The 1968 Arden House Workshop on Fast Fourier Transform Processing

Editor's Note

As has been noted, most of the papers in this special issue are based on talks given at the G-AE-sponsored Workshop on Fast Fourier Transform Processing at Arden House, New York, October 6–8, 1968. The following five short papers are included here to give the reader an appreciation of one of the subtler, but very important, aspects of this conference—the opportunity to hear first-hand the accounts, opinions, and impressions of the birth and development of the FFT by some of those best in a position to know. Those searching for insights into the background and future of the FFT will do well to read the Workshop Keynote Address of J. W. Cooley, R. L. Garwin's highly readable remembrances of things past, and the workshop comments of Session Chairmen Rader, Bogert, and Stockham.

The Impact of the Fast Fourier Transform—Keynote Address

I don't think it is necessary to dwell too long on the reasons for holding a workshop such as this one. Everyone knows full well how the fast Fourier transform (FFT) method, which was unknown three years ago, has become used and developed to such a phenomenal extent that advances in both methods and applications have occurred far faster than their publication and dissemination can take place. In fact, the three-year history of the FFT describes events and achievements in terms of many orders of magnitude. The first order of magnitude, which many of you may have experienced, was in the first use of the FFT in existing problems. That is, if you were doing calculations of Fourier transforms of functions defined on arrays of 1000 points, you could speed up the calculations by something like 2 orders of magnitude. These improved speeds permitted a wider range of application of Fourier methods. Again, there are several orders of magnitude in the increase in the type of problems people can now treat by using FFT. There are many calculations or processes which were done previously by analog devices which can now be done digitally. For example, FFT methods now permit one to do digital processing on acoustical data in real time. One prediction is that someday radio tuners will operate with digital processing units. I have heard this suggested with tongue in cheek, but one can speculate. Finally, we have achieved a stage where applications not possible by any previous device or method are now being carried out on digital computers. One such application was recently described to me by Mme. Connes, who was fortunately able to come to the conference. She is Director of the Computing Center in the National Observatory in Paris, and her husband is Pierre Connes, the astronomer. Incidentally, P. Connes describes the work I am now referring to in the current issue of *Scientific American*. About the time they heard about the FFT, P. Connes was rapidly developing interferometer techniques, which are procedures for measuring infrared spectra. In principle, the idea is essentially that of the Michaelson–Morley experiment. The infrared radiation is divided, made to traverse two different path lengths, and is then brought together. The intensity of the two superimposed beams is measured for a sequence of path length differences. For a given frequency, one sees a periodic rise and fall of the intensity due to cancellation and reinforcement. The Fourier transform of the measurements gives the strengths of

Manuscript received March 7, 1969.

these periodicities and, therefore, the infrared spectrum. Three years ago, they had reached the stage where they could not process this data with sufficient accuracy in the frequency domain (and by "accuracy" I mean enough accuracy to see spectral lines of diatomic molecules in the infrared radiation from a distant planet when the radiation contains the spectrum of the original source of the light; namely, the sun). They could not get the needed accuracy by analog devices, and the digital devices were too slow. Well, the FFT came just in time and is now being used to compute spectra. The next problem is that Connes has built a new spectrometer. This one measures a record of 512 000 points in one run. This is, I believe, the longest single sequence that I've ever heard of in which one has to obtain the Fourier transform for all frequencies.

In view of these rapid changes in FFT processing, it does seem appropriate to bring together those who are directly making these advances, for the purpose of exchanging information on the rapidly changing state of the art in four major areas: software, hardware, mathematical theory, and fields of application. We hope each attendee will be able to get a good appraisal of the level of achievement in all these areas and its impact on his own work. Perhaps even more important, we would like to get from these sessions a judgment as to where the fruitful areas of endeavor lie and what progress we can expect in the future. There is one additional aspect of FFT processing which makes this workshop even more appropriate. This is the intimate interaction brought about between very diverse areas of endeavor. This is typical of the attendance we have. I think, in fact, that it is quite noteworthy that such a very specialized topic is capable of bringing together people of such diverse specialities. We have several M.D.'s who work on spectral analysis of blood flow and electroencephelograms; there are engineers of various kinds, including those working on radar, sonar, acoustical signals, etc.; there are computer designers, statisticians, programming specialists, and many others, all participating in the same conference. I think this is extraordinary for such a meeting, but in a way, it typifies FFT and its impact. I think that it is also noteworthy that everyone here with such a diversity of interests is interested in and can profit from what everyone else has to say. The people with applications are interested in knowing what to expect by way of mathematical techniques and programming. Programming specialists would like to know what applications people are going to need in terms of such things as in-core or out-of-core computation, the use of tapes, disks, and slow storage. Considerations as to fixed versus floating point and number of significant digits required will be discussed. The hardware people want to know the appropriate algorithms to use, since the FFT covers a whole range or a whole class of algorithms which can be reduced to a few canonical forms, but have many variations.

The effects of such interactions can be illustrated in a conversation I had recently with W. M. Gentleman. I said that I and others had been computing with base-2 algorithms, and we thought that this was what all people needed and would satisfy almost any application. He said that he and Sande programmed the arbitrary-base algorithm in their computing center and this is what everybody seems to need and use. So apparently, what people need is kind of "tailored" to what exists and what can be done. I think many of us here are interested in knowing what exists by way of programs, techniques, hardware, and special devices and are then interested in tailoring our needs to them. This points out one of the reasons for our having no parallel sessions; that is, everyone here will attend all sessions. Therefore, the regulations require that each person have only five to seven minutes to speak. It is hoped that each speaker will be able to summarize his efforts and achieve something in this regrettably short session. It is also hoped that these discussions will lead to an elaboration of those details which are of the most interest to the most people. It is naturally expected that our isolated location will result in close contacts between attendees with common interests.

There are several other matters of interest which have arisen in experience with fast Fourier transform methods which I would like to mention here. One is that this experience points out a need for research in computational algorithms. One result which one could have expected from such research would have said that Fourier transforms could be done in less than N^2 operations. In fact, we would now like to know if $N \log N$ is the minimum number of operations possible. Results of the type I am considering have been given by Winograd, at IBM Research, in other problems. One such result determines the minimum time required to perform a multiplication with a given set of circuit elements and their switching times. Another, due to Winograd, is an algorithm for computing a vector product of two N-element vectors in less than N multiplications. These are the beginnings, I believe, of a branch of computer science which will probably uncover and evaluate other algorithms for high-speed computers.

