14.4: Appendix 4 - Programs for Short FFTs

This appendix will discuss efficient short FFT programs that can be used in both the The Cooley-Tukey Fast Fourier Transform Algorithm and the The Prime Factor and Winograd Fourier Transform Algorithms. Links and references are given to Fortran listings that can be used "as is" or put into the indexed loops of existing programs to give greater efficiency and/or a greater variety of allowed lengths. Special programs have been written for lengths: \(N=2,3,4,5,7,8,9,11,13,16,17,19,25,...\)

In the early days of the FFT, multiplication was done in software and was, therefore, much slower than an addition. With modern hardware, a floating point multiplication can be done in one clock cycle of the computer, microprocessor, or DSP chip, requiring the same time as an addition. Indeed, in some computers and many DSP chips, both a multiplication and an addition (or accumulation) can be done in one cycle while the indexing and memory access is done in parallel. Most of the algorithms described here are not hardware architecture specific but are designed to minimize both multiplications and additions.

The most basic and often used length FFT (or DFT) is for \(N=2\). In the Cooley Tukey FFT, it is called a "butterfly" and its reason for fame is requiring no multiplications at all, only one complex addition and one complex subtraction and needing only one complex temporary storage location. This is illustrated in Figure 1: The Prime Factor and Winograd Transform Algorithms and code is shown in Figure 2: The Prime Factor and Winograd Transform Algorithms. The second most used length is \(N=4\) because it is the only other short length requiring no multiplications and a minimum of additions. All other short FFT require some multiplication but for powers of two, \(N=8\) and \(N=16\) require few enough to be worth special coding for some situations.

Code for other short lengths such as the primes \(N=3,5,7,11,13,17,19\) and the composites \(N=9\) and \(N=25\) are included in the programs for the prime factor algorithm or the WFTA. They are derived using the theory in Chapters 5, 6, and 9.
If these short FFTs are used as modules in the basic prime factor algorithm (PFA), then the straightforward development used for the modules in Figure 17.12 are used. However if the more complicated indexing use to achieve in-order, in-place calculation used in \{xxxxx\} require different code.

For each of the indicated lengths, the computer code is given in a Connexions module.

They are not in the collection Fast Fourier Transforms as the printed version would be too long. However, one can link to them on-line from the following buttons: N=2 N=3 N=4 N=5 N=7 N=8 N=9 N=11 N=13 N=16 N=17 N=19 N=25 Versions for the in-place, in-order prime factor algorithm \{pfa\} can be obtained from: N=2 N=3 N=4 N=5 N=7 N=8 N=9 N=11 N=13 N=16 N=17 N=19 N=25 A technical report that describes the length 11, 13, 17, and 19 is in \{report 8105\} and another technical report that describes a program that will automatically generate a prime length FFT and its flow graph is in \{report xxx\}.

Contributor

- ContribEEBurrus