REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-01-0188

The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing the burden to Department of Defense, Washington Headquarters Services Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
30-06-1999	Professional Paper	

4. TITLE AND SUBTITLE
Comparing Traditional FFT Based Frequency Domain Excision with Poly-Phase Transform Excision

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHORS
Abusalem, Hana, SPAWAR Systems Center, San Diego
harris, fred, San Diego State University

5d. PROJECT NUMBER
NG01

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Space and Naval Warfare Systems Center
53560 Hull Street
San Diego, CA 92152-5001

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217-5660

10. SPONSOR/MONITOR'S ACRONYM(S)
ONR

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

20000907 053

13. SUPPLEMENTARY NOTES

14. ABSTRACT
This work is part of an ONR program to imbed DSP in next generation GPS receivers to mitigate GPS vulnerabilities. We address and compare a number of frequency domain methods for suppressing stationary and slowly sweeping narrow-band interferes. Frequency-domain interference excision techniques use the FFT to identify and excise undesired spectral components. The technique is known as hole-punching, a spectral gating operation that sets to zero those spectral components exceeding specified threshold levels.

Published in Proceedings of the 55th Annual ION Conference, Navigational Technology for the 21st Century, pp. 625-634, June 30, 1999.

15. SUBJECT TERMS
Navigation
Fast Fourier Transform (FFT) spread spectrum signal poly-phase
frequency domain hole-punching

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Abusalem, Hana, D73H
U	U	U	UU	10	19B. TELEPHONE NUMBER *(Include area code)* (619) 553-1728

DTIC QUALITY INSPECTED 4

Comparing Traditional FFT Based Frequency Domain Excision with Poly-Phase Transform Excision

Hana Abusalem, IEEE Senior Member
SPAWAR SYS Center, San Diego

fred harris, E&CE Professor
San Diego State University (SDSU)

BIOGRAPHIES: Hana Abusalem received a BSEE and an MSEE from SDSU. Her specialty area is Digital Signal Processing (DSP) with an emphasis in communication systems. She has worked on surveillance, radar, Advanced Computing, Asynchronous Transfer Mode, and GPS programs.

Professor fred harris is a recognized DSP expert in the areas of satellite and cable communications, radar, sonar, software radios, and acoustic surveillance systems. For 15 years, he was the only permanent part time employee at NOSC. He consulted for Lockheed, ESL, Cubic, Hughes, Rockwell, GTE, Brooktree, TRW, Comstream, Logic Devices, GDE, Motorola, Lucent, and Tektronix. He is a frequent keynote speaker at international conferences. He teaches in-house short courses on emerging technologies for companies and foreign and US universities.

ABSTRACT: This work is part of an ONR program to imbed DSP in next generation GPS receivers to mitigate GPS vulnerabilities. We address and compare a number of frequency domain methods for suppressing stationary and slowly sweeping narrow-band interferes. Frequency-domain interference excision techniques use the FFT to identify and excise undesired spectral components. The technique is known as hole-punching, a spectral gating operation that sets to zero those spectral components exceeding specified threshold levels.

The FFT of a signal often exhibits high spectral side-lobes due to the finite aperture processed interval. These side-lobes represent spectral splatter and behave much like a broad band interference source by contributing energy to all FFT spectral bins. Suppressing energy in the main-lobe spectral bins in not adequate due to the residual interference distributed through the spectrum by these side-lobes. Spectral splatter by high side-lobes is traditionally called spectral leakage. We demonstrate that the spectral leakage that survives the hole punching operation severely degrades the performance of the excision process.

Excision performance is improved substantially with a set of algorithms designed to reduce spectral side-lobe levels. We control the spectral side-lobes of the FFT by pre-processing the collected signal prior to the transform. In the spectral analysis community this control is accomplished by the use of data windows. Windows suppress spectral side-lobes at the expense of main-lobe widening and the rejection of data near the boundaries of the data collection interval. We compensate for this signal rejection by overlapping and processing successive intervals. For windows with good side-lobe rejection, the required overlap is 75%. This overlap seems to compound the processing task by requiring us to perform transforms four times as often. We avoid this increase in workload by processing all four overlapped windows simultaneously. The resulting process is called a poly-phase filter bank, and the combination of the poly-phase preprocessor with the FFT is termed a poly-phase transform.

FFT BASED TIME PROCESSING: The Fast Fourier Transform (FFT) is a class of algorithms that efficiently implement the Discrete Fourier Transform (DFT). The DFT, the discrete counterpart to the standard continuous domain Fourier Transform, is traditionally used to form the spectral description of a sampled data time domain signal. In a similar manner, the inverse DFT (IDFT) reverses the process to form the sampled data time series associated with a given spectral description.

Efficient FFT algorithms are readily available when the size of the transform is highly composite, that is, when the length N factors into many small primes (or powers of primes). The direct implementation of an N-point DFT requires N^2 complex operations. The computational workload to perform the same N-point transform as an FFT depends to first order on the number of factors of N, and to second order, on the values of the factors. Structures designed to implement an FFT algorithm range from variants of the Cooley-Tukey to variants of the Good-Thomas (prime factor) with overlays of the Winograd and Rader Transforms. In general, powers of two or four, such as 512 or 1024 (2^9 or 4^5) point Cooley-Tukey variants are the most popular due to the regularity of their butterfly and dragonfly structures and associated memory mapping. A close second in popularity are the mixed-radix Cooley-Tukey routines, which support lengths such as 500 or 1000 (2^2x5^3, or 2^3x5^3) points. These are nearly as efficient

as the single radix versions due to the small number of sub-structures. The third most popular are the mixed radix prime factor routines which support lengths such as 504 or 1023 (2^3x3^2x7, or 3x11x31) points. These are less efficient due to the larger number of sub-structures or due to the larger valued primes.

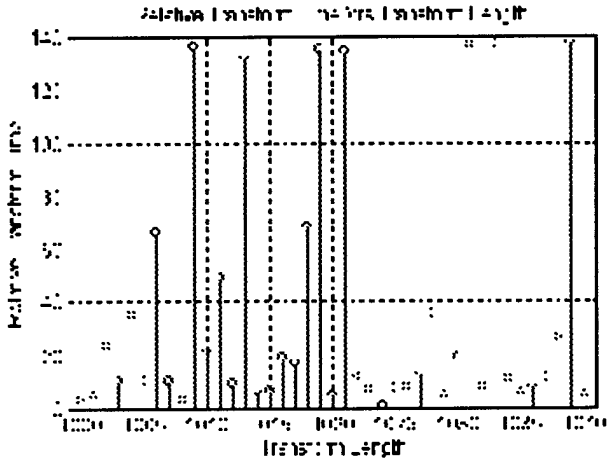

Figure 1. Relative Transform Time Versus Length.

