Illinois State Water Survey Division

CLIMATE & METEOROLOGY SECTION

SWS Contract Report 472.

A STUDY OF GROUND CLUTTER SUPPRESSION AT THE CHILL DOPPLER WEATHER RADAR

by James Tom Peltier

Prepared with the support of
National Science Foundation Grant **ATM** 83-20095

Champaign, Illinois
September 1989

Illinois Department ot Energy and Natural Resources

A STUDY OF GROUND CLUTTER SUPPRESSION AT THE CHILL DOPPLER WEATHER RADAR

by
James Tom Peltier

B.S., State University of New York, 1988

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 1989

Illinois State Water Survey
2204 Griffith Drive
Champaign, Illinois 61820-7495

DEDICATION

To my mother and father,
for their love and devotion.

ACKNOWLEDGMENTS

The author wishes to thank Dr. D. C. Munson and Dr. E. A. Mueller for their support and guidance throughout this project. Thanks must also go to D. A. Brunkow for his assistance in software development and data collection, and to J. D. Nespor for all the discussions about radar and the mountains. This work was supported by the National Science Foundation through the cooperative agreement ATM-8320095.

CONTENTS

1 INTRODUCTION 1

2 PROCESSING WEATHER RETURNS FROM A MONOSTATIC PULSE DOPPLER RADAR 5

2.1 Fundamentals of Monostatic Pulse Doppler Radar 5

2.2 The Baseband Weather Echo Signal 14

2.3 The Pulse-Pair Algorithm 17

2.4 FFT Processing and the Pulse-Pair Algorithm 24

2.5 Estimator Biases from Quadrature Gain Imbalance 26

3 GROUND CLUTTER 33

3.1 The Ground Clutter Problem 33

3.2 Ground Clutter Models 49

3.3 Ground Clutter Variation 53

4 GROUND CLUTTER FILTERS 67

4.1 Linear Filters as a Solution to Ground Clutter 67

4.2 Design of Elliptic Recursive Filters 72

4.3 Performance of Recursive Elliptic Filters 80

4.4 Filter Initialization 111

5 CONCLUSIONS 125

REFERENCES 129

APPENDIX I GENERAL CHARACTERISTICS OF THE CHILL RADAR SYSTEM 133

APPENDIX II SPECIFICS OF THE CHILL SIGNAL FLOW . . . 135

APPENDIX III FOUR-POLE ELLIPTIC FILTER PROGRAM LISTING 140

APPENDIX IV FILTER COEFFICIENT GENERATION PROGRAM LISTING 167

LIST OF TABLES

3.1. Altitudes of some lower troposphere weather phenomena 33

4.1. Computations needed to process a single complex sample through cascade realizations of elliptic filters 73

4.2. Coefficients for three-pole analog prototypes. ... 74

4.3. Coefficients for four-pole analog prototypes 74

LIST OF FIGURES

2.1. Coherent quadrature demodulator 7

2.2. Bias of mean velocity estimator (2.44) due to quadrature gain imbalance 29

2.3. Bias of first variance estimator (2.53) due to quadrature gain imbalance 31

2.4. Bias of second variance estimator (2.54) due to quadrature gain imbalance 32

3.1. Ray propagation paths for the four-thirds earth's radius model 35

3.2. Multi-trip power losses as a function of normalized range for constant target reflectivities 38

3.3. (a) Power spectral density of the echo from a cell filled with a rainshower around a radio tower. (b) Autocorrelation of the same echo. .. 40

3.4. (a) Power spectral density of same echo as in Fig. 3.3 but with the ground clutter removed. (b) Autocorrelation of the echo from (a) 41

3.5. Velocity estimator (3.8a) bias due to the presence of clutter. (a) Wide weather spectrum and narrow clutter spectrum. (b) Narrow weather spectrum and wide clutter spectrum 45

3.6. First variance estimator (3.8b) bias due to the presence of clutter. (a) Wide weather spectrum and narrow clutter spectrum. (b) Narrow weather spectrum and wide clutter spectrum 46

3.7. Second variance estimator (3.8c) bias due to the presence of clutter. (a) Wide weather spectrum and narrow clutter spectrum. (b) Narrow weather spectrum and wide clutter spectrum 47

3.8. CHILL one-way antenna pattern 50

3.9. Cumulative clutter power distributions for cells out to 40 km. (a) Cumulative less than plot. (b) Cumulative more than plot 54

3.10. Average ground clutter power as a function of range for several elevation angles 55

3.11. Comparisons of clutter power distributions for different ground conditions. (a) The presence of vegetation in late summer shows a dramatic increase in clutter power. (b) Wet ground on July 19 causes little increase in clutter power over dry ground 'on July 20 57

3.12. Ground-clutter spectral-width cumulative distribution for returns from the WAND radio tower 59

3.13. Ground-clutter spectral-width cumulative distribution for returns from the Champaign-Urbana metro area 60

3.14. Ground-clutter spectral-width cumulative distribution for returns from the banks of the Sangamon river 61

3.15. Ground-clutter spectral-width cumulative distribution for returns from cultivated farmland 62

3.16. Effects of antenna scanning on ground clutter width 66

4.1. Frequency responses for first- and second-order pulse cancellers. 71

4.2(a). Cascade realization of the three-pole elliptic filter (4.9a) 77

4.2(b). Cascade realization of the four-pole elliptic filter (4.9b) 77

4.3. (a) Three-pole elliptic filter fitting the transfer function (4.9a). (b) Four-pole elliptic filter fitting the transfer function (4.9b). .. 78

4.4. Expanded view of transition bands of the filters in Fig. 4.3 79

4.5. Ratio of filter notch width over clutter spectral width for different clutter suppression levels 81

4.6. Discrete model clutter residue for slow antenna rotation rate through (a) Three-pole filter, (b) Four-pole filter 83

4.7. Discrete model clutter residue for fast antenna rotation rate through (a) Three-pole filter, (b) Four-pole filter 84

4.8(a). Power of clutter remaining after the discrete clutter model passes through a four-pole filter with a passband of 0.04 86

4.8(b). Same as Fig. 4.8(a) except passband is 0.08. .. 87

4.8(c). Same as Fig. 4.8(a) except passband is 0.12. .. 88

4.8(d). Same as Fig. 4.8(a) except passband is 0.16. .. 89

4.9(a). Same as Fig. 4.8(b) except stopband gain is -30 dB 90

4.9(b). Same as Fig. 4.8(b) except stopband gain is -40 dB 91

4.9(c). Same as Fig. 4.8(b) except stopband gain is -50 dB 92

4.9(d). Same as Fig. 4.8(b) except stopband gain is -60 dB 93

4.9(e). Same as Fig. 4.8(b) except stopband gain is -70 dB 94

4.10. Discrete-model clutter spectral densities for (a) slow antenna rotation rate, (b) fast antenna rotation rate . . . 96

4.11(a). Comparisons of reflectivity biases introduced by filtering a weather signal by three different filters . . . 97

4.11(b). Same as Fig. 4.11(a) but velocity biases . . . 98

4.11(c). Same as Fig. 4.11(a) but spectral width biases . . . 99

4.12(a). Reflectivity biases introduced by filtering a weather signal by a four-pole filter having a -50 dB stopband gain . . . 100

4.12(b). Same as Fig. 4.12(a) but velocity biases . . . 101

4.12(c). Same as Fig. 4.12(a) but spectral width biases . . . 102

4.13(a). Reflectivity biases introduced by filtering a weather signal of various spectral widths . . . 103

4.13(b). Same as Fig. 4.13(a) but velocity biases . . . 104

4.13(c). Same as Fig. 4.13(a) but spectral width biases . . . 105

4.14(a). Same as Fig. 4.13(a) except filter width is 0.08 . . . 106

4.14(b). Same as Fig. 4.13(b) except filter width is 0.08 . . . 107

4.14(c). Same as Fig. 4.13(c) except filter width is 0.08 . . . 108

4.15(a). Estimated mean weather velocities from the first block of data out of an uninitialized three-pole filter . . . 114

4.15(b). Same as Fig. 4.15(a) except a four-pole filter was used . . . 114

4.15(c). Same as Fig. 4.15(b) except the Hamming window initialization was employed . . . 115

4.15(d). Same as Fig. 4.15(b) except the one-pulse initialization was employed . . . 115

4.16(a). Same as Fig. 4.15(a) except reflectivity biases . . . 116

4.16(b). Same as Fig. 4.16(a) except a four-pole filter was used. . . . 116

4.16(c). Same as Fig. 4.16(b) except the Hamming window initialization was employed . . . 117

4.16(d). Same as Fig. 4.16(c) except the second block of data is used to form the estimates . . . 117

4.16(e). Same as Fig. 4.16(a) except the one-pulse initialization was employed . . . 118

4.16(f). Same as Fig. 4.16(b) except the one-pulse initialization was employed. . . . 118

4.17(a) Same as Fig. 4.16(a) except estimated weather widths . . . 119

4.17(b) Same as Fig. 4.17(a) except the second output data block was used to form estimates . . . 119

4.17(c) Same as Fig. 4.17(a) except a four-pole filter was used . . . 120

4.17(d) Same as Fig. 4.17(b) except a four-pole filter was used . . . 120

4.17(e) Same as Fig. 4.17(a) except Hamming window initialization was employed . . . 121

4.17(f) Same as Fig. 4.17(c) except Hamming window initialization was employed 121

4.17(g) Same as Fig. 4.17(e) except one-pulse initialization was employed 122

4.17(h) Same as Fig. 4.17(f) except one-pulse initialization was employed 122

AII.1. Block diagram of the CHILL's signal flow 136

AII.2. Block diagram of the CHILL's receiver 137

SYMBOLS

A_c	Carrier amplitude
A	Attenuation factor
c_a	Speed of light in air
I	In-phase signal
f_c	Carrier frequency
f_d	Doppler frequency shift
Q	Quadrature signal
r	Target range
$R(k)$	Autocorrelation of $V(k)$
$S(f)$	Fourier transform of $R(k)$
t_r	Range time
t_x	Arbitrary range time
T_s	Pulse repetition time
v_d	Doppler velocity shift
$V(k)$	Echo signal
Z	Reflectivity factor
$Z(f)$	Fourier transform of $V(k)$
α	Antenna rotation rate
$\beta(\theta)$	One-way antenna-power beam pattern
θ	Azimuth angle
$\theta_{1/2}$	Two-sided half-power beamwidth
λ	Electromagnetic wavelength
σ_f	Doppler frequency spectral width
σ_v	Doppler velocity spectral width
T	Pulse width
τ_s	Range sample time
ϕ	Elevation angle
Ψ_c	Carrier phase
Ψ'	Target-induced phase shift
Ψ	Total phase shift

CHAPTER 1

INTRODUCTION

The CHILL radar is an S-band pulse Doppler radar used primarily for meteorological research. The radar is a National Science Foundation facility currently located at Willard Airport in Savoy, Illinois. Appendix I gives an overview of the radar's hardware.

When using the radar to observe weather phenomena, three features of the weather echoes are usually desired. These are reflectivity, mean velocity and spectral width. Since weather echoes have such a large dynamic range and since such massive volumes of data must be processed in real time, sophisticated floating-point hardware and efficient parameter estimation are needed. Appendix II explains the CHILL's data system used to fulfill these requirements.

Ground clutter echoes from the ground and its associated obstacles (e.g., buildings and telephone poles) surrounding the radar may contaminate the spectral features of weather echoes. When the antenna scans at low elevation angles, the clutter echoes can be quite large, especially at close ranges. At the CHILL radar site ground echoes from nearby cities have been observed to be as much as 50 dB more powerful than those for some of the desired weather echoes. Fortunately, the spectral characteristics of ground clutter are such that its mean velocity is zero. Unfortunately, the power and spectral widths of ground clutter are highly

variable. The area the clutter is coming from, the season, the antenna rotation rate and other factors all influence the clutter's spectral width. In addition to some of the above factors, other factors such as the antenna's elevation angle and the distance between the radar and the clutter affect the intensity of the clutter echoes.

The ground clutter problem can be effectively dealt with by using short recursive dc notch elliptic filters with deep notches and narrow, but adjustable, notch widths. These filters have to be short so that they can be implemented in real time. Appendix III is a listing of a program for the CHILL's processor that implements a general form four-pole dc notch elliptic filter. Appendix IV contains a listing of a FORTRAN program that generates assembly language code for the program in Appendix III. This assembly language code creates banks of filter coefficients for filters with 1 dB passband ripple, stopband gains of -30 to -70 dB and passband widths of 0.01 to 0.25 pi. Using these coefficient banks, the program in Appendix III can choose different filters for different ranges.

The remainder of this thesis is divided into four chapters. Equations that are new or those that are unclear in the literature are all derived. Equations that are well documented are just stated with their sources cited. Some equations, such as those in the pulse-pair spectral parameter estimation algorithm, are rederived since they are considered to be crucial.

Chapter 2 deals with the general procedure of processing weather echoes from a monostatic pulse Doppler

radar. The first section in this chapter gives a brief background in monostatic pulse Doppler radar signals and demodulation. The end of this section contains the derivation of an equation that predicts the spectral broadening caused by antenna rotation. The next section describes the weather echo and its spectral characteristics. The third section derives the pulse-pair algorithm used to efficiently obtain estimates of the weather's spectral features. This section is followed by a brief comparison between the pulse-pair algorithm and FFT techniques. Chapter 2 closes with an analysis of pulse-pair estimator performance in the presence of a hardware anomaly that was discovered while analyzing data for this study.

Chapter 3 deals with the ground clutter problem. The first section shows the effects that ground clutter has on the pulse pair-parameter estimators. The second section discusses two ground clutter models that are fairly accurate and simple to use. Using real data collected at the CHILL, the last section in Chapter 3 shows the effects that various factors have on ground clutter characteristics.

Chapter 4 deals with ground clutter filters. The first section discusses previous methods used to deal with the ground clutter problem and shows why short recursive filters are a good solution for the CHILL. The next section derives the design equations for three- and four-pole elliptic filters, which have been implemented on the CHILL. This is followed by a section that addresses the use and effectiveness that these filters have in improving weather parameter estimates in the presence of ground clutter.

Chapter 4 closes with an examination of two filter initialization procedures that attempt to suppress the filter's transient responses. Chapter 5 summarizes the conclusions to be drawn from this study and outlines several areas that need further development.

CHAPTER 2

PROCESSING WEATHER RETURNS FROM A MONOSTATIC PULSE DOPPLER RADAR

2.1 Fundamentals of Monostatic Pulse Doppler Radar

Weather radars are typically monostatic, that is, they use a common antenr.a for both transmission and reception. A monostatic pulse Doppler system works as follows. First, a short electromagnetic pulse leaves the antenna and travels forward until it encounters a scatterer. When the wave hits the scatterer, it is reflected off it in many directions. The portion of the wave that reflects backwards travels until it is received back at the antenna. This received wave has been modulated with information about the scatterer and this information can be extracted by using signal processing techniques. Before going into these techniques, an understanding of transmitted and received signals is needed.

A simple transmit waveform is the gated sinusoid. Its phasor representation is

$$T(t) = A_c P_\tau(t) \exp[j(2\pi f_c t + \Psi_c)], \qquad (2.1)$$

where A_c, f_c and Ψ_c are the transmit amplitude, frequency and phase, respectively. The real part of $T(t)$ is actually transmitted. $P_\tau(t)$ is a pulse function equal to unity when t is between zero and τ, and it is zero everywhere else. The received waveform from a discrete point target has the

phasor representation [3]

$$R(t) = A_c A P_r(t-t_r)\exp[j(2\pi(f_c+f_d)(t-t_r)+\Psi_c+\Psi')], \quad (2.2)$$

where A is an attenuation factor dependent upon the target's backscatter cross section, its distance from the antenna, and its location in the antenna's two-way electric-field pattern. The time it takes for the wave to propagate from the antenna to the target and back again is t_r. Upon scattering, the target introduces the random phase Ψ'. The target's radial motion, towards or away from the antenna, causes the Doppler frequency shift f_d. Coherent quadrature demodulation, as in Fig. 2.1, can be used to strip off the carrier in the received signal (2.2). The resulting baseband signal is

$$V(t) = I(t)+jQ(t) = AP_r(t-t_r)\exp[j(2\pi f_d t+\Psi)], \quad (2.3)$$

where $\Psi = \Psi'+2\pi(f_c+f_d)t_r$ is introduced for brevity. The I and Q in this equation denote the in-phase and quadrature signals, respectively.

The demodulated signal (2.3) contains two important pieces of information: the target's distance from the radar and its radial velocity. Since electromagnetic waves travel at the speed of light, and the time it takes for waves to travel from the radar to the target and back again is t_r, the distance between the target and radar is

$$r = c_a t_r/2, \quad (2.4)$$

where c_a is the speed of light in air ($2.997\text{x}10^8$ m/s). If

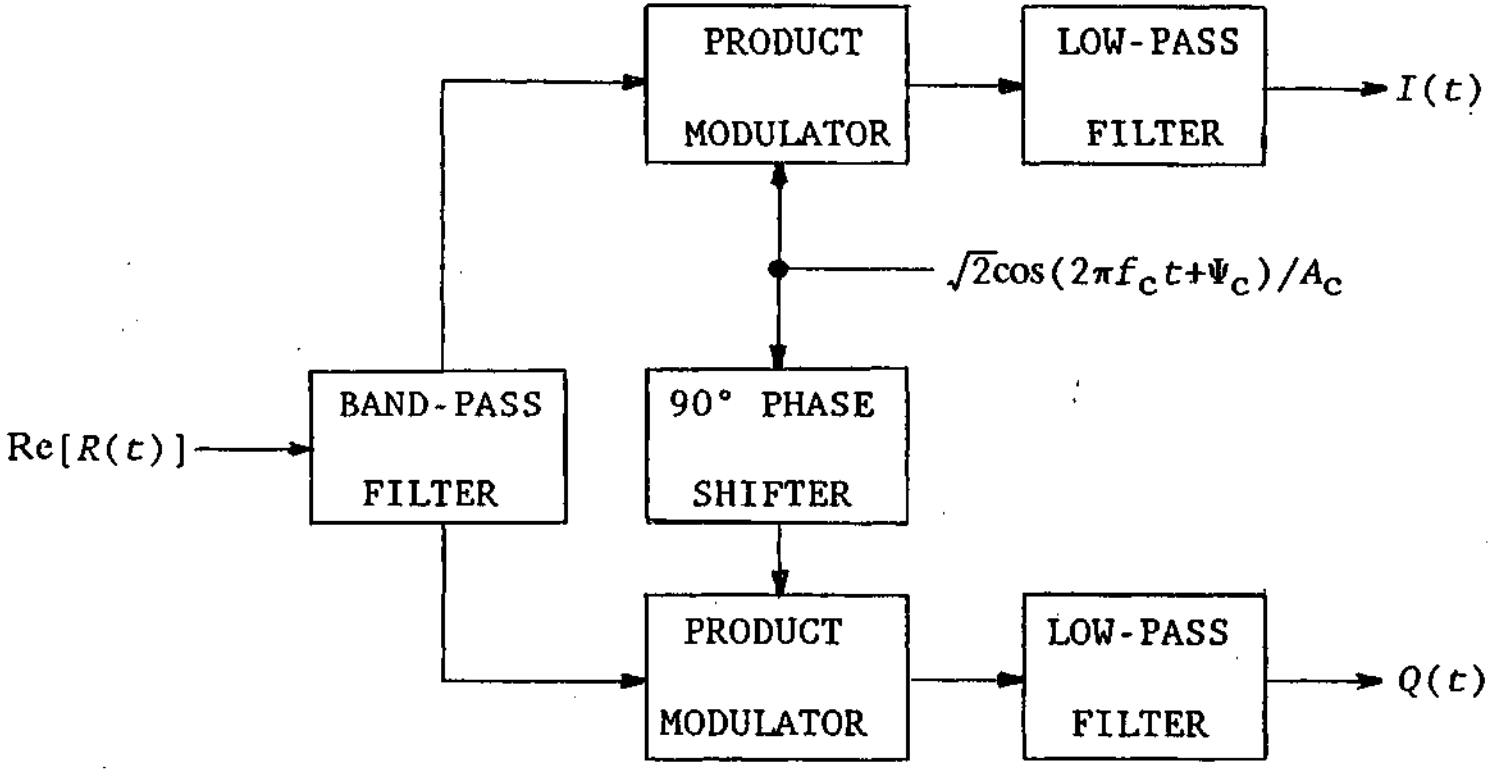


Figure 2.1. Coherent quadrature demodulator.

the target is stationary, electromagnetic waves bounce off it with unaltered frequency. If the target has a radial velocity v_d towards (away from) the radar, then it sees the electric and magnetic fields of the radar's waves fluctuating faster (slower) than the wave's transmitted frequency. This rate increase (decrease) is v_d/λ, where λ is the electromagnetic wave's wavelength. Since the target sees the waves at an altered frequency, it reflects them at this altered frequency. If a new coordinate system is defined with the target at the origin, the receiver then has velocity v_d relative to the target. The receiver now sees the electromagnetic waves just as the target saw them above. Thus a moving target causes a frequency shift in the received signal of

$$f_d = -2v_d/\lambda, \tag{2.5}$$

where velocity is positive going away from the radar.

Weather targets are composed of many individual hydrometeors, which are particles such as raindrops, snowflakes, or hail. Since all of these particles are small compared to the transmitted wave's wavelength ($\lambda \approx 10$ cm at the CHILL), they tend to scatter the electromagnetic waves isotropically. This means weather targets can be modeled as collections of point targets. By transmitting the wave (2.1) and using the coherent quadrature demodulator of Fig. 2.1, the baseband signal resulting from a distributed target is

$$V(t) = \sum_i A_i P_r(t-t_{r,i}) \exp[-j(2\pi f_{d,i} t + \Psi_i)], \tag{2.6}$$

where the subscript i refers to the i^{th} individual point target. Since reception can occur only after transmission, this equation is true only for $t > \tau$.

Consider the echo signal (2.6) at a specific time t_x after transmission. The spatial area of the point targets contributing to this signal defines a resolution volume. This volume can best be described in a spherical coordinate system. With the radar at the origin, let r denote the range, θ denote the azimuth angle, and ϕ denote the elevation angle. Assume N point targets exist. Let them be numbered such that $t_{r,0} \leq t_{r,1} \leq t_{r,2} \leq \ldots \leq t_{r,N-1}$. Now (2.6) becomes

$$V(t_x) = \sum_{i=n}^{m} A_i \exp[-j(2\pi f_{d,i} t_x + \Psi_i)], \qquad (2.7)$$

where $t_x - \tau \leq t_n \leq t_m \leq t_x$. Using (2.4), the range extents of the resolution volume are $c_a(t_x - \tau)/2 \leq r \leq c_a t_x/2$. Although the resolution volume is $c_a\tau/2$ deep, targets in this band do not contribute equally. The classic radar equation (e.g.,[2];pp.6-8) reveals a dependence of $1/r^4$ on the received power from particles of equal cross section. This means the amplitude factors A_i in the echo signal (2.7) have a $1/r^2$ dependence. The resolution volume's azimuth θ and elevation ϕ extents are determined by the spatial weighting of the two-way antenna beam pattern.

Another important concept in dealing with distributed targets is that of velocity distributions. Each point target contributes its own Doppler shifted frequency in the echo

signal (2.7). An energy density function of these Doppler shifted frequencies can be formed by weighting each target's Doppler shifted frequency by the square of the target's attenuation factor A_i. This frequency density can be converted to a velocity density using (2.5).

The analysis so far has dealt only with a single pulse. In practice, many pulses are transmitted using the signal

$$T(t) = \sum_n A_c P_\tau(t-nT_s)\exp[j(2\pi f_c t+\Psi_c)], \qquad (2.8)$$

where T_s is the pulse repetition time and n is an integer. Using the same demodulation as before, the resulting baseband signal is

$$V(t) = \sum_n \sum_i A_{i,n} P_\tau(t-nT_s-t_{r,i,n})\exp[-j(2\pi f_{d,i,n} t+\Psi_{i,n})], \qquad (2.9)$$

where the two subscripts on A, t_r, f_d and Ψ reflect each target's changing values from pulse to pulse. The summation over n in this equation leads to the problem of range ambiguity. Targets farther than $c_a T_s/2$ from the radar have echoes that arrive at the receiver after subsequent pulse transmissions. The $1/r^2$ dependence of the target's A_i factors will usually attenuate these far targets to negligible levels. Thus, this is really a problem only when the far targets are much stronger than the closer ones.

Let $\tau \le t_x \le T_s$ and let k represent an integer. The sampled echo signal $V(t_x+kT_s)$ defines a discrete-time random

process of echoes from a fixed resolution volume. Since the sampling rate of this signal is $1/T_s$, Doppler frequencies greater than $1/2T_S$, or less than $-1/2T_S$, will alias into this interval. Between samples, some point targets will enter the resolution volume while others leave. The rest just move around within it. These movements along with the diversity of the individual target's Doppler frequencies show the turbulence of the contents in the resolution volume.

To facilitate digital processing, the echo signal $V(t)$ is sampled many times per pulse. The echo signal can then be viewed as the discrete time signal

$$V(m,k) = V(t_x - m\tau_s + kT_s), \qquad (2.10)$$

where T_S is the shorter range sampling time and m is an integer from zero to M-1. The number of samples per pulse M must be less than or equal to $((T_s - t_x)/\tau_s) + 1$. Each range sample m defines a resolution volume at the range $c_a(m\tau_s + t_x)/2$. There are M of these resolution volumes, also called range bins, which overlap if $\tau_s < \tau/2$.

In practice, antennas are not stationary. They scan by rotating on their axes. A scanning antenna will change range bins into concentric range rings. Resolution volumes are no longer constant, causing slightly different targets to be seen from pulse to pulse. If the antenna is scanning at a rate a, between n pulses the center of the beam moves $\alpha n T_s r_o \cos[\phi_e]$, where ϕ_e is the antenna elevation angle. If the two-sided half-power antenna beamwidth is denoted by

$\theta_{1/2}$ the antenna pattern is $r_o\theta_{1/2}$ wide at the range r_o. Taking the ratio of these two distances gives the approximate change in resolution volume between n pulses as

$$\alpha n T_s \cos[\phi_e]/\theta_{1/2} \times 100\%. \qquad (2.11)$$

The time it takes to cause a 100% change in the resolution volume can be thought of as a dwell time. This is the maximum time a stationary target will be illuminated by the beam's main lobe during each scan. As the antenna scans faster, the subsequent echoes from the same range ring become less correlated. This causes the range ring's power spectrum to be broadened. A range ring's power spectrum is defined as the square of the magnitude of the Fourier transform of its baseband echoes.

A method has been used to quantify this broadening effect by looking at the return from a stationary point target ([2];pp.478-480,[14]). If the antenna is also stationary, then the return from the point target should be constant from pulse to pulse. Squaring the Fourier transform of this constant return gives a power spectrum of a line at zero frequency. Now assume the antenna has the normalized one-way power beam pattern that is Gaussian in azimuth, given by

$$\beta(\theta) = \exp[-(\theta\cos[\phi_e]/\sigma_\theta)^2/2]. \qquad (2.12)$$

Here again, the $\cos[\phi_e]$ term compensates for the antenna elevation as in (2.11). The σ_Θ factor in this pattern can be expressed in terms of the half-power beamwidth $\theta_{1/2}$ as

$$\sigma_\theta^2 = \theta_{1/2}^2/8\ln[2]. \qquad (2.13)$$

If the antenna rotates at a constant rate α, the echo signal from the point target is

$$\begin{aligned} V(k) &= V(t_r+kT_s) = A\beta(\alpha kT_s) \\ &= A\exp[-\ln[2](2\alpha\cos[\phi_e]kT_s/\theta_{1/2})^2], \end{aligned} \qquad (2.14)$$

where t_r is the target's range time. The echo signal's spectrum is the discrete-time Fourier transform of the signal:

$$\begin{aligned} Z(f) &= \mathrm{DTFT}[V(k)] \\ &= \frac{2A\pi\alpha\cos[\phi_e]\ln[2]^{1/2}}{\theta_{1/2}T_s} \sum_n \exp\left[\frac{-(\pi\theta_{1/2}(f-n/T_s))^2}{\ln[2](2\alpha\cos[\phi_e])^2}\right]. \end{aligned} \qquad (2.15)$$

The spectrum is computed based on many pulse returns, with one sample per pulse taken at the same relative position every pulse. By processing this way, the spectrum is composed of samples from the same resolution volume. If aliasing is assumed negligible, the spectrum is

$$\begin{aligned} Z(f) = \frac{2A\pi\alpha\cos[\phi_e]\ln[2]^{1/2}}{\theta_{1/2}T_s} \sum_n \exp\left[\frac{-(\pi\theta_{1/2}(f-n/T_s))^2}{\ln[2](2\alpha\cos[\phi_e])^2}\right] \\ \times [U[f+1/2T_s-n/T_s]-U[f-1/2T_s-n/T_s]], \end{aligned} \qquad (2.16)$$

where $U[f]$ is a unit step function. Therefore, the power spectrum is

$$S(f) = Z(f)Z^*(f)$$

$$= \ln[2]\frac{(2A\pi\alpha\cos[\phi_e])^2}{(\theta_{1/2}T_s)^2}\sum_n \exp\left[\frac{-(\pi\theta_{1/2}(f-n/T_s))^2}{2\ln[2](\alpha\cos[\phi_e])^2}\right]$$

$$\times [U[f+1/2T_s-n/T_s]-U[f-1/2T_s-n/T_s]]. \qquad (2.17)$$

If the spectrum's width is defined to correspond to the root mean square of the Gaussian function, then

$$\sigma_f = \alpha\cos[\phi_e]\ln[2]^{1/2}/\pi\theta_{1/2}. \qquad (2.18)$$

Using (2.5) to convert this to a velocity value gives

$$\sigma_v = \alpha\lambda\cos[\phi_e]\ln[2]^{1/2}/2\pi\theta_{1/2}. \qquad (2.19)$$

This is the same result achieved by Doviak and Zrnic with a more complex derivation ([4];pp.445-447).

