SANDIA REPORT
SAND88—2306 • UC—261
Unlimited Release
Printed September 1988

A Summary of 1988 Summer Student Research Projects in Wind Energy

Jesse D. Sims, Jocelyn R. Wright

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550
for the United States Department of Energy
under Contract DE-AC04-76DP00789

SF2900Q(8-81)

When printing a copy of any digitized SAND Report, you are required to update the markings to current standards.

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

NOTICE: This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government, any agency thereof or any of their contractors or subcontractors. The views and opinions expressed herein do not necessarily state or reflect those of the United States Government, any agency thereof or any of their contractors.

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

NTIS price codes
Printed copy: A04
Microfiche copy: A01

Distribution
Category UC-261

SAND88-2306
Unlimited Release
Printed September 1988

A SUMMARY OF 1988 SUMMER STUDENT RESEARCH PROJECTS IN WIND ENERGY*

by

Jesse D. Sims

and

Jocelyn R. Wright

Wind Energy Research Division
Sandia National Laboratories
Albuquerque, New Mexico 87185

ABSTRACT

During the summer of 1988, the Wind Energy Research Division hosted Jesse Sims and Jocelyn Wright under the auspices of the Sandia Historically Black Colleges and Universities (HBCU) Summer Student Program. The research projects that they conducted during their tenure at Sandia are reported here. In the first chapter of this report, the development of a software package to determine the calibration constants for a group of aerodynamic sensors is detailed. In the second chapter, the refurburishment activities on the Sandia 5-m VAWT are discussed.

---

*This work was supported by the U. S. Department of Energy at Sandia National Laboratories under contract DE-AC04-76DP00789.

FOREWORD

During the summer of 1988, the Wind Energy Research Division hosted Jesse Sims and Jocelyn Wright under the auspices of the Sandia Historically Black Colleges and Universities (HBCU) Summer Student Program. The research projects that they conducted during their tenure at Sandia are reported here. As their projects did not relate directly to one another, this report is divided into two chapters, one written by each of the two researchers.

The first chapter was written by Jesse Sims. Jesse received a Bachelor of Arts degree in Physics from Texas Southern University in May, 1988. He will be attending graduate school at the University of Houston this fall. His chapter in this report details the development of a software package to determine the calibration constants for a group of aerodynamic sensors.

Jocelyn Wright wrote the second chapter. Jocelyn is a enrolled in a "dual degree" program at Xavier University of Louisana and Southern University. She is currently a senior Mechanical Engineering student at Southern University. This chapter details her refurburishment activities on the Sandia 5-m VAWT. In the appendix to this chapter, Jocelyn discusses her activities in Sandia's material laboratories. This discussion centers on the examination of the fracture surface of an aluminum coupon that had been tested in fatigue bending at Southern University.

Herbert J. Sutherland, Editor
August 1988

CHAPTER 1

SOLUTION ALGORITHM AND SOFTWARE PROGRAM TO AID IN THE CALIBRATION OF SHEAR STRESS SENSORS

by

Jesse D. Sims

ABSTRACT

The objective of this project was to develop a data reduction code, in Fortran, to aid in the calibration of sensors that measure shear stress or drag between the air and the rotors of a wind turbine. Previous work has shown that the data are fit best with a curve of the form $y = A\,x^{1/n} + B$, where A, B and n are constants. These constants must be chosen such that the "correlation coefficient" between the data and the curve fit is maximized. A solution algorithm was formulated and programmed in Fortran 77 for an IBM PC-compatible computer. The program, called SENSOR, permits the data to be plotted directly from the program. The solution technique was tested using three sets of data, in two of which the correlation coefficient and the constants were known. The results obtained from the test data showed that the program calculated the correct fit for all cases.

INTRODUCTION

The objective of this project was to develop a data reduction code to aid in the calibration of sensors that measure shear stress or drag between the air and the rotors on a wind turbine. Previous work has shown that the data are fit best with a curve of the form:

$$y = A\,x^{1/n} + B \quad , \tag{1.1}$$

where A, B and n are constants. These constants must be chosen so that the "correlation coefficient" (C) between the data and the curve fit is maximized. The correlation coefficient used here is given by the relationship

$$C = 1 - \frac{\sqrt{\sum_{j=1}^{N} \left[A\,(x_j)^{1/n} + B - y_j \right]^2}}{N} \quad , \tag{1.2}$$

where N is the number of data points, j is the jth point in the data set, x_j is the wall shear stress and y_j is the output voltage.

The solution algorithm was formulated to calculate C based on a least squares curve fit to a straight line. The program was written in Fortran 77 for an IBM PC-compatible computer. The program, called SENSOR, permits the data to be plotted directly from the program.

The program takes a set of data, sensor voltage vs. wall shear stress, and maximizes the correlation coefficient by choosing the "best" values for the curve fit parameters. The SENSOR program also incorporates two plot subroutines using PLOT88.[1.1] The first subroutine plots C vs. n allowing the user to determine if C has a maximum. The second subroutine plots the wall shear stress raised to the 1/n power vs. the sensor signal, and then plots the calculated best fit through data set.

The report starts with a review of the solution algorithm used in the program. Then the input parameters used in the program are described in detail. Three example problems close the report. Listed in the appendices are the variables used in the program, a copy of the final version of the program, and example outputs.

Solution Algorithm

The solution algorithm for this problem must determine values for A, B and n based upon the constraint that the value of C in Eq. (1.2) is maximized. An iterative technique on the parameter n was chosen. In the technique, the current value of n is updated after each iteration based on an adjustable step size and the slope dC/dn. Convergence of the iterations to a solution is judged by the closeness of succeeding iterations.

Current Values

Let the value of n in the ith step be given by n_i. Assuming that n_i is a constant for this iteration, then the curve fit may be written as

$$y = A\, x_1 + B \,, \tag{1.3}$$

where

$$x_1 = \left[x\right]^{1/n_i} \,. \tag{1.4}$$

The values for A and B may now be found by using a least squares curve-fitting technique to fit a straight line to the order pairs of data points defined by $[(x_1)_j, y_j]_{j=1,N}$ where

$$(x_1)_j = \left[x_j\right]^{1/n_i} \,. \tag{1.5}$$

The least squares curve-fitting algorithm used here was developed by Brain P. Flannery, et al.[1.2]

Substituting these values into Eq. (1.2) yields a current value for C, namely $C(n_i)$, of

$$C(n_i) = 1 - \frac{\sqrt{\sum_{j=1}^{N} \left[A\,(x_j)^{1/n_i} + B - y_j \right]^2}}{N} \,. \tag{1.6}$$

The derivative or slope of this function may be determined numerically by

$$\frac{d}{dn}\left[C(n_i)\right] = \frac{C(n_i + \hbar) - C(n_i - \hbar)}{2\,\hbar} \quad , \tag{1.7}$$

where $\hbar$ is an arbitrarily small increment.

Next Value

To update the value of n_i to n_{i+1}, define Δn_i to be

$$n_{i+1} = n_i + \Delta n_i \quad . \tag{1.8}$$

Here, we chose the value and sign of Δn_i based on an adjustable increment ε_i and the derivative defined by Eq. (1.7); namely,

$$\Delta n_i = \min \left\{ \begin{array}{l} \varepsilon_i \ \mathrm{sign}\left(\dfrac{d}{dn}\left[C(n_i) \right] \right) \\ \\ \varepsilon_i \left(\dfrac{d}{dn}\left[C(n_i) \right] \right) \end{array} \right\} \quad , \tag{1.9}$$

where min(·) is the minimum function of the arguments and sign(·) is the sign function of the argument. This formulation uses the property of a maxima so that the value of n_i should be increased if the sign of its derivative $d[C(n)]/dn$ is positive and should be decreased if the sign is negative. If the derivative $d[C(n)]/dn$ is a smooth function, then the second term in Eq. (1.8) will speed the convergence of n_i to the solution. The value of ε_i is given by

$$\varepsilon_i = \varepsilon_{i-1} \quad , \tag{1.10}$$

if the $\mathrm{sign}\{d[C(n_i)]/dn\} = \mathrm{sign}\{d[C(n_{i-1})]/dn\}$ or by

$$\varepsilon_i = \frac{\varepsilon_{i-1}}{2} \tag{1.11}$$

if the sign{d[C(n_i)]/dn} $\neq$ sign{d[C(n_{i-1})]/dn}. This technique for adjusting the step size ε_i holds the value of the step constant until the solution steps across the maxima of C(n) and then reduces the step by half.

Solution

The iteration is said to have found the maxima if

$$C(n_i) = 1 \quad , \tag{1.12}$$

or

$$C(n_i) - C(n_{i-1}) \leq \delta \quad . \tag{1.13}$$

In the first case, the iteration scheme has found an exact fit to the data, and in the second, the solutions have converged within a specified tolerance, δ.

FORTRAN PROGRAM

The solution algorithm described above was written in Fortran 77. It was compiled for an IBM-compatible PC using a Microsoft Fortran Compiler, Ver. 4.1.[1,3] The code is listed in Appendix 1.A, and a list of variables are given in Appendix 1.B.

The SENSOR program begins by asking the user if the data points are in a file. If the answer is yes, then it asks for the file name. (All the questions in the program that require a yes or no answer may be answered with an upper or lower case "y" or "n".) The file name must be input with the file extension. For example, if the file is a data file called exact, then the user would input *exact.dat*. The program then counts the number of data points and prints the actual data from the user's file. The format for the data points in the file must be of the form in which the x variables are in one column and the y variables are in a column parallel to the x column. The x variable corresponding to the y variable should appear on the same line.

If the data points are in a file, they are printed on the monitor, and the program asks the user if the data points the program read are the correct set. The user responds by answering yes or no. If the answer is yes the program proceeds to the next step. If the answer is no, the program asks the user to input the file name once again. This allows a user to search for a data set if unsure of the file name.

If the data points are not recorded in a file, then the user may input them from the keyboard. First the program asks the user for the number of input data points. The user inputs the x variable first, and then on the same line inputs the y variable, separated by at least one space. After inputing both variables the user hits *enter*. The process is repeated for each pair of data points. The next question prompts the user to

check the input data for accuracy. This procedure not only provides the user with a check, but also provides a way of changing the input to the program. Also the user is able to change the number of input points.

The next question is whether the user wants to use the default parameter set. A listing of the default parameters is contained in the program. If the user does not want to use the default parameters then he is able to input them from the keyboard. Each default parameter has an input statement. For example, for the statement *input tolerance*, the user would input the desired tolerance and hit *enter*, and then repeat the process for the next input statement, hitting *enter* after each entry.

After the parameters have been input, the program calls the FIT subroutine. In the FIT subroutine slope A, intercept B, and correlation coefficient C are calculated for the current value of n. These numbers are passed back to a loop and the value of n is updated. The process is repeated until the program determines that it has found the answer. Then the best n and maximum C, with the corresponding slope and intercept, are printed on the screen.

Next the program asks the user if he wants to plot C vs. n. The user can either have the plot sent to the screen for viewing or to the printer for a hard copy by entering either "s" or "n". This option sets the ioport and the model number variables for the plot package to their proper values. When the ioport and model numbers both equal 97, the hard copy is sent to the screen. And if the ioport and mode numbers are 0 and 1 respectively the plot is sent to the printer.

If the user wants a plot of C vs. n then the first plot subroutine, GRAPH, is called. This subroutine utilizes PLOT88 to draw the axes, the labels, and the title and to plot the actual curve. PLOT88 is a software library that utilizes different call statements. Note: PLOT88 has to be a part of the user's compiler in order for a user to implement it in an uncompiled version of the program. The user must set the parameters within the call statement. For example, the statement *call plots(0,0,1)* sets the ioport, which is the second number in the call statement, at zero. Also the third member in the call statement, the model number, is set at one. Note: The first number is always zero and is included for compatibility.

The next question prompted by the computer is whether the user wants a plot of x vs. y. Again, the user has the choice of having the plot sent to the screen or to the printer. Then a second plot subroutine, GRAPH2, is called. And again, PLOT88 is used to draw the axes, the labels, and the title, but the plot will be a straight line instead of a curve. The same parameters used in the previous plot subroutine are implemented in this subroutine.

The program also gives the user the option of printing the output to a file. The user responds by entering yes or no. If the answer is yes, the data is printed to a file called *output.dat*.

The final question asked by the program is whether the user wants to run the program again. With this option, the user can restart the program without having to leave and

re-enter the program externally. The user can also replot the same data set or plot a different data set using this option.

EXAMPLE PROBLEMS

Data Without Scatter

The initial check of the program was conducted with a known data set. The coefficients A, B and n, see Eq. (1.1), were chosen to be 2.0, 1.0 and 2.5, respectively. A total of seven data points were calculated using these values. The points are summarized in Appendix 1.C.

The seven points were then input into the SENSOR program, and a curve fit was obtained. The initial value of n, tolerance δ, increment $\hbar$, and step size ε were chosen to be 3.0, .00001, .001 and 1.0 respectively. These are the default values chosen for the program.

The solution is also shown in Appendix 1.C. As seen there, the fit chosen by the solution algorithm is identical to the expected values, within numerical round-off. The plots generated by the program are shown in Figs. 1.1 and 1.2. Figure 1.1 is a plot of correlation coefficients vs. n, and it shows a maximum in the correlation coefficient curve. Figure 1.2 is a plot of wall shear stress vs. sensor signal, and it shows the straight-line relationship between the two variables.

Data with Scatter

In this example we changed the third and fifth data points of the first example by adding and subtracting 0.1 to the y value or sensor signal of the data set.