Figure 1 presents a comparison of the relative time duration required to perform an FFT of different lengths of approximately 1K. Here the relative times are referenced to the time required to perform a 1024 point transform. Denoting this reference as 1-time unit, a 1023 point transform requires 6.7 time units, a 1000 point transform requires 3.1 time units, and the (prime length) 1021 point transform requires 140.1 time units. Efficient transforms are available for lengths other than powers of 2. These non-traditional transform sizes can offer certain system advantages.

The ease with which we can move our description of a time signal from the time domain to the frequency domain and back again has made the FFT a core signal processing function in the digital signal processing (DSP) community. A common signal-processing task performed with the FFT that takes advantage of its computational efficiency is that of fast convolution. In this process, the time signal is first transformed to the frequency domain to obtain its spectral description. Here the spectrum is modified using a point-by-point spectral product to form the spectral description of the output time series. This (modified) spectral description is processed by an inverse FFT to obtain the modified time series. This indirect process to perform a convolution is akin to performing an arithmetic multiplication in log-space where the operation (addition) is easier to implement. We compactly describe this process by the statement that a convolution in the time domain is equivalent to a product in the frequency domain.

A note of caution is called for here. Processing consisting of a transform, a spectral product, and an inverse transform in the continuous time domain results in a linear convolution. The same set of operations, consisting of the DFT, a spectral product, and an IDFT results in a circular convolution. When the desired result of the transform based processing is a linear convolution, a mechanism to control time domain wrap-around or aliasing must be imbedded in the block processing of the FFT. Standard control techniques include zero extending, overlap and add, and overlap and discard processing.

The convolution process normally requires M complex products per output point to implement an arbitrary filter of length M. When M is sufficiently large, the transform based fast convolution process offers a more efficient implementation. The computational cost of performing a convolution through the transform can be estimated as follows: (N/2) log2(N) complex operations for the input transform, N complex operations for the spectral product, and (N/2) log2(N) complex operations for the output transform. Adding these contributions, we arrive at the workload for the transform-based convolver as N log2(2N) complex operations. Accounting for the circular convolution properties of this process, M points of the N-output points are not useful, thus the number of useful outputs for the transform-based process is N-M. The workload per output point (in units of complex operations per output) is the ratio (N log2(2N))/(N-M). When the transform length N is large compared to the filter length M, this workload ratio is approximately log2(2N). When N=1024, this workload is 20. Thus we conclude that when a convolver is operating with filter lengths exceeding approximately 20 points, the transform-based implementation offers a significant computational advantage. The actual crossover point between the two options may vary due to secondary considerations such as using filters with real or with symmetric weights as well as length and efficiency of the selected transform. A safe criteria is that the crossover point is somewhere between 16 and 32 points: when the filter is shorter than 16 point, do it directly, when it is greater than 32 point, use the transform, and in between, it's probably a wash.

HOLE POKING WITH FFT: Hole poking is a simplified realization of a circular convolution process. Spectral modification of the hole-punch operation is likened to an ideal filter applied multiplicatively in the frequency domain. The equivalent filter exhibits zero-gain in the stop-band and unity gain elsewhere. It almost sounds too good to be true: a perfect filter! The reality of course is that it is too good to be true. Setting the spectral response of an FFT bin to zero does not guarantee that the filter response will be zero or even adequate in the neighborhood of that spectral zero. In fact the interval between spectral zeros exhibits the high side-lobes associated with sin(x)/x which can be as high as 0.22 (-13 dB) relative to pass-

band gain of unity. A typical spectral response of a hole-punch filter defined on and between the uniform grid of an FFT is shown in figure 2.

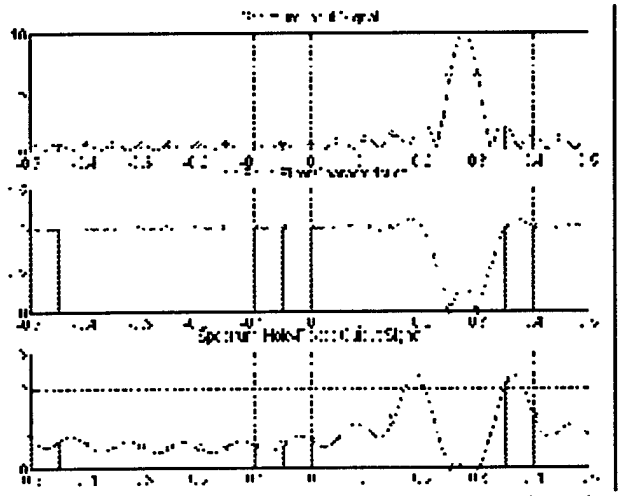

Figure 2. Spectral Sequence: Input, Hole Punch, and Hole-Punched Output

The top section of this figure presents the spectrum of an input sinusoid positioned midway between FFT spectral bin centers. Due to this positioning, the spectral support of the input time signal is not limited to the immediate neighborhood of its peaks. Gating the peaks with the hole-punch filter (shown in the center figure) does not affect the input spectral side-lobes. As seen in the bottom figure, the peak amplitude of the surviving side-lobes is approximately 20% of the original input spectral level.

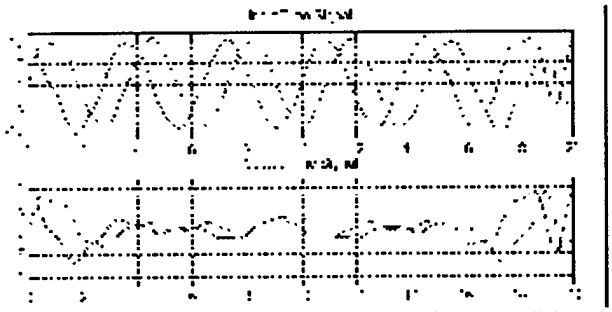

Figure 3. Input and Output Time Series from Traditional Hole-Punching Process Corresponding to Figure 2.

