NREL/TP-442-7112 • UC Category: 1211 • DE95000223

GPP—A General-Purpose Post Processor for Wind Turbine Data Analysis

Marshall L. Buhl, Jr.
Neil D. Kelley
David A. Simms

Prepared for
ASME Wind Energy Symposium
January 29–February 1, 1995
Houston, Texas

NREL

National Renewable Energy Laboratory
1617 Cole Boulevard
Golden, Colorado 80401-3393
A national laboratory of the U.S. Department of Energy
Managed by Midwest Research Institute
for the U.S. Department of Energy
under contract No. DE-AC36-83CH10093

Prepared under Task No. WE429150

October 1994

Available to DOE and DOE contractors from:
Office of Scientific and Technical Information (OSTI)
P.O. Box 62
Oak Ridge, TN 37831
Prices available by calling (615) 576-8401

Available to the public from:
National Technical Information Service (NTIS)
U.S. Department of Commerce
5285 Port Royal Road
Springfield, VA 22161
(703) 487-4650

Printed on paper containing at least 50% wastepaper and 10% postconsumer waste

GPP—A GENERAL-PURPOSE POST PROCESSOR FOR WIND TURBINE DATA ANALYSIS

Marshall L. Buhl, Jr.
Neil D. Kelley
David A. Simms
Wind Technology Division
National Renewable Energy Laboratory
Golden, Colorado

ABSTRACT

GPP[1] (pronounced "jeep") is a General-Purpose Post Processor for wind turbine data analysis. Engineers in the Wind Technology Division (WTD) of the National Renewable Energy Laboratory (NREL) developed it to postprocess test data and simulation predictions. GPP reads data into large arrays and allows you to run many types of analyses on the data in memory. GPP runs on inexpensive computers commonly used in the wind industry. You can even use it on a laptop computer in the field.

We wrote the program in such a way as to make it easy to add new types of analyses and to port it to many types of computers. Although GPP is very powerful and feature-rich, it is still very easy to learn and use. Exhaustive error trapping prevents you from losing valuable work due to input errors. GPP should make a significant impact on engineering productivity in the wind industry.

INTRODUCTION

GPP started out as a loose collection of utility programs. As the number of utilities grew, we decided that there was too much duplication among the programs. Our solution was to write one program that would input the data and then manipulate the data in a variety of ways. This saves us time not only when we want to add a new type of analysis, but also when we do the analysis—we only have to read in the data once for all tools instead of once for each tool.

The first version of GPP included only those original tools and ran only on Silicon Graphics, Inc. (SGI) computers. Since then, others have made requests for many additional features including the ability to run GPP on a variety of platforms.

To make the code platform independent, the authors put all compiler-specific routines into a separate file so that the majority of the code would require no modifications to run on other systems. This includes all operating system calls such as date and time, and all screen output.

As of this writing, GPP runs on UNIX computers from SGI, Sun Microsystems, and Hewlett-Packard (HP). It also runs on personal computers (PCs) and we have compiled it with Lahey Computer Systems' F77L-EM/32 and Microway's NDP FORTRAN. These two PC compilers are often available in the wind energy industry. We chose them because of their availability in the WTD and because they allowed the huge arrays that are necessary for GPP. Since GPP is so platform independent, it should be possible to port it to a new system with less than a day's effort.

One thing we hope to accomplish with GPP is to provide engineers with a user-friendly tool that makes it easy to look at their data in many different ways. Experience has shown us that we need to analyze our data with varied techniques. A power spectral density (PSD) plot comparing test data to simulation predictions may show excellent agreement, while rainflow cycle count spectra tell a very different story.

We wrote GPP to be a menu-driven, question-and-answer type program instead of the old-style batch type program. We created a character-based program instead of one with a graphical-user interface (GUI) for several reasons. Although GUI programs have impressive displays, they are difficult to code, they run much slower than character-based programs, and they are inherently non-portable. The portability requirement also prevented us from adding graphical output. GPP's output files import easily into standard graphics packages.

Because GPP reads entire data sets into arrays, it is *very* fast. Once the data is in memory, a modern PC can perform many analyses in just seconds. Older analysis tools had severe memory restrictions due to the limitations of the technology then available. Since they had to store information on disk, they were very slow.