The nine points were then input into the SENSOR program, and a curve fit was obtained. The default set of solution parameters was used for this solution. The initial values were chosen to be the same as those chosen in the first example, i.e., their default values.

The output is shown in Appendix 1.D. Again, as seen there, the fit used by the solution algorithm is identical to the expected values, within numerical round-off. The plots generated by the program are shown in Figs. 1.3 and 1.4.

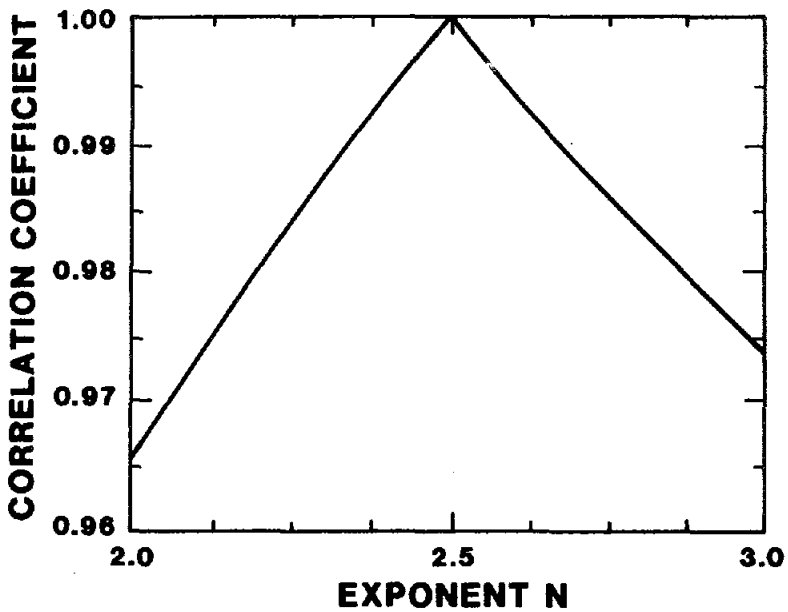

Figure 1.1. Correlation Coefficient for the Data Without Scatter Example.

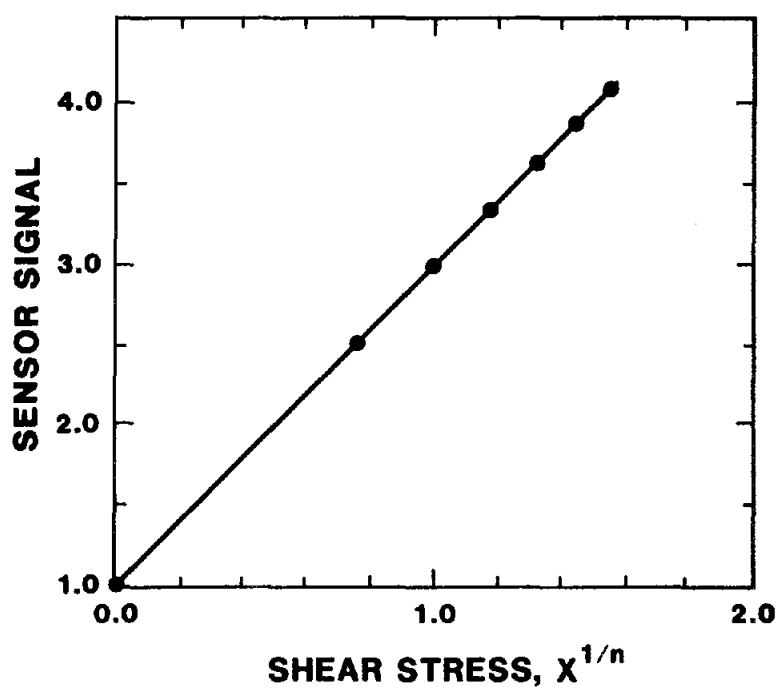

Figure 1.2. Comparison of Input Data to Curve Fit for the No-Scatter Data Fit Example.

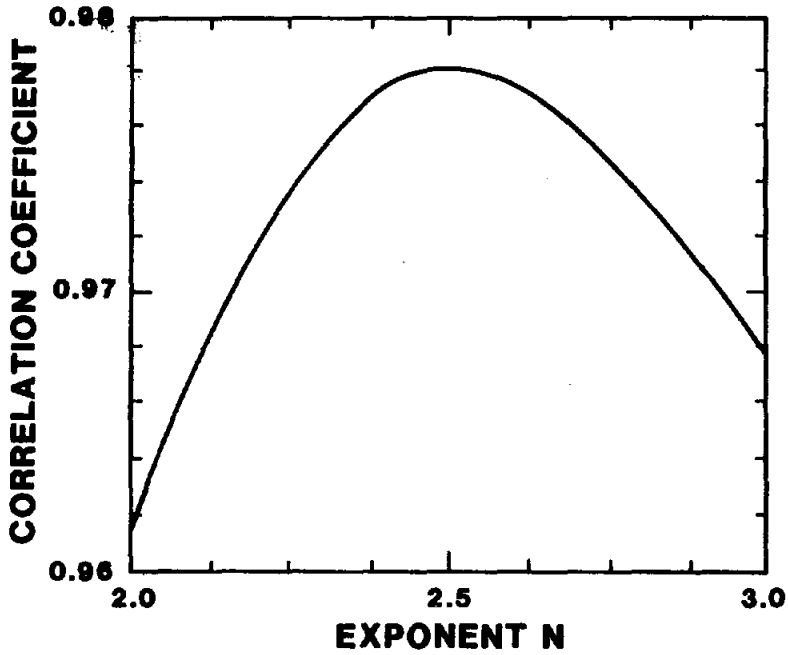

Figure 1.3. Correlation Coefficient for the Data With Scatter Example.

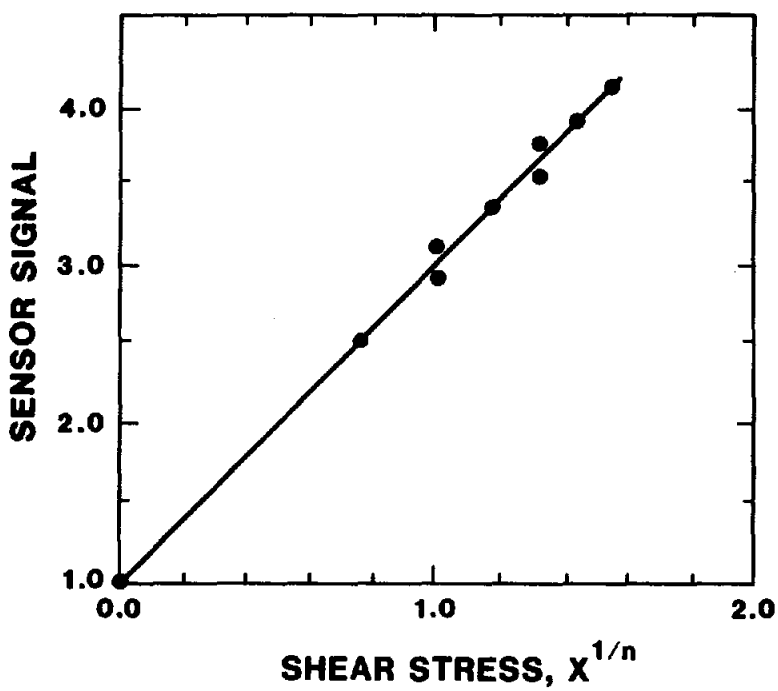

Figure 1.4. Comparison of Input Data to Curve Fit for the Scatter Data Fit Example.

Typical Laboratory Data

The data for this example are given by J. Mathews.[1,4]

In this example, the coefficients A, B, and n were not known. There was a total of eleven data points, which are summarized in Appendix 1.E.

The eleven points were input into the SENSOR program. The default parameter set was also used for this solution. The data generated by the program included slope A, intercept B, n and the correlation coefficient.

The output is shown in Appendix 1.E, and Figs. 1.5 and 1.6. For these data, the expected range for n was between 3 and 4. The n chosen by the solution algorithm was 3.6545, which is within this expected range. The graph in Fig. 1.5 shows that there is a solution for these data, and Fig. 1.6 shows the straight-line relationship between sensor signal and wall shear stress. The conclusion was that the solution chosen by the algorithm was accurate within numerical round-off.

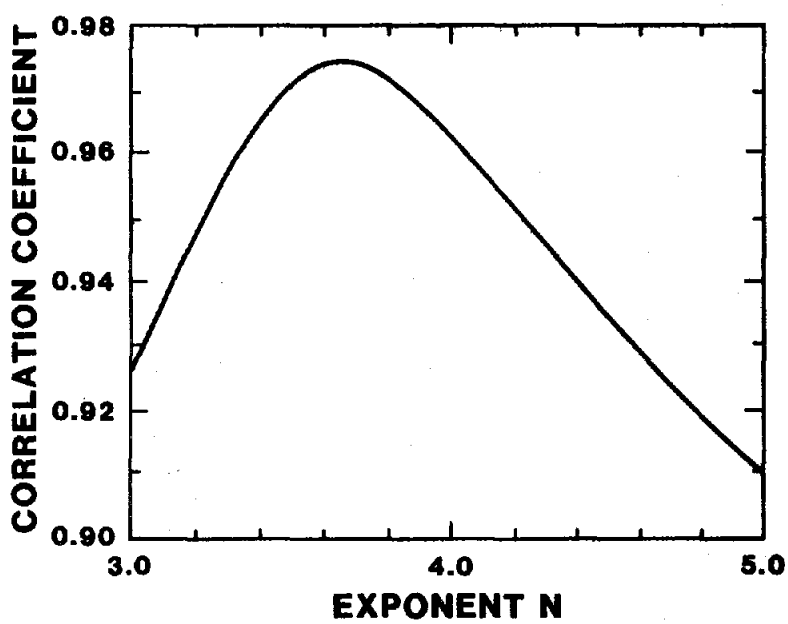

Figure 1.5. Correlation Coefficient for the Typical Laboratory Data.

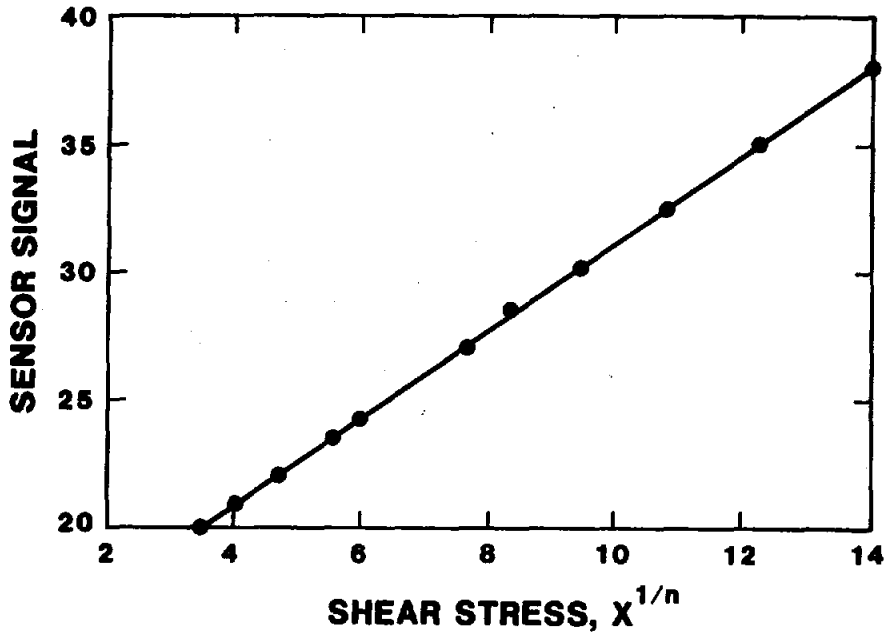

Figure 1.6. Comparison of Input Data to Curve Fit for the Typical Lab Data.

CONCLUSIONS

This report presents a detailed description of the SENSOR program. The SENSOR program is a data reduction code to aid in the calibration of sensors that measure shear stress or drag between the air and the rotors on a wind turbine. The report starts with an introduction, proceeds to a solution algorithm and concludes with example problems. The solution algorithm utilizes an iterative and least square fit technique to calculate the desired output. The program was written in Fortran 77 using an IBM-compatible PC. With the data obtained from the program, shear stress sensors may be calibrated.

REFERENCES

1.1. T.L. Young and M.L. Van Woert, Plot88 Software Library Reference Manual, (La Jolla: Plotworks Inc., 1987).

1.2. B.P. Flannery, et al, Numerical Recipes The Art of Scientific Computing, (Cambridge: Cambridge University Press, 1986), pp. 508-509.

1.3. Microsoft FORTRAN Optimizing Compiler (Microsoft Corporation, 1987).

1.4. J. Mathews, The Theory and Application of Heated Films for the Measurement of Skin Friction, (Cranfield Institute of Technology, College of Aeronautics: Sept., 1985).