Figure 3 presents the input time signal that was presented to the processing represented by figure 2. Also seen is the output signal obtained from that processing. Note that while there is significant reduction in the overall amplitude of the output signal there are sections of the time-response that still has significant amplitude. In particular the input signal amplitude is 10 and sections of the output signal still has amplitudes equal to 10. Figure 4 presents the spectral representation of a hole-punch process applied to windowed input data. Note that in response to the window, the input spectrum has reduced side-lobes with a slightly wider main-lobe. For the same hole-poke gating shown in figure 2, the residual response of the gated spectrum has peak amplitude of 0.7 (as opposed to 2.2 in figure 2). The widening of the main-lobe suggests the gating function should also be widened and in an actual system we would indeed do that.

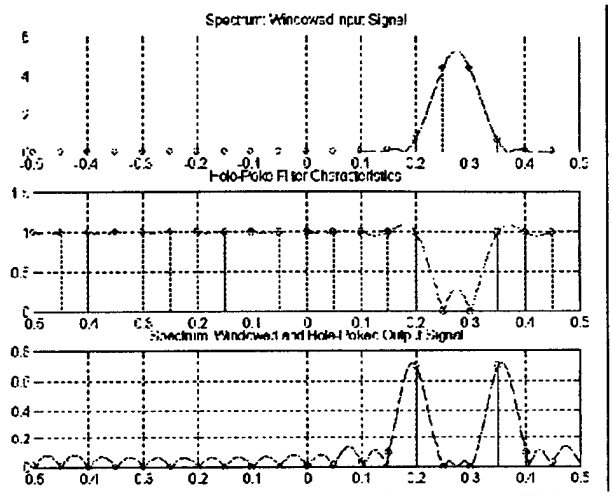

Figure 4. Spectral Sequence: Windowed Input, Hole Punch, and Windowed Hole-Punched Output

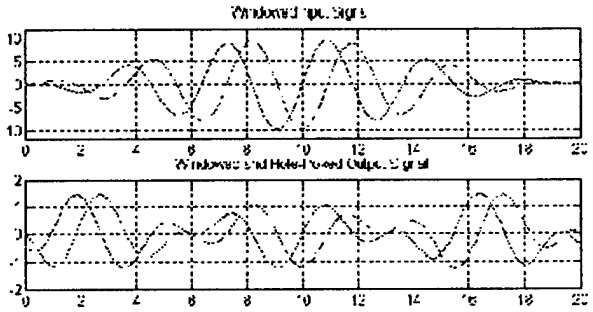

Figure 5. Input and Output Time Series from Windowed Hole-Punching Process Corresponding to Figure 4.

Figure 5 presents the input time signal that was presented to the processing represented by in figure 4. Also seen is the output signal obtained from that processing. This figure should be compared to the equivalent signal set shown in figure 3. Note the significant reduction in the overall amplitude of the output signal time-response. We must also remember that the window achieves this performance by discarding input data near the data boundaries.

In the previous paragraph we mentioned that widening of the spectral main-lobe due to the window suggests use of a wider hole-punch gating function. Widening the hole-punch by one index on each side resulted, for this example, in a peak residual side-lobe of 0.04 and of time response of peak amplitude 0.1. A Kaiser-Bessel window with –40 dB side-lobes was used for this demonstration. Operational systems require additional side-lobe rejection which causes an increase in main-lobe bandwidth.

HOLE POKING AS GPS OVERLAY: In the last section we demonstrated that a windowed transform hole-poking procedure offered significant performance advantages relative to the non-windowed transform. We now demonstrate this advantage and observe an undesired reduction in processing gain of the compression code due to the window. We will follow this with a number of overlapped processing techniques to purchase back the loss due to the windowing.

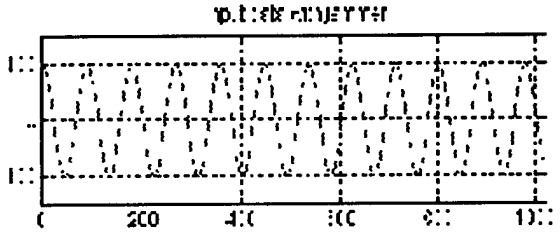

Figure 6: Binary Gold Code with 40 dB Jammer

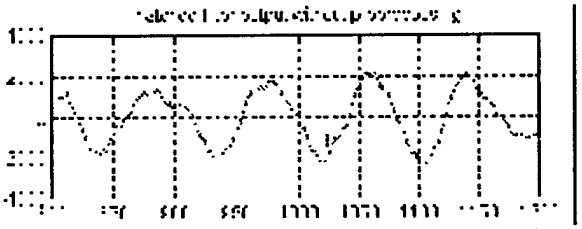

Figure 7: Output of Matched Filter: No Preprocessing

Figure 6 presents a time signal consisting of a sinusoidal jammer of amplitude 100 added to binary-phase Gold code of amplitude 1 for a jammer to signal ratio of 40 dB. Figure 7 presents the result of a matched filter processing the underlying binary phase Gold Code. No attempt is made here to account for analog-to-digital quantization effects or receiver noise. Note the little bump at index 1023 is the desired compressed response from the matched filter. The remainder of the response is dominated by the additive interference.

Figure 8 presents a zoomed section of spectra obtained by the FFT processing of the non-windowed and windowed data. Note the wider spectral width formed by the non-windowed spectrum. The figures also present the spectral gating interval to be removed by the hole-punch processing,

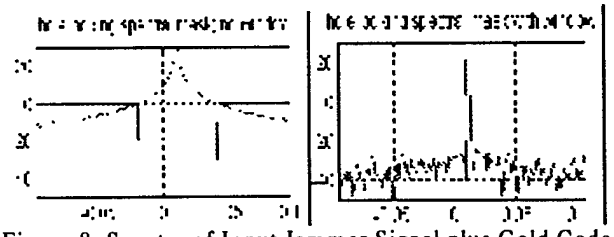

Figure 8. Spectra of Input Jammer Signal plus Gold Code Obtained without and With Data Window

Figure 9 present the outputs of the matched filter obtained after the hole punching operations. The figures are the result of processing the non-windowed and windowed hole-punch respectively.