[1] GPP is a trademark of the National Renewable Energy Laboratory.

In GPP, many prompts have intelligent defaults and the less-obvious questions have context-sensitive help. In some cases, there are even on-line examples to help you. We put great care into the program's usability and we hope we have made it "fool resistant." GPP traps input errors and requests the input again.

The targeted users for GPP are engineers practicing in wind energy research and industry. The program assumes that you are familiar with the concepts and techniques it implements.

This paper serves as an overview of GPP. Please refer to the GPP User's Guide (Buhl, 1994) for details on using it.

USING GPP

Background

Once you understand the working philosophy of GPP, you should be able to use it fairly effortlessly. We have tried to make GPP as friendly as possible; it has a lot of on-line, context-sensitive help, and it's good at trapping input errors.

GPP uses a combination of menus and a question-and-answer coding style. You choose tools from various menus to tell GPP what tasks to perform. Once GPP finishes processing the data with the tool, it returns you to an appropriate menu. GPP removes tools that are not suitable in a given situation from the menus to avoid confusing you. For instance, there is no reason to give you the Azimuth Averaging Tool option until GPP has read in some data.

GPP reads in ASCII (American Standard Code for Information Interchange) data files with white space or comma-separated columns that represent different channels of data. GPP also recognizes several specific types of data files and modifies its behavior to streamline their use. The code currently recognizes ADAMS[2], YawDyn[3], and GPP Merge Tool output files. When GPP reads in one of these special files, it does not need to ask you about headers and columns. GPP also knows how to parse out column headings so that the various tools can show you information based upon column name instead of column number. GPP classes files that it does not recognize as "generic."

Some tool options are such that you would rarely change them, so GPP saves the option settings in a permanent file. You can change these options from any of the high-level menus.

We wrote GPP in a modular fashion so that it would be easy to add new types of analyses as we have the time and inclination. This produced a set of tools that are generally independent of each other. What follows in the next three sections is a description of the user-selectable tools.

Analysis Tools

GPP's analysis tools perform the mathematical and engineering analyses for which we wrote GPP. The other tools "filter" the data or are system related.

The Azimuth Averaging Tool. You can bin your data against azimuth with this tool. It also produces the Fourier coefficients (converted to magnitude and phase) of the averaged data. You can compute these coefficients for as many as ten frequencies. This feature makes the Azimuth Averaging Tool superior to the Binning Tool for azimuth averaging. Figure 1 is an example of a plot generated using data from the Azimuth Averaging Tool.

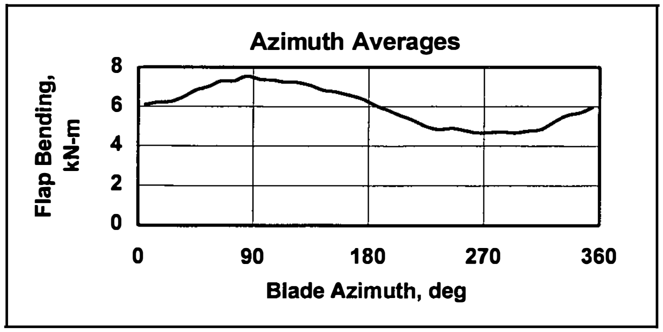

FIGURE 1. SAMPLE PLOT USING DATA GENERATED BY THE AZIMUTH AVERAGING TOOL.

The Binning Tool. This tool allows you to bin one or more columns of the data against an independent column. It is useful for plotting data such as power versus wind speed or lift coefficient versus angle of attack. Figure 2 is an example of a typical power curve using data generated by the Binning Tool.

The Histogramming Tool. You can generate histograms (or probability density functions) of your data with this tool. The Histogramming Tool writes two columns for each variable to the results file. The first of the two columns is the variable bins covering the range from the minimum to the maximum value for a given variable. The second column is the probability densities for those bins. The Histogramming Tool normalizes the densities so the area under them is unity. Figure 3 is an example of a plot generated using data from the Histogramming Tool.

The Least-Squares Fitting Tool. This tool allows you to fit your data with polynomials of up to the tenth order. The tool uses modified versions of the Singular Value Decomposition routines from *Numerical Recipes* (Press, 1990).

