DEDICATED MICROCOMPUTERS AND PERSONAL COMPUTERS AS EXPERIMENTAL TOOLS IN THE PHYSICS LABORATORY

A. VAN DEN BOS

Faculty of Applied Physics
Delft University of Technology

Received March 5, 1987

Abstract

Modern microcomputers and personal computers increasingly provide the user with facilities until recently found only in mainframes. As a result a new category of small scale, but computationally and numerically demanding laboratory experiments and instruments have become realizable on the spot. Three examples of pertinent experiments/instruments designed at the Signals and Systems Department, Faculty of Physics, Delft University are discussed.

Introduction

Until recently observations made in numerically and computationally more demanding physics laboratory experiments had to be transported from the measurement site to a suitable mainframe located elsewhere. This transport could take place via magnetic tape, disk or data communication link. In any case this implied off-line processing and a relatively complicated experimental organization. The compelling reason for use of a mainframe was, of course, its unsurpassed computational speed, the usual availability of numerical high-quality software libraries, the presence of mass memory, and the presence of expensive, high-performance peripherals. From the experimenter's viewpoint the price for these indispensable facilities was high. The inherent off-line character virtually precluded interactive operation. As a result experimentation was time consuming, especially in the case of undetected failure. Moreover, in most institutions the mainframe constitutes the central computing facility. Lengthy jobs are therefore consigned to the off-hours, adding to the delay between the producing and processing of the observations.

During recent years a gradual shift of a number of these mainframe and larger minicomputer facilities towards microcomputers and personal computers could already be observed. Operating systems, compilers and editors became available substantially simplifying the application of microcomputers in physical experiments. A simultaneous and closely connected development in hardware was the advent of relatively inexpensive floppy disk drives and multi-

megabyte hard disk drives. These make time consuming cassette tape loading obsolete while their massive memory capacity enables the experimenter to incoporate existing software packages and libraries in his computer. Furthermore, the introduction of cheap matrix printers and terminals removed a long-standing obstacle to interactive operation and documentation in the laboratory.

These facilities, although impressive, are still not completely statisfactory for many laboratory purposes for two reasons. In the first place, the usual word length of 16 bits seriously limits the amount of directly addressable random access memory. This slows down the operation and complicates the program organization. In particular this is the case if interrupt facilities have to be used as is usual in many laboratory applications. Secondly, for computationally demanding signal processing activities computation speed often still proves insufficient. Fortunately, a new generation of personal computers (PC) will soon be commercially available that will, at least partly, remove these difficulties. These desk top personal computers are so much faster than the present generation that a whole category of computationally demanding numerical problems comes, quite abruptly, within reach. Moreover, through the doubling of the current word length, the amount of directly addressable random access memory increases in no small measure.

The purpose of this paper is to illustrate and discuss this rapid evolution using three examples of dedicated microprocessor based instruments realized, or being realized at the Faculty of Physics of Delft University since the end of 1983. A further purpose is to see how these devices could benefit from the developments in hardware and software that have taken place after their completion and/or from the developments to be expected in the near future.

Examples

Example 1: A microcomputer based spectral analyzer using parametric techniques

The spectral analyzer concerned was completed at the end of 1983. It is described in [1]. Its purpose is real-time estimation of slowly time-varying power density spectra. Conventional digital spectral analyzers use fast Fourier techniques. This analyzer uses a parameter estimation approach. Its input sequence is modelled as a white sequence that has passed a linear, all-pole, discrete-time filter. Such a process is usually called autoregressive. The analyzer estimates the coefficients of the hypothetical filter from its input. The computation of the spectrum from these coefficients is then straightforward.

The conventional way of estimating autoregressive coefficients has the form of a linear least squares problem. This problem can conveniently be brought in recursive form. For the problem considered this implies that the current coefficient estimates are updated each time a new input sample is taken. So, past samples need not be stored. Furthermore, if the analyzer input is stationary, the coefficient estimates converge, and so does the spectrum. Linear least squares also offers the opportunity to incorporate some kind of forgetting. This means that recent input samples have more influence on the estimation result than past ones. In the analyzer exponential forgetting is used: weights in the least squares criterion decrease exponentially with the time elapsed. This enables the analyzer to track a slowly time-varying spectrum. Note that a record of the spectrum can be kept in the form of a record of the, usually few, autoregressive coefficients.

Linear least squares is notorious for ill-conditioning. Therefore, the numerically stable Householder method is used instead of a conventional method as Gaussian elimination. Before the spectrum tracking starts, first the order of the autoregressive filter is selected. This is done by recusively fitting filters of increasing order to a number of input samples by the Levinson-Durbin method. The order of the best fitting filter is selected. This selected order remains fixed during the subsequent tracking operation. For a description of recursive least squares, exponential forgetting and the Levinson-Durbin recursion, the interested reader is referred to [2].