APPENDIX 1.A

PROGRAM LISTING

```
c                    +++++SENSOR.FOR+++++
c
c.....The purpose of the following program is to calculate slope A,
c     intercept B, correlation coefficient(COEF) and the best n,
c     where 1/n is the the exponent for the surface shear stress.
c     The program also plots COEF versus n and signal y versus shear stress
x
c     using a plot library called PLOT88.
c
c
c     Written at Sandia National Laboratories
c     by Jesse Sims
c
c
c*********************************************************************
c
c
c     PROGRAM SENSOR
c
c*********************************************************************
      character answer*1,filename*10,bell*2
      common x(100),y(100),xl(100),npoint,xplot(102),yplot(102)
      real n,nstep,lowern,uppern
10    write(6,1020)
      write(6,2000)
      read(5,4000) answer
c
c     Allows the user to input data from a file
c
      if(answer.eq.'Y'.or.answer.eq.'y')then
         write(6,1020)
         write(6,2010)
         read(5,4010) filename
         goto 60
c
c     Allows the user to input data from the keyboard
c
      elseif(answer.eq.'N'.or.answer.eq.'n')then
         write(6,1020)
         write(6,3040)
         read(5,4020)npoint
         write(6,1020)
         write(6,2030)
```

```
         do 20 i=1,npoint
            read(5,*)x(i),y(i)
20       continue
         goto 30
      else
         bell=char(7)
         write(6,3010) bell
         write(6,1020)
         goto 10
      endif
c
c.....Allows the user to check entered data set
30    write(6,1020)
      write(6,1000)
         do 40 i=1,npoint
            write(6,1010)x(i),y(i)
40       continue
      write(6,3050)
      read(5,4000) answer
      if(answer.eq.'Y'.or.answer.eq.'y')then
         goto 140
      elseif(answer.eq.'N'.or.answer.eq.'n')then
         write(6,1020)
         write(6,3040)
         read(5,4020)npoint
         write(6,1020)
         write(6,3070)
         do 50 i=1,npoint
            read(5,*)x(i),y(i)
50       continue
         goto 30
      else
         bell=char(7)
         write(6,3010) bell
         write(6,1020)
         goto 30
      endif
c
c.....Open text file
c
60    open(unit=1,file=filename,status='unknown')
      npoint=0
c
c.....Read data in text file
      do 70 i=1,100
         read(1,*,end=80) x(i),y(i)
         npoint=npoint+1
70    continue
c
```

```
c.....Write data to the screen
80    write(6,1020)
      write(6,1000)
      do 90 i=1,npoint
         write(6,1010)x(i),y(i)
90    continue
c
c.....Close text file
      close(1)
c
c.....Allows the user to change the file within the program
c
100   write(6,3020)
      read(5,4000) answer
      if(answer.eq.'Y'.or.answer.eq.'y')then
         goto 140
      elseif(answer.eq.'N'.or.answer.eq.'n')then
         write(6,1020)
         write(6,3030)
         read(5,4010) filename
         open(unit=2,file=filename,status='unknown')
         npoint=0
         do 110 i=1,100
           read(2,*,end=120) x(i),y(i)
           npoint=npoint+1
110      continue
120      write(6,1020)
         write(6,1000)
         do 130 i=1,npoint
           write(6,1010)x(i),y(i)
130      continue
         close(2)
         goto 100
      else
         bell=char(7)
         write(6,3010) bell
         write(6,1020)
         goto 80
      endif
c
c.....Want to use the default parameters
c
140   write(6,1020)
      write(6,2020)
      read(5,4000) answer
      if(answer.eq.'Y'.or.answer.eq.'y')then
           deln = 1.0
            tol = 0.00001
              h = 0.001
```

```
            n = 3.0
         goto 160
      elseif(answer.eq.'N'.or.answer.eq.'n')then
         goto 150
      else
         bell=char(7)
         write(6,3010) bell
         write(6,1020)
         goto 140
      endif
c
c.....input delta n or step size
150   write(6,1020)
      write(6,1030)
      read(5,*)deln
c
c.....input h
      write(6,1020)
      write(6,1040)
      read(5,*)h
c
c.....input tolerance
      write(6,1020)
      write(6,1050)
      read(5,*)tol
c
c.....input n
      write(6,1020)
      write(6,1060)
      read(5,*)n
c
c.....calculation of best n
c.....Calling the FIT subroutine
160   oldslp=0
      write(6,1020)
      write(6,2080)
170   call FIT(n,A,B,coef)
      if(coef.ge.1-tol.and.coef.le.1+tol) goto 180
      call FIT(n+h,a0,b0,coef0)
      call FIT(n-h,a1,b1,coef1)
      slope=(coef0-coef1)/(2*h)
      if(slope.ge.-tol.and.slope.le.tol) goto 180
      if(oldslp/slope.lt.0) then deln=deln/2
      oldslp=slope
      delnl=abs(deln*slope)
      nstep=min(deln,delnl)
      n=n+sign(nstep,slope)
      goto 170
c
```

```
c.....Calculation of best n
c.....Gives a better value of n and COEF
180   deln=.01
      tol=.0000001
190   call FIT(n,A,B,coef)
      if(coef.ge.1-tol.and.coef.le.1+tol) goto 200
      call FIT(n+h,a0,b0,coef0)
      call FIT(n-h,a1,b1,coef1)
      slope=(coef0-coef1)/(2*h)
      if(slope.ge.-tol.and.slope.le.tol) goto 200
      if(oldslp/slope.lt.0) then deln=deln/2
      oldslp=slope
      deln1=abs(deln*slope)
      nstep=min(deln,deln1)
      n=n+sign(nstep,slope)
      goto 190
200   write(6,1020)
      write(6,2090)
      write(6,1090)A,B,coef,n
c
c.....Want a hard copy of output?
210    write(6,1080)
       read(5,4000)answer
       if(answer.eq.'Y'.or.answer.eq.'y')then
         goto 220
       elseif(answer.eq.'N'.or.answer.eq.'n')then
         goto 240
      else
         bell=char(7)
         write(6,3010) bell
         write(6,1020)
         goto 200
       endif
c
c.....Opens new text file to store the output
c
220    open(unit=3,file='output.dat',status='unknown')
       write(3,1000)
       do 230 i=1,npoint
         write(3,1010)x(i),y(i)
230    continue
      write(3,2090)
      write(3,1090)A,B,coef,n
c
c.....Close text file
      close(3)
c
c.....Want a plot of COEF vs n ?
240   write(6,1020)
```

```
      write(6,2040)
      read(5,4000) answer
      if(answer.eq.'Y'.or.answer.eq.'y')then
         write(6,1020)
         write(6,2050)
c
c.....Inputing of upper bound and lower bound
         read(5,*)lowern
         write(6,1020)
         write(6,2060)
         read(5,*)uppern
         step=(uppern-lowern)/99
         i=0
         do 250 n=lowern,uppern,step
            call FIT(n,A,B,coef)
            i=i+1
            xplot(i)=n
            yplot(i)=coef
250      continue
      nplot=(uppern-lowern)/step
c
c.....Calling coef vs n plot routine
c
         call GRAPH(nplot)
         goto 260
      elseif(answer.eq.'N'.or.answer.eq.'n')then
         write(6,1020)
         goto 260
      else
         bell=char(7)
         write(6,3010) bell
         write(6,1020)
         goto 240
      endif
c
c.....Want a plot of x vs y ?
c
260   write(6,1020)
      write(6,2070)
      read(5,4000) answer
      if(answer.eq.'Y'.or.answer.eq.'y') then
c
c.....Recalculation of best n
c.....Calling the FIT subroutine
c
      oldslp=0
      write(6,1020)
      write(6,3000)
      deln=1.0
```

```
      tol=.00001
      n=3.0
270   call FIT(n,A,B,coef)
      if(coef.ge.1-tol.and.coef.le.1+tol) goto 280
      call FIT(n+h,a0,b0,coef0)
      call FIT(n-h,a1,b1,coef1)
      slope=(coef0-coef1)/(2*h)
      if(slope.ge.-tol.and.slope.le.tol) goto 280
      if(oldslp/slope.lt.0) then deln=deln/2
      oldslp=slope
      deln1=abs(deln*slope)
      nstep=min(deln,deln1)
      n=n+sign(nstep,slope)
      goto 270
c
c.....Calling y vs x1 plot routine
c
280      xplot(2)=x1(npoint)
         xplot(1)=x1(1)
         yplot(1)=a*x1(1)+b
         yplot(2)=a*x1(npoint)+b
         call GRAPH2
         goto 290
      elseif(answer.eq.'N'.or.answer.eq.'n')then
         goto 290
      else
         bell=char(7)
         write(6,3010) bell
         write(6,1020)
         goto 260
      endif
c
c.....Want to run the program again ?
c
290   write(6,1020)
      write(6,1070)
      read(5,4000) answer
      if(answer.eq.'Y'.or.answer.eq.'y')then
      write(6,1020)
         goto 10
      elseif(answer.eq.'N'.or.answer.eq.'n')then
         goto 300
      else
         bell=char(7)
         write(6,3010) bell
         write(6,1020)
         goto 290
      endif
c
```

```
c
1000   format(/26x'shear stress',6x'sensor signal'/)
1010  format(18x,f18.4,9x,f10.4)
1020  format(//////////////////////)
1030  format(/23x,'input delta n>'//////////)
1040  format(/23x,'input h>'/////////)
1050  format(/23x,'input tolerance>'//////////)
1060  format(/23x,'input n>'/////////)
1070  format(/23x,'Do you want to run the program again ?'
     &       /23x,'Enter Y or N and hit return.>'/////////)
1080  format(/23x,'Do you want to send this output to a file in order'
     &       /23x,'to get a hard copy. Enter Y or N and hit return.>',
     &       /23x,/////)
1090  format(/23x,'Slope:            A = ',f8.4,
     &       /23x,'Intercept:        B = ',f8.4,
     &       /23x,'Coefficient:   COEF = ',f8.4,
     &       /23x,'Best n            n = ',f8.4,/)
2000  format(/23x,'Are your data points in a file ?'
     &       /23x,'Enter Y or N and hit return.>'//////////)
2010  format(/23x,'What is the filename ?>'/////////)
2020  format(/23x,'delta n = 1.000',
     &       /23x,'    tol = 0.00001',
     &       /23x,'      h = 0.001',
     &       /23x,'      n = 3.000'//,
     &       /23x,'Do you want to use these default perimeters?',
     &       /23x,'Enter Y or N and hit return.>'////////)
2030  format(/18x,'Input your data.  Note: The user should input the',
     &       /18x,'x coordinate first, followed by a space and then',
     &       /18x,'the y coordinate. After each x and y input hit enter.',
     &       /18x,'Repeat the procedure for each data point.'/////////)
2040  format(/23x,'Do you want to plot COEF vs n ?',
     &       /23x,'Enter Y or N and hit return.>'//////////)
2050  format(/28x,'Input lower limit.>'///////////)
2060  format(/28x,'Input upper limit.>'///////////)
2070  format(/23x,'Do you want to plot signal (y) vs shear ',
     &       /23x,'stress (x).  Enter Y or N and hit return.>',
     &            ////////)
2080  format(/23x,'CALCULATION OF BEST FIT IN PROGRESS'//////////)
2090  format(/23x,'This the computer generated',
     &       /23x,'best fit of your data.'/)
3000  format(/28x,'PLEASE STAND BY'//////////////)
3010  format(/23x,'INVALID ENTRY',a/////////)
3020  format(/23x,'Are these the data points in your file ?',
     &       /23x,'Enter Y or N and hit return.>'/////)
3030  format(/23x,'Enter your file name again.>'/////////)
3040  format(/23x,'Enter number of data inputs.>'//////////)
3050  format(/23x,'Is this the data set that you input?>'/////)
3060  format(/23x,'Enter data set again.>'//////////)
3070  format(/23x,'Input your data again.>'/////////)
```

```
4000  format(a)
4010  format(10a)
4020  format(i4)
c
c
300    end
c
c
c
c*********************************************************************
c
c                              ++++++FIT++++++
c
c.....This subroutine FIT allows the user to calculate A and B
c     for a given x,y and n
c
      subroutine FIT(n,A,B,coef)
c*********************************************************************
c
c
      character answer*1,bell*1
      common x(100),y(100),x1(100),npoint,xplot(102),yplot(102)
      real n
      bell=char(7)
      sumx1=0
      sumy=0
      st2=0
      A=0
      ss=0
c*********************************************************************
c
c.....Calculation of the x1 value
      do 10 i=1,npoint
          x1(i)=x(i)**(1.0/n)
10    continue
      do 20 i=1,npoint
         sumx1=sumx1+x1(i)
         sumy=sumy+y(i)
20    continue
      ss=npoint
      sxoss=sumx1/ss
      do 30 i=1,npoint
         t=x1(i)-sxoss
         st2=st2+t*t
         A=A+t*y(i)
30    continue
      A=A/st2
      B=(sumy-sumx1*a)/ss
      sum=0
```

```
      do 40 i=1,npoint
         sum=sum+(A*x1(i)+B-y(i))**2
40    continue
      points=npoint
      coef=1-(sqrt(sum)/points)
c
c
c
1005  format(//////////////////////)
2000  format(a)
c
      return
      end
c
c
c
c*********************************************************************
c
c                              ++++++GRAPH++++++
c
c.....This subroutine GRAPH allows the user to plot coef versus n for
c     a given x,y,A and B.
c
      subroutine GRAPH(nplot)
c*********************************************************************
c
c
      character tit2*36,xaxt*3,yaxt*6
      character bell*2
      common x(100),y(100),x1(100),npoint,xplot(102),yplot(102)
      ioport=97
      model=97
      xlt=7
      ylt=4
c
c*********************************************************************
c
c
c.....Initialize the plot routine
c
10    write(6,1010)
      write(6,1020)
      read(5,3000) answer
      if(answer.eq.'P'.or.answer.eq.'p')then
         xzr=1.5
         yzr=1.5
         xt2=xzr -1.15
         call plots(0,0,1)
         write(6,1010)
```

```
         write(6,1030)
         goto 20
      elseif(answer.eq.'S'.or.answer.eq.'s')then
         xzr=1
         yzr=1
         xt2=xzr -.65
         call plots(0,ioport,model)
         goto 20
      else
         bell=char(7)
         write(6,3010) bell
         write(6,1020)
         goto 10
      endif
c
20    call color(2,ierr)
c
c
c.....Draw the outline for the plot
      call plot(xzr,yzr+ylt,3)
      call plot(xzr+xlt,yzr+ylt,2)
      call plot(xzr+xlt,yzr,2)
      call plot(xzr,yzr,-3)
c
c.....Scale y-axis and x-axis
c
      call scale(yplot,ylt,nplot,1)
      call scale(xplot,xlt,nplot,1)
c
c.....Writes the title of the plot
c
      nt2=32
      yt2=yzr+ylt+.2
      tit2=' CORRELATION COEFFICIENT vs N '
      call symbol(xt2,yt2,0.24,tit2,0.0,nt2)
c
c.....Draw the axes
c
      nx=3
      ny=6
      xaxt=' N '
      yaxt=' COEF '
      xfst = xplot(nplot+1)
      xdel = xplot(nplot+2)
      yfst = yplot(nplot+1)
      ydel = yplot(nplot+2)
      call staxis(0.15,0.17,0.1111,0.1111,3)
      call axis (0.0,0.0,yaxt,ny,-ylt,90.0,yfst,ydel)
      call axis (0.0,0.0,xaxt,-nx,-xlt,0.0,xfst,xdel)
```

```
c
c.....Plot the curve
c
      call line(xplot,yplot,nplot,1,0,0)
c
c.....Output the drawing to the screen
c
      call plot(0.,0.,999)
c
c
1000  format(///////)
1010  format(//////////////////////)
1020  format(/20x,'Do you want the plot sent to the Printer (P)',
     &       /20x,'or Screen (S).  Enter P or S and hit return.',
     &       ////////)
1030  format(/28x,'PLOT IN PROCESS'//////////////)
3000  format(a)
3010  format(/2x,'Invalid entry',a)
      return
      end

c
c
c
c**********************************************************************
c
c                              ++++++GRAPH2++++++
c
c.....This subroutine GRAPH2 allows the user to plot sensor signal y
c     vs shear stress x.
c
c
      subroutine GRAPH2
c**********************************************************************
c
c
      character tit2*36,xaxt*3,yaxt*6
      character bell*2
      common x(100),y(100),xl(100),npoint,xplot(102),yplot(102)
      ioport=97
      model=97
      xlt=7
      ylt=4
c
c**********************************************************************
c
c.....Initialize the plot routine
c
10    write(6,1010)
```

```
      write(6,1020)
      read(5,3000) answer
      if(answer.eq.'P'.or.answer.eq.'p')then
         xzr=1.5
         yzr=1.5
         xt2=xzr-.95
         call plots(0,0,1)
         write(6,1010)
         write(6,1030)
         goto 20
      elseif(answer.eq.'S'.or.answer.eq.'s')then
         xzr=1
         yzr=1
         xt2=xzr-.45
         call plots(0,ioport,model)
         goto 20
      else
         bell=char(7)
         write(6,3010) bell
         write(6,1020)
         goto 10
      endif
c
20    call color(2,ierr)
c
c
c.....Draw the outline for the plot
      call plot(xzr,yzr+ylt,3)
      call plot(xzr+xlt,yzr+ylt,2)
      call plot(xzr+xlt,yzr,2)
      call plot(xzr,yzr,-3)
c
c.....Scale y-axis and x-axis
c
      call scale(y,ylt,npoint,1)
      call scale(x1,xlt,npoint,1)
c
c.....Writes the title of the plot
c
      nt2=32
      yt2=yzr+ylt+.2
      tit2='SENSOR SIGNAL vs SHEAR STRESS'
      call symbol(xt2,yt2,0.24,tit2,0.0,nt2)
c
c.....Draw the axes
c
      nx=4
      ny=3
      xaxt=' X1 '
```

```
      yaxt=' Y '
      xfst = x1(npoint+1)
      xdel = x1(npoint+2)
      yfst = y(npoint+1)
      ydel = y(npoint+2)
      xplot(ny)=xfst
      xplot(nx)=xdel
      yplot(ny)=yfst
      yplot(nx)=ydel
      call staxis(0.15,0.24,0.1111,0.1111,3)
      call axis (0.0,0.0,xaxt,-nx,-xlt,0.0,xfst,xdel)
      call axis (0.0,0.0,yaxt,ny,-ylt,90.0,yfst,ydel)
c
c.....Plot the curve
c
      call line(x1,y,npoint,1,-1,11)
      call line(xplot,yplot,2,1,0,0)
c
c.....Output the drawing to the screen
c
      call plot(0.,0.,999)
c
c
1010  format(//////////////////////)
1020  format(/20x,'Do you want the plot sent to the Printer (P)',
     &       /20x,'or Screen (S).  Enter P or S and hit return.',
     &      ////////)
1030  format(/28x,'PLOT IN PROCESS'//////////////)
3000  format(a)
3010  format(/2x,'Invalid entry',a)
      return
      end
```

