AW-System - An Interactive Environment for the Evaluation of Large Time Series

D. Herter, N.O. Chrisander, Chr. Gossweiler
ETH Zürich, Switzerland

Introduction

The data processing program "AW-System" concerns the handling and evaluation of large data sets produced by aerodynamic flow field measurement systems. AW stands for the german "Auswertung", meaning evaluation, and was developed as a part of the project "Fast Response Aerodynamic Probe Measurement System" (see also the contributions by [Gossweiler 92], [Humm 92] and [Kupferschmied 92] in these proceedings). It is designed to meet the demand for a universal and flexible software environment capable of handling large data sets originating from different sources as aerodynamic probes, wall pressure sensors, hot wire probes and the like. AW-System is written in PASCAL and FORTRAN making considerable use of the VMS-specific system and library routines. AW-System can thus be installed on any DEC VAX running the VMS operating system but cannot be transferred to other computers running e.g. UNIX.
This paper describes the measuring system starting with the data acquisition and shows the course of the data evaluation procedure laying special emphasis on the features of the implemented system.

Data Acquisition

The electric signals from the sensors of the aerodynamic high frequency response probe are amplified and recorded in a specially developed eight channel data acquisition system [Hirt 87] including a low-pass linear phase filter (anti-aliasing filter) consisting of an eight-pole Butterworth and an allpass filter for phase correction. This system can store up to 32 MByte of data in its internal RAM and has a maximum sampling rate of 200 kHz when all channels are used. Its Analog-to-Digital Converters (ADC) have a resolution of 12 bits. This allows the usage of the remaining 4 bits of the 2-byte-words for the storage of additional information. Two bits may be used as *event bits* to store trigger data e.g. to indicate when a certain blade has passed the probe; the other two bits contain the channel number (Fig. 1).

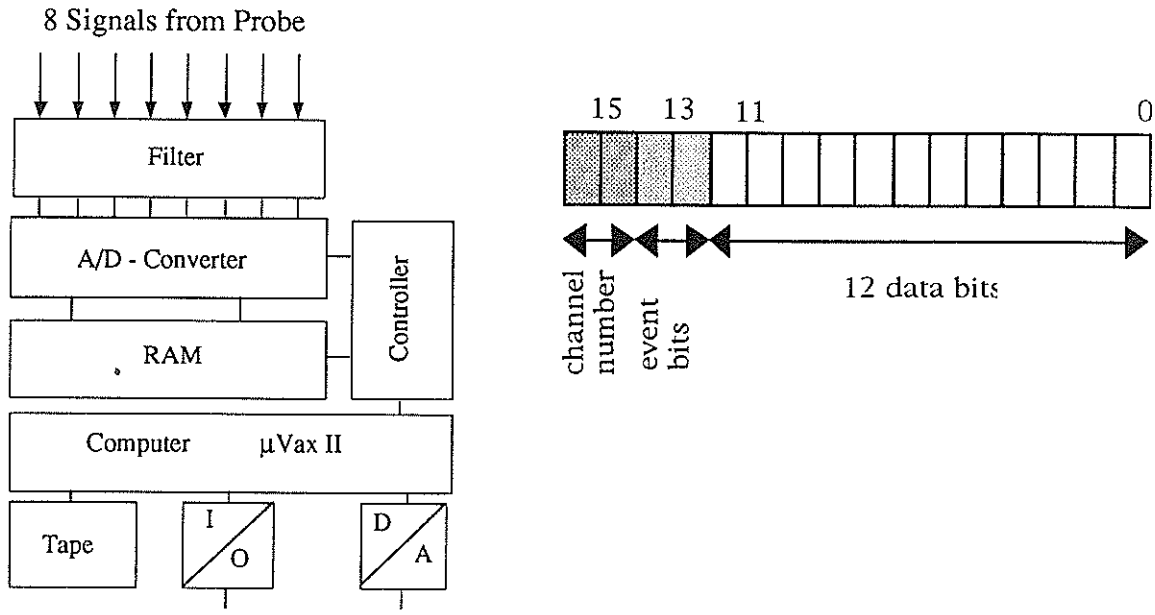


Figure 1: Data acquisition system and format of recorded raw data

Data Evaluation Procedure

As a first step the raw data are read in from disk. After having separated the data from the event bits they are calibrated according to the amplifier setting. The so obtained voltages are then converted into physical quantities by means of probe specific calibration files. After a possible digital filtering the actual data analysis is performed: this includes statistics, spectra, correlations etc. (Fig. 2). As a last step the results are written to disk.
After the first run the results usually aren't satisfying and at least part of the procedure has to be repeated. With conventional data evaluation systems this means a lot of time consuming Input/Output operations, especially if intermediate results have been written to disk because these files are usually very large. Moreover we even might have to edit (compile and link) the programs to adapt them to the specific problem.
With AW-System these problems can be avoided: only at the beginning of a session raw data are read in and results are written to disk at the end. All the processing in between is done on the computer's *virtual memory* so that all data can be accessed at any time (Fig. 3). Intermediate results can be checked and corrected immediately if necessary. The possibility of visual checks allows an easy examination of the results.