To increase speed the analyzer employs a preprocessor and a main processor. Each contains a Digital Equipment Corporation (DEC) LSI-11/02 central processing unit (CPU). Furthermore, the preprocessor contains an analog-to-digital converter and standard DEC random access memory (RAM), timing and input-output (IO) modules. It computes the autoregressive coefficients. The main processor also contains RAM, timing and IO modules, and a DEC compatible video module. It computes the spectrum from the coefficients received from the preprocessor, displays and documents it. For that purpose it is interfaced to a terminal, one or two monitors, and a printer as hardcopy facility. The programs are mainly written in FORTRAN IV, are compiled elsewhere, and are loaded from cassette DEC tape.

The main performance characteristics of the analyzer may be summarized as follows: spectra represented by a 2nd, 4th and 8th order filter allow a maximum sampling frequency of 70 Hz, 30 Hz and 10 Hz, respectively.

Comment – From the performance characteristics it is clear that for many practical applications the bandwidth of the analyzer is too small. The lowcost DEC LSI-11/23 and LSI-11/73, successors to the LSI-11/02, further increase the bandwidth by a factor 2.5 and 12, respectively. This is still too slow for many interesting applications. In particular, audio applications would require a further increase of the bandwidth by a factor of 10.

FORTRAN compilation elsewhere and subsequent tape loading can nowadays be avoided through the advent of low-cost floppy or hard disk drives, and disk resident compilers. This would make the analyzer considerably more self-contained and attractive to the user.

Finally, all FORTRAN programs have been written by the designer. Presently, this programming could probably substantially be reduced by using a numerical library for microcomputers as the NAG PC50 [3] and signal processing/parameter estimation software available [4].

Example 2: A generator for random numbers having both a specified power spectral density and probability density

The random number generator to be described here was completed at the end of 1983 (see [5]). The purpose of the generator is to produce a sequence of random numbers having both a specified power spectral density and a specified marginal amplitude probability density. These sequences are useful in simulations, including Monte Carlo experiments.

Computer generation of uncorrelated normal numbers is a straightforward procedure. These numbers can next be transformed into a sequence having almost any desired power spectrum, or equivalently covariance function, by applying them to a suitable linear discrete-time filter having a selected frequency characteristic. Then the response sequence of the filter is still normal due to filter linearity. This correlated normal sequence can next be given any selected marginal probability density by applying it to a suitable memoryless nonlinear device. Unfortunately, this will alter the spectrum. The solution is to establish first the relation between the covariance of the sequence at the input and that of the sequence of the output of the nonlinear device. This can be done using Price's theorem [6]. By this relation the covariance at the input follows from the specified covariance at the output. The resulting input covariance can then be realized by proper choice of the linear filter preceding the nonlinear device.

The procedure used for computation of the linear filter coefficients and those of a polynomial representing the nonlinearity is as follows. The user specifies the desired power spectrum and probability density in graphical or analytical form. These specifications form the input to a mainframe program for the coefficient computation. Since this program has to solve demanding numerical problems as Chebyshev and least squares approximation, it extensively exploits routines from the well-known NAG mainframe library. The resulting coefficients are to be introduced into the generator.

The hardware of the generator consists of a DEC LSI-11/02 CPU, a DEC multifunction board with IO and RAM, a digital-to-analog converter (DAC)

and a specially designed high-speed maximum length binary sequence (MLBS) generator. The latter has been incorporated to increase the operating speed. The required normal, uncorrelated numbers are obtained by adding twelve uniformly distributed numbers produced by the MLBS generator at a very high rate. The required coefficients of the linear filter and those of the polynomial are introduced into the generator via a terminal, as is the frequency with which the desired sequence is produced. Finally, both a digital output and an analog one via the DAC are available.

The numerical procedure followed by the generator is simple. The uncorrelated normal numbers are applied to the linear filter. Each time a response value is produced the polynomial is evaluated for it. To save computation time this is done by Horner's rule. The result of the polynomial evaluation is the generator output.

Roughly speaking the maximum (production) frequency is between 100 Hz and 500 Hz, depending on the desired characteristics.

Comment — For a number of applications the maximum frequency of the generator is still too low. Examples are audio applications. Use of DEC LSI-11/73 instead of the LSI-11/02 would increase this frequency by a factor of 10. It is clear that through a relatively modest further increase of the computation speed, to be certainly achieved by microcomputers in the near future, the whole audio range will be covered.

A further remark concerns the lack of self-sufficiency of the generator: the computation of the generator coefficients is carried out on a mainframe elsewhere. It is clear that this problem can nowadays be solved by incorporating in the generator a disk drive and using a numerical library as the NAG PC50 library mentioned above.

