Mikrochim. Acta [Wien] 1987, III, 33—45

Mikrochimica Acta

The Re-Discovery of the Fast Fourier Transform Algorithm*

James W. Cooley

Thomas J. Watson Research Center, Yorktown Heights, NY 10598, USA

Abstract. The discovery of the fast Fourier transform (FFT) algorithm and the subsequent development of algorithmic and numerical methods based on it have had an enormous impact on the ability of computers to process digital representations of signals, or functions. At first, the FFT was regarded as entirely new. However, attention and wide publicity led to an unfolding of its pre-electronic computer history going back to Gauss. The present paper describes the author's own involvement and experience with the FFT algorithm.

Key words: FFT, fast Fourier transform, DFT, discrete Fourier transform.

In 1952, a professor specializing in numerical analysis was asked why, with the advent of large high speed electronic computers, one should work on developing faster algorithms. The speed and size of these new machines should permit known algorithms to yield solutions with more than sufficient speed and economy. Recent studies of the history of the fast Fourier transform (FFT) algorithm, going back to Gauss [1], provide an example of exactly the opposite situation. After having been published and used over a period of 150 years without being regarded as having any particular importance, the FFT was re-discovered, developed extensively, and applied on electronic computers in 1965, creating a revolutionary change in the scale and types of problems amenable to digital processes. Thus, as in several other areas of numerical analysis, the advent of the electronic computer has stimulated the development of new algorithms which increase computing power by many orders of magnitude.

The 1965 publication of the paper [2] which has been credited with the "discovery" of the FFT algorithm produced three almost immediate responses:

(1) The algorithm was new and revolutionary and it opened up a new world of digital processing, increasing the power of Fourier methods by many orders of magnitude. No problem would be too large.

---

* The author is grateful for permission from the Association for Computing Machinery to allow the present paper to bear some similarity with the paper, *How the FFT Gained Acceptance,* ref. [28]

(2) The algorithm was not new; it had been known and used before.

(3) One of the first really good applications brought to me was too large. It was a calculation being planned by Janine Connes who was programming the calculation of Fourier transforms of interferometry data.

It was rather flattering to have my name associated with Tukey's and attached to an important algorithm. One reason for the use of our names was that our paper was where most people saw it first. Another, I suspect, was the interesting variety of permutations of the sounds in the names, resulting in names like the "Cool-Turkey algorithm" and the "Cookie-Toolie" algorithm.

I would like to describe here the events surrounding the publication of the FFT paper in 1965 and then discuss its history and applications. Before doing so, I will point out that high-speed electronic computers were in use for only about 12 years in 1965. There was a famous quotation made in the early 1950's when only a few electronic computers were in use which said that these could satisfy the computing needs of the whole world. In 1965, however, the existing computers were saturated with work and computers were sold and put into use as fast as they could be built. There was, at the time, another important technological development which increased the demand for high speed computation by many orders of magnitude. This was the development of analog-to-digital converters. These devices could produce digitized samples of a time-varying voltage several hundred thousand times per second. This meant that signals could be processed digitally instead of with analog devices. Digital signal processing (DSP) gave much better accuracy, it removed the constraint that the input-output relation was limited to what could be built with physical devices, and it permitted one to alter and optimize the input-output relation during processing. Then, with digital signal processing, came the need for numerical algorithms the most important of which were algorithms for Fourier transform and convolutions.

I have been told that this is a very mixed audience which includes specialists in many different fields and varying degrees of acquaintance with numerical processes. For those for whom the signal processing is done inside black boxes, I will simply define the Fourier transform which is at the focus of our attention. In one form, it is the expression of a "signal" or function in terms of sines and cosines, or in other words, its frequency components,

$$x(j) = \sum_j a(k)\cos(jk2\pi/N) + b(K)\sin(jk2\pi/N). \tag{1}$$

I am leaving out many details, but will say that this is the discrete Fourier transform in terms of what we may regard as sample values. Fourier theory includes cases where the sums are replaced by integrals and the functions may be piecewise continuous. Another form, which makes the notation and algebra simpler, is in complex notation,