APPENDIX 1.B

PROGRAM VARIABLES

The symbols used in this program are defined as followed:

A	- Slope of the least squares curve fit
Answer	- Logical Input variable; yes or no
B	- Intercept of the least square fit
COEF	- Correlation Coefficient; the variable C in Eq. (1.6)
deln	- Step size; the variable ε_i in Eq. (1.9)
h	- Small increment; $\hbar$ in Eq. (1.7)
lowern	- Lower bound for C vs. n plot
n or 1/n	- Exponent for the surface shear stress
nplot	- Number of points to plot
npoint	- Number of data points in file; N in Eq. (1.2)
nstep	- Change in n; Δn_i in Eq. (1.8)
nt2	- Number of characters in title of plot
nx	- Number of characters in the x-axis label
ny	- Number of characters in the y-axis label
oldslp	- Stores the old slope value
points	- Real value of N; Equal to npoints
slope	- Derivative of C; $d[C(n_i)]/dn$
st2	- Sum of the squares of the mean deviation
sumx1	- Sum of the x1 data
sumy	- Sum of the y data
sxoss	- Mean
t	- Mean deviation
tol	- Tolerance; δ in Eq. (1.13)
uppern	- Upper bound for C vs. n plot
x	- Array for wall shear stress data
x1	- Array for wall shear stress data raised to 1/n power
xaxt	- Label for x-axis
xdel	- Number of data points per x-axis inch
xfst	- Value at the first tick mark of x-axis
xlt	- Length of the x-axis
xplot	- Array for n data
xt2	- Positions the title with respect to x-axis
xzr	- Zero point for the x-axis
y	- Array for sensor signal data
yaxt	- Label for y-axis
ydel	- Number of data point per y-axis inch
yfst	- Value at the first tick mark of y-axis
ylt	- Length of the y-axis
yplot	- Array for C data
yt2	- Position the title with respect to y-axis
yzr	- Zero point for y-axis

APPENDIX 1.C

Exact Data Example

File: OUTPUT.DAT1

shear stress	sensor signal
.0000	1.0000
.5000	2.5157
1.0000	3.0000
1.5000	3.3522
2.0000	3.6390
2.5000	3.8854
3.0000	4.1037

This the computer-generated
best fit of your data.

Slope:	A =	2.0000
Intercept:	B =	1.0000
Coefficient:	coef =	1.0000
Best n	n =	2.5000

APPENDIX 1.D

Inexact Data Example

File: OUTPUT.DAT2

shear stress	sensor signal
.0000	1.0000
.5000	2.5157
1.0000	2.9000
1.0000	3.1000
1.5000	3.3522
2.0000	3.5390
2.0000	3.7390
2.5000	3.8854
3.0000	4.1037

This the computer-generated
best fit of your data.

Slope:	A =	2.0001
Intercept:	B =	1.0000
Coefficient:	coef =	.9778
Best n	n =	2.5002

APPENDIX 1.E

Typical Laboratory Data Example

File: OUTPUT3.DAT

shear stress	sensor signal
94.2000	20.0000
162.7700	20.9500
291.4300	22.0000
523.6100	23.5000
690.8100	24.2500
1710.7800	27.0000
2331.4700	28.5000
3702.2900	30.2500
6028.5700	32.5000
9514,6500	35.0500
15550.1200	38.0000

This the computer-generated
best fit of your data.

Slope:	A =	1.7120
Intercept:	B =	14.0139
Coefficient:	coef =	.9745
Best n	n =	3.6545

CHAPTER 2

RENOVATION OF THE SANDIA 5-METER WIND TURBINE

by

Jocelyn R. Wright

ABSTRACT

The 5-m wind turbine was renovated so that various blade modification tests, including pumped spoiling, could be performed. The renovation included repairs on the brake system, the sensor system, the anemometry system, verification of the operating procedure, and routine maintenance. The repair of the brake system included replacing the linear actuator, the back-up battery and charger. The sensor system repairs consisted of calibrating the torque sensor and replacing the torque/rpm signal indicator. The anemometry system had to be partially replaced. New wind direction and speed transmitters were installed, and the signal indication modules were reconnected. The operating procedures were verified, and the instrumentation located. The 5-m system was connected to the HP 1000 data analysis system. In addition various routine maintenance tasks were performed. The tower bearings must be lubricated and the belts replaced before full scale testing can commence. Once the remaining repairs are done the turbine will be fully operational.

INTRODUCTION

One of the present goals of the wind energy program at Sandia National Laboratories (SNL) is to design a more cost effective and efficient wind energy conversion system. Pumped spoiling is one method of blade modification that is being examined to achieve these goals. In this system the speed and power output of the turbine are controlled using a series of air jets controlled by valves at the blade ends.[2.1] When the valves are opened, the airflow through the jets spoils the flow over the surface of the blade, thus allowing additional control of the power generated by the turbine during high wind conditions. This in effect increases the life and decreases the cost of the machine.

Pumped spoiling and other blade modification tests will be performed on Sandia's 5-m vertical axis wind turbine (VAWT). This turbine is ideal for testing airfoils because its small size allows ease in removing and installing blades. The problem at hand was to refurbish the 5-m wind turbine.

A general description and background of the 5-m system follow, including a schematic of the electrical system and detailed descriptions of the inoperative systems. The necessary repairs were divided into tasks as follows: I. brake system, II. anemometry, III. sensor system, IV. control instrumentation, V. data acquisition, analysis and storage, and VI. maintenance.

DESCRIPTION OF THE WIND TURBINE

The 5-m VAWT test site is located at Sandia National Laboratories in Albuquerque, New Mexico, Fig. 2.1. The turbine was built in 1974 and was used for testing until 1984. It has produced 1 kilowatt of power in 15 mph winds.[2.2] In 1986, during an inspection, the brake was found to be broken; since that time the turbine has been inoperative.

The turbine stands 8.4 m (28 ft) high and is supported by four steel guy cables positioned 90° apart. The blades are 5 m (15 ft) in diameter at their equator. They were made by extruding 6063-T6 aluminum alloy into an airfoil shape, and then bending the extrusions to form an approximate troposkien.[2.3,2.4] The wind turbine is started using a 230-V ac induction generator, and its speed can be manually varied by changing the pulleys that connect the motor to the turbine shaft.[2.5] The output of the turbine is measured by a torque sensor, which measures the torque output by the turbine shaft and the turbine speed. The braking system consists of an electrical disc brake with an emergency back-up battery and a manual brake.[2.6] The ambient conditions are determined using wind speed and direction, temperature, and pressure transducers. These sensors and the data acquisition system are depicted in Fig. 2.2.

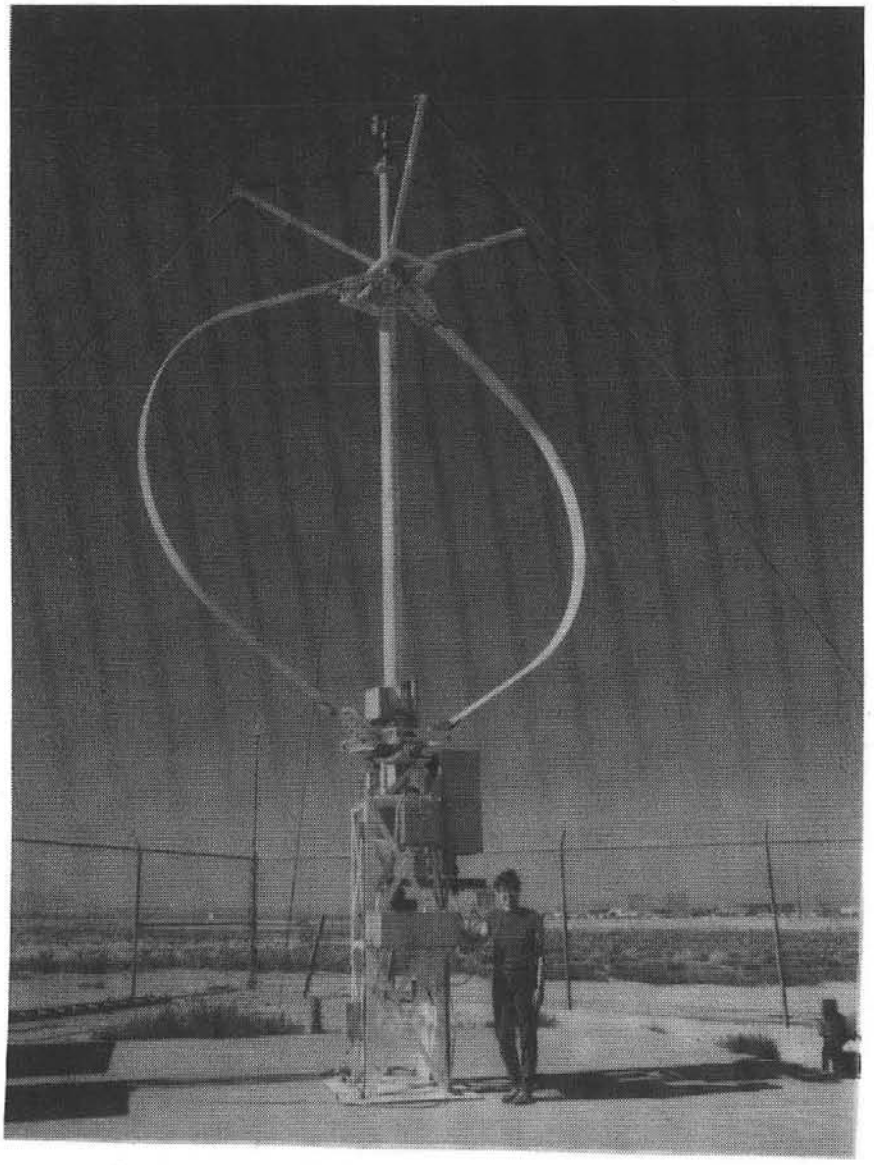

Figure 2.1. The 5-m VAWT.