The tool generates a table of polynomial coefficients for each selected column. It also computes the $\mathcal{X}^2$ goodness of fit. It writes these tables to the results file.

We tested these routines using data generated by evaluating known polynomials. Results for polynomials greater than the third order did not fit the data well. Results for low orders were quite excellent. We may look for a better algorithm for this tool.

The PSD Tool. You can generate power spectral densities (PSDs) of your data with the PSD Tool. The PSD Tool uses routines derived from SPECFT, which is a program we got from the National Center for Atmospheric Research (NCAR). The SPECFT routines call another package called REALFT that NCAR acquired elsewhere. The Fast Fourier Transform (FFT) algorithm in REALFT is desirable in that it does not limit the number of input points to powers of two. We heavily edited the SPECFT routines we got from NCAR to eliminate unused features.

---

[2] ADAMS is a registered trademark of Mechanical Dynamics, Inc.

[3] YawDyn is a trademark of the National Renewable Energy Laboratory.

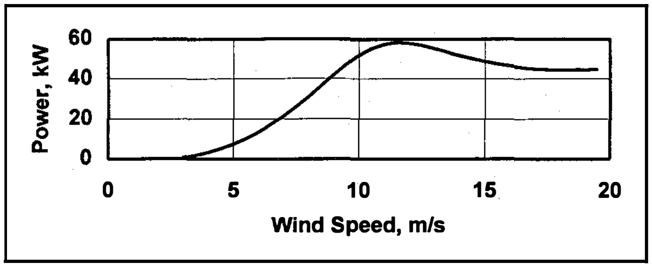

FIGURE 2. SAMPLE POWER CURVE USING DATA GENERATED BY THE BINNING TOOL.

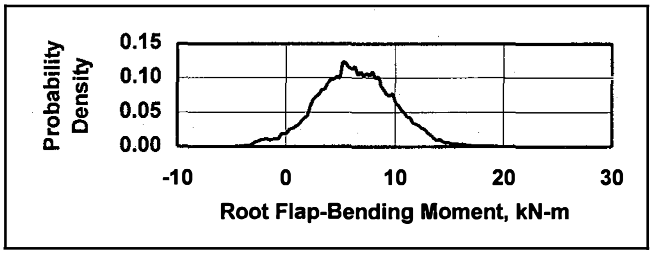

FIGURE 3. EXAMPLE OF A PLOT OF HISTOGRAMMING TOOL RESULTS.

The PSD Tool removes the means from each time series before generating the PSDs. It also tapers the ends of the data with a cosine rolloff and zero-fills the rest of the array that holds the data. You can optionally detrend the data with a straight line. You can set (or clear) this option with the Options Tool.

The default PSD technique uses band smoothing. Other PSD types are cosine (Hamming), triangular (Bartlett), and rectangular windows. Instead of window averaging, you can also choose logarithmic spacing. Initial tests indicate that there is very little difference in the results for different PSD types. The authors often use band smoothing so they can specify the number of output frequencies. You set the type of PSD technique with the Options Tool. Table 1 explains the benefits of each technique.

Figure 4 shows a sample PSD plot using data generated by the PSD Tool using band smoothing.

The Rainflow Cycle-Counting Tool. You can rainflow cycle-count your data with the Rainflow Cycle-Counting Tool. This tool implements the vector-based rainflow counting algorithm derived from Downing (1982) and incorporates a modification by Okamura (1979) to account for half cycles. The tool rainflow counts a time history as it occurs and identifies the same cycles as the two-pass algorithm that requires that the history be rearranged. After the tool identifies the cycles, it bins them to generate curves of cycles per the number of seconds specified for the count period versus the peak-to-peak cycle amplitudes. The tool counts only alternating (or range) cycles.

Figure 5 shows a sample plot of data generated by the Rainflow Cycle-Counting Tool. If you look at the two right-most data points, you will see that they have the same y-value. This is because this algorithm closes all incomplete cycles. The authors consider these artificially closed cycles to be very misleading and suggest that you eliminate them.

TABLE 1. TYPES OF PSDS AVAILABLE FOR GPP.

