The Effects of Clutter-rejection Filtering on Estimating Weather Spectrum Parameters

by

W. T. Davis

Technical Report #10
July 28, 1989

Radar Systems Laboratory
Electrical and Computer Engineering Department
Clemson University
Clemson, SC 29634-0915

Windshear Detection Radar Signal Processing Studies
Grant NAG-1-928
Grant NGT-70055
National Aeronautics and Space Administration
Langley Research Center
Hampton, VA 23665

ABSTRACT

This thesis investigates the effects of clutter-rejection filtering on estimating the weather parameters from pulse Doppler radar measurement data. The pulse pair method of estimating the spectrum mean and spectrum width of the weather is emphasized. The loss of sensitivity, a measure of the signal power lost due to filtering, is also considered.

A flexible software tool developed to investigate these effects is described. It allows for simulated weather radar data, in which the user specifies an underlying truncated Gaussian spectrum, as well as for externally generated data which may be real or simulated. The filter may be implemented in either the time or the frequency domain.

The software tool is validated by comparing unfiltered spectrum mean and width estimates to their true values, and by reproducing previously published results. The effects on the weather parameter estimates using simulated weather-only data are evaluated for five filters: an "ideal" filter, two infinite impulse response filters, and two finite impulse response filters. Results considering external data, consisting of weather and clutter data, are evaluated on a range cell by range cell basis. Finally, it is shown theoretically and by computer simulation that a linear phase response is not required for a clutter rejection filter preceeding pulse-pair parameter estimation.

ACKNOWLEDGEMENTS

The author wishes to express sincere gratitude to several individuals who made contributions to this thesis. He first expresses thanks to his major advisor, Dr. E. G. Baxa, Jr., whose guidance and technical advice has been invaluable. He also thanks the National Aeronautics and Space Administration for their financial support, Langley Research Center personnel for their technical support, and his other committee members, Dr. J. Wilbur and Dr. C. B. Russell, for their time and effort.

The author is also appreciative of the advice and help of the the students of the Radar Systems Laboratory. Finally, he expresses sincere thanks to his friends and family for their continued support and encouragement through his graduate work.

TABLE OF CONTENTS

Table of Contents (Continued)

LIST OF FIGURES

List of Figures (Continued)

CHAPTER I

INTRODUCTION

Radar Detection of the Microburst Hazard

Microbursts are sudden downdrafts of highly turbulent air that, upon hitting the ground, cause powerful windshears in all directions. Microbursts, when occurring in the vicinity of an airport's runway, can be particularly hazardous to a large airliner, which responds slowly to the pilot's input, during its takeoff or landing. A landing aircraft entering a microburst, for example, may encounter head winds which increase its speed and lift, followed by a downdraft which causes it to sink, and then a tail wind which reduces its speed, all in a matter of seconds.

Microbursts are the main killers of U.S. airline passengers, contributing to at least 26 major airline accidents resulting in 626 deaths and over 200 injuries between 1964 and 1985 [1]. This fact has led the Federal Aviation Administration (FAA) and the National Aeronautics and Space Administration (NASA) to research and develop foward-looking means of microburst detection. The FAA is developing a ground-based detection system whereas NASA is developing an airborne-based detection system.

The pulse Doppler radar, which transmits a string of electromagnetic pulses and uses the Doppler frequency shift of the return pulses to determine the velocity of the air, is being considered as a way of obtaining quantitative information from turbulent wind conditions which require both a high range resolution and frequency resolution [2]. The conventional method of processing the radar-return

is on a range cell by range cell basis [1]. The pulse repetition frequency (PRF) of the radar must be high enough such that no velocity ambiguities exist for either the signal or clutter being considered [3]. The PRF determines the maximum unambiguous Doppler frequency, ±PRF/2, for each range cell, and the maximum unambiguous range of the radar, c/(2PRF), where c is the speed of light [2].

The distance that two targets must be separated in order to be distinguishible is called the range resolution. This distance is determined by the duration, T, of a single electromagnetic pulse. The minimum range resolution of the pulse Doppler radar is ±cT/2 [4].

In an airborne-based system, the radar return echos can be assumed stationary if the signal dwell time is below 100 milliseconds/sample [5]. The PRF constrains the number of samples in the airborne-based detection system. A PRF that is less than 6000 samples/second, for example, allows a maximum of 600 samples for the maximum signal dwell time of 100 milliseconds/sample assuming stationarity. The PRF also determines the real-time processing interval. If M represents the number of pulses, then the processing interval is M/PRF seconds.

The radar-return data must be processed in a way that meaningful information such as the energy, average velocity, and turbulence can be obtained. The first three moments of the power spectrum of the radar's in-phase and quadrature-phase (IQ) intermediate frequency output yields this information for each range cell. The zeroth moment is a measure of the water content within a range cell. The first and second moments, normalized to the zeroth moment, are measures of the mean velocity and the spread of velocities within a range cell. The first and second moments are commonly referred to as the spectrum mean and

spectrum width, respectively. The zeroth moment, or echo power, can be estimated by calculating the power of the radar-return data. The pulse-pair algorithm is currently being considered as a means of estimating the first and second moments of the Doppler spectrum [6]. This method is also called the autocovariance method because both the spectrum mean and spectrum width estimates are functions of the complex autocorrelation of the radar's IQ intermediate frequency output.

Another widely used method of estimating the first and second moments is the Fourier Transform method [5] in which the moments are computed from an estimate of the power spectrum obtained by using a discrete Fourier transform (DFT). These two methods have been discussed and compared with the general conclusion that the pulse-pair technique is superior to the Fourier Transform method [5].

The Clutter Problem

A radar-return signal generally includes undesired returns called clutter, in addition to any desired "target" returns. The description of the desired and undesired returns depends on the detection situation. Considering the problem of detecting an aircraft using a ground-based radar, for example, the wanted returns are the echos from that specific aircraft, whereas the unwanted returns can consist of echos from other aircraft, weather, birds, etc [7]. Considering the problem of detecting a microburst by means of an airborne-based radar, the desired returns are the weather echos, which are the echos from rain, dust, insects, or any other object that is trapped in and controlled by the air motion of the microburst. The clutter can consist of undesired returns from buildings, trees, cars and planes.

This clutter is commonly called ground clutter, because the radar returns are from objects on the ground.

The clutter can occur at a range exceeding the unambiguous range of the radar resulting in ambiguous clutter [8]. Ambiguous clutter is commonly called *n*th-time-around clutter, where the second-time-around clutter results comes from the first ambiguous range. Third-time-around clutter results from the second ambiguous range, and Nth-time-around clutter results from the (N-1)th ambiguous range. *N*th-time-around clutter degrades the ability to reject the clutter, and MTI-filtering methods are being researched to reject *n*th-time-around clutter [9]. No provision has been made in this research to reject the clutter outside the unambiguous range of the radar.

As the term implies, clutter interferes with the target detection by cluttering the radar return data with spurious returns. It also affects the radar's automatic gain control (AGC), limiting the radar's effective dynamic range, which affects the number of bits and word size of the analog-to-digital (A/D) converters. This limitation in the dynamic range can cause "misinterpretations of the thunderstorm heights and radial velocity measurements [10]." Clutter particularly degrades the pulse-pair spectrum mean and width estimates since the pulse-pair technique yields estimates which are averages of the entire Doppler spectrum [6]. Therefore, to obtain meaningful pulse-pair estimates of the microburst return alone, a means of rejecting the clutter is paramount before processing the radar IQ data. This problem has been addressed in earlier works [4,11].

Clutter rejection for Land-based Radars

Clutter rejection for a ground-based radar is frequently achieved by preprocessing the radar-return data using clutter maps based on the particular environment in the radar's vicinity and/or high-pass filtering based on an *apriori* knowledge of the clutter characteristics [12]. Both analog and digital clutter cancellers have been considered [6]. Advantages of analog clutter rejection include the fact that it would reduce the dynamic range requirement of the A/D converters following the filter [6]. A smaller number of bits is required to represent the reduced dynamic range. Advantages of digital clutter rejection include the fact that the filter can be controlled more accurately, it offers flexibility, and the implementation of its design is more economical [6]. Several digital FIR and elliptic filters have been suggested [6,11]. Typically, Fourier or pulse-pair methods are used on filtered radar data to estimate the the average windspeed and windspeed variation within a range cell [2,4].

Clutter rejection for Airborne-based Radars

This thesis is related to the problem of clutter rejection when attempting to remotely detect a microburst using an airborne-based pulsed Doppler radar. The pulse-pair technique is considered as a means to estimate the spectrum mean and width of the Doppler radar-return data. A high level of ground clutter is a problem in this application, particularly when the aircraft is operating at a low altitude in the vicinity of an urban terminal area as in the landing scenario. The clutter occurs when the radar antenna's mainlobe or sidelobe is directed at the ground. The sidelobe return can be eliminated through range-gating if the radar starts receiving the

return signals after the sidelode-return signal is dissipated [7]. Therefore, the mainlobe clutter is the bigger problem. For a narrow beam radar in a look-ahead situation, the clutter spectrum is generally centered at the ground speed of the aircraft. With knowledge of the aircraft's ground speed, the spectrum mean of the stationary clutter can be centered at the middle of the Doppler processing bandwidth through heterodyning in the radar's receiver.

Figure 1 provides a qualitative illustration of the ground clutter interference problem in terms of the magnitude of the Fourier transform of the radar IQ data. The spectrum of any particular range cell can contain both weather and clutter returns. Figure 1 illustrates two scenarios in terms of both the signal and clutter components of the return spectrum, which would actually be seen as the superposition of the two. The clutter is illustrated here as being concentrated in the vicinity of zero Doppler, where zero would typically represent the aircraft's ground speed. In Figure 1a, which represents one range cell, the signal and clutter largely occupy the same frequency range. The signal and clutter are separated in frequency in Figure 1b, which represents another range cell.

The basic problem is how the clutter can be removed before estimating the weather parameters. The spectral characteristics of the clutter may differ from that of the weather, allowing the removal of the clutter without destroying the integrity of the weather signal. Even if the total clutter power is greater than the total signal power, some form of notch filtering may be effective at reducing the clutter levels. The relationship between the spectrum mean of the clutter and the spectrum mean and width of the weather is important in determining

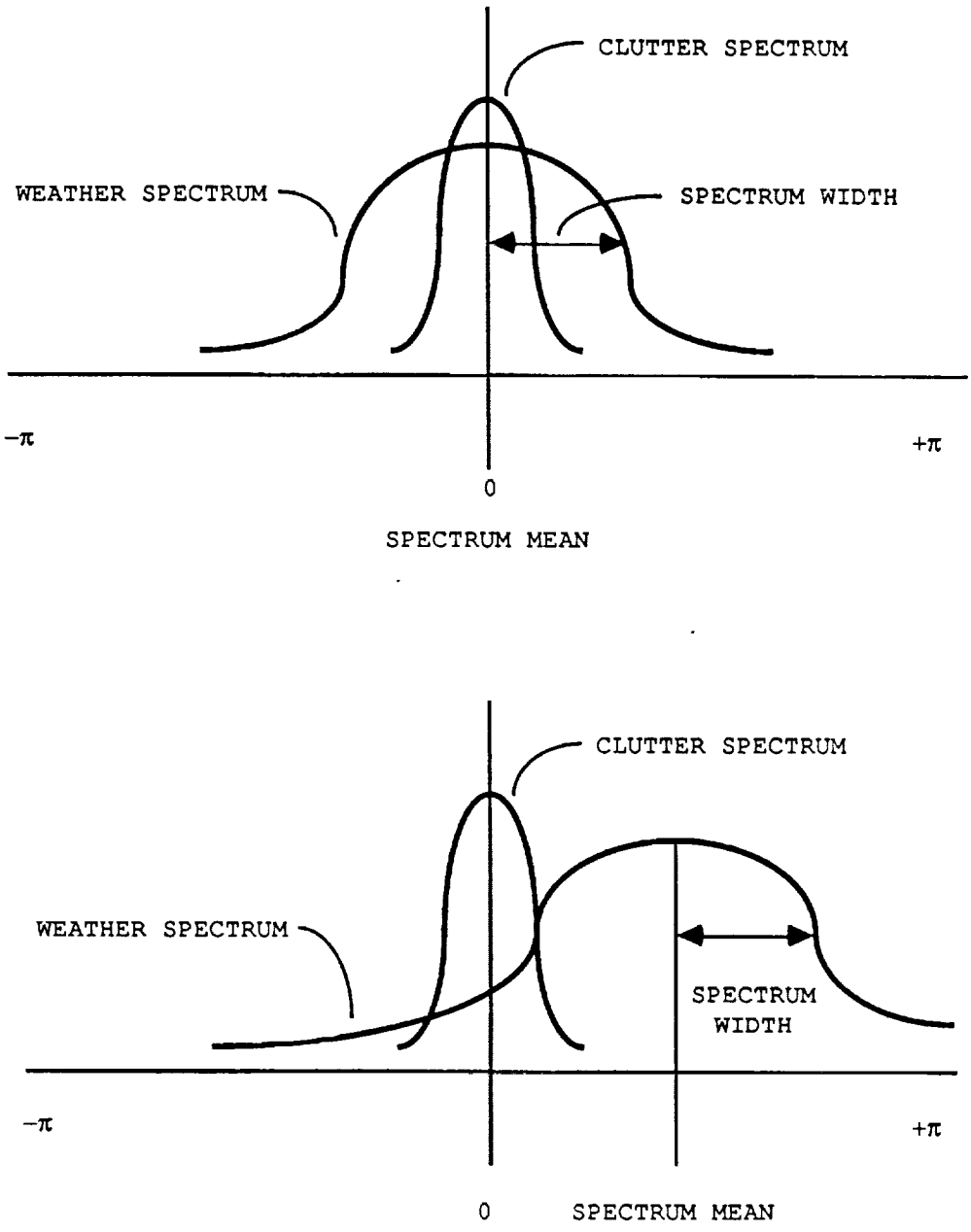


Figure 1. Illustration of Magnitude Response of the DFT of Radar-return Data

the effectiveness of notch filtering since clutter rejection almost always affects the weather data [3]. The loss of sensitivity , a measure of the the signal power lost due to filtering, is also important. In this thesis, the loss of sensitivity is defined as the ratio in decibels of the total signal power after filtering to the total signal power before filtering.

The clutter may also appear as discrete lines appearing throughout the Doppler bandwidth. When a busy interstate is located near the airport's runway, for example, discrete clutter can occur at frequencies determined by the relative velocity between the aircraft and highway traffic. Some form of frequency-domain filtering may be the most effective means of removing discrete clutter.

Any method of rejecting clutter from the radar's IQ data will degrade the weather signal. Understanding how clutter-rejection filtering affects the ability to derive useful weather information is important because a filter that is effective at getting rid of the clutter may also affect the signal to the extent that reliable measures of wind conditions are not possible.

Problem Statement

The clutter returns can frequently be discriminated from the weather returns based on the the frequency content. Rejecting the clutter using a notch filter while minimizing the filter's effects on the weather signal should be evaluated in terms of the filter's effect on the weather signal in addition to the abilily to reject the clutter. The effects of clutter-rejection filtering on the weather data can be evaluated in terms of the filter's effects on the pulse-pair estimates and loss of sensitivity.

Figure 2 illustrates an ideal notch filter centered at zero Doppler with two different weather spectrum situations: one with a mode at zero Doppler and another with a mode in the positive half of the spectrum. The weather spectrum is first filtered. The weather parameters are then estimated, and the estimates are compared to their true values. General conclusions on the filter's efects on the weather parameters are drawn from this analysis.

In Figure 2a, the loss of sensitivity is greatest when the weather spectrum mean is centered at zero frequency because of the high concentration of spectrum power in the filter's notch. Since the post-filter data is symmetrical about zero frequency, the post-filter mean estimate may not be affected. But since a large part of the mode is lost, the spectrum becomes more distributed. Therefore, the post-filter width estimate will be inflated. The variances of the post-filter estimates can be expected to be greater than the theoretical variances of the spectrum mean and width of the unfiltered data because of the large loss of sensitivity and the distributed nature of the post-filter data.

As the spectrum mean of the weather is increased for this case, the center of the weather spectrum moves outside the filter's notch as depicted in Figure 2b. Here, less signal power is lost through filtering, and the loss of sensitivity is not expected to be as great. The post-filter mean estimate is generally expected to be biased high because the filtered weather spectrum is skewed to the positive side of the spectrum. The post-filter spectrum becomes less distributed, and the width estimate will generally be less than the spectrum width of the unfiltered data.

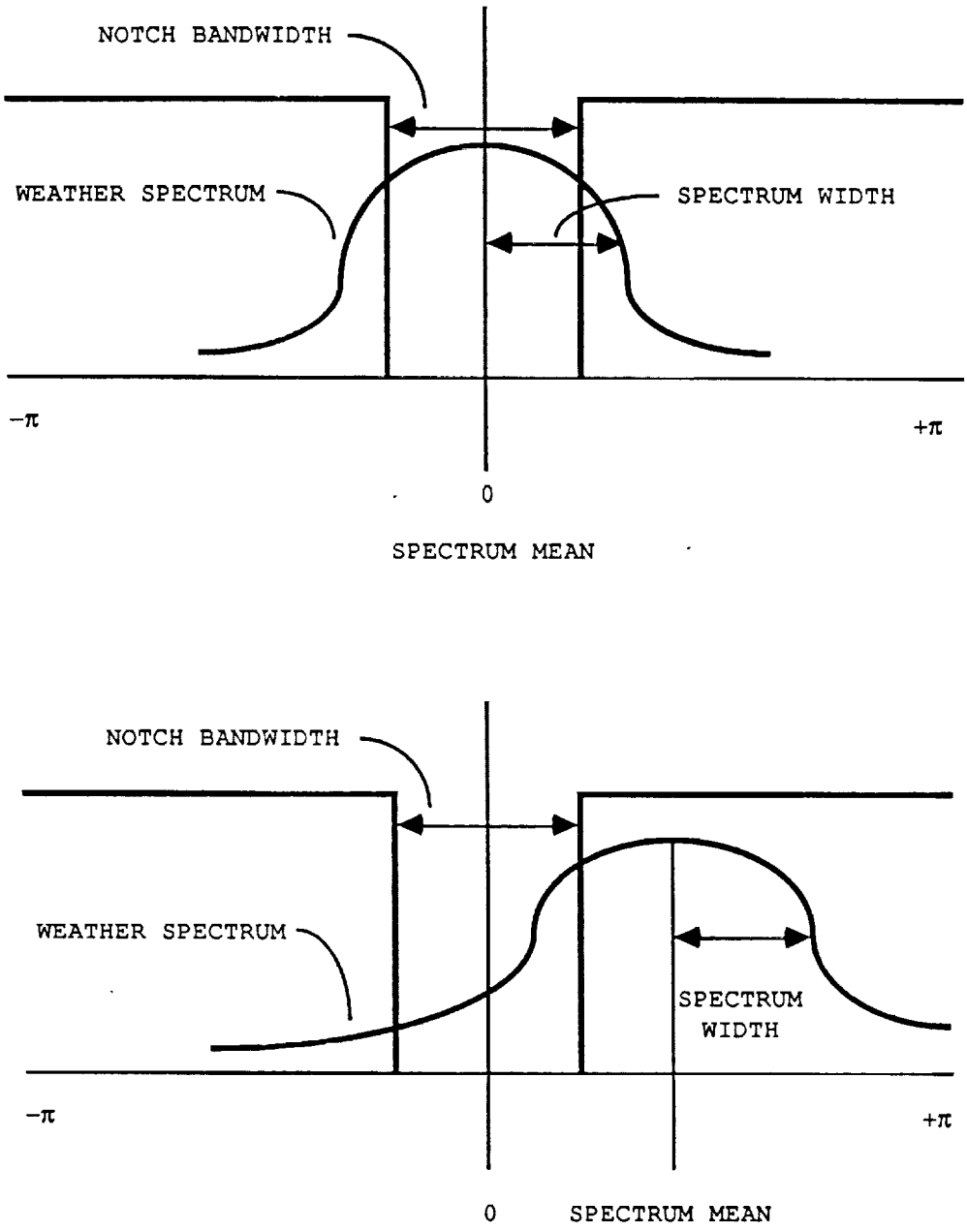


Figure 2. Ideal Notch Filter Superimposed on Weather Spectrum

As the spectrum mean of the weather is increased further for this case, less weather data is lost through filtering. The loss of sensitivity should approach zero, and the post-filter spectrum mean and width estimates should approach the spectrum mean and width of the unfiltered data. The post-filter variances of the spectrum mean and width estimates also should approach the theoretical variances of the spectrum mean and width of the unfiltered data.

Based on Figure 2, it is expected that the bias of the post-filter estimates introduced by filtering will be inflated as the notch bandwidth of the filter is increased. For the case where both the filter and weather spectrum are centered at zero frequency, for example, a larger notch bandwidth should cause more bias in the width estimate than a smaller notch bandwidth. Thus, it is anticipated that a bias in the the post-filter spectrum mean and width estimates will be associated with the loss of sensitivity and the distributed nature of the post-filter weather data. Anticipating and understanding these effects on turbulent weather detection is important when considering the best approach to clutter-rejection filtering.

This thesis will investigate this problem of the clutter-rejection filter's effects on the weather parameters using a computer analysis. A flexible software tool which has been developed for this purpose is described in Chapter II. Provision has been made to generate simulated weather data as well as make use of externally generated weather data which may be simulated or real. Clutter rejection filters may be implemented in the time domain or in the frequency domain.

Validation of the software tool is presented in Chapter III. The unfiltered spectrum mean and width estimates along with the variances

of each are compared to their true values. Previously published results investigating the loss of sensitivity, mean error, and width error are used to validate the analysis procedure.

Clutter-rejection filtering results using the software tool are presented in Chapter IV. Several infinite impulse response (IIR) filters and finite impulse response (FIR) filters are evaluated and compared to a baseline ideal filter. Each filter's unit-sample response, time-domain transient response, and frequency-domain magnitude and phase response are presented. The effects of the filter's magnitude and phase response on the post-filter estimates are independently evaluated. Selected results considering external data are presented to illustrate the use of the software analysis capability in evaluating actual radar data.

Chapter V contains the conclusions and recommendations for future work. Appendix A is included to provide a more complete description of the analysis software. Appendix B provides the data used to illustrate the capability of the analysis software to evaluate actual radar data.

CHAPTER II

DESCRIPTION OF SOFTWARE

Introduction

It is assumed that clutter-rejection filters will generally be implemented in the time domain, because the radar IQ data sequence is a complex time series, and digital filters are readily and efficiently implemented in terms of a difference equation. Additionally, the pulse-pair method of estimating weather parameters, which is emphasized, is based on the complex autocorrelation of the complex time series [2,13]. The implementation of the filter using a difference equation is, of course, an equivalent operation to linearly convolving the filter's unit-sample response with the complex time series [14].

Regardless of the implementation, the filter may be specified in the frequency domain. It may therefore be particularly expeditious for analysis purposes to implement the filter directly in the frequency domain. Since a DFT is used for the time/frequency transformations, the frequency-domain filtering is accomplished using preliminary transformations with zero padding before multiplying the filter's frequency response with the frequency-domain weather data [14]. The resulting final filtered sequence is the same sequence as if the filter's difference equation were used.

A given frequency characteristic may be implemented without regard to the equivalent linear convolution in the time domain, which is particularly useful when investigating an ideal filter. An ideal filter here is one that has a unity amplitude response in the passband(s) and a zero amplitude response in its stopband(s). The

phase response of this filter can be independently and arbitrarily specified. This filter is useful in providing a baseline for performance evaluations when compared to other filters. It also provides a ready means to implement "line-editing," in which selected frequency lines are zeroed, as a part of the filtering process. Line-editing may be particularly useful in eliminating discrete clutter; i.e., clutter power that is concentrated at particular isolated frequencies. This chapter describes a computer analysis tool that has been developed to evaluate clutter-rejection filters. A listing of the software is given in Appendix A.

