALPHA=PHI-THETA(J)-PITNOW/DEG
C
C        GET AIRFOIL CL AND CD
```

```
C
      IF( ALPHA .GT. PI ) ALPHA=ALPHA-PI
      IF( ALPHA .LT. -PI/2. ) ALPHA=ALPHA+PI

      CLA=GETCL(ALPHA)
        IF( CLA .LT. -900. ) CLA=FPL(ALPHA)
      CDA=GETCD(ALPHA)
        IF( CDA .LT. -900. ) CDA=FPD(ALPHA)
C
C     SAVE APPROPRIATE AIRFOIL VALUES FOR PRINTING
C
      IF ( J .EQ. NS )  THEN
        ALF=ALPHA*DEG
        CLL=CLA
        CDD=CDA
        AAA=AXY
      ENDIF
C
      AK = CLA*COS(PHI) + CDA*SIN(PHI)
      BK = CLA*SIN(PHI) - CDA*COS(PHI)
      CK = X + RH*COS(Q(1)) + SL*SIN(Q(1))
      DK = X*SIN(Q(1)) + SL
      YMX = YMX + (CK*AK*SIN(PSI) - DK*BK*COS(PSI)) *W2*C(J)
      S4 = S4 + X*AK*W2*C(J)
C
   10 CONTINUE
C
      YMX = YMX*.05*R*RHO*REVS*REVS
      S4 = S4*.05*RHO*R*REVS*REVS
C
      RETURN
      END
C
C     ************************************************************
C         VEL COMPUTES THE INERTIAL FRAME OF REFERENCE VELOCITY
C             COMPONENTS.
C     ************************************************************
C
      SUBROUTINE VEL(VY,VZ,XB,PSI)
      COMMON/YAW/ AL(30),AD(30),CL(30),CD(30),THETA(10),C(10),
     *          V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *          B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *          VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
C
      X=( XB*COS(Q(1)) + RH )*COS(PSI)
      Y=XB*(SIN(PSI)*COS(Q(1))*COS(Q(3))-SIN(Q(1))*SIN(Q(3)))
     *          +RH*SIN(PSI)*COS(Q(3))-SL*SIN(Q(3))
      Z=XB*(SIN(PSI)*COS(Q(1))*SIN(Q(3))+SIN(Q(1))*COS(Q(3)))
     *          +RH*SIN(PSI)*SIN(Q(3))+SL*COS(Q(3))
C
C     CHOOSE LINEAR WIND SHEAR OR POWER LAW WIND SHEAR
C
      IF ( VLP .EQ. 1 )  THEN
         V1=V*( 1.+HSHR*( Y*COS(DELTA)-Z*SIN(DELTA) )/(1.5*R)
     *          -VSHR*X/(1.5*R)  )
       ELSE
         V1=V* HSHR*( Y*COS(DELTA)-Z*SIN(DELTA) )/(1.5*R)
     *          + V * (1.-X/HH)**(VSHR)
```

```
      ENDIF
C
      VY=V1*SIN(DELTA)
      VZ=V1*COS(DELTA)
      RETURN
      END
C
C     ************************************************************
C         VNMOD COMPUTES THE NORMAL COMPONENT
C         USING SKEWED WAKE CORRECTION.
C     ************************************************************
C
      SUBROUTINE VNMOD(VN,VY,VZ,XB,PSI,A,AXY)
      COMMON/YAW/ AL(30),AD(30),CL(30),CD(30),THETA(10),C(10),
     *          V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *          B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *          VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
      COMMON/DENSE/RHO
C
C     CALCULATE SKEWED WAKE FACTOR, NOTE YAW RATE NOT
C           INCLUDED IN VYZ CALCULATION.
C
      VYZ=VY*COS(Q(3))+VZ*SIN(Q(3))
      SKEW=15.*PI/32.*SQRT((1.-COS(Q(3)))/(1.+COS(Q(3))))
      IF (VYZ .LE. 0.) THEN
         AXY=A*(1.-SKEW*XB/R*SIN(PSI))
      ELSE
         AXY=A*(1.+SKEW*XB/R*SIN(PSI))
      ENDIF
C
C     MODIFY THE NORMAL VELOCITY
C
      VN=VN*(1.-AXY)/(1.-A)
      RETURN
      END
C
C
C     ************************************************************
C           FUNCTION FPL CALCULATES PLATE LIFT COEFFICIENT
C           USING THE VITERNA METHOD
C     ************************************************************
      FUNCTION  FPL(ANG)
      COMMON/YAW/ AL(30),AD(30),CL(30),CD(30),THETA(10),C(10),
     *          V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *          B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *          VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
C
      CE=.5*( C(10)+C(1) )
      AR=R/CE
      CDMAX=1.11 + 0.018*AR
      A1=CDMAX/2.
      ATN=AL(NLIFT)
      SSI=SIN(ATN)
      SCO=COS(ATN)
      A2=( CL(NLIFT) - CDMAX*SSI*SCO )*SSI/SCO/SCO
C
C     REFLECT CL CURVE ABOUT ALPHA=90 DEG
C
```

```
C
      IF( ANG.GT. AL(NLIFT) .AND. ANG .LE. PI/2. ) THEN
      SANG=SIN(ANG)
      COSANG=COS(ANG)
      FPL=2.*A1*SANG*COSANG + A2*COSANG*COSANG/SANG
      ENDIF
      IF( ANG .GT. PI/2. .AND.  ANG .LE. PI-AL(NLIFT) ) THEN
            ANG=PI-ANG
      SANG=SIN(ANG)
      COSANG=COS(ANG)
      FPL=2.*A1*SANG*COSANG + A2*COSANG*COSANG/SANG
      FPL=-FPL
      ENDIF
      IF( ANG .GT. PI-AL(NLIFT) .AND. ANG .LE. PI ) THEN
            ANG=PI-ANG
            FPL=-GETCL(ANG)
      ENDIF
      IF( ANG .GT. -AL(NLIFT).AND. ANG .LE. AL(1) ) THEN
      FPL=CL(1)+(ANG-AL(1))*(CL(NLIFT)+CL(1))/(AL(NLIFT)+AL(1))
      ENDIF
      IF( ANG .LE. -AL(NLIFT) .AND. ANG .GE. -PI/2. )  THEN
            ANG=-ANG
      SANG=SIN(ANG)
      COSANG=COS(ANG)
      FPL=2.*A1*SANG*COSANG + A2*COSANG*COSANG/SANG
      FPL=-FPL
      ENDIF
C
      RETURN
      END
C
C
C     ****************************************************************
C           FUNCTION FPD CALCULATES DRAG COEFFICIENT FOR HIGH
C           ANGLE OF ATTACK USING THE VITERNA METHOD.
C     ****************************************************************
C
      FUNCTION FPD(ANG)
      COMMON/YAW/ AL(30),AD(30),CL(30),CD(30),THETA(10),C(10),
     *          V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *          B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *          VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
C
      CE=.5*( C(10)+C(1) )
      AR=R/CE
      CDMAX=1.11 + 0.018*AR
      ATN=AD(NDRAG)
      SSI=SIN(ATN)**2
      SCO=COS(ATN)
      B22=( CD(NDRAG)-CDMAX*SSI )/SCO
C
C     REFLECT CD ABOUT ALPHA=90 DEG
C
      IF( ANG.GT. AD(NDRAG) .AND. ANG .LE. PI/2. ) THEN
            FPD=CDMAX*SIN(ANG)**2+B22*COS(ANG)
      ENDIF
      IF( ANG .GT. PI/2. .AND.  ANG .LE. PI-AD(NDRAG) ) THEN
            ANG=PI-ANG
```

```
            FPD=CDMAX*SIN(ANG)**2+B22*COS(ANG)
        ENDIF
        IF( ANG .GT. PI-AD(NDRAG) .AND. ANG .LE. PI ) THEN
            ANG=PI-ANG
            FPD=GETCD(ANG)
        ENDIF
        IF( ANG .GT. -AD(NDRAG).AND. ANG .LE. AD(1) ) THEN
        FPD=CD(1)+(-ANG+AD(1))*(CD(NDRAG)-CD(1))/(AD(NDRAG)+AD(1))
        ENDIF
        IF( ANG .LE. -AD(NDRAG) .AND. ANG .GE. -PI/2. )  THEN
            ANG=-ANG
            FPD=CDMAX*SIN(ANG)**2+B22*COS(ANG)
        ENDIF

        RETURN
        END
C
C
C
        ****************************************************************
*
C           FUNCTION GETCL IS INTERPOLATION ROUTINE FOR AIRFOIL CL
C
        ****************************************************************
*
C
        FUNCTION GETCL(ANG)
        COMMON/YAW/ AL(30),AD(30),CL(30),CD(30),THETA(10),C(10),
     *          V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *          B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *          VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
C
        GETCL = -999.
        IF( ANG .LT. AL(1) .OR. ANG .GT. AL(NLIFT) ) RETURN
        N=1
50      N=N+1
        IF( AL(N) .LT. ANG ) GO TO 50
        P=( ANG-AL(N-1) )/( AL(N)-AL(N-1) )
        GETCL = CL(N-1)+P*( CL(N)-CL(N-1) )
        RETURN
        END
C
C
        ****************************************************************
*
C           FUNCTION GETCD IS INTERPOLATION ROUTINE FOR AIRFOIL CD
C
        ****************************************************************
*
C
        FUNCTION GETCD(ANG)
        COMMON/YAW/ AL(30),AD(30),CL(30),CD(30),THETA(10),C(10),
     *          V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *          B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *          VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
C
        GETCD = -999.
        IF( ANG.LT.AD(1) .OR. ANG.GT.AD(NDRAG) ) RETURN
```

```
      N=1
50    N=N+1
      IF( AD(N) .LT. ANG ) GO TO 50
      P=( ANG-AD(N-1) )/( AD(N)-AD(N-1) )
      GETCD = CD(N-1)+P*( CD(N)-CD(N-1) )
      RETURN
      END
```

APPENDIX C

LISTING OF THE UFLAP PROGRAM

```
C
C     ************************UFLAP*****************************
C           UFLAP CALCULATES FLAP ANGLE, FLAP RATE AND
C           FLAP MOMENT FOR EACH BLADE OF A 2 OR 3-BLADED HAWT.
C
C           FILE UFLAP.IPT CONTAINS THE HAWT DATA AND STARTING
C           CONDITIONS. FILE UFLAPPLT.OPT IS THE OUTPUT FILE.
C           ALL UNITS ARE FEET,SLUGS, SECONDS, POUNDS FORCE AND DEGREES.
C
C           FLAP ANGLE AND FLAP MOMENT VS. AZIMUTH FOR ONE SELECTED
C           REVOLUTION OF THE ROTOR ARE STORED IN FILE UFLAP.OPT
C
C           FILE YHARMON.IPT IS CREATED CONTAINING YAW AND FLAP MOMENT
C           RESULTS FOR ONE SELECTED REVOLUTION OF THE ROTOR.
C           THIS FILE IS USED BY PROGRAM YHARMON.FOR TO CALCULATE
C           HARMONIC CONTENT OF THE YAW AND FLAP LOADS.
C
C           WRITTEN BY CUI XUDONG, N. SIEDSCHLAG AND C. HANSEN,
C           UNIVERSITY OF UTAH,   LAST MODIFIED 6/88
C
C     ************************************************************
C
C       LIST OF VARIABLES:
C
C           A1     =MOMENT OF INERTIA OF NACELLE
C           A2     =MOMENT OF INERTIA OF ROTOR SHAFT
C           A3     =MOMENT OF INERTIA OF HUB
C           BM     =BLADE MASS
C           FS     =BLADE HINGE SPRING STIFFNESS
C           B1     =BLADE PITCH MOMENT OF INERTIA
C           B2     =BLADE FLAP MOMENT OF INERTIA
C           B3     =BLADE LAG MOMENT OF INERTIA
C           R      =ROTOR RADIUS
C           RH     =BLADE HINGE OFFSET
C           HH     =HUB HEIGHT
C           RB     =DISTANCE FROM BLADE HINGE TO BLADE CENTER OF MASS
C           B      =NUMBER OF BLADES
C           PC     =PRECONING ANGLE
C           VB     =MEAN AXIAL WIND SPEED
C           RPM    =ROTOR ROTATIONAL SPEED (IN REVOLUTIONS/MINUTE)
C           HSHR   =HORIZONTAL WIND SHEAR
C           VSHR   =VERTICAL WIND SHEAR
C           VELDEF=TOWER SHADOW VELOCITY DEFICIT FRACTION
C           SL     =DISTANCE FROM YAW COLUMN TO ROTOR HUB
C           AV     =VISCOUS DAMPING COEFICIENT (FT-LBS-SECS/RADIAN)
C           AF     =DRY FRICTION MOMENT
C           C1N    =NACELLE AERODYNAMIC MOMENT COEFFICIENT
C           C2N    =NACELLE AERODYNAMIC MOMENT COEFFICIENT
C           AA     =BLADE GEOMETRIC ANGLE OF ATTACK VECTOR
C           CL     =BLADE AERODYNAMIC LIFT COEFFICIENTS
C           CD     =BLADE AERODYNAMIC DRAG COEFFICIENTS
C           THETA =BLADE TWIST ANGLES (AT EACH OF TEN BLADE STATIONS)
C           C      =BLADE CHORD LENGTHS (AT TEN BLADE STATIONS)
C           RHO    =AIR DENSITY
C           N      =NUMBER OF DATA POINTS TO BE COMPUTED
C           SECTOR=NUMBER OF SECTORS
C           PITCH  =PITCH ANGLE OF THE BLADE
C           PITNOW=PITCH OF BLADE IN CURRENT CALCULATION
```

```
C           SKEW   =SKEWED WAKE FACTOR
C           Q      =INSTANTANEOUS VALUES OF iTH BLADE DEGREES-OF-FREEDOM:
C                  FLAP ANGLE, FLAP RATES, YAW ANGLE AND YAW RATE
C           MYAW   = YAW MOMENT
C           MFLAP  = FLAP MOMENT

C
C       ************************************************************
C                MAIN  PROGRAM
C       ************************************************************
C
      COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
     *      V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *      B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *      VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
      COMMON/DENSE/ RHO
      DIMENSION QP(2),F(2,4),Q1(2),Q2(2),Q3(2),Q4(2)
      CHARACTER*80 TITLE
      EQUIVALENCE (F(1,1),Q1(1)),(F(1,2),Q2(1)),
     *            (F(1,3),Q3(1)),(F(1,4),Q4(1))
C
C
C       ************************************************************
C                OPEN INPUT AND OUTPUT FILES
C       ************************************************************
C
      OPEN (UNIT=10, FILE= 'UFLAP.IPT',STATUS='OLD',IOSTAT=IERR)
      IF(IERR.NE.0)  THEN
            WRITE(*,*) 'ERROR OPENING UFLAP.IPT'
            WRITE(*,*) 'IOSTAT=',IERR
            STOP
      ENDIF
C
      OPEN (UNIT=12, FILE= 'UFLAPPLT.OPT',STATUS='NEW',IOSTAT=IERR)
      IF(IERR.NE.0)  THEN
            WRITE(*,*) 'ERROR OPENING UFLAPPLT.OPT'
            WRITE(*,*) 'IOSTAT=',IERR
            STOP
      ENDIF
C
      OPEN (UNIT=14, FILE= 'YHARMON.IPT',STATUS='NEW',IOSTAT=IERR)
      IF(IERR.NE.0)  THEN
            WRITE(*,*) 'ERROR OPENING YHARMON.IPT'
            WRITE(*,*) 'IOSTAT=',IERR
            STOP
      ENDIF
C
      OPEN (UNIT=100, FILE='UFLAP.OPT',STATUS='NEW',IOSTAT=IERR)
      IF(IERR.NE.0)  THEN
            WRITE(*,*) 'ERROR OPENING UFLAP.OPT'
            WRITE(*,*) 'IOSTAT=',IERR
            STOP
      ENDIF
C
C       ************************************************************
C          READ THE INPUT FILE
C       ************************************************************
C
```

```
      DELTA=0.
      READ(10,2100) TITLE
 2100 FORMAT(A)
      READ(10,*) A1,A2,A3,BM,FS
      YI=A1+A2+A3
      READ(10,*) B1, B2, B3
      READ(10,*) R,RB,RH,HH,B,PC
      NB=B
      READ(10,*) VB,RPM,HSHR,VSHR,VELDEF
        PI=4.*ATAN(1.)
        TWOPI=2.*PI
        DEG=180./PI
        REVS=RPM*PI/30.
      READ(10,*) SL,AV,AF,C1N,C2N
      READ(10,*) PITNOW
      READ(10,*) Q(1),Q(2),Q(3),Q(4)
      READ(10,*) N
      READ(10,*) SECTOR,RHO
C      WRITE(*,2100) TITLE
      WRITE(*,*) ' '
      WRITE(*,*) ' SELECTED DATA FROM ONE REVOLUTION OF THE'
      WRITE(*,*) ' ROTOR WILL BE PRINTED TO A DISC FILE'
      WRITE(*,*) ' ENTER THE REVOLUTION NUMBER YOU WISH TO SEE'
      READ(*,*)  MC
C
      WRITE(*,*) ' THE ANGLE OF ATTACK, CL AND CD FOR ONE'
      WRITE(*,*) ' BLADE ELEMENT WILL BE WRITTEN TO THE SAME'
      WRITE(*,*) ' FILE.  ENTER THE BLADE ELEMENT NUMBER YOU'
      WRITE(*,*) ' WISH TO SEE'
      READ(*,*)  NS
C
      WRITE(*,*) ' CHOOSE LINEAR OR POWER LAW VERTICAL WIND SHEAR '
      WRITE(*,*) ' ENTER 1 FOR LINEAR SHEAR'
      WRITE(*,*) ' ENTER 2 POWER LAW VERTICAL SHEAR'
      READ  (*,*)  VLP
C
C     RUN ONLY IN FIXED YAW
C
      IYAC=0
C
      WRITE(*,*) ' THE RESULTS OF EVERY Nth CALCULATION WILL'
      WRITE(*,*) ' BE WRITTEN TO THE SCREEN AND FILE UFLAPPLT.OPT'
      WRITE(*,*) ' ENTER N ( >=1)'
      READ (*,*) IPRINT
C
      WRITE(100,2100) TITLE
      WRITE(100,*) '  WIND SPEED AT HUB (FT/SEC) =' ,VB
      WRITE(100,*) '  ROTOR SPEED (RPM) =' ,RPM
      WRITE(100,*) ' '
      WRITE(100,*) '  ROTOR RADIUS (FT) =' ,R
      WRITE(100,*) '  HUB RADIUS (FT) ='   ,RH
      WRITE(100,*) '  HUB HEIGHT (FT) ='   ,HH
      WRITE(100,*) '  PITCH ANGLE (DEG) =', PITNOW
      WRITE(100,*) '  BLADE CENTER OF GRAVITY (FT) =' ,RB
      WRITE(100,*) '  YAW AXIS-TO-HUB DISTANCE (FT) =',SL
      WRITE(100,*) '  NUMBER OF BLADES =' ,B
      WRITE(100,*) '  PRE-CONING ANGLE (DEG) =' ,PC
      WRITE(100,*) ' '
```

```
      WRITE(100,*) '  MASS OF BLADE (SLUG) =' ,BM
      WRITE(100,1006) B1, B2, B3
 1006 FORMAT('  BLADE MOMENTS OF INERTIA (SLUG*FT^2) =',3(F10.1,1X))
      WRITE(100,1007) A1, A2, A3
 1007 FORMAT('  NACELLE MOMENTS OF INERTIA (SLUG*FT^2) =',3(F10.1,1X))
      WRITE(100,*) '  BLADE STIFFNESS COEF. (LB-FT/RAD) =' ,FS
      FREQ=SQRT(FS/B2)/TWOPI
      WRITE(100,*) '  BLADE NATURAL FREQUENCY (HZ) =',FREQ
      P=FREQ*60./RPM
      WRITE(100,*) '  BLADE NATURAL FREQUENCY (P#) =',P
      WRITE(100,*) '  YAW AXIS FRICTION (FT-LB) = ',AF
      WRITE(100,*) '  YAW AXIS DAMPING (FT-LB-SEC) = ',AV
      WRITE(100,*) ' '
      WRITE(100,*) '  LINEAR HORIZONTAL WIND SHEAR'
      WRITE(100,*) '  SHEAR COEFFICIENT = ' , HSHR
      IF (VLP .EQ. 1)  THEN
         WRITE(100,*) '  LINEAR VERTICAL WIND SHEAR'
         WRITE(100,*) '  SHEAR COEFFICIENT = ' , VSHR
      ELSE
         WRITE(100,*) '  POWER LAW VERTICAL WIND SHEAR '
         WRITE(100,*) '  POWER LAW EXPONENT = ' , VSHR
      ENDIF
      WRITE(100,*) ' '
C
C     READ SECTION CL AND CD CHARACTERISTICS
C     NLIFT=NUMBER OF POINTS DEFINING CL VS. ALPHA CURVE
C     NDRAG=NUMBER OF POINTS DEFINING CD VS. ALPHA CURVE
C
      WRITE (100,1002)
1002  FORMAT(/,'  AIRFOIL CHARACTERISTICS:',/,' ANGLE OF ATTACK(DEG)
     *         LIFT COEF.         ')
      READ(10,*) NLIFT,NDRAG
      DO 10 I=1,NLIFT
            READ(10,*) AL(I),CL(I)
            WRITE(100,1003) AL(I),CL(I)
            AL(I)=AL(I)/DEG
  10  CONTINUE
C
1003  FORMAT(4X, F10.4, 8X, F10.4 )
      WRITE(100,*) ' '
      WRITE(100,1012)
1012  FORMAT(/,'  AIRFOIL CHARACTERISTICS:',/,' ANGLE OF ATTACK(DEG)
     *          DRAG COEF.         ')
      DO 12 I=1,NDRAG
            READ(10,*) AD(I),CD(I)
            WRITE(100,1013) AD(I),CD(I)
            AD(I)=AD(I)/DEG
12    CONTINUE
1013  FORMAT(4X, F10.4, 8X, F10.4 )
      WRITE(100,*) ' '
C
C     READ TWIST ANGLE AND CHORD AT EACH SECTION
C
      WRITE(100,1005)
      WRITE(100,*) ' '
      DO 20 I=1,10
            READ(10,*) THETA(I),C(I)
            WRITE(100,1004) THETA(I),C(I)
```

```
            THETA(I)=THETA(I)/DEG
   20   CONTINUE
 1005   FORMAT(/' TWIST ANGLE(DEG)    CHORD(FT)   ')
 1004   FORMAT(1X,F10.4 ,4X , F7.4 )
        WRITE(100,*) ' '
C
        WRITE(100,*) '  INITIAL YAW ANGLE (DEG) = ' , Q(3)
        WRITE(100,*) ' '
C
C       ************************************************************
C            SET UP THE INITIAL VALUES
C       ************************************************************
C
        PC=PC/DEG
        V=VB
        PSI=0.
        H=2.*PI/SECTOR
        Q(1)=Q(1)/DEG+PC
        Q(2)=Q(2)/DEG/REVS
        Q(3)=Q(3)/DEG
        Q(4)=Q(4)/DEG/REVS
        NB=IFIX(B)
        QP(1)=Q(1)
        QP(2)=Q(2)
C
C
        WRITE(14,4000) IFIX(SECTOR)
 4000   FORMAT(I5)
        WRITE(*,*) 'RUNNING ',N,' POINTS'
        WRITE(*,*) 'WITH ',SECTOR,' POINTS PER REVOLUTION'
C
C       READ(11,*) VENT,V,DEL
C       DELTA=DEL/DEG
C       FILE 11 IS FOR STUDY GUST.
C
C
C          ************************************************************
C              START RUNGE-KUTTA CALCULATION
C          ************************************************************
C
        CALL FK(PSI,QP,Q1,AMFP,ALF,CLL,CDD,AAA)
        CALL RK(PSI,H,QP,AMFP,ALF,CLL,CDD,AAA)
              TIME=PSI/REVS
              REM=AMOD(PSI,TWOPI)
              REMM=REM*DEG
              FE=QP(1)*DEG
              FR=QP(2)*DEG*REVS
        WRITE (*,2000) TIME,FE,FR
 2000   FORMAT(1X,'T= ',F5.1,'  FE= ',E10.2, '  FR= ',E10.2  )
        WRITE(12,130) TIME,FE,FR,AMFP
C
C       SECOND R-K PASS
C
        CALL FK(PSI,QP,Q2,AMFP,ALF,CLL,CDD,AAA)
        CALL RK(PSI,H,QP,AMFP,ALF,CLL,CDD,AAA)
              TIME=PSI/REVS
              REM=AMOD(PSI,TWOPI)
              REMM=REM*DEG
```

```
            FE=QP(1)*DEG
            FR=QP(2)*DEG*REVS
      WRITE (*,2000) TIME,FE,FR
      WRITE(12,130) TIME,FE,FR,AMFP
C
C       THIRD R-K PASS
C
      CALL FK(PSI,QP,Q3,AMFP,ALF,CLL,CDD,AAA)
      CALL RK(PSI,H,QP,AMFP,ALF,CLL,CDD,AAA)
            TIME=PSI/REVS
            REM=AMOD(PSI,TWOPI)
            REMM=REM*DEG
            FE=QP(1)*DEG
            FR=QP(2)*DEG*REVS
      WRITE (*,2000) TIME,FE,FR
      WRITE(12,130) TIME,FE,FR,AMFP
C
C       SWITCH TO PREDICTOR-CORRECTOR CALCULATION
C
      CALL FK(PSI,QP,Q4,AMFP,ALF,CLL,CDD,AAA)
      NN=N-4
      WRITE(100,*) '  FLAP MOMENT INCLUDES AERODYNAMIC,
     *  GRAVITY AND CENTRIFUGAL MOMENTS. '
      WRITE(100,1000) MC,NS
 1000   FORMAT(/'   DATA FOR CYCLE NUMBER',I3,', BLADE ELEMENT ',I2,
     *       //'   PSI
     *  FLAPMOMENT    ALPHA      CL        CD        A'/'   DEG
     *     FT-LB          FT-LB          DEG               ')
      WRITE(100,*) ' '
C
C     LOOP THROUGH PREDICTOR CORRECTOR
C
      DO 50 I=1,NN
      CALL AB(PSI,H,QP,F,AMFP,ALF,CLL,CDD,AAA)
            TIME=TIME+H/REVS
            REM=AMOD(PSI,TWOPI)
            REMM=REM*DEG
            PM1=MC*TWOPI
            PM2=(MC+1)*TWOPI
            IF( PSI.LE.PM2 .AND. PSI .GE. PM1 ) THEN
                WRITE (100,1001) REMM, AMFP,ALF,CLL,CDD,AAA
                WRITE (14,4010) REMM, AMFP
 4010           FORMAT(2E15.6)
            ENDIF
C
 1001       FORMAT( 1X,F7.2 ,4X,E10.4,2X,F8.4,2X,F7.4,2X,
     *                F7.4,2X,F7.4)
            FE=QP(1)*DEG
            FR=QP(2)*DEG*REVS
            IF(I/IPRINT .EQ. FLOAT(I)/FLOAT(IPRINT) ) THEN
C                WRITE (*,2000) TIME,FE,FR
               WRITE( *,* ) TIME, FE
                 WRITE (12,130) TIME,FE,FR,AMFP
            ENDIF
 130        FORMAT( ' ',G10.4, 3(2X,E10.4) )
  50  CONTINUE
C
      CLOSE(10)
```

```
      CLOSE(12)
      CLOSE(14)
      CLOSE(100)
      STOP
      END
C
C     ************************************************************
C     SUBROUTINES
C     ************************************************************
C     AB SOLVES THE FIRST ORDER EQUATION USING ADAMS-BASHFORTH
C     PREDICTOR-CORRECTOR SCHEME.
C     ************************************************************
C
C     THE MATRIX F CONTAINS THE PREVIOUS FOUR SETS OF DERIVATIVE
C     FUNCTIONS EVALUATED BY SUBROUTINE FK.  THE DEGREES-OF-FREEDOM
C     CONTAINED IN VECTOR QP ARE ADVANCED
C     ONE STEP SIZE, H, AND THE MATRIX F IS SHIFTED TO STORE
C     THE NEW SET OF DERIVATIVE FUNCTION VALUES DURING EACH RUN OF AB.
C
      SUBROUTINE AB(PSI,H,QP,F,AMFP,ALF,CLL,CDD,AAA)
      COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
     *      V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *      B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *      VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
      COMMON/DENSE/ RHO
      DIMENSION QP(2),F(2,4),PK(2),CK(2),FK1(2)
      DO 10 I=1,2
            PK(I)=QP(I)+(55.*F(I,4)-59.*F(I,3)+37.*F(I,2)-
     *      9.*F(I,1))*H/24.
   10 CONTINUE
C                    .
C       FILE 11 IS FOR STUDY GUST.
C       READ(11,*) vent,V,DEL
C       DELTA=DEL/DEG
C
      PSI=PSI+H
      CALL FK(PSI,PK,FK1,AMFP,ALF,CLL,CDD,AAA)
      DO 20 I=1,2
            CK(I)=QP(I)+(9.*FK1(I) +19.*F(I,4)-5.*F(I,3)
     *         +F(I,2))*H/24.
            QP(I)=(251.*CK(I)+19.*PK(I))/270.
            F(I,1)=F(I,2)
            F(I,2)=F(I,3)
            F(I,3)=F(I,4)
            F(I,4)=FK1(I)
   20 CONTINUE
      RETURN
      END
C
C       ********************************************************
C            RK SOLVES THE EQUATION USING RUNGE-KUTTA METHOD.
C       ********************************************************
C
C     THE NEW VALUES ARE RETURNED IN QP. SUBROUTINE
C     FK IS USED TO EVALUATE THE DERIVATIVE FUNCTIONS.
C
      SUBROUTINE RK(PSI,H,QP,AMFP,ALF,CLL,CDD,AAA)
      COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
```

```
     *       V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *       B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *       VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
      COMMON/DENSE/ RHO
      DIMENSION QP(2),PK(2),QK(2),RR(2),SK(2),TK(2)
      CALL FK(PSI,QP,TK,AMFP,ALF,CLL,CDD,AAA)
      DO 10 I=1,2
            PK(I)=TK(I)*H
   10 TK(I)=QP(I)+PK(I)/2.
C
C
      PSI=PSI+H/2.
      CALL FK(PSI,TK,QK,AMFP,ALF,CLL,CDD,AAA)
      DO 20 I=1,2
                  QK(I)=QK(I)*H
                  TK(I)=QP(I)+QK(I)/2.
   20 CONTINUE
      CALL FK(PSI,TK,RR,AMFP,ALF,CLL,CDD,AAA)
      DO 30 I=1,2
            RR(I)=RR(I)*H
            TK(I)=QP(I)+RR(I)
   30 CONTINUE
C
C       W AND GAMMA ARE GUST VELOCITY AND DIRECTION
C       V=W
C       DELTA=GAMMA
C
      PSI=PSI+H/2.
      CALL FK(PSI,TK,SK,AMFP,ALF,CLL,CDD,AAA)
      DO 40 I=1,2
                  SK(I)=SK(I)*H
   40             QP(I)=QP(I)+(PK(I)+2.*(QK(I)+RR(I))+SK(I))/6.
      RETURN
      END
C
C       ************************************************************
C           VIND CALCULATES THE AXIAL INDUCTION FACTOR FOR EACH
C           ANNULAR SEGMENT AND SPECIFIED AZIMUTH ANGLE PSI.
C       ************************************************************
C
            SUBROUTINE VIND(A,J,X,PSI,VY,VZ,VN,VT)
      COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
     *       V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *       B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *       VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
      SOLID=B*C(J)/(PI*X*COS(Q(1)))
      TS=0.
      TWOPI=2.*PI
      PSIDEG=DEG*AMOD( PSI, TWOPI )
      IF(PSIDEG .LT. 15.  .OR.  PSIDEG .GT. 345.)
     *        TS=VELDEF*(1.+COS(12.*PSI))/2.
      AI=0.1
      DAI1=.1
      STEP=0.5
      ICOUNT=0
      VT=(X*COS(Q(1))+RH)-VY*COS(Q(3))*COS(PSI)/REVS
     *       -VZ*(1.-TS)*SIN(Q(3))*COS(PSI)/REVS-
     *       (X*SIN(Q(1))+SL)*Q(4)*COS(PSI)
```

```
 10   ICOUNT=ICOUNT+1
      VN=(VZ*(1.-TS)*COS(Q(3))-VY*SIN(Q(3)))*COS(Q(1))/REVS
     *          -SIN(PSI)*SIN(Q(1))*SIN(Q(3))*VZ*(1.-TS)/REVS
     *                  -VY*SIN(PSI)*SIN(Q(1))*COS(Q(3))/REVS
     *          -(X+RH*COS(Q(1))+SL*SIN(Q(1)))*Q(4)*SIN(PSI)
     *                  -X*Q(2)
      VN=VN*(1.-AI)
      PHI=ATAN2(VN,VT)
      W2=VN*VN+VT*VT
      ALPHA=PHI-THETA(J)-PITNOW/DEG
      IF( ALPHA .GT. PI ) ALPHA=ALPHA-PI
      IF( ALPHA .LT. -PI/2. ) ALPHA=ALPHA+PI
C
      CLA=GETCL(ALPHA)
         IF( CLA .LT. -900. ) CLA=FPL(ALPHA)
      CDA=GETCD(ALPHA)
         IF( CDA .LT. -900. ) CDA=FPD(ALPHA)
C
      CH=W2*REVS*REVS*SOLID*(CLA*COS(PHI)+CDA*SIN(PHI))/(2.*VZ**2)
        IF(CH.LT.0.96) THEN
           AI1=(1.-SQRT(1.-CH))/2.
        ELSE
           AI1=.143+SQRT(.0203-.6427*(.889-CH))
        ENDIF
      DAI=AI1-AI
C
C     TEST FOR CONVERGENCE, STOP AFTER 100 ITERATIONS
C
      IF(ICOUNT .GT. 100) THEN
           WRITE(*,*) 'EXCESSIVE ITERATIONS TO FIND INDUCTION FACTOR'
           WRITE(*,*) 'ELEMENT= ',J,'    PSI= ',PSI*DEG
           WRITE(*,*) 'VN= ',VN,'    VT= ',VT
           WRITE(*,*) 'ALPHA= ',ALPHA*DEG,'    CL= ',CLA,'  CD= ',CDA
           WRITE(*,*) 'AI= ',AI,'     DAI= ',DAI
           PAUSE 'ENTER CR TO CONTINUE'
           STOP
      ENDIF
C
      IF(ABS(DAI).LE.0.01) GO TO 14

C      IF( IFIX(SIGN(1.,DAI)) .NE. IFIX(SIGN(1.,DAI1)))
C     *      STEP=.5*STEP

      I1=IFIX( DAI1/ABS(DAI1) )
      I2=IFIX( DAI/ABS(DAI) )
      IF( I1 .NE. I2 )  STEP=.5*STEP
      AI=AI+STEP*DAI
      DAI1=DAI
      GO TO 10
C
  14  A=AI
C
      RETURN
      END
C
C       ************************************************************
C           FK EVALUATES THE FIRST DERIVATIVES OF THE COMPONENTS
C           OF VECTOR QR AND RETURNS THE VALUES IN VECTOR  QS.
```

```
C
C           SUBROUTINES F2 IS USED TO EVALUATE
C           THE DERIVATIVES OF THE VELOCITY COMPONENTS OF QR.
C       ************************************************************
C
      SUBROUTINE FK(PSI,QR,QS,AMFP,ALF,CLL,CDD,AAA)
      COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
     *      V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *      B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *      VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
      COMMON/DENSE/ RHO
      DIMENSION QR(2),QS(2)
      NB=IFIX(B)
C
         Q(1)=QR(1)
         Q(2)=QR(2)
         QS(1)=QR(2)
         CALL F2(F21,PSI,AMFP,ALF,CLL,CDD,AAA)
         QS(2)=F21
C
      RETURN
      END
C
C
*********************************************************************
C
C     F2 COMPUTES THE FLAP ACCELERATION FUNCTION AND THE FLAPPING MOMENT
C     FOR THE HAWT BLADE AT AZIMUTH ANGLE PSI.
C
C
*********************************************************************
C
      SUBROUTINE F2(F21,PSI,FM,ALF,CLL,CDD,AAA)
      COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
     *      V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *      B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *      VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
      COMMON/DENSE/ RHO
C
      CPSI=COS(PSI)
      SPSI=SIN(PSI)
      A1=BM*RB*RH/B2
      D1=B2*REVS*REVS
      A2=A1+(B3-B1)/B2+32.174*BM*RB*CPSI/D1
      A3=A1*SPSI*SPSI-(B3-B1)*CPSI*CPSI/B2
      A4=FS/D1
      F21=A4*PC - Q(4)*(1.+(B3-B1)/B2+2*A1)*CPSI
     *    - (A2+A4)*Q(1)
     *    - Q(4)*Q(4)*(A3*Q(1)-BM*RB*SL/B2)
      CALL YAM( FM,PSI,ALF,CLL,CDD,AAA,XX,YY,VYY,VZZ,AXX )
      WRITE(*,1000) XX,YY,VYY,VZZ,AXX
 1000 FORMAT(5(F12.8,1X))
      F21=F21+FM/D1
      RETURN
      END
C
C
*********************************************************************
```