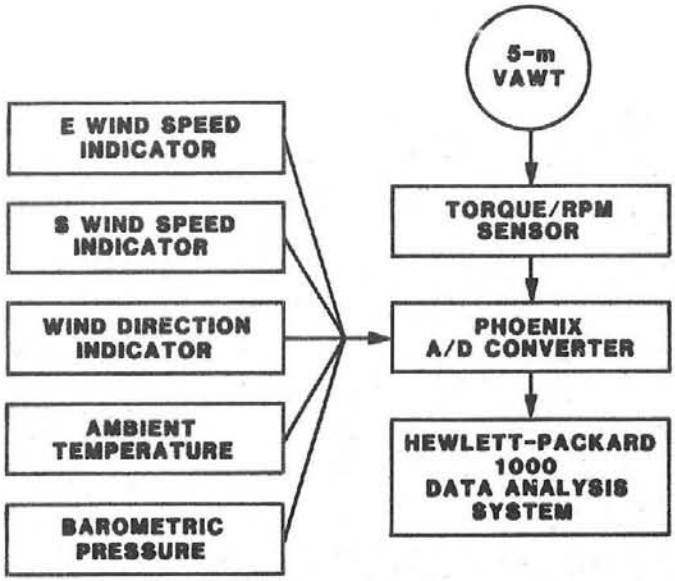

Figure 2.2. The 5-m system flow chart

PROBLEMS AND SOLUTIONS OF SUBSYSTEMS

Brake System

The brake system, pictured in Fig. 2.3, is activated when brake pads fixed to a set of calipers are forced against a disc that is fixed to and rotates with the turbine. The calipers are controlled using an electric linear actuator. The brake system layout is pictured in Fig. 2.4.

The braking system was inoperative because the linear actuator had a burned out motor and a fractured rear clevis mount.[2.7,2.8] In addition, the metal plate on which the actuator rested had perforated the underside of the arm housing. This and the fractured clevis mount were possibly due to an inadequate initial design. As a temporary measure, an actuator was constructed from the broken actuator and a spare that was broken also. Tape was installed around the arm housing to prevent perforation. The repaired actuator was nominally functional when installed on the turbine. In an attempt to have a more reliable braking system, it is recommended that two new linear actuators be ordered. One will replace the present actuator and one will serve as a back-up. Minor modifications may be needed to install the replacement actuator.

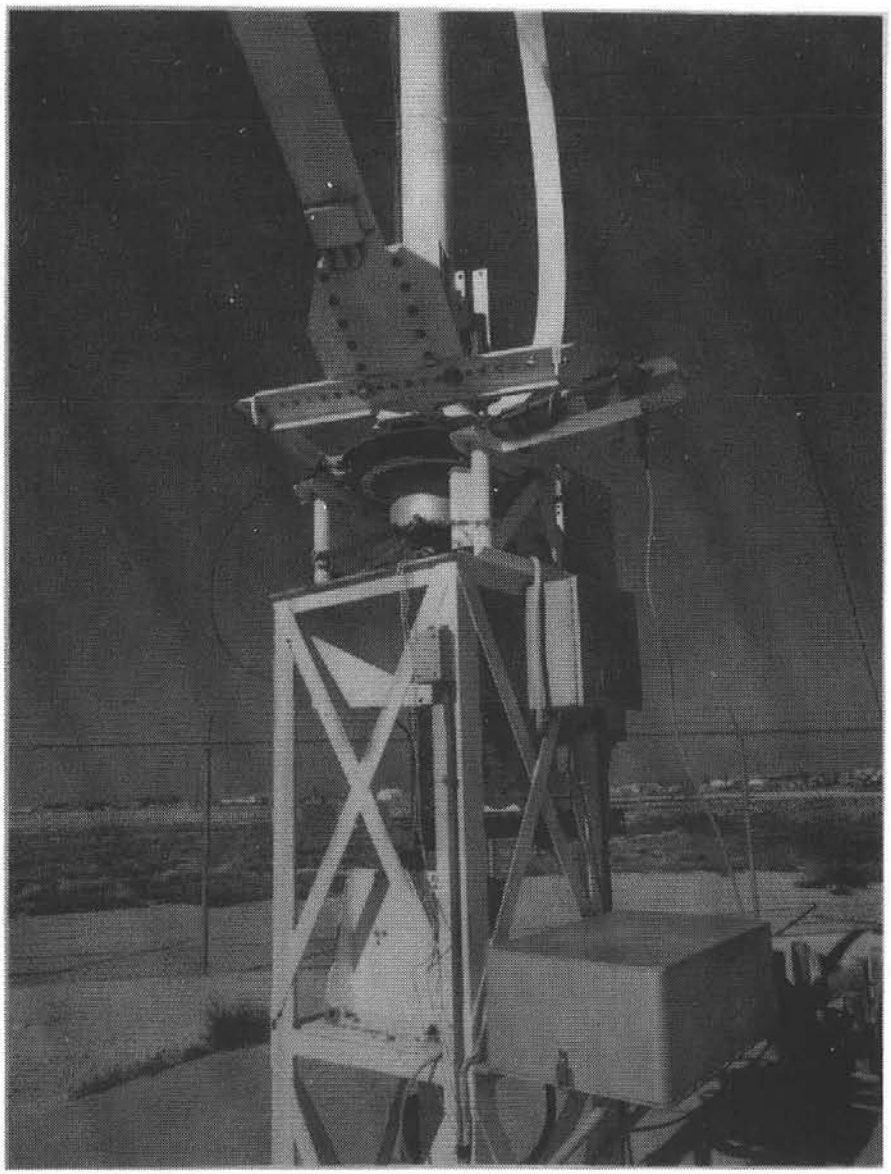

Figure 2.3. The brake system.

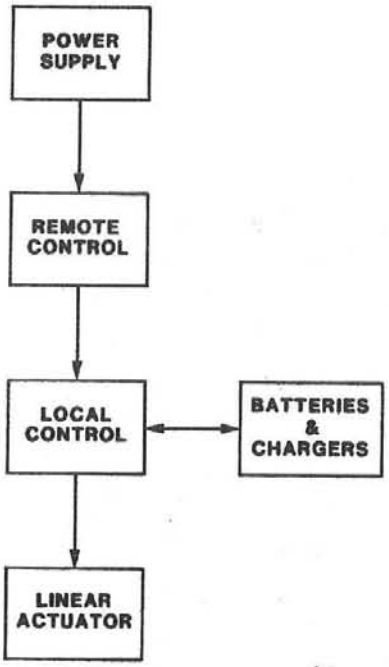

Figure 2.4. The brake system layout.

The 24-V back-up Ni-cad battery could not be charged and was too expensive to replace with another Ni-cad battery. It was replaced with two 12-V gel cell batteries placed in series.[2.9] The 24-V battery charger was replaced with two 12-V chargers.[2.10] The manual brake was functional, but the non-critical components of the arm were frozen. Although lubrication was not entirely effective in releasing the components, it was decided that the manual brake was functioning adequately.

The procedure for operating the disc brake system was unclear because the specific location of the control instrumentation was unknown. Visual inspection of Rack 5, located to the right of the control cabinet, Fig. 2.5, revealed possible brake controls, and test runs of the brake system confirmed that they were the controls. The operating procedure for this machine, called the "safe operating procedure" (SOP) at Sandia, had to be modified because the emergency braking procedure was unclear.[2.11]

Anemometry

To quantify the performance of a wind turbine requires a detailed description of the incoming flow field. At the 5-m site, this characterization is accomplished using wind speed and direction transmitters and indicators, and barometric pressure and ambient temperature transducers.[2.12] The pressure and temperature transducers are mounted on the outside of Building 899. Two wind speed transmitters, Fig. 2.6, are located on

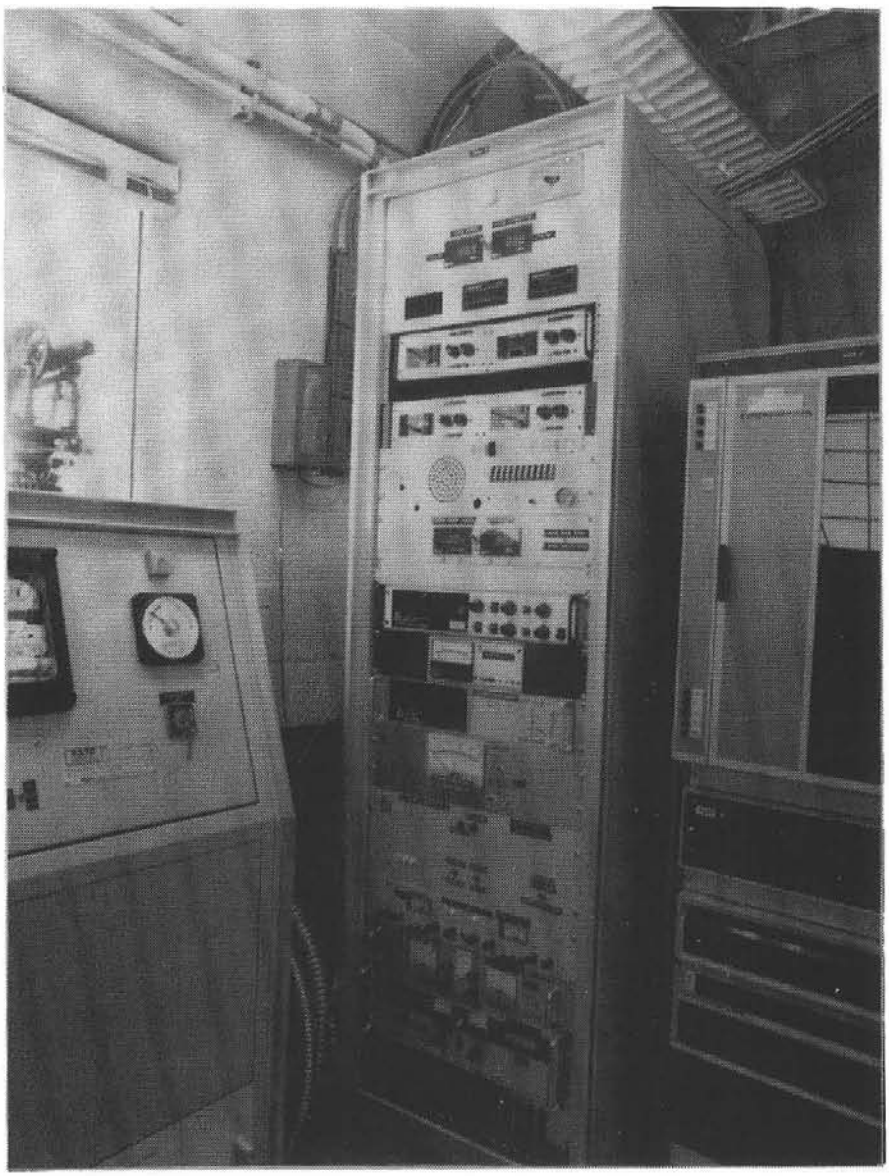

Figure 2.5. Rack 5 containing the brake controls and the torque/rpm sensor analog transducer.