$$x(j) = \sum_j \hat{x}(k) W^{jk}, \tag{2}$$

where

$$W = \exp(-2\pi i/N). \tag{3}$$

The first marvelous property of these transforms is that their inverses, or solutions, $a(k)$, $b(k)$, or $c(k)$ are given by almost the same formulas and can therefore be computed by the same algorithms. Even with this nice inversion property, when the number of terms in the series and the number of x's is N, the number of operations is N^2. The FFT algorithms which I am talking about here reduce this to $N\log(N)$, a speed-up of a factor of $N/\log(N)$. Modest sized sample records start at around $N = 1000$ meaning a speed-up factor of 100. The spectrometry calculation I will mention later had $N = 512000$ meaning that if one even considered the calculation by a direct method, the speed-up factor would be about 12800.

Of the many important theorems concerning Fourier transforms, the most important is probably the convolution theorem. The convolution is defined by

$$y(j) = \sum_n h(n)x(j-n) \tag{4}$$

and is also, in various forms, called a correlation or covariance function or just a moving average. This is expensive to compute. If the number of inputs and outputs are proportional to N, the number of operations will be proportional to N^2. The convolution theorem states that the Fourier transforms $\hat{y}$, $\hat{h}$, and $\hat{x}$ of y, h, and x, respectively, are related by

$$\hat{y}(k) = \hat{h}(k)\hat{x}(k), \tag{5}$$

which requires only N multiplications. Therefore, if one computes the Fourier transforms in a number of operations proportional to $N\log(N)$, then the number of operations for the convolution will be proportional to $N\log(N)$ instead of N^2.

Richard Garwin's Role in the FFT

Such was the state of affairs in late 1963 when I was recently hired at the new IBM Thomas J. Watson Research Center in Yorktown Heights, New York. I was working on a research project of my own when Richard Garwin[1] came to the computing center of the laboratory with some notes he made while with John Tukey at a meeting of President Kennedy's Scientific Advisory Committee, of which they were both members. John Tukey showed that if N, is a composite, $N = ab$, then the Fourier series can be expressed as an a-term series of subseries of b terms each. If one were computing all values of the series, this would reduce the number of operations from N^2 to $N(a+b)$. Tukey also said that if this were iterated, the number of operations would be proportional to $N\log(N)$ instead of N^2. Garwin knew that this was a very important calculation and he wanted to have this idea developed and applied.

---

[1] At that time, a staff member of the Watson Scientific Laboratory at Columbia University. Presently at IBM Watson Research Center, Yorktown Heights, NY

Garwin described his problem of determining the periodicities of the spin orientations in a 3-D crystal of He^3. Later, I found out that he was far more interested in improving the ability to do remote seismic monitoring of nuclear explosions since the Russians would not agree to inspections within their borders thereby hindering efforts at obtaining a nuclear test ban treaty. He also saw a need for the capability of long range acoustical detection of submarines. Like many others, I did not see the significance in this improvement and gave the job a little less priority than my own research. However, I was told of Garwin's reputation and, prodded by his occasional telephone calls (some of them to my manager), I produced a 3-dimensional FFT program. I put some effort into designing the algorithm so as to save storage and addressing by over-writing data and I spent some time working out a 3-dimensional indexing scheme that was combined with the indexing within the algorithm.

The Decision to Publish the FFT Algorithm

Garwin publicized the program at first through his many personal contacts, producing a small but increasing stream of requests for copies of it. I did a write-up and a version for a program library, but did not plan publishing right away. I gave a talk on the algorithm at a seminar series in our Department of Mathematical Sciences. Ken Iverson and Adin Falkoff, the developers of APL, participated and Howard Smith, a member of the APL group, put the algorithm in APL when it was only a language for defining processes and before it was implemented on any machine. This gave the algorithm a thorough working over at the other seminars.