```
C          YAM CALCULATES THE YAW MOMENT AND FLAP MOMENT ON THE BLADE
C          AT CERTAIN AZIMUTH ANGLE PSI.
C
*********************************************************************
C
      SUBROUTINE YAM(FM,PSI,ALFI,CLLI,CDDI,AAAI,XX,YY,VYY,VZZ,AXX)
      COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
     *      V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *      B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *      VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
      COMMON/DENSE/ RHO
      YMX=0.
      FM=0.
C
      DO 10 J=1,10
      X=(J*.1-.05)*R-RH
      IF(X.LE.0.) GO TO 10
      CALL VEL(VY,VZ,X,PSI)
      CALL VIND(A,J,X,PSI,VY,VZ,VN,VT)
      CALL VNMOD(VN,X,PSI,A,AXY,VY,VZ)
      W2=VN*VN+VT*VT
      PHI=ATAN2(VN,VT)
      ALPHA=PHI-THETA(J)-PITNOW/DEG
C
      IF( ALPHA .GT. PI ) ALPHA=ALPHA-PI
      IF( ALPHA .LT. -PI/2. ) ALPHA=ALPHA+PI
C
      CLA=GETCL(ALPHA)
          IF( CLA .LT. -900. ) CLA=FPL(ALPHA)
      CDA=GETCD(ALPHA)
          IF( CDA .LT. -900. ) CDA=FPD(ALPHA)
C
C     SAVE VALUES AT APPROPRIATE STATION
C
      IF ( J .EQ. NS )  THEN
        ALFI=ALPHA*DEG
        CLLI=CLA
        CDDI=CDA
        AAAI=AXY
      XX=VN
      YY=VT
      VYY=VY
      VZZ=VZ
      AXX=A
      ENDIF
      SIPSI=SIN(PSI)
      CSPSI=COS(PSI)
      AK=CLA*COS(PHI)+CDA*SIN(PHI)
      BK=CLA*SIN(PHI)-CDA*COS(PHI)
      CK=X+RH*COS(Q(1))+SL*SIN(Q(1))
      DK=X*SIN(Q(1))+SL
      YMX=YMX+(CK*AK*SIPSI-DK*BK*CSPSI)*W2*(.1*R*C(J))

C      FM=FM+X*AK*W2*(.1*R*C(J))
C
      DFM=X*C(J)*AK
      FM=FM+DFM*W2*(0.1*R)
   10 CONTINUE
```

```
C
      YMX=.5*RHO*YMX*REVS*REVS
      FM=FM*0.5*RHO*REVS*REVS
      RETURN
      END
C
C       ************************************************************
C           VEL COMPUTES THE INERTIAL FRAME OF REFERENCE VELOCITY
C           COMPONENTS.
C       ************************************************************
C
      SUBROUTINE VEL(VY,VZ,XB,PSI)
      COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
     *      V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *      B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *      VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
C
C
      CPSI=COS(PSI)
      SPSI=SIN(PSI)
      CDELTA=COS(DELTA)
      SDELTA=SIN(DELTA)
      SQ1=SIN(Q(1))
      SQ3=SIN(Q(3))
      CQ1=COS(Q(1))
      CQ3=COS(Q(3))
C
      X=XB*CPSI*CQ1 + RH*CPSI
      Y=XB*(SPSI*CQ1*CQ3-SQ1*SQ3) + RH*SPSI*CQ3 - SL*SQ3
      Z=XB*(SPSI*CQ1*SQ3+SQ1*CQ3) + RH*SPSI*SQ3 + SL*CQ3
C
C     CHOOSE LINEAR WIND SHEAR OR POWER WIND SHEAR
C
      IF ( VLP .EQ. 1 )  THEN
          V1=V*( 1.+HSHR*( Y*CDELTA-Z*SDELTA )/(1.5*R)
     *           -VSHR*X/(1.5*R)  )
      ELSE
          V1=V* HSHR*( Y*CDELTA-Z*SDELTA )/(1.5*R)
     *           + V * (1.-X/HH)**(VSHR)
      ENDIF
C
      VY=V1*SDELTA
      VZ=V1*CDELTA
      RETURN
      END
C
C       ************************************************************
C           VNMOD COMPUTES THE NORMAL COMPONENTS
C           OF VELOCITY BY SKEWED WAKE CORRECTION.
C       ************************************************************
C
      SUBROUTINE VNMOD(VN,XB,PSI,A,AXY,VY,VZ)
      COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
     *      V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *      B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *      VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
      COMMON/DENSE/RHO
C
```

```
C     CALCULATE SKEWED WAKE FACTOR
C
      VYZ=VY*COS(Q(3))+VZ*SIN(Q(3))+Q(4)*SL*REVS
      SKEW=15.*PI/32.
      SKEW=SKEW*SQRT((1.-COS(Q(3)))/(1.+COS(Q(3))))
      IF (VYZ .LE. 0.) THEN
           AXY=A*(1.-SKEW*XB/R*SIN(PSI))
      ELSE
           AXY=A*(1.+SKEW*XB/R*SIN(PSI))
      ENDIF
C
C     MODIFY FOR EFFECTS OF TOWER SHADOW
C
      VN=VN*(1.-AXY)/(1.-A)

      RETURN
      END
C
C
C     ****************************************************************
C           FUNCTION FPL CALCULATES PLATE LIFT COEFFICIENT
C             USING THE VITERNA METHOD
C     ****************************************************************
      FUNCTION  FPL(ANG)
      COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
     *      V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *      B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *      VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
C
      CE=.5*( C(10)+C(1) )
      AR=R/CE
      CDMAX=1.11 + 0.018*AR
      A1=CDMAX/2.
      ATN=AL(NLIFT)
      SSI=SIN(ATN)
      SCO=COS(ATN)
      A2=( CL(NLIFT) - CDMAX*SSI*SCO )*SSI/SCO/SCO
C
C     REFLECT CL CURVE ABOUT ALPHA=90 DEG
C
C
      IF( ANG.GT. AL(NLIFT) .AND. ANG .LE. PI/2. ) THEN
         SANG=SIN(ANG)
         COSANG=COS(ANG)
         FPL=2.*A1*SANG*COSANG + A2*COSANG*COSANG/SANG
      ENDIF
      IF( ANG .GT. PI/2. .AND.  ANG .LE. PI-AL(NLIFT) ) THEN
            ANG=PI-ANG
            SANG=SIN(ANG)
            COSANG=COS(ANG)
            FPL=2.*A1*SANG*COSANG + A2*COSANG*COSANG/SANG
            FPL=-FPL
      ENDIF
      IF( ANG .GT. PI-AL(NLIFT) .AND. ANG .LE. PI ) THEN
            ANG=PI-ANG
            FPL=-GETCL(ANG)
      ENDIF
      IF( ANG .GT. -AL(NLIFT).AND. ANG .LE. AL(1) ) THEN
```

```
            FPL=CL(1)+(ANG-AL(1))*(CL(NLIFT)+CL(1))/(AL(NLIFT)+AL(1))
      ENDIF
      IF( ANG .LE. -AL(NLIFT) .AND. ANG .GE. -PI/2. )  THEN
            ANG=-ANG
            SANG=SIN(ANG)
            COSANG=COS(ANG)
            FPL=2.*A1*SANG*COSANG + A2*COSANG*COSANG/SANG
            FPL=-FPL
      ENDIF
C
      RETURN
      END
C
C
C     ****************************************************************
C           FUNCTION FPD CALCULATES DRAG COEFFICIENT FOR HIGH
C           ANGLE OF ATTACK USING THE VITERNA METHOD.
C     ****************************************************************
C
      FUNCTION FPD(ANG)
      COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
     *      V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *      B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *      VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
C
      CE=.5*( C(10)+C(1) )
      AR=R/CE
      CDMAX=1.11 + 0.018*AR
      ATN=AD(NDRAG)
      SSI=SIN(ATN)**2
      SCO=COS(ATN)
      B22=( CD(NDRAG)-CDMAX*SSI )/SCO
C
C     REFLECT CD ABOUT ALPHA=90 DEG
C
      IF( ANG.GT. AD(NDRAG) .AND. ANG .LE. PI/2. ) THEN
            FPD=CDMAX*SIN(ANG)**2+B22*COS(ANG)
      ENDIF
      IF( ANG .GT. PI/2. .AND.  ANG .LE. PI-AD(NDRAG) ) THEN
            ANG=PI-ANG
            FPD=CDMAX*SIN(ANG)**2+B22*COS(ANG)
      ENDIF
      IF( ANG .GT. PI-AD(NDRAG) .AND. ANG .LE. PI ) THEN
            ANG=PI-ANG
            FPD=GETCD(ANG)
      ENDIF
      IF( ANG .GT. -AD(NDRAG).AND. ANG .LE. AD(1) ) THEN
            FPD=CD(1)+(-ANG+AD(1))*(CD(NDRAG)-CD(1))/(AD(NDRAG)+AD(1))
      ENDIF
      IF( ANG .LE. -AD(NDRAG) .AND. ANG .GE. -PI/2. )  THEN
            ANG=-ANG
            FPD=CDMAX*SIN(ANG)**2+B22*COS(ANG)
      ENDIF
C
      RETURN
      END
C
C
```

```
C     ****************************************************************
C         FUNCTION GETCL IS INTERPOLATION ROUTINE FOR AIRFOIL CL
C     ****************************************************************
C
      FUNCTION GETCL(ANG)
      COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
     *      V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *      B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *      VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
C
      GETCL = -999.
      IF( ANG .LT. AL(1) .OR. ANG .GT. AL(NLIFT) ) RETURN
      N=1
50    N=N+1
      IF( AL(N) .LT. ANG ) GO TO 50
      P=( ANG-AL(N-1) )/( AL(N)-AL(N-1) )
      GETCL = CL(N-1)+P*( CL(N)-CL(N-1) )
      RETURN
      END
C
C     ****************************************************************
C         FUNCTION GETCD IS INTERPOLATION ROUTINE FOR AIRFOIL CD
C     ****************************************************************
C
      FUNCTION GETCD(ANG)
      COMMON/YAW/ AL(30),AD(30),CL(30), CD(30),THETA(10),C(10),
     *      V,DELTA,HSHR,VSHR,VELDEF,R,RB,RH,HH,B,PC,REVS,SL,YI,BM,
     *      B1,B2,B3,FS,AV,AF,C1N,C2N,VB,Q(4),PITNOW,PITCH(3),
     *      VLP,MC,NS,PI,DEG,NLIFT,NDRAG,IYAC
C
      GETCD = -999.
      IF( ANG.LT.AD(1) .OR. ANG.GT.AD(NDRAG) ) RETURN
      N=1
50    N=N+1
      IF( AD(N) .LT. ANG ) GO TO 50
      P=( ANG-AD(N-1) )/( AD(N)-AD(N-1) )
      GETCD = CD(N-1)+P*( CD(N)-CD(N-1) )
      RETURN
      END
```

APPENDIX D

WIND TURBINE DYNAMIC DATA ANALYSIS SOFTWARE USER'S GUIDE

WIND TURBINE DYNAMIC DATA ANALYSIS SOFTWARE
USER'S GUIDE

INTRODUCTION

The wind energy industry and research community have depended heavily upon testing to determine the engineering performance of their products. Testing has proven invaluable in the development and improvement of almost all wind systems that have been used in the United States. And until theoretical models improve dramatically this dependence upon testing will continue, and probably deepen. The ultimate tests are those performed in the natural wind but those tests have been hampered by difficulties in analyzing the test results. Atmospheric turbulence creates inherent difficulties in measuring and characterizing the wind inputs to make understanding of the data a formidable task.

The University of Utah, as part of its contracted investigation of yaw dynamics, has developed a comprehensive software package for the analysis of wind turbine dynamic response test data. The software offers a capability for summarizing and understanding large and complex data files. Many of the capabilities are the standard and accepted methods such as the Method of Bins and Azimuth Averaging. In addition, a new capability has been added for the "harmonic analysis" of response data. The software was originally developed for use only by the University of Utah for analysis of yaw response tests. However, interest expressed in the methods by the Solar Energy Research Institute (SERI) and wind turbine manufacturers has resulted in the publication of this brief "User's Guide" in the hope that other organizations will begin to use the software package and/or the methods used in the package.

The reader is cautioned that this report is by no means complete documentation of the software system. Such documentation is beyond the scope and original intent of the SERI contract. However, it is hoped that an experienced wind turbine data analyst should be able to use the report and the program listings to install and operate the software with relative ease. The program listings contain many "comment cards" that are the most complete documentation of the programs.

This report is intended to briefly describe the software and its use. An overview of the capabilities of the package is provided first. This is merely a brief summary. The reader is referred to papers by Hansen (1987), Hansen and Hausfeld (1986) and Hansen (1980), as well as a report by Akins (1978) and for more details on the background and theoretical basis for the methods used. Next the computer system requirements are discussed. Finally a description of each program in the system is provided. (The most complete documentation of each program is found in the program listings.) Listings of the programs are provided to complete this report.

SOFTWARE CAPABILITIES

The attached flow chart (Figure 1) depicts the major elements of the package. The software begins with multi-channel, digitized, time-series data representing wind inputs (such as wind speeds and directions at one or more locations) and machine responses (such as power output, structural loads, structural motions, etc.). Up to 22 channels of data can be input to the system. The first program merely takes ASCII raw data files and rewrites them in a more compact and standard format for subsequent analysis. The next two programs prepare the data by de-spiking, filtering and calculating new parameters such as yaw error, yaw rate, wind shears, spatially averaged wind speeds and directions, and hub-height turbulence intensity.

A variety of analyses are possible with the system. The Method of Bins can be applied to any two channels of information. Azimuth Averaging (the Method of Bins using blade azimuth as the independent variable) can be applied to any dependent variable with periodic response (such as blade root bending moments). A new method of analysis, which represents a time-series of a periodic response channel as a new time-series of Fourier coefficients, is also provided. This method is detailed in a paper by Hansen (1987). Finally, a series of programs is available for transferring analyzed or raw data to commercial software packages for graphics, statistical, time-series or other analysis.

HARDWARE REQUIREMENTS

All of the programs are written in Fortran 77. Care has been taken to avoid use of non-standard features of Fortran 77. To date, the programs have been run on a VAX 11/750 using the VMS operating system and on a variety of IBM PC type systems (including PC, AT, and Compaq 386) using Lahey F77L Fortran 77. No problems have been encountered in running the programs in these environments.

Absolute requirements are minimal, but the software package is impractical if a hard disk (>20 Mb) is not available. A floating point processor is also desirable. To be useful a data file must be quite large. A ten minute test duration is about the minimum required while sample rates of 20-100 Hz are not uncommon for structural response tests. The result is large data files and long analysis times. As an example, a data file containing approximately 30,000 scans requires about one hour (clock time, not CPU time) of processing on the VAX 11/750 to prepare the data files. Each Method of Bins or harmonic analysis takes 5 to 10 minutes on the VAX. On fast microcomputers the analysis times are longer than these times but still quite acceptable.

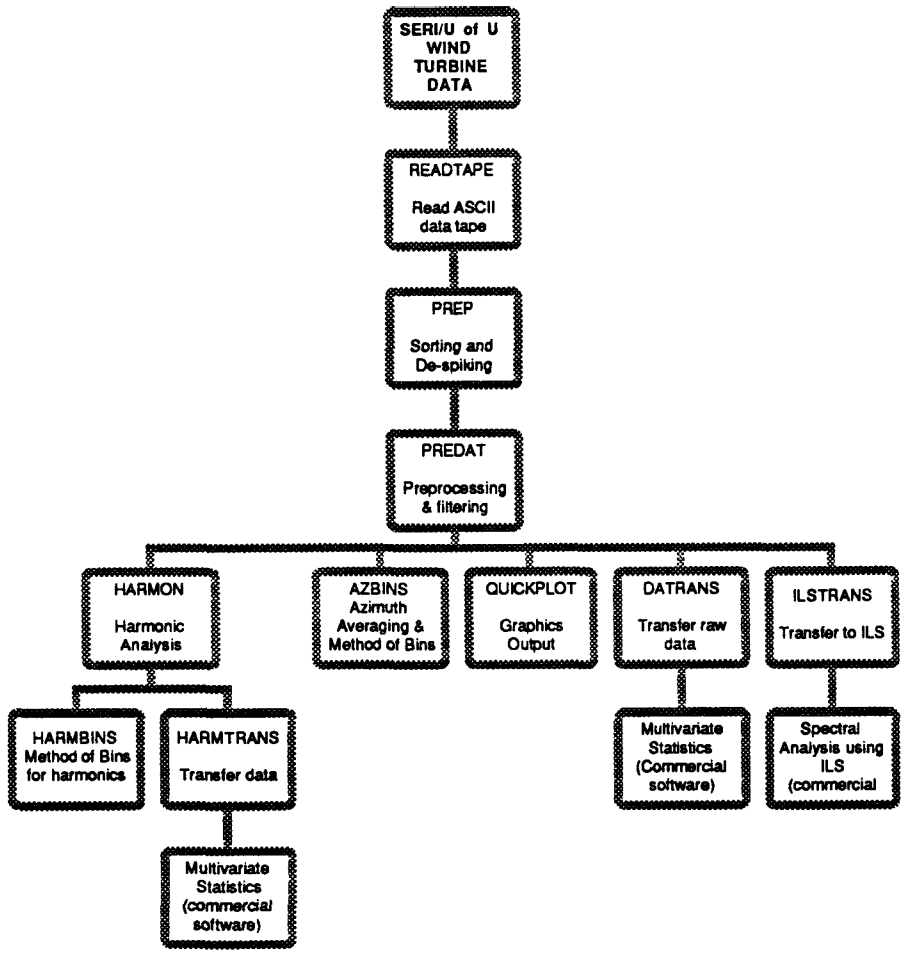

Figure 1. Flow chart showing the major elements of the data analysis package.

PROGRAM DESCRIPTIONS

In the paragraphs that follow a short description will be given for each program. The operation, limitations, and methods of each program will be discussed. However, the best way to understand a program is to understand the listing of the program.

READTAPE

READTAPE prepares data from a digital ASCII tape for further analysis. The purpose of the program is to import data in most any format and export the data in the format required by the next program. It performs the following tasks:

+ Reads an ASCII disc file containing a specified number of data channels (in engineering units). The ASCII data file is generally created on another computer or data acquisition system and transferred to the user's disc from

magnetic tape. Other types of input data can be handled by customizing the program as necessary.

+ Writes the data values to a new disc file in binary form. This is a much more efficient form of storage--both in terms of storage space and access time.

The primary purpose of READTAPE is to act as a simple interface between different computers using the most universal form of data input (ASCII). It can also be easily modified to permit custom alteration of the raw data (such as changing a calibration factor or changing units on a channel) before analysis. It must be expected that READTAPE will be partially rewritten for every new computer or input data file format. The program listing included in this report was written for test of a Howden 330 kW machine. A vertical plane array of anemometers was located upwind of the machine during these tests. This version of READTAPE averages various anemometer readings to create composite, or spatially averaged wind records. It also converts the wind speeds from m/s to mph. Special calculations such as this are easily accomplished in this program. The only firm requirement for READTAPE is that it write the data to file RAWDAT.DAT in an unformatted WRITE, with one scan of data per record. Any number of data channels can be read by READTAPE. Up to 22 channels can be output by READTAPE.

PREP

PREP serves two purposes: 1) It sorts the data into a standard order in a data array. 2) It allows "de-spiking" of the data. The processed data is written to a new, binary file PREDATIN.DAT for later processing.

+ It sorts the data into a standard data array. The data within one scan coming into PREP can be in any order. For example, the hub-height wind speed can be the first channel (array element) or the last. The program prompts the user for the identification of each data item. But the data file created by PREP is always in the same order, regardless of the machine that is being tested. For example, blade azimuth is always the first element in the record written by PREP. The order and items listed are shown in the attached list of data items (Table 1). This list can be modified, but only in selected ways. A few details are provided in Table 1. A zero will be written for any unused channel. Writing zeroes to the files for unused channels is not efficient use of storage but it simplifies understanding and writing the software.

+ The de-spiking process can be used to "clean up" noisy data. It can be very useful in removing drop-outs from the data set or replacing spike values with interpolated, averaged, or manually entered values. However, it can be unstable and destroy a data file if used without carefully considering the method. Obviously it is difficult to automate the process of identifying and eliminating incorrect data in files which contain randomly

varying, but correct values. The method of de-spiking used in this program is not useable for all types of data spikes. **PROCEED WITH CAUTION WHEN DE-SPIKING**. De-spiking is an option of the program that should not be exercised unless the user knows unacceptable spikes or dropouts are present in the data.

The de-spike routine works by testing the difference between any two consecutive values on a data channel and comparing the difference with a user-specified tolerance for that channel. If the difference exceeds the tolerance then the data point is flagged and the user is given options for handling the suspect data point. Options include replacing the suspect point with an average of previous points, replacing it with a value interpolated from previous points, replacing it with a user-specified value, or leaving the suspect point unchanged.

PREDAT

PREDAT is a pre-processor that serves many functions necessary prior to detailed data analysis. It performs the following tasks:

+ Reads the raw data file from PREP
+ Low-pass filters the data to improve correlation. A different cut-off frequency (or no filter) can be specified for each channel.
+ Computes disc-average wind speed and direction if data are available
+ Computes horizontal and vertical wind shear if data available
+ Computes yaw rate and yaw error
+ Computes turbulence intensity as measured by the hub height anemometer
+ Computes mean, standard deviation, maximum, and minimum for each channel of data over the entire record.
+ Stores the data in an unformatted, binary file for future processing

After PREDAT is run a data file containing the filtered, original data plus several channels of computed data (such as yaw rate) is available for detailed processing. The purpose of READTAPE, PREP and PREDAT, then, is to make available a well-conditioned time history for each of several standard test items and a limited number of special, calculated items. The programs that follow are designed to summarize this "raw" time-series data in a variety of ways.

In the current version of the program only low-pass, Sine-Butterworth digital filtering is available. This type of filtering introduces some start-up and ending transients. As a result, the first and last 200 scans of the file are eliminated after filtering to ensure that transients do not contaminate the data. This loss of 400 scans and the computation time required may compel the user to select the option of not filtering the data. However, filtering is generally recommended to eliminate high frequency, uncorrelated fluctuations in the various data channels. The cut-off frequency can be specified for each channel

independently because it is normally desirable to filter wind input channels at a much lower frequency than the structural response channels.

Yaw rate is computed using a two-point central difference technique. The yaw error is defined as the yaw angle minus the wind direction.

PREDAT calculates disc-averaged wind vectors and horizontal and vertical wind shears. A small data file called AVGINF.DAT contains the averaging information used to specify those anemometers to be used in the calculations. The user must edit this data file prior to running PREDAT to select the averaging options. The instantaneous disc-average wind speed is the arithmetic mean of the wind speeds at the specified anemometers. The first line of the AVGINF.DAT file specifies the number of anemometers (N) to be averaged. The next 'N' lines list the item numbers that identify each anemometer to be used. Line N+2 specifies the number of anemometers (M) to be used in the mean wind direction calculation. The next 'M' lines list the wind direction anemometers to be used. Note that the averaging process is simply arithmetic averaging of the wind direction values. The calculated average of two wind readings of 0° and 340° will be 170°, not the correct value of 350°. Vector averaging avoids this problem but is also much more time consuming. If your data contains wind directions on either side of 360°, you should adjust the data by applying an offset or ignore the average wind direction and yaw error results.

A simple measure of wind shear is simply the difference in instantaneous readings at two different anemometers. In PREDAT the horizontal wind shear is the difference between two anemometers at the same height, but displaced laterally from one another. The vertical wind shear is the difference between two anemometers which are displaced vertically from one another. The AVGINF.DAT file also specifies which anemometers are to be used in these calculations. When PREDAT is run the user is shown a list of the anemometers that will be used in these calculations and given the opportunity to accept or reject the list.

Turbulence intensity is defined as the root-mean-square velocity fluctuation divided by the mean velocity (expressed as a percentage). The rms fluctuation depends upon the time interval over which velocities are measured, thus the turbulence intensity also depends upon the averaging time used in the computation. In general the turbulence intensity increases as the averaging time increases because the low frequency fluctuations in the atmosphere are of larger amplitude than the high frequency fluctuations. It is common for meteorologists to use a ten-minute averaging time for defining the turbulence intensity. However, this time is too long for many engineering test data files. Therefore, the software prompts the user to input the averaging time that will be used in the calculations. The turbulence intensity that is reported by the program is a running average value. A value of -1% is returned by the program until one averaging interval has been analyzed. The negative values should of course be ignored in any subsequent analysis of the data.

In the present software item number 30 is not assigned any value. It is available for any calculated value the user wishes to assign.

TABLE 1

TYPICAL DATA ANALYSIS ITEMS

ITEM #	NAME
1	ROTOR AZIMUTH
2	WIND SPEED AT HUB
3	WIND DIRECTION AT HUB
4	YAW ANGLE
5	TEETER ANGLE
6	BLADE ROOT FLAP BENDING
7	LOAD 2
8	LOAD 3
9	ROTOR RPM
10	POWER OUTPUT
11	WIND SPEED 2
12	WIND DIRECTION 2
13	WIND SPEED 3
14	WIND DIRECTION 3
15	WIND SPEED 4
16	WIND DIRECTION 4
17	WIND SPEED 5
18	WIND DIRECTION 5
19	USER CHOICE
20	USER CHOICE
21	USER CHOICE
22	USER CHOICE
23	YAW RATE
24	YAW ERROR
25	DISC AVG. WIND SPEED
26	DISC AVG. WIND DIRECTION
27	HORIZONTAL WIND SHEAR
28	VERTICAL WIND SHEAR
29	TURBULENCE INTENSITY AT HUB HEIGHT
30	NOT CURRENTLY USED

Items 1- 4, and 23-29 cannot be changed without changing the programs. All other names can be changed or used at will. File AVGINFO.DAT is used to provide information on disc averaging of wind speed and direction and for computation of wind shears.

AZBINS

AZBINS performs either a standard Method of Bins analysis or an azimuth averaging analysis of any two channels of data. The independent variable is the variable which is subdivided into intervals or bins and the dependent variable is that which is averaged within each bin. AZBINS allows selective sampling on any two parameters in addition to the two variables being analyzed (in essence, a four dimensional method of bins). A summary table and plot are created (the plot is useable only on the University of Utah VAX). In this and other programs with graphic outputs, the graphics capability is limited to the VAX using PLOT79 graphics software. Other users should simply comment the graphics calls out of the program before compiling.

QUICKPLOT

QUICKPLOT allows plotting of the time history of any data channel or a scatter plot of any channel vs. any second channel. This is useful for data quality checking and visualization as well as scanning for cause-and-effect relationships between different data channels. This program uses graphics routines which are useable only on the University of Utah VAX system. It is included here only because it may be easily modified to work with other graphics hardware.

HARMON

HARMON computes the harmonic content of any single data signal. It provides a means for determining the cyclic content of a signal and presenting the information in a compressed form. It allows investigation of questions such as, "How does the teeter amplitude depend upon the yaw rate?" or, "How does the once-per-revolution (1p) component of blade root bending moment depend upon the vertical wind shear?". The method is described in some detail by Hansen (1987)

The program computes an azimuth-averaged time history for an arbitrary number (typically 1-100) of rotor revolutions, repeating this process for the entire data file. Then the Fourier coefficients are computed and stored for each azimuth average curve. The following items are determined for each azimuth-averaged segment of data:

+ The mean, standard deviation, peak-to-peak and maximum of one parameter for the cycle
+ The harmonic amplitude and phase of the one parameter (up to 10p)
+ All channels not included in harmonic analysis are each averaged over the cycle and stored

Clearly this type of analysis is not appropriate for a parameter which does not exhibit cyclic behavior. But many parameters, such as rotor loads, do exhibit cyclic behavior and this analysis permits the user to better understand the relationship between those loads and the factors that cause them. This method is also inappropriate for data which contains significant response at a frequency which is not an integer multiple of 1p. Wide-band spectral analysis should be performed on the data prior to running HARMON to ensure the data are suitable for harmonics analysis.

HARMBINS

HARMBINS allows method of bins analysis of the harmonic data. For example, a curve of 1p flap bending amplitude can be plotted against wind speed (or any other variable). The output is a table and plot (again, the plot is useable only on the University of Utah computer).

READHARMON

This program simply prints the harmonic coefficients and time averaged data values from HARMON to the screen for quality control purposes. When the program is run it will display the harmonic content of the first azimuth average set on the screen. Subsequent values will be displayed as long as the user give the affirmative response (a value of 1) to the "Want another?" query.

HARMTRANS

This program reads the harmonic data file created by HARMON and writes user-selected parameters to an ASCII data file that is suitable for transfer to other computers or software packages. Its sole purpose is to act as an interface between HARMON and other, commercially available, software. The author uses this method to enter the harmonic data results into a commercially available statistics analysis and graphics package. This permits quick and easy correlations, scatter plots and factor analysis.

DATRANS

DATRANS is similar to HARMTRANS in purpose. However, it prepares raw time-series data files (from PREDAT) for transfer in ASCII form. Any number of user-selected channels of data can be transferred.

ILSTRANS

This program prepares time-series data for transfer to a commercial software package called the Interactive Laboratory System (ILS) by Signal Technology,

Inc. ILS can perform virtually all of the standard frequency-domain operations such as power spectral density, transfer function, correlation, pattern recognition, and digital filtering. ILSTRANS prepares user-specified data from PREDAT in the specific format required by ILS.

SUMMARY

The series of programs just described can be grouped into three categories: 1) Pre-processing routines to condition the data set. 2) Analysis routines to summarize the data. 3) Interface programs to transfer data to other computers or software packages. With this family of programs the wind turbine test engineer or designer can gain considerable insight to the operation and performance of a machine. The method is particularly suited to analysis of large data files selected to cover a wide range of operating conditions while measuring many structural response parameters. However, the method can also be used for operations such as measurement of a power curve using the Method of Bins.

Users of the software are encouraged to provide the author with feedback concerning the ease of use and value of the results generated. Users are also encouraged to make changes to the software and inform the author of the results of the changes.

TABLE 2
SUMMARY OF PROGRAM DATA FILES
AND FORTRAN UNIT NUMBERS

PROGRAM	INPUT FILES	OUTPUT FILES
READTAPE	Input any filename from keyboard (20)	RAWDAT.DAT (30) FILINFO.DAT (10)
PREP	RAWDAT.DAT (20) FILINFO.DAT (30) TOLER.DAT (23) NAMES.DAT (11) ITEMS.DAT (21)	PREDATIN.DAT (25) FILINFO.DAT (30)
PREDAT	PREDATIN.DAT (20) FILINFO.DAT (30)	AZIN.DAT (25) NAMES.DAT (11) AVGINF.DAT (10) FILTER.DAT (23) ITEMS.DAT (21) DATSUM.DAT (10) AZINFO.DAT (30)
AZBINS	NAMES.DAT (40) AZINFO.DAT (30) AZIN.DAT (20)	AZBINOUT.DAT (10)
HARMON	AZIN.DAT (20) AZINFO.DAT (30) NAMES.DAT (40)	BINTBL.DAT (10) HARMCONT.DAT (25) HARMINFO.DAT (35) BINPL.DAT (50)
READHARMON	HARMCONT.DAT (10)	Print to CRT only
HARMTRANS	NAMES.DAT (40) HARMINFO.DAT (30) HARMCONT.DAT (20)	HTRANS.DAT (10)
HARMBINS	NAMES.DAT (40) HARMINFO.DAT (30) HARMCONT.DAT (20)	HBINOUT.DAT (10)
QUICKPLOT	AZINFO.DAT (30) AZIN.DAT (25) NAMES.DAT (11)	Plots only
DATRANS	NAMES.DAT (40) AZINFO.DAT (30) AZIN.DAT (20)	TRANS.DAT (10)
ILSTRANS	NAMES.DAT (40) AZINFO.DAT (30) AZIN.DAT (20)	ILSINFO.DAT (50) TR10x.DAT (11,12,...) (one file for each channel)

TABLE 3
DATA FILES USED IN HARMONIC DATA ANALYSIS SOFTWARE

NAME CREATED BY	CONTENTS	TYPE	USED or
AVGINF.DAT	Wind averaging anemometer ID's	Formatted[1]	PREDAT
AZBINOUT.DAT	Summary of Bins output	Formatted	AZBINS
AZIN.DAT	Pre-Processed time-series data	Unformatted[2]	PREDAT AZBINS HARMON QUICKPLOT DATRANS ILSTRANS
AZINFO.DAT AZIN.DAT	Misc. data file information	Formatted	Same as
BINPL.DAT	Table of plot data	Formatted	HARMON
BINTBL.DAT	Tables summarizing HARMON output	Formatted	HARMON
DATSUM.DAT	Summary of data channels (mean, min, max, etc. from PREDAT)	Formatted	PREDAT
FILTER.DAT	Filter cutoff frequencies for data channels	Formatted	PREDAT
FILINFO.DAT	Summary of raw data file parameters	Formatted	READTAPE PREP
HARMCONT.DAT	"Time series" of harmonic coefficients for each azimuth-averaged cycle	Unformatted	HARMON HARMTRANS HARMBINS READHARMON
HARMINFO.DAT	Misc. information about Harmonics	Formatted	HARMON HARMTRANS HARMBINS
HBINOUT.DAT	Summary of HARMON output	Formatted	HARMON
HTRANS.DAT	ASCII file of harmonic and other data for transfer to other computer	Formatted	HARMTRANS
ILSINFO.DAT	Misc. information about transfer of data to ILS software package	Formatted	ILSTRANS
ITEMS.DAT	Relates channel #'s to Item #'s	Formatted	PREDAT
NAMES.DAT	Names of all data Items	Formatted	PREDAT AZBINS HARMON HARMTRANS HARMBINS QUICKPLOT

[1]ASCII file written with FORTRAN format statements (can be edited, typed, etc.) (sequential access except where noted)

[2]Binary file written with unformatted WRITE (e.g. WRITE(unit #) variable name) (sequential access)

			DATRANS ILSTRANS
PREDATIN.DAT	Sorted, de-spiked time series	Direct Access	PREP PREDAT
RAWDAT.DAT	Raw time series data	Unformatted	READTAPE PREP
TOLER.DAT	Tolerances for de-spiking	Formatted	PREP
TR10x.DAT	One file for each channel of data transferred to ILS	Direct Access	ILSTRANS

LIST OF REFERENCES

Akins, R.E. (1978). "Performance Evaluation of Wind Energy Conversion Systems Using the Method of Bins--Current Status". Report number SAND77-1375, Sandia National Laboratories, Albuquerque, NM.

Hansen, A.C. (1987). "A Method for Analyzing Wind Turbine Dynamic Response Test Data". Submitted to the ASME Journal of Solar Energy Engineering.

Hansen, A.C., (1980). "Effects of Turbulence on Wind Turbine Performance", Transportation Engineering Journal of ASCE, Vol. 106, No. TE 6.

Hansen, A.C. and Hausfeld, T.E., (1986). "Frequency Response Matching to Optimize Wind-Turbine Test Data Correlation". ASME Journal of Solar Energy Engineering. Vol 108, No. 3, August, 1986.

PROGRAM LISTINGS

All of the programs described in this report are listed in this appendix. One additional program, GENDAT, is provided for testing and debugging of the software package using a known data set that is generated by GENDAT. The contents of the appendix are as follows:

```
      PROGRAM GENDAT
C
C     PROGRAM TO GENERATE AN ASCII DATA FILE FOR USE IN TESTING
C     SERI DATA ANALYSIS SOFTWARE
C     GENERATES UP TO 18 CHANNELS OF DATA INCLUDING CONSTANTS,
C     SINE WAVES, WHITE NOISE AND RAMPS.  SUBROUTINE DUMMY CONTAINS
C     THE EQUATIONS USED TO GENERATE EACH CHANNEL OF DATA
C
C     C HANSEN 1/87    UNIVERSITY OF UTAH
C
      DIMENSION DATOUT(18), D(18,50)
      INTEGER*4 NSCAN
C
C     INITIALIZE VARIABLES