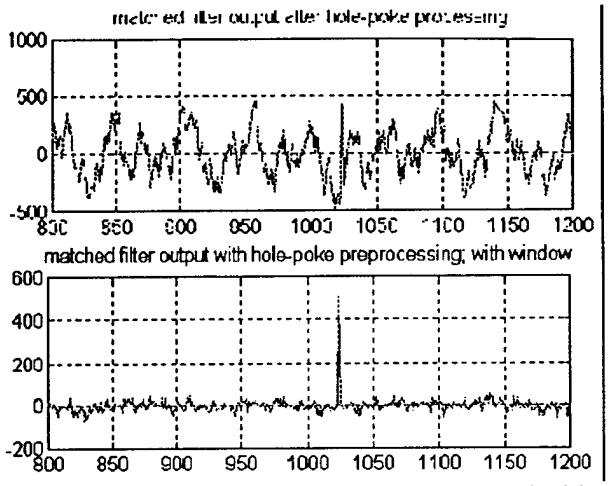

Figure 9. Output of Matched Filter: Preprocessed with Non-Windowed and Windowed Hole-Punch Routines

Note the impressive rejection capability of the windowed hole-punch process. The penalty paid by the windowed scheme is the reduction in processing gain due to the windowing of the received code as well as the received jamming signal. The expected output level from the matched filter is 1023, but the windowed response is 507, a loss related to the average value of the applied window.

OVERLAPPED PROCESSING: The transform used in the hole-punch routine is a block process. The desired result of the processing is a continuous flow of sampled data representing a filtered version of the collected data. Our next task is to convert the block process to a seamlessly merged set of blocks that emulate a continuous flow process. Figure 10 is a visualization of the time line and possible partitions of data into blocks to be processed and merged by the hole-punch routine. The first option is to collect contiguous blocks, process them and reassemble their outputs into contiguous blocks. This is similar to laying bricks. We must exercise boundary control by applying windows to each data processing interval. If we apply the windows to the contiguous intervals as shown in the second option the data located where the window approaches zero is essentially discarded by the process. To recover this data we overlap the intervals to be processed and then assemble the final output by summing the processed overlapped intervals. This process is called merged overlapped windows. These techniques is similar to the overlap-and-add process used in fast convolution but differs in the use of a data window in each interval, a operation that is not used in standard overlap, and add processing.

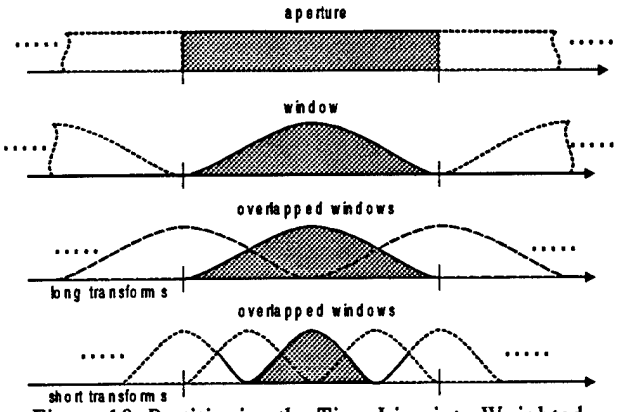

Figure 10. Partitioning the Time Line into Weighted and Overlapped Processing Intervals

The intervals selected for processing can be long or short, relative to some convenient interval such as the time interval required for a 1-K transform. The selection criterion is a function of the stability of the interference signal. Shorter processing intervals offer a small number of wide bandwidth spectral filters while long intervals offer a large number of narrow bandwidth filters to the hole-punch process. Figure 11 presents the input and overlapped windowed time interval partition presented to 1-K transforms.

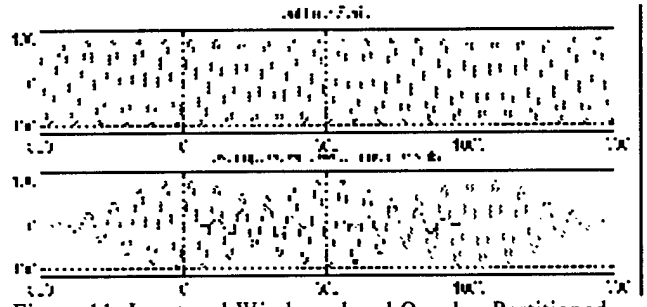
Figure 11. Input and Windowed and Overlap Partitioned Data for Successive 1-K Transforms

The hole-punch algorithm processes each interval and the resulting time series is assembled on the time line as overlapped contributions to the output time series. These overlapped intervals are merged by simple addition and are presented to the matched filter processing for subsequent Doppler-removal and Gold-code compression. Figure 12 presents the real part of the output signal over a section of the time line obtained from the overlapped, windowed, and merged processing. The input data here was a single interfering signal of amplitude 100 centered between 2-FFT bins. The binary phase Gold code can easily be seen on the residual interference of approximate amplitude 1. This represents a 100-to-1 reduction in amplitude to the interfering signal. This is the expected level of suppression that can be obtained from the Hann window selected for this demonstration. Additional suppression can be obtained with better windows but we would require additional overlap on the order of 75% rather than the 50% used here.

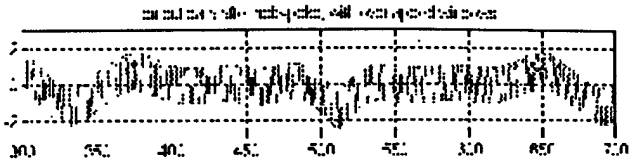
Figure 12. Time Response over Section of Time Line Showing Spreading Code and Residual Interference

Figure 13 presents the output of the matched filter after processing the overlapped, windowed, and merged data intervals with the hole-poking process. As expected, the overlapped processing recovered the processing gain loss incurred by applying the window to the data as well as to the interference signal. Note that the peak output level of the filter is expected to be the code length of 1023 and for this example the output was measured at 1022.

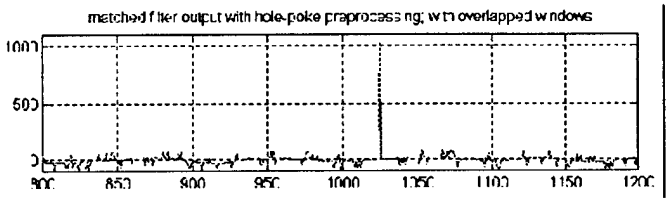

Figure 13. Matched Filter Output after Pre-Processing by Windowed, Overlapped, and Merged Hole-Puncher

The next set of figures demonstrates the performance of the overlapped, windowed, and merged processing using shorter transform intervals. Figure 14 presents the input and overlapped windowed time interval partitions presented to 256 point transforms.