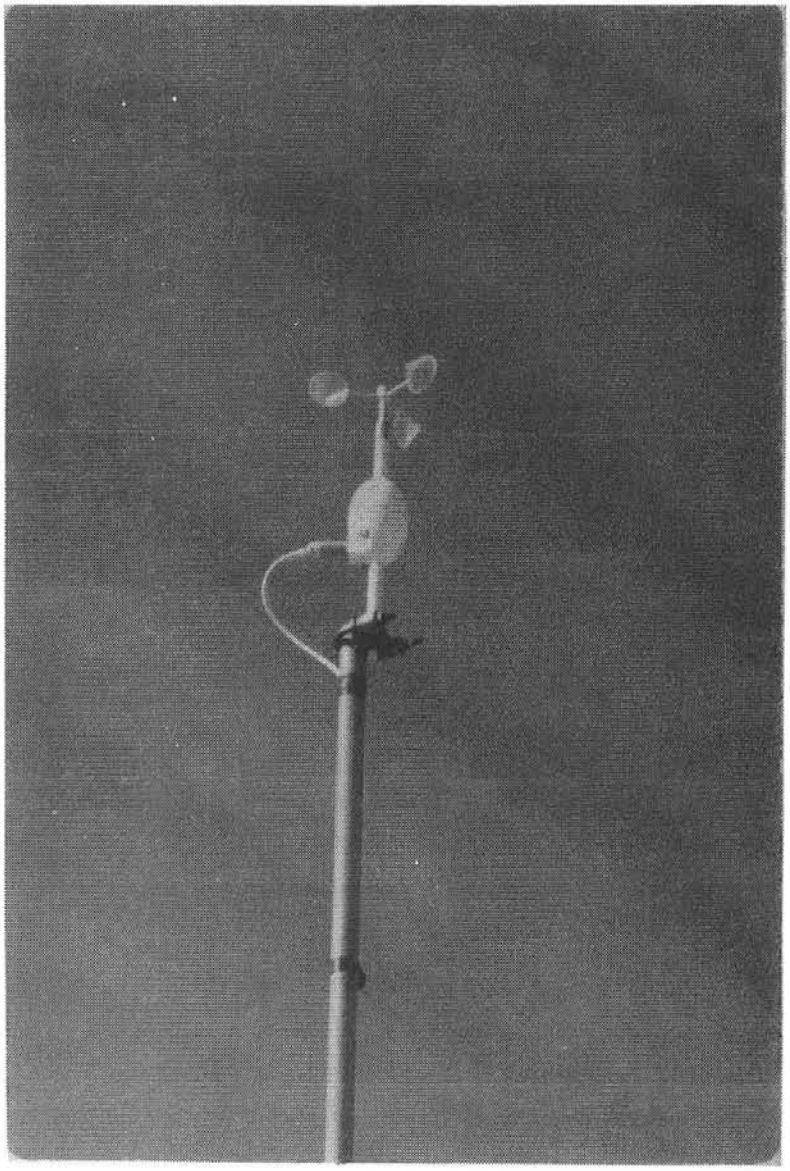

Figure 2.6. The wind speed indicator.

top of towers (at a height even with the equator of the blades), one on the south and one on the east corners of the security fence, surrounding the 5-m site. The wind direction indicator, Fig. 2.7, is located on top of the turbine tower. The signal indicators are located in Rack 1, the right instrumentation rack shown in Fig. 2.8. The system layout for the anemometry is pictured in Fig. 2.9.

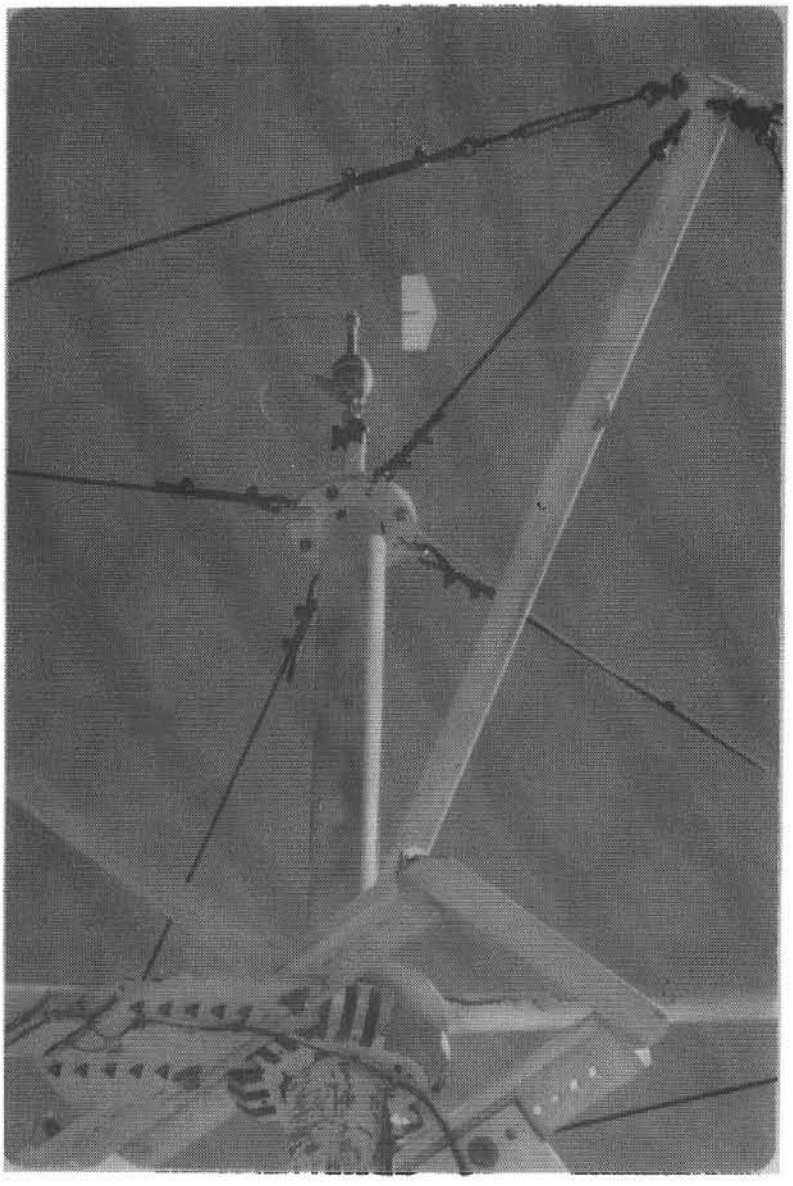

Figure 2.7. The wind direction indicator.

Figure 2.8. The wind speed and direction indicator modules, digital readouts, control switches, Rack 1, and Rack 2, the temperature readout and Phoenix A/D converter.

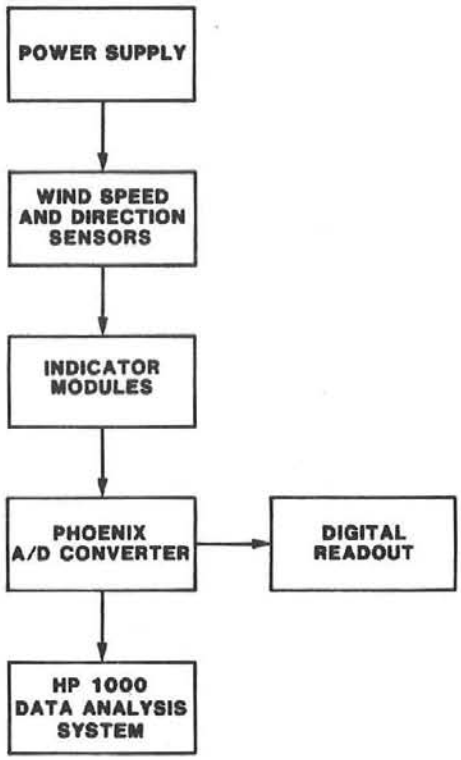

Figure 2.9. The system layout for the anemometry.

The wind speed and direction indicators were not operating. The wind speed indicators had missing cups and the wind direction indicator was missing part of the vane. These were replaced.[2.13]

The anemometry circuitry was discontinuous in Building 899. The indicator modules and power supply were reconnected, leaving the existing wiring intact. The instrumentation then functioned properly.

Control Instrumentation

The 5-m control panel in Fig. 2.10, the second panel from the left in the control panel cabinet, consists of ON and OFF switches and three meters: voltage (volts), power (kilowatts) and current (amps). The start-up procedure was listed in the SOP; however, the instrumentation was not marked, and the location of some instruments was unknown.[2.11] This caused some ambiguity in the procedure.

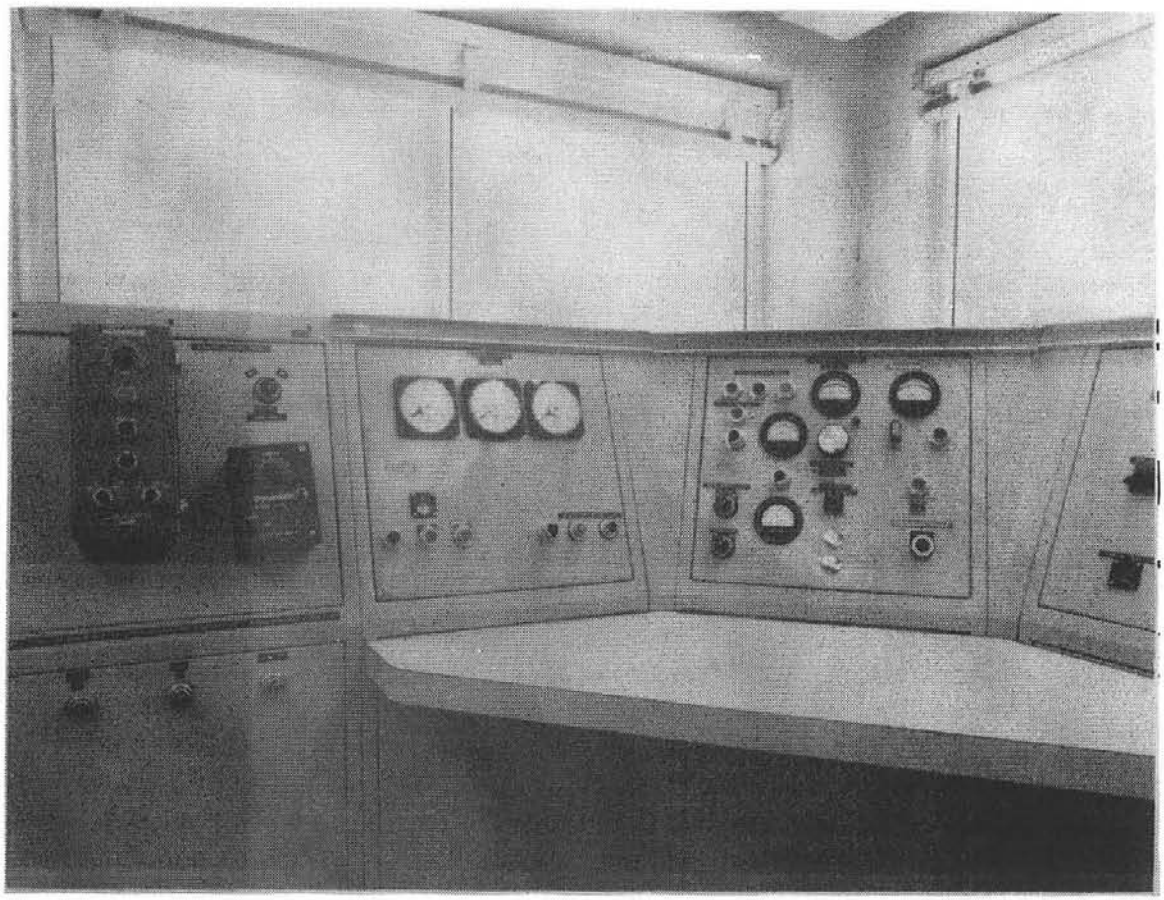

Figure 2.10. The 5-m VAWT control panel.

The ON/OFF controls for the 5-m turbine were located on the control panel adjacent to the controls for a 17-m turbine in Building 899. After the first run of the turbine, it was concluded that they worked and the start-up procedure in the SOP was correct. The circuit breaker was located in another building inside the security fence.

Sensor System

The output of the turbine is determined by characterizing the output torque and the speed of the turbine tower. A torque and rpm transducer is used to measure these parameters. Figure 2.11 shows the torque sensor mounted on the turbine.[2.14] These signals are read using an analog transducer. The output system layout is depicted in Fig. 2.12.

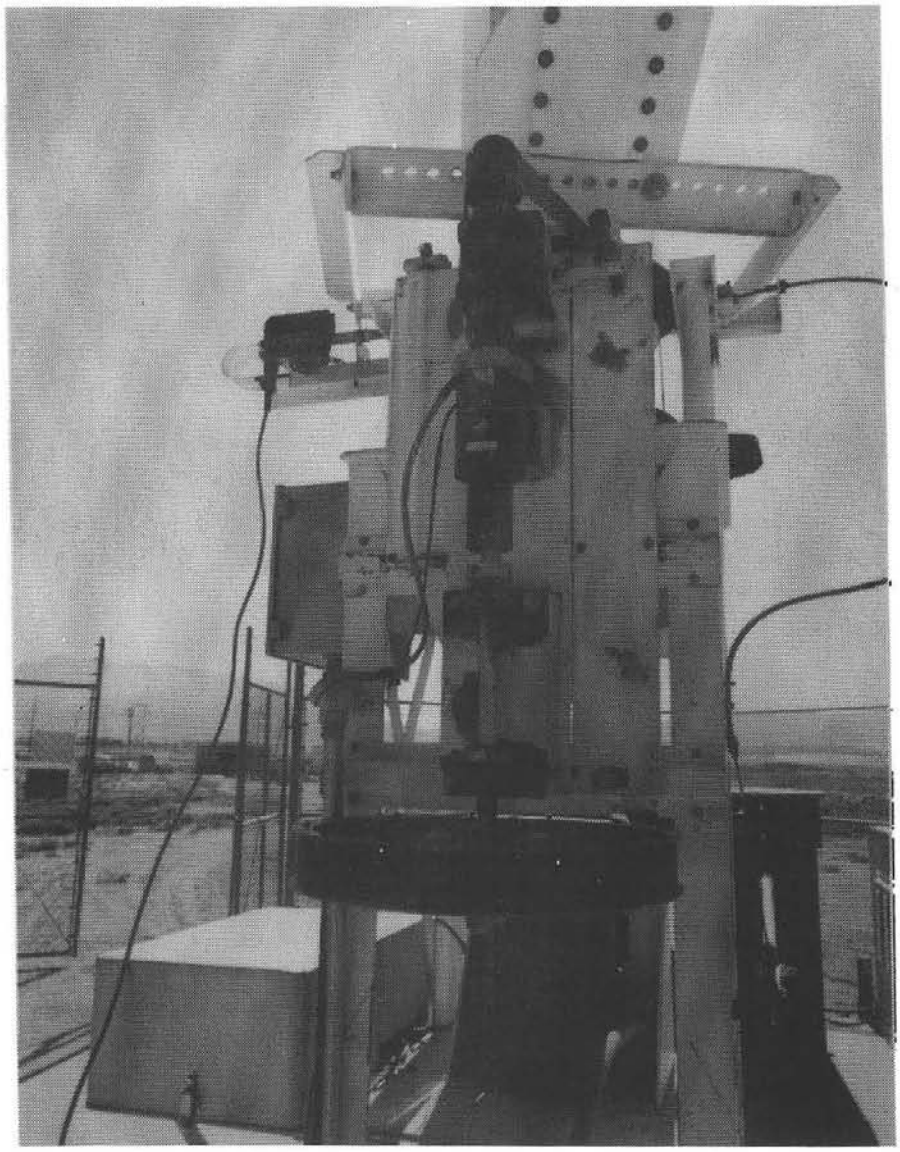

Figure 2.11. The torque sensor transducer mounted on the 5-m VAWT.