      WRITE(*,*) 'THIS IS THE PROGRAM FOR GENERATING TEST DATA'
      WRITE(*,*) 'HOW MANY SCANS DO YOU WANT TO GENERATE?'
      READ(*,*) NSCAN
      WRITE(*,*) 'ENTER SAMPLE RATE (SCANS/SEC TYPICALLY 40 HZ)'
      READ(*,*) SRATE
      DELT=1./SRATE
      WRITE(*,*) 'ENTER INTERVAL FOR WRITING TO CRT (>0)'
      READ(*,*) IPRINT
      NSET=0
      ISCAN=0
C
  10  WRITE(*,*) 'ENTER NUMBER OF CHANNELS '
      READ(*,*) NCHAN
      WRITE(*,*) 'ENTER NUMBER OF SCANS PER RECORD'
      READ(*,*) NUMSCN
      IRECL=13*NCHAN*NUMSCN
      IF(IRECL.GT.1690) THEN
         WRITE(*,*) 'THE PRODUCT OF NCHAN AND NUMSCN CANNOT'
         WRITE(*,*) 'EXCEED 130'
         GO TO 10
      ENDIF
C
C     OPEN DATA FILE FOR OUTPUT
C
      OPEN(UNIT=20,FILE='TEST.DAT',STATUS='NEW',
     +             IOSTAT=IERR,RECL=IRECL)
      IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING TEST.DAT DATA FILE (UNIT 20)'
         WRITE(*,*) 'IERR= ',IERR
         STOP
      ENDIF
C
C     MAIN LOOP
C
 1000 CONTINUE
      DO 100 I=1,NUMSCN
C
C     GET ONE SCAN
C
      CALL DUMMY(DELT,TIME,NSET,DATOUT)
      ISCAN=ISCAN+1
      IF(ISCAN/IPRINT.EQ.FLOAT(ISCAN)/FLOAT(IPRINT)) THEN
         WRITE(*,*) ISCAN
```

```
       WRITE(*,2010) (D(IW,I),IW=1,NCHAN)
      ENDIF
 2010 FORMAT(6E13.6)
      NSET=1
C
      DO 110 J=1,NCHAN
      D(J,I)=DATOUT(J)
  110 CONTINUE
C
  100 CONTINUE
C
C     WRITE NUMSCN SCANS TO OUTPUT FILE
C
      WRITE(20,2000) ((D(JJ,II),JJ=1,NCHAN),II=1,NUMSCN)
 2000 FORMAT(130E13.6)
C
C     RETURN FOR MORE DATA UNTIL NSCAN VALUES
C
      IF(ISCAN.LT.NSCAN) GO TO 1000
      WRITE(*,*) 'FINISHED'
      CLOSE(20)
C
      STOP
      END
C
C     ****************
C
      SUBROUTINE DUMMY(DELT,TIME,NSET,DATOUT)
C
      DIMENSION DATOUT(18)
C
      IF(NSET.EQ.0) TIME=0.0
      PI2=6.2831853
C
C     FREQUENCIES IN HZ
C
      F1=1.0
      F2=2.0
C
C     GENERATE RANDOM NUMBER
C
      CALL TDRAND(RAND)
C
C     AZIMUTH
      DATOUT(1)=AMOD(F1*TIME,1.)*360.
C
C     WIND SPEED @ HUB
      DATOUT(2)=5.*(1.+.05*TIME)
C
C     WIND DIRECTION @ HUB HEIGHT
      DATOUT(3)=10.*SIN(PI2*F2*TIME)
C
C     YAW ANGLE
      DATOUT(4)=5.00*SIN(PI2*F2*TIME)
C
C     TEETER ANGLE
      DATOUT(5)=.005*DATOUT(1)
C
```

```
C     BLADE ROOT FLAP BENDING
      DATOUT(6)=300.*(SIN(PI2*F2*TIME)+SIN(PI2*F2*2.*TIME))
C
C     STRAIN 2
      DATOUT(7)=1000.
C
C     STRAIN 3
      DATOUT(8)=100.*RAND
C
C     ROTOR RPM
      DATOUT(9)=100.+RAND
C
C     POWER OUTPUT
      DATOUT(10)=54.*(DATOUT(2)/10.)**3
C
C     WIND SPEEDS FROM FOUR OTHER ANEMOMETERS
      DATOUT(11)=10. + 10.*SIN(PI2*F1*TIME) + 10.*RAND
      DATOUT(13)=20. + 10.*SIN(PI2*F1*TIME) + 10.*RAND
      DATOUT(15)=30.
      DATOUT(17)=40.
C
C     WIND DIRECTION AT FOUR OTHER ANEMOMETERS
      DATOUT(12)=90.
      DATOUT(14)=95.
      DATOUT(16)=100.
      DATOUT(18)=105.
C
C
      TIME=TIME+DELT
      RETURN
      END
C
C     *****************
C
      SUBROUTINE TDRAND(X)
C
C     GENERATES UNIFORMLY DISTRIBUTED RANDOM NUMBER
C     BETWEEN 0 AND 1 WITH MEAN = 0.5 AND VARIANCE = 1/12
C     FROM TEXT BY ENOCHSON AND OTNES
C
      SAVE I
      DATA I/783637/
C
      I=125*I
      I=I-(I/2796203)*2796203
      X=FLOAT(I)/2796202.
      RETURN
      END
```

```
      PROGRAM READTAPE
C
C     THIS VERSION MODIFIED TO READ HOWDEN 300 KW TEST DATA TAPES
C     FOR FIXED YAW ANGLE TESTS
C     C HANSEN   8/87
C
C     PROGRAM TO READ RAW DATA FILE FROM SERI (MAG TAPE COPIED
C     TO DISC) AND RE-WRITE THE FILE IN MORE EFFICIENT FORM
C     ALSO ALLOWS ANY SPECIAL MODIFICATIONS TO THE DATA
C     AND EASY ACCOMODATION OF ANY SPECIAL TAPE DATA FORMATS
C
      INTEGER*4 NSCAN
      DIMENSION D(24), DNEW(17)
      CHARACTER*80 INFILE,TITLE
C
C     INITIALIZE VARIABLES AND GET SETUP INFORMATION
C
      WRITE(*,*) 'THIS IS THE SERI DATA FILE TRANSFER PROGRAM'
      NCHAN=24
      NCHOUT=17
      SRATE=42.
      DELT=1./SRATE
      WRITE(*,*) 'ENTER INTERVAL FOR WRITING TO CRT (>0)'
      READ(*,*) IPRINT
C
      NSCAN=0
      WRITE(*,*) 'ENTER INPUT DATA FILE NAME'
      READ(*,200) INFILE
  200 FORMAT(A)
      NUMSCN=1
C
      NWRITE=1
C
      WRITE(*,*) 'ENTER TITLE OR DESCRIPTOR OF THIS RUN (<80 CHARS)'
      READ(*,200) TITLE
      WRITE(*,*) 'ENTER YAW ANGLE (FIXED YAW CASE)'
      READ(*,*) DNEW(4)
C
C     OPEN DATA FILES FOR INPUT AND OUTPUT
C
      OPEN(UNIT=20,FILE=INFILE,STATUS='OLD',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING INPUT DATA FILE (UNIT 20) '
         WRITE(*,*) 'FILE NAME= ',INFILE
         WRITE(*,*) 'IERR= ',IERR
         STOP
      ENDIF
C
      OPEN(UNIT=10,FILE='FILINFO.DAT',STATUS='NEW',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING FILINFO.DAT FILE (UNIT 10)'
         WRITE(*,*) 'IERR= ',IERR
         STOP
      ENDIF
C
C
      OPEN(UNIT=30,FILE='RAWDAT.DAT',STATUS='NEW',
     +     FORM='UNFORMATTED')
```

```
C
C     **************
C
C     READ DISC UNTIL END OF FILE AND WRITE UNFORMATTED RECORD
C
 10   READ(20,100,END=999) ID,IH,IM,IS,IMS,(D(J),J=1,NCHAN)
 100  FORMAT(1X,5(I3,1X),7X,24(1X,E13.6))
C
C     MODIFY DATA BEFORE RE-WRITING
C
C     MAKE AZIMUTH PSI=0 WHEN BLADE DOWN, CONVERT WINDS TO MPH
C
      IF(D(1).LT.180.) THEN
          DNEW(1)=D(1)+180.
        ELSE
          DNEW(1)=D(1)-180.
      ENDIF
C
      DNEW(2)=D(21)*2.237
      DNEW(3)=D(22)
      DNEW(5)=D(4)
      DNEW(6)=D(5)
      DNEW(7)=D(6)
      DNEW(8)=D(2)
      DNEW(9)=D(3)
C
C     SELECTED AVERAGING OF WIND SPEEDS AND DIRECTIONS
C
      DNEW(10)=(D(7)+D(9))/2.*2.237
      DNEW(11)=(D(8)+D(10))/2.
      DNEW(12)=(D(11)+D(23))/2.*2.237
      DNEW(13)=(D(12)+D(24))/2.
      DNEW(14)=(D(13)+D(15))/2.*2.237
      DNEW(15)=(D(14)+D(16))/2.
      DNEW(16)=(D(17)+D(19))/2.*2.237
      DNEW(17)=(D(18)+D(20))/2.
C
C     WRITE EVERY SCAN
C
      NSCAN=NSCAN+1
      WRITE(30)   (DNEW(I),I=1,NCHOUT)
C
      IF(NSCAN/IPRINT.EQ.FLOAT(NSCAN)/FLOAT(IPRINT)) THEN
        WRITE(*,*) NSCAN
        WRITE(*,120) ID,IH,IM,IS,IMS
 120    FORMAT(5(1X,I3))
        WRITE(*,110) (DNEW(I),I=1,NCHOUT)
 110    FORMAT(1X,6E13.6)
      ENDIF
      GO TO 10
C     **************
C
C     FINISHED
C
 999  CONTINUE
      WRITE(*,*) 'FINISHED READING TAPE'
      WRITE(*,*) 'NUMBER OF SCANS TRANSFERRED= ',NSCAN
C
```

```
C     WRITE FILE INFORMATION TO FILINFO.DAT
C
      WRITE(10,*) TITLE
      WRITE(10,*) NCHOUT
      WRITE(10,*) NSCAN
      WRITE(10,*) DELT
C
      CLOSE(10)
      CLOSE(20)
      CLOSE(30)
      END
```

```
      PROGRAM PREP
C
C     THIS PROGRAM IS THE SECOND OF SEVERAL PROGRAMS FOR PROCESSING
C     AZIMUTH AVERAGE TEST DATA FROM ASCII DATA TAPES.
C     THE FIRST PROGRAM IS 'READTAPE'.  THAT PROGRAM TRANSFERS DATA
C     FROM AN ASCII MAG TAPE TO A DISC FILE ON THE VAX.
C     PREP READS THE DISC FILE AND ALLOWS
C     VARIOUS TYPES OF PROCESSING BEFORE CREATING A NEW DISC FILE
C     CALLED 'PREDATIN.DAT'.  THE OTHER PROGRAMS RUN BINS, HARMONICS,
ETC.
C
C     DETAILS ON 'PREP' FOLLOW:
C     THE PROGRAM PERFORMS PRE-PROCESSING OPERATIONS ON THE RAW
C     DATA FILE 'RAWDAT'.  THE RAW DATA MUST BE EQUAL TIME INTERVAL
C     SAMPLES WHICH INCLUDE ROTOR AZIMUTH. OPTIONAL
C     CHANNELS OF DATA INCLUDE POWER, WIND SPEED,YAW, WIND DIRECTION,
C     RPM, VARIOUS STRAIN GAGES, TEETER ANGLE AND OTHER ANEMOMETERS.
C     RAWDAT IS THE FILE CREATED BY PROGRAM 'READTAPE'
C
C     PREP FIRST READS THE DATA FILE AND SORTS THE RAW DATA
C     (ARRAY DATA(I) INTO THE ARRAY D(I).  IN ARRAY D THE
C     CHANNEL IDENTIFICATIONS ARE ALWAYS THE SAME (REGARDLESS OF
C     THE CHANNEL ID'S ON THE ANALOG RECORDER).  THEY ARE TAKEN FROM
C     THE NAMES.DAT FILE.  THE ITEMS USED IN ITEMS 1 THRU 4 AND 23 THRU
29
C     ARE USED IN INTERNAL CALCULATIONS AND CANNOT BE CHANGED WITHOUT
C     CHANGES IN THE PROGRAM SOURCE CODES.
C
C     THE INFORMATION RELATING ANALOG TAPE CHANNELS TO THE ABOVE
C     ITEMS IS STORED IN FILE 'ITEMS'.  ITEMS IS ACCESSED AND EDITED
C     BY THE SUBROUTINE 'GITEMS', CALLED BY PREP.
C
C     PREP NEXT ALLOWS DE-SPIKING OF THE DATA.  A SET OF SPIKE
TOLERANCES
C     IS READ FROM FILE SPIKE.DAT AND ANY DATA WITH CHANGES GREATER
C     THAN THE GIVEN TOLERANCE (CHANGES BETWEEN TWO CONSECUTIVE DATA
POINTS)
C     IS FLAGGED AND DE-SPIKED IF DESIRED.
C     DETAILS OF THE DE-SPIKING OPERATION ARE GIVEN IN SUBROUTINE
C     DESPIK
C
C     PREP KEEPS TRACK OF THE MEAN, MINIMUM, MAXIMUM, AND STANDARD
C     DEVIATION OF EACH CHANNEL AND PRINTS A SUMMARY FOR THE INFORMATION
C     OF THE USER.  THE SUMMARY IS OF THE DATA AFTER ANY DE-SPIKING
C     IS COMPLETED.
C
C
C     VARIABLES:
C        TITLE=A USER SUPPLIED TEXT (UP TO 80 CHARS.) THAT
C              IDENTIFIES THE TEST
C        INCHAN=MAX. NUMBER OF INPUT DATA CHANNELS (EXCL. TIME CODE)
C        NCHAN=ACTUAL NUMBER OF ACTIVE CHANNELS OF
C              DATA (EXCLUDING TIME CODE)
C        NSCAN=NUMBER OF SCANS IN THE ORIGINAL DATA SET
C        NSKIP=NUMBER OF SCANS SKIPPED BEFORE ANALYSIS BEGINS
C        NRUN= NUMBER OF SCANS THAT WILL BE ANALYZED
C        ITEM=ARRAY OF CHANNEL ID'S.  FOR EXAMPLE, IF WIND SPEED (ITEM
2)
```

```
C             IS ON TAPE CHANNEL 5, THEN ITEM(5)=2
C        MIN,MEAN,MAX,STDEV=ARRAYS OF SUMMARY INFO FOR EACH CHANNEL
C             OF FILTERED DATA
C        TOLER=ARRAY OF TOLERANCES USED IN DE-SPIKING
C        NSPIKE=COUNT # OF SPIKES CHANGED ON EACH CHANNEL
C        NFLAG=COUNT # SPIKES IDENTIFIED ON EACH CHANNEL
C
C     SUBROUTINES:
C        DESPIK, GSPIKE, GITEMS
C
C     WRITTEN BY C HANSEN  1/87, UNIVERSITY OF UTAH
C     NO DOCUMENTATION OF THIS PROGRAM EXISTS OTHER THAN THE LISTING.
C
      PARAMETER (NITEMS=30)
      PARAMETER (INCHAN=22)
C
      INTEGER*4 NSCAN,ISCAN,NSKIP,NRUN
      CHARACTER*30 NAME(NITEMS)
      CHARACTER*80 TITLE
      REAL*4 DATA(INCHAN)
      DIMENSION TOLER(INCHAN)
      REAL MIN(INCHAN), MAX(INCHAN)
      DOUBLE PRECISION MEAN(INCHAN), STDEV(INCHAN), NPOINT
      COMMON D(INCHAN)
      COMMON/SPIKE/ NSPIKE(INCHAN),NFLAG(INCHAN),OLDAT(INCHAN,3),
     +              ITEM(NITEMS)
C
      DATA MIN/INCHAN*1.E20/,MAX/INCHAN*-1.E20/,MEAN/INCHAN*0./
      DATA STDEV/INCHAN*0.0D00/, NPOINT/0.0D00/
C
      DO 12 I=1,INCHAN
      NSPIKE(I)=0
      NFLAG(I)=0
      D(I)=0.0
  12  CONTINUE
C
C     OPEN 'FILINFO' FILE TO GET FILE INFORMATION
C
      OPEN(UNIT=30,FILE='FILINFO.DAT',STATUS='OLD',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING FILINFO.DAT FILE (UNIT 30)'
         WRITE(*,*) 'IERR= ',IERR
         STOP
      END IF
      READ(30,2000) TITLE
 2000 FORMAT(A)
      READ(30,*) NCHAN
      READ(30,*) NSCAN
      READ(30,*) DELT
C
C     REWIND FILINFO FILE TO PREPARE FOR UPDATING
C
      REWIND(30)
C
C     OPEN 'RAWDAT.DAT' FILE WHICH CONTAINS ALL OF THE DATA
C
      OPEN(UNIT=20,FILE='RAWDAT.DAT',STATUS='OLD',IOSTAT=IERR,
     +     FORM='UNFORMATTED')
```

```
      IF(IERR.NE.0) THEN
         WRITE(*,*)  'ERROR OPENING RAWDAT.DAT (UNIT 20)'
         WRITE(*,*)  'IERR= ',IERR
         STOP
      ENDIF
      REWIND 20
C
C     OPEN UNIT 25 (FILE PREDATIN) TO RECEIVE PROCESSED TIME SERIES DATA
C
      OPEN(UNIT=25,FILE='PREDATIN.DAT',ACCESS='DIRECT',
     +     RECL=INCHAN, STATUS='NEW',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
         WRITE(*,*)  'ERROR OPENING PREDATIN.DAT (UNIT 25)'
         WRITE(*,*)  'IERR= ',IERR
         STOP
      ENDIF
C
C     COUNTER FOR NUMBER OF SCANS PROCESSED (USED IN WRITING
C     THE DIRECT ACCESS FILE)
C
      ISCAN=0
C
      WRITE(*,*) 'THIS IS THE DATA PREPARATION PROGRAM'
      WRITE(*,*) 'FOR SORTING AND DE-SPIKING FILE RAWDAT.DAT'
      WRITE(*,*) ' '
C
C     DETERMINE CHANNEL IDENTIFICATIONS (ITEM AND NAME ARRAYS)
C
      CALL GITEMS(ITEM,NITEMS,NAME,NCHAN)
C
C     DETERMINE DE-SPIKE TOLERANCES IF DESIRED
C
      WRITE(*,*) ' '
      WRITE(*,*) 'DO YOU WANT TO DE-SPIKE THE DATA? (YES=1)?'
      READ(*,*) NYSPIK
      IF(NYSPIK.EQ.1) CALL GSPIKE(TOLER,INCHAN,ITEM,NAME)
C
      WRITE(*,*) ' '
      WRITE(*,*) 'THE DATA FILE CONTAINS ',NSCAN,' SCANS'
      WRITE(*,*) 'AND ',NCHAN,' CHANNELS OF DATA'
      WRITE(*,*) ' '
      WRITE(*,*) 'NUMBER OF SCANS TO SKIP? (0=FIRST RECORD AVAILABLE)'
      READ*,NSKIP
      WRITE(*,*) 'NUMBER OF SCANS TO ANALYZE?'
      READ*, NRUN
C
      WRITE(*,*) 'STARTING PROCESSING.......'
C
C     SKIP INTO THE DESIRED SCAN
C
      IF(NSKIP.EQ.0) GO TO 2
      DO 1 I1=1,NSKIP
 1    READ(20,END=9999) (DATA(I),I=1,NCHAN)
 2    CONTINUE
C
C     SAMPLE DATA UNTIL THE STOP TIME
C
C     READ IN A SCAN OF DATA
```

```
C
  5   READ(20,END=999) (DATA(J),J=1,NCHAN)
      ISCAN=ISCAN+1
C
C     SORT THE SCAN INTO THE DATA ARRAY
C
      DO 30 J=1,NCHAN
      IF(ITEM(J).EQ.0) GO TO 30
      D(ITEM(J))=DATA(J)
 30   CONTINUE
C
C     DE-SPIKE THE DATA IF DESIRED
C
      IF(NYSPIK.EQ.1) CALL DESPIK(TOLER,INCHAN,ISCAN,NAME)
C
C     ACCUMULATE MEAN, MIN, MAX AND STANDARD DEVIATION
C
      DO 40 J=1,INCHAN
      MEAN(J)=MEAN(J) + D(J)
      STDEV(J)=STDEV(J) + D(J)*D(J)
      IF(D(J).LT.MIN(J)) MIN(J)=D(J)
      IF(D(J).GT.MAX(J)) MAX(J)=D(J)
  40  CONTINUE
      NPOINT=NPOINT+1.
C
C     WRITE THE SCAN TO THE DIRECT ACCESS FILE
C
      WRITE(25,REC=ISCAN) (D(IW),IW=1,INCHAN)
C
C     RETURN FOR MORE DATA
C
      IF(ISCAN.LT.NRUN) GO TO 5
C
C     CLOSE RAWDAT FILE
C
 999  CLOSE(20)
C
C     UPDATE THE FILINFO FILE TO SHOW # DATA POINTS PROCESSED
C
      WRITE(30,2000)TITLE
      WRITE(30,*) NCHAN
      WRITE(30,*) ISCAN
      WRITE(30,*) DELT
      CLOSE(30)
C
C     WRITE SUMMARY INFORMATION TO CRT AND DISC FILE 'FILSUM.DAT'
C
      OPEN(UNIT=10,FILE='FILSUM.DAT',STATUS='NEW')
      WRITE(*,1005) TITLE
      WRITE(10,1003)
 1003 FORMAT(1X,'SUMMARY OF DATA FROM PROGRAM PREP')
      WRITE(10,1005) TITLE
 1005 FORMAT(///,1X,A80/)
      IF(NYSPIK.NE.1) THEN
         WRITE(*,*)   ' DE-SPIKE OPERATION NOT PERFORMED'
         WRITE(10,*) ' DE-SPIKE OPERATION NOT PERFORMED'
      ENDIF
      WRITE(*,1010)
```

```
      WRITE(10,1010)
 1010 FORMAT(/6X,'ITEM',24X,'MEAN',10X,'MIN',10X,'MAX',8X,'STDEV')
C
C     COMPUTE FINAL MEANS AND STANDARD DEVIATIONS
C
C     NOTE THAT IF SMALL DIFFERENCES OF LARGE NUMBERS ARE
C     USED FOR STANDARD DEVIATION CALCULATION, ROUND-OFF
C     ERRORS WILL RESULT IN INVALID STDEV.  IF THE
C     ARGUMENT OF THE SQUARE ROOT IS NEGATIVE THE STDEV
C     IS SET EQUAL TO -100.  IF YOU GET STDEV RESULTS=-100
C     IT PROBABLY MEANS THE ACTUAL STDEV IS VERY SMALL
C
C
      DO 110 I=1,INCHAN
C
      MEAN(I)=MEAN(I)/NPOINT
      STDEV(I)=STDEV(I)-NPOINT*MEAN(I)**2
      IF(STDEV(I).LT.0.) THEN
         STDEV(I)=-100.
         GO TO 105
      END IF
      STDEV(I)=DSQRT(STDEV(I)/(NPOINT-1.))
 105  WRITE(*,1000) NAME(I),MEAN(I),MIN(I),MAX(I),STDEV(I)
      WRITE(10,1000) NAME(I),MEAN(I),MIN(I),MAX(I),STDEV(I)
 110  CONTINUE
 1000 FORMAT(1X,A30,4(2X,G10.4))
C
C     PRINT DE-SPIKE INFORMATION IF USED
C
      IF(NYSPIK.NE.1) GO TO 130
      WRITE(*,1100)
      WRITE(10,1100)
 1100 FORMAT(//1X,'INFORMATION ABOUT SPIKE REMOVAL'/)
      WRITE(*,1110)
      WRITE(10,1110)
 1110 FORMAT(/5X,'CHANNEL',20X,'TOLERANCE',5X,'# SPIKES FOUND',
     +        2X,'# SPIKES CHANGED'/)
      DO 120 I=1,INCHAN
      WRITE(*,1120) NAME(I), TOLER(I), NFLAG(I), NSPIKE(I)
      WRITE(10,1120) NAME(I), TOLER(I),NFLAG(I), NSPIKE(I)
 1120 FORMAT(1X,A30,5X,G9.3,5X,I6,10X,I6)
 120  CONTINUE
C
 130  CONTINUE
C
      WRITE(*,1015) NPOINT
 1015 FORMAT(/,1X,'NUMBER OF DATA POINTS= ',F10.0)
      WRITE(*,1020) DELT
 1020 FORMAT(/ 5X,'TIME STEP (DELT)= ',F7.4)
      WRITE(*,1030) NSKIP
 1030 FORMAT(5X,'NUMBER OF SCANS SKIPPED AT START OF FILE= ',I8)
      WRITE(*,1040) ISCAN
 1040 FORMAT(5X,'NUMBER OF SCANS ANALYZED= ',I8)
C
      WRITE(10,1015) NPOINT
      WRITE(10,1020) DELT
      WRITE(10,1030) NSKIP
      WRITE(10,1040) ISCAN
```

```
      WRITE(*,*) 'A COPY OF THIS SUMMARY IS SAVED IN FILE FILSUM.DAT'
      CLOSE(10)
C
      CLOSE(25)
      STOP
C
C
 9999 WRITE(*,*) 'HIT END-OF-FILE GOING TO START POINT'
      STOP
      END
C
C     ***************************
C
      SUBROUTINE DESPIK(TOLER,INCHAN,ISCAN,NAME)
C
C     THIS SUBROUTINE PERFORMS DE-SPIKE OPERATION TO REMOVE DROP-OUTS
C     FROM THE RAW DATA.  THE ROUTINE IDENTIFIES SPIKES BY LOOKING
C     AT THE CHANGE BETWEEN SEQUENTIAL DATA POINTS ON EACH CHANNEL.
C     WHEN A SPIKE IS FOUND THE USER IS GIVEN OPTIONS FOR DEALING
C     WITH THAT SPIKE.  OPTIONS INCLUDE:
C       0- LEAVE THE VALUE UNCHANGED
C       1- CHANGE THE VALUE TO THE AVERAGE OF THE PREVIOUS 3 VALUES
C       2- CHANGE THE VALUE TO ONE EXTRAPOLATED FROM THE PREVIOUS
C          TWO VALUES (LINEAR EXTRAPOLATION)
C       3- MANUALLY ENTER A NEW VALUE FROM THE KEYBOARD
C
C     EACH OF THE FIRST 3 SPIKES ON EACH CHANNEL IS IDENTIFIED
INDIVIDUALLY
C     TO THE OPERATOR.  THE OPERATOR MUST ENTER INSTRUCTIONS FOR
HANDLING
C     THE SPIKE.  AFTER 3 SPIKES HAVE BEEN IDENTIFIED ON A CHANNEL
C     THE USER MUST DECIDE HOW TO DEAL WITH FUTURE SPIKES.  CHOICES ARE:
C       0- IGNORE ALL SUBSEQUENT SPIKES (NO CHANGES TO VALUES)
C       1- CHANGE ALL SUBSEQUENT SPIKES USING AVERAGING
C       2- CHANGE ALL SUBSEQUENT SPIKES USING EXTRAPOLATION
C       3- FLAG NEXT 3 SPIKES FOR OPERATOR ACTION
C
C     THE AZIMUTH CHANNEL IS HANDLED DIFFERENTLY BECAUSE OF THE EXPECTED
C     360 DEGREE DISCONTINUITY EVERY REVOLUTION.  INSTEAD OF LOOKING AT
THE
C     ABSOLUTE VALUE OF THE CHANGE IN TWO VALUES, THE POSITIVE INCREASE
C     IS COMPARED WITH THE TOLERANCE VALUE AND DECREASES IN AZIMUTH ARE
C     COMPARED WITH A CHANGE OF 360 DEGREES.  THOUGH WIND DIRECTION
C     CHANNELS CAN EXPERIENCE THE SAME DISCONTINUITY THIS SUBROUTINE
C     IGNORES THAT POSSIBILITY.  CAUTION MUST BE EXERCISED IF WORKING
C     WITH WIND DIRECTION DATA FLUCTUATING ABOUT THE 0 (360) DEGREE
POINT.
C     NOTE THIS SUBROUTINE ASSUMES AZIMUTH IS IN THE FIRST ELEMENT OF D.
C     THIS SAME ASSUMPTION IS MADE THROUGHOUT THIS SOFTWARE PACKAGE.
C
C     VARIABLES:
C       TOLER=ARRAY OF CHANGE TOLERANCES.  IF TWO CONSECUTIVE DATA
POINTS
C             ON ANY CHANNEL DIFFER (IN ABSOLUTE VALUE) BY A
VALUE>TOLER,
C             THEN THE SECOND DATA POINT WILL BE FLAGGED AS A SPIKE
C       OLDAT=2-D ARRAY THAT RETAINS THE LAST 3 VALUES OF EACH CHANNEL
C             OLDAT(J,3)=MOST RECENT VALUE
```

```
C            OLDAT(J,1)=OLDEST VALUE
C       NSPIKE=ARRAY CONTAINING COUNT OF NUMBER OF SPIKES FOUND AND
C              CHANGED ON EACH CHANNEL.  IF A SPIKE IS IDENTIFIED BUT
C              THE VALUE OF THE DATA POINT IS NOT CHANGED, THE SPIKE IS
C              NOT COUNTED.
C       NFLAG= ARRAY CONTAINING THE NUMBER OF SPIKES FLAGGED ON EACH
C              CHANNEL.  (WHETHER THE SPIKE WAS CHANGED OR NOT)
C       ISCAN=NUMBER OF CURRENT SCAN
C
C     SUBROUTINES CALLED BY DESPIK: CHANGE, DECIDE
C
C     C. HANSEN  1/87   UNIVERSITY OF UTAH
C
      COMMON D(30)
      COMMON/SPIKE/ NSPIKE(22),NFLAG(22),OLDAT(22,3),ITEM(30)
      INTEGER*4 ISCAN
      DIMENSION TOLER(INCHAN)
      CHARACTER*30 NAME(30)
C
C     CANNOT LOOK FOR A SPIKE UNTIL FOUR SCANS HAVE BEEN READ
C
      IF(ISCAN.LT.4) GO TO 900
C
C     CHECK CHANGES AGAINST TOLERANCES, AZIMUTH FIRST THEN OTHER CHANS.
C     APPLY CORRECTIONS TO ELIMINATE SPIKES AS DESIRED
C     IF A VALUE ON AN ACTIVE CHANNEL IS 0, THEN THE VALUE IS
C     FLAGGED AS A SPIKE REGARDLESS OF THE VALUE OF DELTA
C
      ICHK=0
      DELTA=D(1)-OLDAT(1,3)
      IF(ABS(DELTA) .GT. TOLER(1) .AND. D(1) .GT. TOLER(1)) ICHK=1
      IF(DELTA.GT.TOLER(1).OR.DELTA.LT.-360..OR.ICHK.EQ.1) THEN
         NFLAG(1)=NFLAG(1)+1
         CALL DECIDE(IOUT,ISCAN,1,NAME(1))
         IF(IOUT.GT.0) THEN
            CALL CHANGE(1,IOUT)
         ENDIF
      ENDIF
C
      DO 100 ICHAN=2,INCHAN
      IF(ABS(TOLER(ICHAN)).LT.1.E-9) GO TO 100
      DELTA=ABS(OLDAT(ICHAN,3)-D(ICHAN))
      IF(DELTA.GT.TOLER(ICHAN).OR.ABS(D(ICHAN)).LT.1.E-9) THEN
         NFLAG(ICHAN)=NFLAG(ICHAN)+1
         CALL DECIDE(IOUT,ISCAN,ICHAN,NAME(ICHAN))
         IF(IOUT.GT.0) THEN
            CALL CHANGE(ICHAN,IOUT)
         ENDIF
      ENDIF
  100 CONTINUE
C
C     UPDATE THE OLDAT ARRAY
C
      DO 800 J=1,INCHAN
      OLDAT(J,1)=OLDAT(J,2)
      OLDAT(J,2)=OLDAT(J,3)
      OLDAT(J,3)=D(J)
  800 CONTINUE
```

```
C
      RETURN
C
 900  GO TO (910,920,930) ISCAN
 910  DO 915 J=1,INCHAN
      OLDAT(J,1) = D(J)
 915  CONTINUE
      GO TO 950
C
 920  DO 925 J=1,INCHAN
      OLDAT(J,2) = D(J)
 925  CONTINUE
      GO TO 950
C
 930  DO 935 J=1,INCHAN
      OLDAT(J,3) = D(J)
 935  CONTINUE
C
 950  CONTINUE
      RETURN
      END
C
C     ***************************
C
      SUBROUTINE CHANGE(ICHAN,IOUT)
C
C     THIS ROUTINE APPLIES CHANGES TO ARRAY D BASED UPON VALUE OF IOUT
C
C     C HANSEN, 1/87,  UNIVERSITY OF UTAH
C
      COMMON D(30)
      COMMON/SPIKE/ NSPIKE(22),NFLAG(22),OLDAT(22,3),ITEM(30)
C
      GO TO (100,200,300,400) IOUT+1
 100  RETURN
C
C     USE AVERAGE OF PREVIOUS THREE VALUES FOR REPLACEMENT VALUE
C
 200  D(ICHAN)=(OLDAT(ICHAN,1)+OLDAT(ICHAN,2)+OLDAT(ICHAN,3))/3.
      NSPIKE(ICHAN)=NSPIKE(ICHAN) + 1
      RETURN
C
C     USE LINEAR INTERPOLATION FROM LAST TWO POINTS
C
 300  D(ICHAN)=2.*OLDAT(ICHAN,3)-OLDAT(ICHAN,2)
      NSPIKE(ICHAN)=NSPIKE(ICHAN) + 1
      RETURN
C
C     USE MANUALLY ENTERED VALUE
C
 400  WRITE(*,*) 'ENTER REPLACEMENT VALUE FOR CURRENT DATA POINT'
      READ(*,*) D(ICHAN)
      NSPIKE(ICHAN)=NSPIKE(ICHAN) + 1
      WRITE(*,1000) D(ICHAN)
 1000 FORMAT(1X,'CURRENT POINT REPLACED BY ',G11.4)
      RETURN
C
      END
```

```
C
C     ***************************
C
      SUBROUTINE DECIDE(IOUT,ISCAN,ICHAN,NAME)
C
C     THIS ROUTINE PROMPTS OPERATOR FOR DECISIONS REGARDING
C     DE-SPIKING OF DATA
C
C     C HANSEN, 1/87,  UNIVERSITY OF UTAH
C
      COMMON D(30)
      COMMON/SPIKE/ NSPIKE(22),NFLAG(22),OLDAT(22,3),ITEM(30)
      INTEGER*4 ISCAN, LASTSCN(22)
      DIMENSION NCHK(22),I3PT(22)
      CHARACTER*30 NAME
C
      DATA NCHK/22*3/, I3PT/22*-10/
C
      SAVE NCHK,I3PT,LASTSCN
C
      IF(NFLAG(ICHAN).GT.NCHK(ICHAN)) THEN
          IF(I3PT(ICHAN).EQ.-10) THEN
             WRITE(*,*) ' '
             WRITE(*,*) 'MORE THAN 3 SPIKES HAVE BEEN FOUND ON'
             WRITE(*,*)  NAME
             WRITE(*,*) 'ENTER YOUR DECISION FOR HANDLING FUTURE'
             WRITE(*,*) 'SPIKES ON THIS CHANNEL'
             WRITE(*,*) '0 = IGNORE ALL FUTURE SPIKES'
             WRITE(*,*) '1 = CHANGE ALL FUTURE SPIKES BY AVERAGING'
             WRITE(*,*) '2 = CHANGE ALL FUTURE SPIKES BY EXTRAPOLATION'
             WRITE(*,*) '3 = FLAG NEXT 3 SPIKES FOR OPERATOR ACTION'
             READ(*,*) I3PT(ICHAN)
          ENDIF
          GO TO (40,50,60,70) I3PT(ICHAN)+1
  40        IOUT=0
            RETURN
C
  50        IOUT=1
            LASTSCN(ICHAN)=ISCAN
            RETURN
C
  60        IOUT=2
            LASTSCN(ICHAN)=ISCAN
            RETURN
C
  70        NCHK(ICHAN)=NCHK(ICHAN)+3
            I3PT(ICHAN)=-10
      ENDIF
C
      WRITE(*,1000) ICHAN,NAME,ISCAN,NFLAG(ICHAN)
 1000 FORMAT(/1X,'DE-SPIKE ROUTINE, A POTENTIAL SPIKE LOCATED IN'/
     +       1X,'CHANNEL NUMBER      = ',I2,/
     +       1X,'CHANNEL NAME        = ',A30,/
     +       1X,'AT SCAN NUMBER      = ',I6/
     +       1X,'# SPIKES THIS CHAN. = ',I6)
C
      IF(LASTSCN(ICHAN)+1.EQ.ISCAN) WRITE(*,1020)
 1020 FORMAT(/1X,'    ******WARNING******'/
```