Finally, I would like to go into one other aspect of this whole development which I find interesting and on which it may be fruitful to speculate. This is the history of the fast Fourier transform method itself. I think that it is particularly appropriate to talk about this here, since we have several people in attendance who are directly involved in the history of the FFT, and we should be able to come up with some interesting discussions. My own involvement started when R. L. Garwin,

who is on our panel tonight, came to the computing center at IBM Research with some notes he had taken in a conversation with J. W. Tukey. When Garwin mentioned to Tukey that he had been computing Fourier transforms, he asked him if he had a faster way of doing it. Tukey then described the essentials of the fast Fourier transform method. Garwin came to the computing center at IBM Research in Yorktown Heights to have the algorithm programmed. I was new at the computing center and was doing some of my own research. Since I was the only one with nothing important to do, they gave me this problem to work on. It looked interesting, but I thought that what I was doing was more important; however, with a little prodding from Garwin, I got a program out in my spare time and gave it to him. It was his problem and I thought I would hear no more about it and went back to doing some real work. The significance of the factor $N \log N$ versus N^2 was lost on me, since I had never had any use for Fourier transforms and was not aware of the extent of Fourier calculations on computers. Experience since then has given me quite an education and when I realized what had happened, I told Garwin that if he had any more ideas like that I would be glad to help him out again. Garwin is a man who gets around quite a bit and meets many people. As a result of his publicity, I started to get letters requesting programs and write-ups. Requests for a paper then started arriving. I was asked to write a paper and did so, asking Tukey to co-author it. He did, and the paper in *Mathematics of Computation*, in 1965, was the result. By the way, I didn't order reprints at the time since I didn't want to repeat previous experiences of having a closet full of reprints which no one wants. This, again, was an underestimate of the significance of the fast Fourier transform algorithm and resulted in plenty of work for the copying machines.

Garwin called again and asked that we organize a few sessions to publicize the FFT. It seemed strange to me then that such a simple little algorithm could assume such importance and that it had not been thought of before. Then a letter from P. Rudnick of the Oceanic Institution in LaJolla, Calif., arrived, in which Rudnick said that he had a program which computes a Fourier transform in a number of operations proportional to $N \log N$. He said he got the method from a paper published in 1942 by Danielson and Lanczos. This paper referenced a paper published in 1924 by Runge and König in a lecture series in *Matematischen Wissenshaften*. This was probably read by or at least available to graduate students in mathematics at that time.

I then talked to L. H. Thomas of the IBM Watson Laboratory and asked him if he had seen these ideas. He said that he had and, in fact, had used them. When I asked him where the method came from, he replied in a matter-of-fact tone that he had a large calculation of Fourier series to do and simply went to the library and looked the method up in a book. It was indeed in a book by Stumpff, which was like a cookbook on Fourier series calculations for the desk calculator user. In it were all sorts of procedures, using various symmetries of the sines and cosines for special values of N. These methods reduced the work, but when they could be generalized, they still required a number of operations proportional to N^2. In some almost insignificant paragraphs of the book, a doubling algorithm is given, which permits one to obtain a Fourier analysis of $2N$ points from two analyses of N points, in N operations. It is given for special values of N and asks the reader to generalize. A peculiar sidelight is that Thomas generalized it and got a different algorithm, the one requiring that the factors of N be mutually prime. Furthermore, another thing about this is that Thomas did the calculation in 1948 on an IBM accounting machine. I don't know if you are familiar with this kind of equipment, but it is a rather primitive device. Even with an efficient method, the calculation took three months and, I suppose, might not have been done at all if it had not been programmed efficiently.

Another early author who should be mentioned here is I. J. Good, a statistician who generalized some procedures of Yates for factorial experiments and derived an algorithm which, on my first reading, I believed to be Tukey's; however, I didn't read very carefully. What Good had was the mutually prime factor algorithm used by Thomas. If you require mutually prime factors, you cannot make all the factors equal to the optimal values of 2 and 4 or, perhaps, 8. But it is still a very good trick to use, and, in some cases, could be used profitably with the fast Fourier transform method subroutines.

We should learn from this history what to do if, for instance, Garwin comes back to the computing center with another such idea, or if somebody finds some other trick which perhaps didn't take hold before computers existed, but which can be useful now. What will become of these useful ideas? How does one get them to people who need them? How can they be communicated to the mathematicians and analysts who can perhaps extend them and develop them into programs and hardware? Do you publish an idea like this in a *Journal of the Franklin Institute* as Lanczos and Danielson did? No. Perhaps we could publish it in the *IEEE Transactions on Audio and Electroacoustics*, but this seems to be a very specialized group. What do we do? The group on Audio and Electroacoustics has done more than its share by devoting a special issue of its transactions to the FFT and by organizing this and a previous workshop. These are some of the questions I would like to throw around for the panel discussion and for comments from the audience later.

JAMES W. COOLEY
IBM Watson Research Center
Yorktown Heights, N. Y. 10598

The Fast Fourier Transform as an Example of the Difficulty in Gaining Wide Use for a New Technique

Abstract

The history of the fast Fourier transform of Cooley and Tukey, as well as experience with general-purpose optimization programs, suggests that publication alone does not result in wide use of a new and vastly more efficient computational technique. Recounting his role as entrepreneur and missionary in connection with the fast Fourier transform, the author emphasizes these difficulties and notes the need for mechanisms for easing the cross-utilization of valuable new techniques.

Manuscript received February 28, 1969.

Here I intend to indicate how much more than an idea or even a publication is required to bring into wide use a new technique. Perhaps others, not previously aware of these difficulties, can propose solutions, but I thought it useful to note as an example my part in the recent history of the fast Fourier transform.

First I want to disclaim any intellectual contributions at all to the development of the fast Fourier transform. However, I do know a good thing when I see one, and it seemed to me that my calculation involving vast Fourier transforms was not unique but typical of that facing people in many other fields. When I learned of the tremendously exciting possibility of the fast Fourier transform, I resolved that I would do more than use it to solve my own problem; I would do what was necessary to make it practically available to everyone.