2.2 The Baseband Weather Echo Signal

As previously mentioned, the demodulated weather echo signal can be viewed as the complex random process

$$V(m,k)=V(t_x-m\tau_s+kT_s)=I(t_x-m\tau_s+kT_s)+jQ(t_x-m\tau_s+kT_s). \qquad (2.20)$$

An individual in-phase or quadrature sample is a random variable. With the assumption that the resolution volume is filled with many independent hydrometeors, where no hydrometeors dominate, the central limit theorem can be used to show that the random variables I and Q are Gaussian distributed with zero mean ([4];pp.49-51). These random variables are also independent for any fixed m and k in

(2.20), giving the joint probability density function

$$p[I,Q] = (1/2\pi\sigma^2)\exp[-(I^2+Q^2)/2\sigma^2], \qquad (2.21)$$

where σ^2 is the mean-square value of I and Q. The random variables of echo amplitude, $|v| = (I^2+Q^2)^{1/2}$, and phase, $\theta = \arctan(Q/I)$, can be shown to have Rayleigh and uniform probability density functions respectively [16], i.e.,

$$p[|V|] = (|V|/2\pi\sigma^2)\exp[-(|V|^2)/2\sigma^2] \qquad (2.22)$$

and

$$p[\theta] = (\theta/2\pi)[U[\theta]-U[\theta-2\pi]]. \qquad (2.23)$$

The power, $P = C(I^2+Q^2)$, has the exponential density

$$p[P] = (1/2\pi\sigma^2)\exp[-P/2\sigma^2]U[P]. \qquad (2.24)$$

For the rest of this chapter the range index m of the random process (2.20) will be fixed leaving only the pulse index. The resulting process has the autocorrelation

$$R(m,n) = E[V^*(m)V(n)] = E[V^*(t_r+mT_s)V(t_r+nT_s)], \qquad (2.25)$$

where n > m and E[-] denote statistical expectation. The time intervals considered in weather signals are usually small. Pulse repetition times are in the millisecond range and total dwell times are much less than a second. The time that it takes for the statistical properties of weather echoes to change is on the order of several seconds. Under these time scales, the weather echo $V(k)$ can be considered a wide sense stationary random process. Its autocorrelation is then

$$R(n-m) = E[V^*(m)V(n)]. \qquad (2.26)$$

Letting $k = n-m$, (2.26) becomes

$$R(k) = E[V^*(m)V(m+k)] = E[V^*(n-k)V(n)]. \qquad (2.27)$$

By letting $k = -k$ in (2.27), note that

$$R(-k) = R^*(k). \qquad (2.28)$$

Weather echoes are usually considered ergodic. This means that their autocorrelations can be found by time averaging:

$$R(k) = \lim_{N \to \infty} (1/N) \sum_{n=-N/2}^{N/2} V^*(n)V(n+k). \qquad (2.29)$$

The power spectral density is the Fourier transform of its autocorrelation. Let $V^{\sim}(n)$ denote $V(-n)$ and also let $m = -n$. Equation (2.29) now becomes

$$R(k) = \lim_{N \to \infty} (1/N) \sum_{m=-N/2}^{N/2} V^{\sim *}(m)V(k-m). \qquad (2.30)$$

Recognizing this as a convolution, its discrete-time Fourier transform is the product of each function's discrete-time Fourier transform,

$$S(f) = \mathrm{DTFT}[R(k)] = \mathrm{DTFT}[V^{\sim *}(k)]\,\mathrm{DTFT}[V^*(k)] = Z^*(f)Z(f). \qquad (2.31)$$

Doviak and Zrnic have shown how using the central limit theorem, spectral densities of weather echo autocorrelations usually have a Gaussian shape ([4];pp.81-87).

There are three main parameters of meteorological interest in the weather echo signal. The first is the power of the returned signal. This is a measure of water content and rainfall rate in the resolution volume. The second parameter is mean Doppler velocity. This indicates the resolution volume content's average motion towards or away from the radar. When the radar is nearly horizontal, this is also a good measure of air movement. The third important parameter is the width of the returned signal's power spectral density. This indicates the variation in movement within the resolution volume. It points to things such as turbulence, shear and drop size distribution.

2.3 The Pulse-Pair Algorithm

The pulse-pair algorithm, also called autocorrelation processing, is a computationally efficient and accurate way to estimate the three spectral parameters mentioned at the end of the last section. Its name comes from the fact that lags of the autocorrelation function are estimated by using pairs of pulses. These pairs need not be contiguous. The algorithm was developed in 1968 at Bell Telephone Laboratories by W. D. Rummler. A series of three technical memos introduces the work [18]-[20]. Groginsky applied this method for use in weather radar [7], while Zrnic has done much work in analyzing its performance in a wide variety of weather situations [28],[29].

To understand this method, examine the autocorrelation

$$R(k) = \sum_{n} V^{*}(n)V(n+k). \qquad (2.32)$$

The power spectral density is the discrete-time Fourier transform of this autocorrelation given by

$$F(\theta) = \mathrm{DTFT}[R(k)] = \sum_{k} R(k)\exp[-j\theta k]. \qquad (2.33)$$

Notice that this signal is periodic with period 2π. Letting $\theta = 2\pi f T_s$, (2.33) becomes

$$S(f) = \sum_{k} R(k)\exp[-j2\pi f T_s k], \qquad (2.34)$$

which is now periodic with period $1/T_S$. The autocorrelation is recoverable by inverse discrete-time Fourier transforming (2.34). Thus,

$$R(k) = \int_{-1/2T_s}^{1/2T_s} S(f)\exp[j2\pi f T_s k]df. \qquad (2.35)$$

Using the symmetry property of (2.28), it is obvious that $S(f)$ must be real and positive.

The total power (mean-squared value) of a returned signal is the integration of the signal's power spectral density over one period. Using (2.35) and setting k to zero result in

$$R(0) = T_s \int_{-1/2T_s}^{1/2T_s} S(f)df = TOTAL\ POWER. \qquad (2.36)$$

Using a calibration unique to a specific radar, and after stripping off a range dependence, the power is converted to a meteorologically standardized parameter Z. The relationship between Z and power is linear. Z is a reflectivity factor that is usually expressed in a logarithmic scale as dBZ = 10log[Z]. Weather signals tend to have Z values between zero and 70 dBZ.

To find the returned signal's mean Doppler velocity, start by looking at the autocorrelation with a lag of one;

$$R(1) = T_s \int_{-1/2T_s}^{1/2T_s} S(f)\exp[j2\pi fT_s]df. \qquad (2.37)$$

Multiplying both sides by the unity factor,

$$\exp[j2\pi(f_d - f_d)T_s] = 1, \qquad (2.38)$$

gives

$$R(1) = T_s \int_{-1/2T_s}^{1/2T_s} \exp[j2\pi f_d T_s]S(f)\exp[j2\pi(f-f_d)T_s]df. \qquad (2.39)$$

Breaking apart (2.39) with Euler's formula leaves

$$R(1) = T_s \int_{-1/2T_s}^{1/2T_s} \exp[j2\pi f_d T_s] S(f) \cos[2\pi(f-f_d)T_s]\, df$$

$$+ T_s \int_{-1/2T_s}^{1/2T_s} \exp[j2\pi f_d T_s] S(f) \sin[2\pi(f-f_d)T_s]\, df. \qquad (2.40)$$

Notice that the cosine term is even-symmetric about the mean Doppler frequency f_d, while the sine term is odd-symmetric about it. If $S(f)$ is even symmetric about its mean frequency, the cosine integral in (2.40) integrates to a constant while the sine integral integrates to zero. For $S(f)$ even-symmetric about its mean frequency,

$$R(1) = T_s \int_{-1/2T_s}^{1/2T_s} \exp[j2\pi f_d T_s] S(f) \cos[2\pi(f-f_d)T_s]\, df. \qquad (2.41)$$

Taking the argument of both sides gives

$$\arg[R(1)] = 2\pi f_d T_s. \qquad (2.42)$$

Solving for f_d gives

$$f_d = \arg[R(1)]/2\pi T_s. \qquad (2.43)$$

Using (2.5), the mean Doppler velocity can be found as

$$v_d = -f_d\lambda/2 = -\lambda \arg[R(1)]/4\pi T_s . \qquad (2.44)$$

As long as the mean Doppler frequency is not aliased, aliasing in the rest of the spectrum will not bias the mean Doppler velocity estimator (2.44). When aliasing is present in the rest of the spectrum, the power spectral density and the sinusoid terms in (2.40) will keep their symmetries. The integration over the Nyquist frequency $1/T_S$ will still lead to the proper cancellation and the result of (2.44). If the aliasing is so bad as to have the mean Doppler frequency aliased, the mean Doppler velocity estimator (2.44) will still predict the correct velocity to within a multiple of the Nyquist velocity. Another nice feature of this estimator is that the presence of white noise will not affect its accuracy. This is due to the fact that white noise's flat spectrum is even-symmetric with respect to all frequencies.

In order to determine the spectral width estimator, it is convenient to assume a spectral shape. As mentioned in Section 2.2, weather echoes from regions of approximately uniform reflectivity have Gaussian spectra. Let the spectrum be given by

$$F(v) = \frac{S_o}{\sqrt{2\pi}\sigma_v} \exp\left[\frac{-(v-v_d)^2}{2\sigma_v^2}\right] . \qquad (2.45)$$

Using (2.5), (2.45) can be rewritten in terms of frequency as

$$S(f) = \frac{S_o}{\sqrt{2\pi}\sigma_v} \exp\left[\frac{-\lambda^2(f-f_d)^2}{8\sigma_v^2}\right] . \qquad (2.46)$$

The autocorrelation is once again the inverse discrete-time Fourier transform,

$$R(k) = \int_{-1/2T_s}^{1/2T_s} \frac{S_o T_s}{\sqrt{2\pi}\sigma_v} \exp\left[\frac{-\lambda^2 (f-f_d)^2}{8\sigma_v^2}\right] \exp[j2\pi f T_s k]\, df. \qquad (2.47)$$

Employing the shifting property of Fourier transforms, (2.47) becomes

$$R(k) = \int_{-1/2T_s}^{1/2T_s} \frac{S_o T_s}{\sqrt{2\pi}\sigma_v} \exp[j2\pi f_d k T_s] \exp\left[\frac{-\lambda^2 f^2}{8\sigma_v^2}\right] \exp[j2\pi f T_s k]\, df. \qquad (2.48)$$

Using the $\exp[-\pi t^2]$, $\exp[-\pi f^2]$ Fourier transform pair and the scaling property [15], the autocorrelation is

$$R(k) = S_o \exp[j2\pi f_d k T_s] \exp[-8(\pi\sigma_v k T_s/\lambda)^2]. \qquad (2.49)$$

Let m be a non-negative integer and n be a positive integer. Using (2.49), the ratio

$$R(m)/R(m+n) = \exp[-j2\pi f_d n T_s] \exp[8(\pi\sigma_v T_s/\lambda)^2 n(n+2m)]. \qquad (2.50)$$

Taking the natural logarithm of the magnitude of both sides of (2.50) leaves

$$\ln[\,|R(m)/R(m+n)|\,] = 8(\pi\sigma_v T_s/\lambda)^2 n(n+2m). \qquad (2.51)$$

Finally, solving for the variance gives

$$\sigma_v^2 = \lambda^2 \ln[|R(m)/R(m+n)|]/(8n(n+2m)\pi^2 T_s^2). \qquad (2.52)$$

The two most common width estimators are for $m = 0$, $n = 1$ and $m = 1$, n = 1. The corresponding estimators are

$$\sigma_{v1}^2 = \lambda^2 \ln[|R(0)/R(1)|]/(8\pi^2 T_s^2) \qquad (2.53)$$

and

$$\sigma_{v2}^2 = \lambda^2 \ln[|R(1)/R(2)|]/(24\pi^2 T_s^2). \qquad (2.54)$$

Since white noise has a constant spectrum, its autocorrelation is non-zero only at the zero lag. When white noise is present in a system, the total power found by (2.36) is actually signal and noise power. The first width estimator (2.53) should then be corrected for noise:

$$\sigma_{v1}^2 = \lambda^2 \ln[|(R(0)-N)/R(1)|]/(8\pi^2 T_s^2), \qquad (2.55)$$

where N is the total noise power. A problem exists in accurately estimating this noise power. One quite good method is a thresholding technique proposed by Hildebrand and Sekhon [11]. Although this technique is accurate, it is computationally intensive since it requires working in the frequency domain.

The performance of the pulse-pair estimators on actual weather echoes in the presence of noise is thoroughly examined by Srivastava [25]. The results are briefly restated here. For the sake of generality, velocities are normalized to the interval of negative one to one. The mean velocity estimator (2.44) gives errors of less than 0.02 for signal-to-noise power ratios more than about 5 dB. This

error, although possibly too large for some situations, represents a good level of performance. It corresponds to one hundredth of the total velocity interval. When examining the performances of the width estimators, the problem of noise estimation must be included. The first variance estimator, when compensated for noise (2.55) by the thresholding technique, performs about the same as the second variance estimator (2.54). They both have bias errors less than 0.003 when the signal-to-noise power ratios are greater than 10 dB. When the first variance estimator is uncompensated for noise (2.53), a signal-to-noise power ratio of almost 20 dB is needed for the same bias error of 0.003 to be achieved. The conclusion to be drawn from this is that unless a fairly accurate noise power estimate can be calculated more efficiently than the second lag of the autocorrelation (2.32), the second variance estimator (2.54) should be used.

2.4 FFT Processing and the Pulse-Pair Algorithm

Weather parameter estimation can be achieved in the spectral domain by using Fast Fourier Transform (FFT) techniques. To construct the discrete power spectral density of the returned signal, first the discrete signal spectrum is found by taking the FFT of the raw echo data. The discrete power spectrum (samples of $S(f)$) is then just the square of the magnitude of the signal spectrum (see (2.30)). Once the power spectrum is found, the three parameter

estimators are

$$P = \sum_{k=M/2}^{M/2} S(k) \quad , \tag{2.56}$$

$$v_d = -(\lambda/2MT_s)\left[k_m + (1/P)\sum_{k=k_m-M/2}^{k_m+M/2} (k-k_m)S(\mathrm{mod}_M[k])\right] \tag{2.57}$$

and

$$\sigma_v^2 = (\lambda^2/4PT_s^2)\sum_{k=k_m-M/2}^{k_m+M/2} (k/M+2v_dT_s/\lambda)^2 S(\mathrm{mod}_M[k]), \tag{2.58}$$

where the largest Fourier coefficient is $S(k_m)$ ([4];pp.94-113). These equations assume that a data block of length M is being processed.

In order to compare computational efficiencies of the two parameter estimation schemes, assume that both work on a block of data M long. The FFT requires $2M\log_2 M$ real multiplications. Another $2M$ multiplications are needed to form the discrete power spectral density. The velocity estimate takes another M multiplications while the variance estimate needs $2M$. The total number of real multiplications required for the FFT method is therefore $2M\log_2 M+5M$. The pulse pair method must find only the first two or three lags of the autocorrelation, depending on the width estimator used. This corresponds to about $8M$ or $12M$ real multiplications, respectively. In addition to this computational savings, the pulse-pair estimators for mean velocity and spectral width will meet or exceed the FFT estimator's performance for most weather situations. The

exception to this rule is that the FFT mean velocity estimator (2.57) performs slightly better on wide spectra in low signal-to-noise power ratios. The accuracy of these two methods is examined in detail by Zrnic using a perturbation analysis [28] and [29]. Sirmans and Bumgarner compare them to a larger class of estimators in their papers [22] and [23].

2.5 Estimator Biases from Quadrature Gain Imbalance

In analyzing data for this study, an anomaly was found in the coherent quadrature demodulation. A gain imbalance between the in-phase and quadrature channels was present in the CHILL's hardware. This gain imbalance caused an image spectrum to be added to the power spectral density. Sirmans and Bumgarner analyzed the effect this has on the mean velocity estimator (2.44) in [22]. The effect it has on the width estimators (2.53) and (2.54) is analyzed in a similar manner below.

Let the returned signal be represented by

$$V_n = I_n + jQ_n. \tag{2.59}$$

If $m \geq n$, the autocorrelation is

$$R(m-n) = E[V_n^* V_m] = E[(I_n I_m + Q_n Q_m) + j(I_n Q_m - I_m Q_n)]. \tag{2.60}$$

Using the Gaussian spectrum assumption for weather echoes, the following equations hoid ([4];p.441);

$$E[I_n I_m] = E[Q_n Q_m], \tag{2.61a}$$

$$E[I_n Q_m] = -E[I_m Q_n]. \tag{2.61b}$$

The real and imaginary parts of the autocorrelation (2.60) are then

$$\mathrm{Re}[R(m-n)] = 2E[I_n I_m] = 2E[Q_n Q_m], \tag{2.62a}$$

$$\mathrm{Im}[R(m-n)] = 2E[I_n Q_m] = -2E[I_m Q_n]. \tag{2.62b}$$

Now define the imbalanced signal as

$$V_n' = KI_n + jQ_n, \tag{2.63}$$

where the prime denotes imbalance and K is the imbalance. The imbalanced autocorrelation is

$$\begin{aligned} R'(m-n) &= E[V_n'^{*} V_m'] \\ &= E[(K^2(I_n I_m)+Q_n Q_m)+jK(I_n Q_m - I_m Q_n)]. \end{aligned} \tag{2.64}$$

The real and imaginary parts are

$$\mathrm{Re}[R'(m-n)] = \mathrm{Re}[R(m-n)](K^2+1)/2, \tag{2.65a}$$

$$\mathrm{Im}[R'(m-n)] = \mathrm{Im}[R(m-n)]K. \tag{2.65b}$$

With a normalized velocity and the gain imbalance, the mean velocity estimator (2.44) becomes

$$\begin{aligned} v_d' &= -\arg[R'(1)]/\pi \\ &= -\arctan[2K\mathrm{Im}[R(1)]/(K^2+1)\mathrm{Re}[R(1)]]/\pi. \end{aligned} \tag{2.66}$$

If $K = 1$, $\mathrm{Im}[R(1)]/\mathrm{Re}[R(1)]$ must equal $\tan[-v_d\pi]$. Using this fact, (2.66) becomes

$$v_d' = -\arctan[2K\tan[-v_d\pi]/(K^2+1)]/\pi. \qquad (2.67)$$

Therefore, the bias on the velocity estimator (2.44) due to quadrature gain imbalance is

$$v_d - v_d' = v_d - \arctan[2K\tan[v_d\pi]/(K^2+1)]/\pi. \qquad (2.68)$$

This equation is plotted in Fig. 2.2 for several values of the imbalance factor K. The plot reveals that this estimator is quite immune to the gain imbalance. A value of K as small as 0.6 is needed to throw the estimator off by 0.02.

To check the spectral width estimators, the autocorrelation of (2.49) must be rewritten in terms of the new normalized velocity as

$$R(k) = S_o\exp[j2\pi v_d k]\exp[-(\pi\sigma_v k)^2/2]. \qquad (2.69)$$

The first spectral width estimator (2.53) becomes

$$\sigma'^2_{v1} = 2\ln[|R'(0)/R'(1)|]/\pi^2$$
$$= \frac{1}{\pi^2}\ln\left[\frac{((K^2+1)/2)^2\mathrm{Re}^2[R(0)] + K^2\mathrm{Im}^2[R(0)]}{((K^2+1)/2)^2\mathrm{Re}^2[R(1)] + K^2\mathrm{Im}^2[R(1)]}\right]. \qquad (2.70)$$

Using (2.69) for substitution, then

$$\sigma'^2_{v1} = \frac{1}{\pi^2}\ln\left[\frac{((K^2+1)/2)^2\exp[(\pi\sigma_v)^2]}{((K^2+1)/2)^2\cos^2[\pi v_d] + K^2\sin^2[\pi v_d]}\right]. \qquad (2.71)$$

Then, the first variance estimator bias is

$$\sigma^2_{v1} - \sigma'^2_{v1} = \ln[\cos^2[\pi v_d] + (2K/(K^2+1))^2\sin^2[R(1)]]/\pi^2. \qquad (2.72)$$

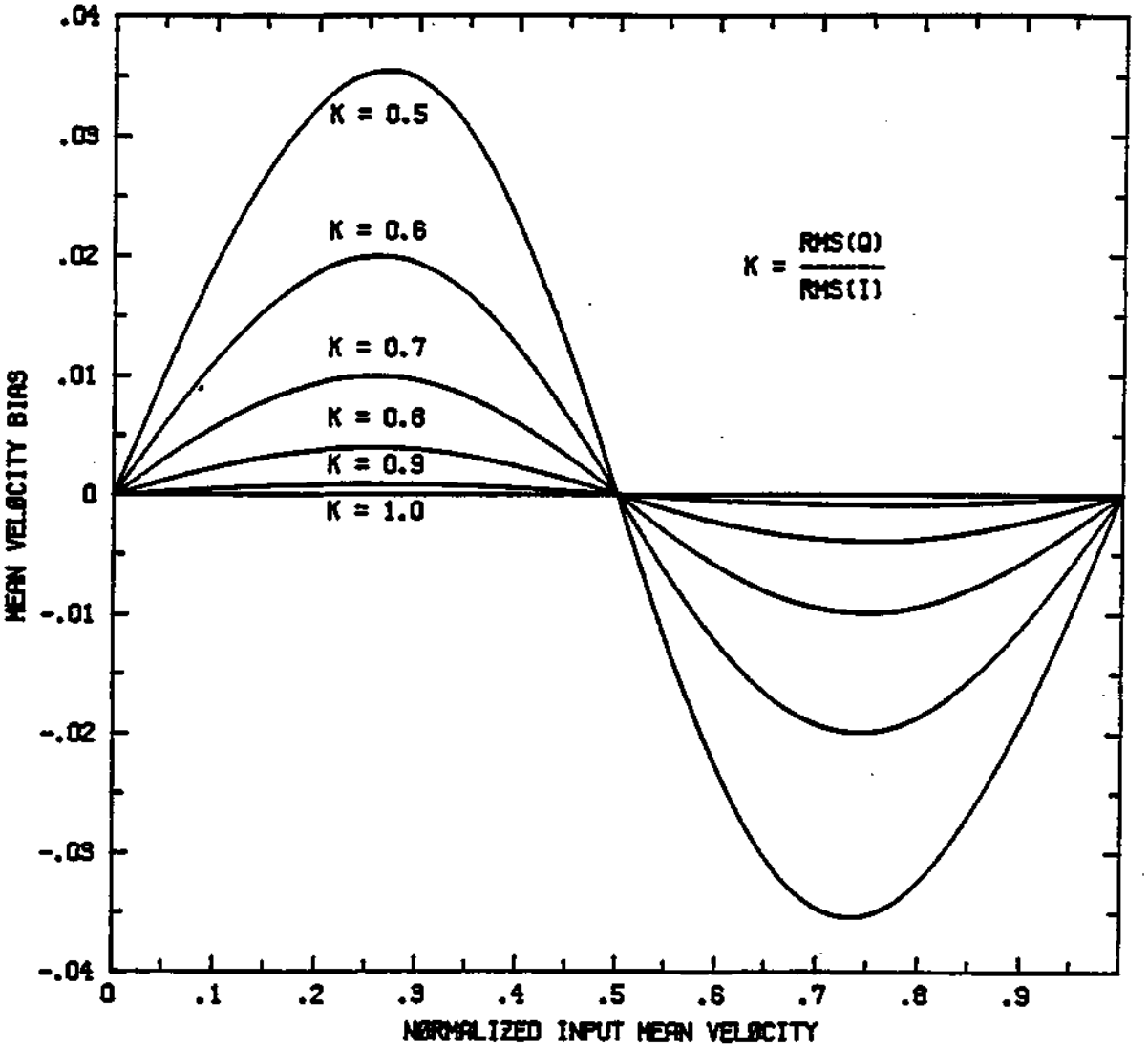


Figure 2.2. Bias of mean velocity estimator (2.44) due to quadrature gain imbalance.

Using a similar derivation, the second variance estimator (2.54) has the bias

$$\sigma_{v2}^2 - \sigma'^2_{v2} = \frac{1}{3\pi^2} \ln\left[\frac{\cos^2[2\pi v_d] + (2K/(K^2+1))^2 \sin^2[2\pi v_d]}{\cos^2[\pi v_d] + (2K/(K^2+1))^2 \sin^2[\pi v_d]}\right]. \quad (2.73)$$

These two width biases are plotted in Figs. 2.3 and 2.4. As the figures show, the second variance estimator performs much better than the first. The bias maxima for the same gain imbalance K are consistently about three times larger for the first estimator. To achieve biases less than 0.003, the first estimator needs an imbalance factor K that exceeds 0.85. The second estimator will satisfy this criterion when K is as small as 0.75.

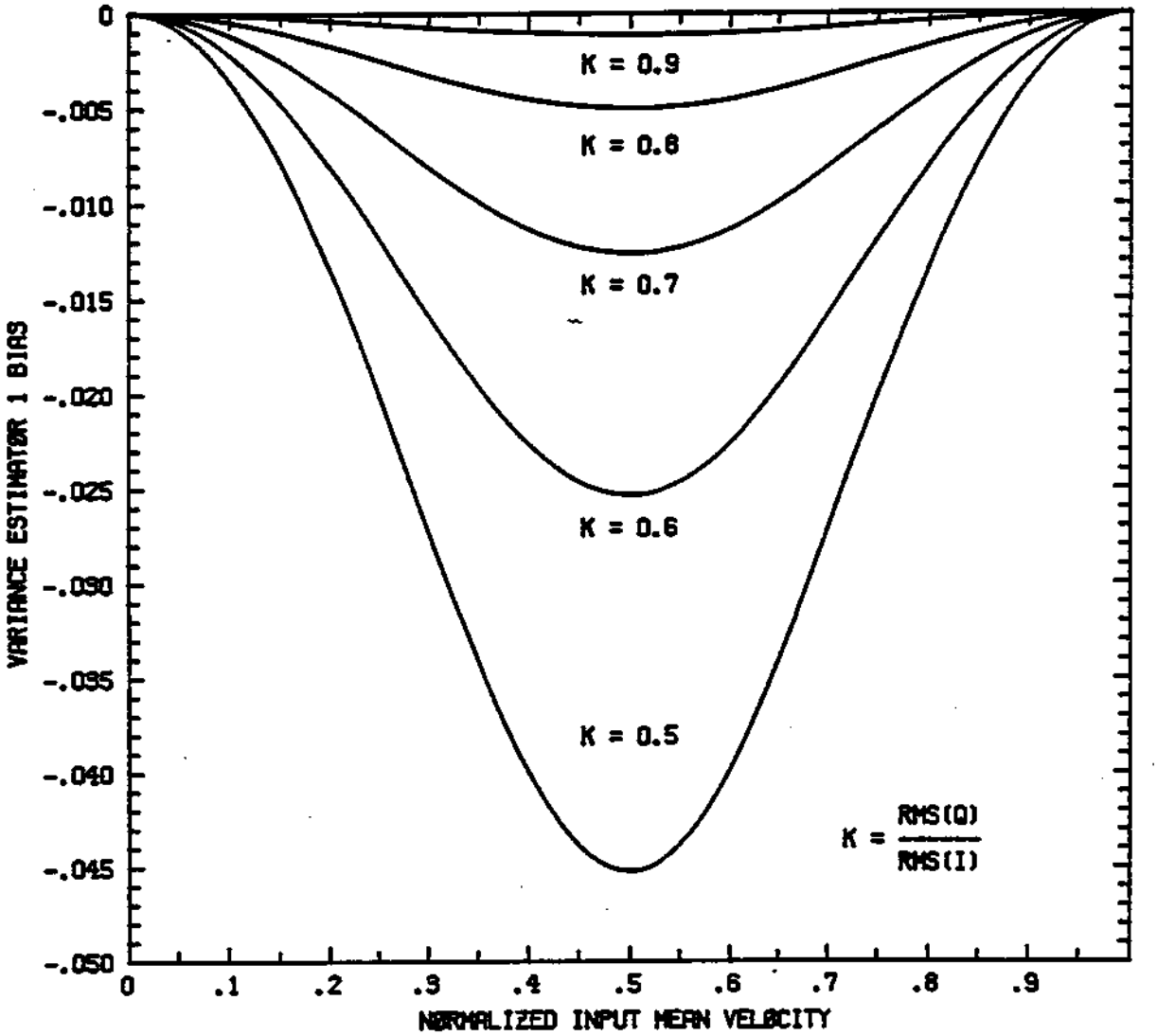


Figure 2.3. Bias of first variance estimator (2.53) due to quadrature gain imbalance.