```
     +    1X,'TWO SCANS IN A ROW ARE FLAGGED AS SPIKES'/
     +    1X,'EXTRAPOLATION MAY RESULT IN ERRORS',/
     +    1X,'USE CAUTION IN DE-SPIKING THIS POINT')
C
      WRITE(*,1010) (OLDAT(ICHAN,I),I=1,3),D(ICHAN)
 1010 FORMAT(1X,'PREVIOUS VALUES (CHRONOLOGICAL ORDER):'/
     +       3(21X,G11.4,/),
     +       1X,'CURRENT DATA VALUE = 'G11.4,/)
  200 WRITE(*,*) 'ENTER YOUR DECISION FOR DISPOSITION OF SPIKE:'
      WRITE(*,*) '0 = LEAVE VALUE UNCHANGED'
      WRITE(*,*) '1 = REPLACE CURRENT VALUE WITH AVERAGE VALUE'
      WRITE(*,*) '2 = REPLACE CURRENT VALUE WITH EXTRAPOLATED VALUE'
      WRITE(*,*) '3 = ACCEPT NEW VALUE FROM KEYBOARD'
      READ(*,*) IOUT
C
C     ERROR TRAP
C
      IF(IOUT.NE.0.AND.IOUT.NE.1.AND.IOUT.NE.2.AND.IOUT.NE.3) THEN
        WRITE(*,*) 'INPUT VALUE MUST BE 0,1,2 OR 3'
        GO TO 200
      ENDIF
C
C     ECHO  RESPONSE
C
      GO TO (10,20,20,30) IOUT+1
   10 WRITE(*,*) 'VALUE WILL BE LEFT UNCHANGED'
      GO TO 100
   20 WRITE(*,*) 'NEW VALUE WILL BE CALCULATED'
      LASTSCN(ICHAN)=ISCAN
      GO TO 100
   30 WRITE(*,*) 'NEW VALUE WILL BE ENTERED FROM KEYBOARD'
      LASTSCN(ICHAN)=ISCAN
  100 CONTINUE
C
      RETURN
      END
C
C     ***************************
C
      SUBROUTINE GSPIKE(TOLER,INCHAN,ITEM,NAME)
C
C       SUBROUTINE OPENS, READS, AND ALLOWS CHANGING OF THE
C       'GSPIKE' FILE.  'GSPIKE' CONTAINS THE TOLERANCES
C       THAT WILL BE APPLIED TO EACH CHANNEL OF DATA.
C
C       EACH CHANNEL CAN BE DE-SPIKED WITH A DIFFERENT TOLERANCE
C
C       VARIABLES:  TOLER=ARRAY OF TOLERANCES (FIRST ELEMENT
C                       CORRESPONDS TO FIRST CHANNEL, ETC.)
C                   NAME=ARRAY CONTAINING ITEM NAMES
C                   NCHAN=NUMBER OF ACTIVE CHANNELS (AS IN MAIN PROGRAM)
C       C HANSEN   1/87, UNIVERSITY OF UTAH
C
      DIMENSION TOLER(1),ITEM(1)
      CHARACTER*30 NAME(1)
C
C     OPEN DATA FILE CONTAINING TOLERANCE INFORMATION
C
```

```
      OPEN(UNIT=23,FILE='TOLER.DAT',STATUS='OLD',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
            WRITE(*,*) 'ERROR OPENING UNIT 23, FILE TOLER.DAT'
            WRITE(*,*)  'IERR= ',IERR
      END IF
C
C       READ FILE AND DISPLAY ON CRT
C
      REWIND 23
      DO 50 I=1,INCHAN
      READ(23,*) TOLER(I)
   50 CONTINUE
      REWIND 23
C
      WRITE(*,*) ' '
      WRITE(*,*) 'SELECT TOLERANCES FOR DE-SPIKING OF DATA'
      WRITE(*,*) 'BE CERTAIN UNUSED CHANNELS HAVE TOLERANCE=0.0'
      WRITE(*,*) 'IF YOU DONT WANT TO DESPIKE A CHANNEL USE',
     +   ' TOLER=0.0'
      WRITE(*,*) ' '
  200 WRITE(*,*) 'CHAN            ITEM                        TOLERANCE'
      DO 10 I=1,INCHAN
      WRITE(*,100) I,NAME(I),TOLER(I)
  100 FORMAT(2X,I2,3X,A30,5X,G10.4)
   10 CONTINUE
   20 WRITE(*,*) 'ENTER CHANNEL # YOU WANT TO CHANGE (0=NO CHANGE)'
      READ*,ICHAN
      IF(ICHAN.NE.0) THEN
   70    WRITE(*,*) 'ENTER NEW TOLERANCE (ENGINEERING UNITS)'
         READ*, TOLER(ICHAN)
         GO TO 200
      END IF
C
C     REWRITE THE TOLER.DAT FILE, STORING CHANGES
C
      DO 60 I=1,INCHAN
      WRITE(23,*) TOLER(I)
   60 CONTINUE
      ENDFILE 23
C
C       CLOSE FILE AND RETURN
C
      CLOSE(23)
      RETURN
      END
C
C     ************************
C
      SUBROUTINE GITEMS(ITEM,NITEMS,NAME,NCHAN)
C
C     READS AND EDITS THE FILE 'ITEMS', CONTAINING
C     CHANNEL IDENTIFICATIONS FOR THE NEFF/AZIMUTH
C     AVERAGING PROGRAM
C
C     A CHANNEL NUMBER IS THE A/D CONVERTER CHANNEL
C     EXCLUDING THE TIME CHANNELS.
C     NOTE THAT AZIMUTH MUST BE PRESENT
C     CHANNELS 1 THROUGH 22 ARE RESERVED FOR MEASURED QUANTITIES
```

```
C     CHANNELS 23 THRU 30 ARE RESERVED FOR CALCULATED
C     QUANTITIES
C
C     AN ITEM NUMBER SIMPLY IDENTIFIES A PARTICULAR QUANTITY
C     FROM THE LIST OF NAMES BELOW.
C        C. HANSEN   6/85
C
      DIMENSION ITEM(1)
      CHARACTER*30 NAME(1)
C
C     READ NAMES FROM NAMES.DAT FILE
C
      OPEN(UNIT=11,FILE='NAMES.DAT',STATUS='OLD',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
      WRITE(*,*) 'ERROR OPENING NAMES.DAT FILE (UNIT 11)'
      WRITE(*,*) 'IERR= ',IERR
      STOP
      ENDIF
C
      DO 5 I=1,NITEMS
      READ(11,*) NAME(I)
   5  CONTINUE
      CLOSE(11)
C
C     OPEN 'ITEMS' FILE FOR CHANNEL IDENTIFICATION
C
      OPEN(UNIT=21,FILE='ITEMS.DAT',STATUS='UNKNOWN',IOSTAT=IERR)
      IF (IERR.NE.0) THEN
         WRITE(*,*) 'ERROR ON OPENING ITEMS FILE'
         WRITE(*,*) 'IERR= ',IERR
         STOP
      ENDIF
      DO 10 I=1,NITEMS
      READ(21,100) ITEM(I)
 100  FORMAT(I2)
 10   CONTINUE
C
      REWIND 21
C
 200  WRITE(*,*)  'CHANNEL IDENTIFICATIONS'
      WRITE(*,*)  'CHANNEL   ITEM       NAME'
      DO 20 I=1,22
      IF(ITEM(I).EQ.0) THEN
         WRITE(*,110) I,ITEM(I),'CHANNEL NOT USED'
         GO TO 20
      ENDIF
      WRITE(*,110) I,ITEM(I), NAME(ITEM(I))
 110  FORMAT(5X,I2,5X,I2,5X,A30)
 20   CONTINUE
C
 40   WRITE(*,*) 'ENTER CHANNEL # YOU WANT TO CHANGE (0=NO CHANGE)'
      READ*,ICHAN
      IF(ICHAN.NE.0) THEN
        WRITE(*,*) 'ENTER NEW ITEM NUMBER (0=THIS ITEM NOT RECORDED)'
        READ*,ITEM(ICHAN)
        GO TO 200
      END IF
C
```

```
C     REPLACE OLD ITEMS FILE WITH NEW VALUES
C
      DO 30 I=1,NITEMS
      WRITE(21,100) ITEM(I)
 30   CONTINUE
      ENDFILE 21
C
C     CLOSE ITEMS FILE AND RETURN TO MAIN PROGRAM
C
      RETURN
      END
```

```
      PROGRAM PREDAT
C
C     THIS PROGRAM IS THE THIRD OF SEVERAL PROGRAMS FOR PROCESSING
C     AZIMUTH AVERAGE TEST DATA FROM ASCII DATA TAPES.
C     THE FIRST PROGRAM IS 'READTAPE'.  THAT PROGRAM TRANSFERS DATA
C     FROM AN ASCII MAG TAPE TO A DISC FILE ON THE VAX.
C
C     THE SECOND PROGRAM, PREP, IS USED TO CONDITION THE DATA FILE
C     AND REMOVE SPIKES FROM THE DATA
C
C     PREDAT READS THE DISC FILE AND ALLOWS
C     VARIOUS TYPES OF PROCESSING BEFORE CREATING A NEW DISC FILE
C     CALLED 'AZIN'.  THE OTHER PROGRAMS RUN BINS, HARMONICS, ETC.
C
C     DETAILS ON 'PREDAT' FOLLOW:
C     THE PROGRAM PERFORMS PRE-PROCESSING OPERATIONS ON THE
C     DATA FILE 'PREDATIN.DAT'.  THE DATA MUST BE EQUAL TIME INTERVAL
C     SAMPLES WHICH INCLUDE ROTOR AZIMUTH. OPTIONAL
C     CHANNELS OF DATA INCLUDE POWER, WIND SPEED,YAW, WIND DIRECTION,
C     RPM, VARIOUS STRAIN GAGES, TEETER ANGLE AND OTHER ANEMOMETERS.
C     PREDATIN.DAT IS THE FILE CREATED BY PROGRAM 'PREP'
C
C     AFTER READING THE D ARRAY, THE PROGRAM PERMITS LOW-PASS
C     FILTERING OF THE DATA.  THIS FILTERING CAN BE USED TO IMPROVE
C     CORRELATION BETWEEN WIND AND MACHINE PARAMETERS.  THE SUBROUTINE
C     GFILT ACCESSES AND EDITS A DATA FILE CONTAINING THE DESIRED
C     FILTER BANDWIDTHS FOR EACH DATA CHANNEL.  EACH CHANNEL CAN
C     BE FILTERED AT A DIFFERENT FREQUENCY, OR NOT AT ALL.
C     THE FILTERED DATA IS THEN USED TO COMPUTE YAW RATE AND YAW ERROR
C     (SUBROUTINES YAWRAT AND YAWERR).  DISC AVERAGED WIND SPEED AND
C     DIRECTION ARE ALSO COMPUTED, IF DESIRED.  IF DISC AVERAGING IS
C     NOT USED THE HUB HEIGHT ANEMOMETER DATA IS PLACED IN THE DISC
C     AVERAGE ROW OF THE DATA ARRAY.  HORIZONTAL AND VERTICAL WIND SHEAR
C     ARE ALSO CALCULATED, IF DESIRED.
C     AFTER FILTERING THE DATA IS WRITTEN TO THE DISC FILE 'AZIN'
C     FOR LATER PROCESSING BY THE AZBINS AND OTHER PROGRAMS.
C
C     PREDAT KEEPS TRACK OF THE MEAN, MINIMUM, MAXIMUM, AND STANDARD
C     DEVIATION OF EACH CHANNEL AND PRINTS A SUMMARY FOR THE INFORMATION
C     OF THE USER.  THE SUMMARY IS OF THE FILTERED DATA, NOT THE RAW
C     DATA.
C
C
C     VARIABLES:
C        TITLE=A USER SUPPLIED TEXT (UP TO 80 CHARS.) THAT
C              IDENTIFIES THE TEST
C        INCHAN=MAX. NUMBER OF INPUT DATA CHANNELS (EXCL. TIME CODE)
C        NCHAN=ACTUAL NUMBER OF ACTIVE CHANNELS OF
C              DATA (EXCLUDING TIME CODE)
C        DELT=TIME INTERVAL OF RAW DATA (SEC)
C        NSCAN=NUMBER OF SCANS IN THE ORIGINAL DATA SET
C        NSKIP=NUMBER OF SCANS SKIPPED BEFORE ANALYSIS BEGINS
C        NRUN= NUMBER OF SCANS THAT WILL BE ANALYZED
C        NDROP=NUMBER OF POINTS DELETED AT BEGINNING AND END
C              OF FILTERED DATA FILE. SET AS PARAMETER.
C              (TO ELIMINATE FILTER STARTUP TRANSIENTS)
C        BW=LOW-PASS FILTER CUTOFF FREQUENCY (-6 DB) ARRAY, IF
C              BW(I)=0 THEN CHANNEL 'I' IS NOT FILTERED.
```

```
C        ITEM=ARRAY OF CHANNEL ID'S.  FOR EXAMPLE, IF WIND SPEED (ITEM
2)
C              IS ON TAPE CHANNEL 5, THEN ITEM(5)=2
C        MIN,MEAN,MAX,STDEV=ARRAYS OF SUMMARY INFO FOR EACH CHANNEL
C              OF FILTERED DATA
C
C     NPASS IS THE NUMBER OF SCANS USED IN ONE FILLING OF THE D ARRAY
C     THE D ARRAY MUST HAVE A SECOND DIMENSION EQUAL OR GREATER THAN
C     NPASS.  THE DATA STATEMENT ASSIGNING D TO ZERO MUST ALSO BE
C     CONSISTENT WITH NPASS.  NOTE THAT CHANGES OF THE DIMENSION
C     OF ARRAY D IN THE MAIN PROGRAM MUST BE ACCOMPANIED BY THE SAME
C     CHANGES IN THE SUBROUTINES.  USE OF A LARGE VALUE OF NPASS
C     IS DESIRABLE TO MINIMIZE END EFFECTS DURING FILTERING AND
C     YAW RATE CALCULATIONS.
C
C     SUBROUTINES:
C        AVGPR, AVINFO
C        DISCAV, HORSHR, VERSHR
C        YAWERR, YAWRAT
C        LPFILT, LPSB
C        GFILT, TURBLNC
C
C     WRITTEN BY C HANSEN  7/85
C     MODIFIED 2/86,11/86,1/87,8/87 BY C HANSEN
C     NO DOCUMENTATION OF THIS PROGRAM EXISTS OTHER THAN THE LISTING.
C
      PARAMETER (NITEMS=30)
      PARAMETER (INCHAN=22)
      PARAMETER (NPASS=1000)
C
      INTEGER*4 NSCAN,ISCAN,NSKIP,NRUN,IFIL,NFILT
      INTEGER HSHEAR,VSHEAR
      INTEGER FILTYP
      CHARACTER*30 NAME(NITEMS)
      CHARACTER*80 TITLE
      DIMENSION ITEM(NITEMS)
      REAL MIN(NITEMS), MAX(NITEMS)
      DOUBLE PRECISION MEAN(NITEMS), STDEV(NITEMS), NPOINT
      COMMON D(NITEMS,NPASS)
      COMMON/WNDAV/ NSANEM,NDANEM,ISANEM(5),IDANEM(5),HSHEAR,VSHEAR,
     +  IWSH1,IWSH2,IWSV1,IWSV2,NOUTPT
      COMMON/FILTER/ XFILT,YFILT,BZERO(30),A1(30,3),
     +               A2(30,3),BW(30),DELT,ICHAN
C
      DATA MIN/NITEMS*1.E20/,MAX/NITEMS*-1.E20/,MEAN/NITEMS*0./
      DATA STDEV/NITEMS*0.0D00/, NPOINT/0.0D00/
C
      NDROP=100
      HSHEAR=0
      VSHEAR=0
      NSANEM=0
      NDANEM=0
C
      DO 1 I=1,5
      ISANEM(I)=0
      IDANEM(I)=0
    1 CONTINUE
C
```

```
      DO 11 I=1,NPASS
      DO 12 J=1,NITEMS
      D(J,I)=0.0
 12   CONTINUE
 11   CONTINUE
C
C     END OF FILE TEST FLAG
C
      IEND=0
C
C     OPEN 'FILINFO' FILE TO GET FILE INFORMATION
C
      OPEN(UNIT=30,FILE='FILINFO.DAT',STATUS='OLD',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING FILINFO.DAT FILE (UNIT 30)'
         WRITE(*,*) 'IERR= ',IERR
         STOP
      END IF
      READ(30,2000) TITLE
 2000 FORMAT(A)
      READ(30,*) NCHAN
      READ(30,*) NSCAN
      READ(30,*) DELT
      CLOSE(30)
C
C     OPEN 'PREDATIN.DAT' FILE WHICH CONTAINS ALL OF THE DATA
C
      OPEN(UNIT=20,FILE='PREDATIN.DAT',ACCESS='DIRECT',
     +     RECL=INCHAN,STATUS='OLD',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
         WRITE(*,*)  'ERROR OPENING PREDATIN.DAT (UNIT 20)'
         WRITE(*,*)  'IERR= ',IERR
         STOP
      ENDIF
C
C     OPEN UNIT 25 (FILE AZIN) TO RECEIVE PROCESSED TIME SERIES DATA
C
      OPEN(UNIT=25,FILE='AZIN.DAT',STATUS='NEW',
     +     FORM='UNFORMATTED',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
         WRITE(*,*)  'ERROR OPENING AZIN.DAT (UNIT 25)'
         WRITE(*,*)  'IERR= ',IERR
         STOP
      ENDIF
      REWIND 25
C
C     OPEN UNIT 99 AS A SCRATCH FILE TO HOLD FILTERED DATA
C     THIS IS A DIRECT ACCESS, UNFORMATTED FILE
C
      OPEN(UNIT=99,ACCESS='DIRECT',RECL=INCHAN,STATUS='SCRATCH',
     +     IOSTAT=IERR)
      IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING FILTER SCRATCH FILE (UNIT 99)'
         WRITE(*,*) 'IERR= ',IERR
         STOP
      ENDIF
C
C     COUNTER FOR NUMBER OF SCANS FILTERED (USED IN WRITING
```

```
C      AND READING THE DIRECT ACCESS FILE)
C
       NFILT=0
       ISCAN=0
C
       WRITE(*,*) 'THIS IS THE AZIMUTH AVERAGING PRE-PROCESS PROGRAM'
       WRITE(*,*) ' '
  420  WRITE(*,*) 'ENTER PARAMETER FOR FILTER TYPE'
       WRITE(*,*) '0=NO FILTERING'
       WRITE(*,*) '1=LOW-PASS SINE BUTTERWORTH'
       WRITE(*,*) '2=RUNNING AVERAGING'
       READ(*,*) FILTYP
       IF(FILTYP.NE.0.AND.FILTYP.NE.1.AND.FILTYP.NE.2) THEN
          WRITE(*,*) 'FILTER TYPE PARAMETER MUST BE 0,1 OR 2'
          GO TO 420
       ENDIF
C
C      DETERMINE CHANNEL IDENTIFICATION NUMBERS AND NAMES
C
       CALL GIDS(ITEM,NITEMS,NAME,NCHAN)
C
C      DETERMINE FILTER BANDWIDTHS IF NECESSARY
C
       IF(FILTYP.EQ.0) THEN
          NDROP=0
          GO TO 400
       ENDIF
       CALL GFILT(BW,DELT,NITEMS,ITEM,NAME,NCHAN,INCHAN)
C
C      GET FILTER COEFFICIENTS
C
       DO 6 ICHAN=1,INCHAN
       IF(BW(ICHAN).EQ.0.0) GO TO 6
       CALL LPSB
    6  CONTINUE
C
C      GET WIND SPEED AND DIRECTION AVERAGING INFORMATION
C
  400  CONTINUE
       CALL AVINFO(NAME)
C
       WRITE(*,*) ' '
       WRITE(*,*) 'THE DATA FILE CONTAINS ',NSCAN,' SCANS'
       WRITE(*,*) 'AND ',NCHAN,' CHANNELS OF DATA'
       WRITE(*,*) ' '
       WRITE(*,*) 'NUMBER OF SCANS TO SKIP? (0=FIRST RECORD AVAILABLE)'
       READ*,NSKIP
       WRITE(*,*) 'NUMBER OF SCANS TO ANALYZE?'
       READ*, NRUN
       WRITE(*,*) 'THE TURBULENCE INTENSITY AT HUB HEIGHT WILL'
       WRITE(*,*) 'BE CALCULATED FOR A SPECIFIED AVERAGING TIME'
       WRITE(*,*) 'ENTER THE AVERAGING TIME YOU WISH TO USE (SEC)'
       READ(*,*) TURBTIM
C
C      SKIP FILTERING SECTION IF DESIRED
C
       IF(FILTYP.EQ.0) GO TO  410
C
```

```
      WRITE(*,*) 'STARTING FILTERING.......'
C
C     READ FIRST SCAN OF DATA TO INITIALIZE FILTERS
C
      READ(20,REC=NSKIP+1) (D(I,1),I=1,INCHAN)
      DO 7 ICHAN=1,INCHAN
      XFILT=D(ICHAN,1)
      CALL LPFILT
    7 CONTINUE
C
C     SAMPLE DATA UNTIL THE STOP TIME
C
C     READ IN A SCAN OF DATA
C
    5 ISCAN=ISCAN+1
      IF(ISCAN.GT.NSCAN) IEND=1
      READ(20,REC=NSKIP+ISCAN) (D(J,1),J=1,INCHAN)
  100 CONTINUE
C
C     FILTER EACH CHANNEL OF DATA WITH ORDER 6 SINE BUTTERWORTH
C     LOW PASS FILTER WITH CUTOFF FREQUENCY BW
C     BUT IF BW=0, DONT FILTER THAT CHANNEL
C
      DO 35 ICHAN=2,INCHAN
      IF(BW(ICHAN).EQ.0.)GO TO 35
      XFILT=D(ICHAN,1)
      CALL LPFIL2
      D(ICHAN,1)=YFILT
   35 CONTINUE
C
C     WRITE THE FILTERED SCAN FOR TEMPORARY STORAGE UNTIL REVERSE
C     FILTERING CAN BE COMPLETED
C
      NFILT=NFILT+1
      WRITE(99,REC=NFILT) (D(IW,1),IW=1,INCHAN)
C
C     RETURN FOR MORE DATA
C
      IF(ISCAN.LT.NRUN.AND.IEND.EQ.0) GO TO 5
C
C     APPLY FILTER IN REVERSE TO REMOVE PHASE SHIFT
C
C     INITIALIZE FILTER ON LAST DATA SCAN (FIRST SCAN IN REVERSE)
C
      READ(99,REC=NFILT) (D(IW,1),IW=1,INCHAN)
      DO 240 ICHAN=1,INCHAN
      XFILT=D(ICHAN,1)
      CALL LPFILT
  240 CONTINUE
C
C     READ FILTERED DATA FROM SCRATCH FILE, THEN FILTER AGAIN
C
      DO 200 IFIL=NFILT,1,-1
      READ(99,REC=IFIL) (D(IW,1),IW=1,INCHAN)
C
      DO 210 ICHAN=1,INCHAN
      IF(BW(ICHAN).EQ.0.0) GO TO 210
      XFILT=D(ICHAN,1)
```

```
      CALL LPFIL2
      D(ICHAN,1)=YFILT
 210  CONTINUE
C
C     REPLACE DATA WITH TWICE-FILTERED DATA
C
      WRITE(99,REC=IFIL) (D(IW,1),IW=1,INCHAN)
 200  CONTINUE
C
C     FILTERING COMPLETED, NOW READ AND PROCESS DATA
C
      WRITE(*,*) 'FILTERING COMPLETED.......'
 410  CONTINUE
      IFIL=NDROP
 320  ICOUNT=0
      DO 300 I=1,NPASS
      IFIL=IFIL+1
      IF(FILTYP.EQ.0) THEN
         READ(20,REC=IFIL+NSKIP) (D(IW,I),IW=1,INCHAN)
      ELSE
         READ(99,REC=IFIL) (D(IW,I),IW=1,INCHAN)
      ENDIF
      ICOUNT=ICOUNT+1
      NPOINT=NPOINT+1.
      IF(IFIL.EQ.NRUN-NDROP) GO TO 310
 300  CONTINUE
C
 310  CONTINUE
C
C     COMPUTE WIND AVERAGES, TURBULENCE AND SHEARS, IF DESIRED
C
      CALL DISCAV(ICOUNT)
      CALL TURBLNC(TURBTIM,DELT,NPOINT,ICOUNT)
C
      IF(HSHEAR.EQ.1) CALL HORSHR(ICOUNT)
      IF(VSHEAR.EQ.1) CALL VERSHR(ICOUNT)
C
C     COMPUTE YAW RATE AND YAW ERROR FOR LAST BLOCK OF DATA
C
      CALL YAWRAT(DELT,ICOUNT)
      CALL YAWERR(ICOUNT)
C
C     WRITE TIME SERIES TO UNIT 25
C
      DO 50 I5=1,ICOUNT
      WRITE(25) (D(I7,I5),I7=1,NITEMS)
 50   CONTINUE
C
C
C     FIND MIN,MAX,ETC FOR LAST BLOCK OF DATA
C
      DO 60 I=1,ICOUNT
C
      DO 40 I1=1,NITEMS
      IF(D(I1,I).LT.MIN(I1)) MIN(I1)=D(I1,I)
      IF(D(I1,I).GT.MAX(I1)) MAX(I1)=D(I1,I)
      MEAN(I1)=MEAN(I1) + D(I1,I)
      STDEV(I1)=STDEV(I1) + D(I1,I)*D(I1,I)
```

```
 40   CONTINUE
 60   CONTINUE
C
C     RETURN FOR MORE DATA UNTIL END OF DATA
C
      IF(IFIL.LT.NRUN-NDROP) GO TO 320
C
C     CLOSE PREDATIN FILE AND SCRATCH FILE
C
      CLOSE(20)
      CLOSE(99)
C
C     CREATE THE AZINFO FILE TO SHOW ACTUAL # DATA POINTS PROCESSED
C
      OPEN(UNIT=30,FILE='AZINFO.DAT',STATUS='NEW',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING AZINFO.DAT FILE (UNIT 30)'
         WRITE(*,*) 'IERR = ',IERR
         STOP
      ENDIF
C
      WRITE(30,2000)TITLE
      WRITE(30,*) NCHAN
      WRITE(30,*) NRUN-2*NDROP
      WRITE(30,*) DELT
      CLOSE(30)
C
C     WRITE SUMMARY INFORMATION TO CRT AND DISC FILE 'DATSUM.DAT'
C
      OPEN(UNIT=10,FILE='DATSUM.DAT',STATUS='NEW')
      WRITE(*,1005) TITLE
      WRITE(10,1003)
 1003 FORMAT(1X,'SUMMARY OF DATA FROM PROGRAM PREDAT')
      WRITE(10,1005) TITLE
 1005 FORMAT(//,1X,A80/)
      CALL AVGPR(NAME)
      WRITE(*,1007) TURBTIM
      WRITE(10,1007) TURBTIM
 1007 FORMAT(1X,'TURBULENCE INTENSITY IS MOVING AVERAGE OVER A'
     +       ,F6.1,' SECOND PERIOD',/)
      WRITE(*,1010)
      WRITE(10,1010)
 1010 FORMAT(6X,'ITEM',24X,'MEAN',7X,'MIN',7X,'MAX',5X,'STDEV',
     +  4X,'FILTER')
C
C     COMPUTE FINAL MEANS AND STANDARD DEVIATIONS
C
C     NOTE THAT IF SMALL DIFFERENCES OF LARGE NUMBERS ARE
C     USED FOR STANDARD DEVIATION CALCULATION, ROUND-OFF
C     ERRORS WILL RESULT IN INVALID STDEV.  IF THE
C     ARGUMENT OF THE SQUARE ROOT IS NEGATIVE THE STDEV
C     IS SET EQUAL TO -100.  IF YOU GET STDEV RESULTS=-100
C     IT PROBABLY MEANS THE ACTUAL STDEV IS VERY SMALL
C
C
      DO 110 I=1,NITEMS

      MEAN(I)=MEAN(I)/NPOINT
```

```
      STDEV(I)=STDEV(I)-NPOINT*MEAN(I)**2
      IF(STDEV(I).LT.0.) THEN
         STDEV(I)=-100.
         GO TO 105
      END IF
      STDEV(I)=DSQRT(STDEV(I)/(NPOINT-1.))
 105  WRITE(*,1000) NAME(I),MEAN(I),MIN(I),MAX(I),STDEV(I),BW(I)
      WRITE(10,1000) NAME(I),MEAN(I),MIN(I),MAX(I),STDEV(I),BW(I)
 110  CONTINUE
 1000 FORMAT(1X,A30,5(1X,G9.3))
C
C
      WRITE(*,1015) NPOINT
 1015 FORMAT(/,1X,'NUMBER OF DATA POINTS= ',F10.0)
      WRITE(*,1020) DELT
 1020 FORMAT(/ 5X,'TIME STEP (DELT)= ',F7.4)
      WRITE(*,1030) NSKIP
 1030 FORMAT(5X,'NUMBER OF SCANS SKIPPED AT START OF FILE= ',I8)
      WRITE(*,1050) NDROP
 1050 FORMAT(5X,'NUMBER OF SCANS DELETED AT BEGINNING',/
     +       5X,'    AND END OF FILE= ',I3)
      WRITE(*,1040) NRUN-2*NDROP
 1040 FORMAT(5X,'NUMBER OF SCANS ANALYZED= ',I8)
C
      WRITE(10,1015) NPOINT
      WRITE(10,1020) DELT
      WRITE(10,1030) NSKIP
      WRITE(10,1050) NDROP
      WRITE(10,1040) NRUN-2*NDROP
      WRITE(*,*) 'A COPY OF THIS SUMMARY IS SAVED IN FILE DATSUM.DAT'
C
      CLOSE(10)
      CLOSE(25)
      STOP
      END
```

```
C     A COLLECTION OF THE SUBROUTINES REQUIRED BY PROGRAM PREDAT
C     SUBROUTINES IN THIS FILE INCLUDE:
C       AVGPR
C       AVINFO
C       DISCAV
C       GFILT
C       GIDS
C       HORSHR
C       LPFILT
C       LPSB
C       VERSHR
C       YAWERR
C       YAWRAT
C       TURBLNC
C
C     COMPILATION MADE 11/86,REVISED 8/87, UNIVERSITY OF UTAH
C
C     **********************
C
      SUBROUTINE AVGPR(NAME)
C
C     THIS SUBROUTINE PRINTS INFORMATION ABOUT THE DISC AVERAGING
C       NOUTPT IS THE UNIT OR DEVICE NUMBER TO RECEIVE THE OUTPUT
C
      INTEGER HSHEAR, VSHEAR
      CHARACTER*30 NAME(1)
      CHARACTER*61 DUMMY
      COMMON/WNDAV/ NSANEM,NDANEM,ISANEM(5),IDANEM(5),HSHEAR,VSHEAR,
     +  IWSH1,IWSH2,IWSV1,IWSV2,NOUTPT
C
 100  FORMAT(5X,A30)
C
      IF(NSANEM.GT.0) THEN
        WRITE(NOUTPT,*)'WIND SPEEDS FROM THE FOLLOWING ANEMOMETERS ',
     +                 'WERE USED IN THE DISC AVERAGE'
        DO 10 I=1,NSANEM
        WRITE(NOUTPT,100) NAME(ISANEM(I))
  10    CONTINUE
        GO TO 20
      ENDIF
      WRITE(NOUTPT,*)'NO WIND SPEED DISC AVERAGING WAS USED'
  20  IF(NDANEM.GT.0) THEN
        WRITE(NOUTPT,*)'WIND DIRECTIONS FROM THE FOLLOWING ',
     +                 'ANEMOMETERS WERE USED IN THE DISC AVERAGE'
        DO 30 I=1,NDANEM
        WRITE(NOUTPT,100) NAME(IDANEM(I))
  30    CONTINUE
        GO TO 40
      ENDIF
      WRITE(NOUTPT,*)'NO WIND DIRECTION DISC AVERAGING WAS USED'
  40  CONTINUE
C
C     NOW PRINT WIND SHEAR CALCULATION INFORMATION
C
      IF(HSHEAR.EQ.0) GO TO 50
      L=INDEX(NAME(IWSH1),'  ')
      DUMMY=NAME(IWSH1)(1:L)
      DUMMY=DUMMY(1:L)//'- '//NAME(IWSH2)
```

```
      WRITE(NOUTPT,*) 'HORIZONTAL SHEAR = ',DUMMY
      GO TO 60
   50 WRITE(NOUTPT,*) 'HORIZONTAL SHEAR NOT CALCULATED'
C
   60 IF(VSHEAR.EQ.0) GO TO 70
      L=INDEX(NAME(IWSV1),'  ')
      DUMMY=NAME(IWSV1)(1:L)
      DUMMY=DUMMY(1:L)//'- '//NAME(IWSV2)
      WRITE(NOUTPT,*) 'VERTICAL SHEAR = ',DUMMY
      WRITE(NOUTPT,*)' '
      GO TO 80
   70 WRITE(NOUTPT,*) 'VERTICAL SHEAR NOT CALCULATED'
   80 CONTINUE
C
      RETURN
      END
C
C     ************************
C
      SUBROUTINE AVINFO(NAME)
C
C     SUBROUTINE TO READ DATA FILE CONTAINING CONTROL INFORMATION
C     FOR DISC AVERAGING.
C
C     VARIABLES:
C     NSANEM=NUMBER OF ANEMOMS.  TO BE USED IN SPEED AVERAGING
C     NDANEM=NUMBER OF ANEMOMS.  TO BE USED IN DIRECTION AVERAGING
C     ISANEM=ARRAY CONTAINING ITEM #'S FOR SPEED AVERAGE
C     IDANEM=ARRAY CONTAINING ITEM #'S FOR DIRECTION AVERAGE
C     HSHEAR,VSHEAR=1 FOR HORIZ. FOR VERTICAL SHEAR CALC, 0 OTHERWISE
C     IWSH1,IWSH2=ITEM #'S FOR HORIZONTAL SHEAR ANEMOMETERS
C     IWSV1,IWSV1=ITEM #'S FOR VERTICAL SHEAR ANEMOMETERS
C       NOUTPT=UNIT NUMBER TO RECEIVE OUTPUT
C
      INTEGER HSHEAR, VSHEAR
      CHARACTER*30 NAME(1)
      COMMON/WNDAV/ NSANEM,NDANEM,ISANEM(5),IDANEM(5),HSHEAR,VSHEAR,
     +  IWSH1,IWSH2,IWSV1,IWSV2,NOUTPT
C
      OPEN(UNIT=10,FILE='AVGINF.DAT',STATUS='OLD',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
          WRITE(*,*) 'ERROR OPENING AVGINF.DAT FILE (UNIT 10)'
          WRITE(*,*) 'IERR= ',IERR
          STOP
      ENDIF
C
      READ(10,*) NSANEM
      IF(NSANEM.EQ.0) GO TO 100
      DO 10 I=1,NSANEM
      READ(10,*) ISANEM(I)
   10 CONTINUE
C
  100 CONTINUE
C
      READ(10,*) NDANEM
      IF(NDANEM.EQ.0) GO TO 110
      DO 20 I=1,NDANEM
      READ(10,*) IDANEM(I)
```

```
  20  CONTINUE
C
 110  CONTINUE
C
C     GET WIND SHEAR INFORMATION
C
      READ(10,*) HSHEAR
      IF(HSHEAR.EQ.1) THEN
          READ(10,*) IWSH1,IWSH2
      ENDIF
C
      READ(10,*) VSHEAR
      IF(VSHEAR.EQ.1) THEN
          READ(10,*) IWSV1,IWSV2
      ENDIF
C
      CLOSE(10)
C
C     DISPLAY AVERAGING INFORMATION AND GIVE OPPORTUNITY TO ABORT RUN
C
      NOUTPT=6
      CALL AVGPR(NAME)
C
      WRITE(*,*) 'IF THIS AVERAGING INFORMATION IS CORRECT, ENTER 1'
      WRITE(*,*) 'IF IT IS INCORRECT, ENTER 0.  THIS WILL ABORT'
      WRITE(*,*) 'THIS RUN SO YOU CAN EDIT THE AVGINF.DAT FILE'
      WRITE(*,*) ' '
      WRITE(*,*) 'ENTER 1 TO CONTINUE, 0 TO ABORT'
      READ(*,*) IANS
      IF(IANS.NE.1) STOP
C
      RETURN
      END
C
C     *************************
C
      SUBROUTINE DISCAV(ICOUNT)
C
C     SUBROUTINE TO COMPUTE THE ARITHMETIC AVERAGE OF SEVERAL WIND
C     SPEEDS AND DIRECTIONS.  THE AVERAGE VALUES ARE RETURNED AS
C     ELEMENT OF THE D ARRAY
C
C     C HANSEN 12/85
C
      INTEGER HSHEAR, VSHEAR
      COMMON D(30,1000)
      COMMON/WNDAV/ NSANEM,NDANEM,ISANEM(5),IDANEM(5),HSHEAR,VSHEAR,
     +  IWSH1,IWSH2,IWSV1,IWSV2
C
C       ITEM NUMBERS FOR OUTPUT OF RESULTS
C
      NWS=25
      NWD=26
      NWSHUB=2
      NWDHUB=3
C
C     COMPUTE AVERAGE WIND SPEED
C     IF AVERAGING IS NOT DESIRED, PLACE THE HUB HEIGHT
```