Analysis Description

The computer analysis has been designed to determine the effects of clutter-rejection filtering on the ability to extract useful information from the Doppler radar weather return data. The data to be processed can be generated from a computer simulation which allows the underlying weather spectrum to be specified in terms of its spectrum mean and width assuming a Gaussian magnitude spectrum [5]. This procedure has been incorporated as part of the overall analysis package. The data can also be externally generated and simply read from a specified file for further analysis, allowing the use of real data or simulated data of any spectrum characteristic.

Provision has been made to operate on the data by filtering it in either the time domain or the frequency domain with any specified clutter rejection filter. The post-filter estimates of the spectrum mean and width are calculated using the pulse-pair algorithm and are compared to the specified mean and width. The standard deviation of each estimate is calculated and compared to theoretical values, and the

loss of sensitivity is computed. For any particular filter, the loss of sensitivity, S, in decibels is defined here as

$$S = -10\log_{10}\left[\frac{P_a}{P_b}\right] \tag{1}$$

where P_a is the total power of the post-filter radar IQ data, and P_b is the total power of the radar IQ data at the filter's input. Note that a positive value is associated with lost power at the filter's output.

The number of samples, which is restricted to a power of two because of the fast Fourier transform (FFT) routine used, and the PRF are specified by the user. Either the filter's unit-sample response or frequency response is also user specified. The true spectrum width and the number of trials of a simple Monte Carlo procedure involving repeated trials are input parameters when simulated data are used. The expected value of the estimate of a weather parameter, E(x), is estimated by [15]

$$E(x) = \frac{1}{N}\sum_{i=1}^{N}(x_i) \tag{2}$$

where x_i is the i^{th} estimate of the weather parameter, x, and N is the number of trials in the Monte Carlo procedure. The unbiased estimate of the variance of a weather parameter, VAR(x), is given by [15]

$$VAR(x) = \frac{\sum_{i=1}^{N} (x_i - \bar{x})^2}{N - 1} \tag{3}$$

where $\bar{x}$ is the expected value of the weather parameter given in (2).

Then the true value of the weather parameter, x_T, is used to estimate a bias error given by

$$E\{x - x_T\} = E\{x\} - x_T \tag{4}$$

where x_T corresponds to the theoretical value of the spectrum mean or spectrum width [2]. The theoretical value of the variance of the weather parameter, σ_x^2, given by (8) and (10) in Chapter 3, is used to estimate a variance error as

$$\text{error} = VAR(x) - \sigma_x^2 \tag{5}$$

where x is the spectrum mean or spectrum width estimate [2]. These bias and variance errors are then used as a basis for evaluating the clutter-rejection filter over a range of situations.

The Main Program for Simulated Data

Figure 3 represents the flowchart of the main program for analyzing clutter-rejection filters using simulated weather data. A clutter-rejection filter to be evaluated is input in terms of the filter's unit-sample response or frequency response as read from an external file. The weather spectrum is simulated in the frequency domain, and the weather and filter specification are passed to one of

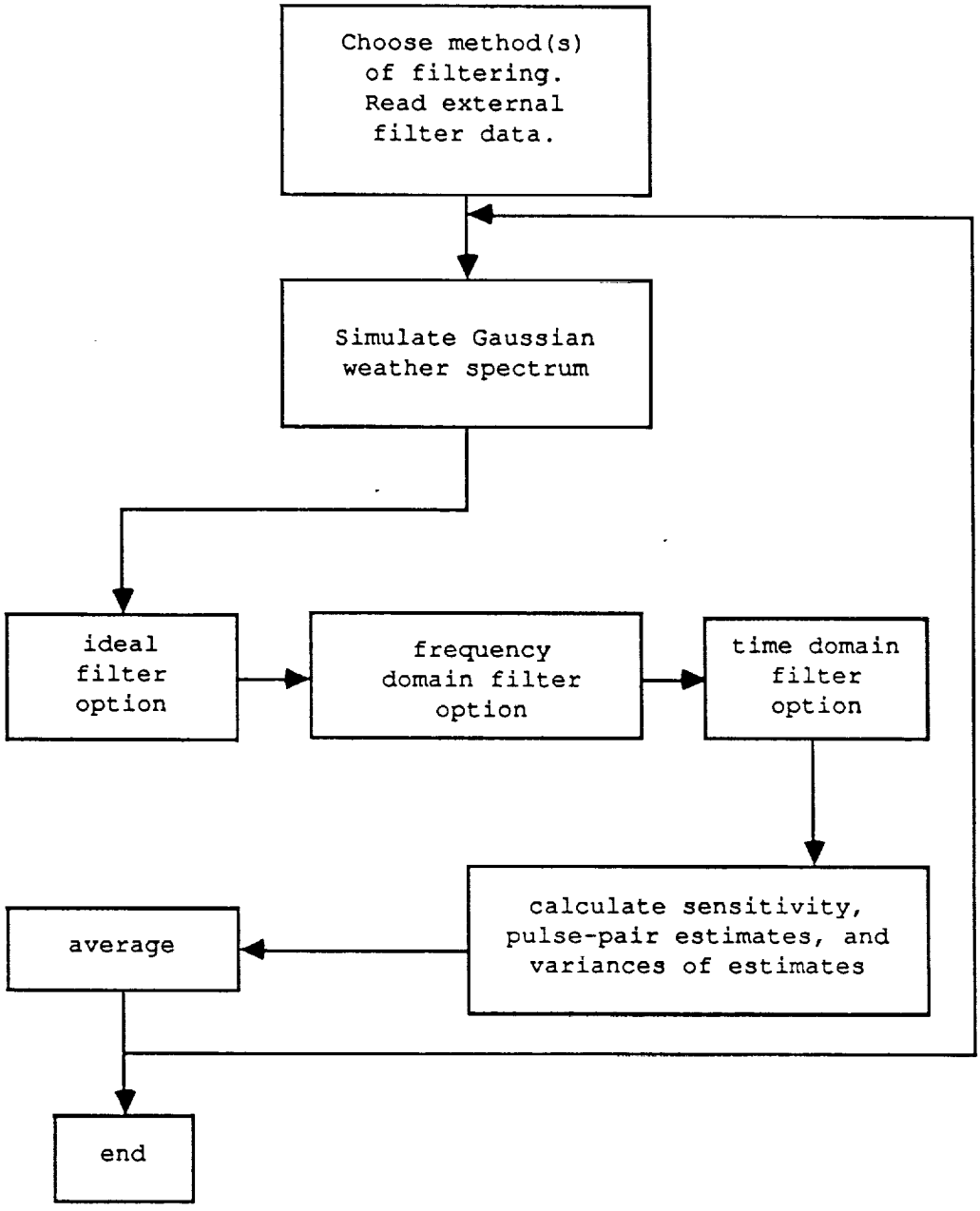


Figure 3. Flow Diagram for Main Program for Simulated Data

the filtering subroutines. As mentioned earlier, a time-domain filter may be preferred since many filters are conveniently represented by a difference equation. Frequency-domain filtering is included to offer the advantage of implementing an ideal filter, i.e. , one with a perfect passband/stopband characteristic. It also provides the capability to independently evaluate the effects of a filter's magnitude and phase response.

The filtered data is returned to the main program as a complex time series which represents a filtered Doppler radar IQ data sequence. The weather parameters are estimated from the time-domain data using the pulse-pair processing algorithm which will be described in Chapter 3. This procedure is repeated for the specified number of trials, and the statistics described by (2) and (3) are computed to represent averages over the ensemble of trials. Appendix A provides a software listing of the program.

Subroutines

The Time-domain Filter Subroutine

To implement a time-domain filter, the complex weather spectrum data is first transformed into the time domain. The filtering is accomplished by linearly convolving the filter's unit-sample response with the time-domain weather data or by passing the time-domain weather data through the filter's difference equation. The resulting complex data sequence is then passed to the main program. A software listing of this subroutine can be found in Appendix A.

The Frequency-domain Filter Subroutine

A general frequency-domain filter implementation is accomplished by the complex multiplication of the filter's frequency response, which is the Fourier transform of the filter's unit-sample response, with the Fourier transform of the complex weather time series [14]. Because a discrete Fourier transform that has a finite length, M, is used for the time/frequency transformations, the multiplication of the frequency-domain filter data with the weather spectrum does not equal the linear convolution of the filter's unit-sample response with the time-domain weather data. The frequency-domain multiplication is actually equivalent to a time-domain circular convolution. Using preliminary transformations with zero-padding, however, the circular convolution can be carried out to yield a linear convolution result [14].

Figure 4 represents the flowchart for the subroutine which filters data in the frequency domain. To implement a frequency-domain filter, the filter and weather data, which are both data vectors of length M, are transformed into the time-domain and are padded with M complex zeroes. The zero-padded weather and filter data are then transformed back into the frequency-domain using a fast Fourier transform (FFT) and are point-by-point multiplied. The result is then transformed back into the time-domain, and the first M complex data points of the time-domain sequence are passed to the main program. The final filtered sequence is the same sequence as if the filter's unit-sample response were linearly convolved with the time-domain weather data.

One special case is an ideal filter, and provision has been made to implement an ideal filter in terms of an amplitude response that is one in the passband and zero in the stopband. The phase response of

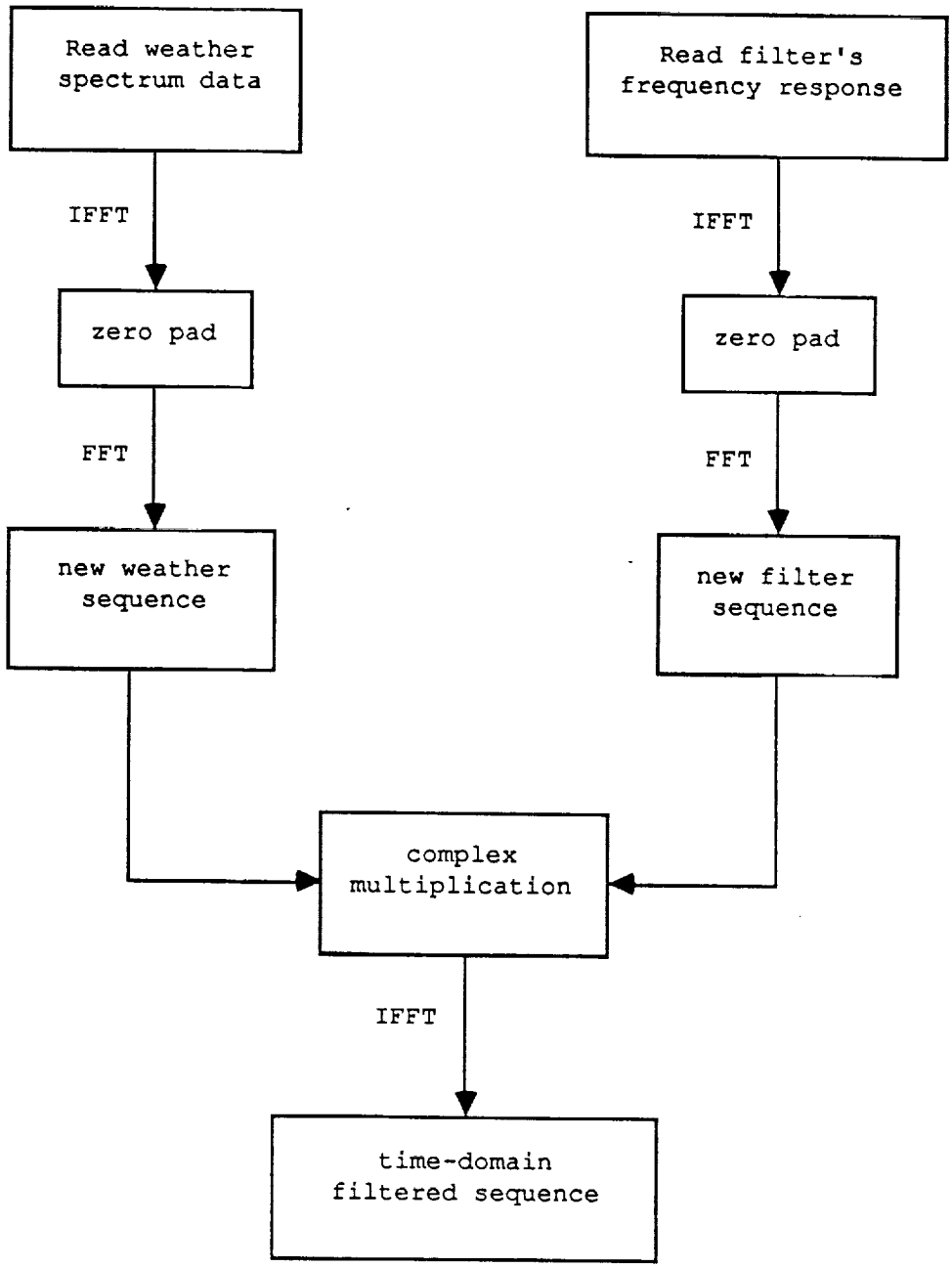


Figure 4. Flow Diagram for Subroutine for Frequency-domain Filtering

the filter is arbitrary. This ideal filter does not have a causal-realizable equivalent in the time domain but can be specified exactly in the frequency domain. Implementing the ideal filter as illustrated in Figure 2 is accomplished by the complex multiplication of the frequency-domain filter data with the weather spectrum. The resulting data are then transformed into the time-domain using an inverse fast Fourier transform (IFFT) and passed to the main program. The software listing to the frequency-domain subroutine can be found in Appendix A.

The Main Program for Externally Generated Data

Filtering of externally provided radar IQ data can be processed as illustrated in the flow diagram of Figure 5. The filter's unit-sample or frequency response specification is read from an external file according to which domain would be used to filter the data. The complex time-domain weather sequence is read from an external file and the filter and weather data are passed to the appropriate subroutine which filters the data. The filtered data are returned to the main program as a complex time-domain sequence, and the estimates of the weather parameters are then calculated from the time-domain data. Appendix A includes the software listing for this program.

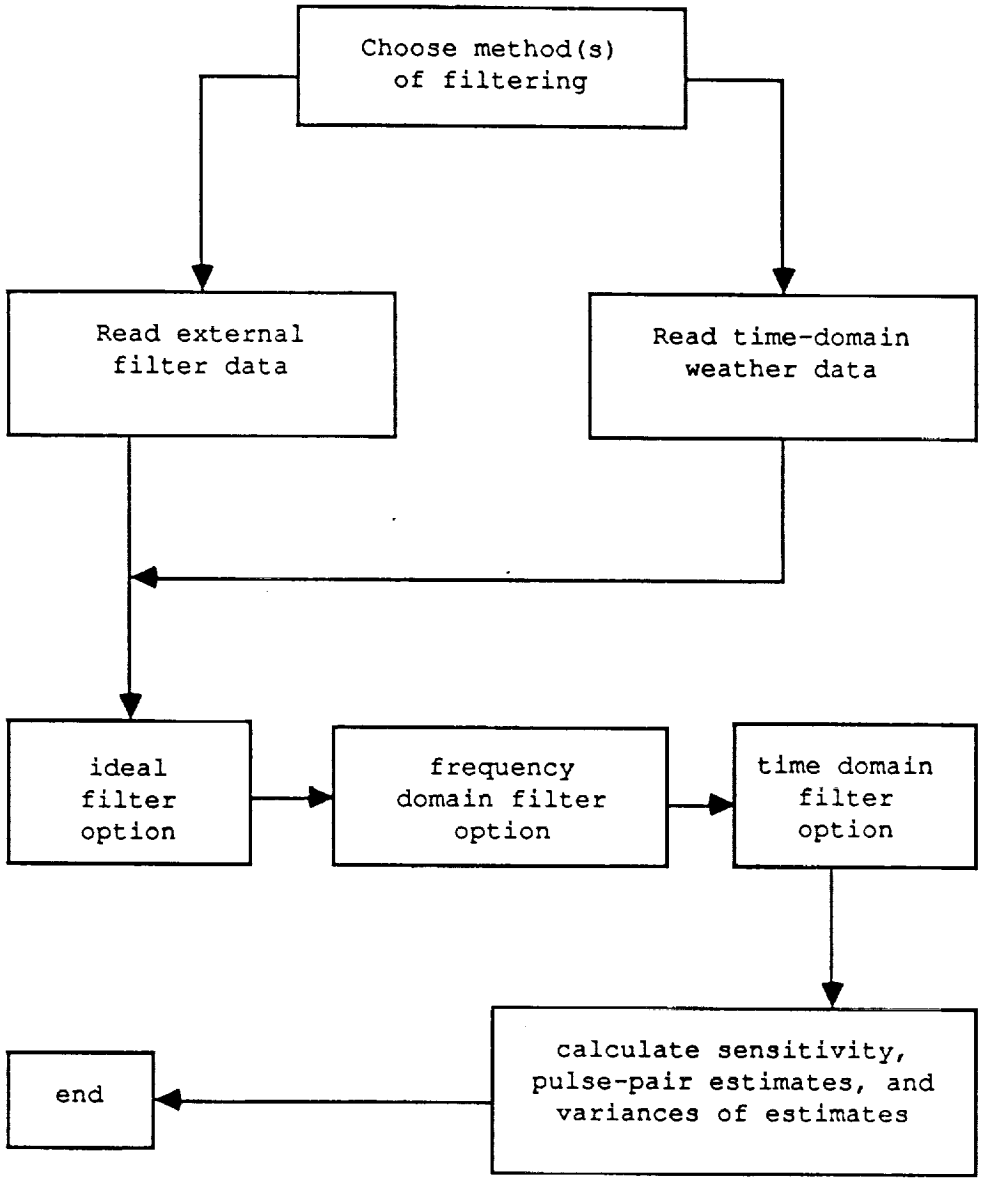


Figure 5. Flow Diagram for Main Program for External Data

CHAPTER III

SOFTWARE VALIDATION

Introduction

This chapter describes the procedure that has been used to validate the analysis software discussed in Chapter II. Results are presented to demonstrate that previously published theoretical pulse-pair estimate quality [4,5,6] can be achieved without filtering using the Monte Carlo procedure. The number of trials used in the Monte Carlo procedure was determined iteratively. The theoretical variances of the spectrum mean and width estimates were first analyzed without filtering for a particular weather spectrum. The weather spectrum was then simulated and filtered. A variance error, the post-filter variance minus the theoretical variance, was computed. The number of trials was increased, and various weather spectra and filters were considered. The variance error was observed as a function of the number of trials to determine the number of trials required for the variance error to converge. This number was determined to be 200, and is used in computing the results presented in this thesis, unless stated otherwise.

The pulse-pair mean and width estimates are compared to their true values, as specified in the simulation of the Gaussian weather spectrum. The computed variances of the estimates are compared to their theoretical variances. Additionally, earlier results for clutter-rejection filter analysis have been reproduced as a further means of validation [4].

Width Estimate

The standard deviation of the Gaussian weather spectrum is referred to as the spectrum width. The pulse-pair width estimate, $\hat{w}$, is defined [2] as

$$\hat{w} = \frac{2}{(2\pi T_s)^2} \left[1 - \frac{|R(T_s)|}{R(0)} \right] \tag{6}$$

where T_S is the interpulse spacing of the transmitted pairs of pulses from the radar. $R(T_S)$ is estimated by

$$\hat{R}(T_s) = \frac{1}{M} \sum_{m=1}^{M-1} V^*(m) V(m+1) \tag{7}$$

where * denotes the complex conjugate, M denoted the number of pulses or samples, and V(m) represent the m^{th} sample of the complex time series. The theoretical variance of the width estimate [2] for narrow widths, large SNR, and contiguous pairs is approximated by

$$VAR(\hat{w}) = \frac{3w}{23\sqrt{\pi MT_s}} \tag{8}$$

where M is the number of pulses, and w is the true width. Using a simulated truncated Gaussian spectrum magnitude with an arbitrary phase, the width bias error, the expected value of the width estimate

as defined by (2) minus the true width as specified in the simulation, is plotted versus the normalized (Nyquist bandwidth scaled from -1 to 1) true width in Figure 6 for three cases: 128 samples, 256 samples, and 512 samples. The number of samples here refers to the number of pulses that would be processed for a pulse Doppler radar return, and does not relate to the number of Monte Carlo trials, N, used in Equations (2) and (3). The normalized true mean is 0.0 for each case. The pulse pair method appears to very slightly underestimate the normalized true width as the normalized true width increases. This occurs because the simulated truncated Gaussian spectrum data that are outside the Nyquist bandwidth are discarded. Since approximately 99% of the weather data are within ±3 standard deviations of the true mean, the lost data are negligible for small true widths. For larger true widths, the spectrum is more distributed across frequencies. This results in more data being discarded, which causes a bias in the estimate. These results are virtually unchanged by increasing the number of Monte Carlo trials beyond 200.