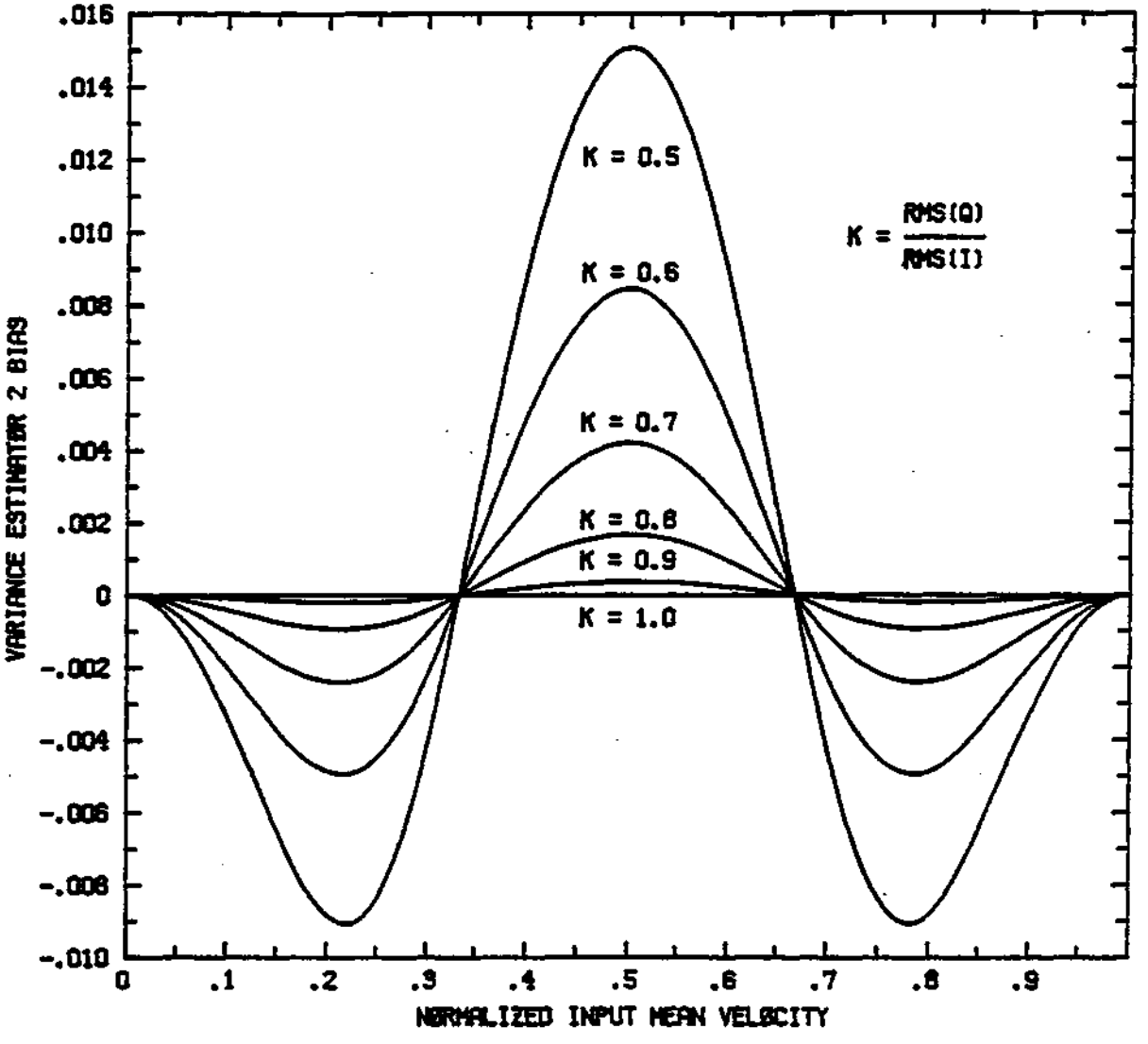


Figure 2.4. Bias of second variance estimator (2.54) due to quadrature gain imbalance.

CHAPTER 3

GROUND CLUTTER

3.1 The Ground Clutter Problem

Many weather phenomena of interest in radar meteorology occur near the bottom of the troposphere. Table 3.1 shows several of these common phenomena and their typical altitude ranges [26]. For the radar to see these phenomena, scans at quite low elevation angles are needed.

Table 3.1. Altitudes of some lower troposphere weather phenomena.

Weather Phenomena	Typical Altitudes (meters)
Severe Thunderstorms	500 - 15,000
Thunderstorms	500 - 7,500
Stratiform Precipitation	0 - 2,000
Gust Fronts	0 - 1,000
Down Bursts	0 - 300

To determine the radar ray's propagation path, that is, its height above ground at a given elevation angle and range, two important factors must be considered. The first is the increase in height due to the earth's curvature. A ray at zero elevation does not travel parallel to the ground, it travels more in a path tangent to it. In fact, if the atmosphere were homogeneous, the ray's path would be perfectly tangent. The actual atmosphere is far from homogeneous. It has a refractivity gradient that tends to

bend rays back towards the earth. By assuming that the atmospheric pressure, temperature and humidity are linearly spherically stratified, the ray's propagation path is found to follow a four-thirds effective earth's radius model. This model gives the ray's height above ground at a given elevation angle ϕ_e and range r as

$$h = [r^2 + a_e^2 + 2ra_e \sin[\phi_e]]^{1/2} - a_e, \qquad (3.1)$$

where a_e is four-thirds the earth's radius (a_e = 8500 km). Figure 3.1 uses this model to show propagation paths for several low elevation angles.

Comparing the curves in Fig. 3.1 with the altitudes of Table 3.1, it is evident just how low elevation angles must be to observe some weather phenomena. Unfortunately, echoes from these low elevations can easily be polluted with ground clutter. Trees, crops and other vegetation, along with more rigid structures such as buildings, telephone poles and power distribution towers, all contribute to the corruption of weather echoes. Since these ground targets may have much stronger reflectivities than that for the surrounding weather, even ground returns in the antenna side lobes may corrupt the signal. As an example, suppose the radar is used to examine a low altitude (600 m) thunderstorm 100 km away. Figure 3.1 shows that an elevation angle of zero is needed. At the closer range of 40 km, this same propagation path leaves the beam center only 100 m off the ground. A typical beam width is one degree. At 40 km, this one degree translates into a beam about 100 m wide. This means that

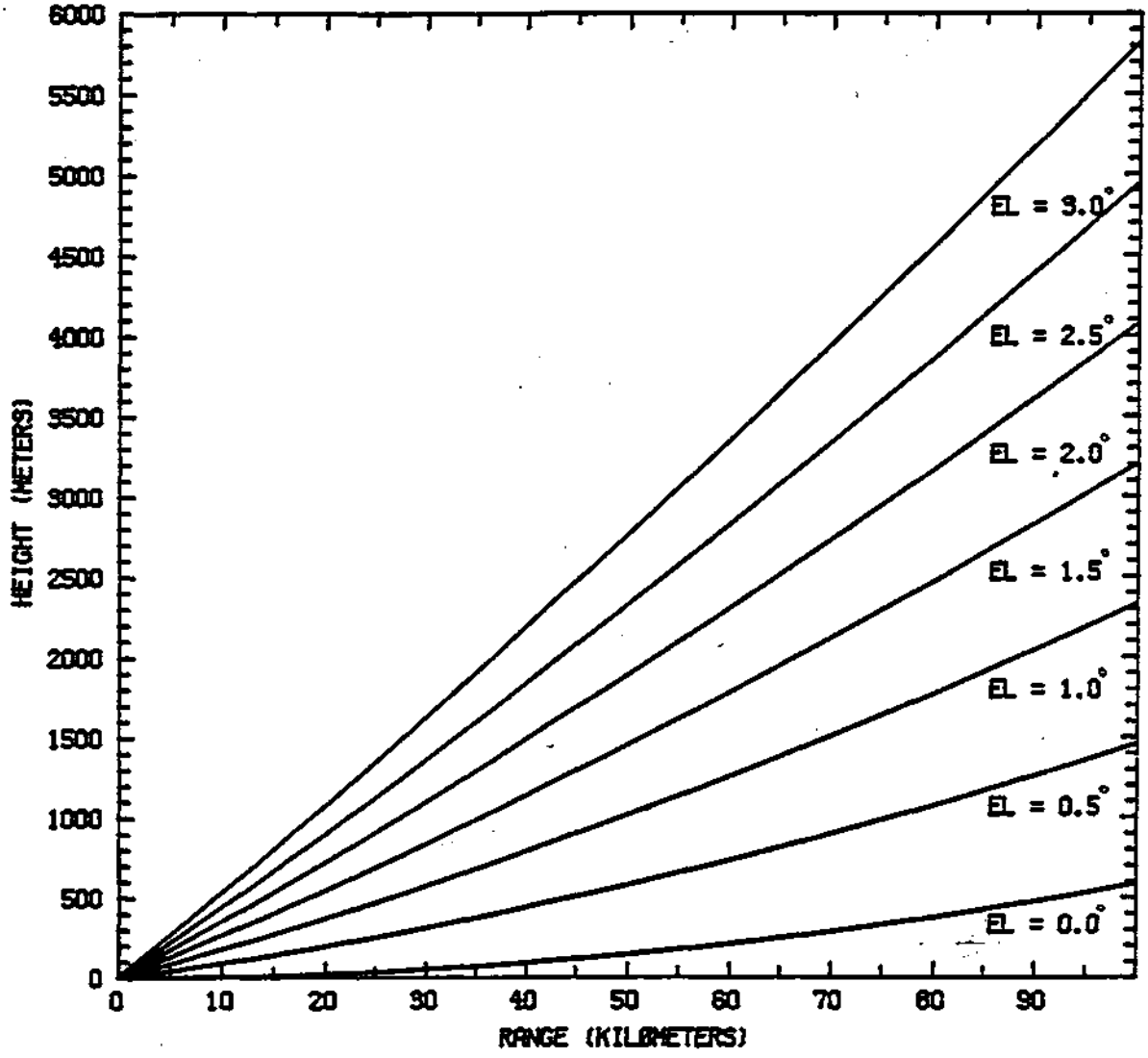


Figure 3.1. Ray propagation paths for the four-thirds earth's radius model.

anything taller than 50 meters is in the beam's main lobe while the side lobes extend all the way to the ground.

Ground returns can corrupt weather echoes in two additional ways, the first being anomalous propagation. This phenomenon is explained in detail by Skolnik in [24]. Briefly, anomalous propagation can occur when during some unusual atmospheric conditions, such as a humidity inversion associated with a thunderstorm, the electromagnetic waves refract much more than the curves of Fig. 3.1 predict. The waves can refract as much as to be bent back to earth. When this happens, the entire ground appears in the beam's main lobe, causing large returns. The other problem concerns multi-trip echoes. As mentioned in Section 2.1, when the return from a target at range r is being received, the return from the previous pulse off a target $cT_s/2$ farther away is also being received. If the far target reflects a large enough echo, it can overpower a weaker target up-close.

To quantify this multi-trip relationship, suppose a distributed target of uniform reflectivity Z_o exists everywhere. From the classic radar equation, the power of the first-trip echo from a narrow transmitted pulse is

$$P_1 = CZ_o/r^2, \qquad (3.2)$$

where C is a calibrated constant. The reason the range dependence is $1/r^2$ and not $1/r^4$ is that the pulse resolution volume grows as r^2. This cancels out an r^2 term in the denominator of the classic radar equation. If r_u denotes the

unambiguous range, the n^{th} -trip echoes have power given by

$$P_n = CZ_o/(r+(n-1)r_u)^2. \qquad (3.3)$$

Taking the ratio of (3.3) over (3.2), the n^{th} -trip echo is seen to contribute power relative to the first-trip echo as

$$10\log[P_n/P_1] = -20\log[1+(n-1)(r_u/r)]. \qquad (3.4)$$

Figure 3.2 plots the results of this equation for the first few trips. This figure shows that second-trip echoes are less than 10 dB below first-trip echoes for the last half of the unambiguous range. This means second-trip clutter echoes will overpower first-trip weather echoes when the clutter echoes are only 10 dB stronger. This is not a large difference when considering the roughly 70 dB dynamic range of weather signals. The saving grace is that unless the unambiguous range is very small, the ray is usually high enough in altitude by the second trip that ground clutter is not a problem. The only time this is really problematic is when multi-trip echoes occur in conjunction with anomalous propagation. Rays that anomalously propagate to earth a few unambiguous ranges away may corrupt first-trip echoes.

The discussion so far has dealt only with ground clutter's spatial characteristics. The clutter's temporal or spectral characteristics must also be considered. The most evident time dependency is that ground clutter echoes tend to take a relatively long time to decorrelate. This translates into a spectrum which is very narrow. The two general types of ground clutter contribute slightly

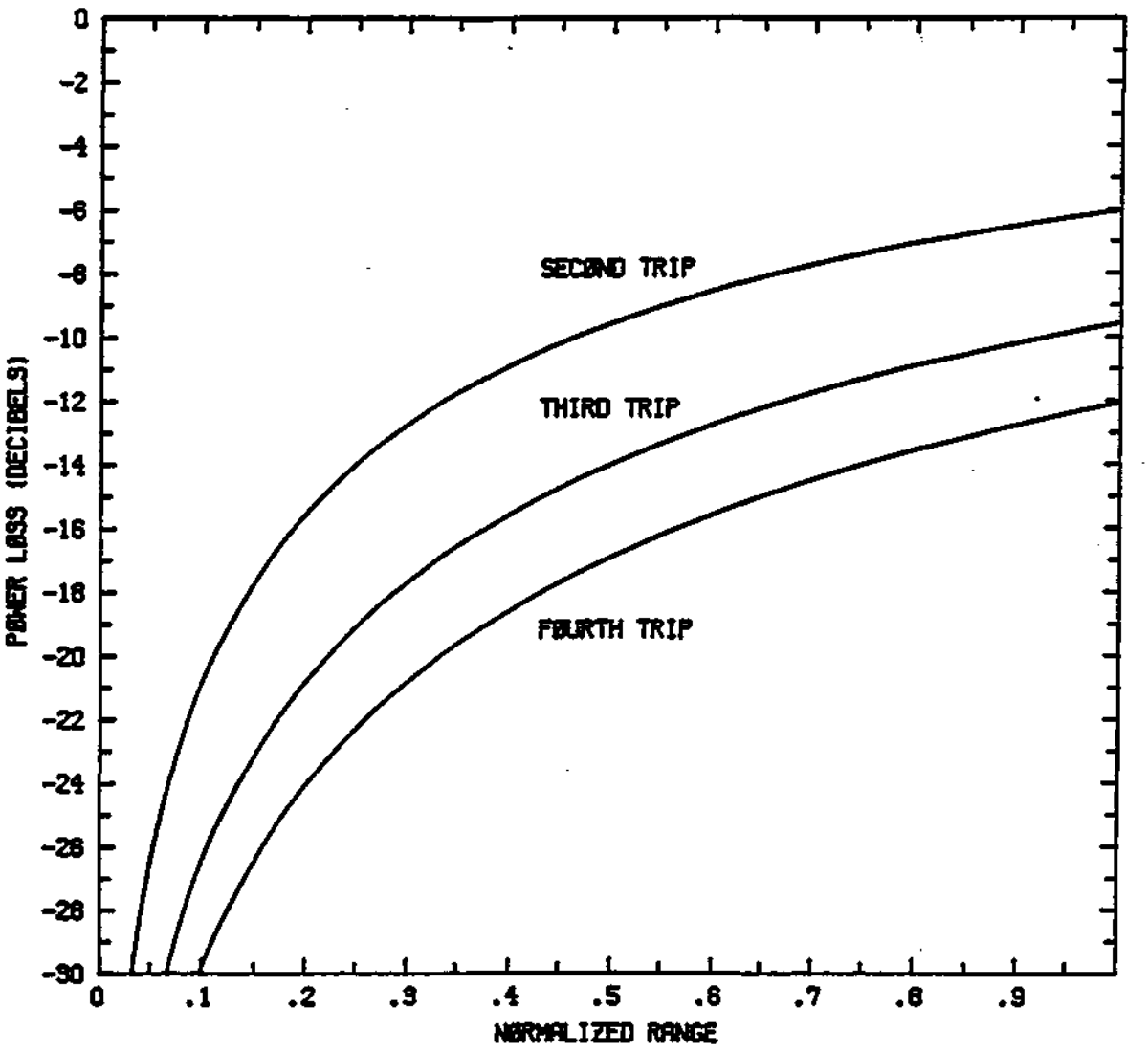


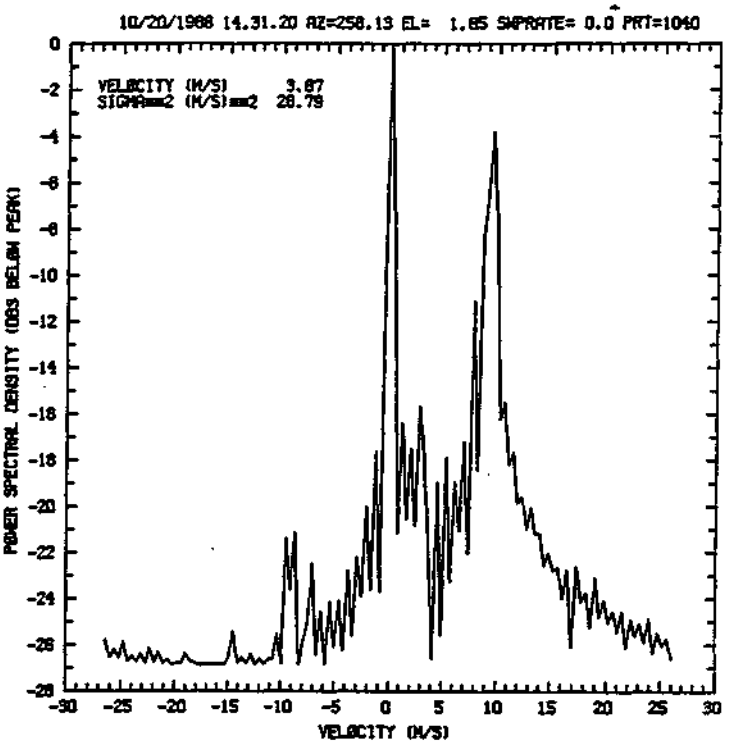
Figure 3.2. Multi-trip power losses as a function of normalized range for constant target reflectivities.

different parts to this spectrum. The discrete rigid structures tend to give just a spectral line at zero velocity. The more distributed structures such as vegetation, along with their movements, contribute a slightly broader spectrum centered at zero velocity. Antenna rotation, along with finite data window effects, tend to broaden spectral widths as a whole.

The effects of ground clutter presence on weather echo parameter estimators can intuitively be seen by examining Figs. 3.3 and 3.4. These data as well as the rest of the real data used in this study were taken at the CHILL with a pulse repetition time of 1.04 ms. Data blocks of 256 complex samples were used for these plots and the autocorrelation functions were Bartlett windowed.

The spectrum in Fig. 3.3(a) is that of a ground-clutter corrupted weather echo. The weather echo is from a small rainshower. It appears in the power spectral density as the peak centered around a 9 m/s velocity. The ground clutter, which appears as a peak around zero velocity, is from a radio tower. The radar antenna is stationary and the illuminated cell is approximately 50 km away. At this distance the beam center is 1800 m in the air. This means the radio tower is entering the echo through an antenna side lobe. Even though the tower is entering through a side lobe, note that it still reflects 4 dB more power than the rainshower in the main lobe. The pulse-pair estimates of mean velocity and variance appear in the upper-left corner of the plot. The variance estimator used is the second one (i.e.,(2.54)). Examining these estimates, it is obvious that

(a)



(b)

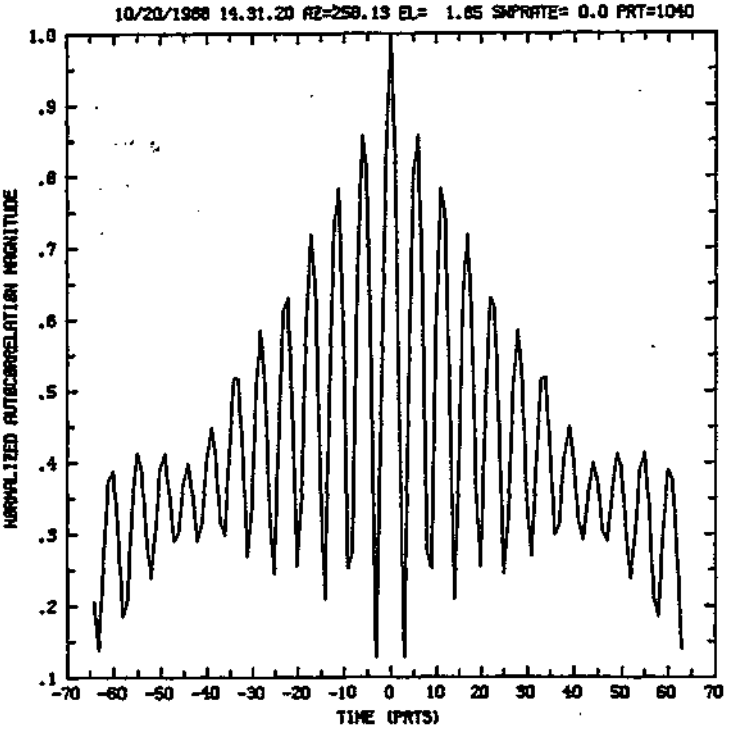
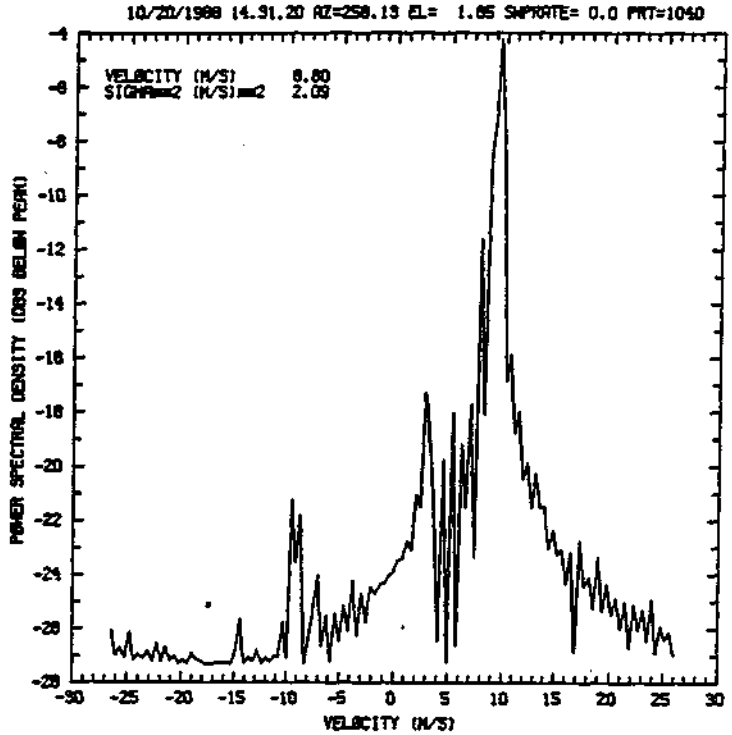


Figure 3.3. (a) Power spectral density of the echo from a cell filled with a rainshower around a radio tower. (b) Autocorrelation of the same echo.

(a)



(b)

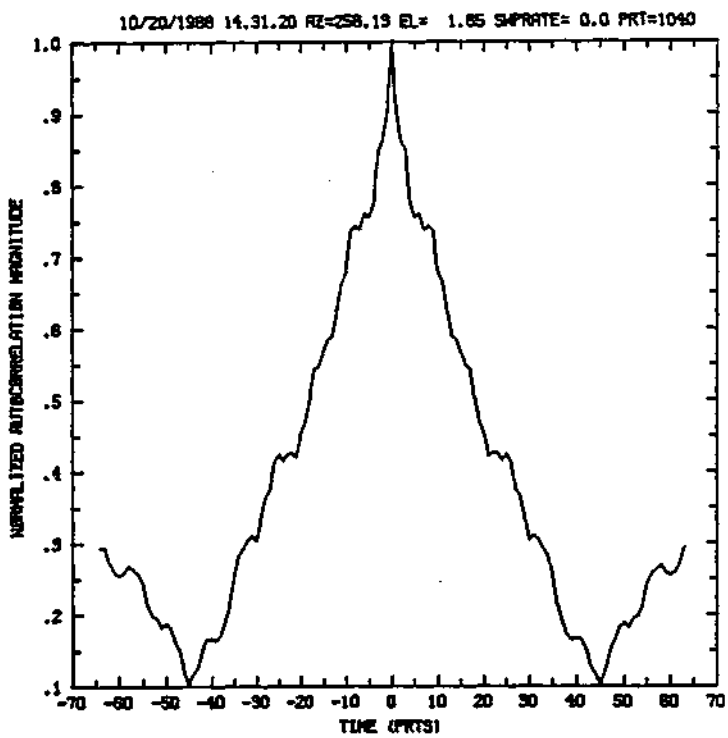


Figure 3.4. (a) Power spectral density of same echo as in Fig. 3.3 but with the ground clutter removed. (b) Autocorrelation of the echo from (a).

the ground clutter biases the mean velocity estimate towards zero. The variance estimate is also biased, to a ridiculously large value. One final thing to note about this figure is the small peak, about 20 dB below the weather peak, centered around a -9 m/s velocity. This is the result of the gain imbalance analyzed in Section 2.5. Figure 3.3(b) shows the autocorrelation of the data. The modulating effect is caused by the ground clutter. If a decorrelation time is defined as the time it takes for the autocorrelation to fall to $1/e$ of its peak value, then the echo in Fig. 3.3(b) decorrelates in about two samples. This corresponds to about 2 ms. Figure 3.4(a) shows the power spectral density of only the weather echo of Fig. 3.3(a). The new mean velocity estimate of 8.8 m/s and the variance estimate of 2.1 m^2/s^2 are much more realistic. Figure 3.4(b) shows the autocorrelation of the weather echo. The modulating effect found in Fig. 3.3(b) is gone. The decorrelation time is increased to about 28 ms.

In order to quantify the errors introduced by the ground clutter signal, assume both weather and clutter have Gaussian spectra. The weather spectrum is centered around some nonzero mean velocity, while the clutter spectrum is centered around zero velocity. Using a velocity variable normalized to unity, the autocorrelations given by (2.69) are

$$R_S(k) = S\exp[-j\pi v_d k]\exp[-(\pi\sigma_S k)^2/2], \qquad (3.5a)$$

and

$$R_c(k) = C\exp[-(\pi\sigma_c k)^2/2], \tag{3.5b}$$

where the c subscript denotes clutter and the s subscript denotes the desired weather signal. If the weather signal is uncorrelated with the clutter signal, the autocorrelation of the sum of the signals is the sum of autocorrelations of the signals:

$$R_{s+c}(k) = R_s(k) + R_c(k). \tag{3.6}$$

Substituting (3.5) into (3.6), the modulation effect in Fig. 3.3(b) is seen to arise from a $\cos[\pi v_d k]$ term:

$$|R_{s+c}(k)| = [S^2\exp[-(\pi\sigma_s k)^2] + C^2\exp[-(\pi\sigma_c k)^2] + 2SC\cos[\pi v_d k]\exp[-\pi^2 k^2(\sigma_s{}^2+\sigma_c{}^2)/2]]^{1/2}. \tag{3.7}$$

The normalized parameter estimates from Section 2.5 are restated below for convenience:

$$v_d = -\arg[R(1)]/\pi, \tag{3.8a}$$

$$\sigma_{v1}{}^2 = 2\ln[|R(0)/R(1)|]/\pi^2, \tag{3.8b}$$

$$\sigma_{v2}{}^2 = 2\ln[|R(1)/R(2)|]/3\pi^2. \tag{3.8c}$$

Using the autocorrelation (3.6), the biased parameter estimates of (3.8) become

$$v_d = \frac{1}{\pi}\arctan\left[\frac{(S/C)\exp[-\pi^2(\sigma_s{}^2-\sigma_c{}^2)/2]\sin[\pi v_d]}{(S/C)\exp[-\pi^2(\sigma_s{}^2-\sigma_c{}^2)/2]\cos[\pi v_d]+1}\right], \tag{3.9a}$$

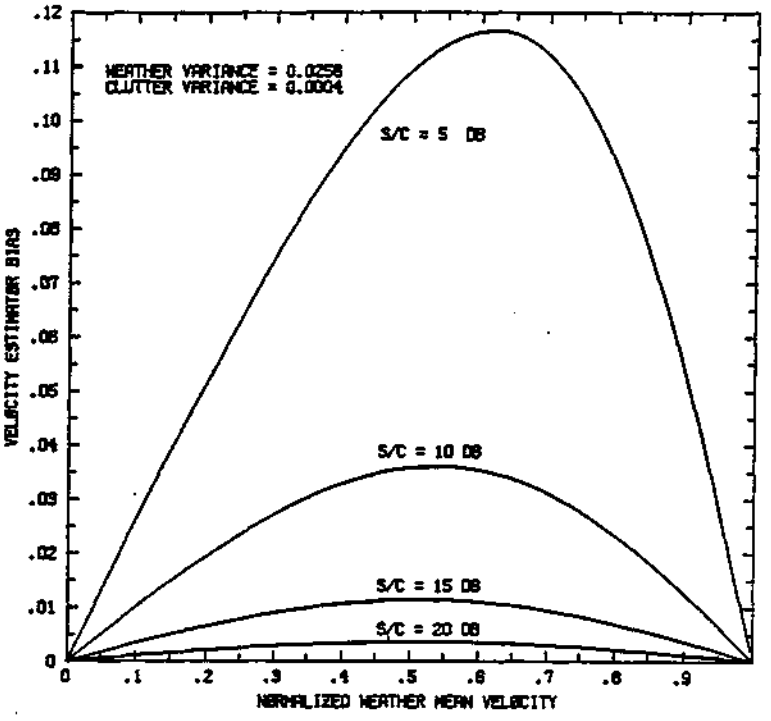
$$\sigma_{v1}^2 = \frac{1}{\pi^2} \ln\left[\frac{((S/C)+1)^2}{[(S/C)^2\exp[-\pi^2\sigma_s^2]+\exp[-\pi^2\sigma_c^2] + 2(S/C)^2\exp[-\pi^2(\sigma_s^2+\sigma_c^2)/2]\cos[\pi v_d]]}\right], \quad (3.9b)$$

$$\sigma_{v2}^2 = \frac{1}{3\pi^2} \ln\left[\frac{[(S/C)^2\exp[-\pi^2\sigma_s^2]+\exp[-\pi^2\sigma_c^2] + 2(S/C)^2\exp[-\pi^2(\sigma_s^2+\sigma_c^2)/2]\cos[\pi v_d]]}{[(S/C)^2\exp[-4\pi^2\sigma_s^2]+\exp[-4\pi^2\sigma_c^2] + 2(S/C)^2\exp[-\pi^2(\sigma_s^2+\sigma_c^2)]\cos[2\pi v_d]]}\right]. \quad (3.9c)$$

The biases that the clutter introduces to the estimators are plotted in Figs. 3.5, 3.6 and 3.7, and are defined as the true weather values minus the values predicted by (3.9). Each estimator is tested under two conditions. The first is for a wide weather spectrum and a narrow clutter spectrum. This would be a typical situation. A wide spectrum is taken as having a variance of 0.0256. If the unambiguous velocity were 25 m/s, this would correspond to a weather spectral width of 4 m/s. A narrow clutter spectrum is taken as having a variance of 0.0004. For the same 25 m/s unambiguous velocity, this would mean a 0.5 m/s clutter spectral width. The second situation the estimators are tested for is a narrow weather width and a wide clutter width. These are both taken as having variances of 0.0016. For the 25 m/s unambiguous velocity, these correspond to 1 m/s spectral widths. This is a more extreme case.