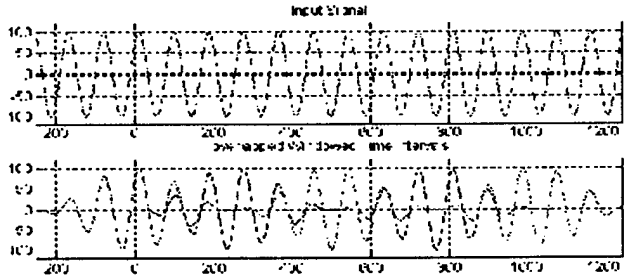
Figure 14. Input and Windowed and Overlap Partitioned Data for Successive 0.25-K Transforms

These transforms are one-fourth the length of the previous example consequently it takes 5 overlapped intervals to span the 1-K interval of the Gold code as opposed to 3 overlapped intervals for the previous example.

Figure 15 presents the real part of the output signal over a section of the time line obtained from the window, overlap, and merge processing. The input data here was the same as that used in the previous example that differs from this one only in the length of the processed intervals.

Here the binary phase Gold code can easily be seen on the residual interference of approximate amplitude 2.0. This is slightly less suppression than that obtained from the larger transform. This is due to the wider bandwidths of the equivalent filters presented to the hole-punch routine.

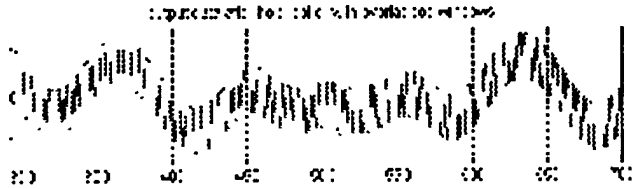

Figure 15. Time Response over Section of Time Line Showing Spreading Code and Residual Interference

Figure 16 presents the output of the matched filter after processing the shorter overlapped, windowed, and merged data intervals with the hole-poking process. Here too, the overlapped processing purchased back the loss we incurred by applying the window to the data as well as to the interference signal. For this example the output was measured at 1023. The small increase in residual interference obtained when using the shorter transform intervals had an insignificant effect on the side-lode levels observed in the output of the matched filter.

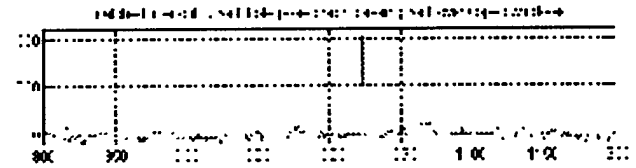

Figure 16. Matched Filter Output after Pre-Processing by Windowed, Overlapped, and Merged Hole-Puncher

WINDOW CONSIDERATIONS: We have demonstrated the value of windowing to control the spectral side-lobe response of the hole-punching algorithm. We have seen the need to overlap the windowed interval to avoid rejecting data contributions in the time intervals over which the window amplitude goes to zero. The data gathered in the overlapped intervals is processed in the individual intervals as if they were uncorrelated. Of course the intervals are correlated and we acknowledge that correlation by re-merging the overlapped intervals by adding the properly delayed and aligned responses to obtain the composite response. This overlap was described for 50% overlap in the text associated with figure 10 and was demonstrated in the text associated with figures 11 and 14. As mentioned earlier, for good side-lobe control the overlap is 75%. An intuitive argument justifying this 75% overlap is based on the Nyquist criterion and the main-lobe bandwidth of the window we use in the hole-poke process. A rectangle window of length N applied to data collected at sample rate f_s has a two-sided bandwidth of f_s/N. The Nyquist rate for data with this bandwidth is f_s/N which is 1/Nth of the input sample rate. This suggests that the output of the N-point transform should be sampled once for every N input points which is the non-overlapped processing scenario. The rectangle window has unacceptably high side-lobes of −13 dB relative to peak response.

We control the side-lobes by applying a window to the data prior to the N-point FFT. If we apply the Hann or the Hamming window to realize −32 dB or −44 dB peak side-lobe level we increase the main-lobe bandwidth by a factor of two to 2 f_s/N or $f_s/(N/2)$. The Nyquist rate for this windowed data is $f_s/(N/2)$ which is 1/(N/2) of the input sample rate. This suggests that the output of the N-point transform should be sampled once for every N/2 input points which is the 50% overlap-processing scenario.

To effect better side-lobe control we must select windows with better side-lobes. Good contenders are the 4-term Blackman-harris window and the Kaiser-Bessel window with time-bandwidth parameter B=11. These windows have peak side-lobes of −94 dB and −100 dB respectively. They also have a mainlobe width (measured from peak to −100 dB level) of 4 f_s/N or $f_s/(N/4)$. Following the argument we used in the previous paragraph, the Nyquist rate for this windowed data is $f_s/(N/4)$. This suggests that the output of the N-point transform should be sampled once for every N/4 input points which is the rationale for 75% overlap processing. Figure 17 shows the partition and the processing flow for the 75% overlapped intervals. In particular we see here the time locations of the partition and merging processes.

We note that any time interval of length N/4 samples in the input data steam contributes to 4-processing intervals and similarly any time interval of length N/4 samples in the output data stream is the result of contributions from 4-processing intervals. We also note that the four contributors to and from the processing tasks are weighted by different quarter sections of the window and as each fourth of the input interval moves through the window in successive overlapped intervals it is eventually weighted by each of the four fourths. Specifically, if we examine the banded section of the overlapped intervals shown in figure 17 we see the following. When a new one-quarter input interval (of length N/4 samples) is brought into the processing interval it is weighted by the first fourth of the window. When the next quarter is introduced it is weighted by the first fourth of the window while the previous quarter interval, previously the newest quarter and subsequently weighted by the first fourth of the weighting function, is now weighted by the second fourth. In like manner as each new quarter interval is introduced into the processing interval the previous intervals slide into later sections of the data window. We can think of the intervals as sliding through the window in much the same way that sampled data slides through a FIR filter.