I previously had some experience in publishing an algorithm for minimizing general functions. In the late 1950's, the methods in use on computers were largely gradient methods which, for the most part, had only exponential approach to the minimum, the error going like exp $(-\alpha n)$, where n is the number of iterations. Unless one was careful about scaling, as many as 1000 iterations or more might be necessary to achieve reasonable accuracy. So H. Reich and I thought of a scheme which was much better and which had quadratic approach to the minimum, so that the error decreased as exp $(-\alpha 2^n)$. We tried it extensively, wrote it up, and sent it to a well-known journal of mathematical techniques and computing. After eleven months we heard that the paper was still on the editor's desk. At that point we decided, probably wrongly, that a new technique by W. C. Davidon was even better than ours, and we never published our paper. Similar methods have since been reinvented and published, but many computations are still done with the old gradient methods. With this in mind, I decided that prompt publication of the fast Fourier transform was important, and that a tested, general-purpose program would have to be made available, but that these two publications would not in themselves be enough. Contrary to the proverb, people do not really beat a path to the door when you make a better mousetrap. Accordingly, it may be useful to recount my involvement with the fast Fourier transform. In this I played the part of entrepreneur, a vital role in society, but one for which the title "missionary" might be as accurate in the present case.

I am myself a physicist, and I first met J. W. Tukey in the wilds of Washington sometime in the 1950's; but our paths crossed again in 1959 when we were together for six weeks on what was called Technical Working Group Number Two or TW2, following the Conference of Experts on the Banning of Nuclear Explosions. Tukey was a member, and I was a kibitzer on that group, being in Geneva for a year to work at CERN on particle physics. The main problem of TW2 was to determine (jointly among nations!) whether nuclear explosions underground could be detected and distinguished reliably from earthquakes. Various experts thus met in Geneva to pool their knowledge as to the behavior of seismographs and their response to earthquakes and bomb tests. Sociologically, it was a very interesting experience. Scientifically, it was somewhat less than that, but occasional interesting problems arose, and one happened to involve me.

There had been extensive discussion between the Americans and the Russians as to whose seismometer was better. Everyone agreed that underground nuclear explosions of some yield could be detected, but the question was the lower limit to the reliably detectable yield. Now, looking back on the record, one can see that there was confusion as to differing interpretations on the two sides of the phrase "a maximum amplification of 10^6 at one cycle per second," which the Americans took to mean that the maximum amplification should be at 1 hertz, while the Russians took it to mean that the seismometer should have no more amplification than 10^6 at 1 hertz, but that the curve was otherwise unconstrained. At the time, in discussion through interpreters, it never occurred to either side that there was a misunderstanding on the fundamentals of the hypothesis. The Soviets claimed that their seismometers were better for detecting underground explosions, and we, as gentlemen, said that ours were better. So finally, I decided that the most useful contribution I could make would be actually to compute the response of the two seismometers to some combination of noise and earthquake signal. Unfortunately, my knowledge of the details of the seismometers was limited to the publications which contained graphs of the logarithm of the magnitude of the amplification versus frequency. Naturally what one could do with such a curve is to compute the phase response which goes with that amplitude response, Fourier transform the noise and the signal, multiply by the complex frequency response of the seismometer, Fourier transform the result, and draw the curve for seismometer output versus time. Well, even with Tukey in residence, I didn't remember how to go about recursive digital filtering and things like that, so what I did was to make a model of the seismometer with inductances and capacitors (all simulated, of course) which would have the same amplitude response. If I had been a mechanical engineer, naturally, I would have made a computer model with springs and masses. I then integrated the differential equation step by step in time, using as an input an approximation to what seismologists told me earthquakes look like after they have propagated through the ground. The next morning, at the conference session, I arrived with two curves, one the response of the Soviet seismometer and the other the response of the U. S. seismometer. To my unbiased eye, the U. S. seismometer looked better, and even the Soviets had to agree that theirs was not very much superior in any case.

Although this experience was in 1959, and although my difficulties were largely due to my own ignorance of computational techniques, I have since seen a great deal of ignorance and inefficient use of computer time. For instance, in underwater acoustics one often wants to obtain a high-resolution Fourier transform of a signal over the range of a few hertz to some thousands of hertz. With resolution on the order of 1 hertz, over a bandwidth of some 10^4 hertz, one has on the order of 10^4 individual Fourier components, and the straightforward way of obtaining all frequency components would thus require 10^8 multiplications. No wonder the people working in that field did not use general-purpose digital equipment for such spectral analysis! Of course, whether you need this resolution throughout the spectrum is another question. But I was acquainted with many problems in reentry physics, in geophysics, in astronomy, etc., where many people were being stymied by the scope of Fourier transforms. My friends the molecular biologists were doing great work determining the structure of biologically important materials—proteins, en-

zymes, DNA. Many X-ray diffraction patterns have as many as 10^5 spots, each of which is a physical representation of the magnitude of a Fourier coefficient. Even if the phases are known, some 10^5 spatial determinations of electron density can be made from these spots; but the ordinary Fourier transform method is extremely lengthy.

Well, on the one hand, I was glad that the burden of all of these people was being eased by large-scale computers; but, on the other hand, I felt that it ought somehow to be easier, and finally I was presented with the problem myself in my own research. As a low-temperature physicist I was doing some work on solid helium. Solid He^3 crystallizes in three phases, and I won't bother you with the details, except that these experiments were yielding very peculiar results. We conjectured that the nuclear spins of the He^3 atoms were aligning themselves either parallel or antiparallel with respect to their nearest neighbors. The parallel arrangement is easily understood, because all the spins in the crystal can be lined up parallel with no difficulty in any crystal arrangement. But if you ask for all the spins to be antiparallel, on the face of it that is a contradiction. In a body-centered cubic lattice, for instance, it is easy to have the mth spin surrounded by nearest neighbors all of which are antiparallel to the direction of the mth, and to have every nearest neighbor of those spins antiparallel to them. In a hexagonal close-packed lattice, this is not possible. So in only some structures can one arrange that every spin have nearest neighbors all pointing in the other direction, and every one of those neighbors have its nearest neighbors pointing in the opposite direction, giving an absolute minimum of "exchange energy" for the system. But I was interested in the hexagonal close-packed structure, for which the minimum energy is obtained from some presumably more complicated periodic arrangement of spins in space.