Figure 3.5 shows plots of the velocity estimator biases under the typical and extreme cases outlined above. To achieve the error limit of 0.02, used at the end of

(a)



(b)

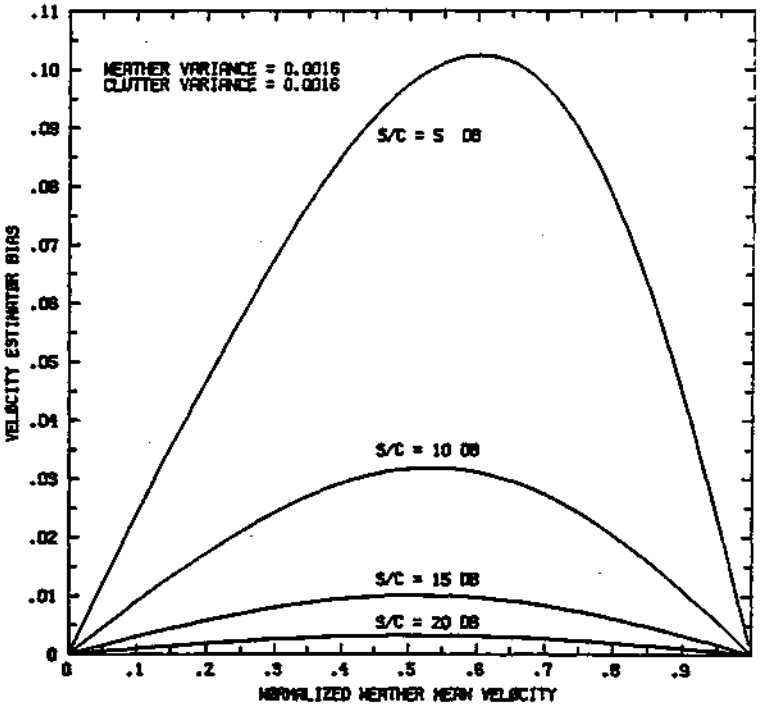
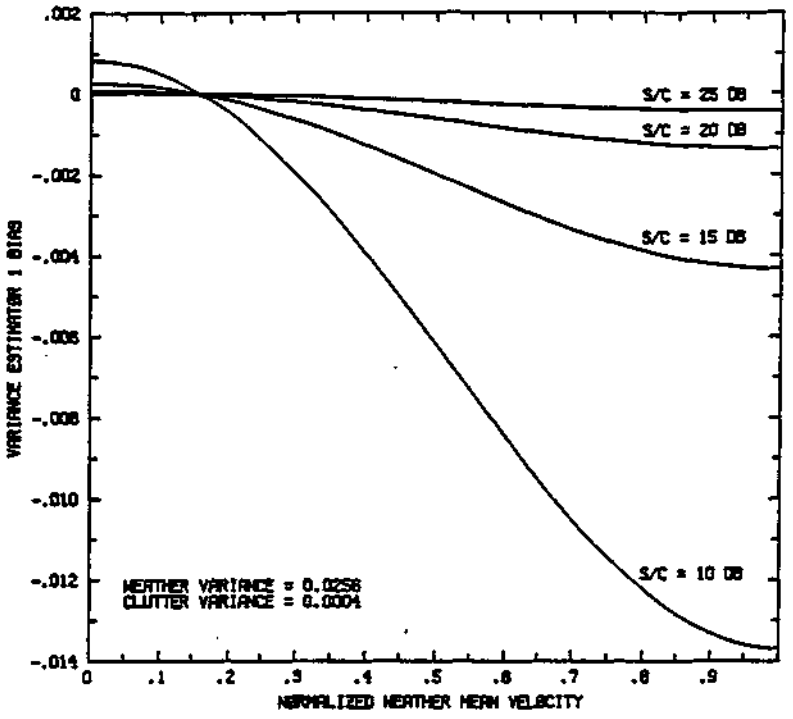


Figure 3.5. Velocity estimator (3.8a) bias due to the presence of clutter. (a) Wide weather spectrum and narrow clutter spectrum. (b) Narrow weather spectrum and wide clutter spectrum.

(a)



(b)

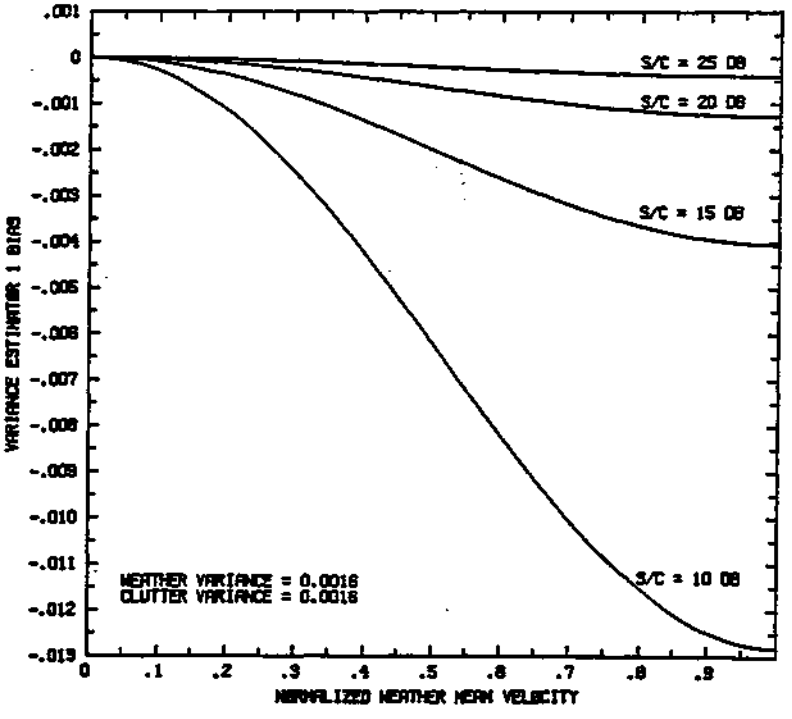
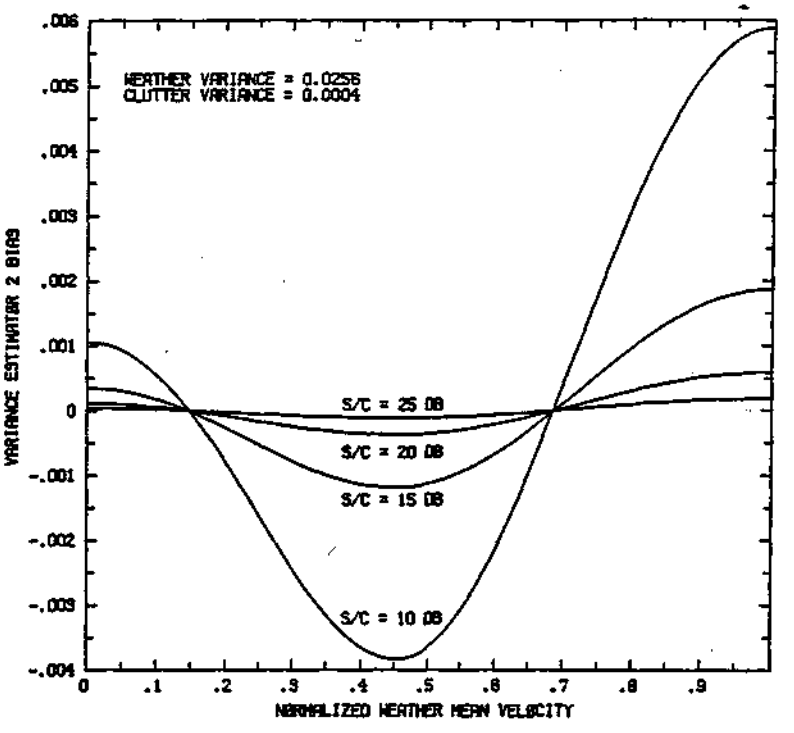


Figure 3.6. First variance estimator (3.8b) bias due to the presence of clutter. (a) Wide weather spectrum and narrow clutter spectrum. (b) Narrow weather spectrum and wide clutter spectrum.

(a)



(b)

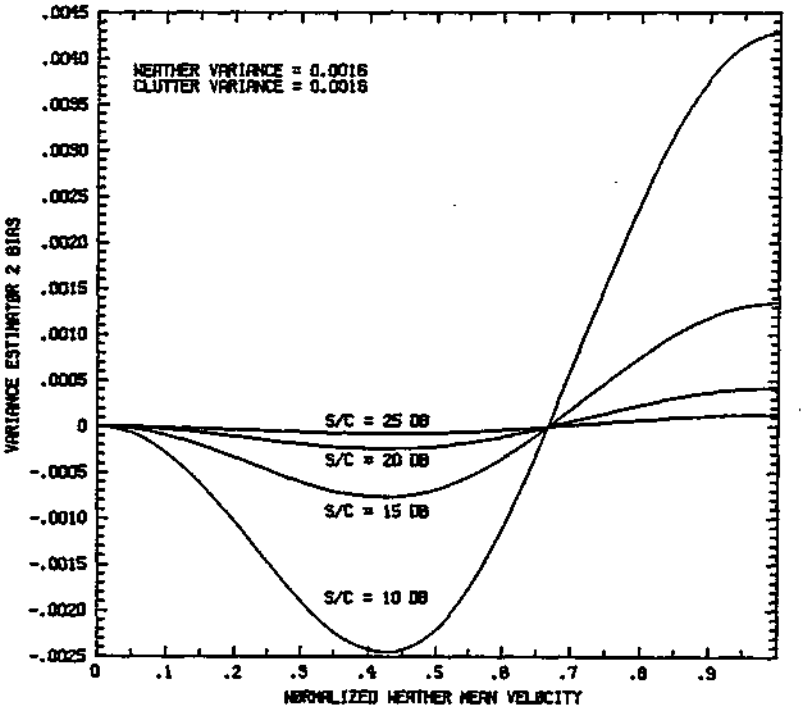


Figure 3.7. Second variance estimator (3.8c) bias due to the presence of clutter. (a) Wide weather spectrum and narrow clutter spectrum. (b) Narrow weather spectrum and wide clutter spectrum.

Section 2.3 in the noise analysis, both width combinations require signal-to-clutter power ratios of about 12 dB. Figure 3.6 shows plots of the clutter bias for the first variance estimator (3.8b). Once again, note the error limits from the noise analysis of Section 2.3. To have biases less than 0.003, this estimator requires a signal-to-clutter power ratio of about 17 dB for both the typical and extreme situations. Figure 3.7 is the same as Fig. 3.6, except the second variance estimator (3.8c) is used. This estimator requires only a 13 dB signal-to-clutter power ratio to meet the 0.003 error limit for the typical situation. The extreme width situation needs only a 12 dB separation.

For all three estimators, the bias differences between typical and extreme width situations are small. In fact, after looking at many more width combinations, these biases seem to change very little so long as normalized variances satisfy $\sigma_c^2 < \sigma_s^2 << 1$. When comparing the two variance estimators, the second one (3.8c) again outperforms the first one (3.8b). Considering the corresponding plots of Figs. 3.6 and 3.7, the first variance estimator consistently needs signal-to-clutter power ratios that are 5 dB larger to perform as well as the second estimator. This result provides another reason for using the second variance estimator instead of the first.

3.2 Ground Clutter Models

As with all models, ground clutter models must balance accuracy with ease of use. The high variability of ground clutter makes this task especially difficult. Models can quickly become quite complex and burdensome to use. As already mentioned, ground returns usually fall into two basic classes, distributed and discrete. The distributed targets are harder to model than the discrete. Discrete models are actually fairly straightforward.

Examples of discrete targets are buildings, telephone poles and radio towers. Since even in strong winds these structures move very little, they should give nearly constant returns when the radar is stationary. If the antenna is moving, the returns should have power that changes in time proportional to the two-way antenna power beam pattern. The amplitude of the return should change in time proportional to the two-way antenna electric-field beam pattern. This is proportional to the one-way antenna power beam pattern for a monostatic radar. Figure 3.8 shows the one-way antenna power beam pattern for the CHILL radar. This is an azimuth cut at a zero degree elevation.

In actual systems the returned signal from a fixed scatterer will have a time-varying phase. This is caused by hardware instabilities and is very hardware dependent ([1],[5];pp.6.1-6.8). Since the variation in this phase is usually small, it can be considered constant, and without loss of generality, zero. Therefore, a segment of the returned signal from a discrete target can be modeled by

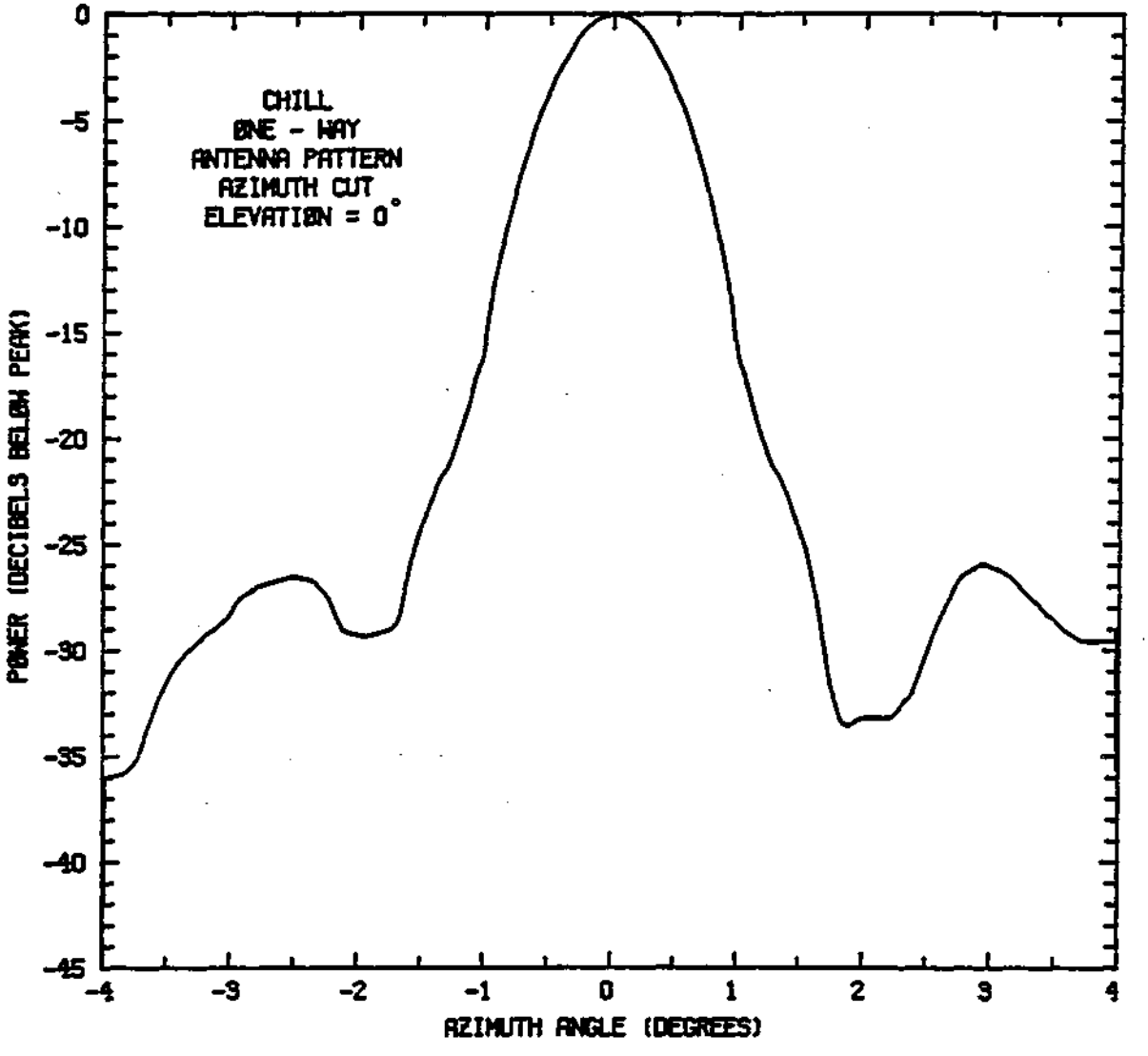


Figure 3.8. CHILL one-way antenna pattern.

$$V(k) = \beta(\alpha k T_s), \tag{3.10}$$

where $\beta(\theta)$ is the one-way antenna power beam pattern.

Distributed ground clutter echoes have traditionally been modeled as Gaussian random processes with Gaussian spectra (e.g.,[31]). If this model is used then, except for mean velocity, both ground clutter and weather echoes have the same model. If the radar is looking perpendicular to the velocity field, then the two processes appear identical. This model is intuitively satisfying, considering the assumptions leading to the Gaussian model for weather echoes in Section 2.2. The assumption of many scatterers moving independently seems well satisfied by corn stalks and the like. Indeed, much ground clutter data have been collected supporting this model (e.g.,[12]). We should be aware, though, that the Gaussian clutter model is not entirely accurate. A problem with the data used to support the Gaussian model is that it has been usually taken from airplanes. The geometry of this situation means the radar beams hit the ground with large grazing angles. The beam's footprint takes in a large spatial area so that many scatterers contribute to the echo. Weather radars, being ground based, see ground clutter a little differently. Ground-based radar beams strike ground targets at very small grazing angles so that some obscuration or shadowing of the clutter is highly probable. Shadowing can lead to a few scatterers being dominant in the echoes, which violates a basic assumption in the central limit theorem used to derive the Gaussian model.

To demonstrate that the Gaussian assumption may be violated, consider the echo signal amplitude distributions from ground targets illuminated at low grazing angles. If the Gaussian assumption holds, the echo amplitude should be Rayleigh distributed as in (2.22). Sekine et al. [21], however, have fit low grazing angle ground return amplitudes to the Weibull distribution

$$p[|V|] = (c/b)(|V|/b)^{(c-1)}\exp[-(|V|/b)^2], \qquad (3.11)$$

where $c > 0$ and b is a positive constant that normalizes the total probability to unity. Actual data fit this distribution quite well for shape parameter c varying from 1.5 to 2. (Notice that a shape parameter of 2 corresponds to the Rayleigh distribution of (2.22).) In addition, we note that if several ground targets dominate the return, making the in-phase and quadrature channels non-Gaussian, then these same targets will contribute velocities that dominate the spectrum. Thus the velocity spectra will also be non-Gaussian.

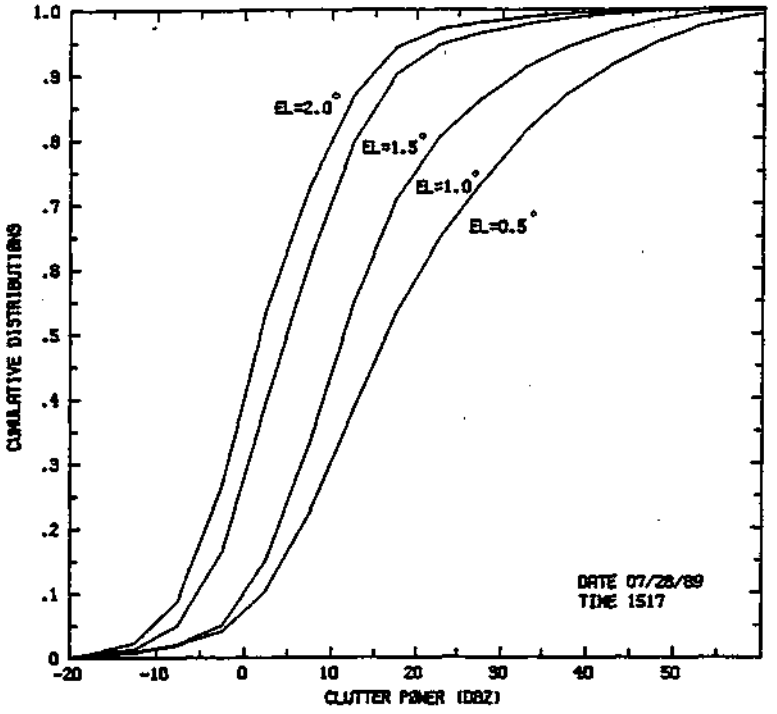
Actual ground clutter spectra have tails that decay faster than a Gaussian function. Still, a Gaussian model seems to be fairly accurate when the antenna is scanning. This occurs because the convolution of the antenna scan spectrum with the actual ground clutter spectrum results in something quite close to a Gaussian spectrum.

3.3 Ground Clutter Variation

The characteristics of ground clutter returns depend on many factors. This section will attempt to show some of these dependencies by using data collected at the CHILL radar.

Figure 3.9 shows cumulative distributions of clutter power for four low elevation angles. The data used were from ground target returns out to 40 km. Full 360 degree scans were used with 150 m range gating. Power estimates were calculated about once every degree scanned. No weather cells or other airborne particles were noticeably present when the data were collected. Figure 3.9(a) shows cumulative less than plots where the lines signify the portion of cells with power less than or equal to the abscissa. These curves are the same as statistical distribution functions. The curves in Fig. 3.9(b) were formed by subtracting the curves in Fig. 3.9(a) from one. These lines signify the portion of cells with power more than or equal to the abscissa. As expected, clutter power decreases as the elevation angle increases. The 95 percentiles of the distributions for elevation angles of 0.5°, 1.0°, 1.5°, and 2.0° are seen to be 48, 38, 24 and 18 dBZ, respectively. Figure 3.10 demonstrates the clutter's range dependency. These data are the same as those used for the distributions in Fig. 3.9. The decrease in clutter power with range is to be expected since the beam is increasing in height as the range increases. The large peak at about 7 km is from the Champaign-Urbana metro area.

(a)



(b)

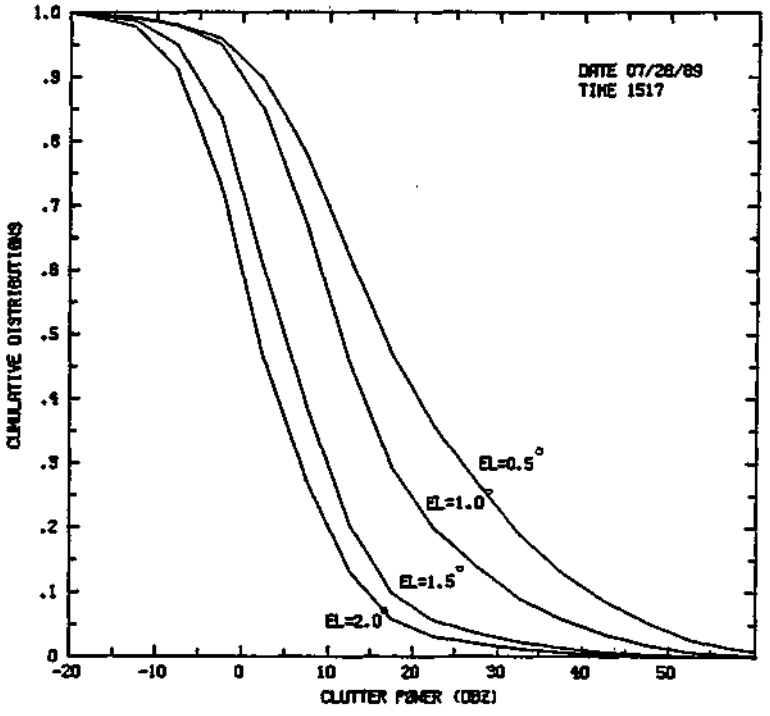


Figure 3.9. Cumulative clutter power distributions for cells out to 40 km. (a) Cumulative less than plot. (b) Cumulative more than plot.

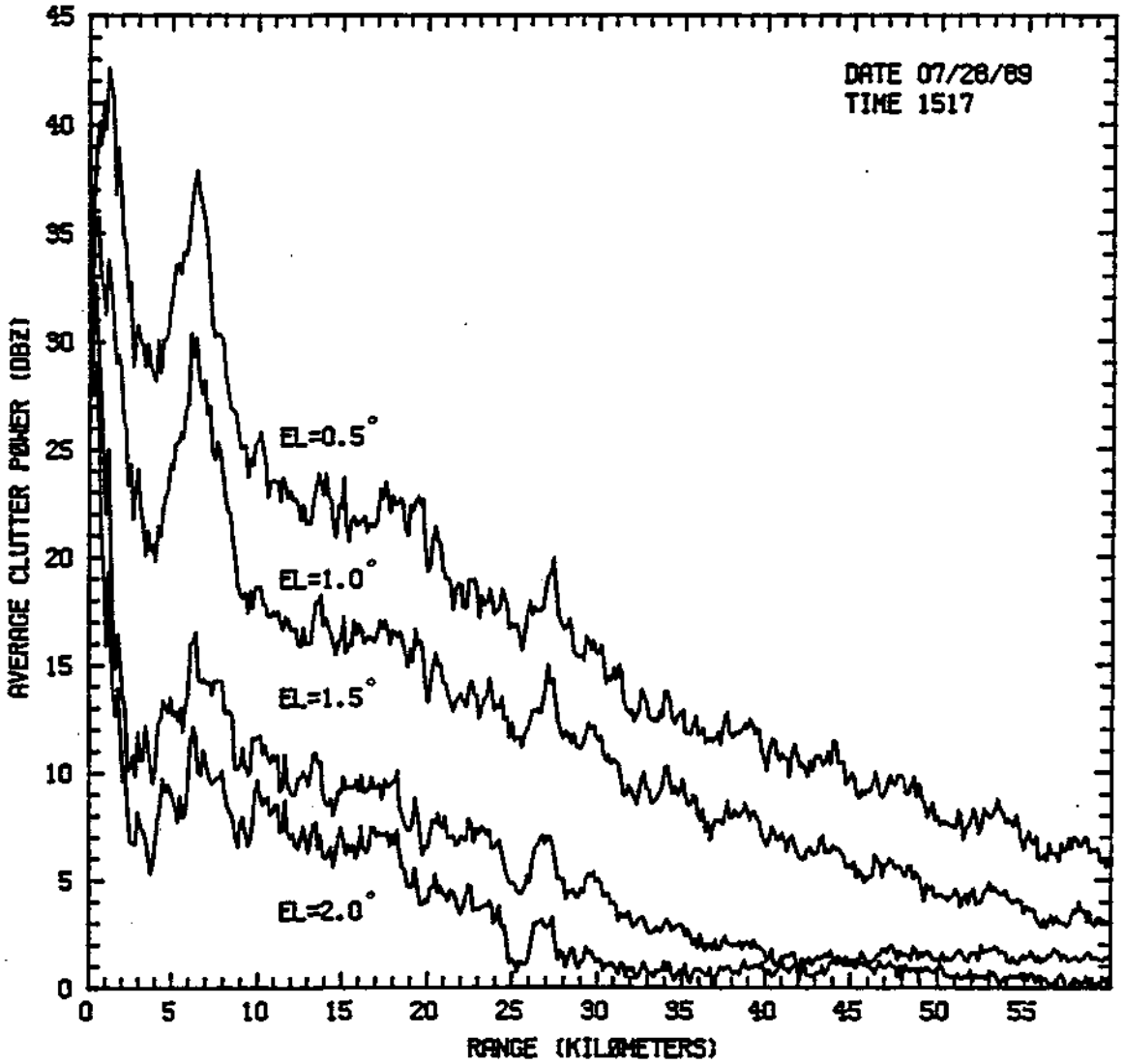
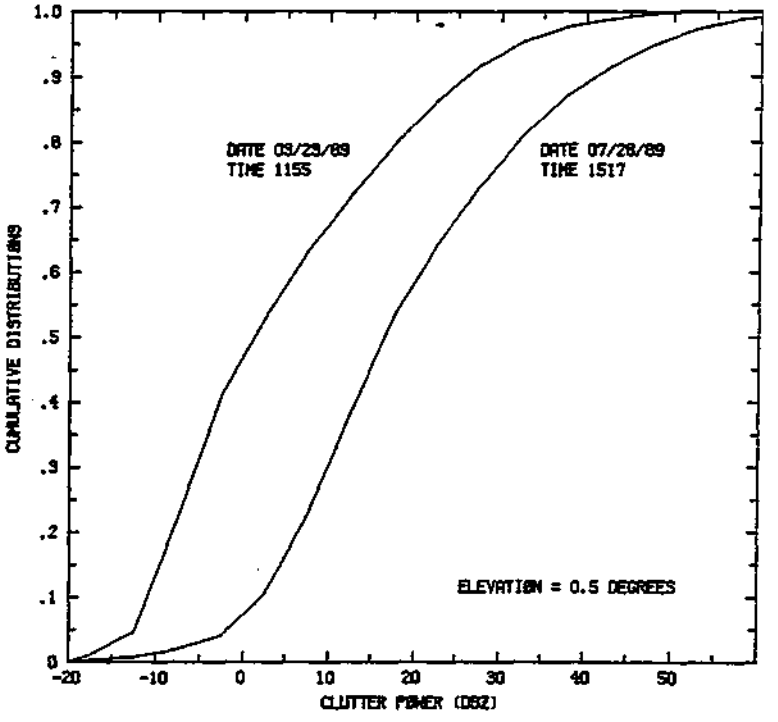


Figure 3.10. Average ground clutter power as a function of range for several elevation angles.