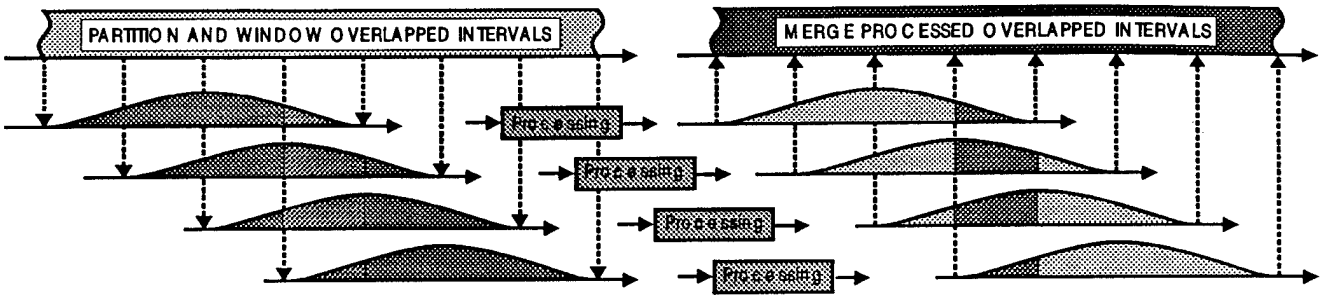

Figure 17. Schematic Representation Standard Processing of Input Interval Partition, Windowing, Processing, and Merging of Processed Data Intervals to Form Data Set in Final Output Series

An interesting awareness concerning constraints on the weighting function is related to the question, "What if the processing, indicated by the processing blocks in figure 17, were simple identity operators"? That is, if we did nothing to the input signal except partition and apply position and partition dependent weights could we perfectly reconstruct the signal? The answer is yes if the folded summation of the four segments of the weighting function is a constant. A sufficient condition for perfect reconstruction is that the weighting term be odd symmetric about the folding positions at N/4 and 3N/4. The symmetry condition just noted is illustrated in figure 18.

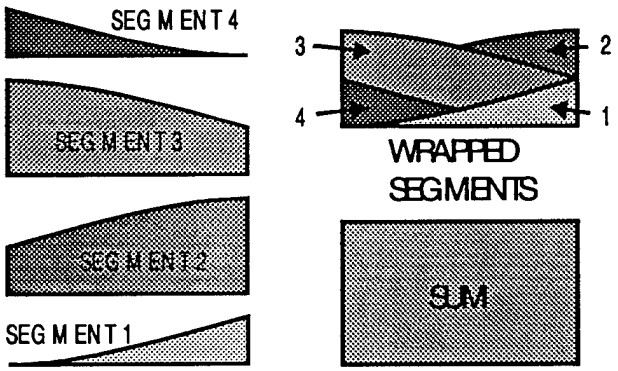

Figure 18. Merging Sections of 4-to-1 Overlapped Segmented Weighting Function by Simple Summation

The impact of the symmetry constraint just noted is that the spectral components of the window are restricted to DC and the fundamental of one cycle per interval. The window must be a raised cosine such as the Hann or Hamming weights. If other weighting functions are used we will have imperfect reconstruction of the time series which manifests itself as periodic modulation of the time-domain envelope. We can elect to live with the reconstruction artifacts or to suppress it with additional post-processing by a second filter equivalent to a deterministic equalizer.

POLYPHASE TRANSFORM: We now take a different approach to the windowed transform processing required to do the spectral hole-punching. Till now, our scenario has been, select an interval and transform it. The side-lobe response of the default rectangle window is unacceptable so we apply a weighting function with good spectral side-lobes to the data interval. The use of the window increased the main lobe width of the transform, an effect that we turn to our advantage in a later discussion. The window also discarded data near the boundary conditions of the interval. We responded to this loss with overlapped windows and a merging scheme. We now revisit the comment that applying the window increased the main-lobe spectral width. When we apply a good window, the increase in spectral width is a factor of 4. Consequently the spectral measurements performed by the FFT are highly correlated, hence represent redundant data. Prior to applying the window, the spacing between spectral centers in the N-point FFT was f_s/N and the bandwidth of the equivalent filter (the transform of the window) was also f_s/N, a good match! After we apply the window the spacing between spectral centers in the N-point FFT is still f_s/N but the bandwidth of the equivalent filter (the transform of the new window) is now $4f_s/N$, a poor match! It can be shown that the summation of the overlapped intervals represents additional processing that reduces the effective bandwidth of these filters while maintaining the desired good side-lobe levels. We can effect the same results with less computational effort by redefining the interval length as well as the window. Our thinking process is we first apply the window. The window widens the main-lobe spectral width to $4f_s/N$. We compensate for the increased numerator by offering a matching increased denominator by building a window of length 4N so that the mainlobe bandwidth is returned to f_s/N while realizing the desired good side-lobe levels.

The first problem we encounter is that the transform is now four times the original size. We must also remember we still have to use the overlapped intervals to avoid discarding data. Let's repair the first problem. If we use a

window of length 4N in a transform of length 4N we obtain spectral centers separated by $f_s/4N$. We originally selected the processing interval of length N because spectral spacing of f_s/N was appropriate to the signal processing task. We also know that the spectral measurements are highly correlated due to widening of the mainlobe width. That is the core of our fix. We choose to sample the output of the transform not at spectral spacing of fs/4N but rather at spacing fs/N so that the spacing and the filter width once again represent a good match. How do we do this spectral resampling? We simply skip samples, taking every fourth one that the 4N-point transform would offer us. This is shown in (1) where (1a) presents the transform points with poor side-lobes but with bandwidth and spacing f_s/N obtained with an N-point FFT. Equation (1b) presents the summation required to obtain the same f_s/N bandwidth with better side-lobes due to the 4N-point window. Finally (1c) and (1d) presents the spectral downsampling from $f_s/4N$ to f_s/N by computing every fourth output point.

$$F_N(k) = \sum_{n=0}^{N-1} f(n)\exp(-j\frac{2\pi}{N}nk) \qquad (1a)$$

$$F_{4N}(k) = \sum_{n=0}^{4N-1} W_{4N}(n) f(n)\exp(-j\frac{2\pi}{4N}nk) \qquad (1b)$$

$$F_{4N}(4k) = \sum_{n=0}^{4N-1} W_{4N}(n) f(n)\exp(-j\frac{2\pi}{4N}n4k) \qquad (1c)$$

$$= \sum_{n=0}^{4N-1} W_{4N}(n) f(n)\exp(-j\frac{2\pi}{N}nk) \qquad (1d)$$

Noting that the exponential kernel in (1d) is periodic in N samples, we can rewrite (1d) to obtain (2). We recognize that (2) is the sum of four transforms each weighted by sections of the window identified in figure 17. The data in the intervals shown in (2) are not the same as that indicated in figure 17. We could perform four short transforms of the segmented and weighted data and then merge them as suggested in figure 17.