So, being frustrated with the experiment and having to spend the days rebuilding the apparatus, I started thinking about this and did a computer experiment. I took some 10 000 spins (limited by the extent of magnetic-core memory available in the computer), arranged them on a lattice, and essentially froze them in position. I then went through the lattice one spin at a time and relaxed each of the spins to the local direction of its neighbors. Ten thousand spins do not make a very large cubic lattice—there are about 20 spins on each side of the cube. Well, after about 40 such relaxation iterations on the whole lattice, the exchange energy as a function of iteration number became pretty much constant. If I put the spins in again at random and relaxed the lattice in this way once more, the exchange energy went exponentionally to the same minimum value, although not to so low a value as obtains for a cubic lattice. (By that time, I had done some more experiments and found that antiferromagnetism was not the explanation for the experimental results; but I was still interested in the theoretical problem.) So in attempting to determine the orientation, magnitude, and polarization of the antiferromagnetic spin wave, I first printed out a cross section of all the spins in the xy plane, then in the xz plane, etc., and couldn't see very much. It occurred to me that there was mathematics invented long ago for this problem, and that all I needed to do was to take a three-dimensional Fourier transform of the spins as a function of position in order to obtain a spectrum, the peak of which would be precisely the spin wave I was looking for. I expected that I would see a large peak, of some width, caused by the "windowing" (because my spin system was cut off sharply in the form of a cube). Instead of having a δ-function for the spectrum as I would expect for an infinite lattice, I would have a finite peak and then some sidelobes falling off slowly from the maximum.

Indeed, this was a feasible and elementary calculation, except for the cost of the required computer time. The relaxation to obtain a state of minimum exchange energy took about 5 minutes on the computer, and the Fourier transform in the most naive possible way would have required 10^4 calculations, each of which contained 10^4 multiplications and additions, and thus on the order of 10^8 multiplications and additions, which on the 7094 would have been a good many minutes. Well, I stewed about that for awhile, and then one day at a meeting of the President's Science Advisory Committee, I saw Tukey doing his usual thing during the meeting, namely, writing. This time he was writing Fourier transforms with his left hand, and I asked him whether he knew anything I didn't know. Naturally, he told me a number of things about Fourier transforms that I found hard to believe for a little while. Then I did believe it and I was faced with a problem. Here was a scheme to do a complete discrete Fourier transform not in N^2 multiplications and additions but in $N \log_2 N$ multiplications and additions, which would save me on the order of a factor 10^3 in my calculation. Ought I have my programmer code this algorithm for my problem alone, as we had done on the optimization routine, or ought I violate my principles and get some outside help to produce a fully tested and fully documented program, in view of the wide applicability of the algorithm far beyond my own concerns with He^3? In delaying the availability of the program for my own work, I realized at that time that there were all kinds of possible applications ranging from the forming of adaptive antenna beams to the processing of high-resolution pictures and to the X-ray diffraction structure determinations of molecular biology.

As an example in the processing of pictures, if one has on the order of 100 line pairs per millimeter across a 10 cm format, the picture has 4×10^8 elements. To do a simple Fourier transform for compensation of image blur, contrast enhancement, improvement of focus, etc. (which would be worthwhile for a unique picture of which there will never be another, perhaps like those from the Lunar Orbiter or the Surveyor), such a complete Fourier transform would take on the order of 10^{17} multiplications, when done in the simplest possible way. Such a calculation would take hundreds of years on the fastest computer extant. Actually, even without the fast Fourier transform the job can be done in a considerably shorter time. But now with the simplest possible fast Fourier transform the job can be done in about 10^{10} multiplications, which is an hour's work on a relatively fast machine.

There are a couple of other applications which had been similarly frustrated by the absence of an efficient Fourier transform algorithm. One of those is the so-called Fourier spectrometry. In this type of optical spectrometer, one uses a Michelson interferometer, the path length of which is scanned linearly in time, thus producing a single temporal frequency for each wavelength of light in the input. So with a broad spectrum of light entering, one obtains from a single photocell a set of superimposed sine waves in time. In the past, this spectrometer was restricted to a low resolution, to the analysis of narrow bandwidths, or to the use of analog filters. The stan-

dard book in the field indicates that N^2 multiplications are required to do a complete Fourier transform in order to isolate N spectral frequencies, and the application of Fourier interferometry would always be limited to relatively low resolution or to narrow bandwidth.

The existence of all these potential applications indicates why I was really very pleased to hear that Tukey had an algorithm which would reduce the work involved by a factor $N/\log_2 N$. Accordingly, I asked H. Goldstine, then Director of Mathematical Sciences at IBM, who promised that he would find a mathematical analyst who would produce a generally useful package. So J. W. Cooley was sold into bondage for a while with the results that you all know.[1]

My experience with the optimization program led me to doubt that the publication of the fast Fourier transform would quickly lead to its adoption in all those computations which could use it so productively. The algorithm exceeded expectations in that the program produced by Cooley did the computation in such a way that no storage was required in excess of that needed for the initial data. Thus the program was done and tested and useful to me, and Cooley and Tukey did prepare the paper for publication, but it was apparent that this in itself would not reach the large mass of users who were interested in their own problems and not in computing techniques as such. A problem of the real world is that although some man-months had been invested in the creation of this program and its publication, the product could not itself be sold directly. A scientist regards his work as finished when he publishes (many regard it as finished before, but I am not among them). No manufacturer regards his work as complete until the product is *sold*. But it was not very easy to get large numbers of IBM scientific salesmen to attend meetings which we organized in order to familiarize them with the fast Fourier transform. Eventually, after several meetings, we did reach many salesmen to inform them of this new tool which would help their current customers and which would make computers more powerful and so open up applications previously unattractive for digital computers. At the same time, I visited several laboratories and carried on a correspondence with anyone who I could see from his publications would benefit from the fast Fourier transform. Thus, I visited the planetary astronomy people at Lincoln Laboratory. Actually, I went not to give them the fast Fourier transform, but I felt so inferior at learning so much from them of radio astronomy and at not being able to give them any radio astronomy in return, that I was careful to explain the fast Fourier transform, which they have used to very good effect. Similarly, I pointed out to workers in underwater sound that they could get digitally computed spectra in much shorter times than by their analog technique, and with much higher resolution. I think we probably helped out a little bit with IBM sales teams who were attempting to sell computers to people who were doing vibration analysis of complex systems. Here, a single tap on the object leads to an accelerometer response, the Fourier transform of which can give the entire modal spectrum. Thus, a few seconds of testing can replace a long program of measurements of linear response to a swept CW excitation.