The overall ground conditions also play an important role in ground clutter's characteristics. Figure 3.11(a) is a comparison of cumulative clutter power distributions between early spring and late summer. As in Fig. 3.10, the data used for these curves are from clutter returns out to only 40 km. The presence of crops and other vegetation in the late summer makes a dramatic difference. The 95 percentile of the distribution increases from 32 dBZ to 48 dBZ from early spring to late summer. Figure 3.11(b) shows the difference that wet ground makes in clutter power distributions. The overall powers are small here because the data for these distributions are from a partial scan of the area. To eliminate weather echoes, only the southeast quarter of a full scan could be used. This area is absent of any large targets such as cities or towns. It is mainly just farmland. The area had rainshowers pass over a few hours before the data were taken on July 19, 1989. The same area was dry when the next scan was taken on July 20, 1989. The wet ground did increase reflectivity, but not by much. When comparing the 95 percentiles of the distributions, the wet ground only increased reflectivity by 2 dBZ.

In the last few paragraphs, clutter power relationships were explored. For the rest of this section some ground-clutter spectral width dependencies are studied. Since ground clutter does not necessarily have a Gaussian power spectrum, pulse-pair estimation can not be used to estimate spectral width. A spectral shape-independent width estimator such as the FFT estimator (2.58) is needed. All of the following analysis was done using the FFT estimator. The

(a)



(b)

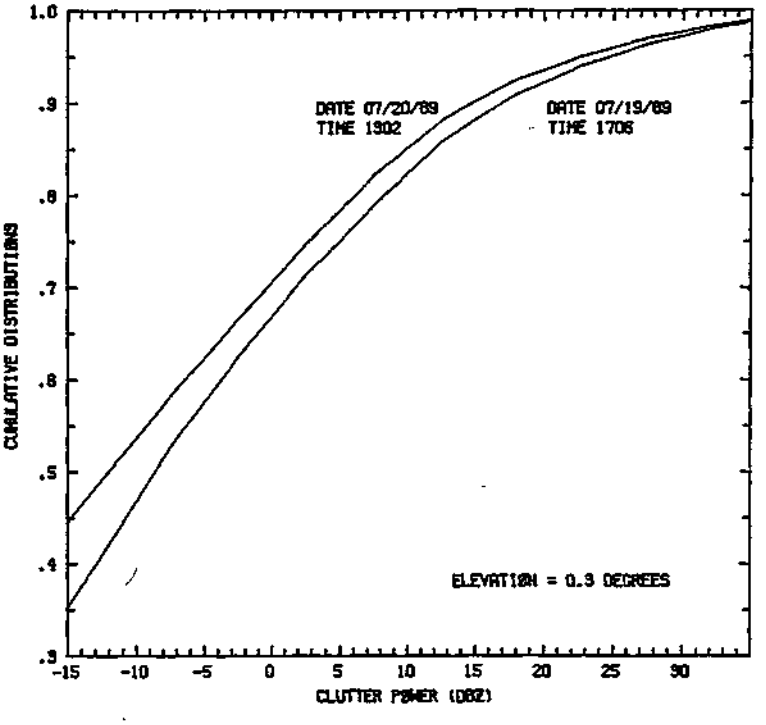


Figure 3.11. Comparisons of clutter power distributions for different ground conditions. (a) The presence of vegetation in late summer shows a dramatic increase in clutter power. (b) Wet ground on July 19 causes little increase in clutter power over dry ground on July 20.

data were weighted with a Hamming window to reduce finite data-length errors. By using data blocks 512 samples long,variance biases were reduced to about 0.002 m^2/s^2. Since these biases are quite small, they were not removed from the estimates. To reduce errors introduced by noise, only velocities in the interval of 3 m/s to -3 m/s were used by the estimator (2.58).

Figures 3.12 through 3.15 are ground-clutter spectral-width distributions for several clutter types. Each distribution is estimated from 100 to 300 data blocks. Each data block contributes one width estimate. The width estimates are placed into a histogram with a fine resolution, 0.005 m/s. Instead of weighting all estimates equally, each estimate is weighted by the power of its data block divided by the total power of all data blocks used for the distribution. This ensures that the wide width estimates obtained from areas void of ground clutter, with their flat spectra, will not corrupt the distributions. Winds were light when all of these data were collected.

Figure 3.12 is a distribution for returns from the WAND radio tower. As expected, the spectra are quite narrow. A full 95 percent of the widths are narrower than 0.2 m/s while 90 percent are narrower than 0.12 m/s. Figure 3.13 shows the distribution for ground returns from the Champaign-Urbana metro area. The 95 percentile of clutter width for these data is 0.38 m/s. The increase in spectral widths of these returns over those from the WAND tower in Fig. 3.12 can probably be attributed to the presence of more distributed targets surrounding the buildings in the city.

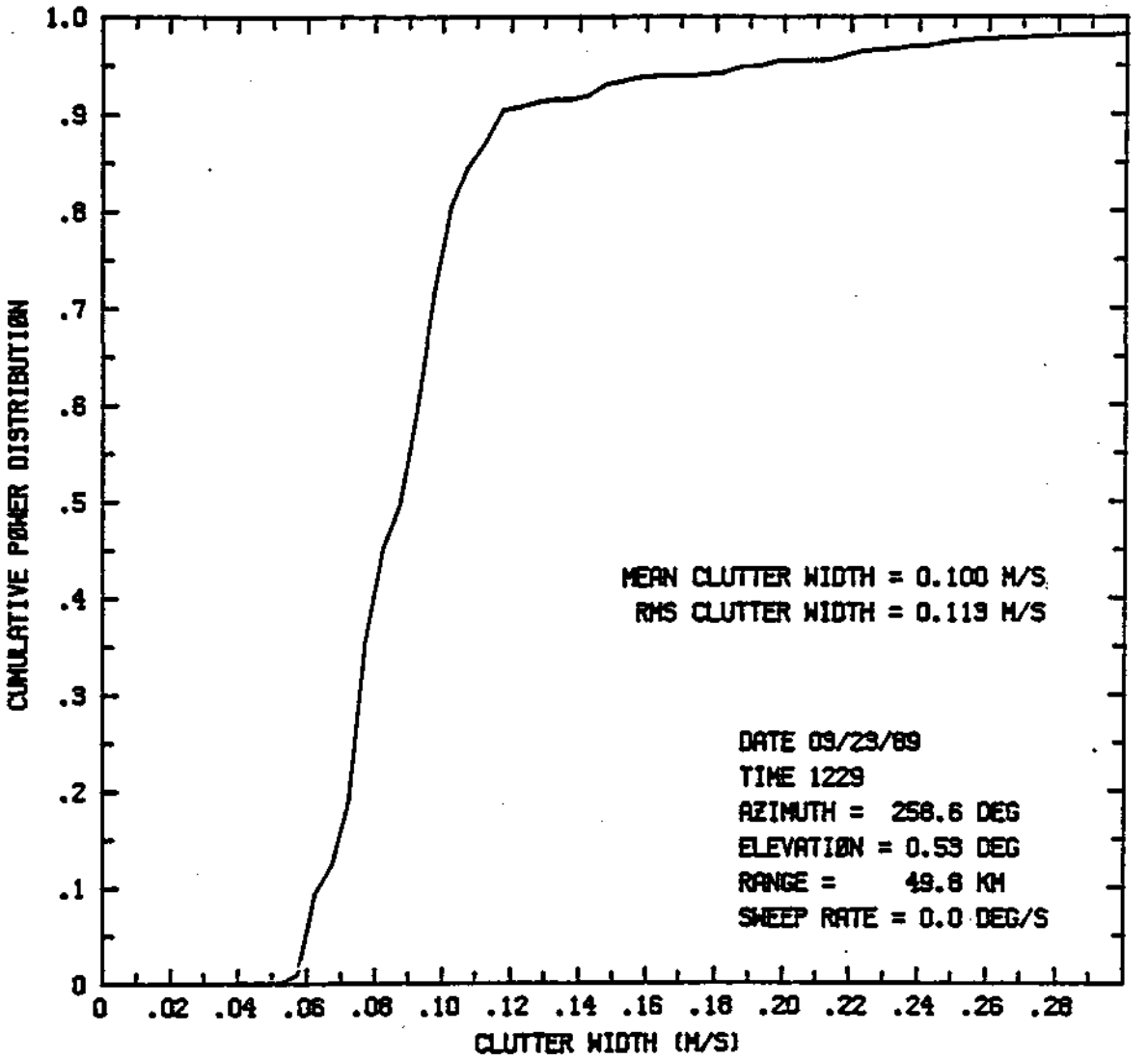


Figure 3.12. Ground-clutter spectral-width cumulative distribution for returns from the WAND radio tower.

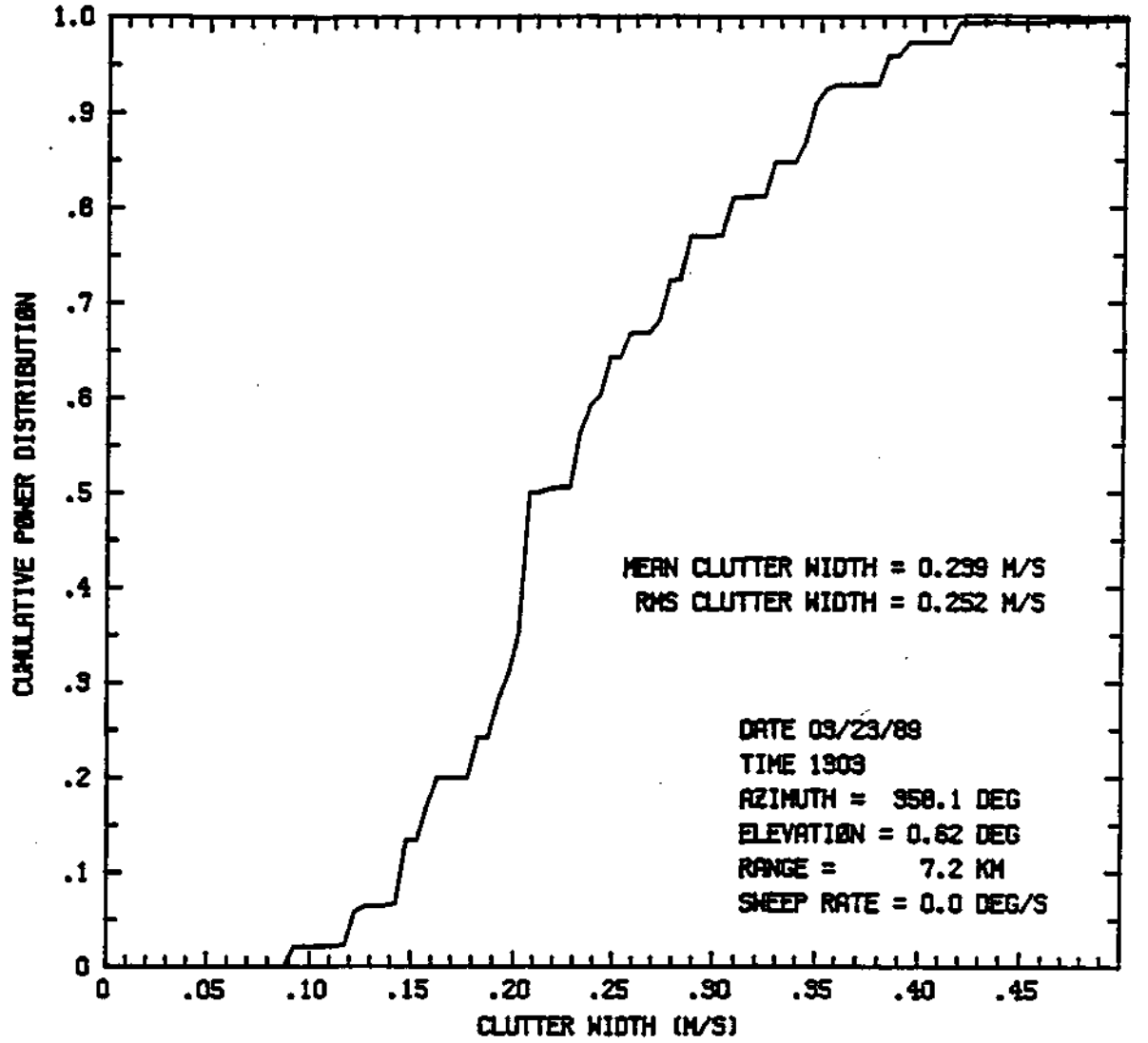


Figure 3.13. Ground-clutter spectral-width cumulative distribution for returns from the Champaign-Urbana metro area.

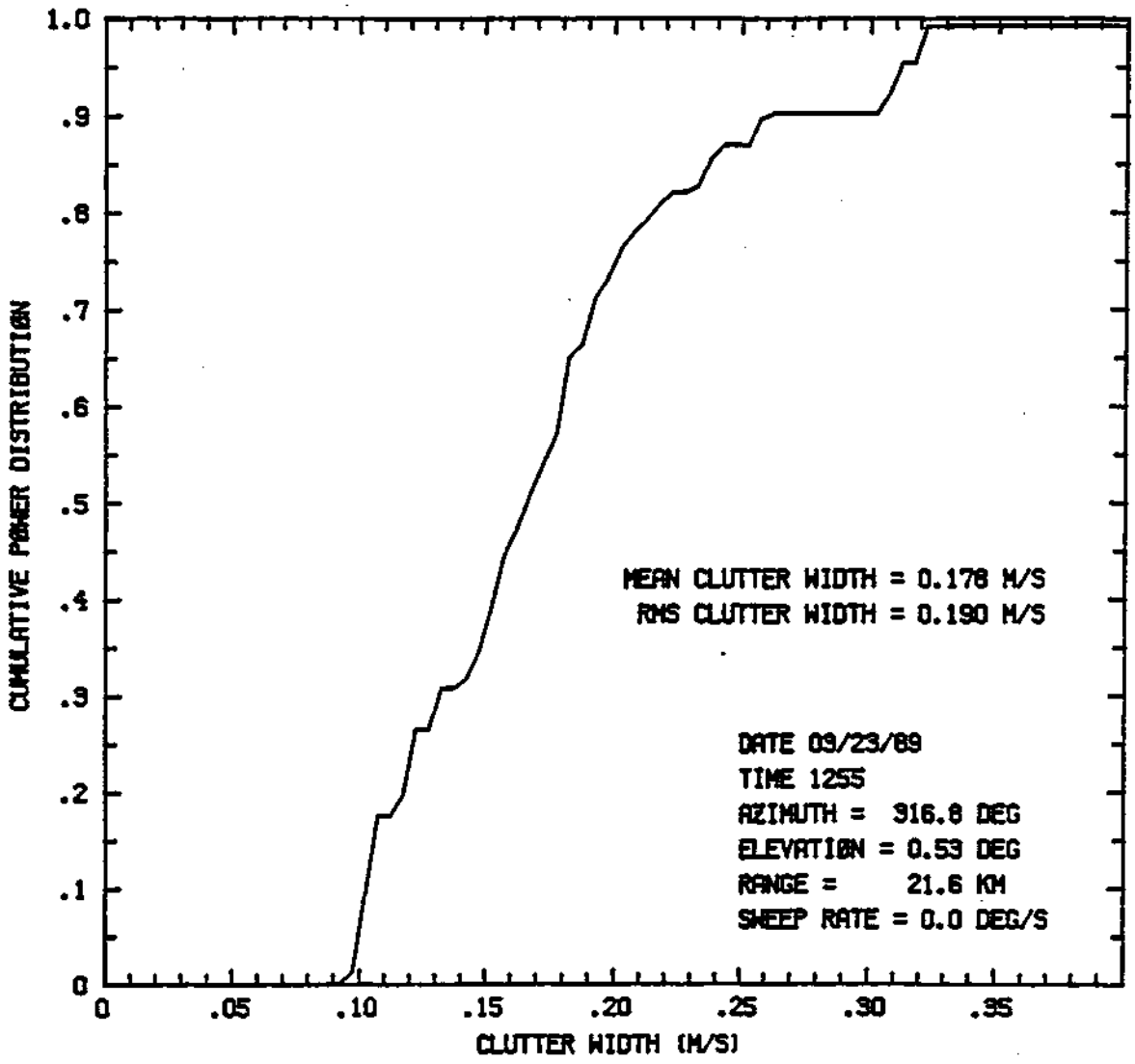


Figure 3.14. Ground-clutter spectral-width cumulative distribution for returns from the banks of the Sangamon river.

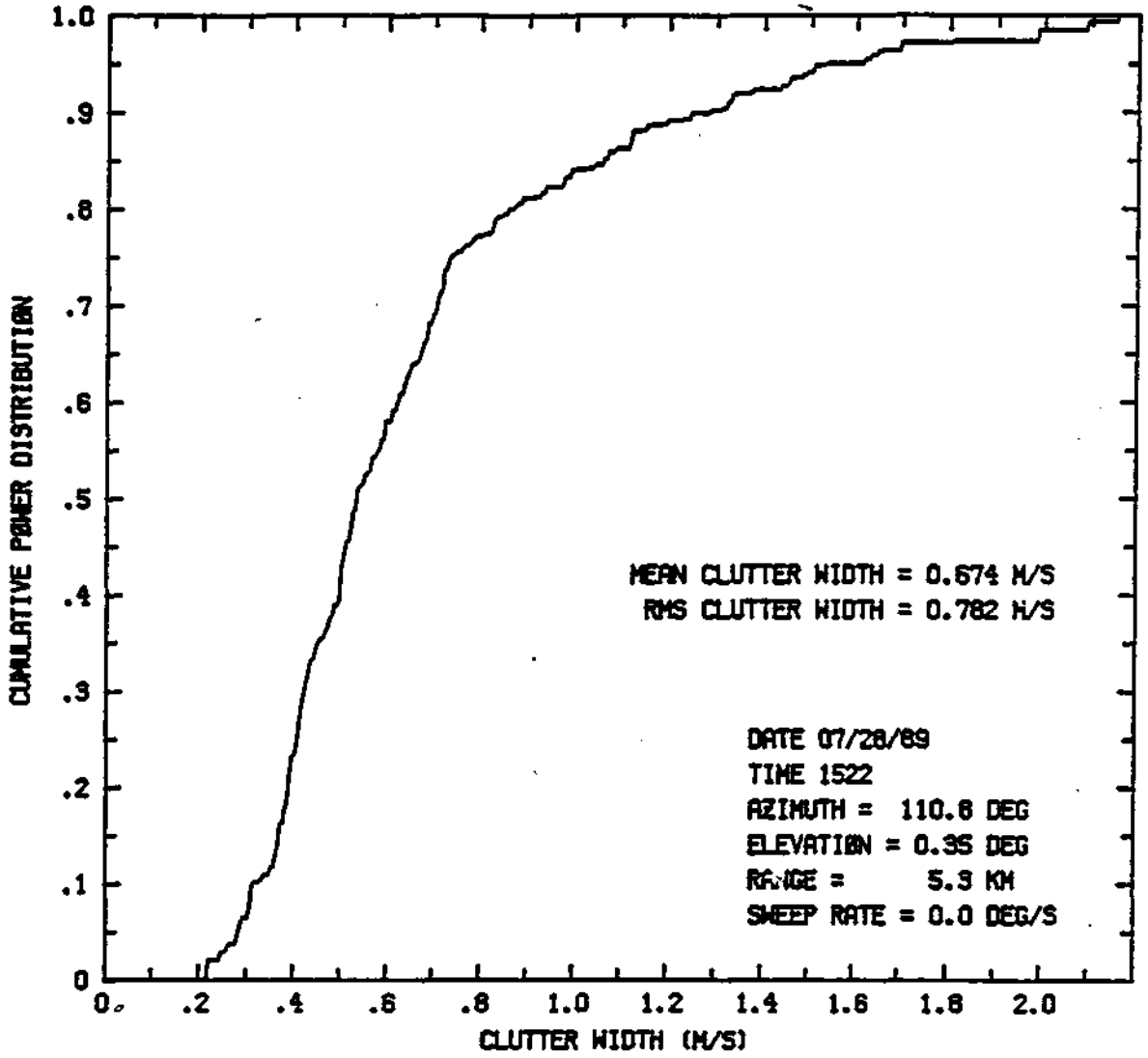


Figure 3.15. Ground-clutter spectral-width cumulative distribution for returns from cultivated farmland.

The area surrounding the WAND tower is nearly void of ground vegetation in the early spring. Figure 3.14 shows the clutter width distribution for ground returns from the banks of the Sangamon river. This is one of the few areas surrounding the radar that is forested. When the data were collected in late March, the trees were just budding. The 95 percentile of this distribution is 0.31 m/s. Figure 3.15 shows the width distribution for ground returns from cultivated farmland. These returns are thought to be from mainly corn and soybean fields and any hedgerows in the area. This is the only ground clutter found that had appreciably wide widths. The 95 percentile occurs at 1.6 m/s.

The above comparisons of clutter-width distributions dealt only with clutter type. Study of further dependencies were not met with much success. When considering clutter width as a function of antenna elevation, it was found that clutter width increased slightly as the elevation angle was increased. This contradicts the author's intuitive feeling that as the elevation increases, fewer distributed targets would be seen, leaving mainly tall rigid targets to contribute to the echo. This would tend to decrease clutter widths with elevation. The problem in drawing conclusions from these experimental results is that as the elevation angle increased, clutter echo power decreased, resulting in decreased signal-to-noise ratios, which, in turn, decreased the FFT width-estimator's accuracy.

In attempting to determine the effects of wind conditions on clutter width, no success was achieved. A

major difficulty is that only surface winds affect the motion of ground targets. Unless rain, dust or other particles are floating in the air close to the surface, no echo exists that can be used to determine wind velocity. A second problem is that even if the radar picks up an echo showing velocity, it is only the component of the velocity along the line of sight of the radar. This may or may not be sufficient. Consider the leaves on a tree. The blowing wind makes the leaves shimmer and shake, causing the resolution volume to appear more turbulent. This translates into a broadening of the resolution volume's power spectrum. Intuitively, the leaves would seem to shake in a similar way no matter which direction the wind blows. Thus, both speed and direction of the wind must be examined. Neither one of these parameters can be determined with a single Doppler radar. On-site wind sensors or dual Doppler radar measurements would be needed. Either one of these is beyond the scope of this study. Groginsky and Glover [8] have proposed that the broadening of the ground clutter's spectral width from wind speed is

$$\sigma_v^2 = 8\times10^{-3}v^{1.2} \text{ m/s}. \qquad (3.12)$$

The data used for deriving this estimator were taken from heavily wooded areas in Massachusetts. Thus, this is really a worst-case situation.

The problem with wind speed effects on clutter width also caused tests of rain effects on clutter width to be inconclusive. The inability to hold wind conditions nearly

constant for a patch of clutter before and after rain led to results that had no validity.

One final clutter width dependency was considered. Fortunately, much more success was found here. Figure 3.16 shows the broadening effects of antenna scan speed on the average clutter width. The widths were calculated for a range ring at about 7 km and an elevation angle of less than half a degree. The data were approximated with the line

$$\sigma = m\alpha + b. \qquad (3.13)$$

Using a least-squares fit, m was found to be 0.01596 m/degree and b was found to be 0.33 m/s. By subtracting off the y intercept, only the bias is left. This line is plotted in the lower part of Fig. 3.16 along with the value predicted by (2.19). The minor difference between the two can be attributed to the real antenna's non-Gaussian pattern. To generalize this result for different pulse repetition times, the slope of this line can be converted to a normalized velocity value by dividing by the unambiguous velocity of the data. The unambiguous velocity for these data was 26.4 m/s. Thus, the slope of the line in normalized units is 605×10^{-6} s/degree.

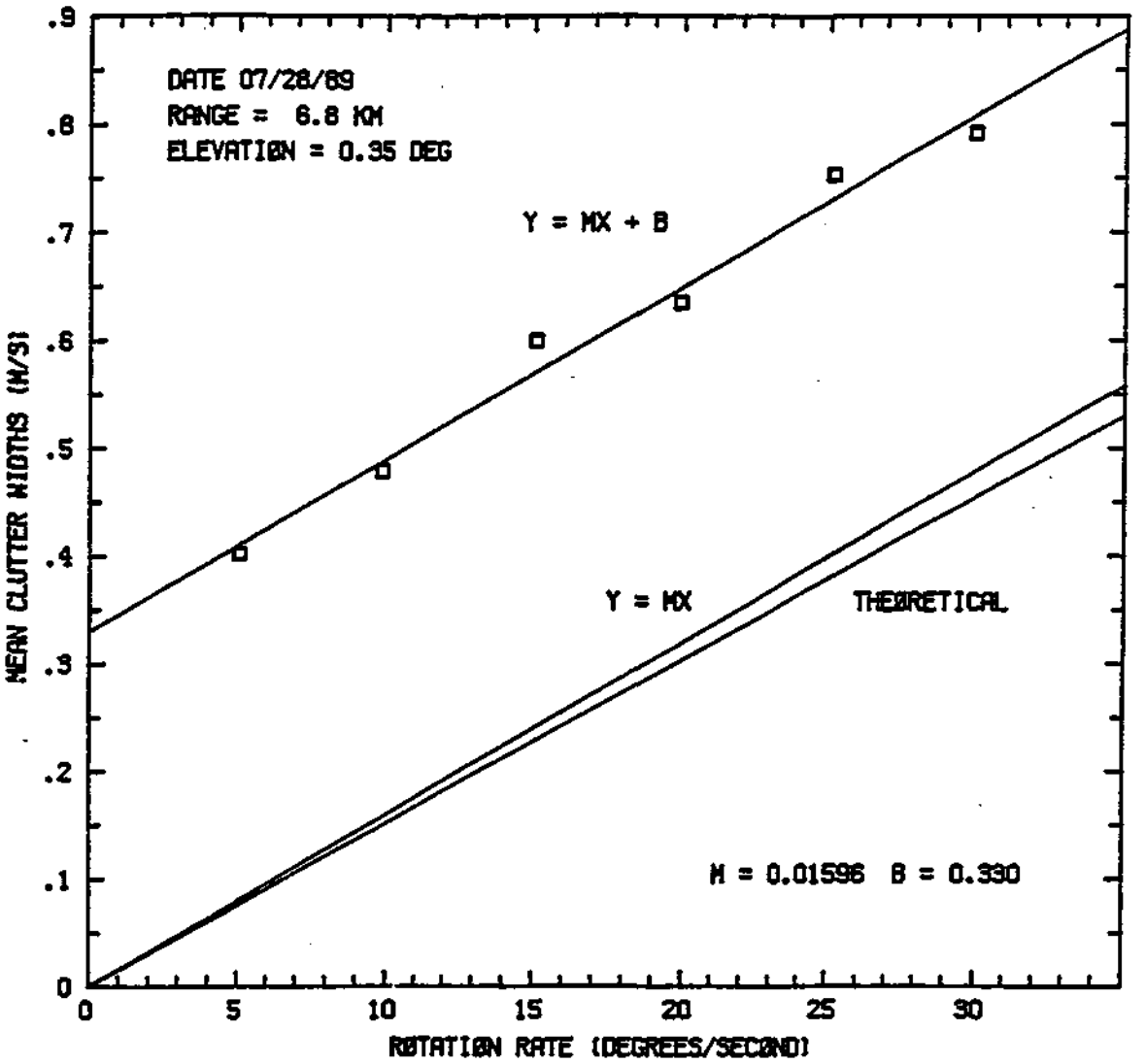


Figure 3.16. Effects of antenna scanning on ground clutter width.

CHAPTER 4

GROUND CLUTTER FILTERS

4.1 Linear Filters as a Solution to Ground Clutter

When the results of the last two chapters are compared, we see that the main difference between ground clutter and weather echoes lies in their mean velocities. Another useful difference is that weather echoes tend to have broader spectra [9]. Traditionally, three methods have been used to deal with the ground clutter problem. These are clutter maps, linear dc notch filters, and maximum-entropy-clutter prediction.