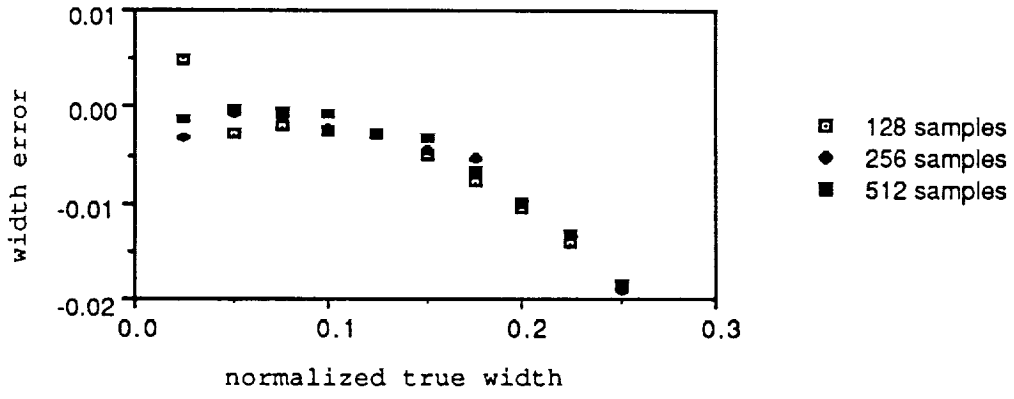


Figure 6. Width Error vs. True Width

Mean Estimate

The pulse-pair mean estimate, $\hat{f}$, is defined [2] as

$$\hat{f} = \frac{ARG[R(T_s)]}{2\pi T_s} \tag{9}$$

where ARG[·] is the arctangent of the ratio of the imaginary part of $R(T_s)$ to the real part of $R(T_s)$. The theoretical variance of the mean estimate [2] for narrow spectrum widths, large signal-to-noise ratios (SNR), and contiguous pairs is approximated by

$$VAR(\hat{f}) = \frac{w}{4\sqrt{\pi} M T_s} \tag{10}$$

The mean bias error, the expected value of the mean estimate minus the true mean, was plotted versus the normalized true mean in Figure 7 for the three cases: 128, 256, and 512 samples corresponding to the number of pulses. For each case, three normalized widths were considered: 0.05, 0.15, and 0.25. The mean bias error is approximately zero for all cases when the normalized true mean is between -0.5 and 0.5. The true mean here is the mean specified to the simulation, and is actually the mode, since the Gaussian magnitude that is simulated is truncated. The mean and the mode are virtually the same for the values considered here. The pulse pair method appears to underestimate the true mean for normalized true means greater than 0.5 and less than -0.5. This also occurs because the simulated truncated Gaussian spectrum data that is outside the Nyquist interval is discarded. The lost data are

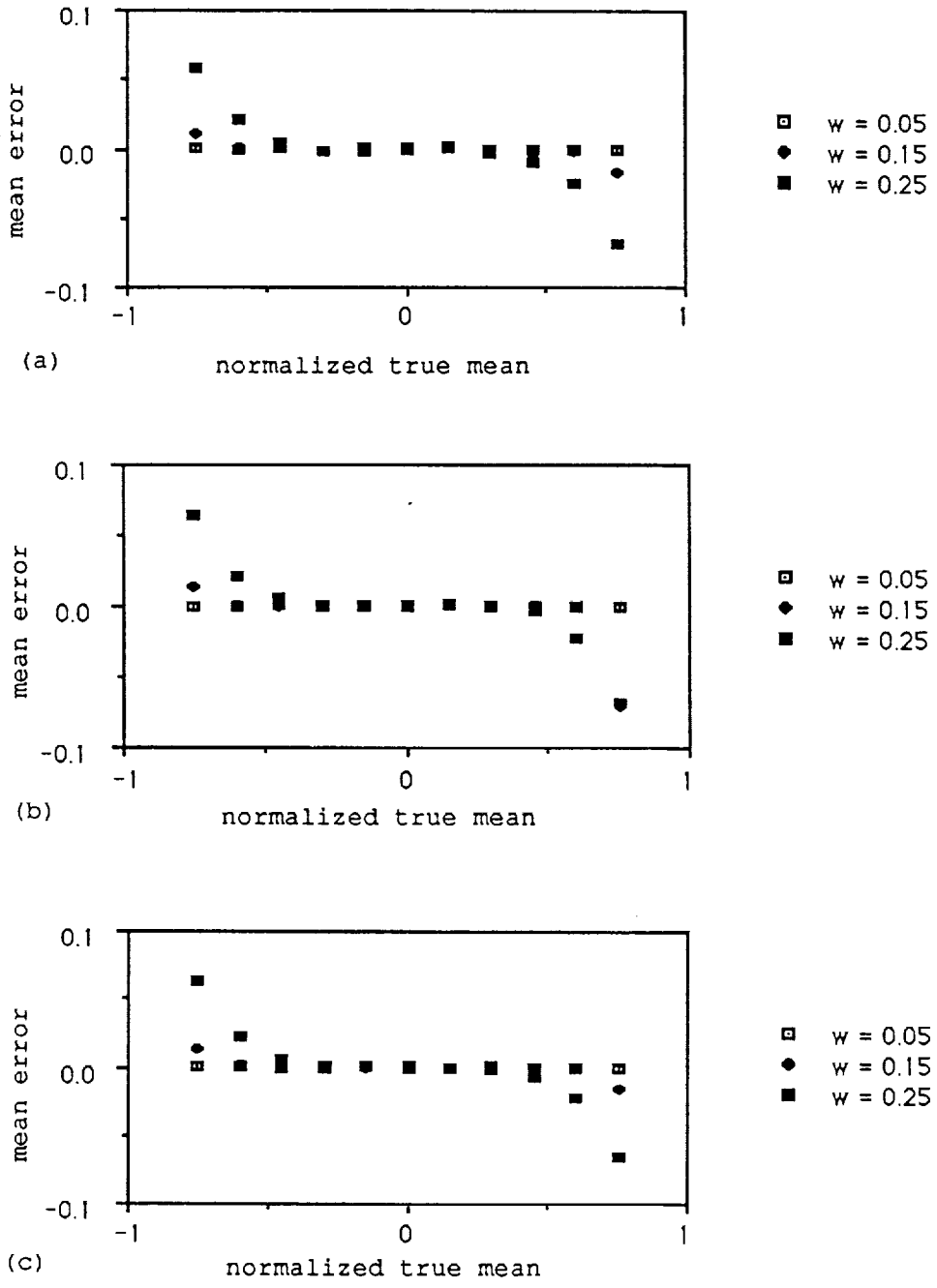


Figure 7. Mean Error vs. True Mean
(a) 128 samples, (b) 256 samples,
(c) 512 samples .

negligible for relatively small true widths when the true mean is not at either extreme in the Nyquist interval. Therefore, for normalized true means greater than 0.5 and less than -0.5, the true widths considered were large enough that the lost data are a factor, and the true mean is underestimated. As expected, the larger true widths resulted in a larger mean error for true means that were near either end of the Nyquist interval. Again, these results are virtually the same when the number of Monte Carlo trials is increased beyond 200.

Variances of Mean and Width Estimates

Since the theoretical variances of both the mean and width estimates are functions of the true width, the theoretical and estimated variances of both estimates are plotted versus the true width for 128, 256, and 512 samples. The estimated variances are computed using (3) of Chapter II. The theoretical variance of each estimate increases linearly as the true width increases and decreases linearly as the number of samples increase.

Figure 8 shows that the estimated variance of the width estimate is approximately equal to the theoretical variance of the width estimate for the true widths considered. The linear relationship that exists between the theoretical variance and the true width does not occur with the estimated variance here because of the limited number of samples considered.

The estimated and theoretical variance of the mean estimate are approximately equal for the true widths considered, as illustrated in Figure 9. For each case, the estimated variance of the mean estimate generally increases as the true width increases and generally decreases

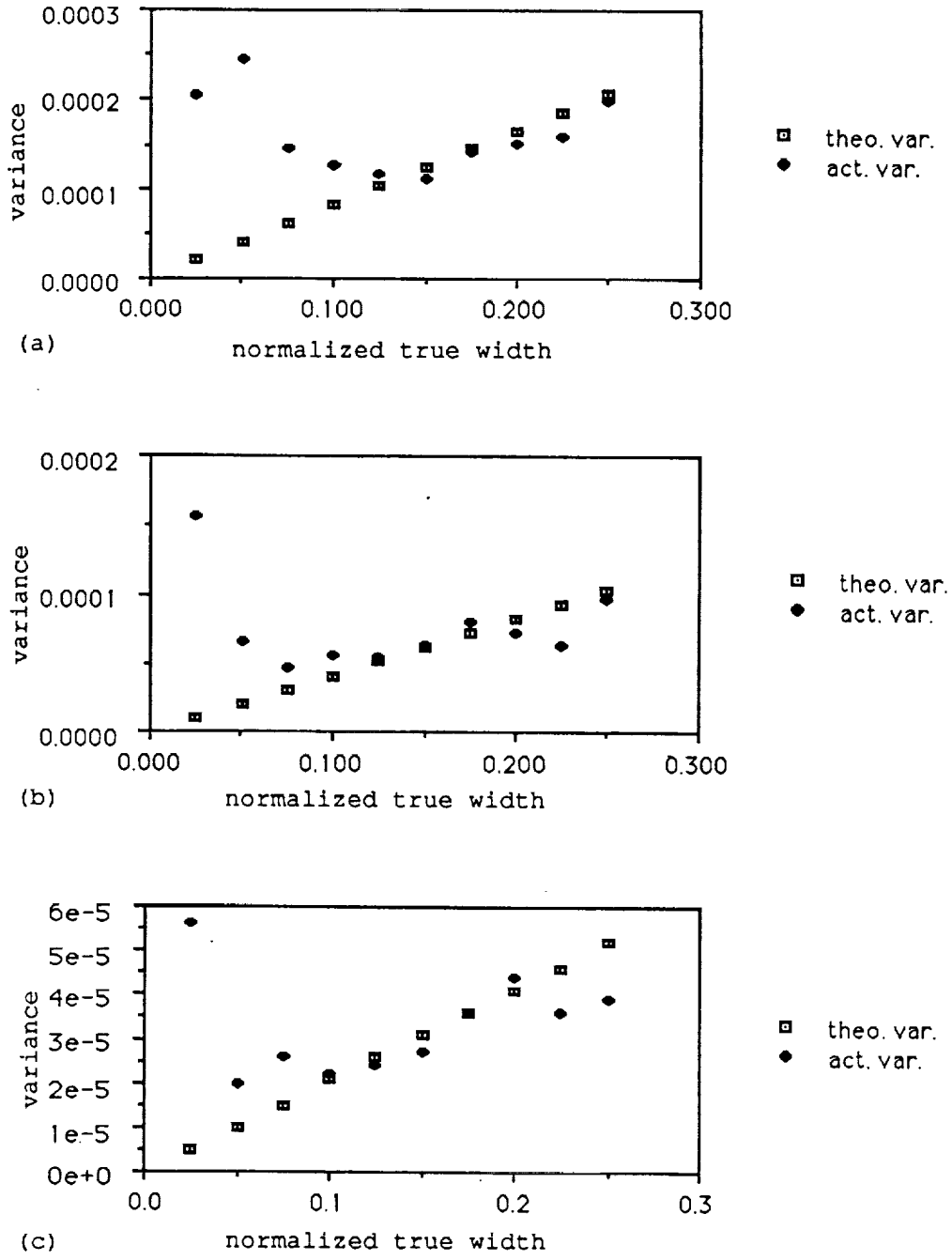


Figure 8. Theoretical and Actual Variance of Width Estimate vs. True Width; (a) 128 samples, (b) 256 samples, (c) 512 samples

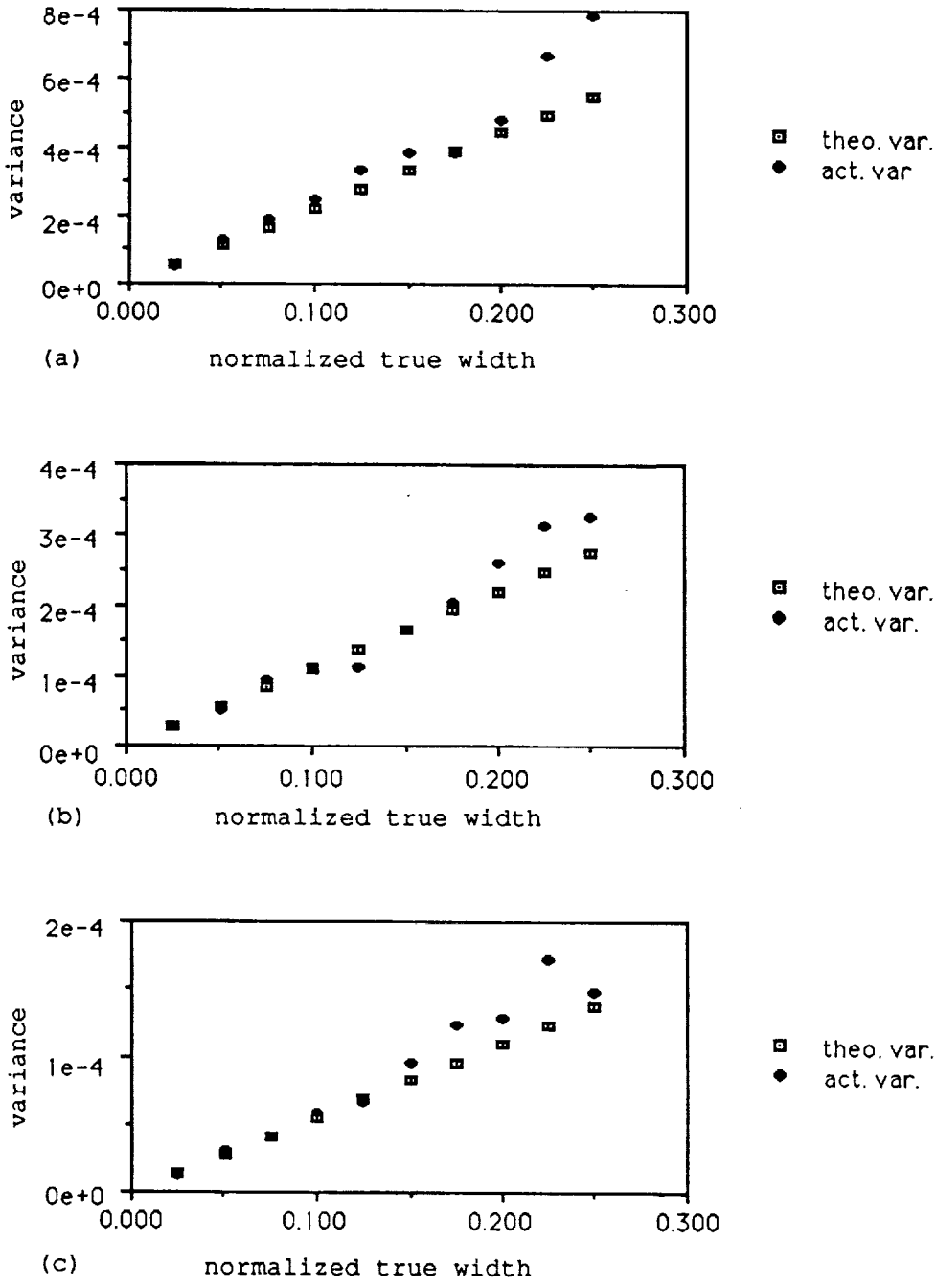


Figure 9. Theoretical and Actual Variance of Mean Estimate vs.True Width; (a) 128 samples, (b) 256 samples, (c) 512 samples

as the number of samples increases. These results were again virtually unchanged with the number of Monte Carlo trials increased beyond 200.

Reproduction of Previous Results

As a means of validating the post-filter analysis software, previously published results [4] based on a Chebyshev FIR filter designed using the Parks-McClellan procedure [16] are reproduced. Using a frequency-domain implementation of the NEXRAD system FIR filter #1 [4], 512 data samples and 500 trials produce the results in Figure 10. These plots represent the loss of sensitivity (reflectivity error), mean bias error, and width bias error versus true mean (velocity) after the simulated Gaussian data are passed through a finite impulse response clutter rejection filter. These results agree with the previously published results.

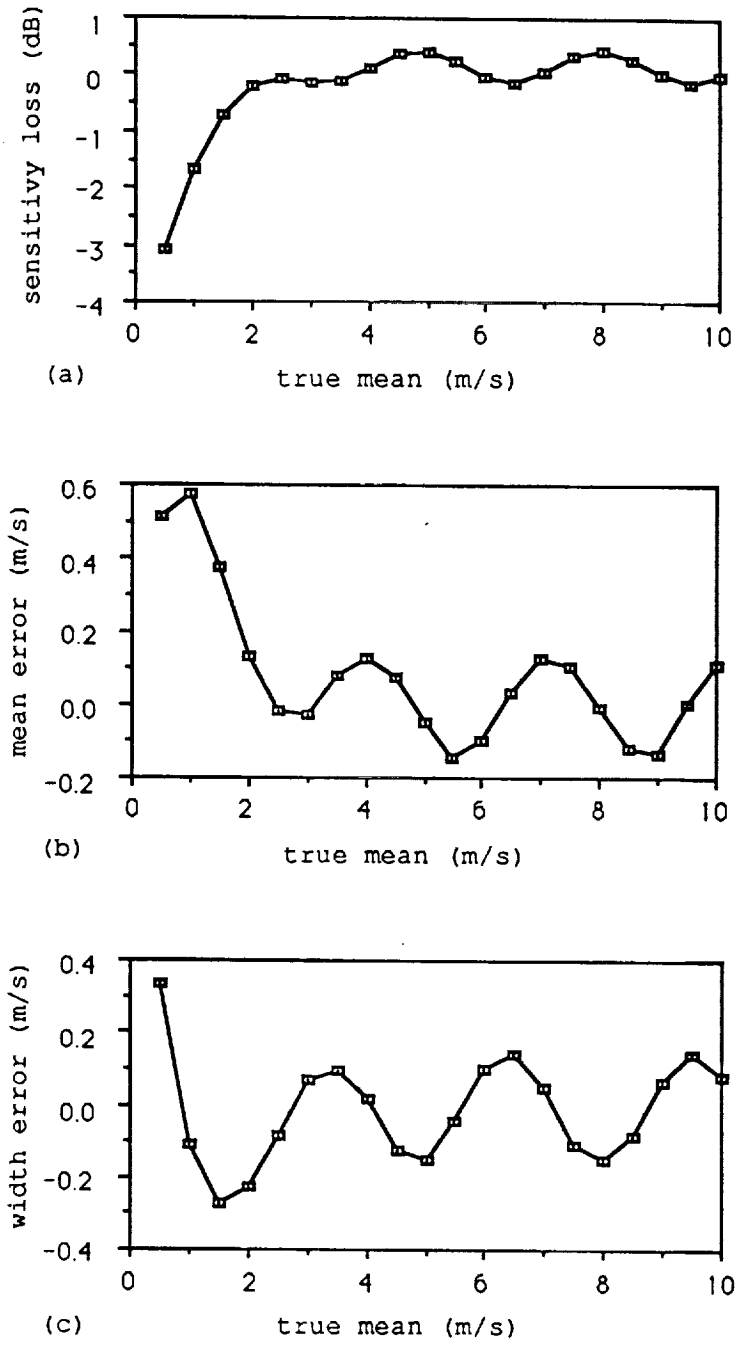


Figure 10. Reproduction of Prevoiusly Published Results

CHAPTER IV

RESULTS

Introduction

Five clutter-rejection filters are evaluated using the analysis software described in Chapters III and IV. Simulated weather data with no added clutter are generated to model a truncated Gaussian magnitude spectrum with a uniformly distributed random phase. Time-domain data consisting of 128 complex samples representing the digitized IQ output of a 9.3 GHz pulse Doppler radar with a pulse repetition frequency (PRF) of 3723 pulses/second, are used in the computation of the pulse-pair estimates as defined by (6) and (9) of Chapter 3. The number of repeated trials used in the Monte Carlo procedure is 200. The Nyquist windspeed Doppler bandwidth corresponding to the selected PRF and radar wavelength is ±30 m/s and the specified simulated weather spectrum width is 2.5 m/s for every case. This width is larger than the narrow widths used in previously published results [4] where the radar platform is stationary. Since the radar platform is moving in the airborne case, the clutter spectrum is more distributed than that for a stationary platform.

This chapter analyzes five clutter-rejection filters in terms of the loss of sensitivity and post-filter spectrum estimates. The loss of sensitivity and the post-filter weather spectrum estimates are plotted versus the specified weather spectrum mean for each filter, as the specified mean (mode) of the truncated Gaussian spectrum is increased from 0 m/s to 15 m/s. Bars representing ±3 standard

deviations of the post-filter estimates are included on the plots of the spectrum estimates.

An ideal notch filter is considered as a baseline to which other filters can be compared. Several non-ideal notch filters are also evaluated including an unnormalized first-order Butterworth filter, a three-stage cascaded first-order Butterworth filter, a simple pulse canceller, and finally a 39-tap finite impulse response (FIR) filter. For each of the non-ideal filters considered in the analysis, plots are provided to show the filter unit-sample response and the magnitude and phase response. The response of the filter to a sinusoid that has a digital frequency corresponding to a windspeed of 15 m/s is also presented. The response of the filter to a sinusoid with a digital frequency well into the filter's passband is used to indicate the length of the filter's transient response.

In addition to the analysis of clutter-rejection filters with the simulated Gaussian weather spectrum, analysis results are provided from external data obtained from a radar simulation model [1]. This simulation uses a finite element approach to create a radar return typical of an aircraft radar illuminating a microburst located along the final approach glide path, and includes a clutter model.

Analysis results are also included to evaluate the effect of the phase response of a notch filter on the post-filter estimates Whether the clutter-rejection filter must have a linear phase response is of particular interest here. The magnitude and phase response of the first-order Butterworth filter is initially considered. The phase response of this Butterworth filter is approximately linear in the filter's passband. The magnitude response of the first-order

Butterworth filter with a zero phase response is considered. This, of course, represents a special case of a linear phase over all frequencies. Finally, the magnitude response of the first-order Butterworth filter with a random phase response is considered. A randomly specified phase response is intended to represent a "worst-case" arbitrary phase response.

Ideal Filter (notch bandwidth = 3.00 m/s)

Figure 11 contains the results of the analysis software when the simulated weather data are filtered using an ideal filter that has a notch bandwidth of ±3.00 m/s. The gain of this filter is unity for frequencies greater than ± 3.00 m/s, and zero for frequencies less than ±3.00 m/s. The results agree with the predicted results described in the problem statement in Chapter I. The loss of sensitivity is greatest, 17.02 dB, when the true weather mean is 0.0 m/s. As the true weather mean is increased, the loss of sensitivity approaches 0.0 dB.

When the true weather mean is 0.0 m/s, the post-filter mean bias is 2.90 m/s, and the variance of the estimate is very large when compared to the variance at other specified means. The post-filter mean bias generally decreases as the specified mean is increased out of the filter's notch. The maximum post-filter mean bias, 4.87 m/s, occurs when the specified mean is 1.5 m/s.

The true weather width is 2.50 m/s for all values of the true mean. The post-filter width bias is 3.08 m/s when the true mean is 0.0 m/s, and the variance of the estimate is large. As the true mean is increased to 3.0 m/s, the post-filter width bias becomes -1.22 m/s. As

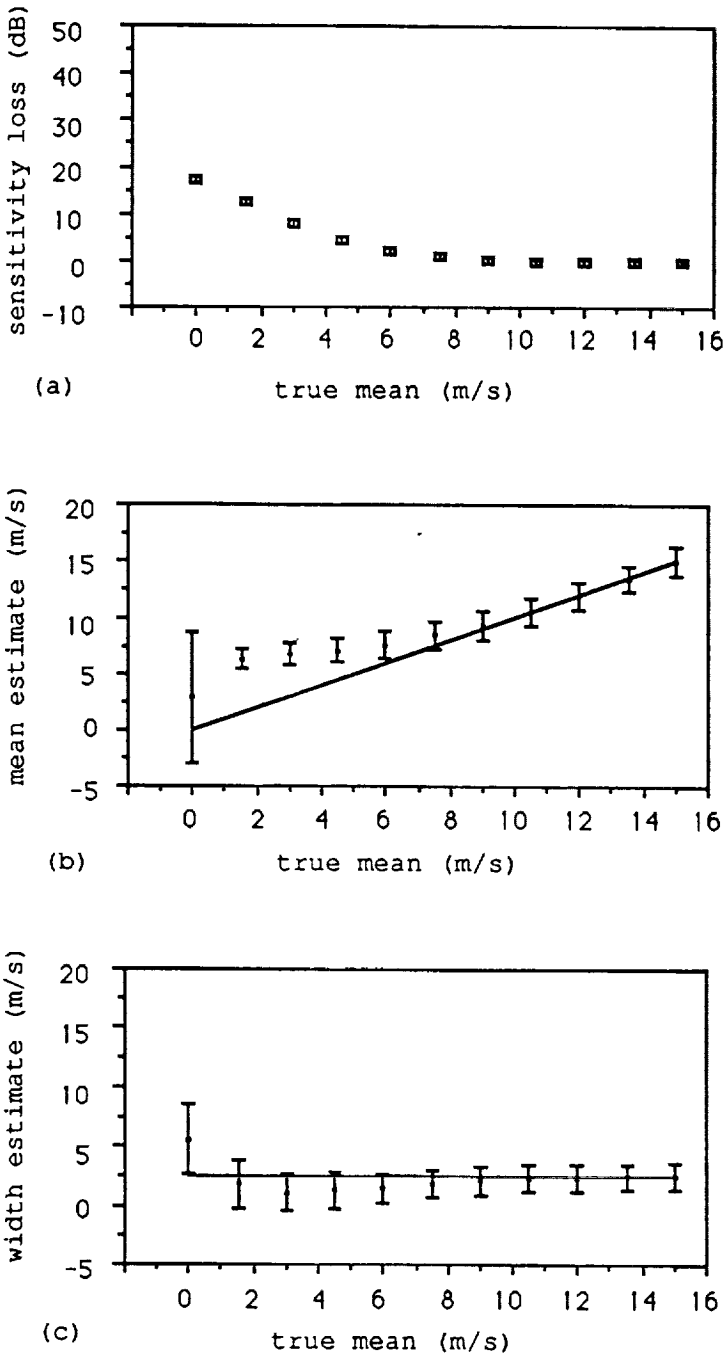


Figure 11. Analysis Software Results for an Ideal Filter (notch bandwidth = ±3.00 m/s)

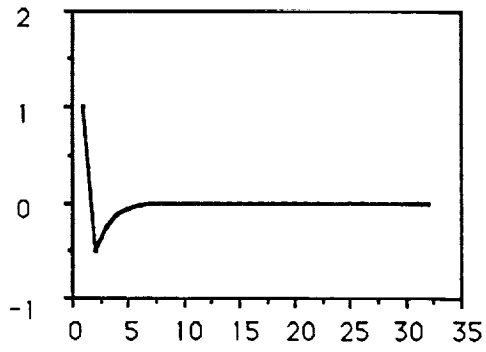
the specified mean is increased further, the post-filter mean estimate becomes unbiased.