```
C     WIND SPEED IN THE DISC AVERAGE ARRAY
C     NOTE THE AVERAGING IS ARITHMETIC, NOT VECTOR AVERAGING
C     THIS IS MUCH FASTER BUT CAN RESULT IN ERRORS FOR LARGE
C     VARIATIONS IN WIND DIRECTION.
C
      IF(NSANEM.EQ.0) GO TO 200
      DO 100 I=1,ICOUNT
      SUM=0.0
      DO 110 J=1,NSANEM
      SUM=SUM+ D(ISANEM(J),I)
  110 CONTINUE
      D(NWS,I) = SUM/FLOAT(NSANEM)
  100 CONTINUE
      GO TO 300
C
C     DEFAULT TO HUB HEIGHT VALUE IF AVERAGING NOT USED
C
  200 DO 210 I=1,ICOUNT
      D(NWS,I)=D(NWSHUB,I)
  210 CONTINUE
  300 CONTINUE
C
C     COMPUTE AVERAGE WIND DIRECTION
C
      IF(NDANEM.EQ.0) GO TO 500
      DO 400 I=1,ICOUNT
      SUM=0.0
      DO 410 J=1,NDANEM
      SUM=SUM+D(IDANEM(J),I)
  410 CONTINUE
      D(NWD,I)=SUM/FLOAT(NDANEM)
  400 CONTINUE
      GO TO 600
C
C     DEFAULT TO HUB HEIGHT VALUE IF AVERAGING NOT USED
C
  500 DO 510 I=1,ICOUNT
      D(NWD,I)=D(NWDHUB,I)
  510 CONTINUE
C
  600 CONTINUE
C
      RETURN
      END
C
C     ***************************
C
      SUBROUTINE GFILT(BW,DELT,NITEMS,ITEM,NAME,NCHAN,INCHAN)
C
C       SUBROUTINE OPENS, READS, AND ALLOWS CHANGING OF THE
C       'FILTER' FILE.  'FILTER' CONTAINS THE FILTER BANDWIDTHS
C       THAT WILL BE APPLIED, USING DIGITAL LOW-PASS SINE BUTTERWORTH
C       FILTERS, TO EACH CHANNEL OF DATA.
C
C       EACH CHANNEL CAN BE FILTERED AT A DIFFERENT BANDWIDTH
C       (THE AZIMUTH AND TIME CHANNELS CANNOT BE FILTERED).
C       THE USER MUST USE CARE TO SELECT BANDWIDTHS THAT
C       ARE WITHIN THE CONSTRAINTS OF THE ORIGINAL SAMPLE RATES.
```

```
C
C       VARIABLES:  BW=ARRAY OF FILTER BANDWIDTHS (FIRST ELEMENT
C                       CORRESPONDS TO FIRST CHANNEL, ETC.)
C                   NAME=ARRAY CONTAINING ITEM NAMES
C                   NCHAN=NUMBER OF ACTIVE CHANNELS (AS IN MAIN PROGRAM)
C       C HANSEN   7/85
C
      DIMENSION BW(1),ITEM(1)
      CHARACTER*30 NAME(1)
C
C     OPEN DATA FILE CONTAINING FILTER INFORMATION
C
      OPEN(UNIT=23,FILE='FILTER.DAT',STATUS='OLD',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
            WRITE(*,*) 'ERROR OPENING UNIT 23, FILE FILTER.DAT'
            WRITE(*,*)  'IERR= ',IERR
      END IF
C
C       READ FILE AND DISPLAY ON CRT
C
      REWIND 23
      DO 50 I=1,NITEMS
      READ(23,*) BW(I)
   50 CONTINUE
      REWIND 23
C
      WRITE(*,*) ' '
      WRITE(*,*) 'SELECT BANDWIDTHS FOR LOW PASS FILTERING OF DATA'
      WRITE(*,*) ' '
  200 WRITE(*,*) 'CHAN            ITEM                    BANDWIDTH'
      DO 10 I=1,INCHAN
      WRITE(*,100) I,NAME(I),BW(I)
  100 FORMAT(2X,I2,3X,A30,5X,F10.4)
   10 CONTINUE
   20 WRITE(*,*) 'ENTER CHANNEL # YOU WANT TO CHANGE (0=NO CHANGE)'
      READ*,ICHAN
      IF(ICHAN.NE.0) THEN
         IF(ICHAN.EQ.1) THEN
             WRITE(*,*) 'YOU CANNOT FILTER AZIMUTH CHANNEL'
             GO TO 20
         END IF
   70    WRITE(*,*) 'ENTER NEW BANDWIDTH (HZ)  (0=NO FILTER)'
         READ*, BW(ICHAN)
         IF(BW(ICHAN).LT..005/DELT.AND.BW(ICHAN).NE.0.0) THEN
            WRITE(*,*) 'RECOMMEND A MINIMUM CUTOFF FREQUENCY'
            WRITE(*,*) 'THAT IS >0.5% OF SAMPLE RATE'
            WRITE(*,*) 'DO YOU WANT TO CHANGE YOUR INPUT FREQUENCY?'
            WRITE(*,*) 'YES=1, NO=0'
            READ(*,*) IANS
            IF(IANS.EQ.0) THEN
               GO TO 200
            ELSE
               GO TO 70
            ENDIF
         ENDIF
        GO TO 200
      END IF
C
```

```
C       REWRITE THE FILTER FILE, STORING CHANGES
C
      DO 60 I=1,NITEMS
      WRITE(23,*) BW(I)
   60 CONTINUE
      ENDFILE 23
C
C       CLOSE FILE AND RETURN
C
      CLOSE(23)
      RETURN
      END
C
C     ************************
C
      SUBROUTINE GIDS(ITEM,NITEMS,NAME,NCHAN)
C
C     READS THE ITEMS AND NAMES FILES FOR PREDAT PROGRAM
C
C        C. HANSEN   1/87
C
      DIMENSION ITEM(1)
      CHARACTER*30 NAME(1)
C
C     READ NAMES FROM NAMES.DAT FILE
C
      OPEN(UNIT=11,FILE='NAMES.DAT',STATUS='OLD',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
      WRITE(*,*) 'ERROR OPENING NAMES.DAT FILE (UNIT 11)'
      WRITE(*,*) 'IERR= ',IERR
      STOP
      ENDIF
C
      DO 5 I=1,NITEMS
      READ(11,*) NAME(I)
    5 CONTINUE
      CLOSE(11)
C
C     OPEN 'ITEMS' FILE FOR CHANNEL IDENTIFICATION
C
      OPEN(UNIT=21,FILE='ITEMS.DAT',STATUS='UNKNOWN',IOSTAT=IERR)
      IF (IERR.NE.0) THEN
         WRITE(*,*) 'ERROR ON OPENING ITEMS FILE'
         WRITE(*,*) 'IERR= ',IERR
         STOP
      ENDIF
      DO 10 I=1,NITEMS
      READ(21,100) ITEM(I)
  100 FORMAT(I2)
   10 CONTINUE
C
      CLOSE(21)
C
      RETURN
      END
C
C     ************************
C
```

```
      SUBROUTINE HORSHR(ICOUNT)
C
C     SUBROUTINE TO COMPUTE THE HORIZONTAL WIND SHEAR
C     FROM TWO ANEMOMETERS.  THE SHEAR IS WS1-WS2 WHERE
C     THE WIND SPEEDS ARE IDENTIFIED BY ITEMS IWSH1, AND IWSH2.
C     THE RESULT IS PLACED IN ARRAY D(27,I)
C     ICOUNT IS THE NUMBER OF VALUES COMPUTED
C
C     FOR USE WITH PREDAT PROGRAM
C
C     C HANSEN   12/85
C
      INTEGER HSHEAR, VSHEAR
      COMMON D(30,1000)
      COMMON/WNDAV/ NSANEM,NDANEM,ISANEM(5),IDANEM(5),HSHEAR,VSHEAR,
     +  IWSH1,IWSH2,IWSV1,IWSV2
C
      DO 100 I=1,ICOUNT
      D(27,I)=D(IWSH1,I)-D(IWSH2,I)
 100  CONTINUE
      RETURN
      END
C
C      **********************************
C
       SUBROUTINE LPFILT
C
C
C       SUBROUTINE TO FILTER TIME SERIES USING LOW PASS, ORDER 6
C       SINE BUTTERWORTH FILTER.  FILTER COEFFICIENTS
C       ARE CALCULATED BY SUBROUTINE LPSB.
C
C     CURRENTLY SET UP TO HANDLE UP TO 30 CHANNELS OF DATA
C     AT A TIME.  THAT IS, THE COEFFICIENTS AND RECENT VALUES
C     FOR EACH OF 30 DIFFERENT CHANNELS CAN BE STORED IN THE
C     ARRAYS.  EACH CALL OF THIS ROUTINE STILL FILTERS ONE
C     DATA POINT FROM ONE CHANNEL.
C
C     SEE OTNES AND ENOCHSON TEXT FOR DETAILS OF DEVELOPMENT
C
C     EACH CALL OF THIS ROUTINE FILTERS ONE POINT (XFILT) ONLY
C
C       MODIFIED FOR USE IN AZIMUTH AVERAGING PROGRAMS
C           C HANSEN 7/85, FURTHER MODIFIED 6/86, UNIVERSITY OF UTAH
C
C     XFILT,YFILT=SINGLE INPUT AND OUTPUT VALUES TO THE FILTER
C       DT=SAMPLING INTERVAL =1/SAMPLE RATE
C       BW=CUTOFF FREQUENCY OF THE FILTER (1/2 POWER POINT
C          OR - 3 DB POINT)  MUST BE LESS THAN 1/(2*DT)
C     A1,A2,BZERO=FILTER COEFFICIENTS FROM SUBROUTINE LPSB
C     Y0,Y1=PREVIOUS VALUES, USED IN RECURSIVE FILTER
C     ICHAN=CHANNEL NUMBER THAT WILL BE FILTERED ON THIS PASS
C           MUST BE SET BY MAIN PROGRAM AND BE
C           APPROPRIATE TO XFILT VALUE ASSIGNED IN MAIN
C
C
      COMMON/FILTER/ XFILT,YFILT,BZERO(30),A1(30,3),
     +               A2(30,3),BW(30),DT,ICHAN
```

```
      DIMENSION Y0(30,4), Y1(30,4)
C
      SAVE Y0,Y1
C
C     INITIALIZE FILTER SETTING ALL Y'S TO THE FIRST X
C
      DO 5 I=1,4
      Y0(ICHAN,I)=XFILT
    5 Y1(ICHAN,I)=XFILT
      RETURN
C
C     ------------------------------
C     ENTRY POINT USED FOR FILTERING.  ENTER AT THIS POINT FOR
C     ALL FILTERING, AT THE INITIAL POINT TO INITIALIZE ONLY.
C
      ENTRY LPFIL2
C
      Y0(ICHAN,1)=XFILT
      DO 20 K=1,3
      YI=BZERO(ICHAN)*Y0(ICHAN,K)-A1(ICHAN,K)*Y0(ICHAN,K+1)
     +                           -A2(ICHAN,K)*Y1(ICHAN,K+1)
      Y1(ICHAN,K+1)=Y0(ICHAN,K+1)
   20 Y0(ICHAN,K+1)=YI
      YFILT=YI
C
      RETURN
      END

C
C       ************************************
C
        SUBROUTINE LPSB
C
C       THIS SUBROUTINE COMPUTES SINE BUTTERWORTH LOWPASS FILTER
C     COEFFICIENTS FOR AN ORDER SIX FILTER.
C
C     SET UP TO COMPUTE AND STORE IN COMMON THE COEFFICIENTS
C     FOR 30 DIFFERENT DATA CHANNELS (EACH WITH DIFFERENT BANDWIDTH)
C
C     VARIABLES:
C
C     BW(ICHAN)=CUT-OFF FREQUENCY (HZ, -3DB/PASS) FOR CHANNEL ICHAN
C     A1(ICHAN,3), A2(ICHAN,3), BZERO(ICHAN)=
C           FILTER COEFFICIENTS FOR CHANNEL ICHAN
C     DT=SAMPLING INTERVAL (SEC)
C
C       FROM 'APPLIED TIME SERIES ANALYSIS'
C       BY R K OTNES AND L ENOCHSON
C       CONVENTION IS.....
C       Y(I)=-A1*Y(I-1)-A2*Y(I-2)+BZERO*X(I)
C
C     C. HANSEN, UNIVERSITY OF UTAH 6/86
C
        COMMON/FILTER/ XFILT,YFILT,BZERO(30),A1(30,3),
     +                 A2(30,3),BW(30),DT,ICHAN
        DOUBLE PRECISION A,B,C,D,E,F,G,H,FN,FACT,WEDGE,SECTOR,ANG
C
C       CHECK CUTOFF FREQUENCY AND DEFAULT TO MAX ALLOWED
```

```
C       IF REQUESTED VALUE TOO LARGE
C
            IF(BW(ICHAN).GT.0.5/DT) THEN
        WRITE(*,*) 'FILTERING ITEM #',ICHAN
              WRITE(*,*) 'REQUESTED BANDWIDTH TOO LARGE, DEFAULT TO
.5/DT'
              BW(ICHAN)=.5/DT
            END IF
C
        FACT=3.14159265*DT*BW(ICHAN)
        FACT=DSIN(FACT)
        F=1.D0
C
C       ORDER 6 FILTER ONLY
C
        M=6
        M1=M/2
        M3=M1
        M2=M
        A=M2
        SECTOR=3.14159265D0/A
        WEDGE=SECTOR/2.D0
C
        DO 5 I=1,M1
        FN=I-1
        B=FACT*DSIN(FN*SECTOR+WEDGE)
        C=1.D0-FACT*FACT
        D=0.5D0*(-C+DSQRT(C*C+4.D0*B**2))
        E=DSQRT(D+1.D0)+DSQRT(D)
        G=2.D0*((2.D0*B*B/D)-1.D0)/(E**2)
        H=-1.D0/(E**4)
        F=F*(1.D0-G-H)
        A1(ICHAN,I)=-G
        A2(ICHAN,I)=-H
   5    CONTINUE
C
        A=M3
        BZERO(ICHAN)=F**(1.D0/A)
C
        RETURN
        END
C
C     *************************
C
      SUBROUTINE VERSHR(ICOUNT)
C
C     SUBROUTINE TO COMPUTE THE VERTICAL WIND SHEAR
C     FROM TWO ANEMOMETERS.  THE SHEAR IS WS1-WS2 WHERE
C     THE WIND SPEEDS ARE IDENTIFIED BY ITEMS I1, AND I2.
C     THE RESULT IS PLACED IN ARRAY D(30,I)
C     ICOUNT IS THE NUMBER OF VALUES COMPUTED
C
C     FOR USE WITH PREDAT PROGRAM
C
C     C HANSEN   12/85
C
      INTEGER HSHEAR, VSHEAR
      COMMON D(30,1000)
```

```
      COMMON/WNDAV/ NSANEM,NDANEM,ISANEM(5),IDANEM(5),HSHEAR,VSHEAR,
     +  IWSH1,IWSH2,IWSV1,IWSV2
C
      DO 100 I=1,ICOUNT
      D(28,I)=D(IWSV1,I)-D(IWSV2,I)
 100  CONTINUE
      RETURN
      END
C
C     *************************
C
      SUBROUTINE YAWERR(ICOUNT)
C
C       SUBROUTINE TO COMPUTE YAW ERROR FROM TURBINE YAW ANGLE AND WIND
C       AZIMUTH DATA.  THE YAW ERROR IS DEFINED AS [YAW-WIND AZIMUTH]
C       THE YAW ERROR IS RETURNED TO MAIN IN ELEMENT D(NYERR,I)
C
C       REQUIRES YAW DATA IN ELEMENT D(NYAW,I)
C              AND WIND DIRECTION IN D(NWD,I)
C       ICOUNT IS THE NUMBER OF DATA PAIRS USED THE THE ARRAY D
(MAX=1000).
C       THIS ROUTINE ACCOUNTS FOR YAW ERROR PROBLEMS NEAR ANGLES=0 (360)
C
C           C HANSEN 6/85
C
      COMMON D(30,1000)
C
C       CHANNEL ID'S
C
      NYAW=4
      NYERR=24
      NWD=26
C
C       MAIN LOOP
C
      DO 10 I=1,ICOUNT
      ERR=D(NYAW,I)-D(NWD,I)
      SGN=SIGN(1.,ERR)
      IF(ABS(ERR).GT.180.)ERR=ERR-SGN*360.
 20   IF(ERR.GE.360.) ERR=ERR-360.
      IF(ERR.LE.-360.)ERR=ERR+360.
      IF(ABS(ERR).GE.360.) GO TO 20
      D(NYERR,I)=ERR
 10   CONTINUE
C
      RETURN
      END
C
C     *****************************
C
      SUBROUTINE YAWRAT(DELD,ICOUNT)
C
C       CALCULATES YAW RATE FROM EQUAL TIME INTERVAL YAW ANGLE DATA
C       THE YAW RATE IS CALCULATED USING A TWO POINT CENTRAL DIFFERENCE
C       FOR ALL BUT THE FIRST AND LAST POINTS IN THE ARRAY.
C       THE RATE FOR THE FIRST POINT COMES FROM A FORWARD DIFFERENCE
C       WHILE THE LAST RATE IS FROM BACKWARD DIFFERENCE
C
```

```
C       THE YAW ANGLE MUST BE IN ELEMENT D(NYAW,I) AND THE YAW RATE
C       IS RETURNED IN ELEMENT D(NRATE,I)
C       ICOUNT IS THE NUMBER OF SAMPLES IN THE ARRAY D (MAX OF 1000)
C       DELT IS THE TIME INTERVAL
C               C HANSEN   6/85
C
      COMMON D(30,1000)
      NYAW=4
      NRATE=23
C
C     BE SURE THERE IS MORE THAN ONE POINT IN D ARRAY
C
      IF(ICOUNT.LT.2) THEN
         WRITE(*,*) 'WARNING!!!'
         WRITE(*,*) 'YAWRAT SUBROUTINE CALLED WITH ICOUNT=',ICOUNT
         WRITE(*,*) 'THE YAW RATE WILL DEFAULT TO ZERO FOR THIS CASE'
         WRITE(*,*) 'RECOMMEND YOU RUN A DIFFERENT NUMBER OF POINTS'
         D(NRATE,1)=0.0
         RETURN
      ENDIF
C
C     USE 2*DELT TO SAVE STEPS LATER
C
      DELT=DELD*2.
C
C     TAKE CARE OF FIRST AND LAST POINTS
C
      D(NRATE,1)=(-3.*D(NYAW,1) + 4.*D(NYAW,2) - D(NYAW,3))/DELT
      D(NRATE,ICOUNT)=(D(NYAW,ICOUNT-2)-4.*D(NYAW,ICOUNT-1)
     +     +3.*D(NYAW,ICOUNT))/DELT
C
C       NOW GET THE REST OF THE POINTS
C
      DO 10 I=2,ICOUNT-1
      D(NRATE,I)=(D(NYAW,I+1)-D(NYAW,I-1))/DELT
 10   CONTINUE
C
      RETURN
      END
C
C     ******************
C
      SUBROUTINE TURBLNC(TURBTIM,DELT,NPOINT,ICOUNT)
C
C     THIS SUBROUTINE COMPUTES A RUNNING AVERAGE OF THE TURBULENCE
C     INTENSITY AT THE HUB HEIGHT OF THE WIND TURBINE.  THE WIND
C     SPEED IS CONTAINED IN VECTOR D(IDWIND,I).  THE RMS OF THE WIND
C     SPEED FLUCTUATIONS AND THE MEAN WIND SPEED ARE COMPUTED OVER A
C     A TIME PERIOD 'TURBTIM' (SECONDS).  THE AVERAGE IS A RUNNING
C     AVERAGE.
C
C     THE NUMBER OF SAMPLES IN A PERIOD 'TURBTIM' IS 'NTURB'.
C
C     THE TURBULENCE INTENSITY IS DEFINED AS THE RMS/MEAN*100.,
C     EXPRESSED AS A PERCENTAGE.
C
C     THE VALUE OF TURBULENCE INTENSITY IS WRITTEN TO VECTOR D(IDTURB,I)
C
```

```
C     THE TURBULENCE INTENSITY CANNOT BE COMPUTED UNTIL THE AVERAGING
C     TIME HAS BEEN REACHED.  UNTIL THIS TIME, THE VALUE OF -1. WILL
C     BE WRITTEN TO PROVIDE A FLAG AND FILL THE ARRAY.
C
C     C HANSEN, 8/87, UNIVERSITY OF UTAH
C
C     INITIALIZE ARRAYS AND ID VALUES
C
      COMMON D(30,1000)
      DOUBLE PRECISION NPOINT, ITOT
C
      SAVE VBAR, V2BAR
C
      DATA VBAR/0.D0/,V2BAR/0.D0/
C
      IDWIND=2
      IDTURB=29
C
C     THESE INTEGERS IDENTIFY THE LOCATIONS (IN D ARRAY) WHERE THE
C     HUB-HEIGHT WIND SPEED CHANNEL IS LOCATED AND WHERE THE
C     COMPUTED TURBULENCE INTENSITY WILL BE WRITTEN.  THESE VALUES
C     MUST BE THE SAME AS USED IN THE NAMES FILE.  THE PROGRAM DOES
C     NOT CHECK THIS COMPATIBILITY, IT IS UP TO THE USER.
C
      NTURB=TURBTIM/DELT
C
      DO 100 I=1,ICOUNT
      ITOT = NPOINT - ICOUNT + I
      V = D(IDWIND,I)
C
      IF(ITOT .LT. NTURB) THEN
C
         VBAR =  ( (ITOT-1) * VBAR + V )/ITOT
         V2BAR = ( (ITOT-1) * V2BAR + V*V )/ITOT
         D(IDTURB,I) = -1.0
C
      ELSE
C
         VBAR = ( REAL(NTURB-1) * VBAR + V )/REAL (NTURB)
         V2BAR = ( REAL(NTURB-1) * V2BAR + V*V )/REAL (NTURB)
         RMS2 = REAL(NTURB)/REAL(NTURB-1) * (V2BAR-VBAR*VBAR)
         D(IDTURB,I) = SQRT(RMS2)/VBAR*100.
C
      ENDIF
C
  100 CONTINUE
C
      RETURN
      END
```

```
      PROGRAM AZBINS
C
C     PROGRAM TO PERFORM AZIMUTH AVERAGING AND NORMAL
C     METHOD OF BINS
C
C     DATA IS READ FROM FILE 'AZIN.DAT', WITH MISCELLANEOUS
C     OTHER INFORMATION ABOUT THE DATA RECORD IN 'AZINFO.DAT'
C
C     VARIABLES:
C        D(I)=RAW DATA ARRAY FROM PROGRAM PREDAT (FILE AZIN.DAT)
C
C        BIN(1,I)=AVERAGE IN EACH OF 30 BINS
C        BIN(2,I)=MINIMUM VALUE IN EACH BIN
C        BIN(3,I)=MAXIMUM VALUE IN EACH BIN
C        BIN(4,I)=STANDARD DEVIATION IN BIN
C        BIN(5,I)=NUMBER OF DATA POINTS IN BIN
C        BIN IS A DOUBLE PRECISION ARRAY
C
C        BINPL IS A SINGLE PRECISION DUPLICATE OF BIN ARRAY FOR PLOTS
C
C        XCHAN=INTEGER IDENTIFICATION OF INDEPENDENT VARIABLE
C        YCHAN=INTEGER IDENTIFICATION OF DEPENDENT VARIABLE
C
C     A SCAN IS ONE 'SNAPSHOT' OF DATA OR ALL CHANNELS AT ONE INSTANT
C     TIMES ARE SPECIFIED IN IRIG DECIMAL TIME--THE NUMBER OF SECONDS
C     FROM THE START OF THE YEAR.
C
C     C HANSEN   7/85
C     MODIFIED 7/86 UNIVERSITY OF UTAH
C
      PARAMETER (NITEMS=30)
      DOUBLE PRECISION BIN(5,30)
      DIMENSION BINPL(5,30)
      DIMENSION D(NITEMS)
      DIMENSION WINDOW(2,2), IWINDO(2)
      INTEGER XCHAN, YCHAN
      INTEGER*4 NSTOP, IREAD, NSTART, NSCAN
      CHARACTER*80 TITLE
      CHARACTER*50 NAME(NITEMS)
      DATA D/NITEMS*0./
      DO 5 I=1,30
      BIN(1,I)=0.D00
      BIN(2,I)=1.E20
      BIN(3,I)=-1.E20
      BIN(4,I)=0.D00
      BIN(5,I)=0.D00
 5    CONTINUE
C
C     READ NAMES FILE
C
      OPEN(UNIT=40,FILE='NAMES.DAT',STATUS='OLD',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING NAMES.DAT FILE (UNIT 40)'
         WRITE(*,*) 'IERR= ',IERR
         STOP
      ENDIF
      DO 2000 I=1,NITEMS
      READ(40,*) NAME(I)
```

```
 2000 CONTINUE
      CLOSE(40)
C
C     OPEN FILE 'AZINFO' TO GET FILE INFORMATION
C
      OPEN(UNIT=30,FILE='AZINFO.DAT',STATUS='OLD',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING AZINFO.DAT (UNIT 30)'
         WRITE(*,*) 'IERR= ',IERR
         STOP
      ENDIF
      READ(30,3000) TITLE
      READ(30,*) NCHAN
      READ(30,*) NSCAN
      READ(30,*) DELT
 3000 FORMAT(A)
C
C     CLOSE UNIT 30---AZINFO
C
      CLOSE(30)
C
C     OPEN FILE 'AZIN' (UNIT 20) TO READ TIME SERIES DATA
C
      OPEN(UNIT=20,FILE='AZIN.DAT',STATUS='OLD',
     +     FORM='UNFORMATTED',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING AZIN.DAT (UNIT 20)'
         WRITE(*,*) 'IERR= ',IERR
         STOP
      ENDIF
      REWIND 20
C
C     INPUT THE SETUP DATA FOR THIS RUN
C
      WRITE(*,*) 'THIS IS THE AZIMUTH AVERAGING, METHOD OF BINS PROGRAM'
      WRITE(*,*) 'DO YOU WANT TO DO AZIMUTH AVERAGING (YES=1)'
      READ*,IANS
      IF(IANS.EQ.1) GO TO 10
      WRITE(*,*) 'ENTER ITEM NUMBER FOR INDEPENDENT VARIABLE '
      READ*,XCHAN
      WRITE(*,*) 'ENTER MINIMUM VALUE OF X FOR BINS ANALYSIS'
      READ*,XMIN
      WRITE(*,*) 'ENTER MAXIMUM VALUE OF X FOR BINS ANALYSIS'
      READ*,XMAX
      WRITE(*,*) 'ENTER NUMBER OF BINS (MAXIMUM OF 30)'
      READ*,NBIN
      GO TO 40
 10   XCHAN=1
      NBIN=24
      XMIN=0.
      XMAX=360.
 40   CONTINUE
      WRITE(*,*) 'ENTER ITEM NUMBER FOR DEPENDENT VARIABLE'
      READ*,YCHAN
C
C
      WRITE(*,*) 'THERE ARE ',NSCAN,' SCANS IN THIS DATA FILE'
      WRITE(*,*) 'ENTER SCAN NUMBER FOR START OF ANALYSIS'
```

```
      READ*,NSTART
      WRITE(*,*) 'ENTER NUMBER OF SCANS YOU WISH TO ANALYZE'
      READ*,NSTOP
C
C     SKIP INTO THE DESIRED START POINT
C
      DO 100 I=1,NSTART
      READ(20,END=240) (D(I7),I7=1,NITEMS)
 100  CONTINUE
C
C
C     GET INFORMATION FOR SAMPLING OF RAW DATA TO SCREEN
C
      WRITE(*,*) 'IF YOU WISH YOU CAN PRINT EVERY Nth DATA PAIR TO CRT'
      WRITE(*,*) 'ENTER N     (0=NO SAMPLING TO SCREEN)'
      READ*, NVIEW
C
C     GET SELECTIVE SAMPLING INFORMATION
C
      WRITE(*,*) 'DO YOU WISH TO USE SELECTIVE SAMPLING? (YES=1)'
      READ*, ISEL
      IF(ISEL.NE.1) GO TO 110
      WRITE(*,*) 'SELECT WINDOW SIZES FOR TWO SELECTIVE SAMPLING'
      WRITE(*,*) 'CHANNELS. IF YOU ONLY WANT TO USE ONE CHANNEL SELECT'
      WRITE(*,*) 'ANY CHANNEL CONTAINING DATA FOR THE SECOND CHANNEL'
      WRITE(*,*) 'AND USE A VERY LARGE WINDOW SIZE (I.E. +/- 1.E20)'
      WRITE(*,*) ' '
      DO 120 IWIN=1,2
      WRITE(*,*) ' '
      WRITE(*,*) 'ENTER ITEM NUMBER OF SELECTION WINDOW',IWIN
      READ*,IWINDO(IWIN)
      WRITE(*,*) 'ENTER MINIMUM ACCEPTABLE VALUE FOR ITEM',IWINDO(IWIN)
      READ*,WINDOW(IWIN,1)
      WRITE(*,*) 'ENTER MAXIMUM ACCEPTABLE VALUE FOR ITEM',IWINDO(IWIN)
      READ*,WINDOW(IWIN,2)
 120  CONTINUE
C
C     INITIALIZE VALUES
C
 110  IREAD=0
      SCALE=(XMAX-XMIN)/FLOAT(NBIN)
C
C     START MAIN LOOP FOR ANALYSIS
C
 200  CONTINUE
C
C     READ ONE SCAN OF DATA
C
      READ(20,END=240) (D(I7),I7=1,NITEMS)
      IREAD=IREAD+1
C
C     PRINT DATA TO CRT IF DESIRED
C
      IF(NVIEW.EQ.0) GO TO 210
      IF(FLOAT(IREAD/NVIEW).EQ.FLOAT(IREAD)/FLOAT(NVIEW))
     +   WRITE(*,*)  IREAD, D(XCHAN), D(YCHAN)
C
C     SORT DATA INTO APPROPRIATE BIN
```

```
C
C     SKIP DATA IF NOT INSIDE 'WINDOWS'
C
  210 IF(ISEL.NE.1) GO TO 230
      CALL SELSAM(D(IWINDO(1)),D(IWINDO(2)),WINDOW,IFLAG)
      IF(IFLAG.EQ.0) GO TO 220
 230  X=D(XCHAN)
      Y=D(YCHAN)
      IBIN=INT((X-XMIN)/SCALE) + 1
C
C     IF DATA OUTSIDE RANGE SELECTED, DEFAULT TO END BIN
C
      IF(IBIN.GT.NBIN) IBIN=NBIN
      IF(IBIN.LT.1) IBIN=1
      BIN(1,IBIN)=BIN(1,IBIN) + Y
      IF(Y.LT.BIN(2,IBIN)) BIN(2,IBIN)=Y
      IF(Y.GT.BIN(3,IBIN)) BIN(3,IBIN)=Y
      BIN(4,IBIN)=BIN(4,IBIN) +Y*Y
      BIN(5,IBIN)=BIN(5,IBIN) + 1.
C
C     RETURN FOR MORE DATA UNTIL STOP TIME
C
  220 IF(IREAD.LT.NSTOP) GO TO 200
C
 240  CONTINUE
C
C
C     CLOSE THE AZIN DATA FILE
C
      CLOSE(20)
C
C     COMPUTE FINAL AVERAGES, ETC
C
      DO 300 I=1,NBIN
      IF(BIN(5,I).LT.1.) THEN
         BIN(5,I)=0.
         GO TO 300
      END IF
      IF(BIN(5,I).EQ.1.) THEN
         BIN(4,I)=0.
         GO TO 300
      END IF
      BIN(1,I)=BIN(1,I)/BIN(5,I)
      BIN(4,I)=BIN(4,I)-BIN(5,I)*BIN(1,I)*BIN(1,I)
      IF(BIN(4,I).LT. 0.) THEN
         BIN(4,I)=-100.
         GO TO 300
      END IF
      BIN(4,I)=SQRT(BIN(4,I)/(BIN(5,I)-1.))
 300  CONTINUE
C
C     PRINT RESULTS
C
      OPEN(UNIT=10,FILE='AZBINOUT.DAT',STATUS='NEW')
      WRITE(*,1100) TITLE
      WRITE(10,1100) TITLE
 1100 FORMAT(1H1//' METHOD-OF-BINS ANALYSIS'//1X,A80)
      WRITE(*,1030) NAME(XCHAN)
```

```
      WRITE(10,1030) NAME(XCHAN)
 1030 FORMAT(/1X,'INDEPENDENT VARIABLE (X) = ',A30)
      WRITE(*,1040) NAME(YCHAN)
      WRITE(10,1040) NAME(YCHAN)
 1040 FORMAT(1X,'DEPENDENT VARIABLE = ',A30)
      WRITE(*,1050) NBIN,XMIN,XMAX
      WRITE(10,1050) NBIN,XMIN,XMAX
 1050 FORMAT(/1X, I2,' BINS USED IN THE ANALYSIS'
     + /1X,'RANGE OF BINS= ',F8.3, ' TO ',F8.3)
      WRITE(*,1060) NSTART,NSTOP
      WRITE(10,1060) NSTART,NSTOP
 1060 FORMAT(/1X,'FIRST SCAN USED= ',I8,'  NUMBER OF SCANS= ',I8/)
      WRITE(*,1110)  DELT
      WRITE(10,1110) DELT
 1110 FORMAT(1X,'SAMPLE INTERVAL (SEC) = ',F8.3/)
      IF(ISEL.NE.1) THEN
         WRITE(*,1200)
         WRITE(10,1200)
 1200    FORMAT(1X,'SELECTIVE SAMPLING NOT USED')
         GO TO 400
      ENDIF
      WRITE(*,1210) NAME(IWINDO(1))
      WRITE(10,1210) NAME(IWINDO(1))
 1210 FORMAT(1X,'SELECTIVE SAMPLING USED IN THIS ANALYSIS',/
     +5X,'THE FIRST WINDOW CHANNEL WAS ',A30)
      WRITE(*,1220) WINDOW(1,1), WINDOW(1,2)
      WRITE(10,1220) WINDOW(1,1), WINDOW(1,2)
 1220 FORMAT(5X,'MINIMUM ACCEPTED VALUE FOR THIS CHANNEL= ',E10.3,/,
     +       5X,'MAXIMUM ACCEPTED VALUE FOR THIS CHANNEL= ',E10.3)
      WRITE(*,1230) NAME(IWINDO(2))
      WRITE(10,1230) NAME(IWINDO(2))
 1230 FORMAT(/5X,'THE SECOND WINDOW CHANNEL WAS ',A30)
      WRITE(*,1220) WINDOW(2,1), WINDOW(2,2)
      WRITE(10,1220) WINDOW(2,1), WINDOW(2,2)
 400  WRITE(*,1020)
      WRITE(10,1020)
 1020 FORMAT(//4X,'I',6X,'X',11X,'MEAN',9X,'MIN',7X,'MAX',7X,'STDEV',
     + 7X, '#SAMPLES'/)
C
C     PRINT TABLE AND FIND TOTAL NUMBER OF SAMPLES PASSED
C
      SUM=0.
      DO 310 I=1,NBIN
      IF(BIN(5,I).EQ.0) GO TO 310
      X=(FLOAT(I)-.5)*SCALE + XMIN
      WRITE(*,1010)I,X,(BIN(J,I),J=1,5)
      WRITE(10,1010) I,X,(BIN(J,I),J=1,5)
 1010 FORMAT(3X,I2,5(2X,G10.4),4X,F6.0)
      SUM=SUM+BIN(5,I)
 310  CONTINUE
C
      WRITE(*,1070) SUM
      WRITE(10,1070) SUM
 1070 FORMAT(/1X,'TOTAL NUMBER OF SAMPLES IN BINS= ',F10.1/)
      WRITE(*,*) ' '
      WRITE(*,*) 'THESE RESULTS HAVE BEEN SAVED IN FILE AZBINOUT.DAT'
C
C
```