Example 3: A dedicated computer for minimization of functions of many variables with many relative minima

This computer has been constructed a year ago. Its purpose is the minimization of functions of many variables with many relative minima. In this case conventional gradient methods fail. They usually get stuck in a relative minimum near the starting point. The proposed computer uses an alternative method called *simulated annealing* [7]. This method is based on ideas used in statistical mechanics simulations. To attain the absolute minimum of the energy, the co-called ground-state, these simulations are started at a relatively high temperature. Then the statistical behavior of the system at that temperature is simulated during a sufficiently large number of time steps. The temperature is subsequently lowered, the simulation repeated, and so on. If the decrease of the temperature is sufficiently slow, the system

does not get stuck in a metastable state, that is, a relative minimum of the energy.

In the computer program used here the same approach is followed, but the function to be minimized and its variables replace the energy and the state variables respectively. One iteration step may be described as follows. From the current point a relatively small step is made. The function value in this new point is computed. If it is lower the new point is accepted. If it is not, it is rejected, but only with a certain prescribed probability depending on the temperature. Note that this implies that the direction of the steps may for one or more consecutive steps be uphill. This is intended to prevent the procedure from being trapped in a relative minimum.

The computer used is composed of a DEC LSI-11/73 CPU, standard DEC IO and RAM boards, a DEC compatible video board and a floppy and hard disk drive. The peripherals are a printer for numerical and graphical output, a monitor for graphical display and a terminal. Thus interactive operation, imperative for this type of experiment, is made possible.

The computer is highly self-contained. It has its own program editing and compiling facilities. All the software is written in ANSI FORTRAN 77. It is modular, structured and is kept portable to future faster systems.

Comment — Compared with the systems of Examples 1 and 2, that of Example 3 is highly self-contained. No other computers are needed for compiling or for off-line computation of parameters required for its operation. As such it is a representative of a new generation of microcomputer/PC applications: dedicated, stand-alone computer systems with peripherals tailored to the job and offering easy-to-use interactive facilities. The microcomputers/PCs involved increasingly have hardware and software characteristics until recently found only in mainframes. They are, in addition, becoming cheaper and cheaper as are their peripherals. Also, being dedicated to one particular job, they ensure long duration operation, uninterrupted by other users. Thus they enable the experimenter to tackle extensive numerical problems, as that of Example 3, in a fashion closely controlled through interaction.

Conclusions

Three microcomputer systems developed at the Faculty of Physics of Delft University have been discussed. The first two of these, a spectral analyzer and a random number generator, were completed at the end of 1983. Since then important developments in computation speed, memory technology and availability of numerical software have taken place. Since these developments were also accompanied by a falling price level, considerable improvements of

both computers would now, only few years later, already be possible. These modernized versions would not only be much faster, but also easier to handle and more self-contained. These properties are clearly visible in the third example, a function minimizing machine whose design started in 1986. Here no help of other computers is needed, while the peripherals used enable the experimenter to easily interact with the computational process. Since, in addition, high-quality numerical software for microcomputers/PCs is now becoming available, and the amount of memory, both random access and background, will hardly form an impediment any more, it may be expected that this type of computer application will in the near future increasingly be found in the physics laboratory.

References

1. KOSTER, A. J.: Development of a user-friendly spectral analyzer using parameter estimation techniques. MSc. Thesis, Part 1. Faculty of Physics, Delft University of Technology. November 1983. (In Dutch)
2. VAN DEN BOS, A.: Parameter estimation. Chapter 8 in: Sydenham, P. H., ed. *Handbook of measurement science*. Chichester: Wiley, 1985. pp 331–377.
3. The NAG Fortran PC50 Library Handbook—Release 1. First Edition. Oxford: The Numerical Algorithms Group, 1983.
4. VAN DEN BOS, A.–EIJKHOFF, P.: Model building and parameter estimation as means for intelligent measurement. Measurement, *Vol. 6*. No. 1, pp. 25–32. 1988.
5. VAN AARLE, S. H. J.: A user-friendly generator for noise with specified amplitude density and autocovariance. MSc. Thesis, Part 1. Faculty of Physics, Delft University of Technology. October 1983. (In Dutch)
6. Price, R. A useful theorem for nonlinear devices having Gaussian inputs. IRE Trans. Inf. Theory, *Vol. IT-4*. June 1959, pp. 69–71.
7. KIRKPATRICK, S.–GELATT, C. D.–VECCHI, M. P.: Optimization by simulated annealing. Science, *Vol. 220*, No. 4598. pp. 671–680.

Dr. Adriaan VAN DEN BOS Faculty of Applied Physics
Delft University of Technology
POB 5046, 2600 GA Delft, The Netherlands