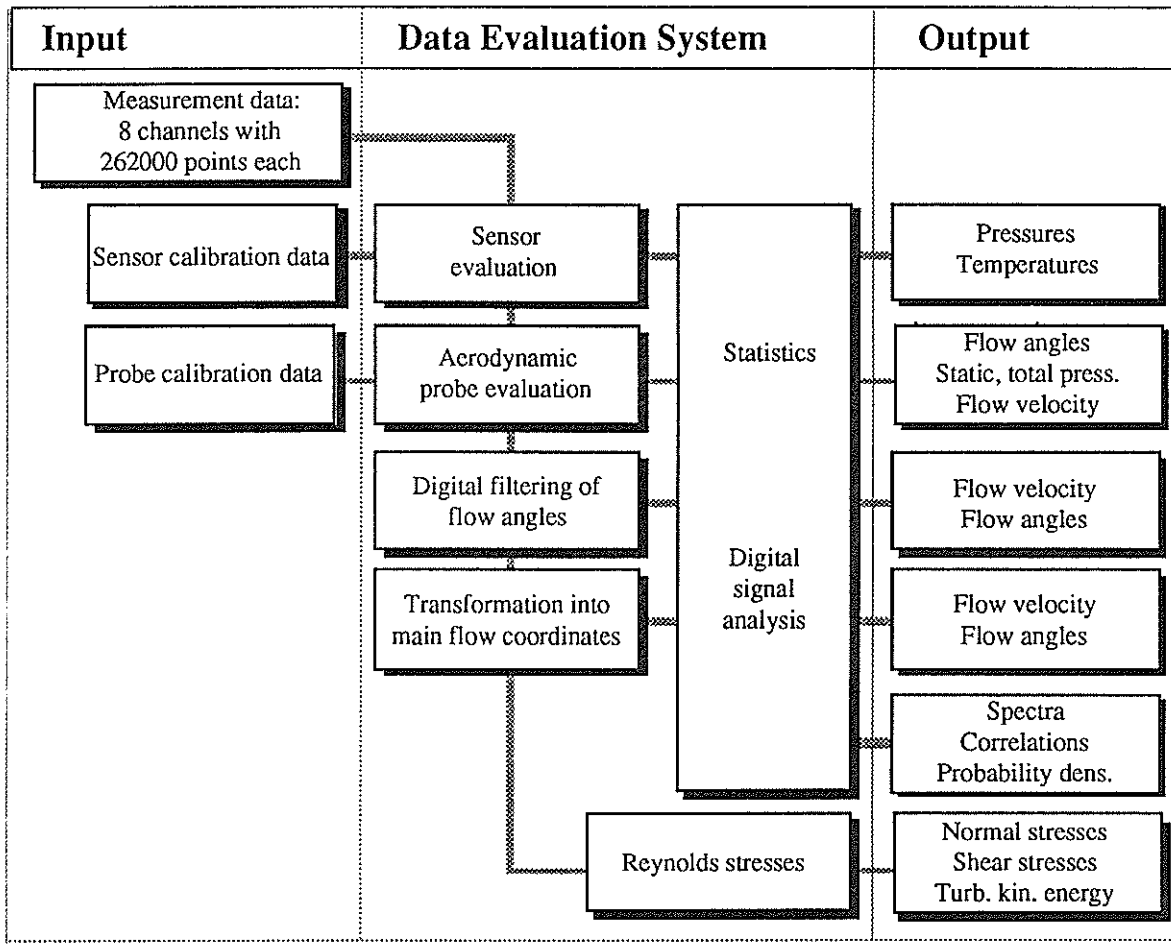


Figure 2: Example of data processing for the 4-sensor fast response probe

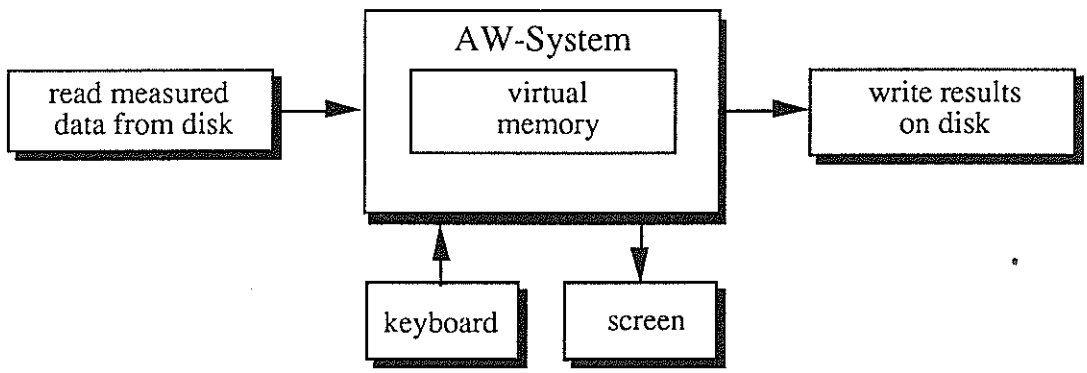


Figure 3: Schematic view of the data evaluation with the AW-System

Main Features of the AW-System

Interactivity and flexibility: The users can process data in any order they like. After each step they can decide which action to take next. If a result is not satisfying the users can repeat these steps e.g. try a different digital filter.

User controlled memory management: To minimize disk I/O the program uses the computer's virtual memory to store data. The users can *dynamically* allocate arrays in arbitrary sizes, an option the usual programming languages do not provide. These arrays containing the data reside in virtual memory and can be accessed very fast. When data are no longer needed the memory used by this array can be returned to the operating system for further use. This means that the same virtual address region can be reused by other arrays during the same session, if necessary with another element data type.

Command Line Interpreter: The users control the evaluation with a line-oriented command language similar to DCL (Digital Command Language). A command string contains a verb that acts on parameters which represent input and output structures. Possible keywords and qualifiers specify the command more accurately or define some minor deviations from the main operation. Often used commands can be abbreviated by symbols.

Example: `ALLOCATE MATRIX /I2/COLUMN=4/ROW=262000 INPUTDATA`
verb keyword /qualifiers parameter

This command allocates an array of Integer*2 data type with four columns and 262,000 rows. `INPUTDATA` is the logical name of the array by which it can be referenced in further commands.

Single Line Editor: allows to recall and modify the last 20 commands.

Command files: Commands can be grouped into command files e.g. for procedures that occur often or to run the program as a batch job.
The availability of some simple control structures (labels, IF, GOTO) together with the possibility to evaluate arbitrary expressions (both boolean and numerical) changes the command language into a programming tool that allows the users the construction of their own algorithms and methods.