The M.I.T. weather radar in Cambridge, Massachusetts, is an example of one that makes use of clutter maps. By taking many scans on a clear day, average clutter power calculations are made for every range gate and azimuth cell. Using these average powers, clutter maps can be constructed for each elevation angle to be used. Ground clutter cancellation is achieved by subtracting these powers from the real-time echo power estimates. The most obvious problem with this method is that it only helps the weather reflectivity estimates. It does nothing for the velocity or spectral width estimators. Another problem is that new maps are needed whenever data block lengths used to form estimates are changed. Changing antenna rotation rates also requires new maps. Both of these things may be done

frequently when collecting data at the CHILL. To accommodate the wide variety of situations, the CHILL is used for, a prohibitively large number of clutter maps would be needed. This method is really only feasible for radars that use a fixed scan sequence to obtain only reflectivity measurements.

Before discussing the pros and cons of linear dc notch filters and maximum-entropy clutter prediction, the computational constraints of the CHILL processor must be addressed. As is explained in Appendix II, signal processing of the CHILL is done in real time by the SP20 dedicated signal processor. Two boards in this processor are allocated for ground clutter cancellation. Each board has a capability of ten-million floating-point multiplications and the same number of floating-point additions every second. Using 1 μs range gating, the above capabilities translate into twenty real multiplications and additions every complex echo sample.

The maximum-entropy clutter prediction method as described by Haykin et al. [10] involves modeling ground clutter as an autoregressive process using an equivalent lattice model of a prediction error filter. The computational constraints above are such that a complex echo sample could be pushed through a three-step predictor but not enough multiplications and additions would be left to update the adaptive filter coefficients. A smaller predictor cannot be used since a three-step predictor is about the minimum size needed to effectively model ground clutter. Since the computational constraints rule this method out, we

are left with linear dc notch filters to solve the ground clutter problem.

When analyzing the frequency responses of linear clutter filters, three errors must be taken into consideration. The first of these errors is from the clutter residue, which is the amount of clutter signal left after filtering. As Section 3.1 outlined, for accurate pulse-pair estimates, clutter residue power must be 15 to 20 dB below weather echo power. Thus, clutter filters need deep notches. The second error to consider is that introduced by the amount of weather signal discarded by the filter. This can be minimized by making the filter notches as narrow as possible. The last error has to do with background noise and changes in signal-to-noise ratios after filtering. Weighting each of these three factors against one another is a non-trivial problem. If weather echoes have wider spectra than ground clutter echoes, then even when the weather echoes have small mean velocities, signal-to-clutter ratios will be improved by filtering. At the same time, filtering these low velocity weather echoes decreases signal-to-noise ratios. This degrades pulse-pair parameter estimator performance. This last effect can be minimized by filtering only when significant ground clutter is present.

Many types of filters have been used to deal with the ground clutter problem. One of the earliest was the pulse canceller. A pulse canceller is a finite impulse response filter with the coefficient at the output of the i^{th} delay element given by

$$a_i = (-1)^i n!/(n-i)!i!, \quad i = 0,1,2,3,\ldots n. \qquad (4.1)$$

The attraction of these filters is that their integer coefficients can be easily implemented by constructing analog delay lines. This simplicity of implementation led to these filters being the first ground clutter cancellation scheme used in real time. Groginsky and Glover have analyzed them for use with the pulse-pair algorithm and found them to be severely lacking in performance [8]. The problems with these filters are two-fold. The first is that their frequency responses are quite poor. Figure 4.1 shows the responses for first- and second-order pulse cancellers. The second problem with these filters is that their notch widths are not adjustable. If the pulse repetition time is halved, the notch width doubles in velocity. Besides not being adjustable, the notch widths are much too wide. The half-power passband cutoff velocities are around half the unambiguous velocity.

A general finite impulse response filter gives a much better frequency response than the pulse canceller. The computational constraints at the CHILL, previously mentioned, restrict the length of this kind of filter to nine delays and ten real coefficients. Using the Remez exchange algorithm, Evans designs an equiripple filter that performs quite well ([5];pp.7.1-7.23). The filter has 1 dB total passband ripple and -50 dB stopband gain. The problem with this filter is that it is 39 taps long. Even if the unnecessary linear phase constraint were excluded, the filter that meets this filter's magnitude characteristics

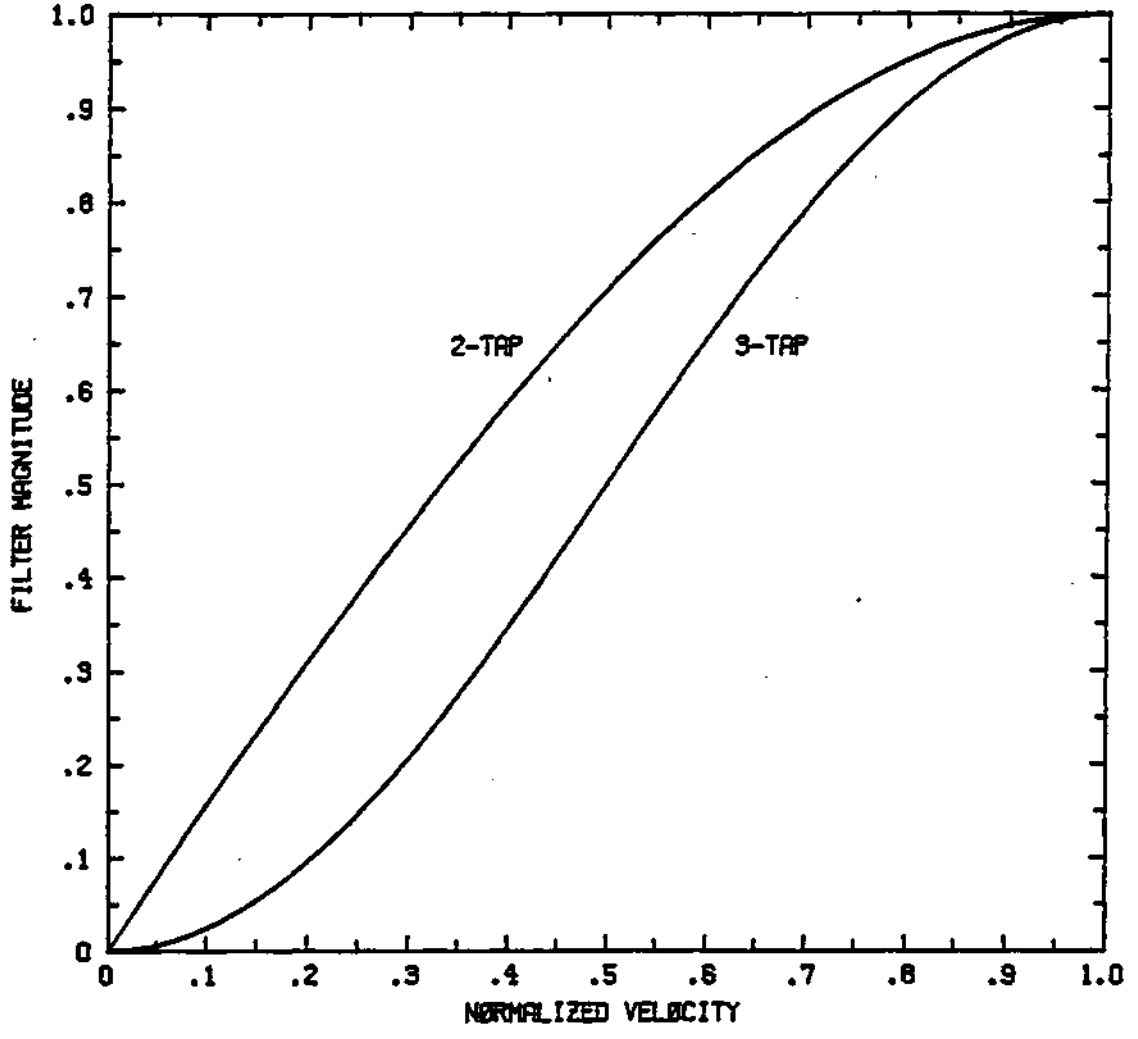


Figure 4.1. Frequency responses for first- and second-order pulse cancellers.

would be twice as long as the real-time computational constraints that the CHILL processor would allow.

The only way to produce a deep and narrow notch in a computationally viable manner is to use an infinite impulse response, also called recursive filter. A recursive elliptic filter needs only three poles to match the magnitude characteristics of the 39-tap finite impulse response filter mentioned above [8]. The resistance to using recursive filters has traditionally stemmed from the problem of initializing them to suppress their transient responses. Since the CHILL processes samples in a continuous fashion, this is not a real concern. It is a problem, however, when block processing is used ([31];pp.16-41). In any event, initialization is discussed in Section 4.4.

4.2 Design of Elliptic Recursive Filters

Elliptic equiripple recursive filters were chosen for implementation on the CHILL because of their optimal magnitude characteristics. Table 4.1 shows the computations needed to process a single complex sample through cascade realizations of elliptic filters with real coefficients. The computational constraints of the SP20 processor limit the filter to an order of five. In practice, though, a five-pole filter could not be implemented due to local bus conflicts in the processor. Both three- and four-pole filters were successfully implemented.

Table 4.1. Computations needed to process a single complex sample through cascade realizations of elliptic filters.

No. of Poles	Real Multiplications	Real Additions
3	10	12
4	14	16
5	16	20
6	20	24

The design of these digital filters is based on low-pass analog prototypes. The transfer functions of three- and four-pole analog low-pasd elliptic filters are

$$H_{LP}(s) = A(s^2+a_1)/(s+b_1)(s^2+b_2s+b_3) \qquad (4.2a)$$

and

$$H_{LP}(s) = A(s^2+a_1)(s^2+a_2)/(s^2+b_1s+b_2)(s^2+b_3s+b_4), \qquad (4.2b)$$

respectively. Tables 4.2 and 4.3 list the coefficients of these transfer functions for several different stopband gains and passband ripples [13]. These filters are normalized with the geometric mean of the passband cutoff and the stopband cutoff equal to unity:

$$\sqrt{\omega_s\omega_p} = 1. \qquad (4.3)$$

The parameter ω_r represents the sharpness of the transition region. It is defined as the ratio

$$\omega_r = \omega_s/\omega_p. \qquad (4.4)$$

If the desired digital filter cutoff frequencies are expressed in terms of a velocity variable normalized to unity, the analog prewarp specifications are

Table 4.2. Coefficients for three-pole analog prototypes.

Passband Ripple = 1/2 dB

Stopband Gain (dB)	-40	-50	-60
a_1	3.55131	5.16223	7.54288
b_1	0.417693	0.353804	0.307274
b_2	0.352622	0.306184	0.258673
b_3	0.422730	0.293107	0.201322
A	0.0476460	0.0184313	0.00709340
ω_r	2.71147	3.90430	5.67937

Passband Ripple = 1 dB

Stopband Gain (dB)	-40	-50	-60
a_1	3.148896	4.56481	6.66138
b_1	0.349732	0.295023	0.255373
b_2	0.292413	0.416088	0.216668
b_3	0.255657	0.288843	0.198506
A	0.0445207	0.0172697	0.00665490
ω_r	2.41619	3.46061	5.02121

Table 4.3 . Coefficients for four-pole analog prototypes.

Passband Ripple = 1/2 dB

Stopband Gain (dB)	-40	-50	-60
a_1	9.15630	12.64431	17.20286
a_2	1.84784	2.37957	3.11024
b_1	0.710817	0.613021	0.529149
b_2	0.274972	0.196244	0.143036
b_3	0.213089	0.211829	0.197979
b_4	0.651784	0.513565	0.396229
A	0.0100001	0.00316227	0.000999995
ω_r	2.71147	2.06924	2.68325

Passband Ripple = 1 dB

Stopband Gain (dB)	-40	-50	-60
a_1	8.20047	11.40195	15.57508
a_2	1.70946	2.18639	2.84628
b_1	0.591849	0.510902	0.441175
b_2	0.238719	0.169815	0.123525
b_3	0.171046	0.173001	0.163210
b_4	0.658897	0.520870	0.402671
A	0.0100002	0.00316227	0.000999992
ω_r	1.51549	1.90819	2.46079

$$\omega_s = 2\tan[v_s\pi/2] \tag{4.5a}$$

and

$$\omega_p = 2\tan[v_p\pi/2], \tag{4.5b}$$

where v_s is the desired stopband cutoff velocity and v_p is the desired passband cutoff velocity. These parameters are not independent. They are related by

$$\omega_u = \omega_s/\sqrt{\omega_r} = \omega_p\sqrt{\omega_r}. \tag{4.6}$$

Taking the transfer functions (4.2) and applying analog low-pass to high-pass transformation

$$s \rightarrow \omega_u/s, \tag{4.7}$$

followed by the analog to digital bilinear transformation

$$s \rightarrow 2(z-1)/(z+1), \tag{4.8}$$

result in the desired digital filters. The transfer functions of these digital filters are

$$H_3(z) = g\frac{(1-k_1z^{-1}+z^{-2})(1-z^{-1})}{(1-k_2z^{-1}+k_3z^{-2})(1-k_4z^{-1})} \tag{4.9a}$$

and

$$H_4(z) = g\frac{(1-k_1z^{-1}+z^{-2})(1-k_2z^{-1}+z^{-2})}{(1-k_3z^{-1}+k_4z^{-2})(1-k_5z^{-1}+k_6z^{-2})}, \tag{4.9b}$$

for the three-pole and four-pole filters, respectively. The coefficients of the transfer function (4.9a) are

$$g = A(a_1+x^2)/(b_1+x)(b_3+b_2x+x^2), \quad (4.10a)$$

$$k_1 = 2(a_1-x^2)/(a_1+x^2), \quad (4.10b)$$

$$k_2 = 2(b_3-x^2)/(b_3+b_2x+x^2), \quad (4.10c)$$

$$k_3 = (b_3-b_2x+x^2)/(b_3+b_2x+x^2), \quad (4.10d)$$

$$k_4 = (b_1-x)/(b_1+x), \quad (4.10e)$$

where $x = \omega_u/2$. The coefficients of the four-pole transfer function (4.9b) are

$$g = A(a_1+x^2)(a_2+x^2)/(b_2+b_1x+x^2)(b_4+b_3x+x^2), \quad (4.11a)$$

$$k_1 = 2(a_1-x^2)/(a_1-x^2), \quad (4.11b)$$

$$k_2 = 2(a_2-x^2)/(a_2-x^2), \quad (4.11c)$$

$$k_3 = 2(b_2-x^2)/(b_2+b_1x+x^2), \quad (4.11d)$$

$$k_4 = (b_2-b_1x+x^2)/(b_2+b_1x+x^2), \quad (4.11e)$$

$$k_5 = 2(b_4-x^2)/(b_4+b_3x+x^2), \quad (4.11f)$$

$$k_6 = (b_4-b_3x+x^2)/(b_4+b_3x+x^2), \quad (4.11g)$$

where $x = \omega_u/2$ as before.

Figure 4.2 diagrams the cascade realizations of these filters [17]. Figure 4.3 shows the frequency responses of some sample filters designed by this method. These filters used analog prototypes with one dB passband ripple and -50 dB stopband gain. The passband cutoff velocities are 0.04, corresponding to 1 m/s if the unambiguous velocity is 25 m/s. Figure 4.4 is an expanded view of the transition bands of the filters in Fig. 4.3. The extra pole of the four-pole filter greatly reduces the transition width from passband to stopband. This results in more signal in the notch being suppressed.

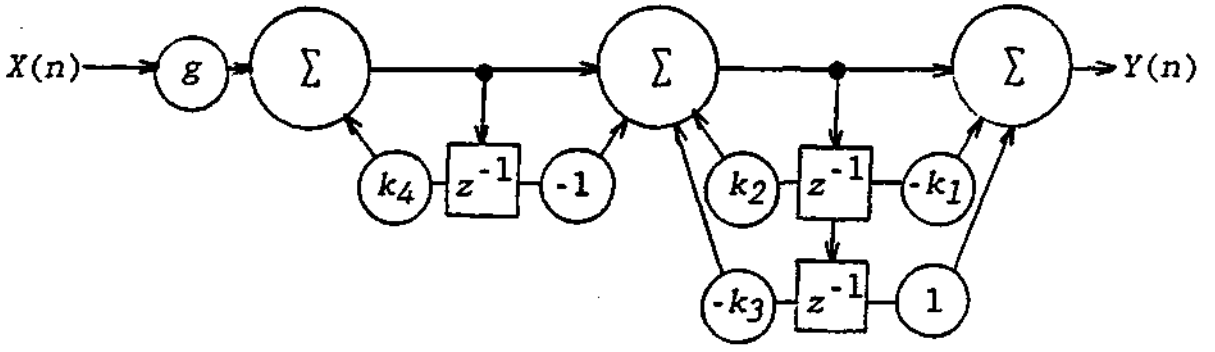


Figure 4.2(a). Cascade realization of the three-pole elliptic filter (4.9a).

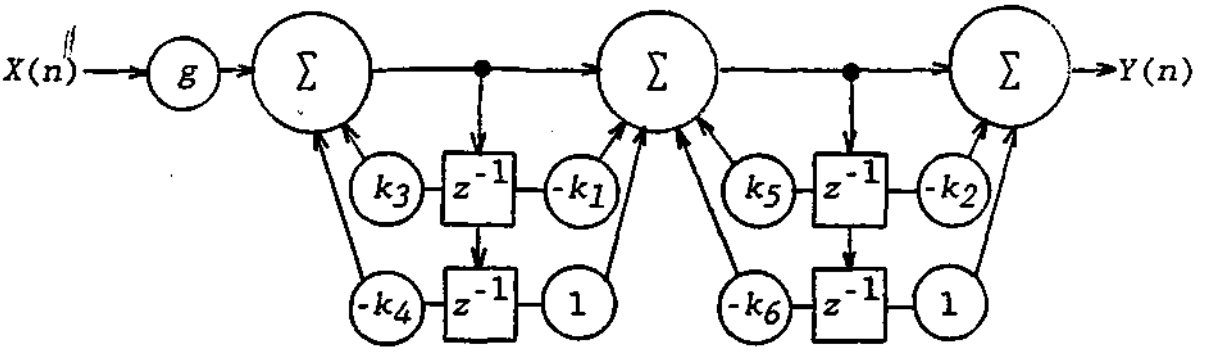
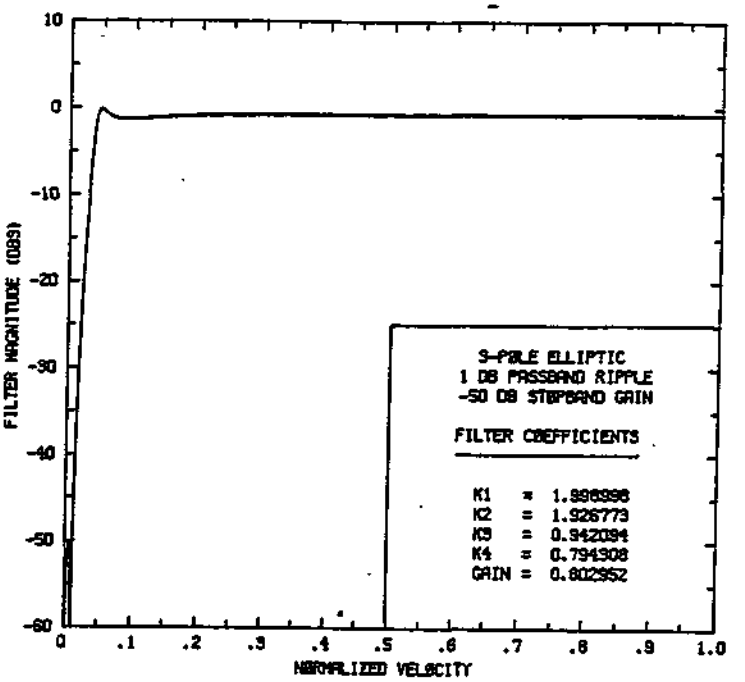


Figure 4.2(b). Cascade realization of the four-pole elliptic filter (4.9b).

(a)



(b)

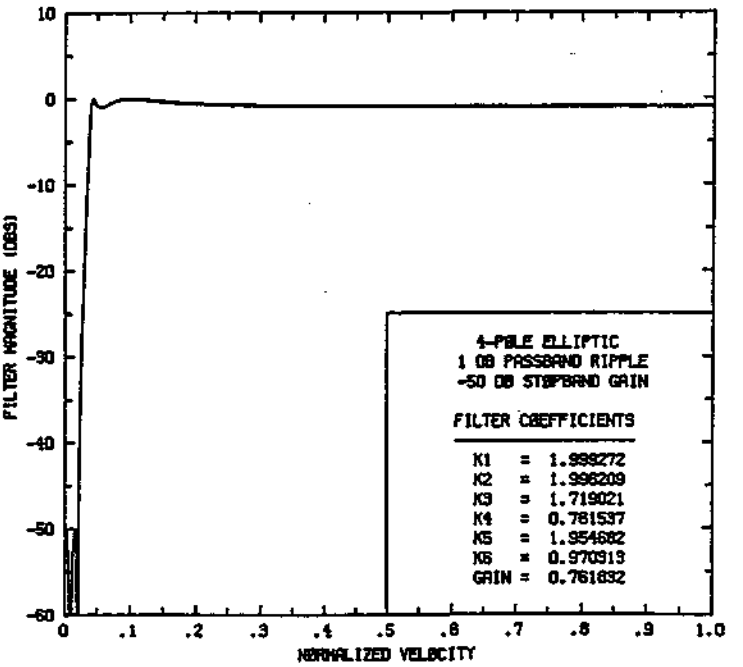


Figure 4.3. (a) Three-pole elliptic filter fitting the transfer function (4.9a). (b) Four-pole elliptic filter fitting the transfer function (4.9b).

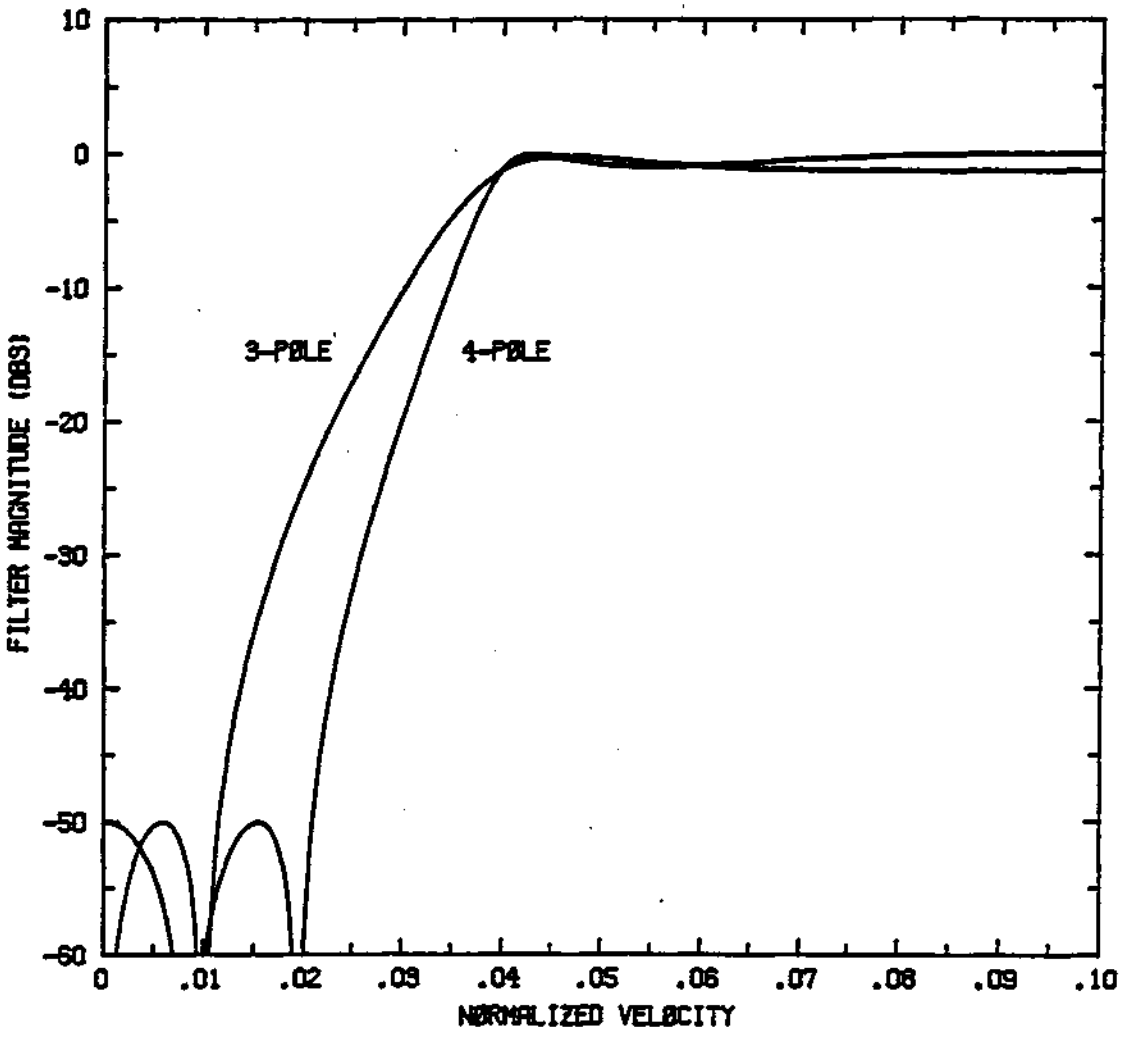


Figure 4.4. Expanded view of transition bands of the filters in Fig. 4.3.

4.3 Performance of Recursive Elliptic Filters

From Section 3.1 we know that signal-to-clutter power ratios of 15 to 20 dB are needed for reliable pulse-pair parameter estimation. The cumulative clutter power distributions of Fig. 3.9 show low elevation clutter powers in the 20 to 50 dBZ range. For reliable observation of weak weather echoes in these environments, clutter suppression on the order of 40 to 70 dB is needed. This is achieved by using stopband gains of -40 to -70 dB. In addition to the filter stopband gain being small, the filter stopband must be wide enough so that the clutter spectral tails are below the level of desired suppression at the filter passband edge. Using the gaussian spectral model of ground clutter, for D decibels of clutter suppression, the filter passband velocity v_p should be such that

$$-D = 10\log[\exp[-v_p 2/2\sigma_c{}^2]]. \qquad (4.12)$$

Solving (4.12) for the filter passband velocity, v_p, gives

$$v_p = \sigma_c\sqrt{D}\ln[10]/5. \qquad (4.13)$$

Figure 4.5 shows the necessary ratios of filter notch width over clutter spectral width for different levels of clutter suppression. For example, 50 dB of clutter suppression require the filter's passband velocity to be 4.8 times larger than the clutter's spectral width.

Filter performance in suppressing discrete clutter echoes can be tested by using the discrete clutter model of Section 3.2. The problem with using the antenna pattern of

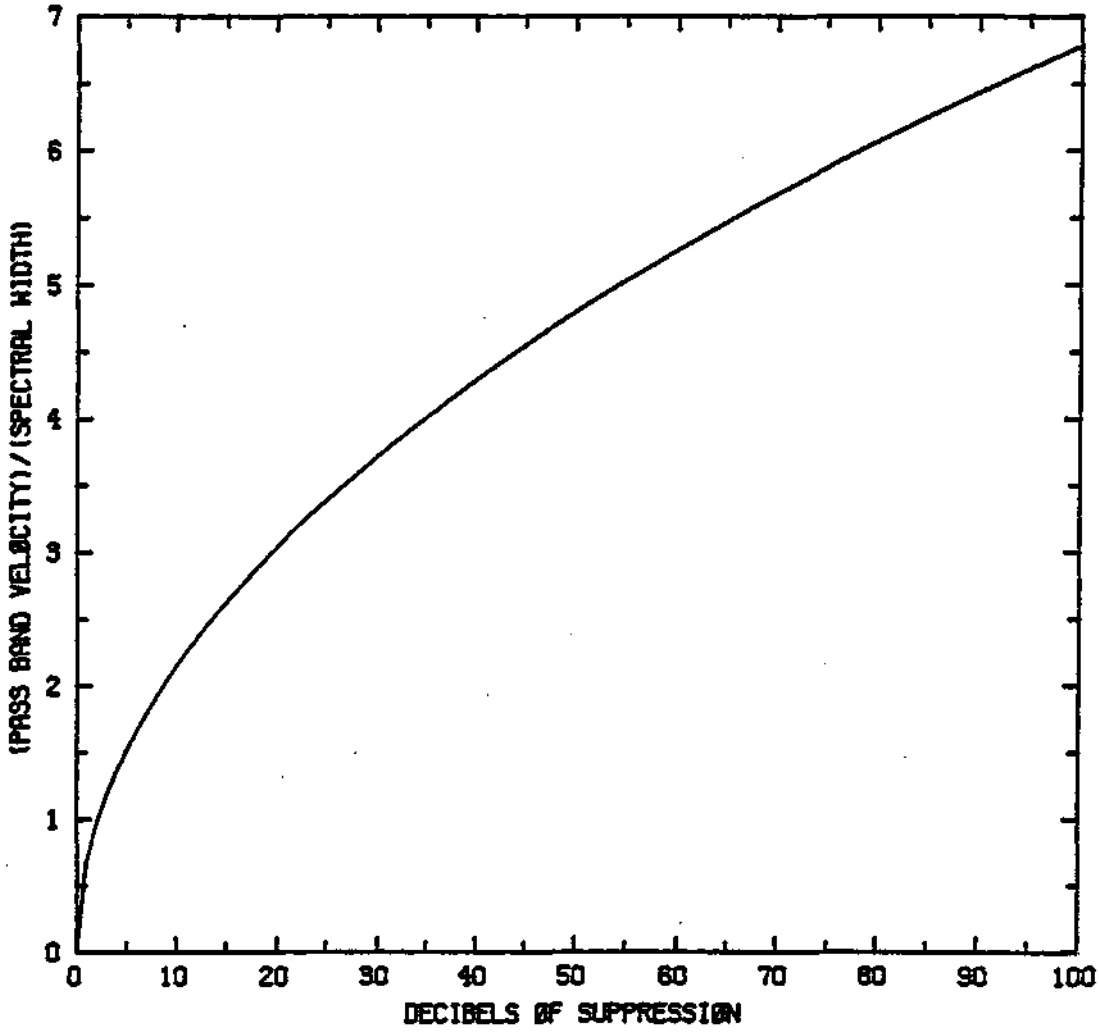
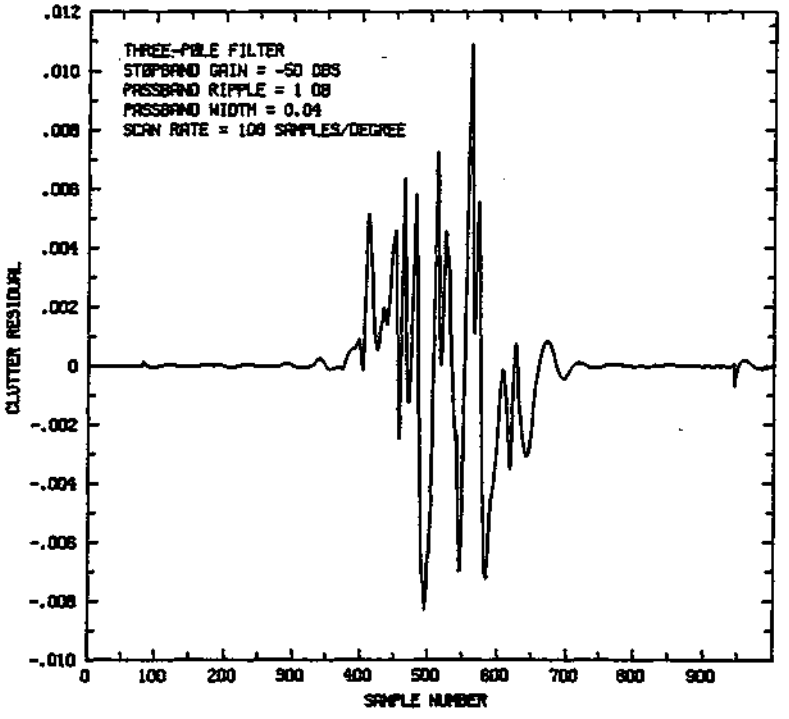


Figure 4.5. Ratio of filter notch width over clutter spectral width for different clutter suppression levels.