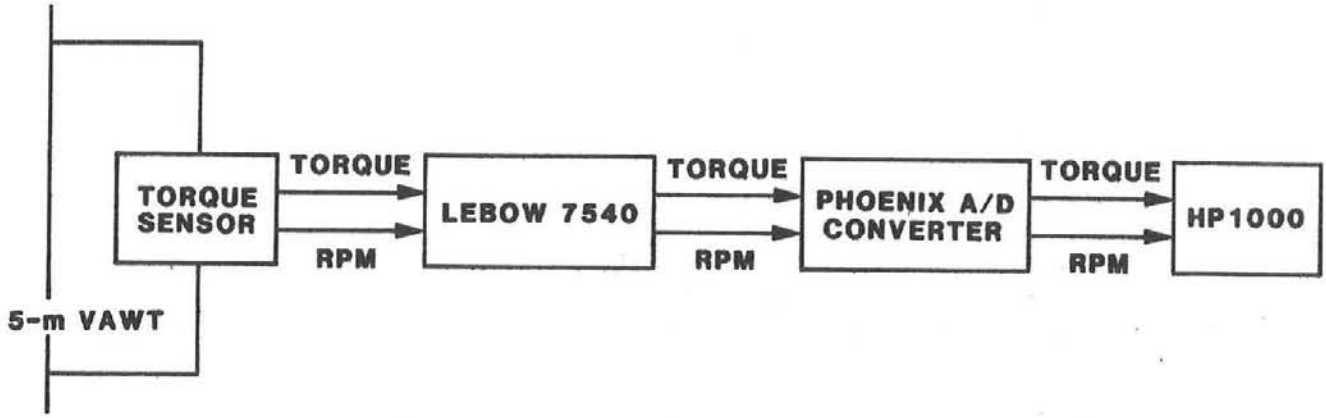

Figure 2.12. The schematic of the output system.

On the first run of the turbine, the torque sensor appeared to work, but it was unknown if its output was accurate. It was decided to have the sensor calibrated to insure it was functioning properly. Brackets were designed and machined, in preparation for calibration. The rpm indicator was not giving logical readings, and it was discovered that its wires had been broken, between the junction boxes JBO2A (located in the building containing the junction boxes) and JBO3B (located on the southeast side of the turbine stand near the concrete slab). The wiring was repaired.

The torque and the rpm indicators were replaced with a single transducer.[2.15] When connected, both rpm and torque indications registered. However, the rpm readings had to be adjusted because the transducer was reading speeds on the order of three magnitudes too high.

Data Acquisition, Analysis and Storage

The Hewlett-Packard 1000 Data Acquisition and Analysis System (DAAS) is used to take and analyze data from testing the turbine.[2.16,2.17] The incoming parameters include wind speed and direction, torque and speed of the turbine, ambient temperature, and barometric pressure. These signals are channeled through the Phoenix analog-to-digital (A/D) converter, then sent to the HP computer.[2.18] At this point the data can be analyzed using data analysis software and/or stored on magnetic tape.

Initially, the 5-m system was not linked to the DAAS system. The connection between the two was accomplished by wiring between the wind speed and direction signals

output from the indicator modules in Rack 1, and the turbine speed and torque output from the rpm/torque indicator to the Phoenix A/D converter, located in Rack 2 (the left instrumentation rack in Fig. 2.8).

Maintenance

The drive belt from the motor to the large pulley on the torque sensor shaft, the belt from the small pulley on the torque sensor shaft to the turbine shaft and the belt around the turbine shaft below the slip rings (a spare) appear to have been damaged by the environment.[2.19] The belts are shown in Fig. 2.11. Visual inspection revealed brittleness and some cracking. Changing the turbine shaft belt and the spare requires disassembling the turbine. It was decided not to change any belts at this time, as the machine can be run with the existing belts. It is recommended that these be changed in the near future.

The maintenance schedule indicates that the lower tower bearings should be lubricated every 3 months, but does not mention how the bearings are accessed.[2.20,2.21] Visual inspection of the turbine revealed no evidence of lubricant. There were no lubricant stains on the tower in the areas where the bearings are suspected to be located. Since there were no indications of tower bearing problems during the first run of the turbine, it was decided to limit the number of test runs of the turbine until the tower bearings can be lubricated.

The bearings located above and below the torque sensor on the torque sensor shaft were lubricated on June 30, 1988.[2.22]

CONCLUSION

The 5-m vertical axis wind turbine was refurbished so that various blade modification tests, including pumped spoiling, could be performed. The components repaired were the brake, anemometry, output and control systems. Repair of the brake system included replacing the linear actuator, the back-up battery, and charger. The output system repairs consisted of calibrating the torque sensor and replacing the torque/rpm signal indicator. The anemometry system had to be partially replaced. New wind direction and speed transmitters were installed, and the signal indication modules were reconnected. The operating procedures were verified and the instrumentation located. The HP 1000 data analysis system was linked with the 5-m system. In addition various routine maintenance tasks were performed.

At this point the turbine is mechanically reliable and able to run. However, a new actuator should be ordered and installed, the pulley belts should be changed and tower bearings located and lubricated before any testing is conducted.

ACKNOWLEDGEMENTS

The author wishes to thank Ted Bryant for his assistance and guidance in activating the 5-m turbine, Paul Klimas for his help in providing information on pumped spoiling, and Herb Sutherland for his assistance in editing this report and encouragement.

REFERENCES

2.1. P. C. Klimas, J. Sladky, "Pumped Spoiling VAWT Control," Proceedings of the Fourth ASME Wind Energy Symposium, Dallas, TX, February 17-21, 1985.

2.2. "Sandia Laboratories Vertical-Axis Wind Turbine", Brochure Number 5/78/15M, Sandia National Laboratories, Albuquerque, NM, 87185.

2.3. N. H. Lipman, P. J. Musgrove and G W-W Pontin, eds., Wind Energy for the Eighties (BWEA), Peter Peregrinus Ltd., New York, NY, pp. 54-8.

2.4. V. D. Hunt, Windpower A Handbook on Wind Energy Conversion Systems, Van Nostrand Reinhold Company, New York, NY, 1981, p. 125.

2.5. Dayton AC Induction Motor, model 6K146E, Dayton Electronic Manufacturing Company, Chicago, IL 60648.

2.6. Kelsey Hayes Caliper 375M2, Kelsey Hayes Wheel Drum and Brake Division, 5300-T Livernois, Detroit, MI, 48232.

2.7. AiResearch Automatic Flap Actuator 25226, AiResearch Manufacturing Company of Arizona.

2.8. VAWT Test Site Daily Log Book, 1974 - 1986, Building 899, Sandia National Laboratories, Albuquerque, NM 87185.

2.9. 12-V gel cell battery CFM12V6.5, no other information available.

2.10. L power CH12FA800 charger, no other information available.

2.11. "Safe operating procedure for the vertical axis wind turbine test facility", Safe Operating Procedure (SOP) 01700 8801, Sandia National Laboratories, Albuquerque, NM 87185.

2.12. YSI Pressure Transducer 2014-710/1050-MB-3-WH, YSI Temperature Transducer 2014-710, Yellow Springs Instrument Co. (YSI), Inc., Yellow Springs, OH 45387.

2.13. Teledyne Geotech Wind Speed Transmitter model 1564B with anemometer cup assembly model 170-41, Teledyne Geotech Wind Direction Transmitter model 1565B with wind vane model 53.2. Teledyne Geotech, 3401 Shiloh Road, Garland, Texas.

2.14. Lebow Torque Sensor model 1604-500. Lebow Associates, Inc., 1728 Mapletown Road, P. O. Box 1089, Troy, Michigan.

2.15. Old torque sensor: Lebow 7535; Old rpm sensor: Lebow 7212; New torque and rpm sensor: Lebow 7540.

2.16. HP 1000 System, Data Systems Division, Hewlett-Packard, HP 1000 Wolfe Rd., Cupertino, CA 95014-9974.

2.17. D. Berg, M. Rumsey, L. Gallo and D. Burwinkle, "Data Acquisition and Analysis System for the Sandia 34-Meter Vertical Axis Wind Turbine," Proceedings of the Seventh ASME Wind Energy Symposium, SED-Vol. 5, A. H. P. Swift and R. W. Thresher, eds., ASME, 1988, p. 53.

2.18. Phoenix Data, Inc., Phoenix, AZ.

2.19. Gates synchro-power timing belt models 800H200 (bottom) and 700H200 (top), Kaman Bearing and Supply Corp. - Intermountain, 1613 First Street NW, P. O. Box 25242, Albuquerque, NM 87125.

2.20. Kaydon ball bearings KF055XP0 (top bearing) and KF055AH0 (bottom bearing), Kaydon Corporation, 2862-TM McCracken Ave., Muskegon, MI 49443.

2.21. 5-M Maintenance Schedules, Building 899, Sandia National Laboratories, Albuquerque, NM 87185, unpublished internal document.

2.22. Federal ball bearing pillow block SC2100, 1" bore, The Federal Bearings Co., Inc., Poughkeepsie, NY.

APPENDIX 2.A

MICROSCOPIC ANALYSIS OF 6063-T6 ALUMINUM ALLOY FAILED IN BENDING FATIGUE

INTRODUCTION

At Southern University in Baton Rouge, Louisiana, 6063-T6 aluminum alloy bending specimens are being fatigue tested. Three samples from a single specimen were mounted and examined using an optical microscope. Based on these results, it was concluded that the crack propagation was inter-granular. Additional examinations of these fracture surfaces were conducted at Sandia National Laboratories using a scanning electron microscope (SEM). A sample was cut from the specimen and was cleaned using an ultrasonic cleaner. It was then mounted on a carbon stub. A thin, uniform film of gold-palladium was deposited on the fracture surface using a vacuum sputtering depositor to insure a continuous conducting surface during examination in the SEM.

RESULTS AND ANALYSIS

The SEM pictured in Fig. 2.A.1 is of the initial crack zone in which mostly inter-granular crack propagation is seen. However, there are small regions of trans-granular cracking.[2.A1]

In Fig. 2.A.2 notice the ductile striations at the bottom and the dimpled region throughout the upper portions of the photo. The striations and the dimpled appearance are indications that this region failed in a ductile mode. It is the overload region.[2.A1]

Fig. 2.A.3 is a schematic of the specimen fracture surface showing the direction of cracking and fracture mode regions.[2.A1] Note the specimen was cycled with a mean stress so that the tensile bending stress was higher on one side than the other; crack initiation occurred first on the high tensile stress side.

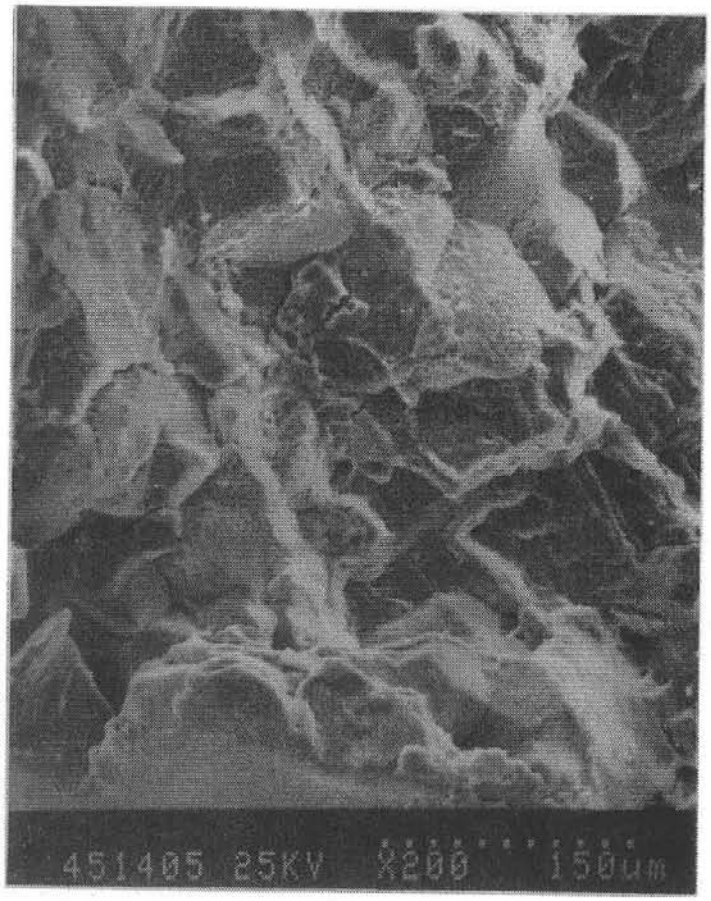

Figure 2.A.1. SEM picture of the initial crack zone.