Help: The whole program with all commands is documented in a built-in help library.

Subprocesses: There is also the possibility to open a subprocess without having to leave the program. This feature can be used e.g. to design a new digital filter or to modify calibration files.

Condition Handler: This module reacts on exception conditions, usually errors, occuring during the execution of a command procedure. These exceptions are then signaled to the users, allowing the localization and correction of the error. Fatal errors do not cause the abortion of the program; the processing of the command is terminated in an orderly fashion so that no data are lost.

Journal and listing files: An evaluation session may be documented on a journal file and a listing file. Every command entered on the interactive level is written into the journal file. Extractions of the journal file can be directly used as new command files. All input to the program and all output on the screen is written to the listing file. This file contains a record of all work done during a session.

Extendability: It is possible to implement new commands (like a special statistical operation) or extend existing ones (like a further filter function) without much effort or particular influence on the existing environment. Problem-specific routines written by the users can also be included.

Commands

There are about 60 commands available that can be grouped into four categories:

System Commands: are used for file and memory management i.e. for allocating arrays, open files etc.

Utilities: include routines like `COPY` and `TRANSPOSE` for the basic handling of data and event vectors. Further commands perform arithmetics and evaluate logical operations on event vectors. Test functions give the possibility to check routines and with the `PLOT` command data can be visualized.

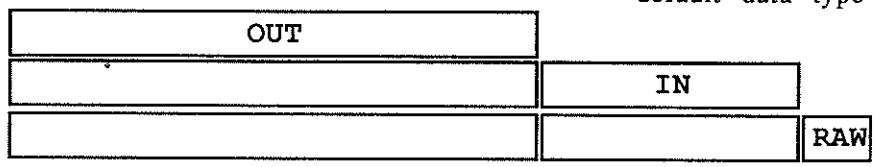
Statistics and Digital Signal Analysis: Contains commands for statistics, spectra, correlations, over and undersampling, phase shift, Fast Fourier Transforms, digital filtering, triggering, conditional averaging and histograms.

User: contains the user and problem specific commands, e.g. commands to process data measured with different probes.