Another participant in the Mathematics seminars was Frank Thomas, a mathematically-inclined patent attorney. I should clarify this by saying it was, and still is, not a common practice to have lawyers at Mathematics seminars. In any case, he was there and he suggested that there may be patent possibilities. He asked if the idea was new, so I called John Tukey and asked about it. John suggested that I look at some papers by F. Yates, G. E. P. Box, and I. J. Good. The last-mentioned had a full development of a very similar idea which differed in that the mapping from a series to subseries required that the factors of N be mutually prime. Good's algorithms [3] had advantages and disadvantages, the chief among the latter being that when the factors are mutually prime, they cannot all be small, meaning that the resulting algorithms will not be as fast. The best present-day subroutines contain program modules for the various factors of N which include both FFT and prime factor algorithms. A good example of this is the set of programs by Burrus and Eschenbacher [4].

With this news, a meeting was held to decide what to do with the FFT. It was suggested that the algorithm be put in the public domain and that this should be done by having Schmuel Winograd and Ray Miller design a device that could carry out the computation. My part of the strategy was to publish a paper with a footnote mentioning Winograd and Miller and their device. I did so and sent my draft copy to John Tukey, inviting him to be co-author. He made some changes and emendations, and added some refer-

ences. Next came the task of getting it published as quickly as possible. I offered it to *Mathematics of Computation* by sending it to Eugene Isaccson at the New York University Courant Institute of Mathematical Sciences, where I had worked before coming to IBM. The paper was published 8 months after submission in the April 1965 issue.

Thus, the paper made only one round trip between me and Tukey and our only collaboration was in a few telephone conversations. However, we had a previous collaboration in 1953 when Tukey was a consultant at John Von Neuman's computer project at the Institute for Advanced Study in Princeton, New Jersey, where I was a programmer. I programmed for him what later became the very popular Blackman-Tukey method of spectral analysis [5]. The important feature of this method was that it gave good smoothed statistical estimates of power spectra without requiring large Fourier transforms. Thus, our two collaborations were first on a method for avoiding large Fourier transforms since they were so costly and then a method for reducing the cost of the Fourier transforms.

The Radix 2 Algorithm

To give some insight into the FFT algorithms without too many details, one may describe the radix 2 algorithm as $\log_2(N)$ iterations, each of which sweeps through the data doing 2-point Fourier transforms. The radix 2 version of the algorithm as published by myself and Tukey is described by

$$X_1(k) = X_0(k) + X_0(k+d)W^e, \tag{6}$$

$$X_1(k+d) = X_0(k) - X_0(k+d)W^e, \tag{7}$$

where the exponent e is a simple function of the index k. The displacement d either starts at 2 and doubles on each iteration or starts at $N/2$ and is halved on each iteration. This can be described as a doubling algorithm. On the first iteration, one obtains a set of 2-point Fourier transforms. On the second, one adjusts half the points with a phase factor and produces a set of 4-point Fourier transforms. This is repeated, giving 8-point transforms and so on.

I wrote my program to overwrite data, thereby using no scratch storage, and saving 2 address formations. A consequence of this is that one has to do a "bit-reversal" permutation at the beginning or at the end. The above computation is commonly referred to as a "butterfly" due to its appearance when described by a signal flow graph, as in Fig. 1.

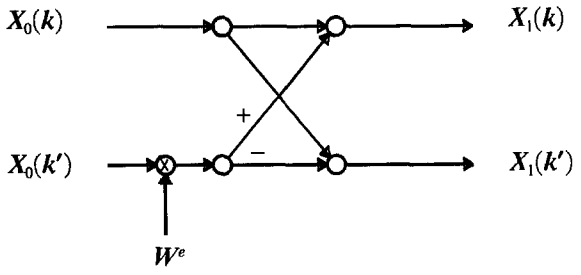

Fig. 1. Signal flow graph for the decimation-in-time radix 2 FFT algorithm. The appearance of this graph gave the name "butterfly" to the computation described by it

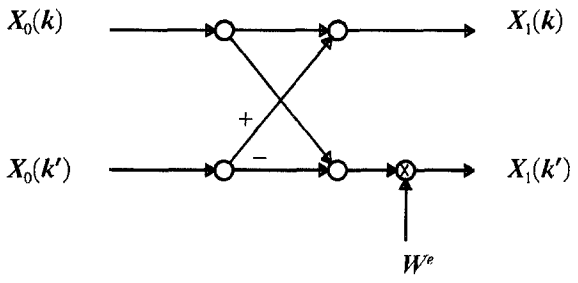

Fig. 2. Signal flow graph for the decimation-in-frequency (Sande-Tukey) radix 2 FFT algorithm