Fig. 3.8 as a source of simulated data is that it was constructed from samples of the antenna pattern only every 1/18 of a degree. To simulate realistic antenna rotation speeds, more samples than this were needed. This problem was solved by adding six samples between each existing data point using a linear interpolation of the logarithmic scale data. After taking the anti-logarithm of the resulting data, the result was smoothed by passing it through a five-point smoothing filter. To insure no biases were introduced by this process, the Fourier transforms of the raw data set and the extended data set were compared and found to have related characteristics. The side lobes in both transforms had the same shapes and amplitudes. The extended data set's transform was just compressed on the velocity axis. The resulting extended data set has 108 samples per degree. If the pulse repetition time is one millisecond, using all 108 samples per degree corresponds to an antenna rotation rate of about 9.3 degrees per second. Using only every other sample, or 54 samples per degree, corresponds to an antenna rotation rate of 18.5 degrees per second.

Figures 4.6 and 4.7 compare the clutter remaining after the above data have passed through three- and four-pole elliptic filters. The filters have passband velocities of 0.04, stopband gains of -50 dB and total passband ripples of one dB. Figure 4.6 shows the results when the slower antenna rotation rate of 108 degrees per sample was used. The differences between the three- and four-pole performances is minimal. Both give suppressions of at least 40 dB. Figure 4.7 is the same as Fig. 4.6 except the faster

(a)



(b)

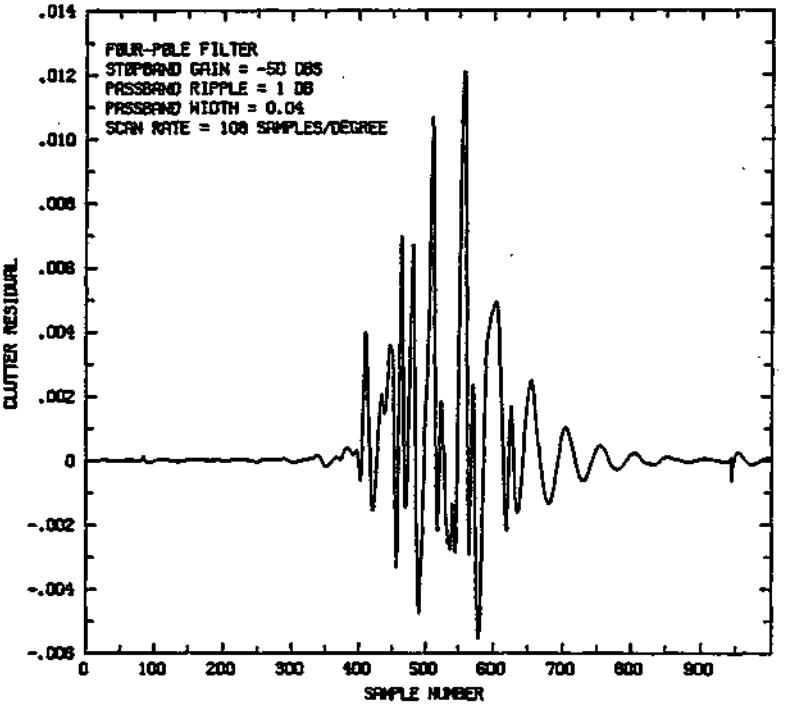
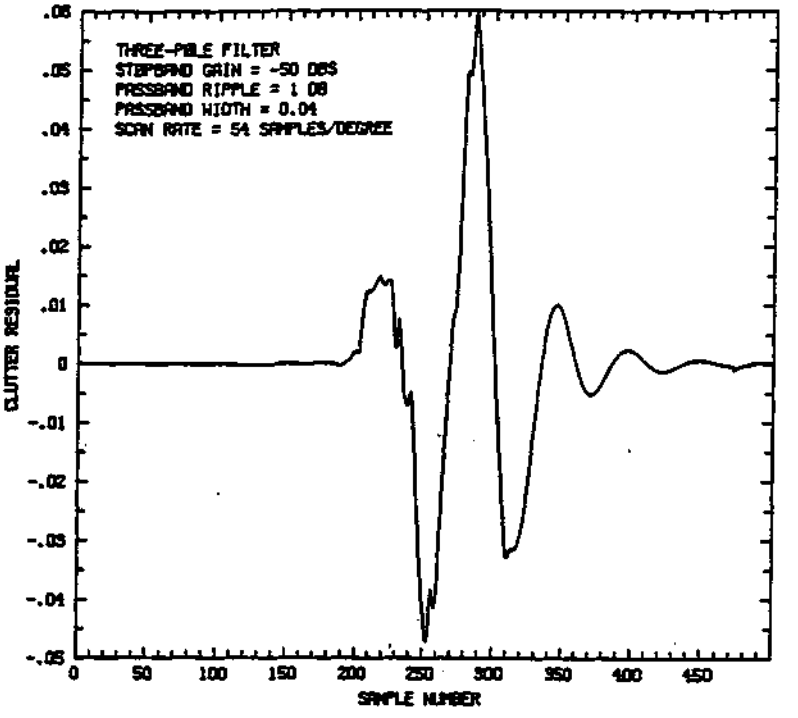


Figure 4.6. Discrete model clutter residue for slow antenna rotation rate through (a) Three-pole filter, (b) Four-pole filter.

(a)



(b)

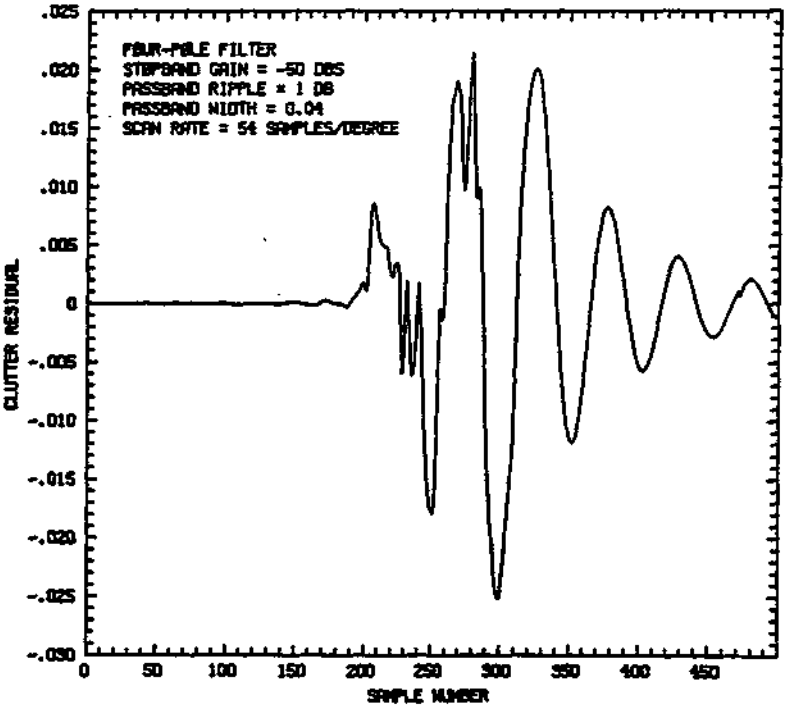


Figure 4.7. Discrete model clutter residue for fast antenna rotation rate through (a) Three-pole filter, (b) Four-pole filter.

antenna rotation rate of 54 samples per degree was used. Unlike the slower rotation rate, the differences between three- and four-pole performances are dramatic. The four-pole filter achieves a minimum clutter suppression of 32 dB, while the three pole achieves only 25 dB. The difference can be attributed to the fact that the filter widths are not quite large enough to achieve the desired suppression, i.e., the clutter spectrum is not 50 dB below its peak value at the velocity 0.04. This is where the superior notch rejection of the four-pole filter, as shown in Fig. 4.4, comes into play.

Figure 4.8 shows the effect that the filter width has on discrete clutter suppression. The clutter model is the same one used in Fig. 4.7. The filter is the same four-pole filter used in Fig. 4.7(b) except that the passband velocity changes. The passband velocities of the filters used in Figs. 4.8(a), (b), (c), and (d) are 0.04, 0.08, 0.12 and 0.16, respectively. The minimum clutter power suppressions are 32, 38, 43 and 41 dB, respectively. The minor changes in suppressions of the wider notch filters can be explained by the fact that most of the clutter power is present at velocities less than 0.1. The wider filters are really not rejecting much more clutter than the narrower filters.

Figure 4.9 is the same as Fig. 4.8 except that the filter stopband gains are varied instead of the filter notch widths. The passband velocity of these filters is 0.08. The stopband gains of Figs. 4.9(a), (b), (c), (d) and (e) are -30, -40, -50, -60 and -70, respectively. The minimal suppressions are 28, 36, 38, 39 and 40 dB, respectively. The

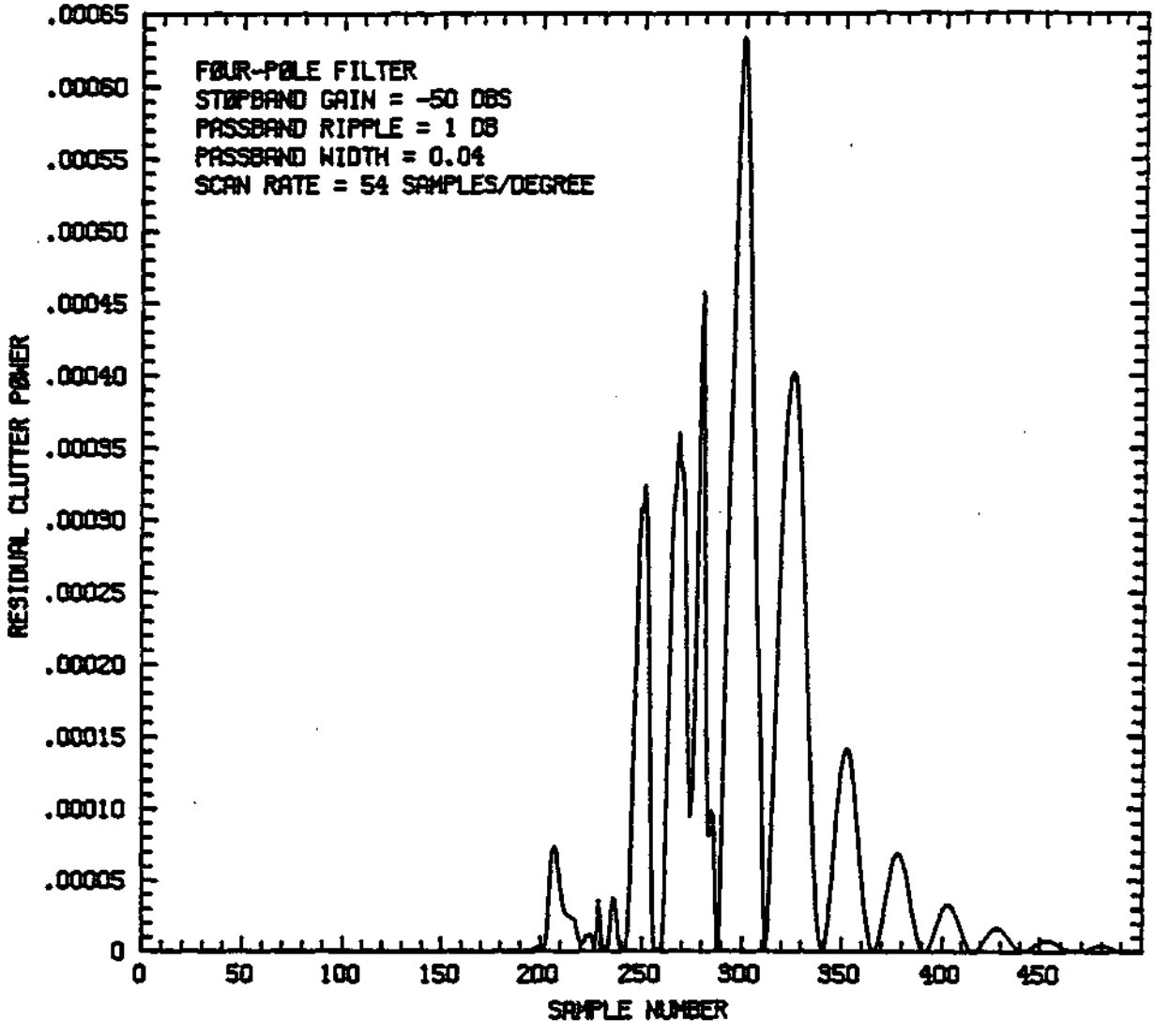


Figure 4.8(a). Power of clutter remaining after the discrete clutter model passes through a four-pole filter with a passband of 0.04.

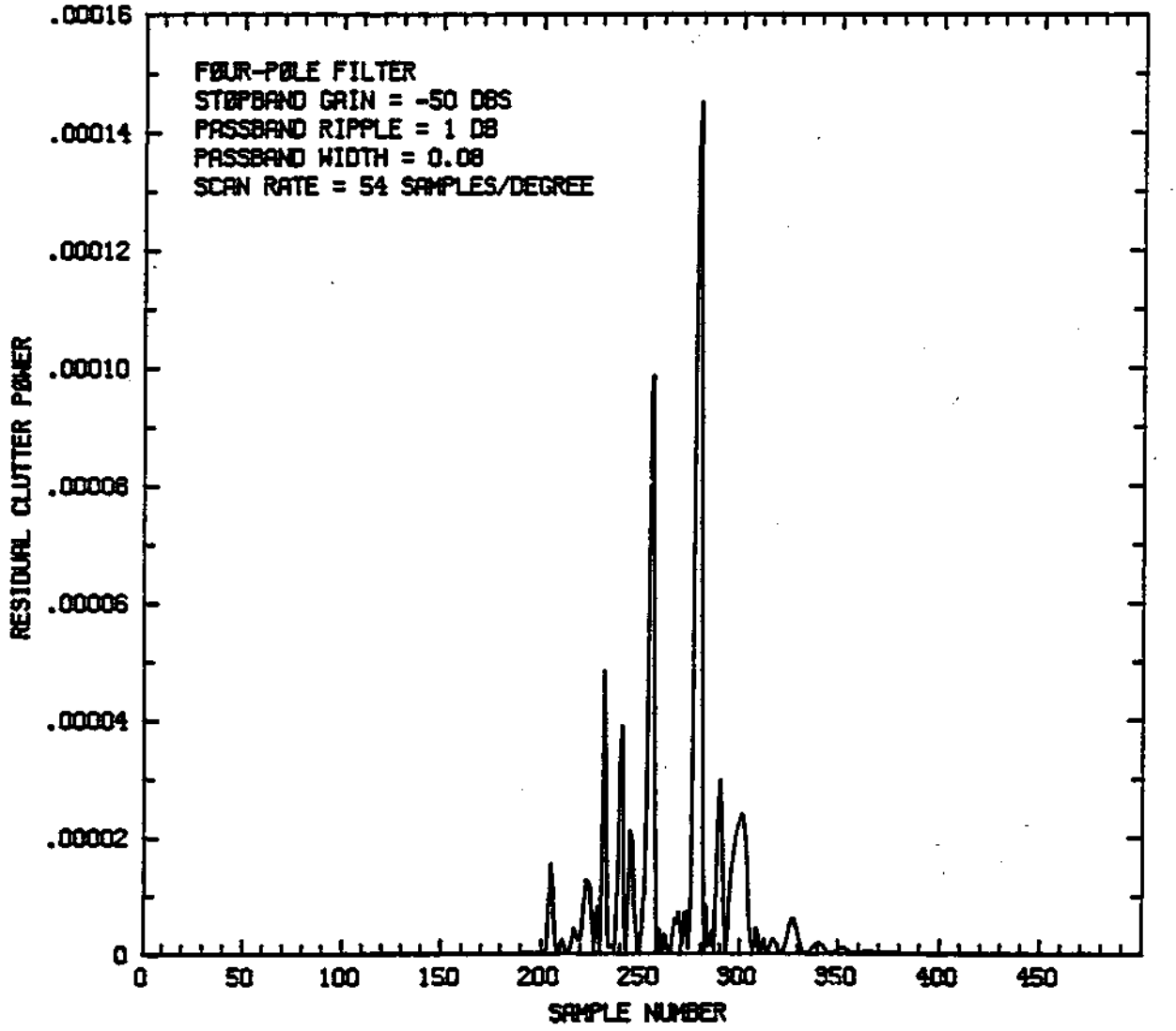


Figure 4.8(b). Same as Fig. 4.8(a) except passband is 0.08.

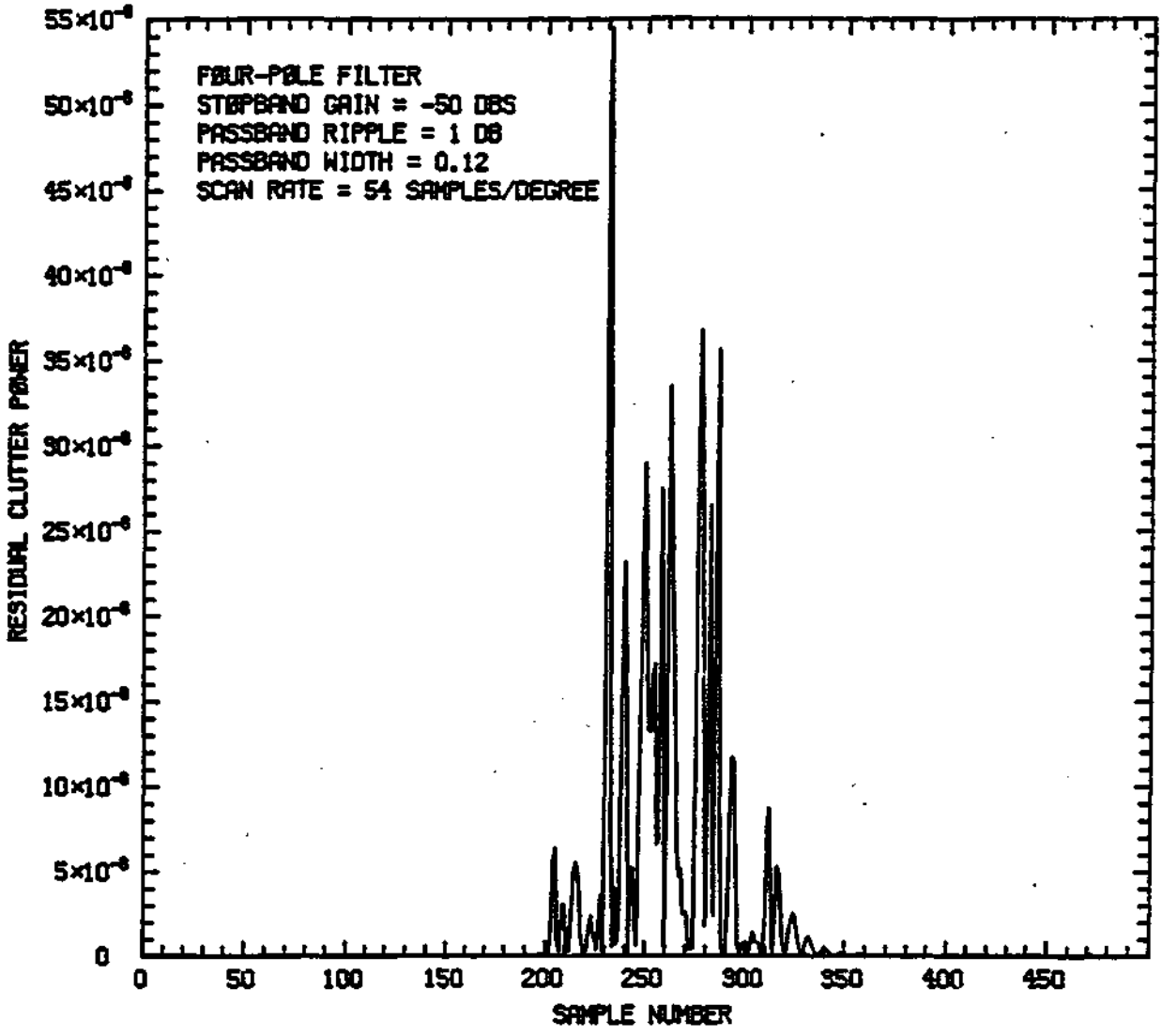


Figure 4.8(c). Same as Fig. 4.8(a) except passband is 0.12.

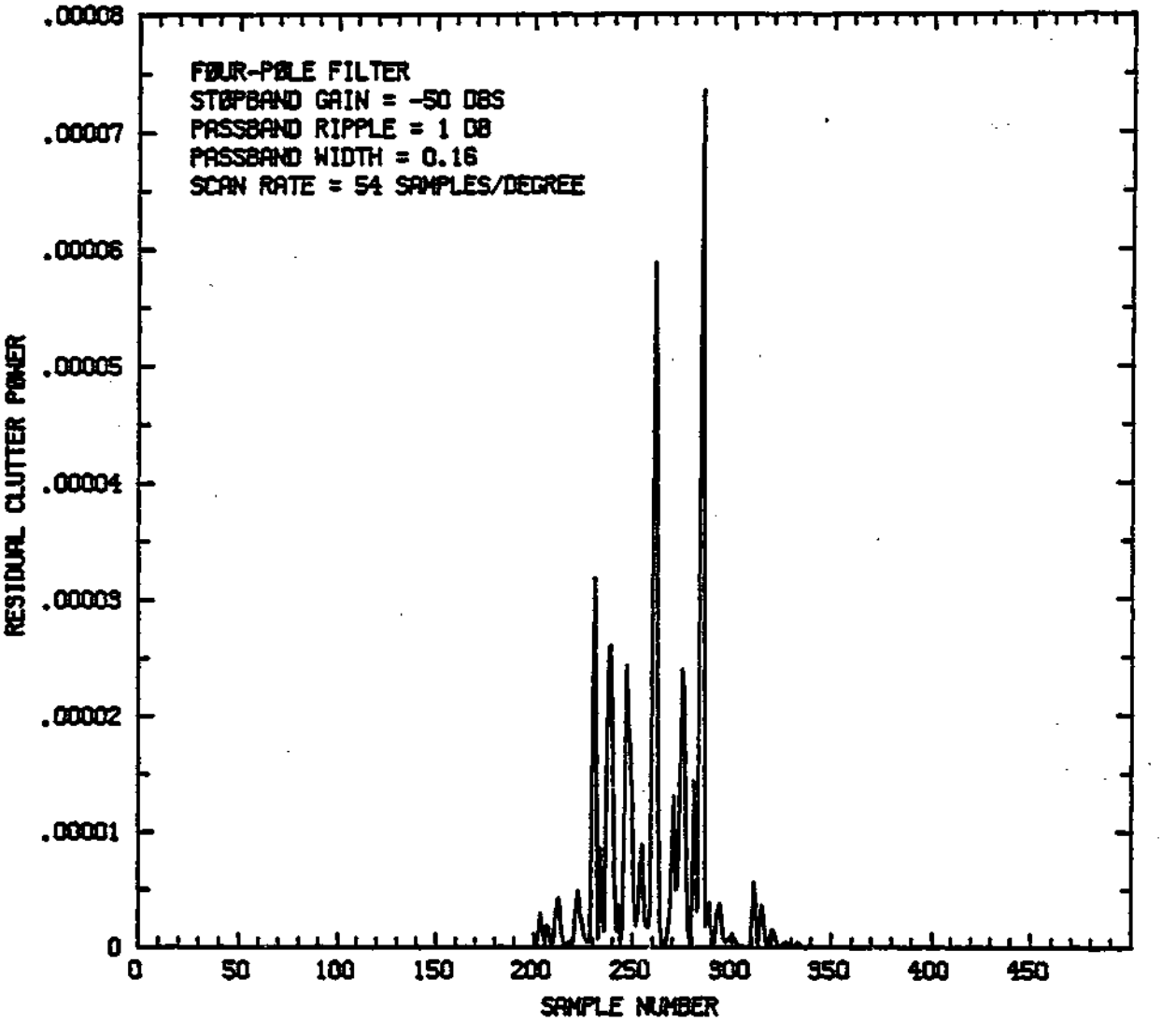


Figure 4.8(d). Same as Fig. 4.8(a) except passband is 0.16.

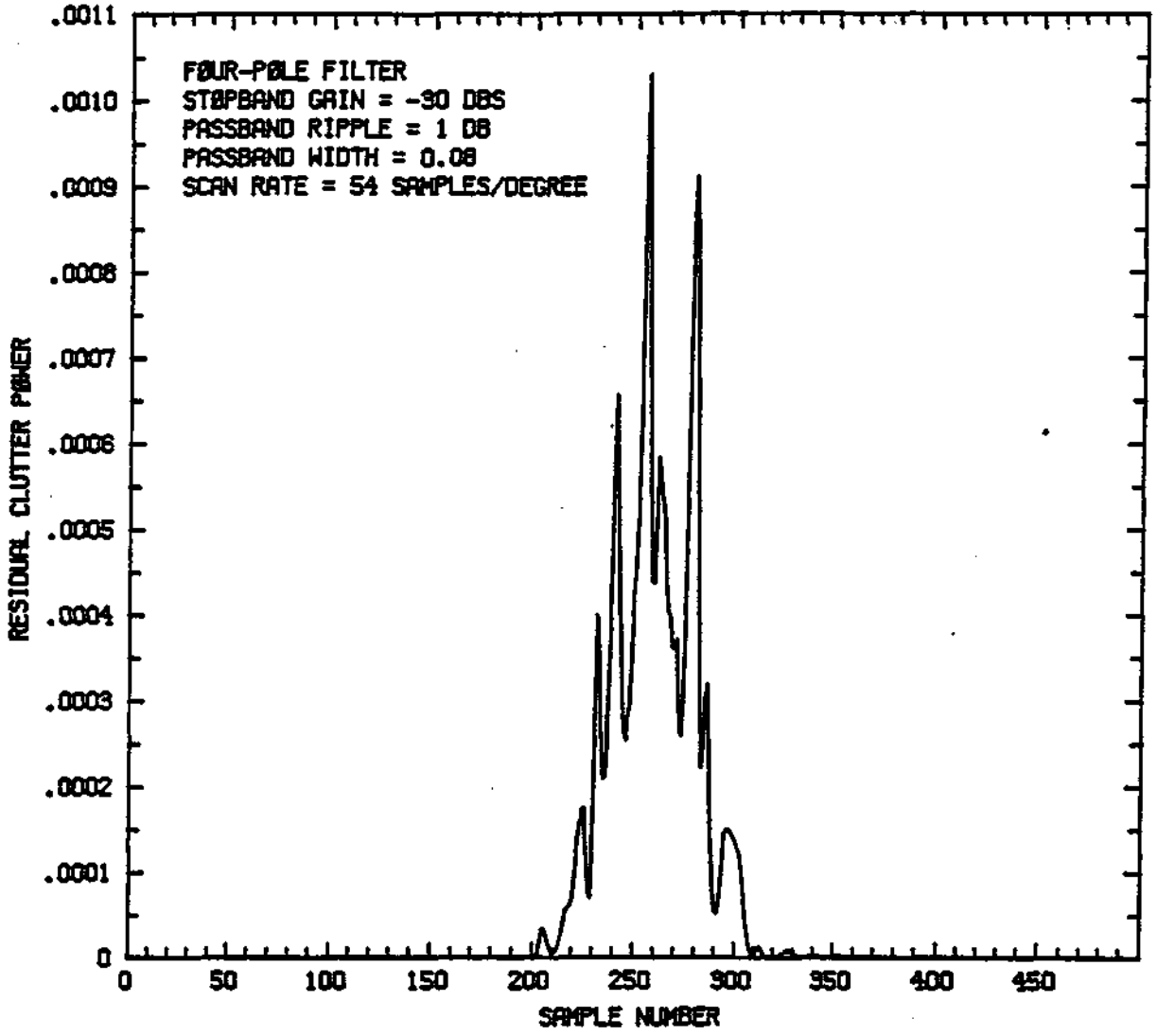


Figure 4.9(a). Same as Fig. 4.8(b) except stopband gain is -30 dB.