Band Smoothing	Band smoothing allows you to specify any number of output frequencies (up to #points/6). This allows you to reduce the volume of output and is a good choice if you have a lot of data.
Cosine Windows	Using Hamming-style cosine windows is the best choice for random signals. Use this type of PSD if you are unsure. It produces half as many output frequencies as there were data points.
Logarithmic Spacing	Logarithmic spacing of output frequencies will give you less resolution at high frequencies.
Rectangular Windows	Rectangular windows are a good choice if your data has discontinuities or singularities. It produces half as many output frequencies as there were data points.
Triangular Windows	The triangular (Bartlett) window gives good results for deterministic signals like sine waves. It produces half as many output frequencies as there were data points.

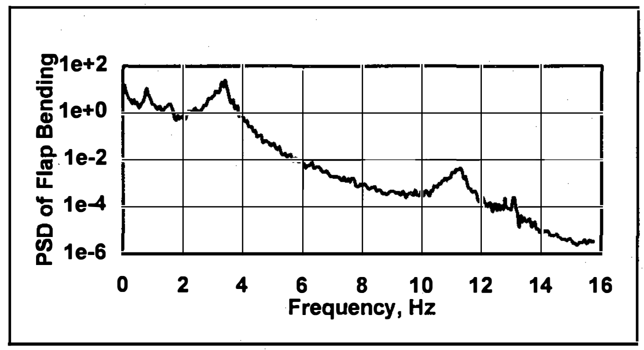

FIGURE 4. EXAMPLE PSD PLOT USING DATA GENERATED BY THE PSD TOOL.

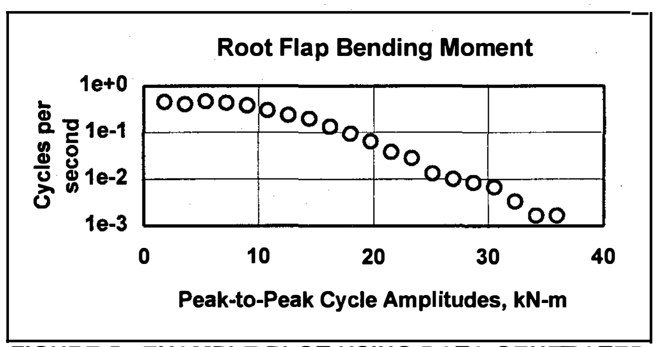

FIGURE 5. EXAMPLE PLOT USING DATA GENERATED BY THE RAINFLOW CYCLE-COUNTING TOOL.

The Statistics Tool. The Statistics Tool displays statistics of your data. These statistics include the following: minimum, mean, maximum, standard deviation, and skewness. They also include the data record numbers on which the minimums and maximums occurred. Figure 6 shows a sample statistics screen.

In the example, since the data came from a familiar file, the Choose Tool knew how to determine the column headings. Thus, the tool includes column descriptions in the output. If this had been a generic file, the first column in the table would have

Statistics for data from "adams.rq":

Parameter	Minimum (Index)	Average	Maximum (Index)	Std. Dev.	Skewness
Time	5.0 (1)	35.	65. (1921)	17.	-5.63E-07
AZIMUTH	-1.80E+02(1311)	0.50	1.80E+02(782)	1.04E+02	-5.89E-03
BLD_1_RF	2.6 (1)	6.7	11. (146)	2.7	0.13
BLD_1_RE	-4.3 (19)	1.9	8.1 (243)	4.2	2.12E-02
LSS_TOR	5.5 (19)	6.3	6.9 (25)	0.36	-0.18
POWER	-34. (25)	-31.	-27. (19)	1.8	0.18
YAW_MOM	-6.54E+03(17)	-5.89E+03	-5.46E+03(5)	1.39E+02	0.22
FA_THRST	-1.58E+02(796)	-7.10E-02	1.6 (8)	3.6	-43.
SS_THRST	-3.9 (796)	-8.06E-02	2.4 (609)	0.43	-1.6

FIGURE 6. AN EXAMPLE DISPLAY FROM THE STATISTICS TOOL.

been the column number instead. This is another example of how GPP changes its behavior for familiar file types.

When the Choose Tool reads in data, it also includes a kurtosis calculation and writes all of the statistics to a file. Here, GPP tells you where it got the data, when it generated the statistics, and includes the original header. Then, it lists the statistics for each column of data. GPP writes the statistics file with one more significant digit than for the screen display of the same data.