A statistics student of Tukey's, Gordon Sande, who was exposed to the factorization idea in one of Tukey's courses, derived a slightly different form of the algorithm which is described as follows:

$$X_1(k) = X_0(k) + X_0(k+d), \tag{8}$$

$$X_1(k+d) = [X_0(k) - X_0(k+d)]W^e. \tag{9}$$

This is known as the Sande-Tukey form of the FFT and is described by the signal flow graph in Fig. 2. The complex phase function W^e has been given the popular name "twiddle factor" by Sande. Both forms of the FFT algorithm are useful since one permits putting the bit-reversal at the beginning and the other permits doing it at the end. The total number of arithmetic operations is the same but one or the other may be preferred due to the particular architecture of the machine.

Public Relations and Merchandising of the FFT

Another member of our department, Lee Alsop, who was a geophysicist and adjunct professor at the Lamont Geophysical Laboratory of Columbia University decided to try the new algorithm on a record of 2048 samples of a strain seismograph of the Rat Island earthquake. He found that he not only got the promised speed-up ratio, but further testing with numerically generated data showed that the accuracy was much greater than that obtained by conventional methods. IBM made efforts to publicize the algorithm to the extent that it ran full page advertisements in Scientific American and a few other publication, describing Lee Alsop's results with the algorithm.

Then I was shown an excellent paper by Gordon Sande, in which he carried the subject further, showing how the fast Fourier transform (FFT) could be used to reduce computation in covariance calculations. After hearing about our paper going out to *Mathematics of Computation*, he did not publish his in its original form. However, he distributed many good subroutines and published an excellent paper with Morven Gentleman [6] on the algorithm and a Wilkinson type error analysis. Their results showed that on an average, the FFT error was lower by a factor which is the same as the speed-up factor, $N/\log(N)$, thereby substantiating what Lee Alsop found in his calculations. Thus, the new algorithm was not only faster but more accurate.

Almost simultaneously, Lloyd Welch, a mathematician at the Jet Propulsion Laboratory, Pasadena, California, designed an FFT algorithm by algebraic methods at the request of someone who needed it, independent of any previous ideas or suggestions. His work was only published in reports [7]. I have heard of several other discoveries of the FFT algorithm and I am sure that there were even more that I did not hear of.

Another result of Garwin's efforts was a seminar run at the IBM Watson Research Center to publicize the algorithm and familiarize IBMers with it. I say that I was lucky when I happened to be sitting where I was when Dick Garwin came along and I was lucky again when Hirsh Cohen, then department director, asked two excellent statisticians, Peter D. Welch and Peter A. W. Lewis, to join me in running the seminar. In a fairly short time we prepared notes which later became a thick research report describing the algorithm and developing some important theory and applications. This proved to be a very fertile field for research. The work for the seminar was extended and rewritten in a series of papers on the FFT and its applications. These papers elaborated on the theory of the discrete Fourier transform and showed how standard numerical methods should be revised as a result of the economy in the use of the FFT. These included a paper on the neglected theory of the finite Fourier transform [8], new methods for digital filtering [9] and spectral analysis [10] and [11]. The work on spectral analysis was essentially to make the revisions brought about by the reduced cost of the Fourier transforms, i.e., to re-write the Blackman-Tukey methods. This was a period of great productivity for many who had early contact with the FFT and were able to develop its applications. For me, it was also a very great education since it brought me in contact with many people with important applications. Furthermore, these people all thought I knew much more than I did, so I had to do a lot of studying to come closer to expectations.

Since the idea of solving the Poisson equation was of interest to me since the days of programming numerical weather forecasting, a paper on using Fourier methods for its solution, by Roger Hockney [12], caught my attention. He used the very clever cyclic reduction method combined with a form of the FFT algorithm, before seeing any of the FFT papers, and got very good results. I contacted him and eventually got IBM to hire him as a member of our Mathematics department where he did very good work in solving plasma problems, doing galaxy simulations and solving particle models for transistors using FFT subroutines [13].