By any criterion, I think that the fast Fourier transform has been a success, and I am glad to have been associated with it. We need suggestions as to how to avoid this problem in the future, and particularly how to make sure that techniques which are of use in Field A can cross the boundary to Field B, especially when the individual researchers who are specialists in Field B do not read the proper journal of technique in Field A. Research journals ordinarily will not accept papers describing general-purpose tools, just because they do not have the vision to see that they are, in fact, generally applicable. On the other hand, the person who invents the technique ordinarily does not benefit by its adoption by ten thousand other people across the country or by giving it to the whole world, so he contents himself with a publication which may reach only a few people.

Then there is the question of what to name a technique. Fast Fourier transform is a pretty good name, for anybody who knows about Fourier transform. My other experience with the general-purpose minimization program was not such a happy one, because I was astonished after several years to find out that similar programs can be found under the title "Mathematical Programming," as well as under "Optimization," "Minimization," or "Maximization." In fact, "Mathematical Programming" was also empty of efficient techniques at the time we did our work, but I would not have known had there been adequate techniques available.

With the fast Fourier transform, there was eventually clear evidence that publication is not in itself enough, as has been indicated.[2] Such methods had been used in the past in 1942 and had roots extending back to the turn of the century. The problem is that anybody who had worked extensively with hand calculation of Fourier series tried very hard to minimize the amount of computation he had to do when he personally had to sit down with the desk calculator to do it. In 1948, L. H. Thomas spent three months doing calculations of Fourier series on a tabulating machine, also by an efficient method, analogous to the fast Fourier transform. But more recently, as the speed of computation increased by orders of magnitude, people have relied on the speed of machines rather than on the depth of analysis, and many machines have done calculations which could have been done for a factor 100 less money and in shorter time if the analysts had sought a bit longer for a better way already in being. On the other hand, we all know people who never finish a job because they are always refining the tools they have to do one portion of it. Somehow, we must strike a balance between those who seize upon the first tool, no matter how expensive, and those who forever refine their tool without using it.

In solving problems, I think there are three stages of improvement possible. One is to improve the speed of the operation currently in use to do the job. Thus, one can use a faster computer which does the arithmetic in one-tenth the previous time. The same number of multiplications and additions are done as before. The next step is to use a new algorithm. The

[1] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," *Math. Comp.*, vol. 19, pp. 297–301, April 1965.

[2] J. W. Cooley, P. A. W. Lewis, and P. D. Welch, "Historical notes on the fast Fourier transform," *IEEE Trans. Audio and Electroacoustics*, vol. AU-15, pp. 76–79, June 1967.

fast Fourier transform typifies this type of improvement. A different algorithm for Fourier transform can thus make a factor 10^2, 10^3, or 10^6 improvement in speed; the larger the job the larger the improvement. But finally, one can reanalyze the problem so that in this case it doesn't even require a Fourier transform and maybe not even a digital computer. Probably it is in this step that major progress will continue to be made in the next five to ten years, although there is, no doubt, much improvement still to be made in efficient algorithms.

On the one hand, increasing the speed of arithmetic is a job for the computer manufacturers, of which there are many. Improving the means of analysis is a job for specialists in the application field; but there is a middle ground, contributed to by the manufacturers, as well as by the users, which I think can benefit everyone. This broad middle ground is to have a new algorithm which can be isolated as a package, a subroutine, or a technique publication, and which can do the work of many people in many different fields. Perhaps in the next few years we will see a good many new algorithms, and I hope that they are as productive as the fast Fourier transform.

RICHARD L. GARWIN
IBM Watson Lab.
New York, N. Y. 10025

Remarks on the Fast Fourier Transform

I would like to start by scoffing, just a little bit, at the great new tool which is the subject of this conference. I would like to present a negative view, which does not exactly represent my point of view, but certainly one aspect of it, and an aspect which is often ignored. I would like to present the negative idea that, perhaps, since the Fourier transform has so greatly reduced the penalty paid in attacking large problems, we are that much more likely to attack such problems in the wrong way. We can, in other words, get into the bad habit of looking to the FFT as a likely avenue of solution to all our problems, thus perhaps missing the simpler solutions. Probably most of us, presented with a large data processing problem, would automatically think of the FFT and try to work it in somehow.

Consider Hilbert transforming a signal. In the example Garwin suggested, deducing the characteristics of a filter from its log-magnitude response, the relationship used (assuming a so-called minimum-phase filter) is the Hilbert transform. Basically, the Hilbert transform of a signal is the convolution of the signal with a weighting function that looks like $1/n$. One way to do the convolution is by high-speed convolution. Ignoring any differences between circular and ordinary convolution, the procedure required is to 1) transform the signal, 2) multiply half of the transform points by -1, and 3) inverse transform.[1] Since two FFTs are required, the computational effort is probably not costly enough to scare anybody off, and this would have been an excellent tool for Garwin to have used at the time, or today. *However*, for most data processing applications in which the computation of a Hilbert transform is required, what is really required is a pair of signals which are Hilbert transforms of each other. Both signals can be derived from the original with a simple all-pass phase-splitting network whose outputs are 90° out of phase with each other. Such a network, realized as a recursive digital filter, can be programmed with perhaps three to ten multiplications per output point, depending on the specifications to be met. These are real multiplications, the storage requirements are negligible, continuous data can be processed, there are no special sectioning considerations, etc. In the analog world, the use of such a phase-splitter is a well recognized technique and, as such, it should quickly suggest itself to us in digital data processing, yet it will almost certainly tend to be overlooked when there exists an FFT program with an option for computing Hilbert transforms. The digital processor has gotten used to thinking of the FFT as the best possible algorithm, and to trying to squeeze all his problems into the format of the Fourier transform. The simpler method, the phase-splitting network, is likely to be missed.