Example of a Command File with Schematic View of the Virtual Memory Map

```
ALLOCATE MATRIX OUT[262000,18]
ALLOCATE MATRIX IN[262000,8]
ALLOCATE MATRIX/I2 RAW[262000,4]
```

allocate 3 data arrays with 262,000 rows: OUT with 18 columns for the physical quantities, IN with 8 col. for the voltages and RAW with 4 col. for the raw data. IN and OUT have default data type Real*4

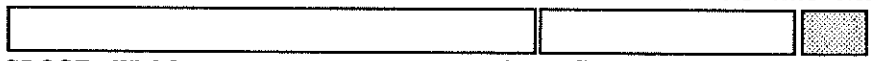



```
OPEN/READ/DATA HLOG DISK97:EXP1.DAT
```

open first data file on DISK97 named EXP1.DAT and give it the logical name HLOG

```
READ HLOG RAW/NBLOCK=4094
```

read 4094 blocks from HLOG containing the first 4 channels into RAW

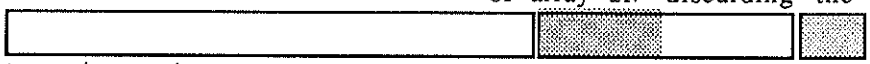


```
CLOSE HLOG
```

close first data file

```
BITFILTER/READ/GAIN=5000/NELEM=262000 RAW IN
```

separate 262,000 data points from the event bits, calibrate them according to the amplifier setting of +/-5V and store the result in the first (default) 4 channels of array IN discarding the event bits

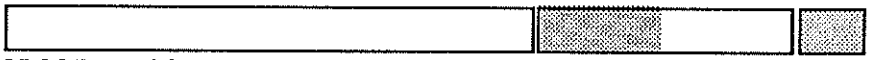


```
OPEN/READ/DATA HLOG DISK97:EXP2.DAT
```

open second data file with the same logical name HLOG

```
READ HLOG RAW/NBLOCK=4094
```

read the data from second file containing the other 4 channels into RAW thus overwriting the old raw data

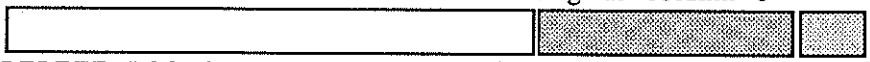


```
CLOSE HLOG
```

close second data file

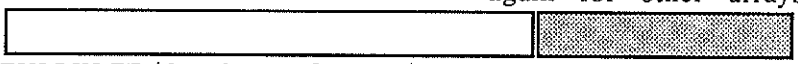
```
BITFILTER/READ/GAIN=5000/NELEM=262000 RAW IN/COL=5
```

convert raw data of second 4 channels into voltages and store the result in array IN starting at column 5



```
DELETE LOGICAL RAW
```

delete array RAW; this space can now be used again for other arrays



```
EVALUATE/OFFSET=T2.EXP/PROBE=Z25MK2DPG.CAL IN OUT
```

evaluate 18 physical quantities from the 8 voltages using two calibration files containing sensor and aerodynamical data and store the result in array OUT

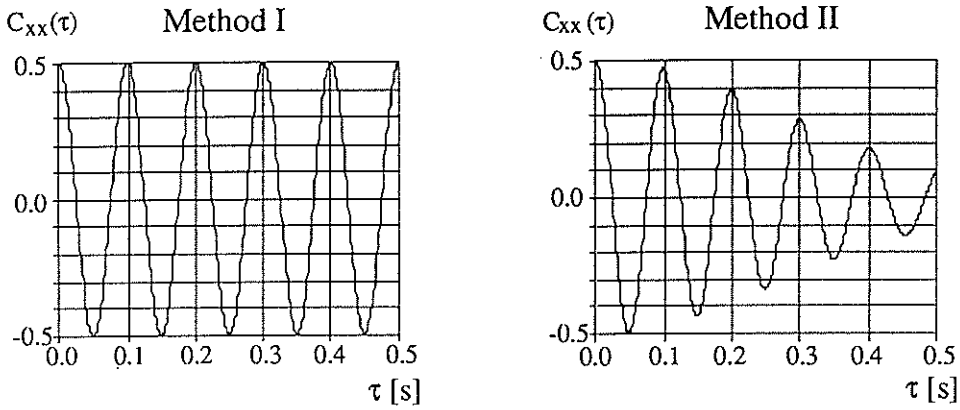
```
DELETE LOGICAL IN
```

delete array IN with the voltages if the calibration was successful

Performance

To obtain a good precision of statistical estimates the data vectors have at least 262,000 elements. Depending on the extent of the data evaluation several thousand operations for each element of the time series are necessary. This requires a fast computer. Our group's standard data processing machine is a VAX 9000-420 with two CPUs, one vector unit and 256 MByte RAM. This machine has a maximum performance of 125 MFlops (millions of floating point operations per second). The accompanying solid state disk (ESE20) is about a factor of 25 faster than usual hard disks.