The IEEE ASSP Digital Signal Processing Committee

The next level of activity came through contact with the speech and signal processing people at Massachusetts Institute of Technology. Tom Stockham telephoned me and excitedly told how important the algorithm was to them and what they were doing with it. Through him, I came into long-lasting and fruitful contacts with Charlie Rader, Ben Gold, Alan Oppenheim, Charles Rabiner, Ron Crochiere, Cliff Weinstein, to name just a few, all of whom are now important names in digital signal processing. They had

developed digital methods for processing speech, music, and images and the very great obstacle to making their methods feasible was the amount of computing required. This was the first really impressive evidence to me of the importance of the FFT. We invited the MIT people to the IBM Research Center where Stockham gave lectures and demonstrations showing how he "corrected" old Enrico Caruso records and did such things as removing violins from the background. Later, Stockham became founder of Soundstream, one of the two large digital recording companies in the United States.

I was invited to join them, some very inspiring Bell Laboratories people such as James Kaiser, Howard Helms, and others on the Digital Signal Processing Committee of the IEEE Society for Audio and Electroacoustics. This committee was undertaking the task of publicizing and developing the FFT algorithms. To reflect the shift in emphasis, the name was later changed to the IEEE Acoustics Speech and Signal Processing Society.

Fourier Spectrometry

One of the contacts which proved to have great mutual benefit was with Janine Connes, a member of the advisory committee of this conference. She first wrote to say that she had to compute Fourier transforms of very long data sequences. These were interferometer measurements made by a new and more powerful device built by her husband the astronomer, Pierre Connes. One extraordinary thing about this was that a single record of data was about 512 000 points and all values of the spectrum were needed. This was beyond the capacity of the high speed memory of existing machines. Another novel aspect was that the data was a real even sequence, while the FFT algorithm, as written, was for complex sequences. Thus, there was a four-fold redundancy in the complex transform which had to be exploited due to the amount of data. At that time, people were doing transforms of real data by just running the algorithm with zeroes in the imaginary parts. It was somewhat wasteful, but the FFT brought such an improvement, it seemed insignificant with smaller problems. Janine Connes visited the IBM Research Center to discuss the FFT and its application to her problem. I told her that the IEEE DSP committee was planning a workshop for people like herself and suggested that she come.

The Arden House Workshops

Bill Lang, manager of the acoustics laboratory at IBM and then chairman of the DSP committee suggested that we run a workshop on the FFT in Arden House, the Harriman mansion on top of a mountain in Harriman Park, in New York. The first was in 1967 [14] and another was held in 1968 [15]. These were unique in several respects. One was that they included people from many different disciplines: There were heart surgeons, statisticians, geologists, university professors, and oceanographers, just to name a few. The common interest was in the use of the FFT algorithms and every one of the approximately 100 attending had something useful to say in his presentation. Another thing that was unique was that work was really done.

People got together to formulate and work out solutions to problems. Here, we got to work on Connes' problem. I and others had worked out algorithms which permitted one to do such tricks as doing two real FFT's with one complex FFT and for doing an N-point real transform via an $N/2$-point complex transform. At the workshop, I devised an algorithm for computing a real cosine transform, i.e. a transform of real even data. Norman Brenner, then a graduate student at MIT, working with Charles Rader, was a prolific programmer of all kinds of FFT's. He worked with Connes and designed a program that computed the FFT of a large sequence by scheduling the algorithm so it could use auxiliary storage with very little overhead. Since then, this has been done in many environments, including the use of magnetic drums, tapes, disks, and lately, hierarchical cache storage systems and parallel processors. These ideas and a monumental effort by Janine Connes resulted in her calculation of the infrared spectra of the planets which has become a standard reference book [16].