$$F_{4N}(4k) = \sum_{n=0}^{N-1} W_{4N}(n) f(n)\exp(-j\frac{2\pi}{N}nk) + \qquad (2)$$

$$\sum_{n=0}^{N-1} W_{4N}(n+N) f(n+N)\exp(-j\frac{2\pi}{N}nk) +$$

$$\sum_{n=0}^{N-1} W_{4N}(n+2N) f(n+2N)\exp(-j\frac{2\pi}{N}nk) +$$

$$\sum_{n=0}^{N-1} W_{4N}(n+3N) f(n+3N)\exp(-j\frac{2\pi}{N}nk)$$

Since the transform is a linear operator we can rearrange the sum, and perform a transform of a sum rather than a sum of transforms. This is shown in (3) where the second summation in (3) that computes $P_4(n)$ is the poly-phase preprocessor that merges the weighted segments prior to the transform as opposed to after.

$$F_{4N}(4k) = \sum_{n=0}^{N-1} P_4(n)\exp(-j\frac{2\pi}{N}nk) \qquad (3)$$

$$where \quad P_4(n) = \sum_{r=0}^{3} f(n+rN) W_{4N}(n+rN)$$

The impact of this relationship is quite remarkable. We can perform non-overlapped N-point Transforms on 75% overlapped and windowed sequences of length 4N. This is shown in figure 19.

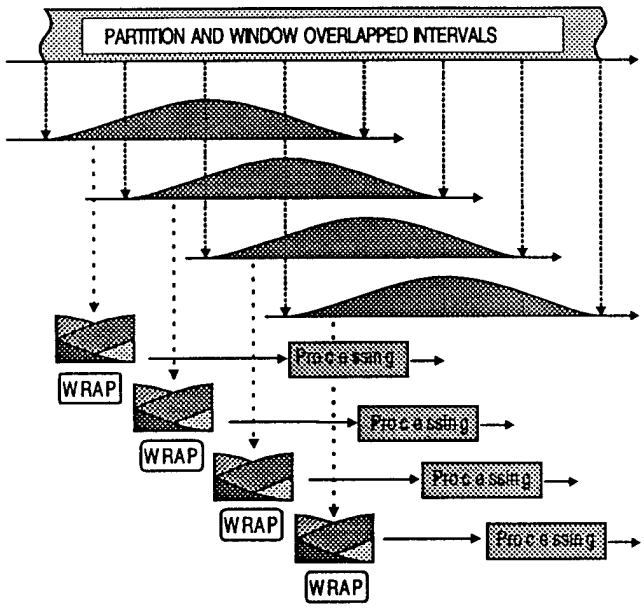

Figure 19. Processing of 4-to-1 Overlapped and Windowed 4N Point Intervals with Non-Overlapped N-Point Transforms

The process of wrapping a 4-N point weighted data set into an N-point Transform is called a poly-phase Transform. The preprocessing is often called data folding but it is in fact a data wrapping as can seen in figure 18. It is intentional time domain aliasing related to the 4-to-1 downsampling conducted in (1). Poly-phase is a common process used in communication systems to build high performance spectrum analyzers and orthogonal frequency division multiplexers. The process requires a poly-phase filter bank which pre-process short sequences of data in each bank and then passes the processed data vectors to the transform. This pre-processor is the second equation in (3).

After modifying the spectrum by the hole-punching process we have to return to the time domain. In the previous version of this process we returned with overlapping processed windows which had to be merged by addition of the separate data sets. In this method the data was merged prior to the transform so we ask "Are we finished"? Do we simply take the inverse Transform and merge the intervals by juxtapositioning them? Not quite! We have to perform the inverse of the input procedure. As expected, the inverse entails the inverse transform but it also includes another poly-phase filter bank that forms weighted sums across successive intervals to invert the folding process that occurred in the input process. The inversion would be perfect if the window had the symmetries described earlier. The windows or filter weights we use to control side-lobe levels contain additional spectral components so that the inverse of the input process does not result in perfect time domain reconstruction. This can be easily verified by skipping the forward and inverse transform and simply have the input poly-phase filter talk to the output poly-phase filter.

The reason we do not have perfect reconstruction is that we have replaced the Dirichlet kernels (the periodic sin(x)/x) equivalent filters obtained by the rectangle weight with better filters. The summation over the frequency domain of the translated Dirichlet kernels (the transform center frequencies) results in a perfectly flat or constant spectrum. The inverse transform of this constant spectrum is the impulse response of the system, which happens to be a single impulse. If we perform the summation of spectral filters formed by our windowed and down sampled process we find a periodic ripple or scalloping of the spectrum. This ripple in the frequency domain is observed in the time domain as pre-and post echoes separated by the transform length. Thus an impulse applied to the input of the poly-phase filter bank and then extracted from the output bank would be accompanied by three post echoes and three pre echoes (for 4-to-1 overlap). Only two of the echoes are significant and they can be reduced to any desired level by additional post processing with separate deterministic equalizers after each poly-phase output stage. What we have accomplished by this sequence of operations is the replacement of the output poly-phase stages with a new set obtained by convolving each stage with its deterministic equalizer. Figure 20 presents the impulse response of the standard poly-phase filter bank and the impulse response of the equalized set.

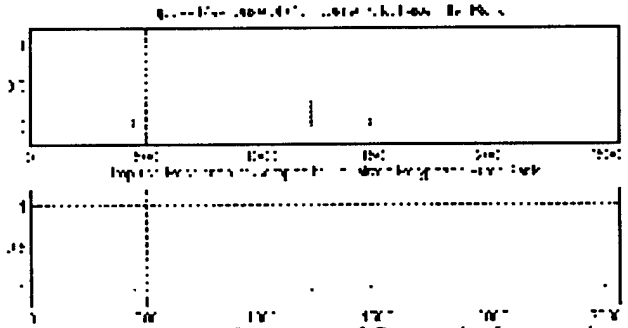

Figure 20. Impulse Response of Composite Input and Output Poly-phase Filter Banks and Equalized Banks

Figure 21 presents the time series obtained from the equalized poly-phase filter bank. The input interference was the same sinusoid of amplitude 100 used for earlier windowed hole-poking demonstrations. Note the interfering signal is completely suppressed and the binary phase Gold code can be seen in the output series. There is slight distortion in the output series due to the residual system echoes remaining from imperfect equalization.