Unnormalized 1st-order Butterworth Filter

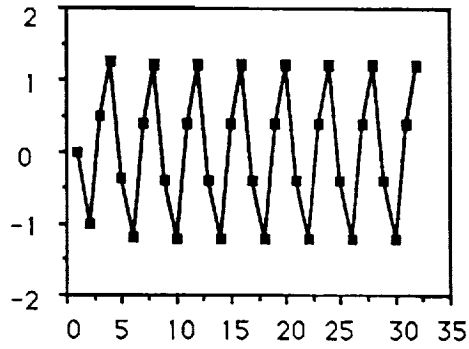
An unnormalized first-order Butterworth filter that is represented by

$$y(n) = 0.5y(n - 1) + x(n) - x(n - 1)$$

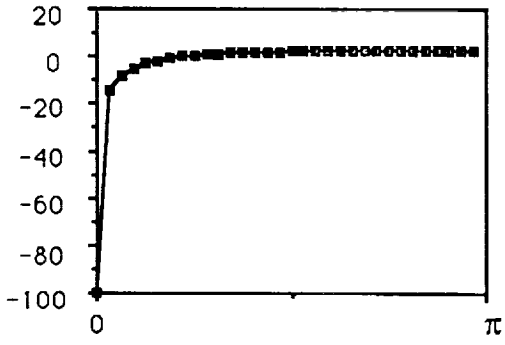
has been considered in previous research. The 3dB bandwidth, ±6.15 m/s, of this filter with respect to its maximum gain, 2.5 dB, appears adequate for rejecting the clutter from the simulated data of Appendix B. This filter is referred to as "unnormalized" because it does not have a gain of unity at 30 m/s. The gain is actually 2.5 dB. The unnormalized filter is of interest because it is potentially much simpler to implement than the normalized version with its coefficients: 0.5, 1.0, and -1.0. The only multiplication required to implement the difference equation is actually a power of two, which can be accomplished with a bit shift. Analysis of the system function in the z-domain by taking the z-transform of the above difference equation will show that this filter has a zero at $z = 1$, which corresponds to 0 m/s, and a pole at $z = 0.5$, which corresponds to 30 m/s. The filter's unit-sample response and response to a 15 m/s sinusoid is presented in Figure 12. Figure 12b shows that the transient response of the filter is negligible. The filter's magnitude and phase response are also plotted in Figure 12. The magnitude response of this filter is negative infinity at 0.0 m/s on the dB scale because there is a zero at $z = 1$. The magnitude response of the filter is 2.5 dB at 30 m/s



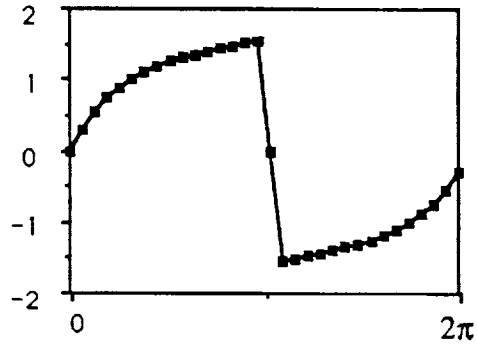
(a) Unit-sample Response



(b) Response to a Sinusoid of Period $\pi/2$



(c) Magnitude Response in dB



(d) Phase Response in Radians

Figure 12. Filter Characteristics of Unnormalized First-order Butterworth

representing a gain in the filter's passband. The phase response of this filter is non-linear, but may be considered approximately linear in the filter's passband.

The results with the unnormalized first-order Butterworth filter used to filter the simulated Gaussian weather data are presented in Figure 13. The loss of sensitivity is greatest, 6.77 dB, when the specified weather mean is 0.0 m/s as expected. The loss of sensitivity approaches -1.98 dB as the specified mean is increased. A loss of sensitivity that is less than 0.0 dB is a result of the power of the post-filter weather data actually being greater than that of the unfiltered weather data. This occurs because the filter has a positive gain in its passband. The loss of sensitivity for small specified means is not nearly as great as it is for the ideal filter whose notch bandwidth is ±3.00 m/s. This result occurs because the ideal filter eliminates all the power in the filter's notch whereas a non-ideal filter only attenuates the power in the filter's notch. Of course, the ideal filter will also eliminate the clutter power in the notch better than the Butterworth filter.

The post-filter mean estimate is approximately equal to the true weather spectrum mean for the Butterworth filter when the true mean is 0.0 m/s. As the true weather mean is increased, a slight bias occurs which converges to zero as the true mean increases out of the filter's notch. The maximum bias in the mean estimate, 1.89 m/s, occurs when the true mean is 3.0 m/s. Compared to the ideal filter, the bias error in the post-filter mean estimates, particularly for small true weather means, is not nearly as great for the Butterworth filter. Also the large variance of the post-filter mean estimate associated with the

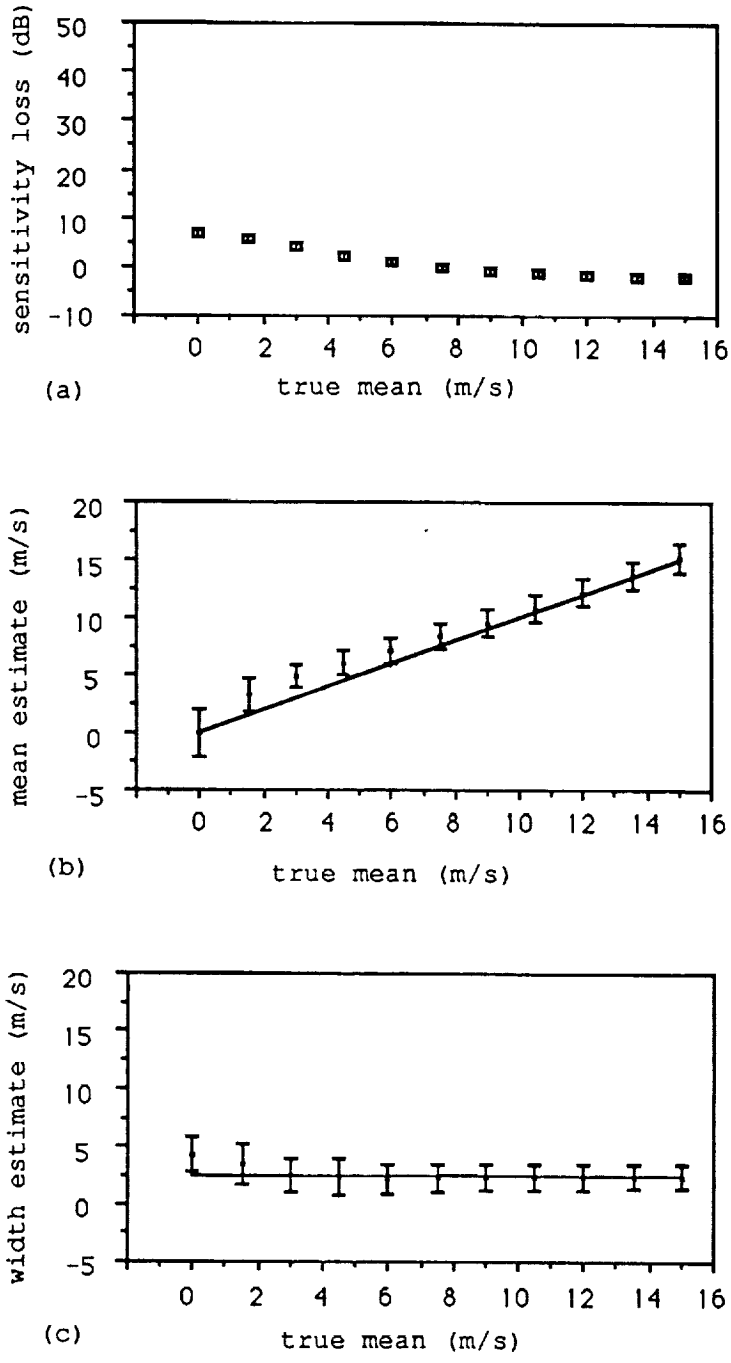


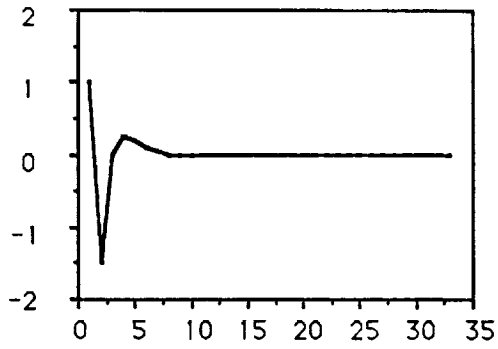
Figure 13. Analysis Software Results for Unnormalized First-order Butterworth

ideal filter at the true weather mean of 0.0 m/s is not present with the Butterworth filter.

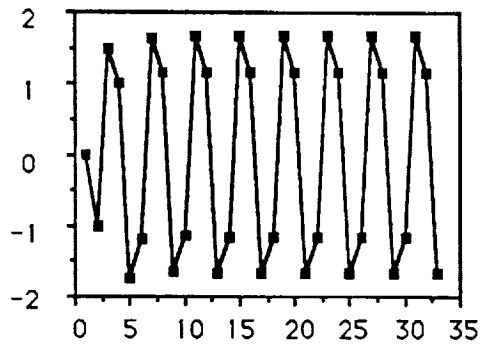
The post-filter width bias is 1.8 m/s when the true mean is 0.0 m/s. As the true mean is increased to 6.0 m/s, the post-filter width bias decreases to -0.29 m/s. The post-filter width estimate approaches the true width as the true width is increased further. The error in the post-filter width estimates considering the Butterworth filter is not nearly as great as that for the comparable ideal filter. As with the post filter mean estimate, the large variance of the post-filter width estimate for the ideal filter at the true weather mean of 0.0 m/s is not present with the Butterworth filter. The non-ideal filter appears to offer less error than the ideal filter when considering weather parameter estimation. This is reasonable since the weather data in the notch are not completely eliminated. This, however, must be weighed against the fact that the ideal filter will do a much better job at clutter rejection.

Three-stage Cascaded Normalized 1st-order Butterworth Filter

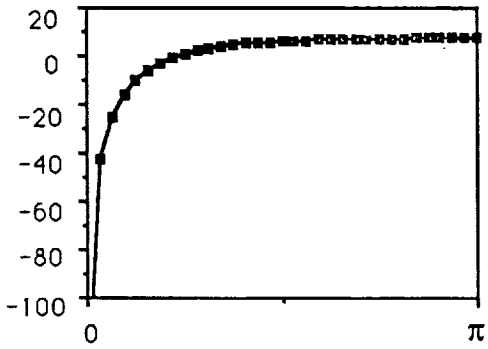
Three stages of the previous Butterworth filter are cascaded as a simple way to obtain a higher-order filter [14]. The resulting third-order filter is easily realizable since it is simply three stages of the previous unnormalized first-order Butterworth filter. The system function in the z-domain has three zeros at $z = 1$, three poles at $z = 0.5$, and a 3-dB notch bandwidth of ±11 m/s with respect to the maximum gain, 5.98 dB. Figure 14 represents the unit-sample response of the three-stage Butterworth filter and the response of the filter to a sinusoid that has a discrete frequency of 15 m/s. Figure 14b shows



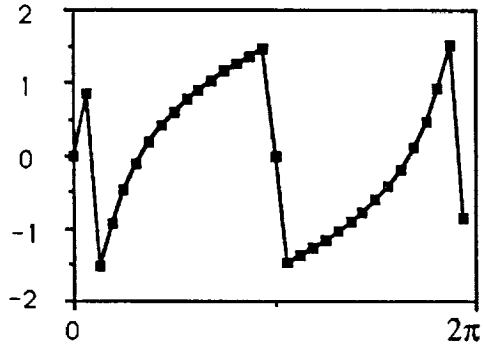
(a) Unit-sample Response



(b) Response to a Sinusoid of Period $\pi/2$



(c) Magnitude Response in dB



(d) Phase Response in Radians

Figure 14. Filter Characteristics of Three-stage Cascaded Unnormalized First-order Butterworth

that the transient response of this filter is negligible. The filter's magnitude and phase responses are also plotted in Figure 14. The magnitude response is negative infinity at 0.0 m/s, -3 dB at 5.54 m/s, and 7.50 dB at 30 m/s. The phase response is non-linear.

The analysis output is represented in Figure 15. The loss of sensitivity is 11.70 dB when the true mean is 0.0 m/s and decreases to -5.98 dB when the true mean is 15.0 m/s. The post-filter mean estimate is approximately equal to the true mean at 0.0 m/s. The variance of the estimate for a true mean of 0.0 m/s is large when compared to the variance at other values of the true mean. The error of the post-filter mean estimate is its maximum, 3.88 m/s, when the true mean is 1.5 m/s, and the mean estimate bias approaches zero as the specified mean is increased.

The post-filter width estimate bias is 4.79 m/s when the true mean is 0.0 m/s. The width estimate bias becomes -0.27 m/s as the true mean is increased to 7.5 m/s. The variance of the estimate is large for small values of the true mean, and generally decreases as the true mean increases out of the filter's notch.

Pulse Canceller

The simple pulse canceller which can be represented by

$$y(n) = x(n) - x(n - 1)$$

is a commonly used clutter-rejection filter [11]. This filter's system function in the z-domain has a zero at $z = 1$, and a pole at $z = 0$. The pulse cancellor's unit-sample response and transient response are illustrated in Figure 16. Figure 16b shows that the transient response

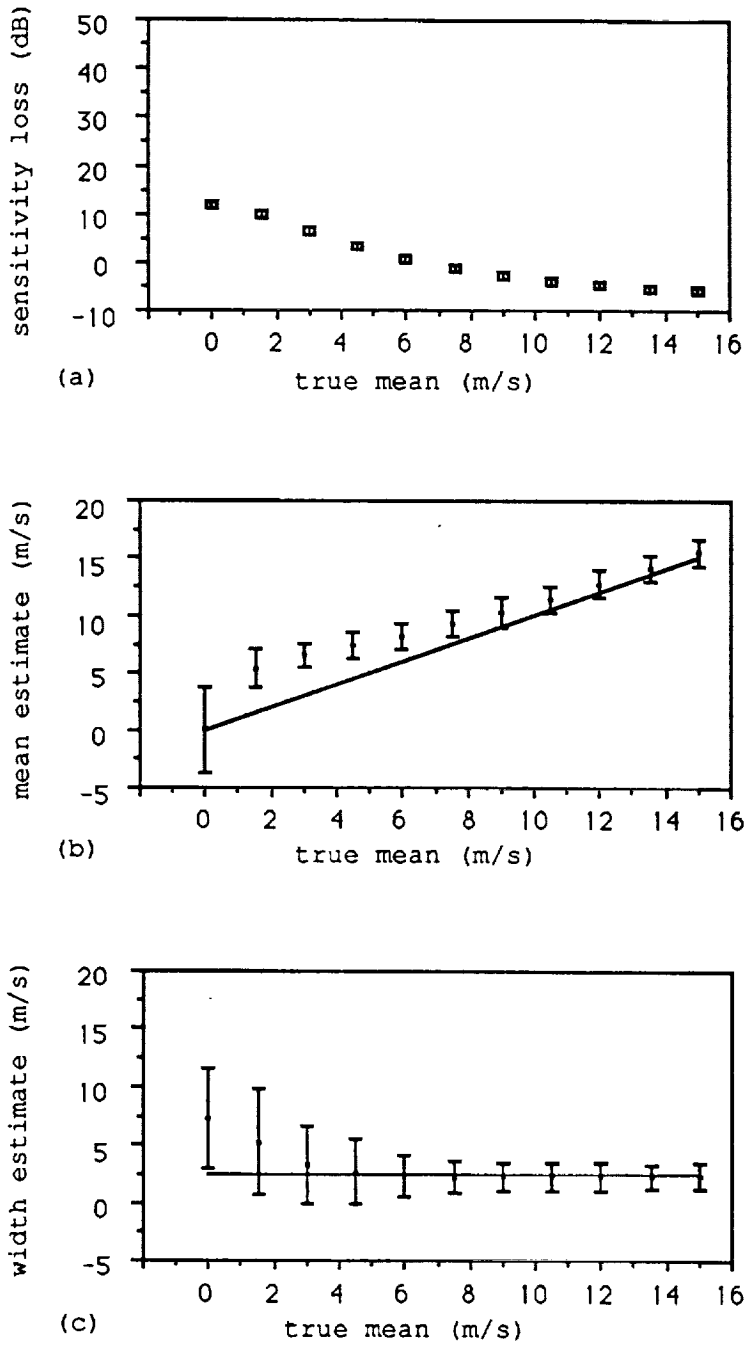
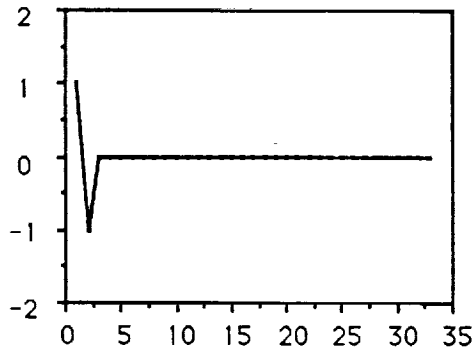
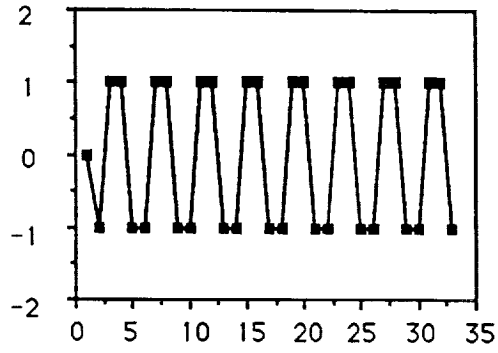


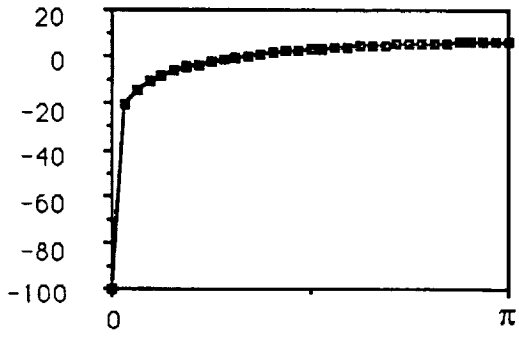
Figure 15. Analysis Software Results for Three-stage Cascaded Unnormalized First-order Butterworth



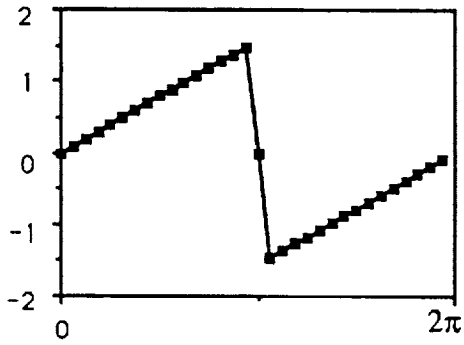
(a) Unit-sample Response



(b) Response to a Sinusoid of Period $\pi/2$



(c) Magnitude Response in dB



(d) Phase Response in Radians

Figure 16. Filter Characteristics for Pulse Canceller

of the filter is negligible. Figure 16 also shows the filter's magnitude and phase response. The magnitude response is negative infinity at 0.0 m/s because of the zero at z = 1. The gain is 3 dB below its maximum value, 6.02 dB, at 15 m/s. The phase response of the pulse canceller is linear.

Figure 17 provides analysis of the pulse canceller's effects on the weather parameters. The loss of sensitivity is 11.22 dB when the true mean is 0.0 m/s, and decreases to -3.00 dB when the true mean is 15 m/s. The post-filter mean estimate bias is essentially zero when the true mean is 0.0 m/s, and the variance of the estimate is slightly larger than that at other values of the true mean, as expected. The bias of the mean estimate reaches its maximum value, 2.51 m/s, when the true mean is increased to 3.0 m/s, and approaches zero as the true mean is increased further.

The post-filter width estimate bias is 3.19 m/s when the true mean is 0.0 m/s. As the specified mean is increased to 13.5 m/s the post-filter width estimate bias becomes -0.16m/s. The width estimate bias approaches zero as the true mean is increased to 15.0 m/s. The variance of the estimates is much larger at small values of the true mean than at larger values, as expected.