The History of the FFT

Meanwhile, I started learning the history of the FFT. Garwin questioned his colleague, Professor L. H. Thomas of the IBM Watson Scientific Laboratory of Columbia University, who had been my thesis advisor when I was a graduate student there. Thomas gave Garwin a paper he published in 1963 [17] which described a large Fourier series calculation he did in 1948 on IBM punched card machines: a tabulator and a multiplying punch. He said that he simply went to the library and looked up a method. He found a book by Karl Stumpff [18] that was a cook-book of methods for Fourier transforms of various sizes. Most of these used the symmetries of the trigonometric functions to reduce the number of computations by a constant factor. In a very few places Stumpff showed how to obtain larger transforms from smaller ones, and then left it to the reader to generalize. Thomas made a generalization that used mutually prime factors and got the same type of algorithm as I. J. Good. His computing scheme was for a special value of N for which this was a very efficient procedure but his paper describes the prime factor in very general terms.

Shortly after the publication of the *Mathematics of Computation* paper, a letter from Philip Rudnick of the Scripps Institution of Oceanography in San Diego, California, said that he had programmed the radix 2 algorithm using a method published by Danielson and Lanczos in 1942 [19] in the *Journal of the Franklin Institute*, a journal of great repute which publishes articles in all areas of science, but which did not enjoy a wide circulation among numerical analysts. Rudnick published improvements in the algorithm [20] in 1966. It was interesting to meet him and ask why he did not publish. He said that he was only interested in having a computer program to do his data analysis. This was another failure in the communication of good and useful ideas, the primary point of Garwin's 1969 Arden house keynote address [21].

I would like to mention two important concepts in numerical algorithms which, it is said, appeared first in the FFT algorithm.

— The first is the divide-and-conquer approach. If a large N-size problem requires effort that increases like some power of N, then one can, in general, reduce computation by dividing the problem into parts having the same structure.
— The second concept is the asymptotic behavior of the number of operations. This was not significant for small N so the importance of early forms of the FFT algorithms was not noticed even where they would have been very useful.

I can illustrate this point by going back to the Danielson and Lanczos paper [19]. They consider the problem of computing Fourier coefficients from a set of equally-spaced samples of a continuous function. It is not only a long laborious calculation, but one also has the problem of verifying accuracy. Errors could arise from mistakes in computing, which were frequent on desk calculators, or from undersampling the data. Lanczos pointed out that although the use of the symmetries of the trigonometric functions, as described by Runge, reduced computation, one still had an N^2 algorithm. In a previous reading of this paper, I obtained and published [22] the mistaken notion that Lanczos got the doubling idea from Runge. He really only attributes the use of symmetries to Runge, citing papers published in 1903 and 1905 which I could not find. The Stumpff paper [18] gave a reference to Runge and König [23] which contains the doubling algorithm but does not give it much prominence. Thus, it appears that Lanczos independently discovered the clever doubling algorithm, embodied in the butterflies above, and used it to solve the problems of both computational economy and error control. He writes, in the introduction to [19] on page 366, "We shall show that, by a certain transformation process, it is possible to double the number of ordinates with only slightly more than double the labor". He goes on to say:

> "In the technique of numerical analysis the following improvements suggested by Lanczos were used: (1) a simple matrix scheme for any even number of ordinates can be used in place of available standard forms; (2) a transposition of odd ordinates into even ordinates reduces an analysis for $2n$ coefficients to two analyses for n coefficients; (3) by using intermediate ordinates it is possible to estimate, before calculating any coefficients, the probable accuracy of the analysis; (4) any intermediate value of the Fourier integral can be determined from the calculated coefficients by interpolation. The first two improvements reduce the time spent in calculation and the probability of making errors, the third tests the accuracy of the analysis, and the fourth improvement allows the transform curve to be constructed with arbitrary exactness. Adopting these improvements the approximation times for Fourier analyses are: 10 min for 8 coefficients, 25 min for 16 coefficients, 60 min for 32 coefficients, and 140 min for 64 coefficients.[2]"

---

[2] Reprinted from the *Journal of the Franklin Institute,* Vol. 233, pp. 365—380 and pp. 435—452, with permission from Pergamon Journals Ltd, Elmsford, NY 10573, USA