As an example to demonstrate the need for high computer performance the crosscorrelation function of two coherent sine functions with 100,000 data points each is examined by using two different algorithms. Method I uses a standard math library function for computing the sample crosscorrelation function of two stationary time series without making any approximations. With a lag of 1000 this routine finishes after 167 seconds of CPU time. The time efficient method II uses a routine based on Fast Fourier Transforms. This algorithm needs only 1.2 seconds of CPU time but yields an inaccurate result.



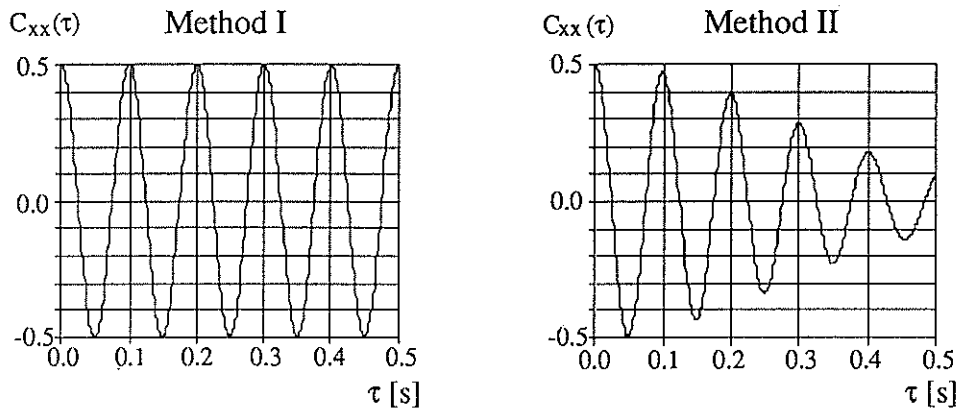
Conclusions

The AW-System is an interactive and powerful environment for evaluating large time series. Because of its modular construction the program can easily be adapted to new needs and by including user-specific routines it can evaluate data measured by different probes and in different experimental conditions.

Performance

To obtain a good precision of statistical estimates the data vectors have at least 262,000 elements. Depending on the extent of the data evaluation several thousand operations for each element of the time series are necessary. This requires a fast computer. Our group's standard data processing machine is a VAX 9000-420 with two CPUs, one vector unit and 256 MByte RAM. This machine has a maximum performance of 125 MFlops (millions of floating point operations per second). The accompanying solid state disk (ESE20) is about a factor of 25 faster than usual hard disks.

As an example to demonstrate the need for high computer performance the crosscorrelation function of two coherent sine functions with 100,000 data points each is examined by using two different algorithms. Method I uses a standard math library function for computing the sample crosscorrelation function of two stationary time series without making any approximations. With a lag of 1000 this routine finishes after 167 seconds of CPU time. The time efficient method II uses a routine based on Fast Fourier Transforms. This algorithm needs only 1.2 seconds of CPU time but yields an inaccurate result.



Conclusions

The AW-System is an interactive and powerful environment for evaluating large time series. Because of its modular construction the program can easily be adapted to new needs and by including user-specific routines it can evaluate data measured by different probes and in different experimental conditions.

Acknowledgements

The authors wish to thank the Swiss National Science Foundation for supporting this work.

References

Ch. Gossweiler, D. Herter, P. Kupferschmied
Fast Response Aerodynamic Probe Measurements in a Turbulent Pipe Flow
Proc. XIth Symposium on Measuring Techniques for Transonic and Supersonic Flows in Cascades and Turbomachines, München 1992

P. Hirt
Betriebsanleitung CS RBX 1 800 kHz Datalogger
CDA Electronics, Birr (Switzerland) 1987

H. J. Humm, J. I. Verdegaal
Aerodynamic Design Criteria for Fast Response Probes
Proc. XIth Symposium on Measuring Techniques for Transonic and Supersonic Flows in Cascades and Turbomachines, München 1992

P. Kupferschmied, Ch. Gossweiler
Calibration, Modeling and Data Evaluation Procedures for Aerodynamic Multihole Pressure Probe Measurements on the Example of a Four Hole Probe
Proc. XIth Symposium on Measuring Techniques for Transonic and Supersonic Flows in Cascades and Turbomachines, München 1992