```
      WRITE(*,*) 'WANT TO PLOT THE RESULTS? (YES=1)'
      READ*,IANS
      IF(IANS.EQ.1) THEN
        DO 500 I=1,5
        DO 510 J=1,30
        BINPL(I,J)=SNGL(BIN(I,J))
  510   CONTINUE
  500   CONTINUE
        CALL BINPLOT(BINPL,NBIN,XMIN,XMAX,
     +               NAME(XCHAN),NAME(YCHAN),TITLE)
      ENDIF
C
      CLOSE(10)
      STOP
      END
```

```
      SUBROUTINE BINPLOT(BIN,NBIN,XMIN,XMAX,XNAME,YNAME,TITLE)
C
C     ROUTINE PLOTS RESULTS OF THE METHOD-OF-BINS ANALYSIS
C     USING THE PLOT79 PACKAGE ON THE VAX
C
C     THIS ROUTINE IS PART OF THE SERI WIND ENERGY DATA ANALYSIS
C     PACKAGE
C
C     THE PLOT SHOWS THE MEAN AND  MEAN + OR - ONE STANDARD ERROR
C     OF THE MEAN VERSUS THE INDEPENDENT VARIABLE.
C     THE STANDARD ERROR OF THE MEAN ONLY HAS MEANING IF THE SAMPLES
C     ARE INDEPENDENT OF ONE ANOTHER.  THIS MEANS THAT THE DATA SHOULD
C     BE TIME-AVERAGED SUCH THAT EACH DATA POINT IS REASONABLY
INDEPENDENT
C
C     VARIABLES:
C       BIN=OUTPUT OF BINS PROGRAM AS DESCRIBED IN AZBINS.FOR
C       XNAME=TITLE OF X-AXIS (INDEPENDENT VARIABLE)
C       YNAME=TITLE OF Y-AXIS (DEPENDENT VARIABLE)
C       TITLE=TITLE OF PLOT (FROM PREVIOUS PROGRAMS)
C       Y(1,I)=PLOT VARIABLE = MEAN (=BIN(1,I))
C       Y(2,I)=MEAN + STD. ERROR OF MEAN (STD. DEV./SQRT(N))
C       Y(3,I)=MEAN - STD. ERROR OF MEAN
C       Y(4,I)=NUMBER OF POINTS IN BIN (=BIN(5,I))
C       Y(5,I)=X(I) INDEPENDENT VARIABLE
C
C     WRITTEN BY C. HANSEN   4/86  UNIVERSITY OF UTAH
C
      DIMENSION BIN(5,30), Y(5,30)
      CHARACTER*80 TITLE
      CHARACTER*50 XNAME, YNAME
      LOGICAL XLOG,XSCALE,YLOG,YSCALE
      DIMENSION ICMD(3,9), BNDRY(2,3)
C
C     DO 5 I=1,2
C     DO 6 J=1,3
C     BNDRY(I,J)=0.
C  6  CONTINUE
C  5  CONTINUE
C
C     FILL ARRAY TO BE PLOTTED, INCLUDING ONLY THE POINTS
C     WITH DATA POINTS IN THE BIN
C
      SCALE=(XMAX-XMIN)/FLOAT(NBIN)
      NPLOT=0
C
      DO 10 I=1,NBIN
      IF(BIN(5,I).LT.1.0) GO TO 10
      NPLOT=NPLOT+1
      Y(1,NPLOT)=BIN(1,I)
      Y(2,NPLOT)=BIN(1,I) + BIN(4,I)/SQRT(BIN(5,I))
      Y(3,NPLOT)=BIN(1,I) - BIN(4,I)/SQRT(BIN(5,I))
      Y(4,NPLOT)=BIN(5,I)
      Y(5,NPLOT)=(FLOAT(I)-.5)*SCALE + XMIN
  10  CONTINUE
C
C     ASSIGN PLOT VARIABLES
C
```

```
      NCOLS=30
      NROWS=5
      NLINES=3
      NFONT=42
      IBOX=2
      XLOG=.FALSE.
      YLOG=.FALSE.
      XSCALE=.TRUE.
      YSCALE=.TRUE.
C
      DO 20 I=1,NLINES
      ICMD(I,1)=NPLOT
      ICMD(I,2)=5
      ICMD(I,3)=1
      ICMD(I,4)=1
      ICMD(I,5)=I
      ICMD(I,6)=1
      ICMD(I,7)=1
      ICMD(I,8)=1
   20 CONTINUE
C
      ICMD(1,9)=1
      ICMD(2,9)=5
      ICMD(3,9)=5
C
      CALL GRAPH2D(Y,NROWS,NCOLS,ICMD,NLINES,NFONT,TITLE,
     + XNAME,XLOG,XSCALE,YNAME,YLOG,YSCALE,IBOX,BNDRY)
C
      RETURN
      END
```

```
      SUBROUTINE SELSAM(D1,D2,WINDOW,IFLAG)
C
C     THIS SUBROUTINE APPLIES THE SELECTIVE SAMPLING TEST
C     FOR THE AZIMUTH BINS PROGRAM (AZBINS)
C     THE VALUE 'IFLAG' IS RETURNED AS 1 IF THE DATA
C     POINTS D1 AND D2 ARE WITHIN THEIR RESPECTIVE WINDOWS
C     AND IFLAG=0 IF THE DATA IS OUTSIDE THE WINDOW
C
C     C HANSEN   7/85
C
      DIMENSION WINDOW(2,2)
      IFLAG=1
      IF(D1.LT.WINDOW(1,1)) IFLAG=0
      IF(D1.GT.WINDOW(1,2)) IFLAG=0
      IF(D2.LT.WINDOW(2,1)) IFLAG=0
      IF(D2.GT.WINDOW(2,2)) IFLAG=0
      RETURN
      END
```

```
      PROGRAM HARMON

C     THIS PROGRAM USE THE METHOD OF BINS WITH/WITHOUT AZIMUTH AVERAGING
C     AND COMPUTES THE HARMONIC CONTENT OF A SIGNAL WHICH IS A CYCLIC
C     FUNCTION OF ROTOR AZIMUTH ANGLE.  THE INPUT DATA
C     IS THE FILE AZIN.DAT (FROM PROGRAM PREDAT).
C
C     THE PROGRAM DOES THE FOLLOWING CALCULATIONS:
C     ONE CHANNEL IS SELECTED FOR HARMONIC ANALYSIS.  THIS WOULD
C       TYPICALLY BE A BLADE LOAD OR TEETER ANGLE BUT MAY BE ANY
C     CHANNEL THAT IS PRODUCED BY PROGRAM PREDAT.
C     THE INPUT DATA FILE IS READ INTO ARRAY D(I) FROM FILE AZIN.DAT.
C
C     DATA ARE READ FOR THE DESIRED NUMBER (NSTOP) OF REVOLUTIONS OF THE
C     ROTOR. THE DATA ARE THEN AZIMUTH AVERAGED WITH THE METHOD OF BINS
C     AND FOURIER ANALYZED, DETERMINING THE HARMONIC CONTENT (UP TO
NOUT*P),
C     THE MEAN, MAX, PK-PK AND STANDARD DEVIATION OF A SINGLE SELECTED
C     CHANNEL FOR N REVOLUTIONS OF THE ROTOR.  SUBROUTINE FOURCO IS
C     USED FOR THIS CALCULATION.
C
C       THIS PROCESS IS REPEATED, WITH NSTOP REVS IN EACH PASS, UNTIL
C       THE DESIRED PORTION OF THE DATA FILE IS ANALYZED.
C
C       ALL COMPUTED QUANTITIES ARE WRITTEN TO A NEW DATA FILE
(HARMCONT.DAT)
C
C     VARIABLES:
C           D=ARRAY THAT IS ONE SCAN OF ALL CHANNELS FROM PREDAT
C      X=ARRAY CONTAINING THE SINGLE CHANNEL FOR ONE REV. OF THE ROTOR
C      ANGLE=ARRAY CONTAINING AVERAGE AZIMUTH ANGLES OF THE ROTOR
C      SUM=ACCUMULATED MEAN OF EACH CHANNEL.
C      NITEMS=SAME AS IN PREDAT
C      HARM=ARRAY CONTAINING HARMONIC DATA (DESCRIBED IN FOURCO DOC.)
C      NOUT=NUMBER OF HARMONICS CALCULATED (UP TO NOUT*P)
C        NREV=NUMBER OF REVOLUTIONS OF ROTOR ACTUALLY ANALYZED
C       ICHAN=ITEM NUMBER OF CHANNEL FOR HARMONIC ANALYSIS
C        NSCAN=NUMBER OF SCANS (FROM PREDAT)
C        XCHAN=INTEGER IDENTIFICATION OF INDEPENDENT VARIABLE
C        YCHAN=INTEGER IDENTIFICATION OF DEPENDENT VARIABLE
C        BIN(1,I)=AVERAGE IN EACH OF 30 BINS
C        BIN(2,I)=MINIMUM VALUE IN EACH BIN
C        BIN(3,I)=MAXIMUM VALUE IN EACH BIN
C        BIN(4,I)=STANDARD DEVIATION IN BIN
C        BIN(5,I)=NUMBER OF DATA POINTS IN BIN
C        BIN IS A DOUBLE PRECISION ARRAY
C        BINPL IS A SINGLE PRECISION DUPLICATE OF BIN ARRAY FOR PLOTS
C
C     EXTERNAL DATA FILES:
C     NAMES.DAT=INPUT FILE CONTAINS NAMES OF EACH CHANNEL
C     AZIN.DAT=INPUT TIME SERIES DATA FROM PREDAT PROGRAM
C     AZINFO.DAT=CONTAINS INFORMATION ABOUT AZIN.DAT
C     BINTBL.DAT=OUTPUT DATA OF METHOD BINS (UNIT 10)
C     HARMCONT.DAT=OUTPUT DATA OF HARMONIC CONTENT (UNIT 25)
C     HARMINFO.DAT=OUTPUT DATA (UNIT 35)
C
C     SUBROUTINES:
C       SELSAM
```

```
C     FOURCO
C     BINSUB (FILBINS, BINSAVE, FOURFUN, ERRORFUN,SCRNVIEW)
C     DATPLOT
C
C       A SCAN IS ONE 'SNAPSHOT' OF DATA OR ALL CHANNELS AT ONE INSTANT
C       TIMES ARE SPECIFIED IN IRIG DECIMAL TIME--THE NUMBER OF SECONDS
C       FROM THE START OF THE YEAR.
C
C     WRITTEN BY C HANSEN, UNIV OF UTAH, 4/86
C     MODIFIED 9/86 UNIVERSITY OF UTAH
C
        PARAMETER (NITEMS=30)
        DOUBLE PRECISION BIN(5,30)
        DIMENSION D(NITEMS),B(100),ANGLE(100),HARM(12,2)
        DIMENSION WINDOW(2,2), IWINDO(2),SUMCHAN(30)
      DIMENSION SCRNPL(8,3,30),BINPL(9,1000)
      COMMON ERRFUN(30)
        INTEGER XCHAN, YCHAN
        INTEGER*4 NSTOP, IREAD, NSTART, NSCAN
        CHARACTER*80 TITLE
        CHARACTER*50 NAME(NITEMS),XLABLE,YLABLE
        DATA D/NITEMS*0./
C
C     OPEN FILE 'NAMES.DAT'
C
      OPEN(UNIT=40,FILE='NAMES.DAT',STATUS='OLD',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
      WRITE(*,*) 'ERROR OPENING NAMES.DAT FILE (UNIT 40)'
      WRITE(*,*) 'IERR=',IERR
      STOP
      ENDIF
      DO 6 I=1,NITEMS
      READ(40,*) NAME(I)
 6    CONTINUE
      CLOSE(40)
C
C       OPEN FILE 'AZINFO' TO GET FILE INFORMATION
C
        OPEN(UNIT=30,FILE='AZINFO.DAT',STATUS='OLD',IOSTAT=IERR)
        IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING AZINFO.DAT (UNIT 30)'
         WRITE(*,*) 'IERR= ',IERR
         STOP
        ENDIF
        READ(30,3000) TITLE
        READ(30,*) NCHAN
        READ(30,*) NSCAN
        READ(30,*) DELT
 3000   FORMAT(A)
        CLOSE(30)
C
C       OPEN FILE 'AZIN' (UNIT 20) TO READ TIME SERIES DATA
C
        OPEN(UNIT=20,FILE='AZIN.DAT',STATUS='OLD',
     +       FORM='UNFORMATTED',IOSTAT=IERR)
        IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING AZIN.DAT (UNIT 20)'
         WRITE(*,*) 'IERR= ',IERR
```

```
         STOP
        ENDIF
        REWIND 20
C
C       INPUT THE SETUP DATA FOR THIS RUN
C
        WRITE(*,*) 'THIS IS THE HARMONICS ANALYSIS PROGRAM'
        WRITE(*,*) 'DO YOU WANT TO DO AZIMUTH AVERAGING (YES=1)'
        READ*,IANS
C
        IF(IANS.EQ.1) GO TO 10
C
        WRITE(*,*) 'ENTER ITEM NUMBER FOR INDEPENDENT VARIABLE '
        READ*,XCHAN
        WRITE(*,*) 'ENTER MINIMUM VALUE OF X FOR BINS ANALYSIS'
        READ*,XMIN
        WRITE(*,*) 'ENTER MAXIMUM VALUE OF  FOR BINS ANALYSIS'
        READ*,XMAX
        WRITE(*,*) 'ENTER NUMBER OF BINS (MAXIMUM OF 30)'
        READ*,NBIN
        GO TO 40
C
 10   CONTINUE
        XCHAN=1
        NBIN=24
        XMIN=0.
        XMAX=360
C
 40   WRITE(*,*) 'DO YOU WANT TO PERFORM FOURIER (HARMONIC) ANALYSIS?'
      WRITE(*,*) '(YES = 1)'
      READ*, NFOUR
        WRITE(*,*) 'ENTER THE NUMBER OF HARMONICS TERMS YOU WANT'
        WRITE(*,*) 'VALUES BETWEEN 2 AND 10 ARE ALLOWED'
        READ(*,*) NOUT

C
        WRITE(*,*) 'ENTER ITEM NUMBER FOR DEPENDENT VARIABLE'
        READ*,YCHAN
C
C
        WRITE(*,*) 'THERE ARE ',NSCAN,' SCANS IN THIS DATA FILE'
        WRITE(*,*) 'ENTER SCAN NUMBER FOR START OF ANALYSIS'
        READ*,NSTART
C
C       SKIP INTO THE DESIRED START POINT
C
        DO 100 I=1,NSTART
        READ(20,END=240) (D(I7),I7=1,NITEMS)
 100    CONTINUE
C
C       GET INFORMATION FOR SAMPLING OF RAW DATA TO SCREEN
C
        WRITE(*,*) 'THE DATA FILE WILL BE ANALYZED IN BLOCKS'
      WRITE(*,*) 'OR SETS. A METHOD OF BINS OUTPUT AND'
      WRITE(*,*) 'HARMONIC ANALYSIS WILL BE PRODUCED FOR EACH SET'
        WRITE(*,*) ' '
      WRITE(*,*) 'ENTER THE NUMBER OF SCANS YOU WISH TO USE'
      WRITE(*,*) 'IN EACH AZIMUTH AVERAGING PASS'
```

```
        READ*,NSTOP
C
C     CHOOSE SETS OF SCANS
C
      NTIMES=(NSCAN-NSTART)/NSTOP
      WRITE(*,*)'THERE ARE',NTIMES,'  SETS OF SCANS IN THE DATA'
      WRITE(*,*)'FILE, ENTER NUMBER OF SETS YOU WANT TO ANALYZE'
      READ*, NSETS
C
        WRITE(*,*) 'IF YOU WISH YOU CAN PRINT EVERY Nth DATA PAIR'
        WRITE(*,*) 'TO CRT,  ENTER N  (0=NO SAMPLING TO SCREEN)'
        READ*, NVIEW
C
      IF(NSETS.GT.8) THEN
          WRITE(*,*) 'ONLY 8 SETS OF PLOTS AND TABLES CAN BE OUTPUT,'
          WRITE(*,*) 'THE FIRST 8 SETS ARE AUTOMATICALLY CHOSEN,'
        WRITE(*,*) 'BUT ALL DATA WILL BE ANALYZED FOR HARMONIC'
          WRITE(*,*) 'CONTENT AND THE RESULTS WILL BE STORED'
          WRITE(*,*) ' '
          WRITE(*,*) ' '
      ENDIF
C
C       GET SELECTIVE SAMPLING INFORMATION
C
 101    WRITE(*,*) 'DO YOU WISH TO USE SELECTIVE SAMPLING? (YES=1)'
        READ*, ISEL
C
        IF(ISEL.NE.1) GO TO 110
        WRITE(*,*)'SELECT WINDOW SIZES FOR TWO SELECTIVE SAMPLING'
        WRITE(*,*)'CHANNELS. IF YOU ONLY WANT TO USE ONE CHANNEL SELECT'
        WRITE(*,*)'ANY CHANNEL CONTAINING DATA FOR THE SECOND CHANNEL'
        WRITE(*,*)'AND USE A VERY LARGE WINDOW SIZE (I.E. +/- 1.E20)'
        WRITE(*,*) ' '
        DO 120 IWIN=1,2
        WRITE(*,*) ' '
        WRITE(*,*) 'ENTER ITEM NUMBER OF SELECTION WINDOW',IWIN
        READ*,IWINDO(IWIN)
        WRITE(*,*) 'ENTER MINIMUM ACCEPTABLE VALUE FOR ITEM',
     +               IWINDO(IWIN)
        READ*,WINDOW(IWIN,1)
        WRITE(*,*) 'ENTER MAXIMUM ACCEPTABLE VALUE FOR ITEM',
     +               IWINDO(IWIN)
        READ*,WINDOW(IWIN,2)
 120    CONTINUE
C
C       INITIALIZE VALUES
C
 110    SCALE=(XMAX-XMIN)/FLOAT(NBIN)
      SUMERR=0.
C
C     OPEN OUTPUT FILES
C
        OPEN(UNIT=25,FILE='HARMCONT.DAT',STATUS='NEW',
     +          FORM='UNFORMATTED',IOSTAT=IERR)
        IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING HARMCONT.DAT FILE (UNIT 20)'
       WRITE(*,*) 'IERR= ',IERR
       STOP
```

```
        ENDIF
C
        OPEN(UNIT=10,FILE='BINTBL.DAT',STATUS='NEW',IOSTAT=IERR)
        IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING BINTBL.DAT FILE (UNIT 10)'
       WRITE(*,*) 'IERR= ',IERR
       STOP
        ENDIF
C
        OPEN(UNIT=35,FILE='HARMINFO.DAT',STATUS='NEW',IOSTAT=IERR)
        IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING HARMINFO.DAT FILE (UNIT 35)'
       WRITE(*,*) 'IERR= ',IERR
       STOP
        ENDIF
C
      IF (NFOUR.EQ.1) THEN
        OPEN(UNIT=50,FILE='BINPL.DAT',STATUS='NEW',IOSTAT=IERR)
         IF(IERR.NE.0) THEN
           WRITE(*,*) 'ERROR OPENING BINPL.DAT FILE (UNIT 50)'
         WRITE(*,*) 'IERR= ',IERR
         STOP
         ENDIF
      ENDIF
C
C     WRITE HEADING AND ETC
C
        WRITE(10,1101) TITLE
        WRITE(10,1030) NAME(XCHAN)
        WRITE(10,1040) NAME(YCHAN)
        WRITE(10,1050) NBIN,XMIN,XMAX
        WRITE(10,1110)  DELT
 1101   FORMAT(1H1//' METHOD-OF-BINS ANALYSIS'//1X,A80)
 1030   FORMAT(/1X,'INDEPENDENT VARIABLE (X) = ',A30)
 1040   FORMAT(1X,'DEPENDENT VARIABLE = ',A30)
 1050   FORMAT(/1X, I2,' BINS USED IN THE ANALYSIS'
     +         /1X,'RANGE OF BINS= ',F8.3, ' TO ',F8.3)
 1110   FORMAT(1X,'SAMPLE INTERVAL (SEC) = ',F8.3/)
        IF(ISEL.NE.1) THEN
          WRITE(10,1200)
 1200     FORMAT(1X,'SELECTIVE SAMPLING NOT USED')
        GOTO 400
        ENDIF
        WRITE(10,1210) NAME(IWINDO(1))
        WRITE(10,1220) WINDOW(1,1), WINDOW(1,2)
        WRITE(10,1230) NAME(IWINDO(2))
        WRITE(10,1220) WINDOW(2,1), WINDOW(2,2)
 1210   FORMAT(1X,'SELECTIVE SAMPLING USED IN THIS ANALYSIS',/
     +             5X,'THE FIRST WINDOW CHANNEL WAS ',A30)
 1220   FORMAT(5X,'MINIMUM ACCEPTED VALUE FOR THIS CHANNEL= ',E10.3,/,
     +         5X,'MAXIMUM ACCEPTED VALUE FOR THIS CHANNEL= ',E10.3)
 1230   FORMAT(/5X,'THE SECOND WINDOW CHANNEL WAS ',A30)
 400  ISETS=1
      NCOL=1
C
      IF(NFOUR.EQ.1) THEN
      WRITE(*,*) 'DO YOU WANT THE AZIMUTH AVERAGES ROTATED SO THE'
      WRITE(*,*) 'FUNCTION IS CONTINUOUS AT ZERO DEG.? (YES = 1)'
```

```
      READ*, NROT
      IF(NROT.EQ.1) WRITE(10,*) 'AZIMUTH AVERAGE ROTATED'
        WRITE(*,*) ' '
      WRITE(*,*) 'DO YOU WANT TO PLOT THE RESULTS? (YES = 1)'
      READ*, NSCREEN
      ENDIF
      XLABLE=NAME(XCHAN)
      YLABLE=NAME(YCHAN)
C
C       START MAIN LOOP FOR ANALYSIS
C
        WRITE(*,*) 'STARTING ANALYSIS.....'
 111  CONTINUE
      IREAD=0
      IF(ISETS.LE.8) WRITE(10,1060) ISETS,NSTART,NSTOP
 1060   FORMAT(1H1,1X,'SET # =',I3,', FIRST SCAN USED= ',I8,
     +         ', NUMBER OF SCANS= ',I8)
C
      NSTART=NSTOP*ISETS+1
C
        DO 5 I=1,30
      SUMCHAN(I)=0.
      ANGLE(I)=0.
        BIN(1,I)=0.D00
        BIN(2,I)=1.E20
        BIN(3,I)=-1.E20
        BIN(4,I)=0.D00
        BIN(5,I)=0.D00
 5      CONTINUE
C
 200    CONTINUE
C
C       READ ONE SCAN OF DATA
C
        READ(20,END=240) (D(I7),I7=1,NITEMS)
        IREAD=IREAD+1
C
C     SUM UP EACH CHANNEL FOR COMPUTING AVERAGES
C
      DO 205 I=1,NITEMS
      SUMCHAN(I)=SUMCHAN(I)+D(I)
 205  CONTINUE
C
C       PRINT DATA TO CRT IF DESIRED
C
        IF(NVIEW.EQ.0) GO TO 210
         IF(FLOAT(IREAD/NVIEW).EQ.FLOAT(IREAD)/FLOAT(NVIEW))
     +   WRITE(*,*)  IREAD, D(XCHAN), D(YCHAN)
C
C       SORT DATA INTO APPROPRIATE BIN
C
C       SKIP DATA IF NOT INSIDE 'WINDOWS'
C
 210    IF(ISEL.NE.1) GO TO 230
         CALL SELSAM(D(IWINDO(1)),D(IWINDO(2)),WINDOW,IFLAG)
        IF(IFLAG.EQ.0) GO TO 220
 230    X=D(XCHAN)
        Y=D(YCHAN)
```

```
C
      CALL FILBINS(BIN,ANGLE,XMIN,XMAX,SCALE,NBIN,X,Y)
C
C       RETURN FOR MORE DATA UNTIL STOP TIME
C
 220    IF(IREAD.LT.NSTOP) GO TO 200
 240  CONTINUE
C
C       COMPUTE FINAL AVERAGES, ETC. FOR EACH SET OF SCANS
C
      CALL BINSAVE(BIN,ANGLE,NBIN,XMIN,XMAX,NFOUR,NROT)
C
      DO 350 I=1,NITEMS
      SUMCHAN(I)=SUMCHAN(I)/FLOAT(IREAD)
 350  CONTINUE
C
C     PRINT RESULTS
C
      IF(ISETS.LE.8) THEN
      WRITE(10,1020)
 1020   FORMAT(/4X,'BIN',4X,'X',11X,'MEAN',9X,'MIN',9X,'MAX',
     +             7X,'STDEV',7X, '#SAMPLES'/)
C
C     PRINT TABLE AND FIND TOTAL NUMBER OF SAMPLES PASSED
C
        SUM=0.
      DO 310 I=1,NBIN
      WRITE(10,1010) I,ANGLE(I),(BIN(J,I),J=1,5)
 1010   FORMAT(3X,I2,5(2X,G10.4),4X,F6.0)
        SUM=SUM+BIN(5,I)
 310    CONTINUE
C
      WRITE(10,1070) SUM
 1070   FORMAT(/1X,'TOTAL NUMBER OF SAMPLES IN BINS= ',F10.1/)
      ENDIF
C
C     BEGIN FOURIER ANALYSIS LOOP IF DESIRED
C
      IF(NFOUR.NE.1) GOTO 660
      DO 360 I=1,NBIN
      B(I)=BIN(1,I)
 360  CONTINUE
C
      CALL FOURCO(B,ANGLE,HARM,NBIN,NOUT,AO)
C
      IF(ISETS.LE.8) THEN
        WRITE(10,1280) ISETS
        WRITE(10,1290)
        WRITE(10,1300) ((HARM(II,JJ),JJ=1,2),II=1,2)
        WRITE(10,1303)
 1280   FORMAT(/1X,'HARMONIC CONTENT OF SET #',I3)
 1290         FORMAT(/8X,'MEAN',11X,'STD DEV',8X,'MAX',11X,'PK-PK')
 1300   FORMAT(/3X,4(F12.4,3X)/)
 1303     FORMAT(/1X,'         AMPLITUDE         PHASE ANG'/)
       DO 500 II=1,NOUT
        WRITE(10,1310) II,HARM(II+2,1),HARM(II+2,2)
 1310   FORMAT(1X,I1,'P',2X,F12.4,5X,F12.4)
 500   CONTINUE
```

```
      ENDIF
C
C     COMPUTE RECONSTRUCTED FOURIER SERIES LINE FOR COMPARISON
C       WITH ORIGINAL AZIMUTH AVERAGED CURVE
C
      IF(ISETS.GT.8) GOTO 610
C
      CALL FOURFUN(ANGLE,HARM,BINPL,NCOL,NOUT,NPLOT,AO)
C
      NCOL=NCOL+1
 610  CONTINUE
C
C     COMPUTE ROOT MEAN SQUARE ERROR IN RECONSTRUCTED CURVE
C       COMPARED WITH ORIGINAL CURVE
C
      CALL ERRORFUN(BIN,B,ANGLE,HARM,SCRNPL,NBIN,NOUT,ISETS,NSCREEN,
     +                AO,PCTERROR)
C
      IF(ISETS.LE.8) WRITE(10,1360) PCTERROR
 1360 FORMAT(/ ' RMS ERROR IS',F8.4,' PERCENT OF PEAK-TO-PEAK')

      SUMERR=SUMERR+PCTERROR
C
C     OUTPUT TO FILE HARMCONT.DAT
C
      WRITE(25) ((HARM(II,JJ),II=1,NOUT+2),JJ=1,2),PCTERROR,
     +            (SUMCHAN(I),I=1,NITEMS)
C
C     RETURN FOR MORE SETS OF SCANS
C
 660  ISETS=ISETS+1
      IF(ISETS.LE.NSETS) GOTO 111
C
C     STORE PLOT OUTPUT FILE IF FOURIER ANALYSIS IS PERFORMMED
C
      IF(NFOUR.NE.1) GOTO 850
      AVERR=SUMERR/NSETS
      NLINES=NCOL-1
      DO 700 IROW=1,NPLOT
        WRITE(50,1400) (BINPL(ICOL,IROW),ICOL=1,NCOL)
 1400   FORMAT(1X,9(G11.5,1X))
 700  CONTINUE
      WRITE(*,1460) NSETS,NSTOP
      WRITE(*,1450) AVERR
      WRITE(10,1460) NSETS,NSTOP
      WRITE(10,1450) AVERR
 1460 FORMAT(/' TOTAL NUMBER OF SCANS =',I4,' *',I6)
 1450 FORMAT(/' AVERAGE R.M.S. ERROR FOR THIS RUN =',F8.4/)
        WRITE(*,*) 'THESE RESULTS HAVE BEEN SAVED IN FILE BINTBL.DAT'
      WRITE(*,*) NLINES,' SETS OF PLOT DATA HAVE BEEN STORED IN'
      WRITE(*,*) '          BINPL.DAT'
C
C     SHOW GRAPHS ON THE SCREEN IF DESIRED
C
      IF(NSCREEN.NE.1) GOTO 860
C
      CALL SCRNVIEW(NBIN,NPLOT,XLABLE,YLABLE,TITLE,SCRNPL,BINPL)
C
```

```
      GOTO 860
 850    WRITE(*,*) 'THESE RESULTS HAVE BEEN SAVED IN FILE BINTBL.DAT'
C
 860  CONTINUE
        WRITE(35,2000) TITLE
 2000 FORMAT(A)
        WRITE(35,2020) NSETS,NOUT,YCHAN
 2020   FORMAT(4(1X,I10))
C
C     CLOSE ALL DATA FILES
C
      IF(NFOUR.EQ.1) CLOSE(50)
      CLOSE(10)
        CLOSE(20)
        CLOSE(25)
        CLOSE(35)
C
        STOP
C
 9999   WRITE(*,*) 'HIT END-OF-FILE GOING TO START POINT'
        STOP
        END
```

```
C       A COLLECTION OF SUBROUTINES USED WITH PROGRAM HARMON
C       SUBROUTINES IN THIS COLLECTION INCLUDE:
C
C         FILBINS
C         BINSAVE
C         FOURFUN
C         ERRORFUN
C         SCRNVIEW
C
C       THE SUBROUTINE DATPLOT IS CALLED FROM WITHIN THESE ROUTINES
C
C       THIS FILE LAST MODIFIED 11/86
C       AC HANSEN, UNIV OF UTAH
C
C       ************************
C
      SUBROUTINE FILBINS(BIN,ANGLE,XMIN,XMAX,SCALE,NBIN,X,Y)
C
C     SUBROUTINE TO BE USED AS PART OF THE HARMON PROGRAM.
C     USING THE METHOD OF BINS TO SORT DATA INTO APPROPRIATE
C     BINS
C       IF DATA OUTSIDE RANGE SELECTED, DEFAULT TO END BIN
C
      DOUBLE PRECISION BIN(5,30)
      DIMENSION ANGLE(30)
C
        IBIN=INT((X-XMIN)/SCALE) + 1
        IF(IBIN.GT.NBIN) IBIN=NBIN
        IF(IBIN.LT.1) IBIN=1
                      ANGLE(IBIN)=ANGLE(IBIN)+(X-XMIN)
                             BIN(1,IBIN)=BIN(1,IBIN) + Y
        IF(Y.LT.BIN(2,IBIN)) BIN(2,IBIN)=Y
        IF(Y.GT.BIN(3,IBIN)) BIN(3,IBIN)=Y
                             BIN(4,IBIN)=BIN(4,IBIN) +Y*Y
                             BIN(5,IBIN)=BIN(5,IBIN) + 1.
C
C
      RETURN
      END
C
C       ********************
C
      SUBROUTINE BINSAVE(BIN,ANGLE,NBIN,XMIN,XMAX,NFOUR,NROT)
C
C     SUBROUTINE USED AS PART OF THE PROGRAM HARMON
C     AVERAGING THE CUMULATED BINS VALUES, IF THE BIN
C     IS EMPTY, FILL THE BINS WITH AVERAGE VALUES OF
C     THE NEIGHBOURING BINS. IF TWO ADJACENT BINS ARE EMPTY
C       STOP THE PROGRAM
C
      DOUBLE PRECISION BIN(5,30)
      DIMENSION ANGLE(30)
C
        DO 30 I=1,NBIN
          IF(BIN(5,I).LT.1.) THEN
            BIN(4,I)=0.
            GO TO 30
          END IF
```

```
          IF(BIN(5,I).EQ.1.) THEN
            BIN(4,I)=0.
          ANGLE(I)=ANGLE(I)+XMIN
            GO TO 30
          END IF
        BIN(1,I)=BIN(1,I)/BIN(5,I)
      ANGLE(I)=ANGLE(I)/BIN(5,I)+XMIN
        BIN(4,I)=BIN(4,I)-BIN(5,I)*BIN(1,I)*BIN(1,I)
          IF(BIN(4,I).LT. 0.) THEN
            BIN(4,I)=-100.
            GO TO 30
          END IF
        BIN(4,I)=SQRT(BIN(4,I)/(BIN(5,I)-1.))
 30    CONTINUE
C
      IF(NFOUR.NE.1) GOTO 40
      DO 50 I=1,NBIN-1
      IF(BIN(5,I).EQ.0.0 .AND. BIN(5,I+1).EQ.0.0) THEN
      WRITE(*,*) 'CONSECUTIVE ZEROS IN BINS, RUN HAS BEEN TERMINATED'
      WRITE(*,*) 'TRY MORE SCANS IN EACH SET OR REDUCE THE NUMBER OF'
      WRITE(*,*) 'AZIMUTH BINS'
      STOP
      ENDIF
        IF(BIN(5,I).EQ.0.0) THEN
          BIN(1,I)=(BIN(1,I-1)+BIN(1,I+1))/2.
          BIN(2,I)=BIN(1,I)
          BIN(3,I)=BIN(1,I)
          BIN(4,I)=0.0
          ANGLE(I)=((FLOAT(I)-0.5)/NBIN)*(XMAX-XMIN)+XMIN
        ENDIF
 50   CONTINUE
C
C      HANDLE END POINT CHECK, USE LINEAR EXTRAPOLATION IF NEEDED
C
        IF(BIN(5,NBIN).EQ.0.0) THEN
          BIN(1,NBIN)=2.*BIN(1,NBIN-1) - BIN(1,NBIN-2)
          BIN(2,NBIN)=BIN(1,NBIN)
          BIN(3,NBIN)=BIN(1,NBIN)
          BIN(4,NBIN)=0.0
          ANGLE(NBIN)=((FLOAT(NBIN)-0.5)/NBIN)*(XMAX-XMIN)+XMIN
        ENDIF
C
C     ROTATE THE AVERAGE BINS IF DESIRED
C
      IF(NROT.EQ.1) THEN
        DIFF=BIN(1,NBIN)-BIN(1,1)
        SLP=DIFF/(ANGLE(NBIN)-ANGLE(1))
       DO 55 I=1,NBIN
         BIN(1,I)=BIN(1,I)-SLP*ANGLE(I)
 55    CONTINUE
      ENDIF
C
 40   CONTINUE
C
      DO 60 I=1,NBIN
      IF(BIN(5,I).EQ.0) THEN
         BIN(2,I)=0.
         BIN(3,I)=0.
```

```
         BIN(4,I)=0.
         ANGLE(I)=((FLOAT(I)-0.5)/NBIN)*(XMAX-XMIN)+XMIN
       ENDIF
 60    CONTINUE
C
C
       RETURN
       END
C
C
C        ********************
C
       SUBROUTINE FOURFUN(ANGLE,HARM,BINPL,NCOL,NOUT,NPLOT,AO)
C
C      SUBROUTINE TO RECONSTRUCT THE TIME SERIES FROM THE
C        FOURIER COEFFICIENTS FOUND IN SUBROUTINE FOURCO
C
       DIMENSION ANGLE(30),HARM(12,2),BINPL(9,1000)
C
       PI=4*ATAN(1.0)
       NPLOT=0
C
       DO 10 X=0.,2*PI,0.05
         FX=0.0
        DO 20 I=1,NOUT
        FX=FX+(HARM(I+2,1)*COS(I*X+HARM(I+2,2)/(180/PI)))
 20     CONTINUE
         FX=AO+FX
         NPLOT=NPLOT+1
       IF(NCOL.EQ.1) BINPL(1,NPLOT)=X*180/PI
         BINPL(NCOL+1,NPLOT)=FX
 10    CONTINUE
C
C
       RETURN
       END
C
C
C        **********************
C
       SUBROUTINE
ERRORFUN(BIN,B,ANGLE,HARM,SCRNPL,NBIN,NOUT,ISETS,NSCREEN,
     +                          AO,PCTERROR)
C
C      SUBROUTINE TO COMPUTE THE ROOT MEAN SQUARE ERROR IN THE
C        RECONSTRUCTED TIME SERIES COMPARED WITH THE ORIGINAL TIME SERIES
C      THE ERROR IS EXPRESSED AS A PERCENT OF THE PEAK-TO-PEAK VALUE
C
       DOUBLE PRECISION BIN(5,30)
       DIMENSION ANGLE(30),HARM(12,2),B(30),SCRNPL(8,3,30)
       COMMON ERRFUN(30)
C
       PI=4*ATAN(1.)
       ERROR=0.
C
       DO 10 I=1,NBIN
          FX=0.0
        DO 20 J=1,NOUT
```