Lanczos' matrix scheme in (1) reduces the data to even and odd components of the function so that real cosine and sine transforms may be computed. The rest of the process makes use of the symmetries of the sines and cosines, similar to the methods of Runge. After this, he uses the doubling algorithm. Step (2) is what we have been calling the twiddle factor multiplication and in Step (3) he does the butterfly calculation but observes accuracy by comparing the two inputs: the Fourier coefficients of the subseries. If they are the same, the high frequency half of the new transform is zero and, assuming that there are no frequency components beyond that, the sampling rate is sufficient and no errors have occurred. Thus, it appears that Lanczos had the FFT algorithm and if he had an electronic computer, he would have been ready to write a program permitting him to go to arbitrarily high power of 2. It seems strange, for us, to see his remark on p. 376, "If desired, this reduction process can be applied twice or three times."

This is an outstanding example of the difference in point of view between different generations of numerical analysts. They had the doubling algorithm, which was capable of doing Fourier transforms in $N\log(N)$ operations but its potential power did not seem to be appreciated. It was considered as much a method for checking accuracy as for reducing computing and many did not foresee the possibility of automating the procedure. Then, publishing it in the *Journal of the Franklin Institute* caused this important paper to go unnoticed until Philip Rudnick, who was not a numerical analyst, revived it. Rudnick also ignored the opportunity to expose it to the world. Lanczos later published his *Applied Analysis* in 1956 [24] with only a few words and a footnote (p. 239) referring to the Danielson and Lanczos paper and I find no references at all in his later books including his 1966 book, *Discourse on Fourier Series* [25].

Gauss and the FFT

After reading the above early papers, I wrote what I thought to be the history of the FFT algorithm [22] believing that I did not have to back earlier than to Runge. Some years later, while working on his book, Herman Goldstine [26] gave me a reference to a section of a book by Gauss [27] that contained the essentials of the FFT algorithm. I got a copy of the paper, which was in a neo-classic Latin that I could not read. The formulas and a slight recognition of a few words indicated he was doing a kind of Lagrangian interpolation that presumably would lead to the basic FFT algorithm. I put this aside as a very interesting post-retirement activity.

Several years later, some old signal processing friends, Don Johnson and Sidney Burrus at Rice University, told me that they and a very energetic graduate student, Michael Heideman, were studying Gauss and the FFT. Heideman not only got the paper translated but found and described many others who wrote of FFT methods, between Gauss and my early references.

Conclusion

This story of the FFT can be used to give one incentive to investigate not only new and novel approaches, but to occasionally look over old papers

and see the variety of tricks and clever ideas which were used when computing was, by itself, a laborious chore which gave clever people great incentive to develop efficient methods. Perhaps among the ideas discarded before the days of electronic computers, we may find more seeds of new algorithms.

Acknowledgement. I would like to thank Gerald Present of the IBM Thomas J. Watson Laboratory for reviewing the manuscript and suggesting numerous corrections at all levels. Since many names and events had to be recalled by memory, I am grateful to Richard L. Garwin and John W. Tukey for reviewing and correcting the factual information in the manuscript of this paper.

A paper similar to this was given at the ACM Conference [28]. Although the audience and the main points of the ACM paper were different, a small portion of the material is the same. Where similarities exist, they are published here with the kind permission of the Association for Computing Machinery. I also express my gratitude to Pergamon Journals, Inc. for permitting the quotations from the Danielson and Lanczos paper [19].

References

[1] M. T. Heideman, D. H. Johnson, C. S. Burrus, *The ASSP Magazine (Oct.)* **1984,** *1,* 14.

[2] J. W. Cooley, J. W. Tukey, *Math. Comp.* **1965,** *19,* 297.

[3] I. J. Good, *J. Royal Statist. Soc., Ser. B.* **1958,** *20,* 361; **1960,** *22,* 371 (MR 21 1674; MR 23 A4231).

[4] C. S. Burrus, P. W. Eschenbacher, *IEEE Trans. Acoust. Speech and Signal Processing* **1981,** *ASSP-29,* 806.

[5] R. B. Blackman, J. W. Tukey, *The Measurement of Power Spectra,* Dover, New York, 1959.