But enough of this negative attitude. Personally, I've paid the price of computing spectra the old way, at half an hour apiece, and I can appreciate the advantage of computing them in five seconds instead. But I have been even more appreciative of the opportunity the algorithm has afforded me to learn some things that are mathematically fundamental, and the opportunity to use my head instead of my computer (or perhaps both together). It is tempting to think that some of the means of utilizing hidden order which lead to the FFT algorithm can yield other algorithms for other computations. A recent paper by Gentleman[2] demonstrates that other matrices than $\exp(j 2\pi mn/N)$ can be factored in a manner similar to the DFT, and that matrices so favorable could multiply other matrices or vectors quite quickly. This may be much like the observation that scalar multiplication by powers of two is easy; e.g., such matrices may be too rare to be important. But one nontrivial example is multiplication by the Hadamard matrix. I also know of some work at the Syracuse University Research Corporation by Blackwell, Shurtleff, and Rudolph, regarding finite Abelian transforms, which, as I understand them, are exactly that class of transforms which, like the DFT, can be expressed as multiplication by a matrix which is a direct product. Surprisingly, this work led to the design of logic circuits made of microwave elements.

Recognizing a principle often makes bearable learning something already well known by others, which one might otherwise have successfully avoided learning. This has certainly been the case with me. For example, the idea is well known that a convolution transforms to the product of transforms, and I supposed that, since a DFT is approximately a Fourier transform, a product of DFTs should be approximately the transform of a convolution. However, I had quite successfully avoided thinking about the nature of the approximation because it promised to be uninteresting. I was excited to learn that the expression was exact, even though it was exactly a circular convolution rather than exactly an ordinary convolution. But recognition of this simple result led me to realize that

Manuscript received January 16, 1969.
This work was sponsored by the U. S. Air Force.

[1] This procedure is not meant to be precisely correct, only essentially so. I trust that anyone wanting to use it would work out the details himself.

[2] W. M. Gentleman, *Bell Sys. Tech. J.*, vol. 47, July 1968.

there is a complete set of DFT relationships analogous to those of continuous Fourier theory, although the analogies sometimes lead to properties which we would not ordinarily think of as analogous (for example, the discrete analogy to the time scaling property is a relation involving permutation of the points in the transform). It was only after the appreciation of the FFT that I learned enough to properly appreciate the DFT.

The FFT has also forced me to learn something about accuracy of numerical computations. The first reports about the FFT mentioned briefly that it was more accurate than standard methods (the direct sum), and I had the impression that this was due to the reduced number of operations involved. Yet a little bit of thought will reveal that, while the total number of operations performed is fewer, the number of operations leading to the result for any given frequency is greater for the FFT than for the direct method. The accuracy derives, rather, as Cooley pointed out to me, from the order in which the operations are performed. The FFT is naturally computed by adding sums pair-wise, then adding the pairs of sums pair-wise, and so on. This would be a very unnatural way of computing the DFT from the defining expression but it is the most natural way to compute the FFT. If the unnatural way of computing the DFT from the defining expression is used, the accuracy is somewhat better than for the FFT, although the difference is not important. The above holds for floating-point computation, for which addition is a noisy process. For fixed-point computation, addition is noiseless, assuming no overflows, and therefore the FFT is less accurate than the defining expression, regardless of the order of performing the operations.

The previous speakers have discussed, somewhat, the education of the scientific community about the existence of the FFT and what it does. I have some comments based on my experience.

A widespread misconception for the technical community is the idea that the number of points must be a power of two and that the transform takes $N \log N$ operations. I think that everyone in this room must know that the true restriction is only that N must be a composite number, and that the number of operations is N times the sum of the factors of N. Why does the misconception arise? I venture to suggest that it has arisen, not because radix-two programs are easier to write, or more readily available, which they are, but rather because "$N \log N$" is so easy to say and easy to remember. We accept $N \log N$ as a legitimate formula, whereas "N times the sum of the factors of N" seems more in the realm of something yet to be reduced to a formula, and therefore more disquieting and less to be trusted. Of course, in the last year it has become known that the DFT can be computed in "fast" times even for noncomposite N. This is even possible when not all points of the transform are to be computed, or even when the z-transform is to be computed on a contour other than the unit circle. Hopefully, the publicity attendant to these new results will be quicker in taking effect than the publicity of the FFT.

It is interesting that the FFT was rediscovered more than once, but there are other waveform processing tools which are discovered repeatedly. For example, the digital Butterworth filter has been discovered independently, and the discovery published, by perhaps a dozen people, including myself, and I don't know what can be done to prevent this waste of effort. I've actually seen the discovery of the digital Butterworth filter published in the same journal by two different authors in two successive issues. Apparently most scientific communication is by word of mouth, a strange communication process indeed. Garwin mentioned that he told people at Lincoln Laboratory about the FFT in 1963. I'm with Lincoln Laboratory. How did I find out about the FFT? T. Crystal of M.I.T.[3] mentioned to me quite casually one day that he knew of an algorithm for rapidly computing a spectrum, but he didn't understand it. This was February, 1965. He had gotten a preprint of the Cooley–Tukey paper from Tukey's secretary. So I made a copy of the preprint and so did T. G. Stockham, and together we figured it out one afternoon, or thought we did. Anyway we understood it well enough to produce a program. But, as we didn't know, E. Gehrels, of Lincoln Laboratory, had already written such a program, and used it in a radar processing problem. Most people at Lincoln Laboratory and M.I.T. first learned about the FFT from a couple of seminars given by Stockham and myself[4] in May, 1965, despite the fact that, as I learned for the first time tonight, Garwin had given us the inside track about two years earlier.

To change the subject slightly, I'd like to mention the signal flow graphs which have become popular descriptions of the FFT. When Stockham and I were trying to understand the Cooley–Tukey paper, we constructed a Mason flow graph for the case of an eight-point transform. This is a fairly natural approach for anyone with an engineering background. Probably everyone has seen the kind of picture which results. Instantly, the strange properties of the FFT become clear—the bit reversal, the in-place computation, the $N \log N$ operations, and the expression of an N-point DFT as two $(N/2)$-point DFTs of suitably combined inputs. This flow graph can be a much better representation of a digital waveform processing algorithm than the customary flow chart which programmers are used to. The reason is that the flow graph describes the relations of the signals nicely, while the flow chart is most efficient in describing the flow of control in the operation of a program. Both descriptions have their uses for the FFT, of course, but for digital filters and the like, the superiority of the flow graph as a representation is considerable.