```
      FX=FX+(HARM(J+2,1)*COS(J*ANGLE(I)+HARM(J+2,2)/(180/PI)))
 20   CONTINUE
        FX=AO+FX
          IF(NSCREEN.EQ.1 .AND. ISETS.LE.8) THEN
            SCRNPL(ISETS,1,I)=ANGLE(I)*180/PI
            SCRNPL(ISETS,2,I)=BIN(1,I)
            SCRNPL(ISETS,3,I)=FX
          ENDIF
        ERRFUN(I)=FX
 10   CONTINUE
      DO 30 I=1,NBIN
        ERROR=ERROR+(B(I)-ERRFUN(I))**2.
 30   CONTINUE
      ERROR=SQRT(ERROR/FLOAT(NBIN))
      PCTERROR=100*ERROR/HARM(2,2)
C
C
      RETURN
      END
C
C
C       ******************
C
      SUBROUTINE SCRNVIEW(NBIN,NPLOT,XLABLE,YLABLE,TITLE,
     +                          SCRNPL,BINPL)
C
C     SUBROUTINE TO BE USED WITH PROGRAM HARMON
C     THE PURPOSE OF THIS PROGRAM IS TO GRAPH ORIGINAL AND/OR
C       RECONSTRUCTED TIME SERIES.  RECONSTRUCTED AND ORIGINAL
C       SERIES CAN BE COMPARED OR SEVERAL DIFFERENT RECONSTRUCTED
C       SERIES CAN BE COMPARED.
C
      DIMENSION SCREEN(3,30),SCRNPL(8,3,30),ILINE(9),ICMD(9,9),
     +         BINPL(9,1000),PLOT1(2,1000),PLOT2(3,1000),
     +             PLOT3(4,1000),PLOT4(5,1000),PLOT5(6,1000),
     +             PLOT6(7,1000),PLOT7(8,1000),PLOT8(9,1000)
        CHARACTER*80 TITLE
        CHARACTER*50 XLABLE,YLABLE
      LINESTYL=0
C
      WRITE(*,*) 'DO YOU WANT TO COMPARE THE RECONSTRUCTED AND'
      WRITE(*,*) 'THE AVERAGE BINS FUNCTION?  (YES = 1)'
      READ*, ICOMPARE
      IF(ICOMPARE.NE.1) GOTO 30
C
 10   WRITE(*,*) 'ENTER NUMBER OF THE SET YOU WANT TO COMPARE'
      READ*, ISCREEN
C
      DO 20 I=1,NBIN
      SCREEN(1,I)=SCRNPL(ISCREEN,1,I)
      SCREEN(2,I)=SCRNPL(ISCREEN,2,I)
      SCREEN(3,I)=SCRNPL(ISCREEN,3,I)
 20   CONTINUE
C
      CALL DATPLOT(SCREEN,NBIN,2,XLABLE,YLABLE,TITLE,ICMD,LINESTYL)
C
      WRITE(*,*) 'WANT TO COMPARE MORE SETS OF'
        WRITE(*,*) 'RECONSTRUCTED AND ORIGINAL CURVES? (YES = 1)'
```

```
      READ*, MORE
      IF (MORE.EQ.1) GOTO 10
C
 30   WRITE(*,*) 'DO YOU WANT TO SEE RECONSTRUCTED'
      WRITE(*,*) 'TIME SERIES LINES? (YES =1 )'
      READ*, LOOKS
      IF(LOOKS.NE.1) RETURN
C
 33   WRITE(*,*) 'HOW MANY LINES DO YOU WANT TO SEE?'
      READ*, LINE
      DO 35 I=1,LINE
      WRITE(*,*) 'ENTER SET NUMBER FOR LINE #',I
      READ*,ILINE(I)
 35   CONTINUE
C
        DO 45 J=1,LINE
        DO 40 I=1,NPLOT
      IF(LINE.EQ.1) THEN
        IF(J.EQ.1) PLOT1(1,I)=BINPL(1,I)
        PLOT1(J+1,I)=BINPL(ILINE(J)+1,I)
        GOTO 40
        ENDIF
      IF(LINE.EQ.2) THEN
        IF(J.EQ.1) PLOT2(1,I)=BINPL(1,I)
        PLOT2(J+1,I)=BINPL(ILINE(J)+1,I)
        GOTO 40
        ENDIF
      IF(LINE.EQ.3) THEN
        IF(J.EQ.1) PLOT3(1,I)=BINPL(1,I)
        PLOT3(J+1,I)=BINPL(ILINE(J)+1,I)
        GOTO 40
        ENDIF
      IF(LINE.EQ.4) THEN
        IF(J.EQ.1) PLOT4(1,I)=BINPL(1,I)
        PLOT4(J+1,I)=BINPL(ILINE(J)+1,I)
        GOTO 40
        ENDIF
      IF(LINE.EQ.5) THEN
        IF(J.EQ.1) PLOT5(1,I)=BINPL(1,I)
        PLOT5(J+1,I)=BINPL(ILINE(J)+1,I)
        GOTO 40
        ENDIF
      IF(LINE.EQ.6) THEN
        IF(J.EQ.1) PLOT6(1,I)=BINPL(1,I)
        PLOT6(J+1,I)=BINPL(ILINE(J)+1,I)
        GOTO 40
        ENDIF
      IF(LINE.EQ.7) THEN
        IF(J.EQ.1) PLOT7(1,I)=BINPL(1,I)
        PLOT7(J+1,I)=BINPL(ILINE(J)+1,I)
        GOTO 40
        ENDIF
      IF(LINE.EQ.8) THEN
        IF(J.EQ.1) PLOT8(1,I)=BINPL(1,I)
        PLOT8(J+1,I)=BINPL(ILINE(J)+1,I)
        ENDIF
 40   CONTINUE
 45   CONTINUE
```

```
C
      IF(LINE.EQ.1) THEN
      CALL DATPLOT(PLOT1,NPLOT,LINE,XLABLE,YLABLE,TITLE,ICMD,LINESTYL)
      GOTO 48
      ENDIF
      IF(LINE.EQ.2) THEN
      CALL DATPLOT(PLOT2,NPLOT,LINE,XLABLE,YLABLE,TITLE,ICMD,LINESTYL)
      GOTO 48
      ENDIF
      IF(LINE.EQ.3) THEN
      CALL DATPLOT(PLOT3,NPLOT,LINE,XLABLE,YLABLE,TITLE,ICMD,LINESTYL)
      GOTO 48
      ENDIF
      IF(LINE.EQ.4) THEN
      CALL DATPLOT(PLOT4,NPLOT,LINE,XLABLE,YLABLE,TITLE,ICMD,LINESTYL)
      GOTO 48
      ENDIF
      IF(LINE.EQ.5) THEN
      CALL DATPLOT(PLOT5,NPLOT,LINE,XLABLE,YLABLE,TITLE,ICMD,LINESTYL)
      GOTO 48
      ENDIF
      IF(LINE.EQ.6) THEN
      CALL DATPLOT(PLOT6,NPLOT,LINE,XLABLE,YLABLE,TITLE,ICMD,LINESTYL)
      GOTO 48
      ENDIF
      IF(LINE.EQ.7) THEN
      CALL DATPLOT(PLOT7,NPLOT,LINE,XLABLE,YLABLE,TITLE,ICMD,LINESTYL)
      GOTO 48
      ENDIF
      IF(LINE.EQ.8) THEN
      CALL DATPLOT(PLOT8,NPLOT,LINE,XLABLE,YLABLE,TITLE,ICMD,LINESTYL)
      ENDIF
C
 48   WRITE(*,*) 'WANT TO SEE MORE RECONSTRUCTED LINES? (YES = 1)'
      READ*,IOTHER
      IF(IOTHER.EQ.1) GOTO 33

C
      RETURN
      END
```

```
      SUBROUTINE DATPLOT(BIN,NPOINT,NLINES,XNAME,YNAME,TITLE,ICMD,
     +                   LINESTYL)
C
C     ROUTINE PLOTS MULTIPLE OR SINGLE LINES
C     USING THE PLOT79 PACKAGE ON THE VAX
C
C     THIS ROUTINE IS PART OF THE SERI WIND ENERGY DATA ANALYSIS
C     PACKAGE AND ALSO CAN BE USED AS A GENERAL PLOTTING SUBROUTINE
C
C     VARIABLES:
C       BIN=OUTPUT OF HARMON PROGRAM AS DESCRIBED IN HARMON.FOR
C         ( ARRAY TO BE PLOTTED )
C       XNAME=NAME OF X-AXIS (INDEPENDENT VARIABLE)
C       YNAME=NAME OF Y-AXIS (DEPENDENT VARIABLE)
C       TITLE=TITLE OF PLOT (FROM PREVIOUS PROGRAMS)
C
C     WRITTEN BY C. HANSEN   4/86  UNIVERSITY OF UTAH
C     LAST MODIFIED  11/86
C
        DIMENSION BIN(NLINES+1,NPOINT),ICMD(NLINES,9),BNDRY(2,3)
        CHARACTER*80 TITLE
        CHARACTER*50 XNAME, YNAME
        LOGICAL XLOG,XSCALE,YLOG,YSCALE
C
        DO 5 I=1,2
        DO 6 J=1,3
        BNDRY(I,J)=0.
  6     CONTINUE
  5     CONTINUE
C
C     ASSIGN PLOT VARIABLES
C
        NCOLS=NPOINT
        NROWS=NLINES+1
        NFONT=42
        IBOX=2
        XLOG=.FALSE.
        YLOG=.FALSE.
        XSCALE=.TRUE.
        YSCALE=.TRUE.
C
        DO 20 I=1,NLINES
        ICMD(I,1)=NPOINT
        ICMD(I,2)=1
        ICMD(I,3)=1
        ICMD(I,4)=1
        ICMD(I,5)=I+1
        ICMD(I,6)=1
        ICMD(I,7)=1
        ICMD(I,8)=1
      ICMD(I,9)=I
      IF(LINESTYL.NE.0)  ICMD(I,9)=LINESTYL
  20    CONTINUE
C
C
        CALL GRAPH2D(BIN,NROWS,NCOLS,ICMD,NLINES,NFONT,TITLE,
     +       XNAME,XLOG,XSCALE,YNAME,YLOG,YSCALE,IBOX,BNDRY)
C
```

```
RETURN
END
```

```
      SUBROUTINE FOURCO(POINT,ANGLE,HARM,NDAT,NTERM,AO)
C
C     THIS SUBROUTINE REPRESENTS THE DATA FUNCTION AS A FOURIER SERIES.
C     THE FOURIER COEFFICIENTS ARE CALCULATED AND PRESENTED
C       IN  AMPLITUDE, AND PHASE ANGLE FORM
C
C     VARIABLES:
C         POINT=INPUT AZIMUTH SERIES FUNCTION
C         ANGLE=AZIMUTH ANGLE INPUT IN DEGREES
C         HARM(1,1)=ARITHMETIC MEAN OF POINT OVER THE ANGLE CYCLE
C         HARM(1,2)=STANDARD DEVIATION OVER THE CYCLE
C         HARM(2,1)=MAXIMUM VALUE
C       HARM(2,2)=PK-PK VALUE
C       HARM(J+2,1)=FOURIER AMPLITUDE COEFFICIENTS
C       HARM(J+2,2)=FOURIER PHASE ANGLE IN DEGREES
C         NTERM=NUMBER OF HARMONICS CALCULATED (USUALLY 4)
C         NDAT=NUMBER OF DATA POINTS IN CYCLE (MUST BE >=2*NTERM)
C
C       THE AMPLITUDE AND PHASE ARE "WRAPPED" TO LIMIT PHASE ANGLE
C       TO THE RANGE 0<PHASE<180.  THIS MEANS THAT AMPLITUDES
C       CAN BE NEGATIVE OR POSITIVE.
C
C       WRITTEN 9/86 H. FAN, UNIV. OF UTAH
C       MODIFIED 11/86 C. HANSEN, UNIV OF UTAH
C
C       VAX FORTRAN 77
C
      DIMENSION POINT(100),ANGLE(100),HARM(12,2)
        DIMENSION SLOPE(100),PTINT(100),AN(10),BN(10)
      PI = 4*ATAN(1.0)
        PI2INV=0.5/PI
C
      IF(NDAT.LT.2*NTERM) THEN
        WRITE(*,*) 'NOT ENOUGH SAMPLES IN CYCLE FOR FOURIER ANALYSIS'
          WRITE(*,*) 'USING NTERM=',NTERM,' HARMONICS.  CHANGE NDAT'
          WRITE(*,*) 'NUMBER OF HARMONICS (NTERM) IN SOURCE CODE'
          STOP
      ENDIF
C
C       COMPUTE AVERAGE, STD. DEV., MIN., MAX., PK-PK, LINEAR TREND
C
      SUM=0.0
      SUM2=0.0
      XMIN=1.E20
      XMAX=-1.E20
C
      DO 10 I=1,NDAT
      SUM=SUM+POINT(I)
      SUM2=SUM2+POINT(I)*POINT(I)
      IF(POINT(I).LT.XMIN) XMIN=POINT(I)
      IF(POINT(I).GT.XMAX) XMAX=POINT(I)
 10   CONTINUE
      HARM(1,1)=SUM/FLOAT(NDAT)
C
C       COMPUTE STD. DEVIATION
C
      HARM(1,2)=SUM2-NDAT*HARM(1,1)*HARM(1,1)
      IF(HARM(1,2).LT.0.0) THEN
```

```
         HARM(1,2)=0.0
         GOTO 5
       END IF
       HARM(1,2)=SQRT(HARM(1,2)/(NDAT-1))
 5     HARM(2,1)=XMAX
       HARM(2,2)=XMAX-XMIN
C
C      CONVERT ANGLE TO RADIANS
C
       DO 15 I=1,NDAT
         ANGLE(I)=ANGLE(I)*.01745329
 15    CONTINUE
C
C        CALCULATE SLOPE AND INTERCEPT FOR EACH INTERVAL
C        THE FOURIER COEFFICIENTS ARE CALCULATED BY REPRESENTING
C        EACH CYCLE AS A SERIES OF STRAIGHT LINE SEGMENTS.  THE
C        FOURIER INTEGRALS ARE EVALUATED EXACTLY FOR EACH LINE SEGMENT
C        AND ALL SEGMENTS ARE SUMMED TO COMPLETE THE CYCLE.
C        THE SLOPE AND INTERCEPT CALCULATED HERE ARE FOR EACH SEGMENT
C
       DO 30 I = 1,NDAT-1
       SLOPE(I)=(POINT(I+1)-POINT(I))/(ANGLE(I+1)-ANGLE(I))
       PTINT(I)=(POINT(I)+POINT(I+1)-SLOPE(I)*(ANGLE(I+1)
      +              +ANGLE(I)))/2.0
 30    CONTINUE
C
C        CALCULATE COEFFICIENTSF Ao, An, Bn, AMPLITUDE, AND PHASE ANGLE--
C        (LOOP 40)
C
       DO 40 N=1,NTERM
             C=FLOAT(N)
         DO 50 I=1,NDAT-1
C
               A = ANGLE(I)
             B = ANGLE(I+1)
C
C        SUMMATION FROM ZERO DEGREES TO ANGLE(1)
C
       IF(I.EQ.1) THEN
           IF(N.EQ.1)  AO=PI2INV*((SLOPE(I)/2)*(A*A)+PTINT(I)*(A))
           AN(N)=FUNCTAN(SLOPE(I),C,A,0,PTINT(I),PI)
           BN(N)=FUNCTBN(SLOPE(I),C,A,0,PTINT(I),PI)
       END IF
C
C        SUMMATION FOR COEFFICIENTS FROM ANGLE(1) TO ANGLE(NDAT)
C
       IF(N.EQ.1) AO=AO+PI2INV*((SLOPE(I)/2)*(B*B-A*A)
      +                    +PTINT(I)*(B-A))
       AN(N)=AN(N)+FUNCTAN(SLOPE(I),C,B,A,PTINT(I),PI)
       BN(N)=BN(N)+FUNCTBN(SLOPE(I),C,B,A,PTINT(I),PI)
C
C        SUMMATION FOR COEFFICIENTS FROM ANGLE(NDAT) TO 360 DEGREES
C
       IF(I.EQ.NDAT-1) THEN
             PNTLAST=2*PI
               IF(N.EQ.1) AO=AO+PI2INV*((SLOPE(I)/2)*(PNTLAST**2
      +                    -B**2)+PTINT(I)*(PNTLAST-B))
             AN(N)=AN(N)+FUNCTAN(SLOPE(I),C,PNTLAST,B,PTINT(I),PI)
```

```
            BN(N)=BN(N)+FUNCTBN(SLOPE(I),C,PNTLAST,B,PTINT(I),PI)
          END IF
 50       CONTINUE
 40       CONTINUE
C
C           CONVERT COEFFICIENTS TO AMPLITUDE AND PHASE ANGLE
C
          DO 60 I=1,NTERM
          HARM(I+2,1)=SQRT(AN(I)*AN(I)+BN(I)*BN(I))
          HARM(I+2,2)=ATAN2(-BN(I),AN(I))*180./PI
 60       CONTINUE
C
          RETURN
          END
C
C
C           ***************************
C
          REAL FUNCTION FUNCTAN(SLOPE,CN,B,A,POINT,PI)
C
C         THIS FUNCTION IS THE EXACT INTEGRATION OF An OF FOURIER SERIES
C         COEFFICIENT.
C
          FUNCTAN=(1/PI)*(SLOPE*((1/CN**2)*(COS(CN*B)-COS(CN*A))+(B/CN)
         +                *SIN(CN*B)-(A/CN)*SIN(CN*A))+(POINT/CN)*(SIN(CN*B)
         +           -SIN(CN*A)))

          RETURN
          END
C
C           ***************************
C
          REAL FUNCTION FUNCTBN(SLOPE,CN,B,A,POINT,PI)
C
C         THIS FUNCTION IS THE EXACT INTEGRATION OF Bn OF FOURIER SERIES
C         COEFFICIENT.
C
          FUNCTBN=(1/PI)*(SLOPE*((1/CN**2)*(SIN(CN*B)-SIN(CN*A))-(B/CN)
         +                *COS(CN*B)+(A/CN)*COS(CN*A))-(POINT/CN)*(COS(CN*B)
         +             -COS(CN*A)))

          RETURN
          END
```

```
      PROGRAM HARMBINS
C
C     ONE OF THE SERIES OF PROGRAMS FOR SERI WIND TURBINE
C     DATA ANALYSIS
C
C     PROGRAM TO PERFORM METHOD OF BINS ANALYSIS
C     OF HARMONIC CONTENT DATA FROM PROGRAM HARMON
C     THE HARMONIC AMPLITUDE OR PHASE A 1P, 2P, ...NOUT(P)
C     CAN BE CORRELATED WITH CYCLE AVERAGE VALUES OF ANY
C     OF THE STANDARD DATA ITEMS (WIND SPEEDS, YAW RATE, ETC.)
C     OR WITH ANOTHER HARMONIC DATA CHANNEL
C
C     DATA IS READ FROM FILE 'HARMCONT.DAT', WITH MISCELLANEOUS
C     OTHER INFORMATION ABOUT THE DATA RECORD IN 'AZINFO.DAT'
C
C     VARIABLES:
C        HARM(I,2)=ARRAY OF HARMONIC DATA DESCRIBED IN PROGRAM HARMON
C        D(J)=DATA ARRAY FROM PROGRAM HARMON (FILE HARMCONT.DAT)
C             THIS ARRAY CONTAINS THE CYCLE AVERAGED VALUES OF
C             THE STANDARD DATA ITEMS
C
C        BIN(1,I)=AVERAGE IN EACH OF 30 BINS
C        BIN(2,I)=MINIMUM VALUE IN EACH BIN
C        BIN(3,I)=MAXIMUM VALUE IN EACH BIN
C        BIN(4,I)=STANDARD DEVIATION IN BIN
C        BIN(5,I)=NUMBER OF DATA POINTS IN BIN
C
C        BIN IS A DOUBLE PRECISION ARRAY
C
C        BINPL IS A SINGLE PRECISION DUPLICATE OF BIN FOR PLOTTING
C
C        XCHAN=INTEGER IDENTIFICATION OF INDEPENDENT VARIABLE
C        YCHAN=INTEGER IDENTIFICATION OF DEPENDENT VARIABLE
C
C     A CYCLE IS ONE REVOLUTION OF THE ROTOR
C
C     C. HANSEN  7/86   UNIVERSITY OF UTAH
C
      PARAMETER (NITEMS=30)
      DOUBLE PRECISION BIN(5,30)
      DIMENSION BINPL(5,30)
      DIMENSION D(NITEMS), HARM(12,2)
      DIMENSION WINDOW(2,2), IWINDO(2)
      INTEGER XCHAN, YCHAN
      INTEGER*4 NSTOP, IREAD, NSTART, NSCAN
      CHARACTER*80 TITLE
      CHARACTER*50 NAME(NITEMS), HNAME(24), YNAME(24)
      DATA D/NITEMS*0./
      DO 5 I=1,30
      BIN(1,I)=0.D00
      BIN(2,I)=1.E20
      BIN(3,I)=-1.E20
      BIN(4,I)=0.D00
      BIN(5,I)=0.D00
 5    CONTINUE
C
C     GET NAMES OF HARMONIC VARIABLES
C
```

```
      HNAME(1)='CYCLE MEAN '
      HNAME(2)='CYCLE STD. DEV. '
      HNAME(3)='CYCLE MAX '
      HNAME(4)='CYCLE PK-PK '
      HNAME(5)='1P CYCLIC '
      HNAME(6)='1P PHASE '
      HNAME(7)='2P CYCLIC '
      HNAME(8)='2P PHASE '
      HNAME(9)='3P CYCLIC '
      HNAME(10)='3P PHASE '
      HNAME(11)='4P CYCLIC '
      HNAME(12)='4P PHASE '
      HNAME(13)='5P CYCLIC '
      HNAME(14)='5P PHASE '
      HNAME(15)='6P CYCLIC '
      HNAME(16)='6P PHASE '
      HNAME(17)='7P CYCLIC '
      HNAME(18)='7P PHASE '
      HNAME(19)='8P CYCLIC '
      HNAME(20)='8P PHASE '
      HNAME(21)='9P CYCLIC '
      HNAME(22)='9P PHASE '
      HNAME(23)='10P CYCLIC '
      HNAME(24)='10P PHASE '
C
C     READ NAMES FILE
C
      OPEN(UNIT=40,FILE='NAMES.DAT',STATUS='OLD',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING NAMES.DAT FILE (UNIT 40)'
         WRITE(*,*) 'IERR= ',IERR
         STOP
      ENDIF
      DO 2000 I=1,NITEMS
      READ(40,*) NAME(I)
 2000 CONTINUE
      CLOSE(40)
C
C     OPEN FILE 'HARMINFO' AND GET FILE INFORMATION
C
      OPEN(UNIT=30,FILE='HARMINFO.DAT',STATUS='OLD',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING HARMINFO.DAT (UNIT 30)'
         WRITE(*,*) 'IERR= ',IERR
         STOP
      ENDIF
      READ(30,3000) TITLE
      READ(30,3010) NREV,NOUT,ICHAN
 3000 FORMAT(A)
 3010 FORMAT(3(1X,I10))
C
C     CLOSE UNIT 30---HARMINFO.DAT
C
      CLOSE(30)
C
C     OPEN FILE 'HARMCONT.DAT' (UNIT 20) TO READ HARMONIC DATA
C
      OPEN(UNIT=20,FILE='HARMCONT.DAT',STATUS='OLD',
```

```
     +     FORM='UNFORMATTED',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING HARMCONT.DAT (UNIT 20)'
         WRITE(*,*) 'IERR= ',IERR
         STOP
      ENDIF
      REWIND 20
C
C     INPUT THE SETUP DATA FOR THIS RUN
C
      WRITE(*,*) 'THIS IS THE HARMONIC DATA METHOD OF BINS PROGRAM'
      WRITE(*,*) ' '
      WRITE(*,*) 'THE HARMONIC DATA CHANNEL IS ',NAME(ICHAN)
      WRITE(*,*) ' '
      WRITE(*,*) 'ENTER ITEM NUMBER FOR INDEPENDENT VARIABLE '
      WRITE(*,*) 'THIS CAN BE ANY ITEM FROM THE BASIC LIST'
      READ*,XCHAN
      WRITE(*,*) 'ENTER MINIMUM VALUE OF X FOR BINS ANALYSIS'
      READ*,XMIN
      WRITE(*,*) 'ENTER MAXIMUM VALUE OF X FOR BINS ANALYSIS'
      READ*,XMAX
      WRITE(*,*) 'ENTER NUMBER OF BINS (MAXIMUM OF 30)'
      READ*,NBIN
C
      WRITE(*,*) 'SELECT THE DEPENDENT VARIABLE FROM THE FOLLOWING LIST'
      WRITE(*,*) 'NUMBER       VARIABLE'
      WRITE(*,*) ' 1     CYCLE MEAN'
      WRITE(*,*) ' 2     CYCLE STANDARD DEVIATION'
      WRITE(*,*) ' 3     CYCLE MAXIMUM'
      WRITE(*,*) ' 4     CYCLE PEAK TO PEAK'
      DO 500 I=5,2*NOUT+4,2
      IP=I/2-1
      WRITE(*,510) I,IP,I+1,IP
 510  FORMAT(1X,I2,5X,'MAGNITUDE OF ',I2'P CYCLIC'/
     +       1X,I2,5X,'PHASE OF ',I2,'P CYCLIC')
 500  CONTINUE
C
      WRITE(*,*) 'ENTER NUMBER FOR DEPENDENT VARIABLE'
      READ*,YCHAN
      IY=(YCHAN-1)/2+1
      JY= 2 - MOD(YCHAN,2)
C
C     SAVE COMPLETE VARIABLE NAME
C
      DO 520 I=1,20
      JJ=INDEX(HNAME(I),'  ')
      YNAME(I)=HNAME(I)(1:JJ)//NAME(ICHAN)
  520 CONTINUE
C
C
      WRITE(*,*) 'ENTER CYCLE NUMBER FOR START OF ANALYSIS'
      WRITE(*,*) ' (0=START OF FILE)'
      READ*,NSTART
      WRITE(*,*) 'THERE ARE ',NREV,' CYCLES AVAILABLE IN THE FILE'
      WRITE(*,*) 'ENTER NUMBER OF CYCLES YOU WISH TO ANALYZE'
      READ*,NSTOP
C
C     SKIP INTO THE DESIRED START POINT
```

```
C
      IF(NSTART.EQ.0) GO TO 50
      DO 100 I=1,NSTART
      READ(20,END=240) ((HARM(II,JJ),II=1,NOUT+2),JJ=1,2),ERROR,
     +                  (D(II),II=1,NITEMS)
 100  CONTINUE
C
C
C     GET INFORMATION FOR SAMPLING OF RAW DATA TO SCREEN
C
  50  WRITE(*,*) 'IF YOU WISH YOU CAN PRINT EVERY Nth DATA PAIR TO CRT'
      WRITE(*,*) 'ENTER N     (0=NO SAMPLING TO SCREEN)'
      READ*, NVIEW
C
C     GET SELECTIVE SAMPLING INFORMATION
C
      WRITE(*,*) 'DO YOU WISH TO USE SELECTIVE SAMPLING? (YES=1)'
      READ*, ISEL
      IF(ISEL.NE.1) GO TO 110
      WRITE(*,*) 'SELECT WINDOW SIZES FOR TWO SELECTIVE SAMPLING'
      WRITE(*,*) 'CHANNELS. IF YOU ONLY WANT TO USE ONE CHANNEL SELECT'
      WRITE(*,*) 'ANY CHANNEL CONTAINING DATA FOR THE SECOND CHANNEL'
      WRITE(*,*) 'AND USE A VERY LARGE WINDOW SIZE (I.E. +/- 1.E20)'
      WRITE(*,*) ' '
      DO 120 IWIN=1,2
      WRITE(*,*) ' '
      WRITE(*,*) 'ENTER ITEM NUMBER FOR SELECTION WINDOW',IWIN
      READ*,IWINDO(IWIN)
      WRITE(*,*) 'ENTER MINIMUM ACCEPTABLE VALUE FOR ITEM',IWINDO(IWIN)
      READ*,WINDOW(IWIN,1)
      WRITE(*,*) 'ENTER MAXIMUM ACCEPTABLE VALUE FOR ITEM',IWINDO(IWIN)
      READ*,WINDOW(IWIN,2)
 120  CONTINUE
C
C     INITIALIZE VALUES
C
 110  IREAD=0
      SCALE=(XMAX-XMIN)/FLOAT(NBIN)
C
C     START MAIN LOOP FOR ANALYSIS
C
 200  CONTINUE
C
C     READ ONE CYCLE OF DATA
C
      READ(20,END=240) ((HARM(II,JJ),II=1,NOUT+2),JJ=1,2),ERROR,
     +                  (D(II),II=1,NITEMS)
      IREAD=IREAD+1
      X=D(XCHAN)
      Y=HARM(IY,JY)
C
C     PRINT DATA TO CRT IF DESIRED
C
      IF(NVIEW.EQ.0) GO TO 210
      IF(FLOAT(IREAD/NVIEW).EQ.FLOAT(IREAD)/FLOAT(NVIEW))
     +   WRITE(*,*)  IREAD, X, Y
C
C     SORT DATA INTO APPROPRIATE BIN
```

```
C
C     SKIP DATA IF NOT INSIDE 'WINDOWS'
C
  210 IF(ISEL.NE.1) GO TO 230
      CALL SELSAM(D(IWINDO(1)),D(IWINDO(2)),WINDOW,IFLAG)
      IF(IFLAG.EQ.0) GO TO 220
  230 IBIN=INT((X-XMIN)/SCALE) + 1
C
C     IF DATA OUTSIDE RANGE SELECTED, DEFAULT TO END BIN
C
      IF(IBIN.GT.NBIN) IBIN=NBIN
      IF(IBIN.LT.1) IBIN=1
      BIN(1,IBIN)=BIN(1,IBIN) + Y
      IF(Y.LT.BIN(2,IBIN)) BIN(2,IBIN)=Y
      IF(Y.GT.BIN(3,IBIN)) BIN(3,IBIN)=Y
      BIN(4,IBIN)=BIN(4,IBIN) +Y*Y
      BIN(5,IBIN)=BIN(5,IBIN) + 1.
C
C     RETURN FOR MORE DATA UNTIL STOP POINT
C
  220 IF(IREAD.LT.NSTOP) GO TO 200
C
 240  CONTINUE
C
C
C     CLOSE THE HARMCONT.DAT DATA FILE
C
      CLOSE(20)
C
C     COMPUTE FINAL AVERAGES, ETC
C
      DO 300 I=1,NBIN
      IF(BIN(5,I).LT.1.) THEN
         BIN(5,I)=0.
         GO TO 300
      END IF
      IF(BIN(5,I).EQ.1.) THEN
         BIN(4,I)=0.
         GO TO 300
      END IF
      BIN(1,I)=BIN(1,I)/BIN(5,I)
      BIN(4,1)=BIN(4,I)-BIN(5,I)*BIN(1,I)*BIN(1,I)
      IF(BIN(4,1).LT. 0.) THEN
         BIN(4,I)=-100.
         GO TO 300
      END IF
      BIN(4,I)=DSQRT(BIN(4,1)/(BIN(5,I)-1.))
 300  CONTINUE
C
C     PRINT RESULTS
C
      OPEN(UNIT=10,FILE='HBINOUT.DAT',STATUS='NEW')
      WRITE(*,1100) TITLE
      WRITE(10,1100) TITLE
 1100 FORMAT(1H1//' METHOD-OF-BINS ANALYSIS OF HARMONIC DATA'//1X,A80)
      WRITE(*,1030) NAME(XCHAN)
      WRITE(10,1030) NAME(XCHAN)
 1030 FORMAT(/1X,'INDEPENDENT VARIABLE (X) = ',A30)
```

```
      WRITE(*,1040) YNAME(YCHAN)
      WRITE(10,1040) YNAME(YCHAN)
 1040 FORMAT(1X,'DEPENDENT VARIABLE = ',A50)
      WRITE(*,1050) NBIN,XMIN,XMAX
      WRITE(10,1050) NBIN,XMIN,XMAX
 1050 FORMAT(/1X, I2,' BINS USED IN THE ANALYSIS'
     + /1X,'RANGE OF BINS= ',F8.3, ' TO ',F8.3)
      WRITE(*,1060) NREV
      WRITE(10,1060) NREV
 1060 FORMAT(/1X,'NUMBER OF AZIMUTH AVERAGES ANALYZED= ',I8,/)
      IF(ISEL.NE.1) THEN
         WRITE(*,1200)
         WRITE(10,1200)
 1200    FORMAT(1X,'SELECTIVE SAMPLING NOT USED')
         GO TO 400
      ENDIF
      WRITE(*,1210) NAME(IWINDO(1))
      WRITE(10,1210) NAME(IWINDO(1))
 1210 FORMAT(1X,'SELECTIVE SAMPLING USED IN THIS ANALYSIS',/
     +5X,'THE FIRST WINDOW CHANNEL WAS ',A30)
      WRITE(*,1220) WINDOW(1,1), WINDOW(1,2)
      WRITE(10,1220) WINDOW(1,1), WINDOW(1,2)
 1220 FORMAT(5X,'MINIMUM ACCEPTED VALUE FOR THIS CHANNEL= ',E10.3,/,
     +       5X,'MAXIMUM ACCEPTED VALUE FOR THIS CHANNEL= ',E10.3)
      WRITE(*,1230) NAME(IWINDO(2))
      WRITE(10,1230) NAME(IWINDO(2))
 1230 FORMAT(/5X,'THE SECOND WINDOW CHANNEL WAS ',A30)
      WRITE(*,1220) WINDOW(2,1), WINDOW(2,2)
      WRITE(10,1220) WINDOW(2,1), WINDOW(2,2)
 400  WRITE(*,1020)
      WRITE(10,1020)
 1020 FORMAT(//4X,'I',6X,'X',11X,'MEAN',9X,'MIN',9X,'MAX',7X,'STDEV',
     + 7X, '#SAMPLES'/)
C
C     PRINT TABLE AND FIND TOTAL NUMBER OF SAMPLES PASSED
C
      SUM=0.
      DO 310 I=1,NBIN
      IF(BIN(5,I).EQ.0) GO TO 310
      X=(FLOAT(I)-.5)*SCALE + XMIN
      WRITE(*,1010)I,X,(BIN(J,I),J=1,5)
      WRITE(10,1010) I,X,(BIN(J,I),J=1,5)
 1010 FORMAT(3X,I2,5(2X,G10.4),4X,F6.0)
      SUM=SUM+BIN(5,I)
 310  CONTINUE
C
      WRITE(*,1070) SUM
      WRITE(10,1070) SUM
 1070 FORMAT(/1X,'TOTAL NUMBER OF SAMPLES IN BINS= ',F10.1/)
      WRITE(*,*) ' '
      WRITE(*,*) 'THESE RESULTS HAVE BEEN SAVED IN FILE HBINOUT.DAT'
C
C     PLOT THE RESULTS IF DESIRED
C
      WRITE(*,*) 'WANT TO PLOT RESULTS? (YES=1)'
      READ*,IANS
      IF(IANS.EQ.1) THEN
         DO 600 I=1,5
```

```
      DO 610 J=1,30
      BINPL(I,J)=SNGL(BIN(I,J))
610   CONTINUE
600   CONTINUE
      CALL BINPLOT(BINPL,NBIN,XMIN,XMAX,
     +                NAME(XCHAN),YNAME(YCHAN),TITLE)
    ENDIF
    CLOSE(10)
    STOP
    END
```