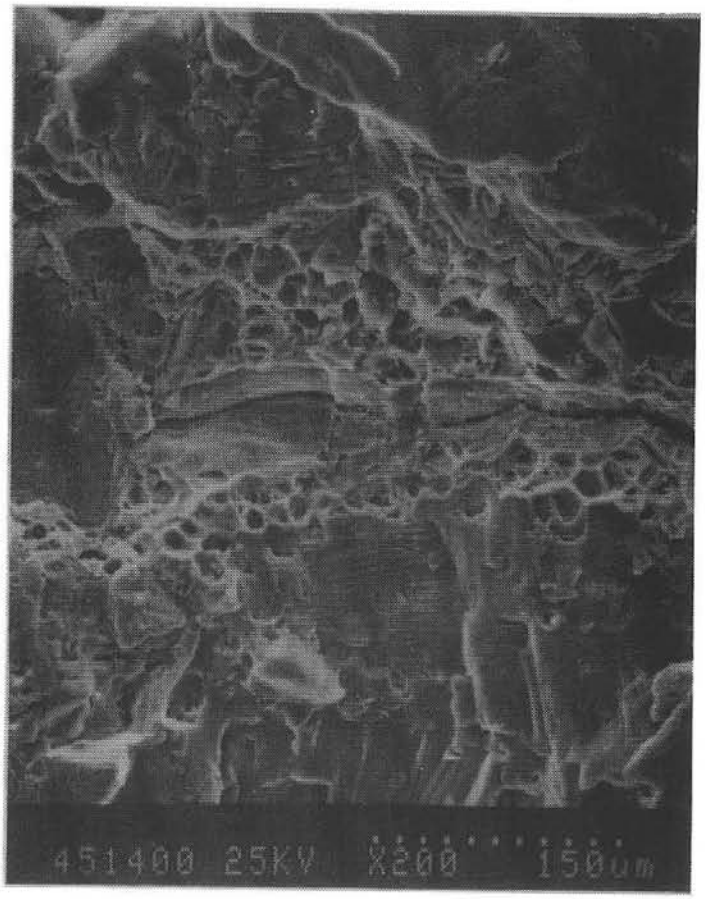

Figure 2.A.2. SEM picture of the overload zone which is characterized by high stress.

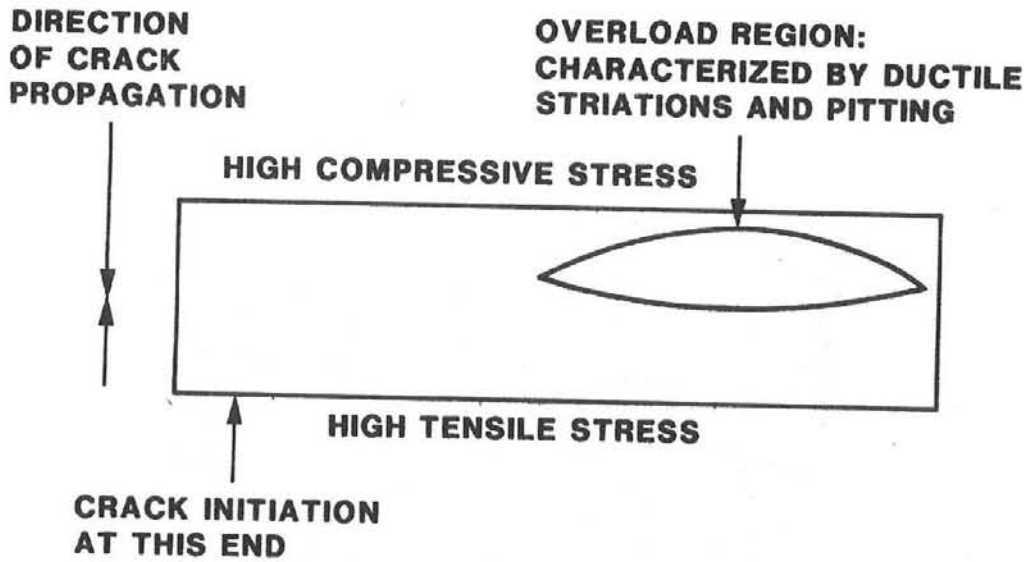

Figure 2.A.3. Schematic of the fracture surface.

Figure 2.A.4 shows the region of Fig. 2.A.3 just below the dimpled region. The higher magnification, 600X, allows distinction of the ductile striations leading to the final overload region. Notice that the spacing between the striations increases the closer to the dimpled region.

CONCLUSION

The 6063-T6 aluminum alloy specimen fractured in bending fatigue was examined using the scanning electron microscope. Analysis of the fracture surface revealed mainly inter-granular cracking, but there was evidence of some intra-granular cracking. Examination also revealed ductile failure in the overload region. More specimens need to be examined microscopically before any conclusive results can be drawn regarding the behavior of this material.

The SEM is a much more powerful microscope than the optical microscope with which the fracture surface could not be viewed with any clarity. Thus the fracture modes presented above were determined with much more accuracy and detail.

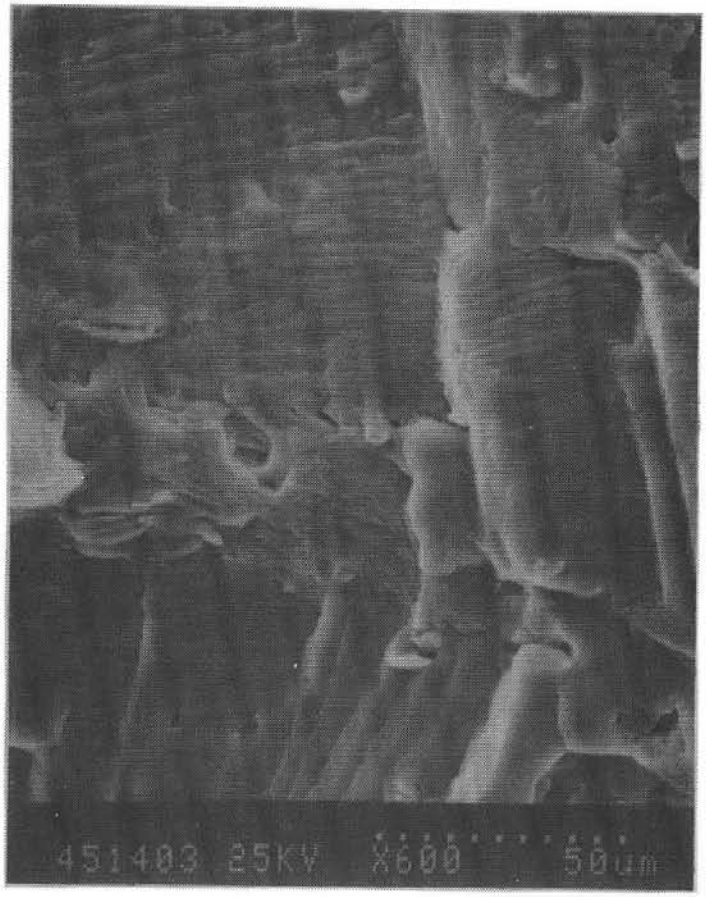

Figure 2.A.4. SEM picture of the region leading to the overload region.

ACKNOWLEDGEMENTS

The author wishes to thank James VanDenAvyle for his help in evaluating this specimen. The author also wishes to thank James VanDenAvyle and Paul Hatch for their informative tour of the materials testing laboratory.

REFERENCES

2.A1. James VanDenAvyle, June 29, 1988, SEM Laboratory, Sandia National Laboratories, Albuquerque, NM.

DISTRIBUTION:

Amarillo College
Amarillo, TX 79100
Attn: E. Gilmore

American Wind Energy Association
1017 King Street
Alexandria, VA 22314

Battelle-Pacific Northwest
Laboratory
P.O. Box 999
Richland, WA 99352
Attn: L. Wendell

Colorado State University
Dept. of Civil Engineering
Fort Collins, CO 80521
Attn: R. N. Meroney

DOE/ALO/ETWMD
Albuquerque, NM 87115
Attn: G. P. Tennyson

DOE/ALO
Energy Technology Liaison Office
NGD
Albuquerque, NM 87115
Attn: Capt. J. L. Hanson, USAF

DOE Headquarters (5)
Wind/Oceans Technologies Division
1000 Independence Avenue
Washington, DC 20585
Attn: L. J. Rogers
P. R. Goldman
O. Dyes

Fayette Manufacturing Corporation
P.O. Box 1149
Tracy, CA 95378-1149
Attn: W. Thompson

FloWind Corporation
1183 Quarry Lane
Pleasanton, CA 94566
Attn: L. Schienbein

Dr. I. J. Graham
Southern University
Dept. of Mechanical Engineering
P.O. Box 9445
Baton Rouge, LA 70813-9445

Iowa State University
Agricultural Engineering
Room 213
Ames, IA 50010
Attn: L. H. Soderholm

K. Jackson
West Wind Industries
P.O. Box 1705
Davis, CA 95617

Robert Lynette
R. Lynette & Assoc., Inc.
15042 NE 40th Street
Suite 206
Redmond, WA 98052

Massachusetts Institute of
Technology
77 Massachusetts Avenue
Cambridge, MA 02139
Attn: Professor N. D. Ham
W. L. Harris, Aero/Astro
Dept.

Michigan State University
Division of Engineering Research
East Lansing, MI 48825
Attn: O. Krauss

New Mexico Engineering
Research Institute (2)
Campus P.O. Box 25
Albuquerque, NM 87131
Attn: G. G. Leigh
D. Morrison

Ohio State University
Aeronautical & Astronautical Dept.
2070 Neil Avenue
Columbus, OH 43210
Attn: Professor G. Gregorek

Oklahoma State University
Mechanical Engineering Dept.
Stillwater, OK 76074
Attn: D. K. McLaughlin

Oregon State University
Mechanical Engineering Dept.
Corvallis, OR 97331
Attn: R. E. Wilson

RANN, Inc.
260 Sheridan Ave., Suite 414
Palo Alto, CA 94306
Attn: A. J. Eggers, Jr.

Dr. R. Ganesh Rajagopalan,
Asst. Prof.
Aerospace Engineering Department
Iowa State University
404 Town Engineering Bldg.
Ames, IA 50011

Gwen Schreiner
Librarian
National Atomic Museum
Albuquerque, NM 87185

Solar Energy Research Institute
1617 Cole Boulevard
Golden, CO 80401
Attn: R. W. Thresher

Stanford University
Dept. of Aeronautics and
Astronautics Mechanical Engr.
Stanford, CA 94305
Attn: Holt Ashley

Texas Tech University
Mechanical Engineering Dept.
P.O. Box 4289
Lubbock, TX 79409
Attn: J. W. Oler

Tulane University
Dept. of Mechanical Engineering
New Orleans, LA 70118
Attn: R. G. Watts

University of California
Institute of Geophysics
and Planetary Physics
Riverside, CA 92521
Attn: Dr. P. J. Baum

University of Colorado
Dept. of Aerospace
Engineering Sciences
Boulder, CO 80309
Attn: J. D. Fock, Jr.

University of Massachusetts
Mechanical and Aerospace
Engineering Dept.
Amherst, MA 01003
Attn: Dr. D. E. Cromack

University of Oklahoma
Aero Engineering Department
Norman, OK 73069
Attn: K. Bergey

The University of Tennessee
Dept. of Electrical Engineering
Knoxville, TN 37916
Attn: T. W. Reddoch

USDA
Agricultural Research Service
Southwest Great Plains Research
Center
Bushland, TX 79012
Attn: Dr. R. N. Clark

W. A. Vachon
W. A. Vachon & Associates
P.O. Box 149
Manchester, MA 01944

Washington & Lee University
P.O. Box 735
Lexington, VA 24450
Attn: Dr. R. E. Akins

Washington State University
Dept. of Electrical Engineering
Pullman, WA 99163
Attn: F. K. Bechtel

West Texas State University
Government Depository Library
Number 613
Canyon, TX 79015

West Texas State University
Department of Physics
P.O. Box 248
Canyon, TX 79016
Attn: V. Nelson

West Virginia University
Dept. of Aero Engineering
1062 Kountz Avenue
Morgantown, WV 26505
Attn: R. Walters

Wichita State University
Aero Engineering Department (2)
Wichita, KS 67208
Attn: M. Snyder
W. Wentz

Wind Power Digest
P.O. Box 700
Bascom, OH 44809
Attn: Michael Evans

Jocelyn R. Wright (10)
P.O. Box 1971
Baton Rouge, LA 70821

Jesse D. Sims (10)
515 Slumberwood
Houston, TX 77013

400	R. C. Maydew
1520	L. W. Davison
1550	C. W. Peterson
3141	S. A. Landenberger (5)
3151	W. I. Klein (3)
3154-3	J. F. Hayes, Actg. (8) For DOE/OSTI (Unlimited Release)
3161	P. S. Wilson
3000	D. Roth
3500	R. C. Bonner
3510	M. R. Baca
3510	P. J. Salisbury (10)
6000	D. L. Hartley
6200	V. L. Dugan
6220	D. G. Schueler
6221	J. L. Chamberlin
6225	H. M. Dodd (50)
6225	T. D. Ashwill
6225	D. E. Berg
6225	T. C. Bryant
6225	P. C. Klimas
6225	S. D. Nicolaysen
6225	M. E. Ralph
6225	D. C. Reda
6225	M. A. Rumsey
6225	L. L. Schluter
6225	W. A. Stephenson
6225	H. J. Sutherland (5)
6410	N. R. Ortiz
7544	D. O. Smallwood