39-tap FIR Filter

The 39-tap FIR filter whose specifications are given in Table 1 has been considered in previous research [9]. The stopband/passband characteristics of this filter, whose notch is much narrower than that of the other filters presented here, is based on models for distributed and discrete clutter characteristic of ground-based stationary platform pulse-Doppler radar [9]. This filter, unlike the other nonideal

TABLE 1

FINITE IMPULSE RESPONSE FILTER SPECIFICATIONS

IMPULSE RESPONSE

H(1) = -0.145E+00 = H(39)
H(2) = -0.052E-01 = H(38)
H(3) = -0.159E-01 = H(37)
H(4) = -0.166E-01 = H(36)
H(5) = -0.173E-01 = H(35)
H(6) = -0.179E-01 = H(34)
H(7) = -0.187E-01 = H(33)
H(8) = -0.191E-01 = H(32)
H(9) = -0.197E-01 = H(31)
H(10) = -0.204E-01 = H(30)
H(11) = -0.208E-01 = H(29)
H(12) = -0.212E-01 = H(28)
H(13) = -0.215E-01 = H(27)
H(14) = -0.219E-01 = H(26)
H(15) = -0.221E-01 = H(25)
H(16) = -0.225E-01 = H(24)
H(17) = -0.226E-01 = H(23)
H(18) = -0.228E-01 = H(22)
H(19) = -0.229E-01 = H(21)
H(20) = +0.977E+00 = H(20)

	BAND1	BAND2
LOWER BAND EDGE	0.0000	0.0167
UPPER BAND EDGE	0.0050	0.5000
DESIRED VALUE	0.0000	1.0000
WEIGHTING	5.0000	1.0000

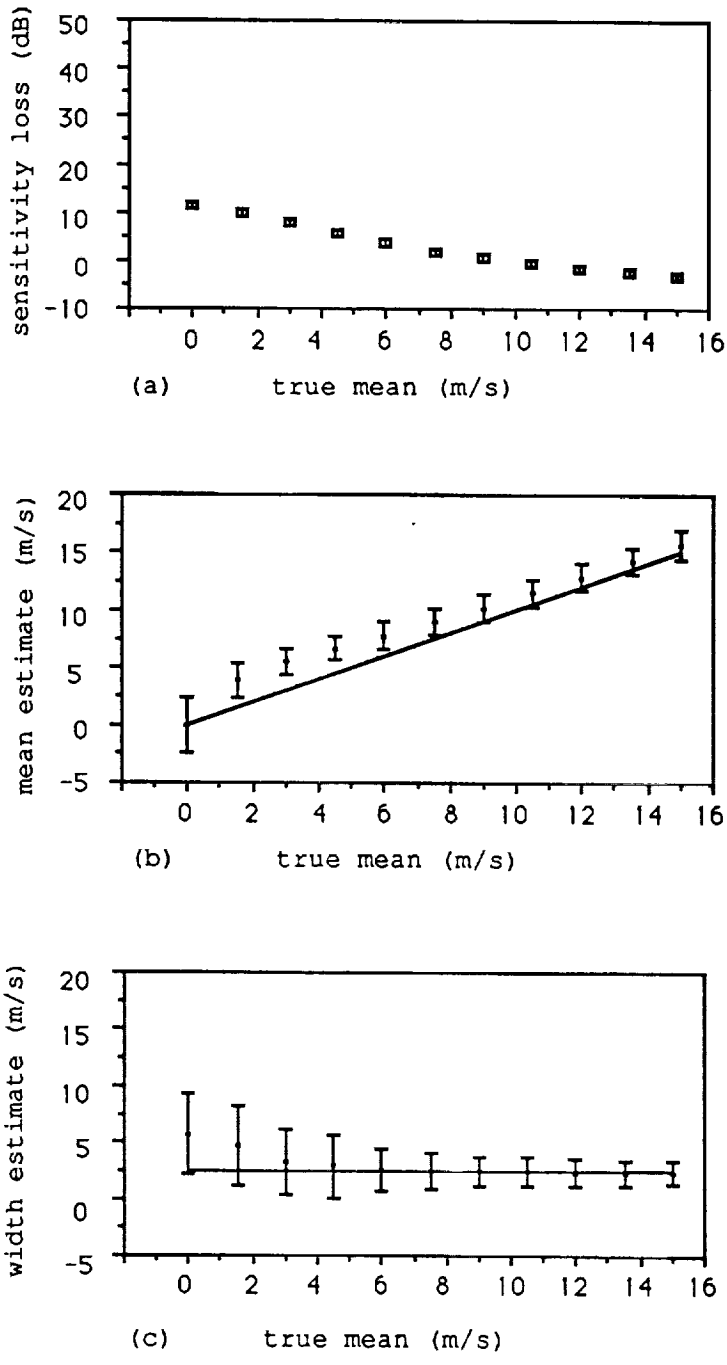
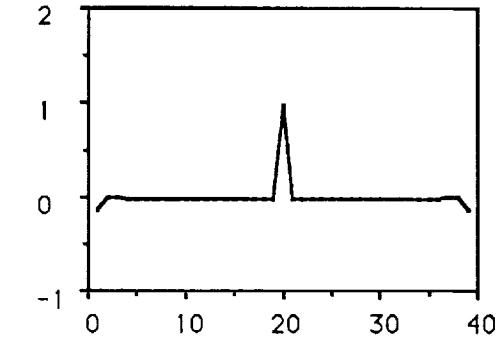


Figure 17. Analysis Software for Pulse Canceller

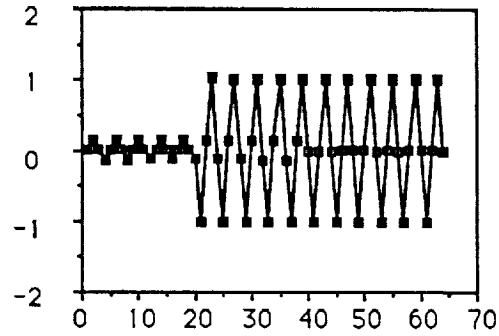
filters considered here, does not have a zero at z = 1. The lowest null, -60.30 dB occurs at a frequency of 0.09 m/s. Figure 18a represents the 39-tap FIR filter's unit-sample response, which resembles a 20-sample-delay delta function. This delay is also exhibited in Figure 18b, which represents the filter's response to a sinusoid in its passband. The filter's magnitude response is plotted from zero to π radians in Figure 18c. The magnitude response is -30.02 dB when the frequency is 0.0 m/s and -3 dB when the frequency is 0.85 m/s. This figure shows that the magnitude response has a $\pm$2 dB ripple in the filter's passband. Figure 18d represents the filter's magnitude response from 0 to $\pi/25$ radians. This figure shows that the filter's notch is not its deepest at 0.0 m/s. Figure 18e illustrates that the phase response of the 39-tap FIR filter is linear.

Figure 19 represents the analysis output using the 39-tap FIR filter. The loss of sensitivity is 1.04 dB when the true weather mean is 0.0 m/s and decreases to -0.17 dB when the true mean is 15.0 m/s. A small ripple in this curve occurs for large values of true means because of the ripple in the filter's passband. The post-filter mean estimate bias is approximately zero at a true mean of 0.0 m/s. As the true mean is increased to 3.0 m/s, the mean estimate bias increases to 0.38 m/s. When the true mean is increased to 11.5 m/s, the mean estimate bias becomes 0.11 m/s. The bias approaches zero as the specified mean is increased further.

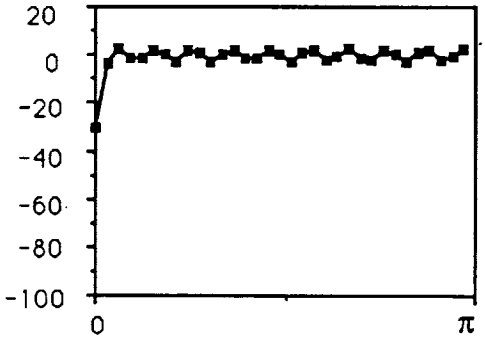
The post-filter width estimate is 2.91 m/s when the true mean is 0.0 m/s. As the true mean is increased, the width estimate bias fluctuates slightly as it approaches zero. The variance of the estimate does not change significantly.



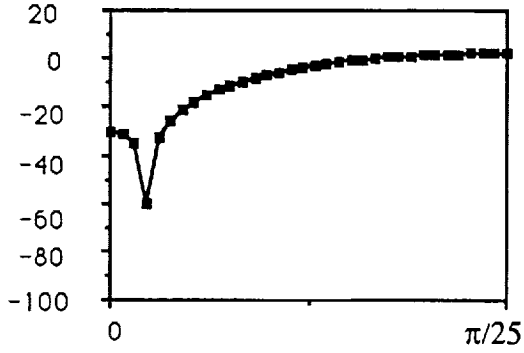
(a) Unit-sample Response



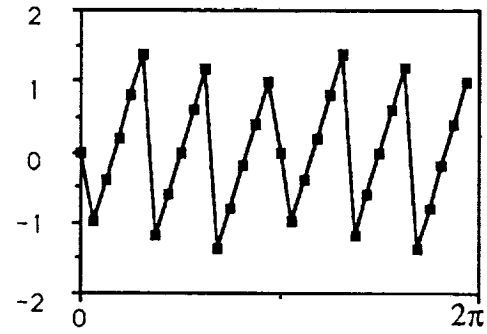
(b) Response to a Sinusoid of Period $\pi/2$



(c) Magnitude Response in dB



(d) Magnitude Response in dB



(e) Phase Response in Radians

Figure 18. Filter Characteristics of 39-tap FIR Filter

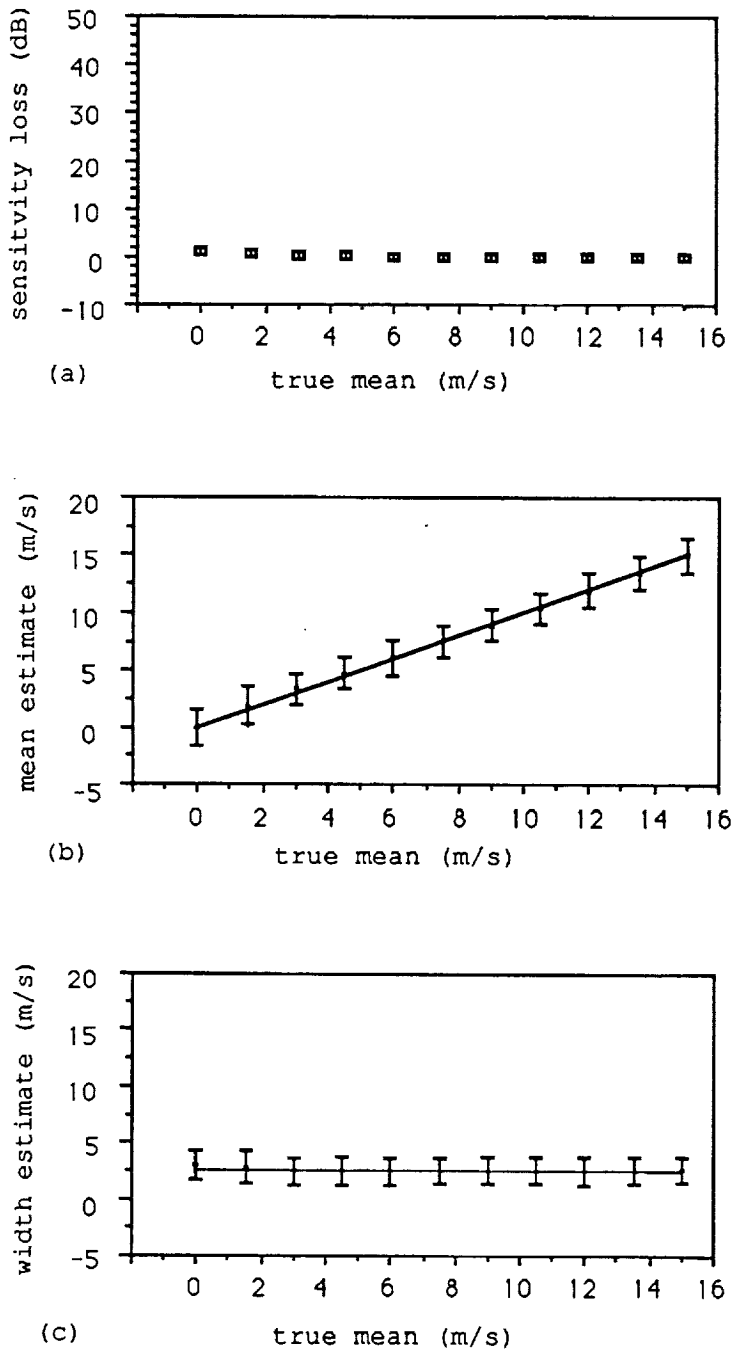
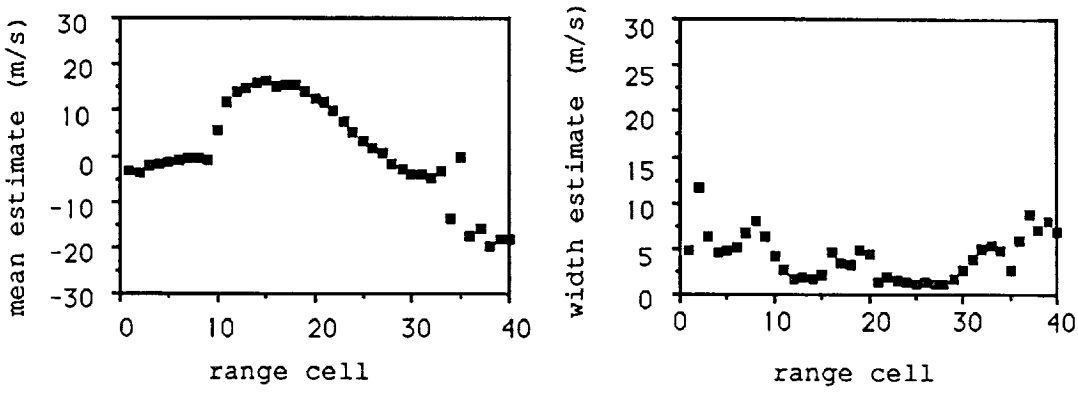


Figure 19. Analysis Software Results for 39-tap FIR Filter

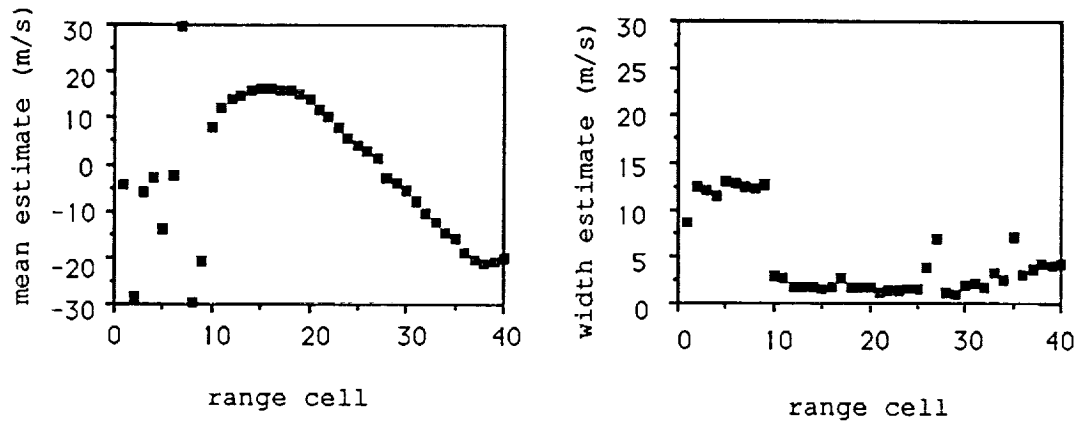
Results from External Data

Results are presented using external data, obtained from a radar-return simulation model [1] which creates typical radar returns that might result from an aircraft radar sensing a microburst which is on the final approach glide path. The magnitude response for each range cell is provided in Appendix B. The data in range cells 1 - 9 consists only of clutter, and the data in range cells 10 - 40 consists of a wet microburst and clutter. The energy of the clutter in range cells 30 - 35 is much greater and more distributed than that in the other range cells. Results of the analysis from the 40 range cells are presented in Figure 20.

The estimate of the average windspeed estimated using the pulse-pair algorithm given in (6) and (9) is plotted in Figure 20a for each range cell. The estimate is approximately -2.5 m/s for range cells 1 - 9, the range cells which consist of clutter-only data. The estimate then increases steadily to 18 m/s at range cell 15 and then decreases to -18 m/s at range cell 38. The estimate fluctuates at range cells 30 - 35 due to the high clutter levels, yielding erroneous estimates for the average weather windspeed. The estimate of the average windspeed after filtering the data with the normalized first-order Butterworth filter is presented in Figure 20b. The estimate appears random for the first nine range cells because the filter almost completely eliminates the signal. The curve for the remaining range cells, however, is smoother than the curve before filtering. It, too, increases steadily to 18 m/s at range cell 15, and then decreases steadily to -18 m/s at range cell 38. The filter



(a) Without Filtering Radar IQ Data



(b) After Filtering Radar IQ Data
using Unnormalized 1st-order Butterworth Filter

Figure 20. Pulse-pair Estimates vs. Range Cell

reduces the variation of the mean estimate in range cells 30 - 35, resulting in a more accurate measure of the average weather windspeed.

Figure 20c represents the width estimate versus range without filtering. The width estimate fluctuates between 4 m/s and 12.5 m/s across all range cells. The post-filter width estimate is plotted versus range in Figure 20d. The estimate is generally much higher in the first nine range cells, which contain clutter-only data, when compared with the remaining range cells. Range cells 26 and 27 have a large width estimate because the mean of the weather is around 0.0 m/s, and range cell 35 has a large width estimate because the clutter energy is unusually large and distributed in that range cell.

As can be seen in Appendix B, the signal, centered away from zero Doppler, is well separated from the clutter, centered at zero Doppler, in range cell 16. The clutter level is about 40 dB above the -100 dB noise level. Therefore, about 40 dB of clutter suppression appears to be required. Although it is not easily seen from Figure 20, the mean and width estimates before filtering were 15.5 m/s and 5.0 m/s, respectively. The post-filter estimates were then determined considering the normalized first-order Butterworth filter presented earlier, and are 16.5 m/s and 2.0 m/s, respectively. This shows that the Butterworth filter yields meaningful weather estimates for this case. These results also show that the pulse-pair technique yields estimates which are averages of the entire spectrum.

Evaluation of the Effects of the Filter's Phase Response

The effects of the filter's phase response on the post-filter estimates were evaluated considering the normalized first-order

Butterworth filter. A theoretical solution was developed and compared to the results of the analysis software using simulated data. The number of pulses considered was 128, and 200 trials in the Monte Carlo procedure were used.

As seen from (6) and (9) in Chapter III, both the pulse-pair spectrum mean and spectrum width estimates involve the computation of specific values of the complex autocorrelation of the radar's IQ data. With a clutter rejection filter specified by a system function $H(\omega)$, the filtered IQ sequence $y(n)$ is used in this autocorrelation computation. Consider a linear, time-invariant system where $x(n)$ is a complex sequence representing the radar's IQ data at the system input. Defining $S_{xx}(\omega)$ as the spectrum of $x(n)$, $S_{yy}(\omega)$ as the spectrum of $y(n)$, and "$\star$" as a linear convolution operator, then [17]

$$S_{yy}(\omega) = |H(\omega)|^2 S_{xx}(\omega) .$$

This states that the spectrum of $y(n)$ depends on the filter's amplitude response but does not depend on the filter's phase response. The autocorrelation of the post-filter IQ data can be determined as [17]

$$\begin{aligned} R_{yy}(T) &= F^{-1}\{|H(\omega)|^2 \, S_{xx}(\omega)\} \\ &= F\text{-}1\{|H(\omega)|^2 \, F\{R_{xx}(T)\}\} \\ &= F^{-1}\{|H(\omega)|^2\} \star F^{-1}\{F\{R_{xx}(T)\}\} \\ &= F^{-1}\{|H(\omega)|^2\} \star R_{xx}(T) . \end{aligned}$$

where $F^{-1}\{\cdot\}$ is the Fourier transform operator. This result shows that the autocorrelation at the input of the filter specified by $H(\omega)$ is

modified only by the magnitude of $H(\omega)$ in determining the autocorrelation of the output. The filter's phase response is not relevant. This result can be reduced further in terms of a unit-sample response to yield

$$\begin{aligned} R_{yy}(T) &= F^{-1}\{H^*(\omega)H(\omega)\} \star R_{xx}(T) \\ &= F^{-1}\{H^*(\omega)\} \star F\text{-}1\{H(\omega)\} \star R_{xx}(T) \\ &= [h^*(-n) \star h(n)] \star R_{xx}(T). \end{aligned}$$

This analysis shows that the complex autocorrelation after filtering is not affected by the phase response of the filter. Therefore the phase response of the filter theoretically has no effect on the post-filter spectrum mean and width estimates as computed using the pulse-pair algorithm given by (6) and (9) of Chapter III.

To further investigate the effects of the clutter-rejection filter's phase response, simulation analysis results obtained using the unnormalized Butterworth filter's magnitude and phase response are compared to results obtained using the filter's magnitude response with a zero phase response and also the filter's magnitude response with a random phase response. The results are plotted in Figure 21. Figure 21a shows that the phase response of the filter has no effect on the average value of the post-filter mean estimate. A random phase response, however, does appear to affect the variance of the mean estimate, particularly when the specified weather mean is small, as Figure 21b illustrates.

Figure 21c shows the post-filter width estimate as a function of the true weather mean for the three cases. This plot illustrates that

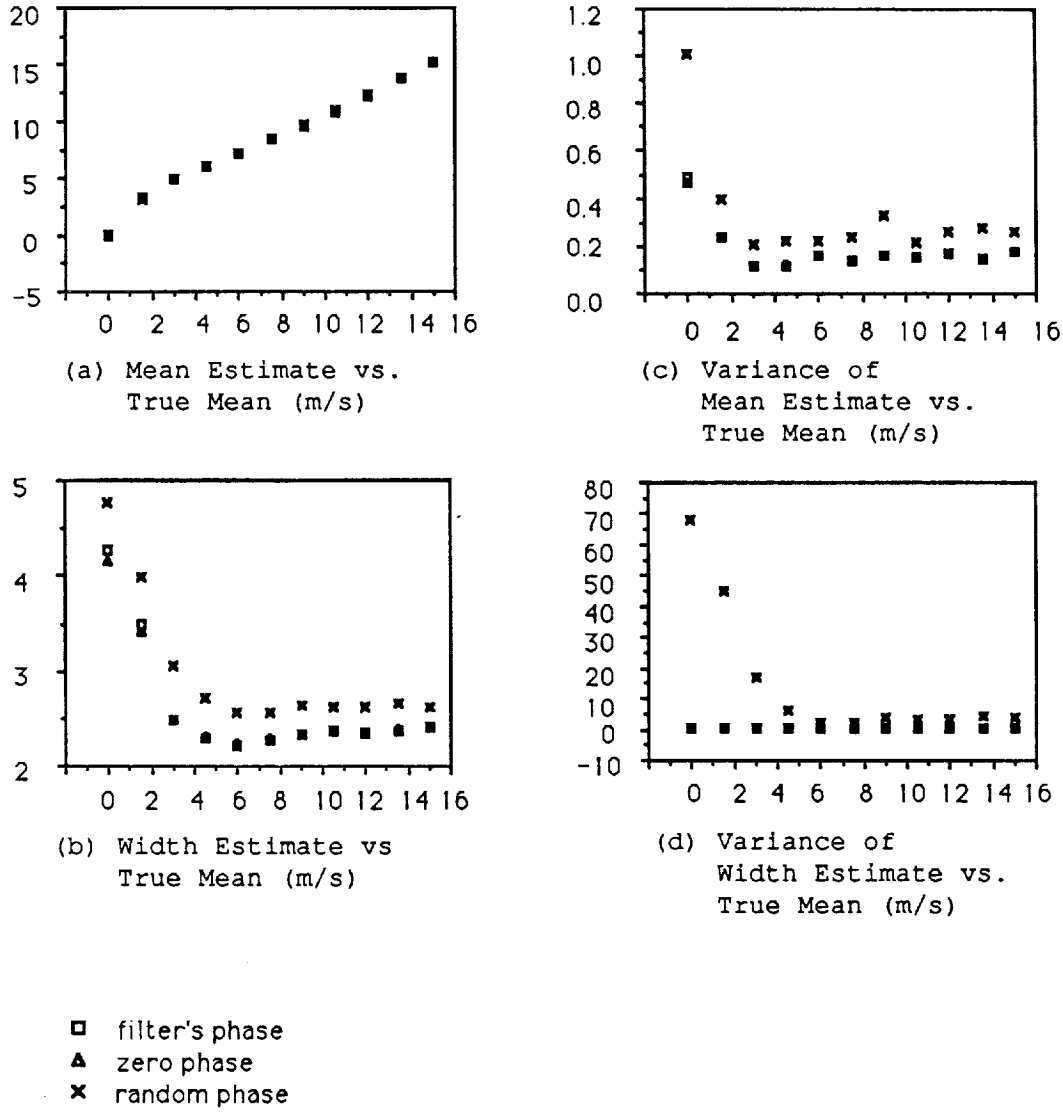


Figure 21. Results of Evaluation of Filter's Phase Response on Pulse-pair Estimates

forcing a random phase response does appear to affect the average value of the post-filter width estimate for this case. In this analysis, forcing a random phase response affected the variances of both estimates, especially at small values of the true weather mean. The random phase response also has a small effect on the post-filter width estimate. Those effects occur because of the small number of samples, 128, used in the simulation. Although the results are not presented here, when the number of samples is increased, the performance of the random phase response filter approaches that of a filter that has a linear phase response. These results illustrate that a linear phase response clutter-rejection filter is certainly not required with pulse-pair processing. When considering the estimation of weather parameters using short data records, however, these results suggest that the clutter rejection filter phase response should be well-behaved, i.e. it should not have large instantaneous fluctuations as frequency varies.

CHAPTER V

CONCLUSIONS

Airborne radar sensing can complement ground-based sensing in the forward-looking detection of hazardous wind conditions. This thesis concerns the problem of cancelling clutter from the return of an airborne-based pulse-Doppler radar. Emphasis has been placed on developing a tool to aid in the understanding of how clutter-rejection filters affect the ability to estimate weather parameters.

Rejecting the clutter will almost always affect the weather signal since the two signals have concurrency in both the time and frequency domain. Understanding how a clutter-rejection filter affects the ability to derive useful weather information from the radar I/Q data, therefore may be important. A flexible software tool designed to evaluate these effects is described and validated. The pulse-pair technique was used to estimate the average windspeed and average turbulence per range cell. Another widely used technique, the Fourier method, for deriving useful weather was not considered in this thesis.