[6] W. M. Gentleman, G. Sande, *Fast Fourier Transforms for Fun and Profit* (1966 Fall Joint Computer Conf., AFIPS Proc. Vol. 29), Spartan, Washington, DC, 1966, pp. 563—578.

[7] L. R. Welch, *Computation of Finite Fourier Series* (Tech. Report No. SPS 37-37, Vol. 4), Jet Propulsion Lab, Pasadena, CA, 1966. Also, *A Program for Finite Fourier Transforms* (Tech. Report No. SPS 37-40, Vol. 3), Jet Propulsion Lab, Pasadena, CA, 1966.

[8] J. W. Cooley, P. A. Lewis, P. D. Welch, *IEEE Trans. Audio Electroacoustics* **1969,** *AU-17,* 77.

[9] J. W. Cooley, P. A. W. Lewis, P. D. Welch, *IEEE Trans. on Education* **1969,** *E-12,* 27.

[10] P. D. Welch, *IEEE Trans. Audio Electroacoustics* **1969,** *AU-15,* 209.

[11] J. W. Cooley, P. A. W. Lewis, P. D. Welch, *J. Sound Vib.* **1970,** *12,* 339.

[12] R. W. Hockney, *J. Assoc. Computing Machinery* **1965,** *12,* 95.

[13] R. W. Hockney, *The Potential Calculation and Some Applications* (Methods in Computational Physics, Vol. 9), Academic Press, New York, 1970; *I.B.M. Research Report R.C. 2870,* 1970.

[14] Special Issue on Fast Fourier Transform and Its Application to Digital Filtering and Spectral Analysis, *IEEE Trans. Audio Electroacoustics* **1967,** *AU-15,* 43.

[15] Special Issue on Fast Fourier Transform, *IEEE Trans. Audio Electroacoustics* **1969,** *AU-17,* 65.

[16] J. Connes, P. Connes, J.-P. Maillard, *Atlas des Spectres dans le Proche Infrarouge de Vénus, Mars, Jupiter et Saturne,* Éditions du Centre de la Recherche Scientifique, Paris, 1969.

[17] L. H. Thomas, in: *Applications of Digital Computers,* Ginn, Boston, 1963.

[18] K. Stumpff, *Grundlagen und Methoden der Periodenforschung,* Springer, Berlin, 1937; K. Stumpff, *Tafeln und Aufgaben zur Harmonischen Analyse und Periodogrammrechnung,* Springer, Berlin, 1939.
[19] G. C. Danielson, C. Lanczos, *J. Franklin Inst.* **1942,** *233,* 365 and 435.
[20] P. Rudnick, *Math. Comp.* **1966,** *20,* 429.
[21] R. L. Garwin, *IEEE Trans. Audio Electroacoustics* **1969,** *AU-17,* 69.
[22] J. W. Cooley, P. A. Lewis, P. D. Welch, *IEEE Trans. Audio Electroacoustics* **1967,** *AU-15,* 76.
[23] C. Runge, H. König, *Vorlesungen über Numerisches Rechnen* (Die Grundlehren der Mathematischen Wissenschaften, Band XI), Springer, Berlin, 1924.
[24] C. Lanczos, *Applied Analysis,* Prentice Hall, Englewood Cliffs, NJ, 1956.
[25] C. Lanczos, *Discourse on Fourier Series,* Oliver and Boyd, Edinburgh – London, 1966.
[26] H. H. Goldstine, *A History of Numerical Analysis from the 16th Through the 19th Century,* Springer, New York – Heidelberg – Berlin, 1977, pp. 249—253.
[27] C. F. Gauss, *Nachlaß: Theoria interpolationis methodo nova tractata* (Carl Friedrich Gauss, Werke, Band 3), Königliche Gesellschaft der Wissenschaften, Göttingen, 1866, pp. 265—303.
[28] J. W. Cooley, *How the FFT Gained Acceptance,* Proceedings of the Association for Computing Machinery Conference on the History of Scientific and Numeric Computation, Princeton, NJ, May 13—15, 1987.

Received August 24, 1987.