This kind of picture was the key to my initial understanding of the FFT. Actually, there are many suitable ways of understanding what is going on, in terms of Fourier theory, matrix theory, number theory, multidimensional transforms, flow graphs, and others. One of the most interesting ways, which I hesitate to label, is based on some work in connection with beam-forming networks for array radars. This has, in fact, some part in the history of the fast Fourier transform, and seems worth mentioning if only for that reason. Suppose we have an array of equally spaced radar receivers in a straight line and a radar wavefront is approaching the array at an angle. The phasor measured by each sensor will be out of phase with the phasor at each other sensor, in general, and the sum of all the phasors will be zero, or small, because of the different phases. However, if each sensor output is subjected to an appropriate phase shift, they can all be added together in phase to produce a big output. A radar beam-forming network is a collection of phase shifters and adders which produce out-

[3] Presently with Signatron, Inc., Lexington, Mass.
[4] At least, that is my impression.

puts which are maximum when the wavefront is impinging on the array from a given direction. This turns out to be a Fourier transform.

The conventional method of forming beams is the so-called Blass array—an assemblage of N^2 phase shifters and adders which is analogous to the conventional scheme of computing the discrete Fourier transform. A phase shifter operating on a sinusoidal waveform is equivalent to multiplying a complex number by a root of unity. J. Butler and, independently, J. P. Shelton had figured out a way of connecting together phase shifters and adders to the sensors so as to require only $N \log N$ elements to generate N beams for N antennas. The device is called a Butler matrix, and is completely analogous to the fast Fourier transform. When you look at a diagram of a Butler matrix, having seen the flow graph for an FFT, the connection is seen immediately. When I lectured about the FFT at Lincoln Laboratory, W. Delaney asked, "Isn't that a Butler matrix?" At that point I had never heard of a Butler matrix, and afterwards he explained it to me. Here is the crowning irony: engineers who were completely familiar with the Butler matrix had been using conventional DFT algorithms to evaluate the beam patterns of various antenna feed structures on a computer. They hadn't recognized the Butler matrix as a computational aid as well as a structural simplification. This is not surprising because they were antenna designers and were not much interested in programming problems.

In conclusion, in spite of the FFT idea being new to us in 1965 only because of the poor publicity attendant to its earlier discovery, I think we can fairly consider Cooley and Tukey to have disproved the old proverb, "There's nothing new under the sun."

C. M. Rader
M.I.T. Lincoln Lab.
Lexington, Mass. 02173

Further Remarks on the Fast Fourier Transform

I don't plan to take very much of your time because you have heard just about everything I wanted to say. I would like to make two small points. In the first place, I think that the FFT algorithm has been very helpful in bringing together the practicing statistician and electrical engineer. As I see it, there is a tendency for many electrical engineers to indulge in theoretical modeling and to choose to ignore, more than they should, disagreeable realities. Applied statisticians, on the other hand, are expected to make sense out of the disagreeable realities. The use of spectral techniques has been a useful link between the electrical engineer and the applied statistician, and FFT will make this link even stronger.

The other point is that the practical application of the FFT algorithm, in terms of its hardware embodiment, is likely to produce considerable activity in trying to understand the effects produced by the specific embodiments of the algorithm. An example of this in a related field is the digital array phasing technique called DIMUS (Digital Multibeam Steering) invented by V. C. Anderson and described in the *Journal of the Acoustical Society of America* in 1960. One ordinarily expects to perform array beam steering by that idealized linear operator called time delay. In digital systems, a linear time delay will follow digitization with an appropriate number of bits. Obviously, the more bits the closer the practical embodiment approaches the idealized linear operator. In the interests of hardware simplicity, DIMUS uses but one bit. This raises important problems of degradation of array performance which in turn resulted in theoretical work describing this degradation quantitatively.

I think a similar situation will apply to FFT. My own interest in FFT, besides the computer time savings for digital spectrum analysis, was whether one could really build hardware that would give the advertised savings of FFT without some hidden bugs. We don't usually get something for nothing, yet with the FFT algorithm, when incorporated in appropriate hardware, we found that we got almost everything we had hoped for. But this is not to say that we had no problems due to the particular embodiment of the algorithm, which was motivated by simplicity and speed. As time goes on, I expect we will see a dialogue between practical constraints on the hardware embodiment and consequences in terms of the function performed. In a specific hardware design we may be after a finite Fourier transform, but what we get may be slightly different, and the difference may well be significant in the application we have in mind.

B. P. Bogert
Bell Telephone Labs.
Whippany, N. J. 07981

Applications of the Fast Fourier Transform

Since the introduction of the fast Fourier transform to the computing world in 1965, we have seen the invention of many applications. It has been both interesting and rewarding to watch this activity. Some of these applications have been traditional, relying on well-established theories, but making use of the FFT speed advantages to overcome economic obstacles. Others have been innovative, applying the Fourier transform in ways not before deemed appropriate. The examples of high-speed spectrum analysis and high-speed convolution come quickly to mind. Then, too, we must not overlook the fact that benefits have ensued merely because the FFT has been on people's minds. Much creative thinking that might not otherwise have transpired has been induced. For example, consider the invention of the so-called chirp z transform which Rabiner, Shafer, Rader, and Bluestein have studied and developed. Also, consider the effect that the fast Fourier transform has had upon the use of the digital computer as a laboratory waveform processing instrument.

There will be more applications like the above which we will see in the future. However, it seems to me that a still more important benefit is yet to come and that it will emerge in the area of electronic design. It is exciting to listen to those who tell us what we may expect from the technology of large-scale

Manuscript received February 19, 1969.

Manuscript received January 27, 1969.

integration. There can be no question that the kinds and numbers of electronic signal processing devices of the future would overwhelm us if we had them now. Therefore, we must soon answer specific questions about the designs for these equipments. I suggest that the fast Fourier transform and its derivatives, which I choose to call fast linear transforms, will play an important role in answering some of these questions. Two considerations motivate this statement. The first is the obvious fact that the linear system for which the Fourier transform is the basis of analysis will continue to play an extremely important role in the design of systems.