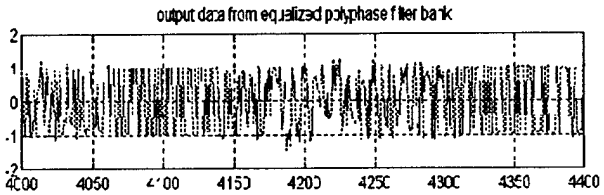

Figure 21. Time Response over Section of Time Line Showing Spreading Code and Residual Interference

Figure 22 presents the output of the matched filter after processing the output of the equalized poly-phase filter bank. As expected the full compressed signal is present along with the side-lobe levels of the despread code. Essentially no degradation results in the expected signal structure due to the hole-poking and the overlapped poly-phase processing. The expected peak output level is 1023, the measured response was 1004.

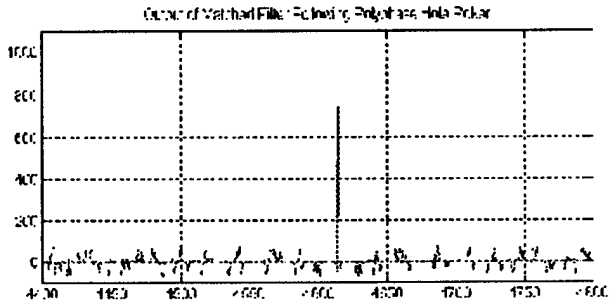

Figure 22. Matched Filter Output after Pre-Processing by Equalized Poly-Phase Transform Based Hole-Puncher

Finally, for completeness, figure 23 presents the output of the matched filter after processing the output of the non-equalized poly-phase filter bank. The compressed signal is present along with the side-lobe levels of the despread code. We also see degradation of the expected signal due to the system echoes. In particular note the two echoes at time indices 4613 and 5125. The Main peak response is at location 4869, so echoes are where they were expected at $\pm$ N (transform length N=256). The expected peak output level is 1023, the measured response was only 868.

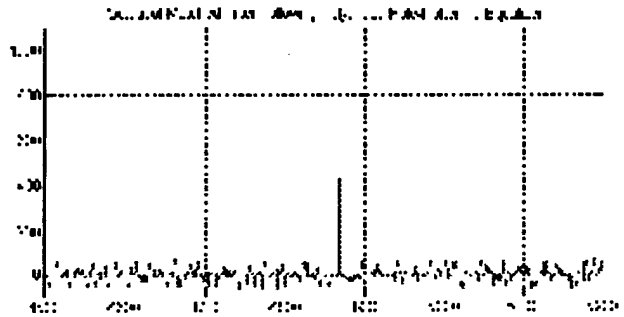

Figure 23. Matched Filter Output after Pre-Processing by Non-Equalized Poly-Phase Transform Based Hole-Puncher

CONCLUSIONS: We demonstrated a number of FFT based hole-punching routines to excise strong narrow-band interference signals co-received with a GPS spread spectrum signal. We showed the importance of boundary effects on spectral artifacts hence on the efficacy of the hole-punch process. Significant performance improvements resulted from the use of good windows. The processing loss of signal due to windowing was recovered by the use of overlapped and windowed processing intervals. We described two scenarios to address the merging of the overlapped intervals. In the first, we merged the data after the processing, In the second, we merged the data prior to the processing. The second process, the poly-phase filter scheme required a modification of the output stage to suppress pre-and post-echoes.

In this paper we presented one powerful application of the transform to excise strong spot interference. The transform can efficiently perform convolutions and correlations. In our proposed next generation receiver architecture we have access to the transform which we use to perform several tasks that are not implemented in current receivers. The transform assists in carrier acquisition by defining Doppler lines. It also assists in code acquisition by performing the entire code phase hypothesis test in a single look at the data. The transform is also used to invert multi-path channels by absorbing the channel inverse in the transform of the spreading code's matched filter. This work will be documented in forthcoming publications.

ACKNOWLEDGEMENT: We are grateful to Drs. Neil Gerr, Sherman Gee, and Peter Reynolds from the Office of Naval Research for support and funding this work under the Navigation block.

BIBLIOGRAPHY:

f. j. harris, "On the Use of Windows for Harmonic Analysis with the Discrete Fourier Transform", Proceedings of the IEEE. Jan. 1978, pp. 51-83.

f. j. harris, "The Discrete Fourier Transform Applied to Time Domain Signal Processing", IEEE Communication Magazine, May 1982, pp. 13-22.

f. j. harris, "On the Use of Overlapped and Windowed FFTs to Form Time Series with Arbitrary and Time Varying Spectra", 16th Annual Asilomar Conference on Circuits, Systems, and Computers, Pacific Grove, CA, Nov. 1982.

f. j. harris, "On Reconstructing Filtered Time Series From a Sequence of Overlapped, Windowed, and Folded Fast Fourier Transforms", 19th Annual Asilomar Conference on Circuits, Systems, and Computers, Pacific Grove, CA, 6-8 Nov. 1985.

R. E. Blahut, "Fast Algorithms for Digital Signal Processing", Chapter 4, Addison-Wesley, 1985.

f. j. harris, and K. Knapp, "Exact Reconstruction of Time Series with Arbitrary Bandwidths Formed by Merging Windowed DFT Spectral Bins", MILCOM'86, Monterey, CA, 5-9 October 1986.

D. F. Elliot, "Handbook of Digital Signal Processing", Chapters 6, 7, and 8, Academic Press, 1987.

f. j. harris, "On the Relationship Between Multirate Polyphase FIR Filters and Windowed, Overlapped, FFT Processing", 23rd Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA, 30-Oct. to 1-Nov. 1989.

P.P. Vaidyanathan, "Multirate Systems and Filter Banks", Chapter 10, Prentice-Hall, 1993.

f. j. harris and W. Lowdermilk, "Design and Performance of Fading Insensitive Orthogonal Frequency Division Multiplexing (OFDM) Using Polyphase Filtering Techniques", 30th Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA 28-30 Nov. 1996.

f. j. harris, "Spectral Analysis Windows", Wiley Encyclopedia of Electrical and Electronics Engineering", Vol-20, Wiley, 1999, pp. 88-105.