Clutter data is continually being evaluated to determine the optimum clutter-rejection filter characteristics. Only distributed clutter near 0.0 m/s was considered in the analysis results presented here. A provision has been made in the software, however, to allow for the evaluation of virtually any form for clutter-rejection filters since the user may specify the filter in the frequency domain or time domain. Generally 50 dB of clutter suppression is required at the filter's notch [1,4,11]. The results of this thesis may lead to

specific requirements for the width of the filter's notch, width of the filter's transition band, and shape of the filter's phase response. This thesis also shows that with pulse-pair processing, a linear phase response is not a requirement of a clutter-rejection filter.

The analysis results presented here give an indication of the maximum error in estimating the weather statistics introduced by a particular filter. The 'best case' is considered: a weather signal with a very high signal-to-noise ratio and no clutter. Therefore, a filter whose maximum error in estimating the weather statistics exceeds the maximum tolerable error introduced by clutter cancellation may be rejected on this basis.

The ideal filter whose notch width is ± 3.00 m/s affects the weather statistics to a much greater extent than the normalized 1st-order Butterworth filter with a 3-dB notch width of ±6.15 m/s, particularly when the filter's notch and the weather spectrum occupy the same frequency range. This does not suggest, however, that the Butterworth filter should be preferred over the ideal filter since, of course, the ideal filter would more completely reject the clutter. The results also indicate that the resulting notch bandwidth of cascaded filters is wider than that of one stage, causing a greater bias error in the weather parameters. The cascaded filter will of course also eliminate more clutter than the single-stage filter.

The analysis of the external data illustrates the effects of clutter rejection filtering on the radar IQ data on a cell by cell basis. The evaluation of the filter's phase response shows that certainly a linear phase response is not required of a clutter-rejection filter when the pulse-pair technique is used to

estimate the weather spectrum mean and width. A phase response that does not fluctuate greatly over frequency is desired, however, since for short data records the quality of the pulse-pair estimates were not as good with a random phase response clutter rejection filter.

This research indicates that 128 pulses (samples) may be sufficient for estimating parameters. The results considered in this thesis for 128 pulses are almost identical to that for 512 samples.

This work provides a means to evaluate future clutter-rejection filter designs which result from a more thorough analysis of the clutter characteristics associated with airborne radar measurements in the vicinity of urban airports.

APPENDICES

Appendix A

Software Listing

This appendix contains the Fortran code for two programs designed to evaluate the effects of notch filtering on estimating weather parameters. One simulates a truncated Gaussian magnitude spectrum, and the other reads in a complex weather time series which may be representative of radar returns typical of an aircraft radar illuminating a microburst located along the final approach glide path.

Program for Simulated Data

The user specifies the number of radar pulses, the radar's pulse repetition frequency (PRF), the normalized (Nyquist bandwidth scaled from -1 to 1) one-sided spectrum width of the simulated Gaussian spectrum, and the number of repeated trials in the Monte Carlo Procedure. The PRF must be a power of two because of the fast Fourier transform (FFT) routine used. The user then choses any combination of three types of filters to implement.

The first two types are frequency-domain filters. One type filters the data in the frequency domain without regard to a time-domain equivalent filter. It is useful for implementing an ideal filter, one with a unity gain in its passband(s), a zero gain in its stopband(s), and an arbitrary phase response. The other frequency-domain filter filters the data with regard to its time domain causal-realizable equivalent filter [14]. The complex frequency response of each filter is read from an external file corresponding to frequencies ranging from 0 to 2π. The third type is a time-domain filter. The user can

implement the filter by reading the filter's complex unit-sample response from an external file or by adding the code to implement the filter's difference equation.

The Gaussian magnitude spectrum is simulated and filtered. The post-filter spectrum mean and width estimates are calculated using the pulse-pair technique [2]. The loss of sensitivity, S, in decibels defined by

$$S = -10\log_{10}\left[\frac{P_a}{P_b}\right]$$

is also computed, where P_a is the total power of the post-filter radar in-phase and quadrature-phase (IQ) data, and P_b is the total power of the radar IQ data at the filter's input.

This procedure is repeated, and the expected value of the estimate of each weather parameter, E(x), is estimated by [15]

$$E(x) = \frac{1}{N}\sum_{i=1}^{N}(x_i)$$

where x_i is the i^{th} estimate of the weather parameter, x, and N is the number of trials in the Monte Carlo procedure. The true value of the weather parameter, x_T, is used to estimate a bias error given by

$$E\{x - x_T\} = E\{x\} - x_T$$

where x_T corresponds to the theoretical value of the spectrum mean or

spectrum width [2]. The estimate of the variance of a weather parameter, VAR(x), is given by [15]

$$VAR(x) = \frac{\sum_{i=1}^{N} (x_i - \bar{x})^2}{N - 1}$$

where $\bar{x}$ is the expected value of the weather parameter given above. The theoretical variances of the pulse-pair estimates [2] are also calculated.

Program for External Data

This program reads in a complex time series and filters the data as in the previous program. The loss of sensitivity and pulse-pair mean and width estimates are computed. The inputs to this program are the number of pulses and the radar's PRF. The Monte Carlo procedure is not used.

```
c     ****************************************************************
c     Program for Simulated Data
c     ****************************************************************
c
      dimension g(1024), s(1024), xp(1024), p(1024), sx(1024),
     c    sy(1024), x(1024), y(1024), areal(1024), aimag(1024),
     c    yimean(1024), yistd(1024), yimean2(1024), yistd2(1024),
     c    tfiltr(1024), tfilti(1024), filtr(1024), filti(1024),
     c    tdatar(1024), tdatai(1024), hreal(1024), himag(1024),
     c    datar(1024), datai(1024)
c
      open (unit = 1, file = 'out1.dat', status = 'unknown')
      open (unit = 2, file = 'out2.dat', status = 'unknown')
c
c     ****************************************************************
c     Input parameters
c     ****************************************************************
c
      print*, 'enter number of samples ( a power of 2 )'
         read*, n
      print*, 'enter normalized width'
         read*, std
      print*, 'enter pulse repetition frequency'
         read*, fs
      print*, 'enter number of trials'
         read*, m
      print*, 'Do you wish to implement an ideal filter'
      print*, '   1 = YES   0 = NO'
         read*, q1
      print*, 'Do you wish to filter data in the frequency domain'
      print*, '   1 = YES   0 = NO'
         read*, q2
      print*, 'Do you wish to filter data in the time domain'
      print*, '   1 = YES   0 = NO'
         read*, q3
c
      pi = 4.0 * atan(1.0)
c
c     ****************************************************************
c     Determine theoretical variance of mean and width estimates
c     ****************************************************************
c
      theomean = std / (2.0 * sqrt(pi) * real(n))
      theostd = 3.0 * std / (16.0 * sqrt(pi) * real(n))
c
c     ****************************************************************
c     Headers for output files
c     ****************************************************************
c
      write(1,*) 'number of samples                    ', n
      write(1,*) 'normalized width                     ', std
      write(1,*) 'pulse repetition frequency           ', fs
      write(1,*) 'number of trials                     ', m
      write(1,*) '     '
      write(1,*) '  true mean    est mean   mean bias',
     c '   est width  width bias'
c
      write(2,*) 'number of samples                    ', n
```

```
      write(2,*) 'normalized width                    ', std
      write(2,*) 'pulse repetition frequency          ', fs
      write(2,*) 'number of trials                    ', m
      write(2,*) 'true variance of mean estimate      ', theomean
      write(2,*) 'true variance of width estimate     ', theostd
      write(2,*) '     '
      write(2,*) '  true mean      est var    est var ',
     c '     signal  '
      write(2,*) '                 of mean   of width  ',
     c 'power loss   '
c
c     ****************************************************************
c     Simulation of truncated Gaussian magnitude spectrum
c     ****************************************************************
c
      n2 = 2.0 * n
      prp = 1.0 / fs
      snr = 200
      gs = 0.0
      i1 = 0
      i2 = 0
c
      do 1000 k = 1, 11
        fm = 0.05 * real(k - 1)
        avgmean = 0.0
        avgwidth = 0.0
c
      do 2000 l = 1, m
c
      sumg = 0.0
      do 10 i = 1, n
        f = -1.0 + 2.0 * real(i - 1) / real(n)
        exponent = (-(f - fm) ** 2) / (2.0 * std ** 2)
        g(i) = exp(exponent) / (sqrt(2.0 * pi) * std)
        sumg = sumg + g(i)
10    continue
c
      pt = 0.0
      do 20 i = 1, n
        c = (10.0 ** (snr / 10.0)) / sumg
        s(i) = c * g(i)
        call randu(i1, i2, a)
        xp(i) = -1.0 * alog(a) * (s(i) + 1.0 / real(n))
        pt = pt + xp(i)
 20   continue
c
      do 30 i = 1, n
        p(i) = xp(i) / pt
        call randu(i1, i2, b)
        sx(i) = sqrt(p(i)) * cos(2.0 * pi * b)
        sy(i) = sqrt(p(i)) * sin(2.0 * pi * b)
 30   continue
c
c     shift data so that range of spectrum will be -pi/2 to pi/2
c
      powerb4 = 0.0
      do 40 i = 1, n
        if (i .le. n / 2) x(i) = sx(i + n / 2)
```

```
          if (i .le. n / 2) y(i) = sy(i + n / 2)
          if (i .gt. n / 2) x(i) = sx(i - n / 2)
          if (i .gt. n / 2) y(i) = sy(i - n / 2)
        powerb4 = powerb4 + x(i)**2 + y(i)**2
 40     continue
        powerb4 = powerb4 / real(n)
c
c       ****************************************************************
c       Ideal Filter Implementation
c       ****************************************************************
c
        if (q1 .eq. 0) goto 44
        open (unit = 3, file = 'ideal.dat', status = 'unknown')
        do 42 i = 1, n
           read (3,*) filtr(i), filti(i)
 42    continue
       do 43 i = 1, n
           x(i) = filtr(i) * x(i) - filti(i) * y(i)
           y(i) = filti(i) * x(i) + filtr(i) * y(i)
 43     continue
        close (unit = 3)
c
c       ****************************************************************
c       Frequency Domain Filter Implementation
c       ****************************************************************
c
 44     if (q2 .eq. 0) goto 70
        open (unit = 4, file = 'freq.dat', status = 'unknown')
        do 46 i = 1, n
           read (4,*) filtr(i), filti(i)
           datar(i) = x(i)
           datai(i) = y(i)
 46     continue
c
        call fft842(1, n, filtr, filti)
c
        do 50 i = 1, n
           tfiltr(i) = filtr(i)
           tfilti(i) = filti(i)
 50     continue
c
        call fft842(1, n, datar, datai)
c
        do 52 i = 1, n
           tdatar(i) = datar(i)
           tdatai(i) = datai(i)
 52     continue
c
        do 54 i = 1, n2
           if (i .gt. n) tfiltr(i) = 0.0
           if (i .gt. n) tfilti(i) = 0.0
           if (i .gt. n) tdatar(i) = 0.0
           if (i .gt. n) tdatai(i) = 0.0
 54     continue
c
        call fft842(0, n2, tfiltr, tfilti)
c
        do 56 i = 1, n2
```

```
            filtr(i) = tfiltr(i)
            filti(i) = tfilti(i)
 56      continue
c
        call fft842(0, n2, tdatar, tdatai)
c
        do 58 i = 1, n2
            datar(i) = tdatar(i)
            datai(i) = tdatai(i)
 58      continue
c
        do 60 i = 1, n2
            x(i) = filtr(i) * datar(i) - filti(i) * datai(i)
            y(i) = filti(i) * datar(i) + filtr(i) * datai(i)
 60      continue
        close (unit = 4)
c
c       ****************************************************************
c       Transform Data to Time Domain
c       ****************************************************************
c
 70      nprime = n
        if (q2 .eq. 1) nprime = n2
        call fft842(1, nprime, x, y)
c
        do 72 i = 1, n
            areal(i) = x(i)
            aimag(i) = y(i)
 72      continue
c
c       ****************************************************************
c       Time Domain Filter Implementation
c       ****************************************************************
c
        if (q3 .eq. 0) goto 78
        open (unit = 5, file = 'time.dat', status = 'unknown')
        do 74 i = 1, n
            read (5,*) hreal(i), himag(i)
            sumr = 0.0
            sumi = 0.0
            do 76 k2 = 1, i
                sumr = sumr + hreal(i - k2 + 1) * areal(k2)
                sumi = sumi + himag(i - k2 + 1) * aimag(k2)
 76         continue
        x(i) = sumr
        y(i) = sumi
 74     continue
        do 77 i = 1, n
            areal(i) = x(i)
            aimag(i) = y(i)
 77     continue
        close (unit = 5)
c
c       ****************************************************************
c       compute loss of sensitivity
c       ****************************************************************
c
 78     powafter = 0.0
```

```
      do 80 i = 1, n
         powafter = powafter + areal(i)**2 + aimag(i)**2
 80   continue
      powerloss = -10.0 * alog10( powafter / powerb4)
c
c     ****************************************************************
c     compute pulse pair estimates
c     ****************************************************************
c
      realsum = 0.0
      aimagsum = 0.0
c
      do 82 i = 1, n - 1
        prod1 = areal(i) * areal(i + 1)
        prod2 = aimag(i) * areal(i + 1)
        prod3 = areal(i) * aimag(i + 1)
        prod4 = aimag(i) * aimag(i + 1)
        realsum = prod1 + prod4 + realsum
        aimagsum = prod3 - prod2 + aimagsum
 82   continue
c
      amagnrts = sqrt(realsum ** 2 + aimagsum ** 2) / real(n - 1)
      if ((aimagsum .eq. 0.0) .or. (realsum .eq. 0.0)) argrts = 0.0
      if ((aimagsum .ne. 0.0) .and. (realsum .ne. 0.0))
     c argrts = atan(aimagsum / realsum)
      if ((aimagsum .lt. 0.0) .and. (realsum .lt. 0.0))
     c argrts = atan(aimagsum / realsum) - pi
      if ((aimagsum .gt. 0.0) .and. (realsum .lt. 0.0))
     c argrts = atan(aimagsum / realsum) + pi
      sum = 0.0
c
      do 84 i = 1, n
        amaglsqr = areal(i) ** 2 + aimag(i) ** 2
        sum = amaglsqr + sum
 84   continue
c
      r0 = sum / (real(n))
      eofmean = argrts / pi
      amult = 2.0 / ((2.0 * pi * prp) ** 2)
      bmult = abs(1.0 - (abs(amagnrts) / r0))
      eofstd = sqrt(amult * bmult) * 2.0 / fs
c
c     ****************************************************************
c     compute expected value and standard deviation of estimates
c     ****************************************************************
c
      if (l .eq. 1) avgmean = eofmean
      if (l .eq. 1) avgstd = eofstd
      if (l .eq. 1) avgpwr = powerloss
      if (l .gt. 1) avgmean = (avgmean * real (l - 1) + eofmean)
     c / real(l)
      if (l .gt. 1) avgstd = (avgstd * real(l - 1) + eofstd)
     c / real(l)
      if (l .gt. 1) avgpwr = (avgpwr * real(l - 1) + powerloss)
     c / real(l)
c
      biasmean = avgmean - fm
      biasstd = avgstd - std
```

```
c
      yimean(l) = eofmean
      yistd(l) = eofstd
      yimean2(l) = eofmean ** 2
      yistd2(l) = eofstd ** 2
c
      i1 = i1 + 1
      i2 = i2 + 1
c
 2000 continue
c
      yimsum = 0.0
      yimsum2 = 0.0
      yiwsum = 0.0
      yiwsum2 = 0.0
c
      do 86 l = 1, m
        yimsum = yimean(l) + yimsum
        yimsum2 = yimean2(l) + yimsum2
        yiwsum = yistd(l) + yiwsum
        yiwsum2 = yistd2(l) + yiwsum2
 86   continue
c
      vmean = (yimsum2 - ((yimsum ** 2) / real(m))) / (real(m) - 1.0)
      vstd = (yiwsum2 - ((yiwsum ** 2) / real(m))) / (real(m) - 1.0)
c
c     ****************************************************************
c     output data
c     ****************************************************************
c
      write(1,500) fm, avgmean, biasmean, avgstd, biasstd
      write(2,600) fm, vmean, vstd, avgpwr
 500  format (4x, f8.6, 4x, f8.6, 4x, f8.6, 4x, f8.6, 4x, f8.6)
 600  format (4x, f8.6, 4x, f8.6, 4x, f8.6, 4x, f8.4)
c
      print*, k
 1000 continue
c
      close (unit = 2)
      close (unit = 1)
      stop
      end
```

```
c      ****************************************************************
c      Program for External Data
c      ****************************************************************
c
       dimension areal(1024), aimag(1024), breal(1024), bimag(1024),
     c    hreal(1024), himag(1024), x(1024), y(1024), filtr(1024),
     c    filti(1024), datar(1024), datai(1024), tdatar(1024),
     c    tdatai(1024), tfiltr(1024), tfilti(1024)
       open (unit = 1, file = 'out3.dat', status = 'unknown')
       open (unit = 2, file = 'iqrc.dat', status = 'unknown')
c
c      ****************************************************************
c      Input Parameters
c      ****************************************************************
c
       print*, 'enter number of samples ( a power of 2 )'
          read*, n
       print*, 'enter pulse repetition frequency'
          read*, fs
       print*, 'Do you wish to implement an ideal filter'
       print*, '   1 = YES   0 = NO'
          read*, q1
       print*, 'Do you wish to filter data in the frequency domain'
       print*, '   1 = YES   0 = NO'
          read*, q2
       print*, 'Do you wish to filter data in the time domain'
       print*, '   1 = YES   0 = NO'
          read*, q3
c
c      ****************************************************************
c      Header for Output File
c      ****************************************************************
c
       write(1,*) 'number of samples                    ', n
       write(1,*) 'pulse repetition frequency           ', fs
       write(1,*) '     '
       write(1,*) '      range    est mean   est width  power loss'
c
       pi = 4.0 * atan(1.0)
       n2 = 2.0 * n
       prp = 1.0 / fs
c
c      ****************************************************************
c      read external weather time series
c      ****************************************************************
c
       do 2000 l = 1, 40
c
       read(2,01) number, range
 01    format(i4,e12.5)
 02    format(2e12.5)
c
       powerb4 = 0.0
       do 10 i = 1, 512
         read (2,02) areal(i), aimag(i)
         powerb4 = powerb4 + areal(i) ** 2 + aimag(i) ** 2
 10    continue
c
```

```
c      ****************************************************************
c      Time Domain Filter Implementation
c      ****************************************************************
c
       if (q3 .eq. 0) goto 25
       open (unit = 3, file = 'time.dat', status = 'unknown')
       do 15 i = 1, n
          read (3,*) hreal(i), himag(i)
          sumr = 0.0
          sumi = 0.0
          do 20 k2 = 1, i
             sumr = sumr + hreal(i - k2 + 1) * areal(k2)
             sumi = sumi + himag(i - k2 + 1) * aimag(k2)
 20       continue
       x(i) = sumr
       y(i) = sumi
 15    continue
       do 22 i = 1, n
          areal(i) = x(i)
          aimag(i) = y(i)
 22    continue
       close (unit = 3)
c
c      ****************************************************************
c      Transform Data to Frequency Domain
c      ****************************************************************
c
 25    call fft842(0, n, areal, aimag)
c
       do 30 i = 1, n
          x(i) = areal(i)
          y(i) = aimag(i)
 30    continue
c
c      ****************************************************************
c      Ideal Filter Implementation
c      ****************************************************************
c
       if (q1 .eq. 0) goto 45
       open (unit = 4, file = 'ideal.dat', status = 'unknown')
       do 35 i = 1, n
          read (4,*) filtr(i), filti(i)
 35    continue
       do 40 i = 1, n
          x(i) = filtr(i) * x(i) - filti(i) * y(i)
          y(i) = filti(i) * x(i) + filtr(i) * y(i)
 40    continue
       close (unit = 4)
c
c      ****************************************************************
c      Frequency Domain Filter Implementation
c      ****************************************************************
c
 45    if (q2 .eq. 0) goto 85
       open (unit = 5, file = 'freq.dat', status = 'unknown')
       do 50 i = 1, n
          read (4,*) filtr(i), filti(i)
          datar(i) = x(i)
```

```
            datai(i) = y(i)
 50     continue
c
        call fft842(1, n, filtr, filti)
c
        do 55 i = 1, n
           tfiltr(i) = filtr(i)
           tfilti(i) = filti(i)
 55     continue
c
        call fft842(1, n, datar, datai)
c
        do 60 i = 1, n
           tdatar(i) = datar(i)
           tdatai(i) = datai(i)
 60     continue
c
        do 65 i = 1, n2
           if (i .gt. n) tfiltr(i) = 0.0
           if (i .gt. n) tfilti(i) = 0.0
           if (i .gt. n) tdatar(i) = 0.0
           if (i .gt. n) tdatai(i) = 0.0
 65     continue
c
        call fft842(0, n2, tfiltr, tfilti)
c
        do 70 i = 1, n2
           filtr(i) = tfiltr(i)
           filti(i) = tfilti(i)
 70     continue
c
        call fft842(0, n2, tdatar, tdatai)
c
        do 75 i = 1, n2
           datar(i) = tdatar(i)
           datai(i) = tdatai(i)
 75     continue
c
        do 80 i = 1, n2
           x(i) = filtr(i) * datar(i) - filti(i) * datai(i)
           y(i) = filti(i) * datar(i) + filtr(i) * datai(i)
 80     continue
        close (unit = 5)
c
c       ****************************************************************
c       Transform Data to Time Domain
c       ****************************************************************
c
 85     nprime = n
        if (q2 .eq. 1) nprime = n2
        call fft842(1, nprime, x, y)
c
        do 90 i = 1, n
           areal(i) = x(i)
           aimag(i) = y(i)
 90     continue
c
c       ****************************************************************
```

```
c       compute loss of sensitivity
c       ***************************************************************
c
        powafter = 0.0
        do 95 i = 1, n
           powafter = powafter + areal(i)**2 + aimag(i)**2
 95     continue
        powerloss = -10.0 * alog10( powafter / powerb4)
c
c       ***************************************************************
c       compute pulse pair estimates
c       ***************************************************************
c
        realsum = 0.0
        aimagsum = 0.0
c
        do 100 i = 1, n - 1
          prod1 = areal(i) * areal(i + 1)
          prod2 = aimag(i) * areal(i + 1)
          prod3 = areal(i) * aimag(i + 1)
          prod4 = aimag(i) * aimag(i + 1)
          realsum = prod1 + prod4 + realsum
          aimagsum = prod3 - prod2 + aimagsum
 100    continue
c
        amagnrts = sqrt(realsum ** 2 + aimagsum ** 2) / real(n - 1)
        if (realsum .eq. 0.0) argrts = 0.0
        if (aimagsum .eq. 0.0) argrts = 0.0
        if ((aimagsum .ne. 0.0) .and. (realsum .ne. 0.0))
     c argrts = atan(aimagsum / realsum)

        if ((aimagsum .lt. 0.0) .and. (realsum .lt. 0.0))
     c argrts = atan(aimagsum / realsum) - pi
        if ((aimagsum .gt. 0.0) .and. (realsum .lt. 0.0))
     c argrts = atan(aimagsum / realsum) + pi

        sum = 0.0
c
        do 110 i = 1, n
          amag1sqr = areal(i) ** 2 + aimag(i) ** 2
          sum = amag1sqr + sum
 110    continue
c
        r0 = sum / (real(n))
        eofmean = 30.02 * argrts / pi
        amult = 2.0 / ((2.0 * pi * prp) ** 2)
        bmult = abs(1.0 - (abs(amagnrts) / r0))
        eofstd = 30.02 * sqrt(amult * bmult) * 2.0 / fs
c
        write(1,500) real(1),eofmean, eofstd, powerloss
 500    format (4x, f8.4, 4x, f8.4, 4x, f8.4,4x,f8.4)
c
 2000   continue
        close (unit = 2)
        close (unit = 1)
        stop
        end
```