Filtering Tools

The Filtering Tool. The Filtering Tool allows you to eliminate unwanted information from your data. Currently, the Filtering Tool can average, decimate, or limit blocks or cycles of data. A future release may let you low-pass, high-pass, band-pass, or notch filter your data.

The Interpolation Tool. This tool allows you to interpolate the dependent columns of your data using a new set of independent values (a new column 1). You can read in the new independent values from a single-column file or specify a minimum, maximum, and step size. You may use either linear or cubic-spline interpolation. We use modified versions of the cubic-spline routines found in *Numerical Recipes* (Press, 1990). The second derivatives are zero at the end points.

Example:

Suppose you have two files with different time steps and want to merge them using the time history of the first of them. You can use the Merge Tool to copy the time column of the first file to a new file. Then, read in the second data file with the Choose Tool. Start the Interpolation Tool and tell it to get the independent series from the new, time-only file. The file created by the Interpolation Tool will be compatible with the first file so you can merge them together.

The Limit Tool. This subtool of the Filtering Tool is very powerful. You can specify which blocks of data you want to keep for later processing. Blocks can be of any length (even a single record) and you can specify them by number of records or amount of time. You can also specify block length as being a single rotor cycle. The Limit Tool detects cycle boundaries and does not assume the turbine is rotating at a constant rate.

The Limit Tool asks you for a list of limiting criteria. Each criterion consists of a column, a minimum, a maximum, and a type. Types can be the minimum value, the mean, the maximum value, the standard deviation, or every value of a candidate block. Entering the type as a negative tells the Limit Tool to pass only those blocks for which the statistic lies outside the specified range. Table 2 shows the values you enter for the different types of limiting factors.

Examples:

Suppose you want to exclude all rotor cycles where the wind speed (channel 5) exceeds 15 mps even once over the cycle. Using 500 as an arbitrarily large number, you would limit by cycles and specify the limiting criterion as:

```
5,15,500,-3
```

Suppose you want to find all one-minute time blocks where the wind direction (channel 6) changed little. For this, you should limit by blocks of 60 seconds and may want to specify that all values in the block lie between –10 and +10 degrees:

```
6,-10,10,5
```

Suppose the azimuth (channel 2) goes from –180 to +180 degrees. If you want to include all records that lie outside the tower shadow, you would limit by individual records and specify the limiting criterion as:

```
2,-170,170,5
```

The Merge Tool. You can use this tool to eliminate unwanted columns from your data set or to merge columns from multiple files. After using the Merge Tool, you can analyze the merged data, write them to a Merge Tool file, or both. You can read merged files later using the Choose Tool and analyze them like other files. GPP recognizes Merge Tool files and takes advantage of their structure as it does for YawDyn or ADAMS files.

When you merge data, the Merge Tool creates a header that gives you a history of merging operations. This header tells you which files you merged and when you merged them. The column labels indicate which file they came from and what were the original columns. When you manipulate the data, the tools you use update the header to record your changes. Figure 7 shows an example header.

The Trim Tool. The Trim Tool eliminates rows from your data. You can specify a block of rows by row numbers. If the first column has monotonically increasing values (like time), you can also eliminate rows by values.

TABLE 2. TYPES OF LIMITING FACTORS.

1	The minimum value in each block.
2	The mean of each block.
3	The maximum value in each block.
4	The standard deviation of each block.
5	All values in each block.

The Units-Conversion Tool. With the Units-Conversion Tool, you select columns for which you want to rescale the data using scales and offsets.

We built many automatic conversions into the Units-Conversion Tool. When asked for the scale and offset, you can enter a ? to see the list of those available. For example, instead of entering 0.55555,32 to convert from Fahrenheit to Celsius, you can enter F2C. This way, you do not need to look up conversion factors. Figure 8 shows the help screen for these conversions.

System Tools

The Choose Tool. The Choose Tool allows you to choose a file to read into memory. You can not read files with more than 20 header lines, 20 columns, or 20,000 rows of data[4]. The current version of the Choose Tool can read only ASCII files. We may add the ability to read binary files in the future. After reading in the data, the Choose Tool will compute statistics.