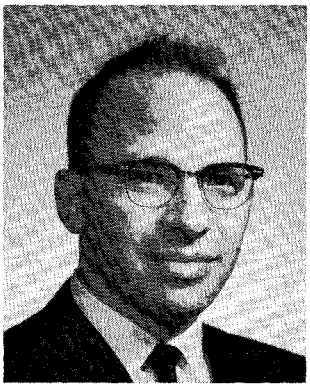
The second far more important consideration is the conviction that linear systems, both stationary and nonstationary, will also play a central role in the design of new and fascinating nonlinear information processing mechanisms. The possibility of this trend has been given considerable substance by the theory of generalized superposition of Oppenheim. This theory brings for the first time in my knowledge a broad framework of analysis and synthesis to the subject of nonlinear communications and signal processing. When coupled with the ultimate availability of light, inexpensive, and compact fast linear transform hardware, it points to an excitingly rich and flexible technology for tomorrow.

THOMAS G. STOCKHAM, JR.
University of Utah
Salt Lake City, Utah 84112

James W. Cooley was born in New York, N. Y., on September 18, 1926. He received the B.A. degree in 1949 from Manhattan College, New York, and the M.A. degree in mathematics and the Ph.D. degree in applied mathematics in 1951 and 1961, respectively, from Columbia University, New York.

He was a Programmer at the Institute for Advanced Study from 1953 to 1956, and was employed at the Computing Center of the Courant Institute of Mathematical Sciences, New York University, where he was engaged in quantum-mechanical computations. He joined the IBM Corporation in 1962 and has been working in numerical analysis and computation. His work has included the calculation of molecular wavefunctions, the solution of the partial differential equations of several models of biological membranes, and the solution of electro-diffusion equations for semiconductors. He is currently at IBM Watson Research Center, Yorktown Heights, N. Y.

Dr. Cooley is a member of the Association for Computing Machinery and the Scientific Research Society of America.



Richard L. Garwin was born in Cleveland, Ohio, on April 19, 1928. He received the B.S. degree from Case Institute of Technology, Cleveland in 1947, and the Ph.D. degree under Enrico Fermi from the University of Chicago, Chicago, Ill., in 1949.

After three years on the faculty of the University of Chicago, he joined IBM Corporation in 1952, and is at present a Fellow at the IBM Watson Laboratory, New York, N. Y., and Adjunct Professor of Physics at Columbia University, New York. In addition, he is a consultant to the U. S. Government on matters of military technology, arms control, etc. His recent fields of research include work on liquid helium, superconductors, fundamental particles of physics, and on novel computer and communication elements and systems. He has made contributions in the design of nuclear weapons, in instruments and electronics for research in nuclear and low-temperature physics, in the establishment of the nonconservation of parity and the demonstration of some of its striking consequences, in computer elements and systems including superconducting devices, in communication systems, in the behavior of solid helium, and in military technology.

He is a member of the President's Science Advisory Committee, the Defense Science Board, and the National Academy of Sciences. He is a Fellow of the American Physical Society and a member of Sigma Xi.

Charles M. Rader (S'59–M'62) was born in Brooklyn, N. Y., on June 20, 1939. He received the B.E.E. and M.E.E. degrees from the Polytechnic Institute of Brooklyn in 1960 and 1961, respectively.

He has been with the M.I.T. Lincoln Laboratory, Lexington, Mass., since 1961, where he has worked on techniques for speech bandwidth compression and computer simulation.

Mr. Rader is a member of Eta Kappa Nu, Tau Beta Pi, and the Acoustical Society of America.

Bruce P. Bogert (SM'68) was born in Waltham, Mass., on September 26, 1923. He received the B.S. degree in physics and the M.S. and Ph.D. degrees in applied mathematics from the Massachusetts Institute of Technology, Cambridge, in 1944, 1946, and 1948, respectively.

He joined Bell Telephone Laboratories, Inc., Murray Hill, N. J., in 1948. During his early career he carried out theoretical and experimental research in acoustics, including the theory of hearing, propagation of sound in tubes, speech analysis and synthesis, classified underwater acoustics, and speech bandwidth reduction systems. From 1955 to 1958, he engaged in studies of gyrator circuits, phase equalization of telephone circuits, and supervised a group studying speculative designs for telephone station apparatus. From 1959 to 1963, he studied applications of signal analysis to seismological problems. He was appointed head of the Transmission Research Department in 1962, and became head of the Underwater Research Department at Bell Telephone Laboratories, Whippany, N. J., in 1965. He is responsible for a group investigating signal processing techniques in underwater acoustics.

Dr. Bogert is a member of the Seismological Society of America, the Acoustical Society of America, the American Geophysical Union, and Sigma Xi.

Thomas G. Stockham, Jr. (S'55–M'60) was born in Passaic, N. J., on December 22, 1933. He received the S.B., S.M., and Sc.D. degrees in 1955, 1956, and 1959, respectively, from the Massachusetts Institute of Technology, Cambridge. His graduate education included work in communications theory, network theory, computer science, and linear system theory, with emphasis on the latter. His doctoral thesis involved research in the field of nonlinear signal processing.

Concurrent with his graduate studies, he was a Teaching Assistant from 1955 to 1957 and an Instructor from 1957 to 1959, receiving the Goodwin Teaching Award in 1957. In 1959, he was appointed Assistant Professor in the M.I.T. Department of Electrical Engineering. Since 1958 he has centered his research in the field of information processing. Early work included problems in computer graphics, systems for on-line symbolic debugging, and algorithms for the convenient efficient plotting of data. From 1962 to 1964, he participated in an elaborate experiment which successfully isolated the detailed effects of room acoustics from those of a loudspeaker by measuring the Green's function of a room and using it in a computer convolution with selections of speech and music. The desire to reduce the attendant computing times led to the development of the high-speed convolution algorithm. From 1966 to 1968, he was a Staff Member of the M.I.T. Lincoln Laboratory, where he examined the uses of the principle of generalized superposition in processing multiplied signals. In July, 1968, he was appointed Associate Professor of Electrical Engineering at the University of Utah, Salt Lake City, where he is continuing his work in digital waveform processing.

Dr. Stockham is a member of Tau Beta Pi, Sigma Xi, Eta Kappa Nu, and the Association for Computing Machinery.