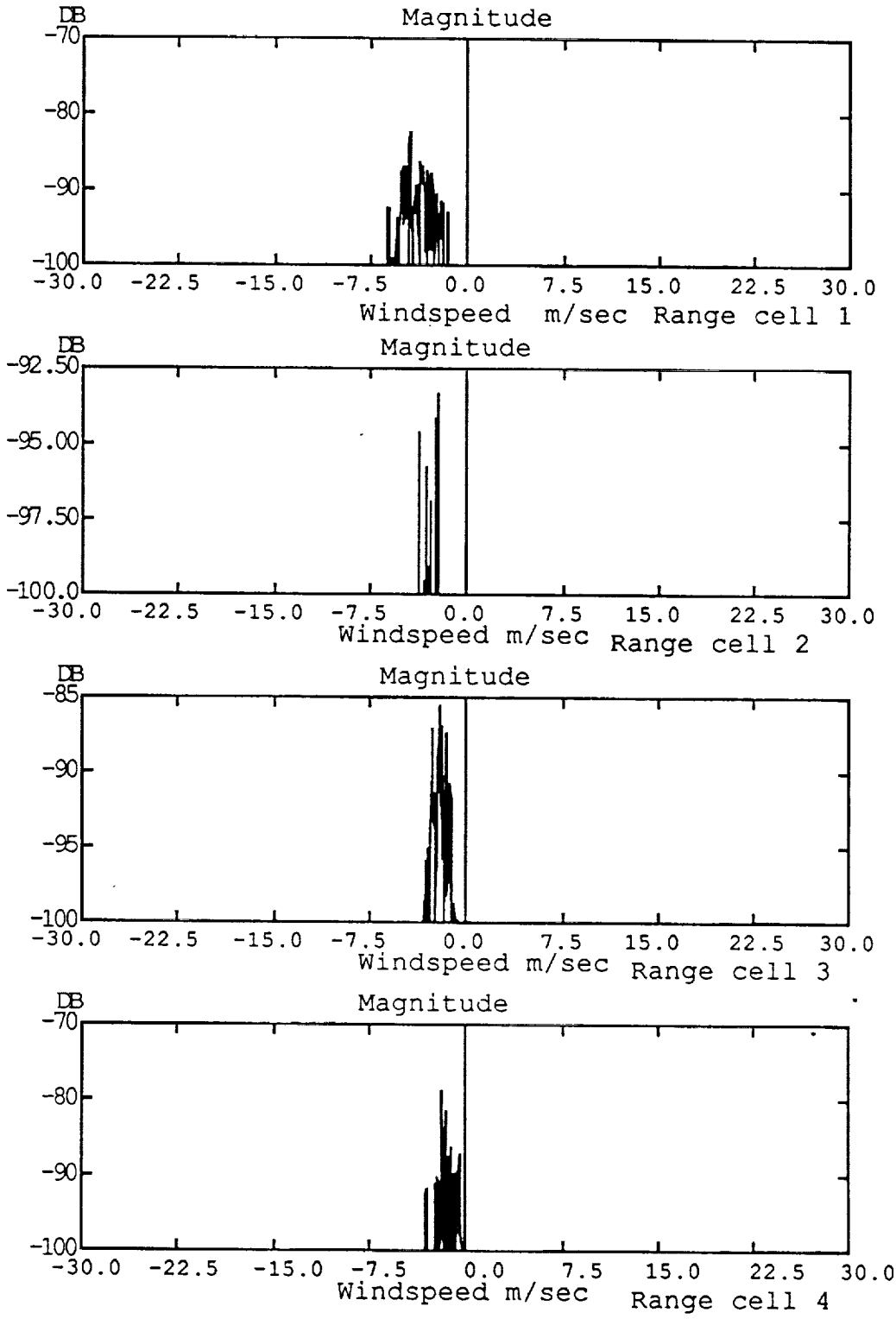
Appendix B

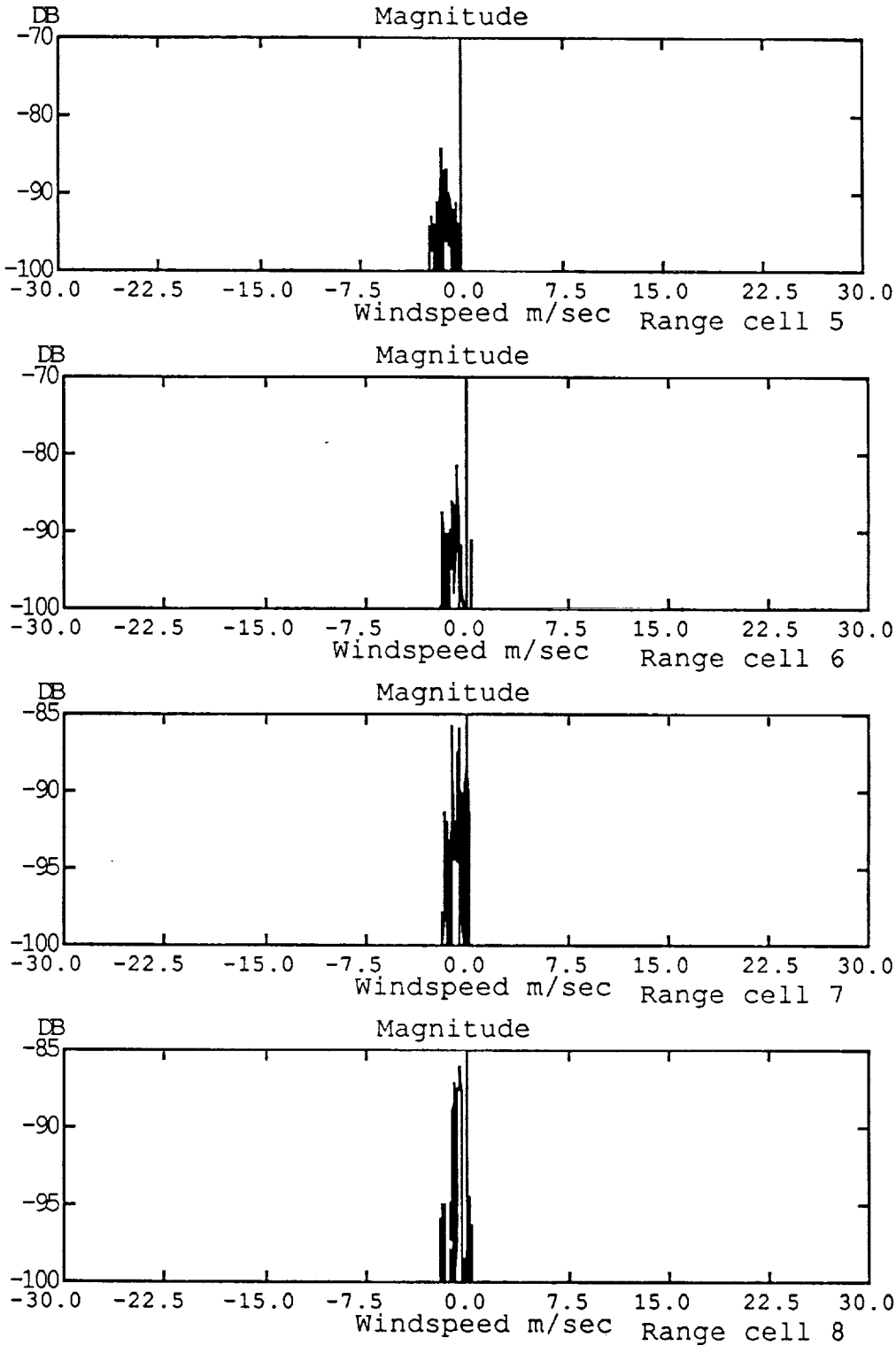
Externally Generated Data

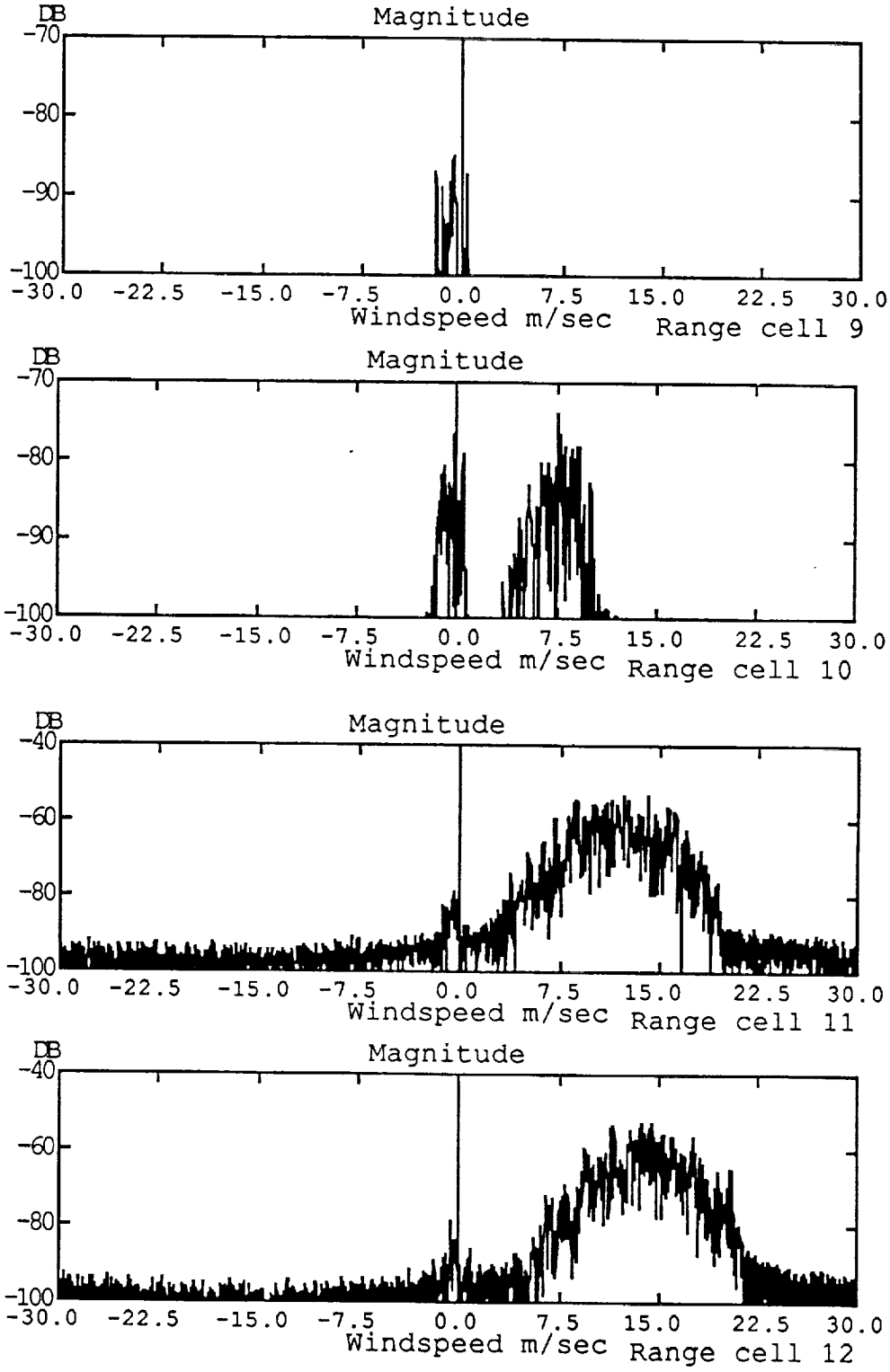
The magnitude of the Fourier transform of simulated data is presented. The simulation utilizes a finite element approach to create a radar return typical of an airborne-based radar sensing a microburst which is on the final approach glide path [1]. The input parameters to the simulation are given. The aircraft is located 7 km from touchdown, traveling on a three degree glideslope. The microburst is located 2 km before the touchdown point. The number of range cells processed is 40, with the initial range cell located 1 km in front of the aircraft.

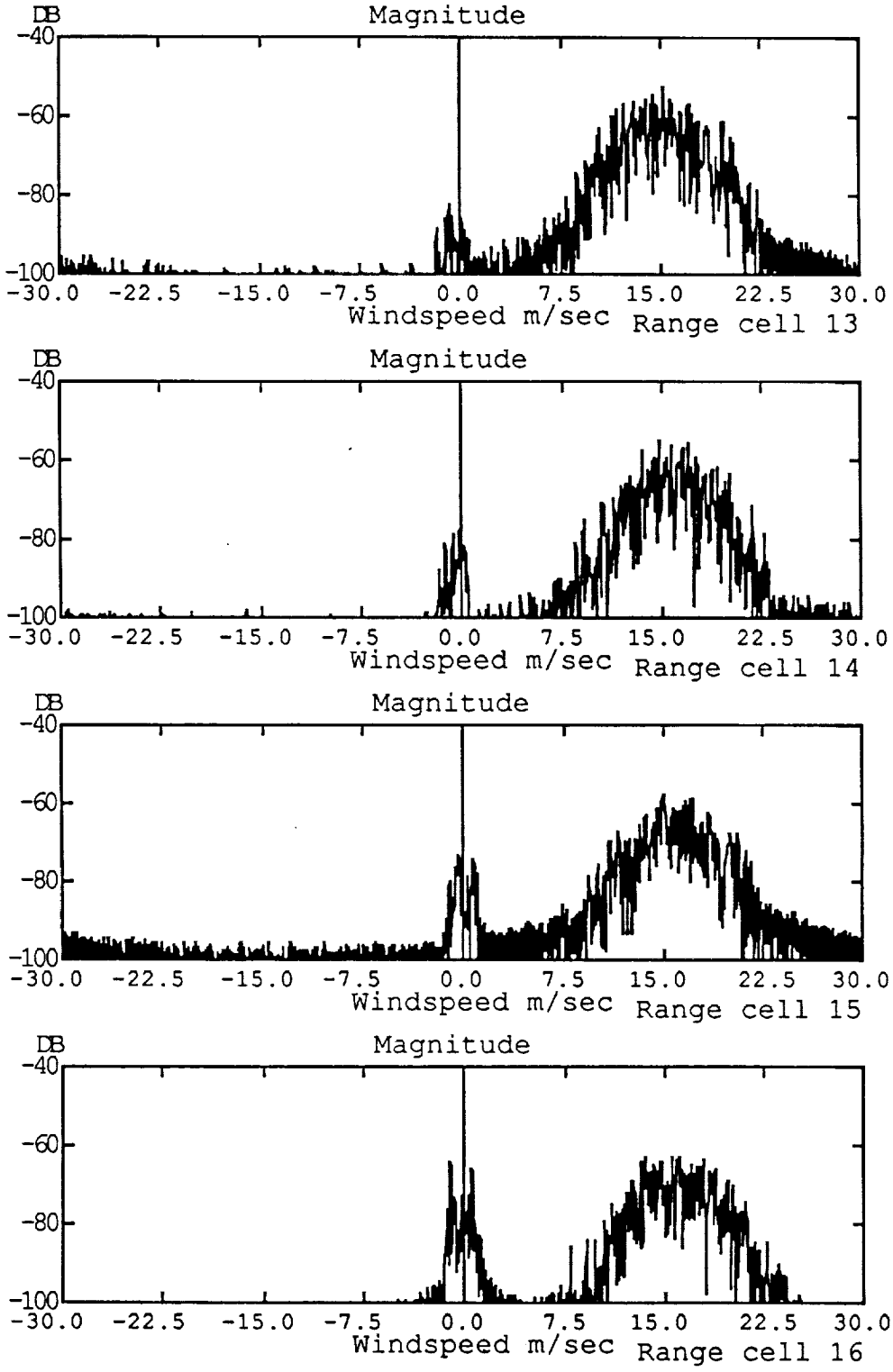
SIMULATION PARAMETERS	0.
	0.
A/C Distance to touchdown (km)	7.
Aircraft Velocity (kts)	150.
Glideslope Angle (deg)	3.
Roll Attitude (deg)	0.
Pitch Attitude (deg)	0.
Yaw Attitude (deg)	0.
Az Integration Range/2 (deg)	6.0
Az Integration Increment (deg)	.3
El Integration Range/2 (deg)	4.0
El Integration Incerment (deg)	.2
Rng Integration Increment (m)	100.
Random Number Seed (0 - 1)	.224
	0.
MICROBURST & CLUTTER	0.
	0.
Along Track Offset from TD (km)	-2.
Cross Track Offset fron TD (km)	0.
Rain Standard Deviation (m/s)	1.
Clutter Standard Deviation (m/s)	.5
Clutter Calc. Flag (1=ON, 0=OFF)	1.
Reflectivity Calc. Thres. (dBz)	200.
Minimum Reflectivity (dBz)	-15.
Attenuation Code (0,1,2)	2.
	0.
RADAR PARAMETERS	0.
	0.
Initial Radar Range (km)	1.
Number of Range Cells	40.
Antenna Az - if no scan (deg)	0.

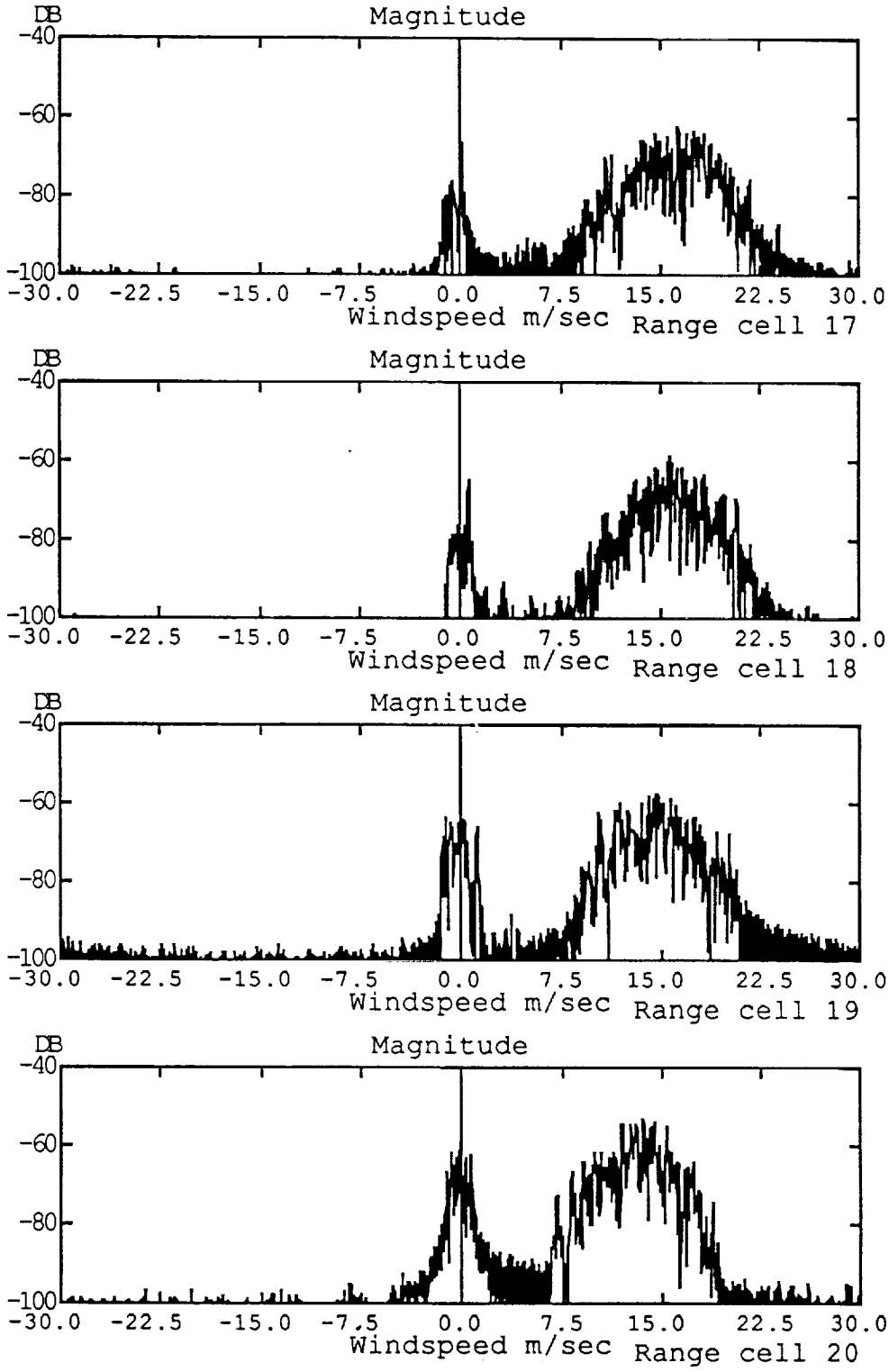
Azimuth Scan Range/2 (deg)	0.
Azimuth Scan Increment (deg)	3.
Antenna Elevation (deg)	1.
Transmitted Power (watts)	2000.
Frequency (GHz)	9.3
Pulse Width (microsecs)	1.
Pulse Interval (microsecs)	268.6
Receiver Noise Figure (dB)	4.
Receiver Losses (dB)	3.
Antenna Type (1=para., 2=flat)	1.
Antenna Radius (m)	.381
Aperature Taper Parameter	.316
RMS Trans. Phase Jitter (deg)	.2
RMS Trans. Freq. Jitter (Hz)	0.
	0.
SIGNAL PROCESSING	0.
	0.
Number of Pulses	512.
Number of A/D bits	12.
AGC Gain Factor	.6
Processing Threshold (dB)	4.
Clutter Filter Code (-2 to N)	0.
Clutter Filter Cutoff (m/s)	3.
No. of Bins for F-factor Avr.	5.