The Divide Tool. You can divide an ADAMS output file with the Divide Tool. It will create a new file for each ADAMS request in the file. You can read in these new files using the Choose Tool to analyze them.

The Escape Tool. You can temporarily escape to the operating system (for example, DOS or UNIX) with this tool. It allows you to look around and perform other commands without having to leave GPP.

The Help Tool. You can get on-line help for GPP. You will see a new menu of help options from which you can repeatedly ask for help on different subjects.

The Options Tool. The Options Tool allows you to review or change program options. If you do not like the defaults, you can change them for just the current run or for all future GPP sessions. GPP saves permanent changes to the options in a user file.

Options currently available for GPP include strings that identify ADAMS or YawDyn files. Another option is the "dot display rate." This determines how many operations elapse before GPP displays a dot on the screen to let you know that it is processing data. Another option determines if GPP prompts you before you quit GPP. Additional options are for generating PSDs. See above for information on the PSD Tool options. There is also an option to set the count period for the Rainflow Cycle-Counting Tool. Figure 9 is an example of the Options Menu.

[4] You can change these limits by modifying parameter statements in an include file. You must then recompile GPP to put them into effect.

```
GPP Merge Tool file from "adams.rq"; column 2 converted by
       "DEG2RAD".
Generated by GPP on 31-May-94 at 12:40:07; 31-May-94 at 13:04:10.
1-Time  1-AZIMUTH   1-BLD_1_RF  1-BLD_1_RE  1-LSS_TOR   1-POWER
```

FIGURE 7. EXAMPLE OF A MERGE TOOL FILE HEADER.

The Writing Tool. With the Writing Tool, you can write out the results of a merge to a file for processing later.

REQUIREMENTS FOR RUNNING GPP

Most modern PCs can run GPP. Unmodified, GPP needs about 4MB of RAM and disk storage. Tests on a 33MHz 486 PC indicated that the program is slow to load, but it is still quite usable. The 60 MHz Pentium PC used to port GPP to the PC runs it quite easily.

To modify GPP on a PC, you will need either Lahey's F77L-EM/32, Microway's NDP FORTRAN, or another protected-mode compiler. If you use a different compiler, you will need to modify some of the system-specific routines to be compatible with it. You will also need to modify some of the batch files and/or makefiles used to compile the code.

One of our primary uses for GPP is to post process ADAMS output. So, we made sure it could run on PCs that run ADAMS and that users can compile it with the NDP FORTRAN compiler needed for the AeroDyn[5] routines we link with ADAMS.

Almost any UNIX workstation can run GPP with few or no modifications to the code. If you want to modify GPP or run it on something other than a MIPS R4400-based SGI system or Sun SPARCstation 10, you will need a FORTRAN compiler. We developed GPP on a 64-bit, R4000-based, SGI Indigo^2. GPP is very fast on the Indigo^2 and on the Sun SPARCstation 10.

CONCLUSIONS

In tests at NREL, GPP has proved to be a very powerful tool for analyzing wind turbine data. GPP is very fast on modest PCs that are commonly available today. It is very easy to learn and use. It traps input errors to prevent you from wasting a lot of effort. We wrote the program in such a way as to make it easy to add new features and to port it to different compilers and operating systems. We expect that GPP will serve a significant role in wind energy research and industry.

We hope the easy-to-use features of GPP will allow people to use the code with little or no support. We will send copies of the GPP User's Guide (Buhl, 1994) to those who request it with the understanding that it is for evaluation purposes only. The guide tells you how to obtain the code from our servers at NREL. NREL will therefore not need to repeatedly send out floppies as the code continually evolves.