```
      PROGRAM HARMTRANS
C
C     ONE OF THE SERIES OF PROGRAMS FOR SERI WIND TURBINE
C     DATA ANALYSIS
C
C     PROGRAM TO REWRITE HARMONIC DATA IN ASCII FORMAT
C        FOR TRANSFER TO MACINTOSH OR OTHER COMPUTER
C
C     DATA IS READ FROM FILE 'HARMCONT.DAT', WITH MISCELLANEOUS
C     OTHER INFORMATION ABOUT THE DATA RECORD IN 'AZINFO.DAT'
C     AND HARMINFO.DAT
C
C     VARIABLES:
C        HARM(I,2)=ARRAY OF HARMONIC DATA DESCRIBED IN PROGRAM HARMON
C        D(J)=DATA ARRAY FROM PROGRAM HARMON (FILE BINH2.DAT)
C             THIS ARRAY CONTAINS THE CYCLE AVERAGED VALUES OF
C             THE STANDARD DATA ITEMS
C      ERROR=RMS ERROR IN RECONSTRUCTED SERIES, COMPARED WITH ORIGINAL
C
C        XCHAN=INTEGER IDENTIFICATION OF INDEPENDENT VARIABLE
C        YCHAN=INTEGER IDENTIFICATION OF DEPENDENT VARIABLE
C
C     A CYCLE IS ONE REVOLUTION OF THE ROTOR
C
C
C     C. HANSEN  5/86   UNIVERSITY OF UTAH
C
      PARAMETER (NITEMS=30)
      DIMENSION D(NITEMS), HARM(12,2), ISCHAN(NITEMS),IHCHAN(24)
      DIMENSION WINDOW(2,2), IWINDO(2), X(20)
      INTEGER*4 NSTOP, IREAD, NSTART, NSCAN
      CHARACTER*80 TITLE
      CHARACTER*50 NAME(NITEMS)
      DATA D/NITEMS*0./
C
C     READ NAMES FILE
C
      OPEN(UNIT=40,FILE='NAMES.DAT',STATUS='OLD',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING NAMES.DAT FILE (UNIT 40)'
         WRITE(*,*) 'IERR= ',IERR
         STOP
      ENDIF
      DO 2000 I=1,NITEMS
      READ(40,*) NAME(I)
 2000 CONTINUE
      CLOSE(40)
C
C     OPEN FILE 'HARMINFO.DAT' AND GET FILE INFORMATION
C
      OPEN(UNIT=30,FILE='HARMINFO.DAT',STATUS='OLD',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING HARMINFO.DAT (UNIT 30)'
         WRITE(*,*) 'IERR= ',IERR
         STOP
      ENDIF
      READ(30,3000) TITLE
      READ(30,3010) NREV,NOUT,ICHAN
```

```
 3000 FORMAT(A)
 3010 FORMAT(4(1X,I10))
C
C     CLOSE UNIT 30---HARMINFO.DAT
C
      CLOSE(30)
C
C     OPEN FILE 'HARMCONT.DAT' (UNIT 20) TO READ TIME SERIES DATA
C
      OPEN(UNIT=20,FILE='HARMCONT.DAT',STATUS='OLD',
     +     FORM='UNFORMATTED',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING HARMCONT.DAT (UNIT 20)'
         WRITE(*,*) 'IERR= ',IERR
         STOP
      ENDIF
      REWIND 20
C
C     OPEN FILE TO RECEIVE ASCII DATA THAT IS TO BE TRANSFERRED
C
      OPEN(UNIT=10,FILE='HTRANS.DAT',STATUS='NEW',IOSTAT=IERR,
     +     RECL=20*11)
      IF(IERR.NE.0)THEN
         WRITE(*,*) 'ERROR OPENING HTRANS.DAT FILE (UNIT 10)'
         WRITE(*,*) 'IERR= ',IERR
         STOP
      ENDIF
C
C     INPUT THE SETUP DATA FOR THIS RUN
C
      WRITE(*,*) 'THIS IS THE HARMONIC DATA TRANSFER PROGRAM'
      WRITE(*,*) ' '
      WRITE(*,*) 'THE HARMONIC DATA CHANNEL IS ',NAME(ICHAN)
      WRITE(*,*) ' '
      WRITE(*,*) 'ENTER THE CHANNEL(S) YOU WISH TO TRANSFER'
      WRITE(*,*) 'IN TWO STEPS.  FIRST, SPECIFY THE HARMONIC'
      WRITE(*,*) 'DATA CHANNEL(S), THEN THE STANDARD CHANNEL(S)'
      WRITE(*,*)
      WRITE(*,*) 'YOU CAN TRANSFER UP TO 20 CHANNELS TOTAL'
      WRITE(*,*) ' '
      WRITE(*,*) 'ENTER THE NUMBER OF HARMONIC CHANNELS TO TRANSFER'
      READ*, NHTRAN
      DO 300 J=1,NHTRAN
      WRITE(*,*) 'ENTER THE NUMBER FOR TRANSFER CHANNEL',J
C
      WRITE(*,*) 'SELECT THE TRANSFER VARIABLES FROM THIS LIST'
      WRITE(*,*) 'NUMBER       VARIABLE'
      WRITE(*,*) ' 1     CYCLE MEAN'
      WRITE(*,*) ' 2     CYCLE STANDARD DEVIATION'
      WRITE(*,*) ' 3     CYCLE MAXIMUM'
      WRITE(*,*) ' 4     CYCLE PEAK TO PEAK'
      DO 500 I=5,2*NOUT+4,2
      IP=I/2-1
      WRITE(*,510) I,IP,I+1,IP
 510  FORMAT(1X,I2,5X,'MAGNITUDE OF ',I2'P CYCLIC'/
     +       1X,I2,5X,'PHASE OF ',I2,'P CYCLIC')
 500  CONTINUE
      WRITE(*,512) 2*NOUT+5
```

```
 512  FORMAT(1X,I2,'     PERCENT ERROR IN HARMONICS')
      READ*, IHCHAN(J)
  300 CONTINUE
C
      WRITE(*,*) 'ENTER NUMBER OF STANDARD CHANNELS TO TRANSFER'
      READ*, NSTRAN
      DO 310 J=1,NSTRAN
      WRITE(*,*) 'TRANSFER CHANNEL NUMBER ',J
      WRITE(*,*) 'ENTER ITEM NUMBER FOR THIS CHANNEL'
      READ*, ISCHAN(J)
  310 CONTINUE
      NTRAN=NHTRAN+NSTRAN
C
C
      WRITE(*,*) 'ENTER CYCLE NUMBER FOR START OF ANALYSIS'
      WRITE(*,*) ' (0=START OF FILE)'
      READ*,NSTART
      WRITE(*,*) 'THERE ARE ',NREV,' SETS AVAILABLE IN THE FILE'
      WRITE(*,*) 'ENTER NUMBER OF CYCLES YOU WISH TO ANALYZE'
      READ*,NSTOP
C
      WRITE(*,*) 'DO YOU WANT PHASE ANGLES IN RADIANS OR DEGREES?'
      WRITE(*,*) '0=DEGREES'
      WRITE(*,*) '1=RADIANS'
      READ(*,*) IRAD
C
C     SKIP INTO THE DESIRED START POINT
C
      IF(NSTART.EQ.0) GO TO 50
      DO 100 I=1,NSTART
      READ(20,END=240) ((HARM(II,JJ),II=1,NOUT+2),JJ=1,2),ERROR,
     +                  (D(II),II=1,NITEMS)
  100 CONTINUE
C
   50 CONTINUE
C
C     GET SELECTIVE SAMPLING INFORMATION
C
      WRITE(*,*) 'DO YOU WISH TO USE SELECTIVE SAMPLING? (YES=1)'
      READ*, ISEL
      IF(ISEL.NE.1) GO TO 110
      WRITE(*,*) 'SELECT WINDOW SIZES FOR TWO SELECTIVE SAMPLING'
      WRITE(*,*) 'CHANNELS. IF YOU ONLY WANT TO USE ONE CHANNEL SELECT'
      WRITE(*,*) 'ANY CHANNEL CONTAINING DATA FOR THE SECOND CHANNEL'
      WRITE(*,*) 'AND USE A VERY LARGE WINDOW SIZE (I.E. +/- 1.E20)'
      WRITE(*,*) ' '
      DO 120 IWIN=1,2
      WRITE(*,*) ' '
      WRITE(*,*) 'ENTER ITEM NUMBER OF SELECTION WINDOW',IWIN
      READ*,IWINDO(IWIN)
      WRITE(*,*) 'ENTER MINIMUM ACCEPTABLE VALUE FOR ITEM',IWINDO(IWIN)
      READ*,WINDOW(IWIN,1)
      WRITE(*,*) 'ENTER MAXIMUM ACCEPTABLE VALUE FOR ITEM',IWINDO(IWIN)
      READ*,WINDOW(IWIN,2)
  120 CONTINUE
C
C     INITIALIZE VALUES
C
```

```
 110  IREAD=0
C
C     START MAIN LOOP FOR ANALYSIS
C
 200  CONTINUE
C
C     READ ONE SET OF HARMONIC DATA
C
      READ(20,END=240) ((HARM(II,JJ),II=1,NOUT+2),JJ=1,2),ERROR,
     +                  (D(II),II=1,NITEMS)
      IREAD=IREAD+1
C
C     SKIP DATA IF NOT INSIDE 'WINDOWS'
C
  210 IF(ISEL.NE.1) GO TO 230
      CALL SELSAM(D(IWINDO(1)),D(IWINDO(2)),WINDOW,IFLAG)
      IF(IFLAG.EQ.0) GO TO 220
  230 CONTINUE
      DO 320 I=1,NHTRAN
      IF (IHCHAN(I).EQ.2*NOUT+5) THEN
        X(I)=ERROR
        GOTO 320
      END IF
      JJ=(IHCHAN(I)-1)/2 + 1
      KK=2 - MOD(IHCHAN(I),2)
C
C     PUT PHASE IN DESIRED UNITS OF DEGREES OR RADIANS
C
      IF(KK/2.EQ.FLOAT(KK)/2. .AND. JJ.GT.2 .AND. IRAD.EQ.1)
     +      HARM(JJ,KK)=HARM(JJ,KK)/57.296
      X(I)=HARM(JJ,KK)
  320 CONTINUE
      DO 330 I=NHTRAN+1,NTRAN
      X(I)=D(ISCHAN(I-NHTRAN))
  330 CONTINUE
      WRITE(10,400) (X(I),I=1,NTRAN)
  400 FORMAT(20(G10.3,1X))
C
C     RETURN FOR MORE DATA UNTIL STOP TIME
C
  220 IF(IREAD.LT.NSTOP) GO TO 200
C
 240  CONTINUE
C
C
C     CLOSE THE HARMCONT.DAT DATA FILE
C
      CLOSE(20)
      WRITE(*,*) ' '
      WRITE(*,*) 'THESE RESULTS HAVE BEEN SAVED IN FILE HTRANS.DAT'
C
      CLOSE(10)
      STOP
      END
```

```
      PROGRAM READHARMON
C
C     PROGRAM TO READ HARMON DATA FILE AND TYPE TO CRT
C     DATA ARE READ FROM FILE HARMCONT.DAT (CREATED BY HARMON.FOR)
C
C     C. HANSEN, UNIVERSITY OF UTAH, 11/86
C
      PARAMETER (NITEMS=30)
      DIMENSION HARM(12,2),SUM(NITEMS)
      CHARACTER*80 TITLE
      INTEGER YCHAN
C
      OPEN(UNIT=10,FILE='HARMCONT.DAT',STATUS='OLD',
     +    FORM='UNFORMATTED',IOSTAT=IERR)
C
C     OPEN AND READ HARMONICS INFORMATION FILE
C
      OPEN(UNIT=20,FILE='HARMINFO.DAT',STATUS='OLD')
C
      READ(20,2000) TITLE
 2000 FORMAT(A)
      READ(20,2010) NSETS,NOUT,YCHAN
 2010 FORMAT(3(1X,I10))
      WRITE(*,*) 'NUMBER OF SETS= ',NSETS
      WRITE(*,*) 'NUMBER OF HARMONIC TERMS= ',NOUT
      WRITE(*,*) 'DATA ITEM IDENTIFICATION NUMBER= ',YCHAN
C
C     READ AND WRITE ONE SET OF HARMONIC DATA
C
   10 READ(10,END=999) ((HARM(I,J),I=1,NOUT+2),J=1,2),ERROR,
     +                 (SUM(K),K=1,NITEMS)
      WRITE(*,*) 'HARMONIC DATA:'
C
      DO 300 II=1,NOUT+2
      WRITE(*,100) HARM(II,1),HARM(II,2)
  300 CONTINUE
  100 FORMAT(2(1X,G12.5))
C
      WRITE(*,150) ERROR
  150 FORMAT(' RMS ERROR (%)= ',F7.1)
C
      WRITE(*,*) ' AVERAGED DATA:'
      WRITE(*,200) (SUM(I),I=1,NITEMS)
  200 FORMAT(5(G12.4,1X))
      WRITE(*,*) 'WANT ANOTHER?'
      READ(*,*) IANS
      IF(IANS.EQ.1) GO TO 10
C
      GO TO 1000
C
  999 WRITE(*,*) 'END OF FILE ENCOUNTERED DURING READ'
C
 1000 CONTINUE
      CLOSE(10)
      END
```

```
      PROGRAM QUICKPLOT
C
C        PROGRAM TO PLOT DATA FROM THE OUTPUT FILE OF PROGRAM PREDAT
C        READS THE AZIN.DAT FILE, ALLOWS PLOTTING OF ANY CHANNEL
C        VERSUS ANOTHER OR TIME
C
C     THE CHANNEL ID'S ARE TAKEN FROM THE NAMES.DAT FILE
C
C     WRITTEN BY C HANSEN  11/86
C     NO DOCUMENTATION OF THIS PROGRAM EXISTS OTHER THAN THE LISTING.
C
           INTEGER*4 NSCAN
        PARAMETER (NITEMS=30)
        CHARACTER*50 NAME(NITEMS), XNAME, YNAME
        CHARACTER*80 TITLE
        DIMENSION ITEM(NITEMS), BW(NITEMS), D(NITEMS,1000), ICMD(1,9),
     +            BPLOT(2,1000)
C
C
C     OPEN 'AZINFO' FILE TO GET FILE INFORMATION
C
        OPEN(UNIT=30,FILE='AZINFO.DAT',STATUS='OLD',IOSTAT=IERR)
        IF(IERR.NE.0) THEN
         PRINT*,'ERROR OPENING AZINFO.DAT FILE (UNIT 30)'
         PRINT*,'IERR= ',IERR
         STOP
        END IF
        READ(30,2000) TITLE
        READ(30,*) NCHAN
        READ(30,*) NSCAN
        READ(30,*) DELT
 2000   FORMAT(A)
C
C     CLOSE AZINFO FILE
C
        CLOSE(30)
C
C     OPEN UNIT 25 (FILE AZIN) CONTAINING TIME SERIES DAT
C
        OPEN(UNIT=25,FILE='AZIN.DAT',STATUS='OLD',
     +     FORM='UNFORMATTED',IOSTAT=IERR)
        IF(IERR.NE.0) THEN
         PRINT*, 'ERROR OPENING AZIN.DAT (UNIT 25)'
         PRINT*, 'IERR= ',IERR
         STOP
        ENDIF
        REWIND 25
C
C     DETERMINE CHANNEL IDENTIFICATIONS
C
        OPEN(UNIT=11,FILE='NAMES.DAT',STATUS='OLD',IOSTAT=IERR)
        IF(IERR.NE.0) THEN
         PRINT*,'ERROR OPENING NAMES.DAT FILE (UNIT 11)'
         PRINT*,'IERR= ',IERR
         STOP
        ENDIF
        DO 5 I=1,NITEMS
        READ(11,*) NAME(I)
```

```
   5    CONTINUE
        CLOSE(11)
C
C     REWIND AZIN.DAT FILE AND READ DATA FOR PLOTTING
C
 3001   REWIND 25
C
C     GET PLOT INFORMATION
C
      WRITE(*,*) 'THIS IS THE QUICKPLOT PLOTTING PROGRAM'
      WRITE(*,*) 'YOU MAY PLOT UP TO 1000 DATA POINTS'
        WRITE(*,*) 'VERSUS TIME OR ANOTHER CHANNEL'
        WRITE(*,*) ' '
        WRITE(*,*) 'ENTER SCAN NUMBER FOR START OF PLOT'
        READ(*,*) NSTART
 1000 WRITE(*,*) 'ENTER NUMBER OF POINTS TO READ FROM DATA FILE'
      WRITE(*,*) '(UP TO 1000 POINTS CAN BE USED)'
      READ(*,*) NPASS
      IF(NPASS.GT.1000.OR.NPASS.LT.1) GO TO 1000
        WRITE(*,*) 'IF YOU WISH YOU CAN SAMPLE ONLY EVERY Nth POINT'
        WRITE(*,*) 'FROM THE DATA FILE,    ENTER N'
        READ(*,*) NINT
C
C     SKIP TO THE START TIME
C
        DO 3050 I=1,NSTART
        READ(25,END=9999) (D(IROW,1),IROW=1,NITEMS)
 3050   CONTINUE
C
        DO 3000 I=1,NITEMS
        DO 3010 J=1,NPASS
        D(I,J)=0.0
 3010   CONTINUE
 3000   CONTINUE
C
C     FILL DATA ARRAY
C
        DO 3100 I=1,NPASS
        DO 3110 J=1,NINT
        READ(25,END=3300) (D(IROW,I),IROW=1,NITEMS)
 3110   CONTINUE
 3100   CONTINUE
C
 3500 WRITE(*,*) 'ENTER ITEM NUMBER FOR INDEPENDENT'
      WRITE(*,*) 'VARIABLE (0=TIME)'
      READ*, IXCHAN
      WRITE(*,*) 'ENTER ITEM NUMBER FOR DEPENDENT VARIABLE'
      READ*, IYCHAN
      WRITE(*,*) 'ENTER NUMBER OF POINTS TO BE PLOTTED'
      READ*, NPOINT
      WRITE(*,*) 'ENTER LINE OR DATA POINT STYLE'
      WRITE(*,*) '( LINE ONLY=0; DATA POINTS ONLY (+ SYMBOL)=1 )'
      READ*, LINESTYL
      IF(LINESTYL.EQ.1)THEN
        LINESTYL=-2
      ELSE
        LINESTYL=0
      ENDIF
```

```
C
      DO 3200 I=1,NPOINT
      IF (IXCHAN.EQ.0) THEN
        BPLOT(1,I)=(I-1)*DELT*NINT + NSTART*DELT
        ELSE
        BPLOT(1,I)=D(IXCHAN,I)
        END IF
      BPLOT(2,I)=D(IYCHAN,I)
 3200 CONTINUE
C
      IF (IXCHAN.EQ.0) THEN
       XNAME='TIME (SEC)'
      ELSE
       XNAME=NAME(IXCHAN)
      END IF
C
      YNAME=NAME(IYCHAN)
 3300   CONTINUE
C
C     CALL PLOTTING ROUTINE
C
        CALL DATPLOT(BPLOT,NPOINT,1,XNAME,YNAME,TITLE,ICMD,LINESTYL)
C
        WRITE(*,*) 'ENTER 0 TO STOP THIS PROGRAM'
        WRITE(*,*) 'ENTER 1 TO PLOT OTHER CHANNELS FROM THE SAME'
        WRITE(*,*) '         DATA SET'
        WRITE(*,*) 'ENTER 2 TO READ AND PLOT A NEW DATA SET'
        READ(*,*) IANS
        IF(IANS.EQ.1)GO TO 3500
        IF(IANS.EQ.2)GO TO 3001
C
        CLOSE(25)
        STOP
C
C
 9999   PRINT*,'HIT END-OF-FILE GOING TO START POINT'
        STOP
        END
```

```
      PROGRAM DATRANS
C
C     PROGRAM TO PREPARE DATA FOR TRANSFER TO ASCII DATA FILE
C     FOR STATISTICAL ANALYSIS.  THE DATA CAN BE AVERAGED
C     FOR A SPECIFIED TIME PRIOR TO THE TRANSFER.  THIS IS
C     OFTEN ADVISABLE BECAUSE OF THE LONG TRANSFER TIME
C     FOR A LARGE DATA SET.
C
C     DATA IS READ FROM FILE 'AZIN.DAT', WITH MISCELLANEOUS
C     OTHER INFORMATION ABOUT THE DATA RECORD IN 'AZINFO.DAT'
C
C     A SCAN IS ONE 'SNAPSHOT' OF DATA OR ALL CHANNELS AT ONE INSTANT
C     TIMES ARE SPECIFIED IN IRIG DECIMAL TIME--THE NUMBER OF SECONDS
C     FROM THE START OF THE YEAR.
C
C     C HANSEN
C     WRITTEN 5/86 UNIVERSITY OF UTAH
C
      PARAMETER (NITEMS=30)
      DIMENSION D(NITEMS),X(NITEMS),ICHAN(NITEMS),TEMP(NITEMS)
      INTEGER*4 NSTOP, IREAD, NSTART, NSCAN
      CHARACTER*80 TITLE
      CHARACTER*50 NAME(NITEMS)
      DATA D/NITEMS*0./,X/NITEMS*0./,TEMP/NITEMS*0./
C
C     READ NAMES FILE
C
      OPEN(UNIT=40,FILE='NAMES.DAT',STATUS='OLD',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING NAMES.DAT FILE (UNIT 40)'
         WRITE(*,*) 'IERR= ',IERR
         STOP
      ENDIF
      DO 2000 I=1,NITEMS
      READ(40,*) NAME(I)
 2000 CONTINUE
      CLOSE(40)
C
C     OPEN FILE 'AZINFO' TO GET FILE INFORMATION
C
      OPEN(UNIT=30,FILE='AZINFO.DAT',STATUS='OLD',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING AZINFO.DAT (UNIT 30)'
         WRITE(*,*) 'IERR= ',IERR
         STOP
      ENDIF
      READ(30,3000) TITLE
      READ(30,*) NCHAN
      READ(30,*) NSCAN
      READ(30,*) DELT
 3000 FORMAT(A)
C
C     CLOSE UNIT 30---AZINFO
C
      CLOSE(30)
C
C     OPEN FILE 'AZIN' (UNIT 20) TO READ TIME SERIES DATA
C
```

```
      OPEN(UNIT=20,FILE='AZIN.DAT',STATUS='OLD',
     +     FORM='UNFORMATTED',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING AZIN.DAT (UNIT 20)'
         WRITE(*,*) 'IERR= ',IERR
         STOP
      ENDIF
      REWIND 20
C
C     OPEN FILE TRANS.DAT TO RECEIVE OUPUT DATA FOR TRANSFER
C
      OPEN(UNIT=10,FILE='TRANS.DAT',STATUS='NEW',IOSTAT=IERR)
      IF(IERR.NE.0)THEN
         WRITE(*,*) 'ERROR OPENING TRANS.DAT (UNIT 10)'
         WRITE(*,*) 'IERR= ',IERR
         STOP
      ENDIF
C
C     INPUT THE SETUP DATA FOR THIS RUN
C
C
      WRITE(*,*) 'NUMBER OF CHANNELS YOU WANT TO TRANSFER? (UP TO 14)'
      READ*, NTRANS
      DO 10 I=1,NTRANS
      WRITE(*,*) 'TRANSFER CHANNEL NUMBER',I
      WRITE(*,*) 'ENTER ITEM NUMBER FOR THIS CHANNEL'
      READ*, ICHAN(I)
   10 CONTINUE
C
      WRITE(*,*) 'WANT TO PRE-AVERAGE DATA BEFORE TRANSFER?'
      READ*, IAVG
      IF(IAVG.EQ.1) THEN
         WRITE(*,*) 'ENTER NUMBER OF SAMPLES TO AVERAGE'
         READ*, NAVG
      ELSE
         NAVG=1
      ENDIF
C
      WRITE(*,*) 'THERE ARE ',NSCAN,' SCANS IN THIS DATA FILE'
      WRITE(*,*) 'ENTER SCAN NUMBER FOR START OF ANALYSIS'
      READ*,NSTART
      WRITE(*,*) 'ENTER NUMBER OF SCANS YOU WISH TO ANALYZE'
      READ*,NSTOP
C
C     SKIP INTO THE DESIRED START POINT
C
      DO 100 I=1,NSTART
      READ(20,END=240) (D(I7),I7=1,NITEMS)
  100 CONTINUE
C
C     INITIALIZE VALUES
C
      IREAD=0
C
C     START MAIN LOOP FOR ANALYSIS
C
  200 CONTINUE
C
```

```
C     INITIALIZE FOR PREAVERAGING
C
      DO 30 I=1,NTRANS
      TEMP(I)=0.
   30 CONTINUE
      DO 40 IAVG=1,NAVG
C
C     READ ONE SCAN OF DATA
C
      READ(20,END=240) (D(I7),I7=1,NITEMS)
      IREAD=IREAD+1
C
C     SAVE ONLY THE DESIRED CHANNELS
C
      DO 20 I=1,NTRANS
      TEMP(I)=D(ICHAN(I))+TEMP(I)
   20 CONTINUE
   40 CONTINUE
C
      DO 50 I=1,NTRANS
      X(I)=TEMP(I)/FLOAT(NAVG)
   50 CONTINUE
      WRITE(10,500) (X(I),I=1,NTRANS)
  500 FORMAT(14(G10.3,1X))
C
C     RETURN FOR MORE DATA UNTIL STOP TIME
C
      IF(IREAD.LT.NSTOP) GO TO 200
C
 240  CONTINUE
C
C
C     CLOSE THE AZIN DATA FILE
C
      CLOSE(20)
C
      WRITE(*,*) ' '
      WRITE(*,*) 'THESE DATA HAVE BEEN SAVED IN FILE TRANS.DAT'
C
C
      CLOSE(10)
      STOP
      END
```

```
      PROGRAM ILSTRANS
C
C     PROGRAM TO PREPARE DATA FOR TRANSFER TO ILS PACKAGE
C     FOR STATISTICAL ANALYSIS.  THE DATA CAN BE AVERAGED
C     FOR A SPECIFIED TIME PRIOR TO THE TRANSFER.  THIS IS
C     OCCASIONALLY ADVISABLE IF ONLY LOW FREQUENCY RESPONSE
C     IS NEEDED FROM A LARGE DATA SET.
C
C     DATA IS READ FROM FILE 'AZIN.DAT', WITH MISCELLANEOUS
C     OTHER INFORMATION ABOUT THE DATA RECORD IN 'AZINFO.DAT'
C
C     DATA FILES ARE CREATED AND FILLED WITH INTEGER REPRESENTATIONS OF
THE
C     ORIGINAL DATA FILE.  ONE OUTPUT DATA FILE IS CREATED
C     FOR EACH CHANNEL OF DATA THAT IS TRANSFERRED.
C     A MAXIMUM OF NINE CHANNELS CAN BE TRANSFERRED AT ONE TIME.
C
C     ILS REQUIRES THAT SAMPLED DATA FILES BE WRITTEN AS DIRECT
C     ACCESS FILES CONTAINING INTEGER*2 VALUES, WRITTEN WITH RECL=256.
C     ONE SUCH DATA FILE IS CREATED FOR EACH CHANNEL OF DATA TRANSFERRED
C
C     C HANSEN
C     WRITTEN 10/86 UNIVERSITY OF UTAH
C
      PARAMETER (NITEMS=30)
      PARAMETER (NTRCHAN=9)
      DIMENSION D(NITEMS),ICHAN(NTRCHAN),TEMP(NTRCHAN),SCALE(NTRCHAN)
      REAL MIN(NTRCHAN), MAX(NTRCHAN)
      INTEGER*2 IDATA(NTRCHAN,256)
      INTEGER*4 NSTOP, IREAD, NSTART, NSCAN
      CHARACTER*80 TITLE
      CHARACTER*50 NAME(NITEMS)
      DATA D/NITEMS*0./,TEMP/NTRCHAN*0./
C
C     READ NAMES FILE
C
      OPEN(UNIT=40,FILE='NAMES.DAT',STATUS='OLD',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING NAMES.DAT FILE (UNIT 40)'
         WRITE(*,*) 'IERR= ',IERR
         STOP
      ENDIF
      DO 2000 I=1,NITEMS
      READ(40,*) NAME(I)
 2000 CONTINUE
      CLOSE(40)
C
C     OPEN FILE 'AZINFO' TO GET FILE INFORMATION
C
      OPEN(UNIT=30,FILE='AZINFO.DAT',STATUS='OLD',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING AZINFO.DAT (UNIT 30)'
         WRITE(*,*) 'IERR= ',IERR
         STOP
      ENDIF
      READ(30,3000) TITLE
      READ(30,*) NCHAN
      READ(30,*) NSCAN
```

```
      READ(30,*) DELT
 3000 FORMAT(A)
C
C     CLOSE UNIT 30---AZINFO
C
      CLOSE(30)
C
C     OPEN FILE 'AZIN' (UNIT 20) TO READ TIME SERIES DATA
C
      OPEN(UNIT=20,FILE='AZIN.DAT',STATUS='OLD',
     +     FORM='UNFORMATTED',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING AZIN.DAT (UNIT 20)'
         WRITE(*,*) 'IERR= ',IERR
         STOP
      ENDIF
      REWIND 20
C
C     OPEN FILE TO RECEIVE MISCELLANEOUS INFORMATION ABOUT TRANSFER
C
      OPEN(UNIT=50,FILE='ILSINFO.DAT',STATUS='NEW',IOSTAT=IERR)
      IF(IERR.NE.0) THEN
         WRITE(*,*) 'ERROR OPENING ILSINFO.DAT (UNIT 50)'
         WRITE(*,*) 'IERR= ',IERR
         STOP
      ENDIF
      WRITE(50,*) ' PARAMETERS OF ILS DATA FILE TRANSFER'
      WRITE(50,*) ' SAMPLE INTERVAL = ',DELT
      WRITE(50,*) ' SAMPLE RATE (HZ) = ',1./DELT
      WRITE(50,*) ' SCALING INFORMATION:'
      WRITE(50,*) ' FILENAME   ITEM#       SLOPE     OFFSET'
C
C     INPUT THE SETUP DATA FOR THIS RUN
C
C
      WRITE(*,*) 'NUMBER OF CHANNELS YOU WANT TO TRANSFER? (<=9)'
      READ*, NTRANS
C
      DO 10 I=1,NTRANS
      WRITE(*,*) 'TRANSFER CHANNEL NUMBER',I
      WRITE(*,*) 'ENTER ITEM NUMBER FOR THIS CHANNEL'
      READ*, ICHAN(I)
      WRITE(*,*) 'YOU HAVE SELECTED ',NAME(ICHAN(I))
      WRITE(*,*) 'ENTER THE MINIMUM VALUE OF THIS CHANNEL (FOR SCALING)'
      READ(*,*) MIN(I)
      WRITE(*,*) 'ENTER THE MAXIMUM VALUE OF THIS CHANNEL'
      READ(*,*) MAX(I)
      SCALE(I) = (MAX(I)-MIN(I))/4096.
C
C     TRUNCATE LARGE VALUES OF SCALE (ILS IDIOSYNCRASY)
C
      IF(SCALE(I).GT.10) SCALE(I)=INT(SCALE(I)+1.)
C
  10  CONTINUE
C
C     OPEN FILES TR10X. TO RECEIVE OUTPUT DATA IN ILS FORMAT
C
      DO 700 ITRANS=1,NTRANS
```

```
      OPEN(UNIT=10+ITRANS,FILE='TR10'//CHAR(ITRANS+48)//'.',
     +     STATUS='NEW',IOSTAT=IERR,ACCESS='DIRECT',RECL=128)
      IF(IERR.NE.0) THEN
        WRITE(*,*) 'ERROR OPENING TR10--',ITRANS
        WRITE(*,*) 'IERR= ',IERR
        STOP
      ENDIF
  700 CONTINUE
C
C     WRITE TRANSFER INFORMATION
C
      DO 610 I=1,NTRANS
      WRITE(50,600) I,ICHAN(I), SCALE(I), .5*(MIN(I)+MAX(I))
 600  FORMAT('  TR10',I1,6X,I3,4X,G12.5,1X,G12.5)
 610  CONTINUE
C
C
      WRITE(*,*) 'WANT TO PRE-AVERAGE DATA BEFORE TRANSFER?'
      READ*, IAVG
      IF(IAVG.EQ.1) THEN
         WRITE(*,*) 'ENTER NUMBER OF SAMPLES TO AVERAGE'
         READ*, NAVG
         WRITE(50,620) NAVG
 620     FORMAT(1X,'PREAVERAGING OF ',I4,' SAMPLES WAS USED')
      ELSE
         NAVG=1
      ENDIF
C
      WRITE(*,*) 'THERE ARE ',NSCAN,' SCANS IN THIS DATA FILE'
      WRITE(*,*) 'ENTER SCAN NUMBER FOR START OF ANALYSIS'
      READ*,NSTART
      WRITE(*,*) 'ENTER NUMBER OF SCANS YOU WISH TO ANALYZE'
      READ*,NSTOP
      WRITE(50,630) NSTOP/NAVG
 630  FORMAT(1X,'NUMBER OF AVERAGED SCANS TRANSFERRED = ',I8)
C
C     SKIP INTO THE DESIRED START POINT
C
      DO 100 I=1,NSTART
      READ(20,END=240) (D(I7),I7=1,NITEMS)
 100  CONTINUE
C
C     INITIALIZE VALUES
C
      IREAD=0
C
C     START MAIN LOOP FOR ANALYSIS
C
C
C     RECORD COUNTER FOR DIRECT ACCESS FILE
C
      IREC=1
C
 200  CONTINUE
C
C     WRITE 256 VALUES TO ONE RECORD FOR ILS
C
      DO 60 JCOUNT=1,256
```

```
C
C     INITIALIZE FOR PREAVERAGING
C
      DO 30 I=1,NTRANS
      TEMP(I)=0.
   30 CONTINUE
      DO 40 IAVG=1,NAVG
C
C     READ ONE SCAN OF DATA
C
      READ(20,END=240) (D(I7),I7=1,NITEMS)
      IREAD=IREAD+1
C
C     SAVE ONLY THE DESIRED CHANNELS
C
      DO 20 I=1,NTRANS
      TEMP(I)=D(ICHAN(I))+TEMP(I)
   20 CONTINUE
   40 CONTINUE
C
      DO 50 I=1,NTRANS
      VALUE=TEMP(I)/FLOAT(NAVG)
      IDATA(I,JCOUNT) = (VALUE-MIN(I))/SCALE(I) - 2048
   50 CONTINUE
C
C
   60 CONTINUE
C
      DO 70 ITRANS=1,NTRANS
      WRITE(10+ITRANS,REC=IREC) (IDATA(ITRANS,KK),KK=1,256)
   70 CONTINUE
C
      IREC=IREC+1
C
C     RETURN FOR MORE DATA UNTIL STOP TIME
C
      IF(IREAD.LT.NSTOP) GO TO 200
C
 240  CONTINUE
C
C
C     CLOSE THE AZIN DATA FILE
C
      CLOSE(20)
C
      WRITE(*,*) ' '
      WRITE(*,*) 'THESE DATA HAVE BEEN SAVED IN FILES TR10X.'
      WRITE(*,*) ' '
      WRITE(*,*) 'MISCELLANEOUS TRANSFER INFORMATION CAN BE'
      WRITE(*,*) 'SEEN BY TYPING FILE ILSINFO.DAT'
C
C
      CLOSE(10)
      STOP
      END
```

Document Control Page	1. SERI Report No. SERI/STR-217-3476	2. NTIS Accession No.	3. Recipient's Accession No.

4. Title and Subtitle
Yaw Dynamics of Horizontal Axis Wind Turbines: Second Annual Report

5. Publication Date
March 1989

6.

7. Author(s)
A. Hansen, C. Xudong

8. Performing Organization Rept. No.

9. Performing Organization Name and Address

Department of Mechanical Engineering
University of Utah
Salt Lake City, Utah

10. Project/Task/Work Unit No.
WE911203

11. Contract (C) or Grant (G) No.
(C) XL-6-05078-2
(G)

12. Sponsoring Organization Name and Address

Solar Energy Research Institute
1617 Cole Blvd.
Golden, Colorado 80401-3393

13. Type of Report & Period Covered
Technical Report

14.

15. Supplementary Notes
SERI Technical Monitor: A. Wright

16. Abstract (Limit: 200 words)

Numerous problems associated with yaw exist on free-yaw and yaw-driven wind turbines. Yaw-driven machines experience failures in the yaw drive mechanism due to excessive loads, while free-yaw machines experience high yaw rates and yaw misalignment problems. This report describes validation of a model for prediction of wind turbine yaw behavior. The model accounts for the coupled yaw and blade flapping motion of wind machines. The results from the model are compared to results from other analysis methods and to test data. Wind tunnel tests of a rigid-hub, two-bladed rotor, as well as full scale field test data from a rigid-hub, three-bladed turbine are used as the comparison cases in this report. The model predictions show good qualitative agreement with test data, but the magnitudes of predicted yaw moments are in error for some comparison cases. Future work will involve a more in-depth comparison of model results to comprehensive machine test data. Refinements will be made to the analytical model in order to improve correlations between theory and experiment.

17. Document Analysis

a. Descriptors
Wind energy ; wind turbine ; computer code ; yaw dynamics ; yaw

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, Virginia 22161

19. No. of Pages
202

20. Price
A10