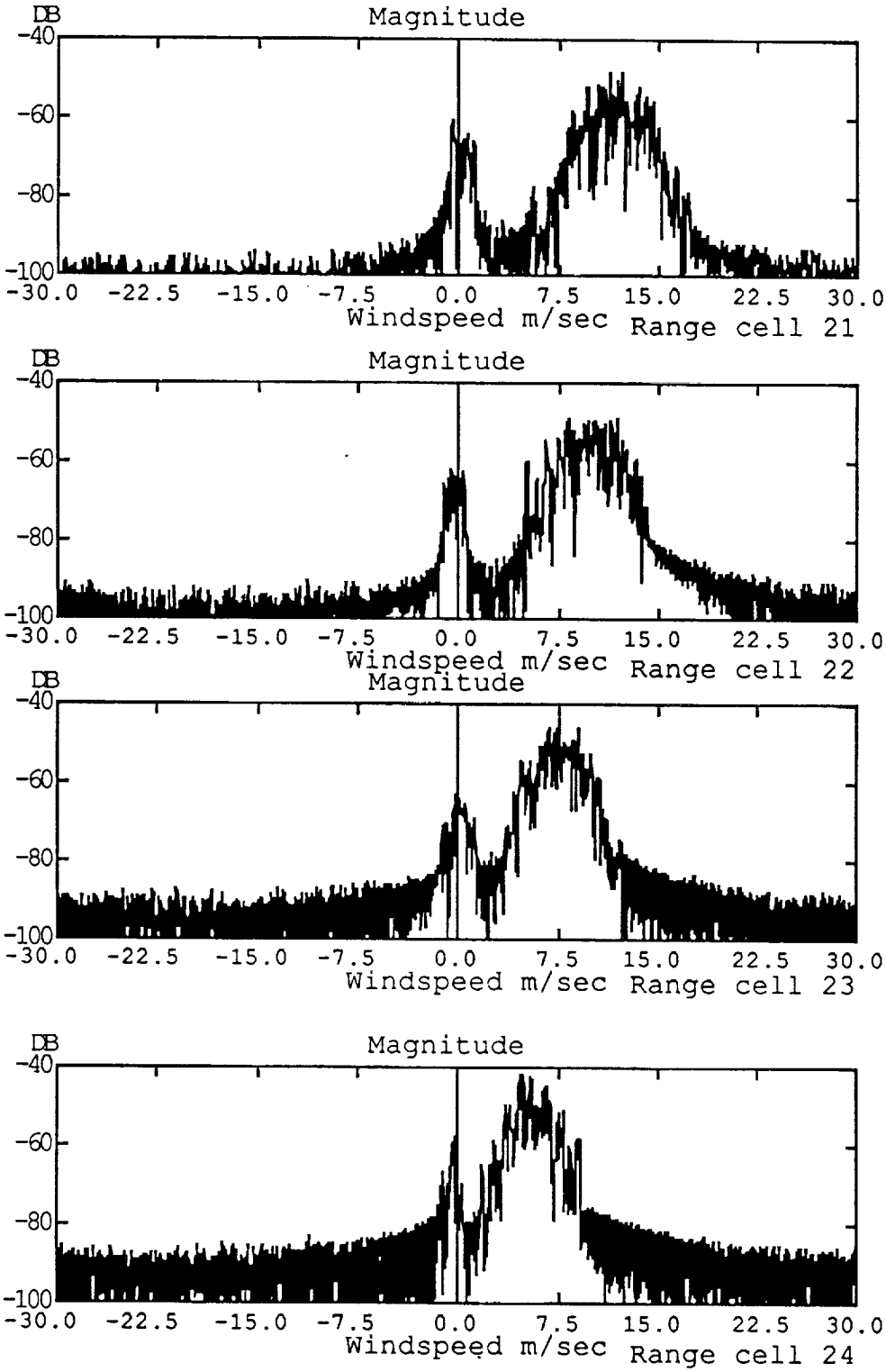

DB
Magnitude
-70
-80
-90
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec Range cell 1
DB
Magnitude
-92.50
-95.00
-97.50
-100.0
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec Range cell 2
DB
Magnitude
-85
-90
-95
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec Range cell 3
DB
Magnitude
-70
-80
-90
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec Range cell 4

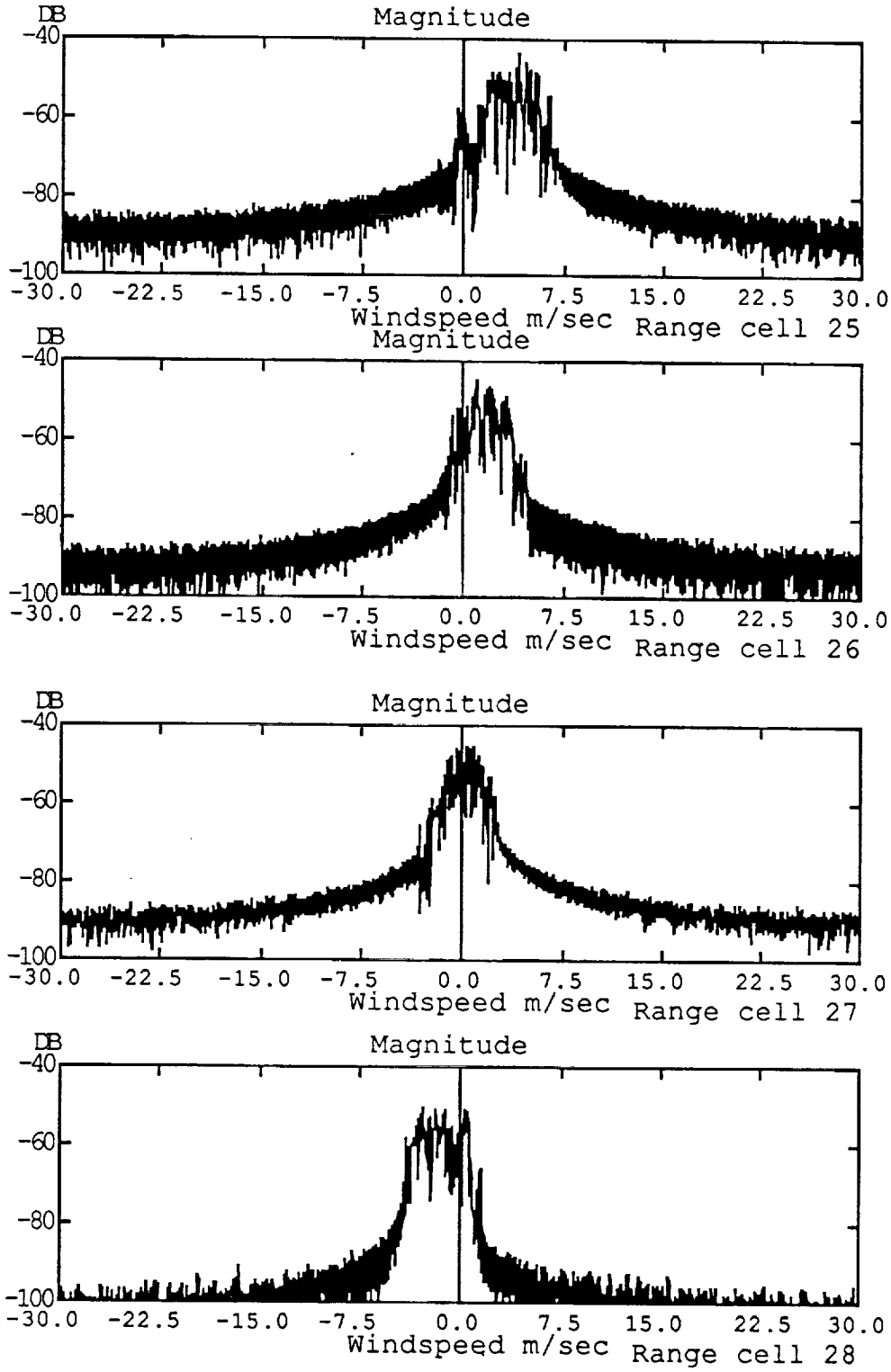

DB
Magnitude
-70
-80
-90
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 5
DB
Magnitude
-70
-80
-90
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 6
DB
Magnitude
-85
-90
-95
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 7
DB
Magnitude
-85
-90
-95
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 8

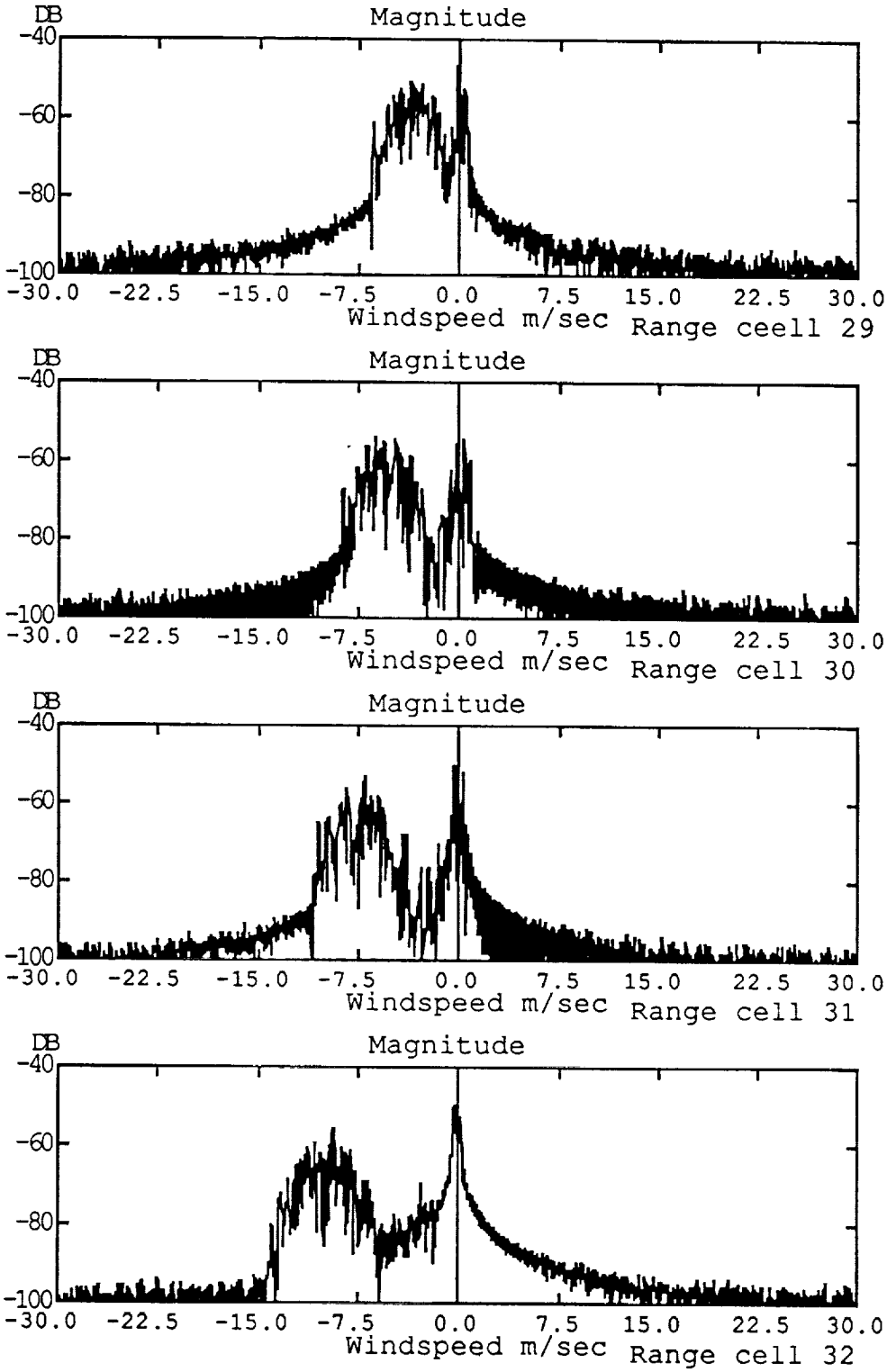

DB
Magnitude
-70
-80
-90
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 9
DB
Magnitude
-70
-80
-90
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 10
DB
Magnitude
-40
-60
-80
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 11
DB
Magnitude
-40
-60
-80
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 12

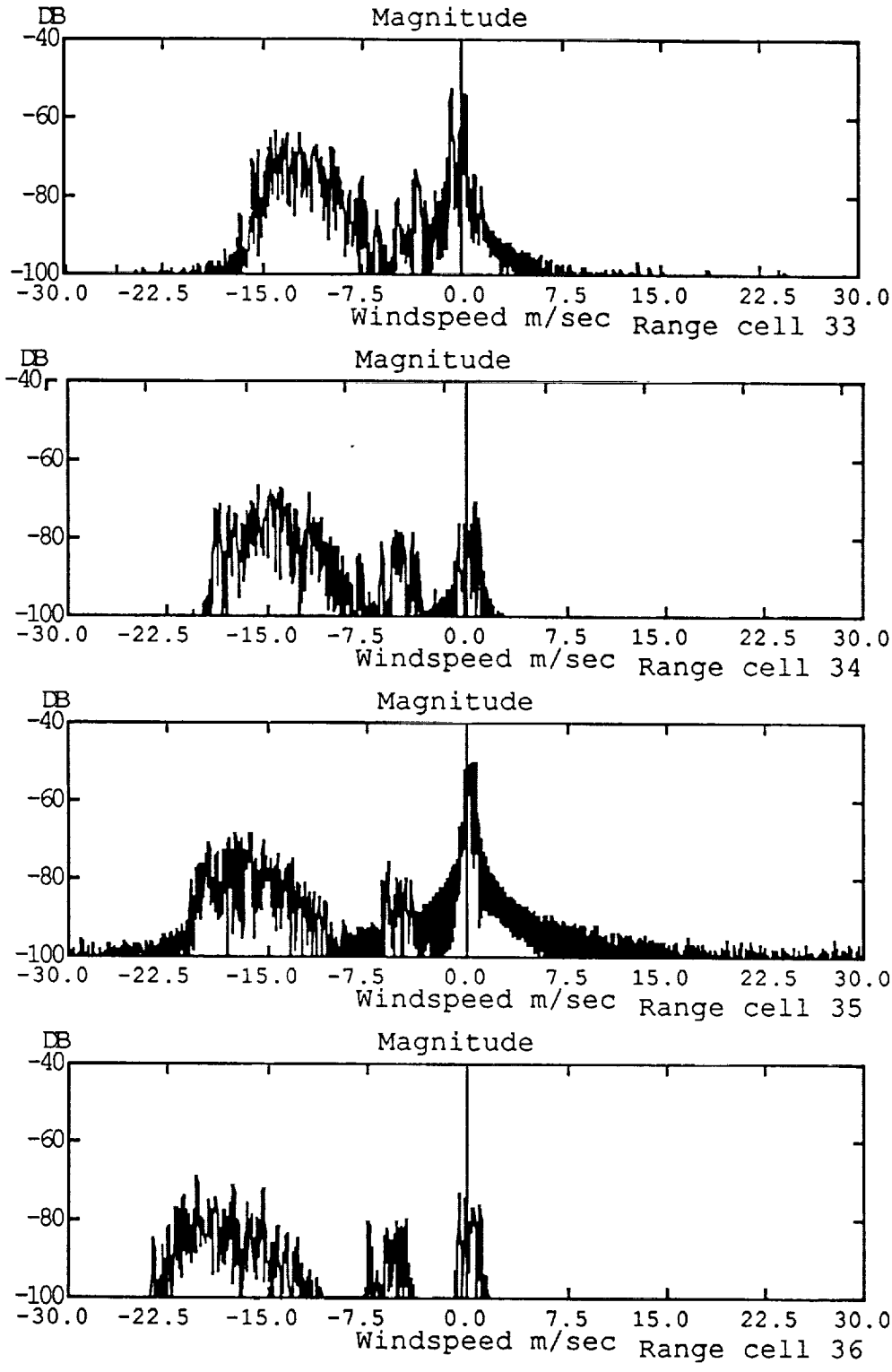

DB
Magnitude
-40
-60
-80
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 13
DB
Magnitude
-40
-60
-80
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 14
DB
Magnitude
-40
-60
-80
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 15
DB
Magnitude
-40
-60
-80
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 16

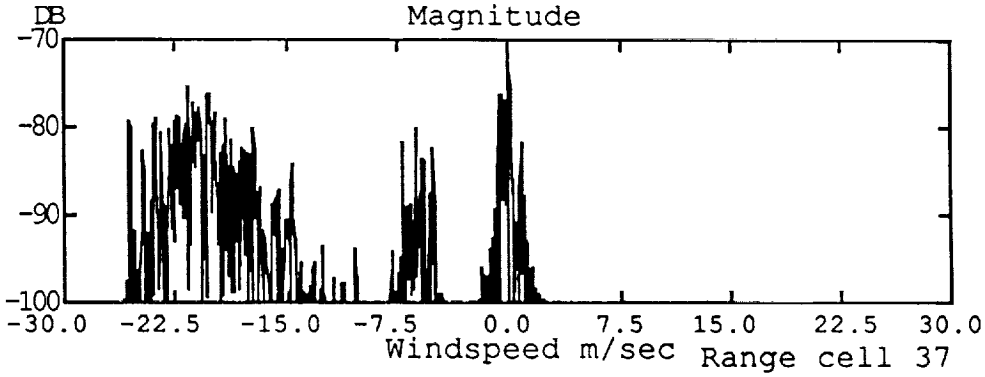

DB
Magnitude
-40
-60
-80
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 17
DB
Magnitude
-40
-60
-80
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 18
DB
Magnitude
-40
-60
-80
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 19
DB
Magnitude
-40
-60
-80
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 20

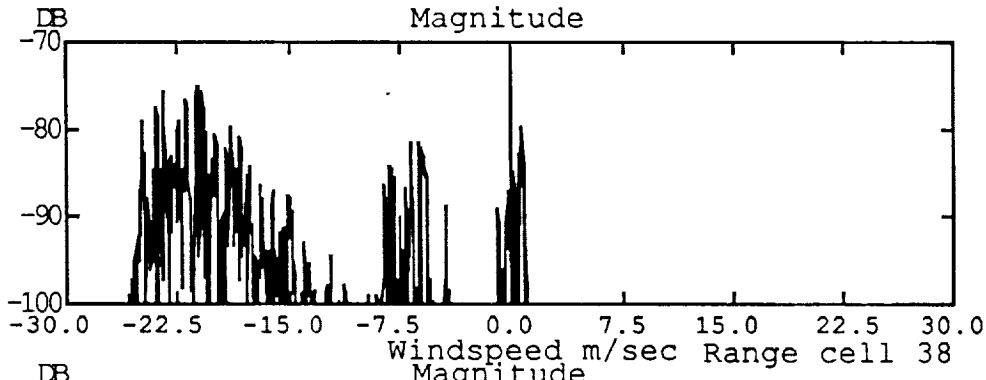

DB
Magnitude
-40
-60
-80
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 21
DB
Magnitude
-40
-60
-80
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 22
DB
Magnitude
-40
-60
-80
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 23
DB
Magnitude
-40
-60
-80
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 24

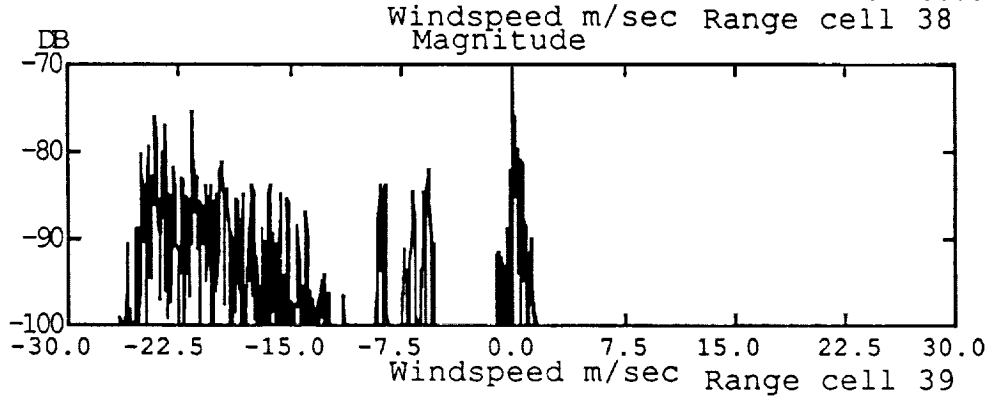

DB
Magnitude
-40
-60
-80
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec Range cell 25
DB
Magnitude
-40
-60
-80
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec Range cell 26
DB
Magnitude
-40
-60
-80
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec Range cell 27
DB
Magnitude
-40
-60
-80
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec Range cell 28

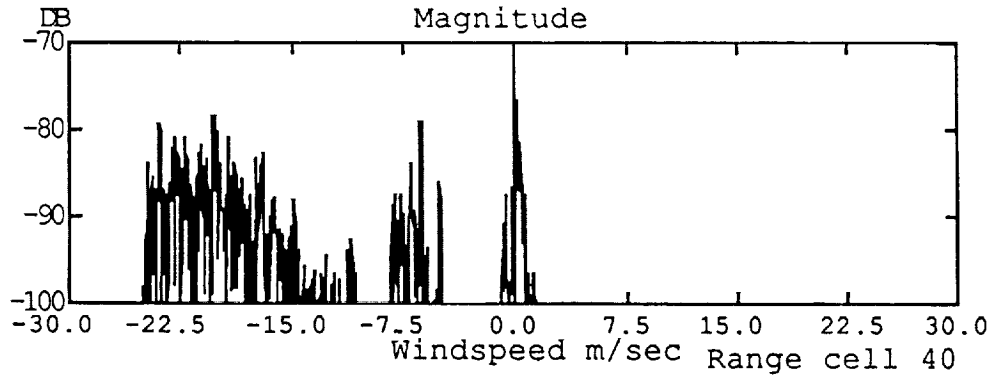

DB
Magnitude
-40
-60
-80
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range ceell 29
DB
Magnitude
-40
-60
-80
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 30
DB
Magnitude
-40
-60
-80
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 31
DB
Magnitude
-40
-60
-80
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 32


DB
Magnitude
-40
-60
-80
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 33
DB
Magnitude
-40
-60
-80
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 34
DB
Magnitude
-40
-60
-80
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 35
DB
Magnitude
-40
-60
-80
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec
Range cell 36


DB
Magnitude
-70
-80
-90
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec Range cell 37


DB
Magnitude
-70
-80
-90
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec Range cell 38


DB
Magnitude
-70
-80
-90
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec Range cell 39


DB
Magnitude
-70
-80
-90
-100
-30.0 -22.5 -15.0 -7.5 0.0 7.5 15.0 22.5 30.0
Windspeed m/sec Range cell 40

REFERENCES CITED

1. Bracalente, E. M. and Britt, C. L, "*Airbourne Doppler Radar Detection of Low Altitude Windshear,*" paper presented at the AIAA Conference on Sensor and Measurement Techniques for Aeronautical Application, Atlanta, GA; September 7-9, 1988.

2. Doviak, R. J. and Zrnic, D. S., *Doppler Radar and Weather Observations*, Orlando, FL; Academic Press, 1984.

3. Brookner, E., *Radar Technology*, Norwood, MA; Artech House, 1977.

4. Evans, J. E., "*Ground Clutter Cancellation for the NEXRAD System,*" report available through National Technical Information Service, Springfield, VA; 1983.

5. Sitterle, J. J. and Baxa, E. G., "*Estimation of Spectral Moments in Small Wavalength Doppler Radar,*" International Conference on Acoustics, Speech, and Signal Processing, *IEEE Proceedings*, San Diego, CA; 1984.

6. Groginsky, H. L., "*Pulse Pair Estimation of Doppler Spectrum Parameters,*" Preprints 15th Radar Meteorology Conference, American Meteorological Society, Boston, MA; 1972.

7. Skolnick, M. I., *Introduction to Radar Systems*, New York, NY; McGraw-Hill, 1980.

8. Ward, H. R., "*Doppler Processor Rejection of Range Ambiguous Clutter,*" *IEEE Transactions on Aerospace and Electronic Systems;* 1975.

9. Carlsson, S., "*MTI-Filtering for Multiple Time Around Clutter Suppression in Coherent on Receive Radars,*" *Advances in Radar Techniques*, London, UK; Peter Peregrinus Ltd.,1985.

10. Doviak, R. D., Zrnic, D. S., and Sirmans, D. S., "*Doppler Weather Radar,*" *IEEE Proceedings*, San Diego, CA; 1979.

11. Groginsky, H. L. and Glover, K., "*Weather Radar Cancellor Design,*" Preprints 19th Radar Meteorology Conference, American Meteorological Society, Boston, MA; 1980.

12. Bernella, D. M., "*Weather Radar Studies,*" document available through National Technical Information Service, Springfield, VA; 1986.

13. Noviak, L. R. and Glover K. M., "*Spectral Mean and Variance Estimation via Pulse Pair Processing,*" Preprints 16th Radar Meteorology Conference, American Meteorological Society, Boston, MA; 1975.

14. Ludeman, L. C., *Fundamentals of Digital Signal Processing,* Philadelphia, PA; Harper and Row, 1986.

15. Mendenhall, W. and Sinch T., Statistics for the Engineering and Computer Sciences, San Francisco, CA; Dellen Publishing, 1984.

16. Crone, W. R., *Programs for Digital Signal Processing,* New York, NY; IEEE Press, 1979.

17. Oppenheim, A. V. and Schafer, R. W., *Digital Signal Processing,* Englewood Cliffs, NJ; Prentice-Hall, 1975.