FUTURE WORK

We plan to add more features to GPP as time permits. We intend to add data files generated by NREL's ADAS (Advanced Data Acquisition System) units to the list of "familiar" files. We

[5] AeroDyn is a trademark of the National Renewable Energy Laboratory.

```
Units conversion is done with the following formula:
    Xnew = Scale*( Xold - Offset )

For a shortcut, some conversions have been set up for you.
Enter the mnemonic for the old units, a "2", then the
mnemonic for the new units.  Here's a list of conversions:

 C   - degrees Celsius    KG  - kilograms        MPS - meters/second
 DEG - degrees            KPA - kiloPascals      N   - newtons
 DPS - degrees/second     LBF - pounds force     NM  - newton-meters
 F   - degrees Fahrenheit LBM - pounds mass      PSI - pounds/inch^2
 FP  - foot-pounds        M   - meters           RAD - radians
 FPS - feet/second        MB  - millibars        RPM - revolutions/minute
 FT  - feet               MPH - miles/hour       RPS - radians/second

For example, if you enter "DEG2RAD", GPP will convert all
selected columns from degrees to radians.

Enter the Scale and Offset (? for help; 0,0 aborts) [1,0] >
```

FIGURE 8. A HELP SCREEN FOR THE UNITS-CONVERSION TOOL.

```
Current option settings are:

 A - ADAMS identifier          = "1ADAMS"
 D - Dot display rate          = 2000
 L - Linear detrending for PSDs = Yes
 P - Prompt before quitting    = No
 R - Rainflow count period (sec) = 1
 T - Type of PSD               = "Band Smoothing"
 Y - YawDyn identifier         = "    Time   Azimuth"

 Q - Quit changing options

Enter option you wish to change (A,D,L,P,T,Y,=Q) >
```

FIGURE 9. EXAMPLE OPTIONS MENU.

are also considering the pros and cons of adding binary input/output. We may add computation of standard deviations for the bins in azimuth averaging and binning. We would like to add several different types of filtering such as low-pass filtering. We may add more automatic conversion factors to the Units-Conversion Tool. We also want to add calculated channels. If we can find a least-squares algorithm that works well for the higher orders, we will replace the *Numerical Recipes* (Press, 1990) routines.

If future compiler technology permits, we hope to dynamically allocate the storage arrays at run time. This requires the Fortran-90 ALLOCATE feature. So far, the only compiler we have that supports this feature is Lahey's F77L-EM/32. When our other compilers support it, we will modify the program. This modification of the program will produce smaller executable files (hence easier to copy and load into memory) and eliminates the need to recompile the code for different array sizes. The user will set array sizes with the Options Tool.

If we can find a good, public-domain graphics library that is available on many platforms, we may someday add plotting capabilities to GPP. We do not intend to replace sophisticated commercial packages, but to allow users to get a quick view of their data.

ACKNOWLEDGMENTS

The authors would like to thank all the people who helped make GPP a reality. We thank Bob Lackman of NCAR for writing the SPECFT routines. The SPECFT routines definitely make our PSD Tool much more versatile than the routines we originally got out of *Numerical Recipes* (Press, 1990).

Alan Wright came up with the requirements for the Azimuth Averaging Tool and helped debug the Fourier coefficient part of the code. Alan also found other bugs while using the code. We appreciate Alan's patience and encouraging words.

Dave Laino, beta-tester extradordinaire, found quite a few bugs and made many suggestions to improve the usability of GPP. Sandy Butterfield helped us obtain funding for this effort. George Scott taught us a few UNIX tricks and helped us find the causes of some of the bugs. Chris Courtright and Vallerie Shelton helped keep the computers behaving properly.

Mechanical Dynamics, Inc. let us use their HP workstation to port it to that platform while attending an ADAMS/WT training class at their site. The HP has a very different compiler and it forced us to make the code even more portable than it was before.

We wrote this paper to document work done at the National Renewable Energy Laboratory in support of the U.S. Department of Energy under contract number DE-AC02-83CH10093.

REFERENCES

Buhl, Jr., M. L., 1994, *GPP User's Guide*, NREL/TP-442-7111, National Renewable Energy Laboratory, Golden.

Downing, S. D., Socie, D. F., 1982, "Simple Rainflow Counting Algorithms," *International Journal of Fatigue*, Vol. 4, pp. 31–40.

Okamura, H., Sakai, S., Susuki, I., 1979, "Cumulative Fatigue Damage Under Random Loads," *Fatigue Engineering, Materials, and Structures, Vol. 1*, pp. 409–419.

Press, W. J.; Flannery, B. P.; Teukolsky, S. A.; Vetterling, W. T., 1990, *Numerical Recipes, The Art of Scientific Computing (FORTRAN Version),* Cambridge University Press, Cambridge, pp. 52–64, 86–89, 515–520.