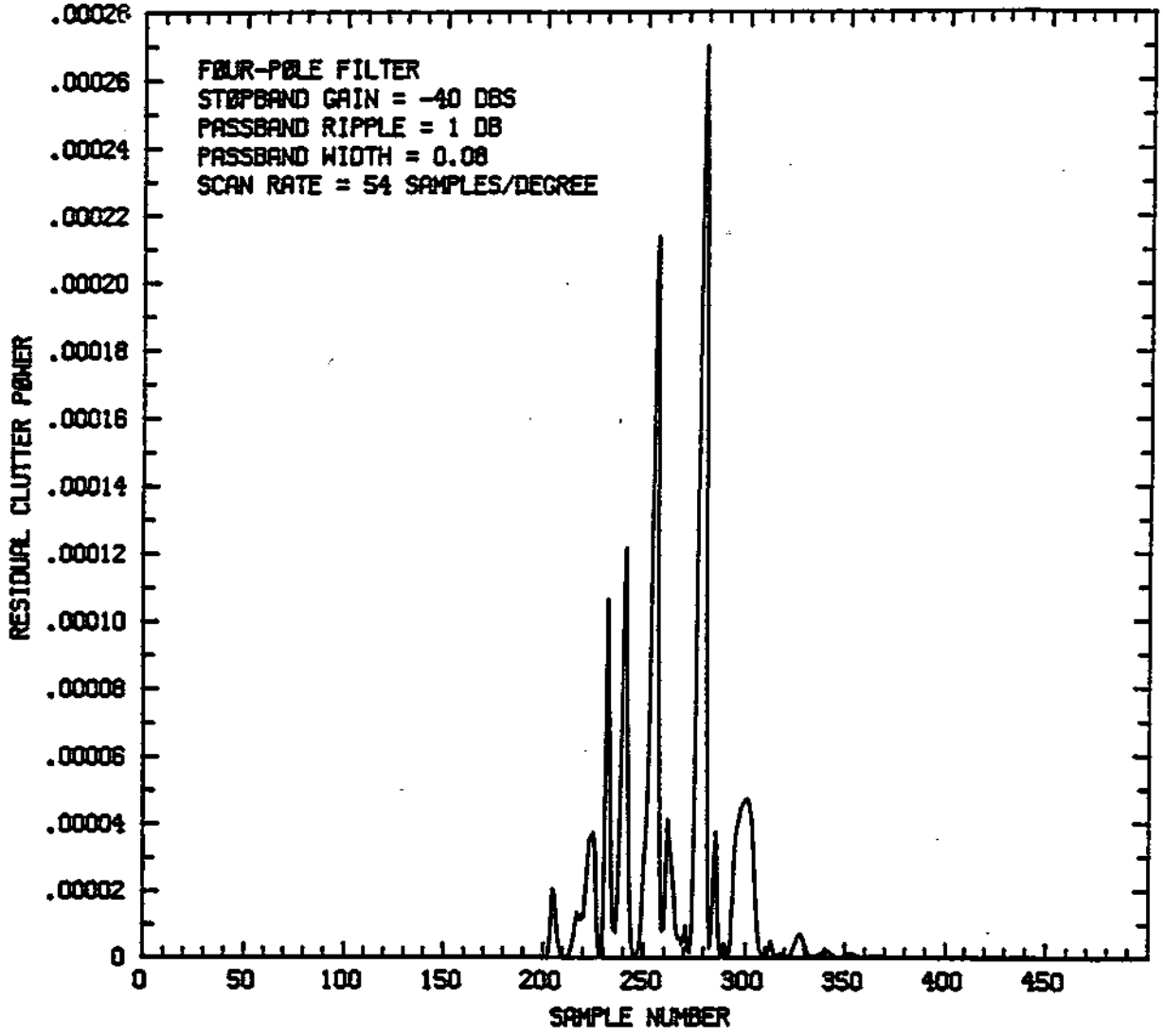


Figure 4.9(b). Same as Fig. 4.8(b) except stopband gain is -40 dB.

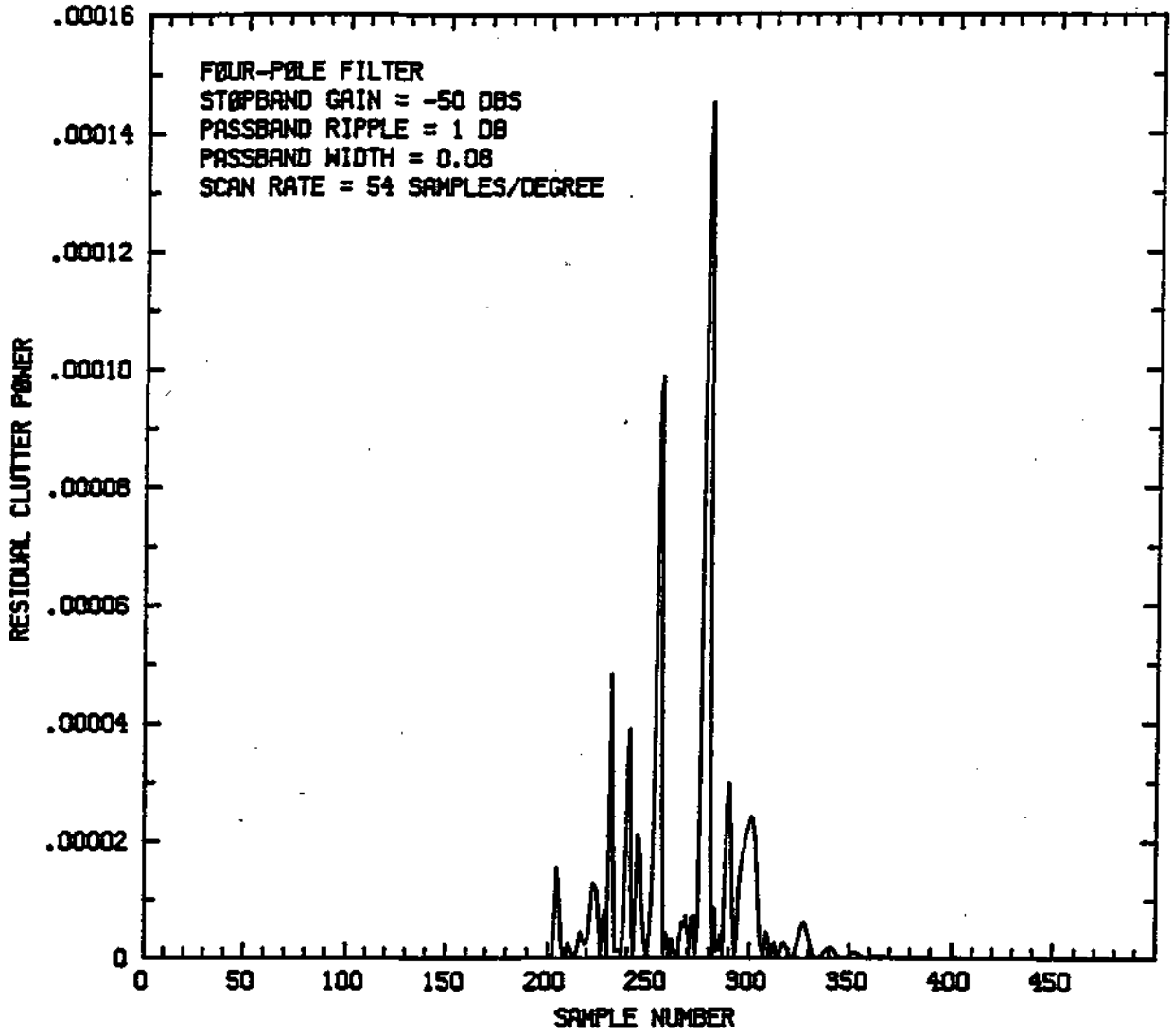


Figure 4.9(c). Same as Fig. 4.8(b) except stopband gain is -50 dB.

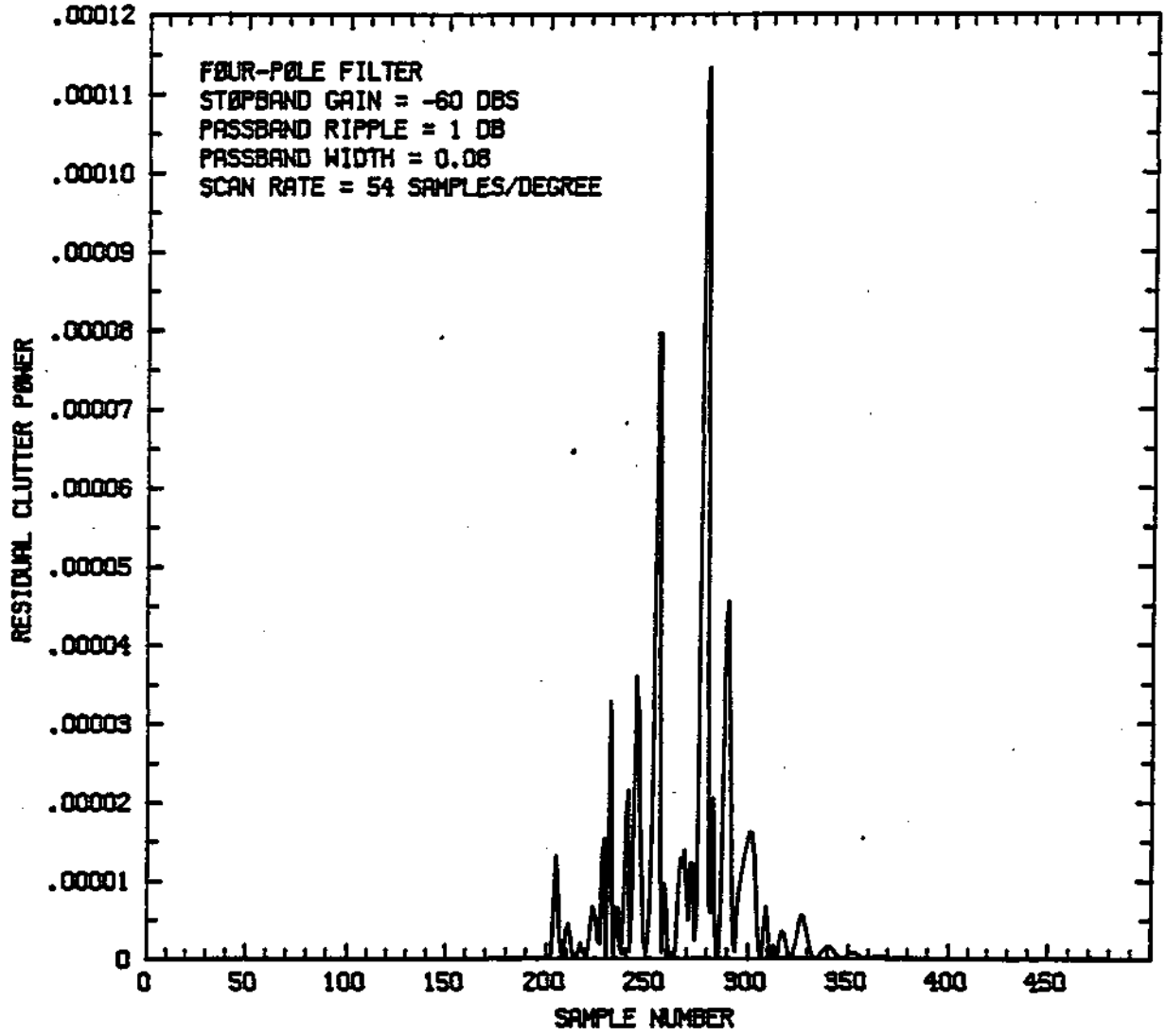


Figure 4.9(d). Same as Fig. 4.8(b) except stopband gain is -60 dB.

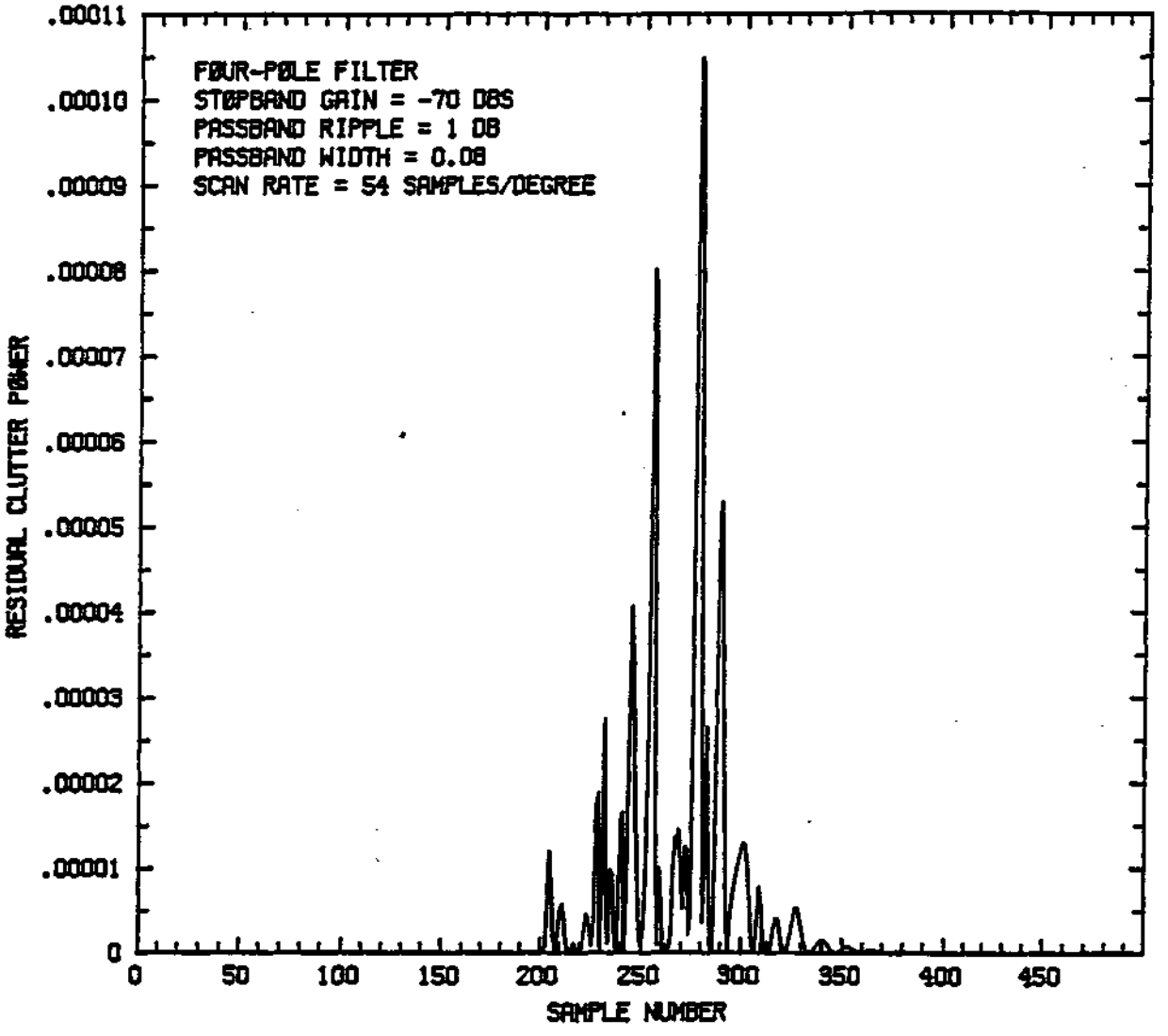


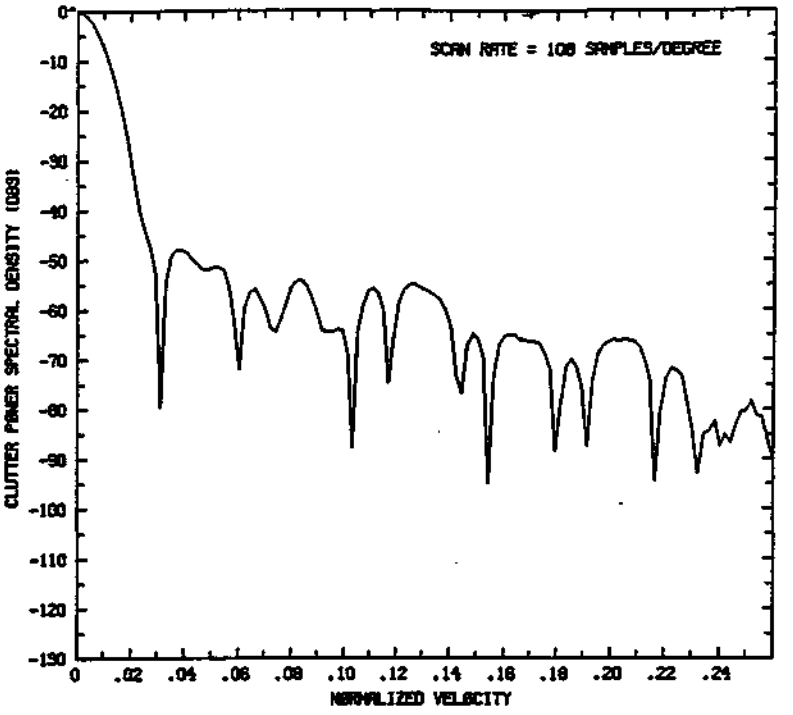
Figure 4.9(e). Same as Fig. 4.8(b) except stopband gain is -70 dB.

minimal suppression increases for the filters with stopband gains less than -50 dB can be easily explained by examining Fig. 4.10. Figures 4.10(a) and (b) are plots of clutter spectral densities for the discrete clutter models used. These spectral densities are identical to the antenna scan spectra. The effects of antenna scanning in the spectral domain consist of convolving these spectra with the target echo spectra. The spectra show a first side lobe that is only down about 48 dB. Thus, filters with stopband gains less than -48 dB can not suppress the clutter power below this side-lobe level. Even if the filter is wide enough to consume the first side lobe, the usefulness of stopband gains of less than -55 dB are defeated by other side-lobes.

Taking the spectra in Fig. 4.10 into consideration, reexamine the suppressions found in Figs. 4.6 through 4.9. As a specific example, look at Fig. 4.8(b). According to Fig. 4.10(b), a filter with a - 50 dB stopband gain and a passband width of 0.08 should result in a suppression of 50 dB since all side lobes outside of 0.08 are less than -50 dB. When examining Fig. 4.8(b), the minimum suppression is seen to be only 38 dB. This apparent contradiction can be explained by the fact that the 38 dB suppression is only a minimum instantaneous value while the 50 dB suppression is a total value. To obtain a total power suppression, a summation over the instantaneous power is needed. Dividing this value by total input signal power would give a total power suppression.

Figures 4.11 through 4.14 show errors introduced by filtering weather signals in the absence of clutter. The

(a)



(b)

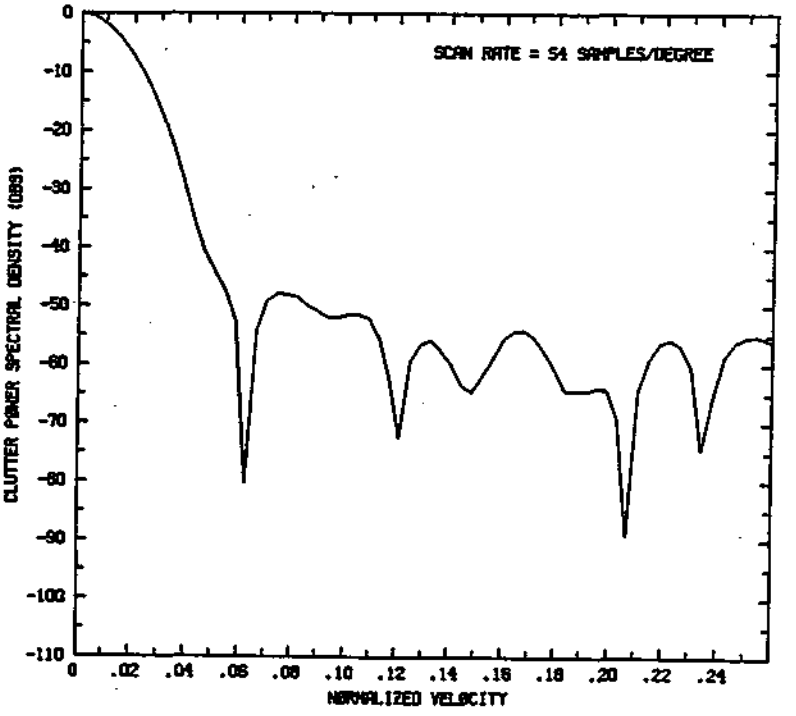


Figure 4.10. Discrete-model clutter spectral densities for (a) slow antenna rotation rate, (b) fast antenna rotation rate.

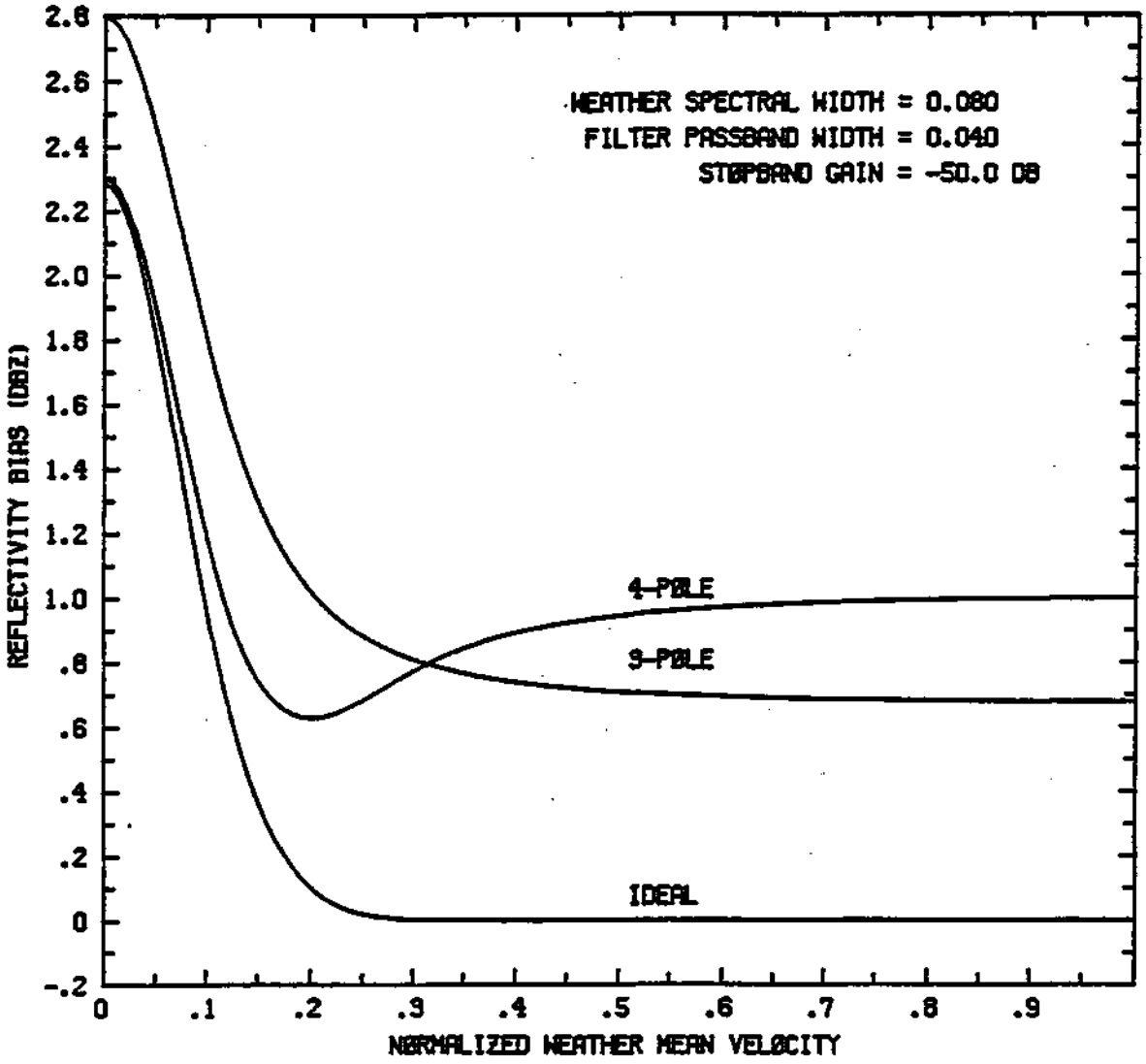


Figure 4.11(a). Comparisons of reflectivity biases introduced by filtering a weather signal by three different filters.

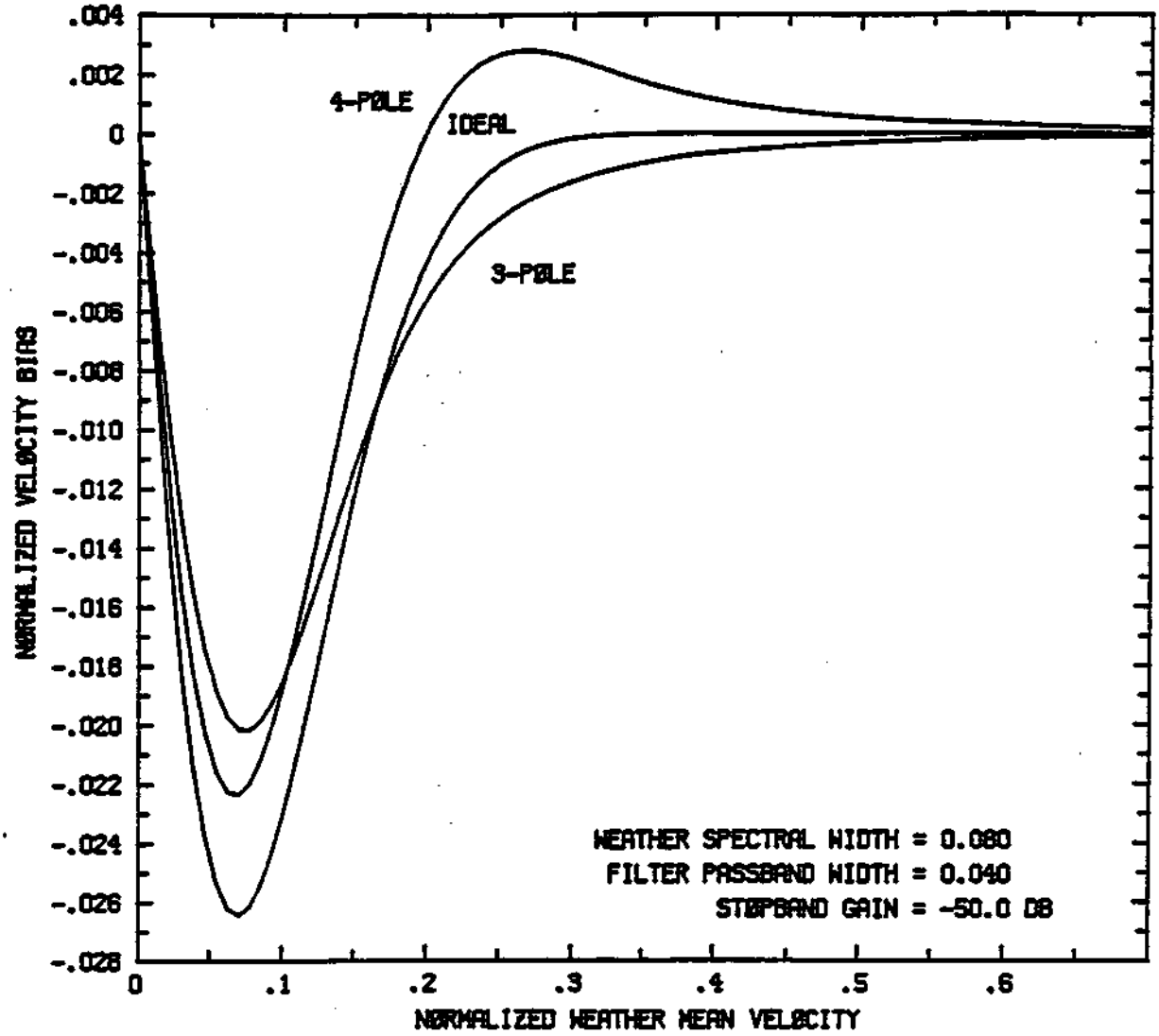


Figure 4.11(b). Same as Fig. 4.11(a) but velocity biases.

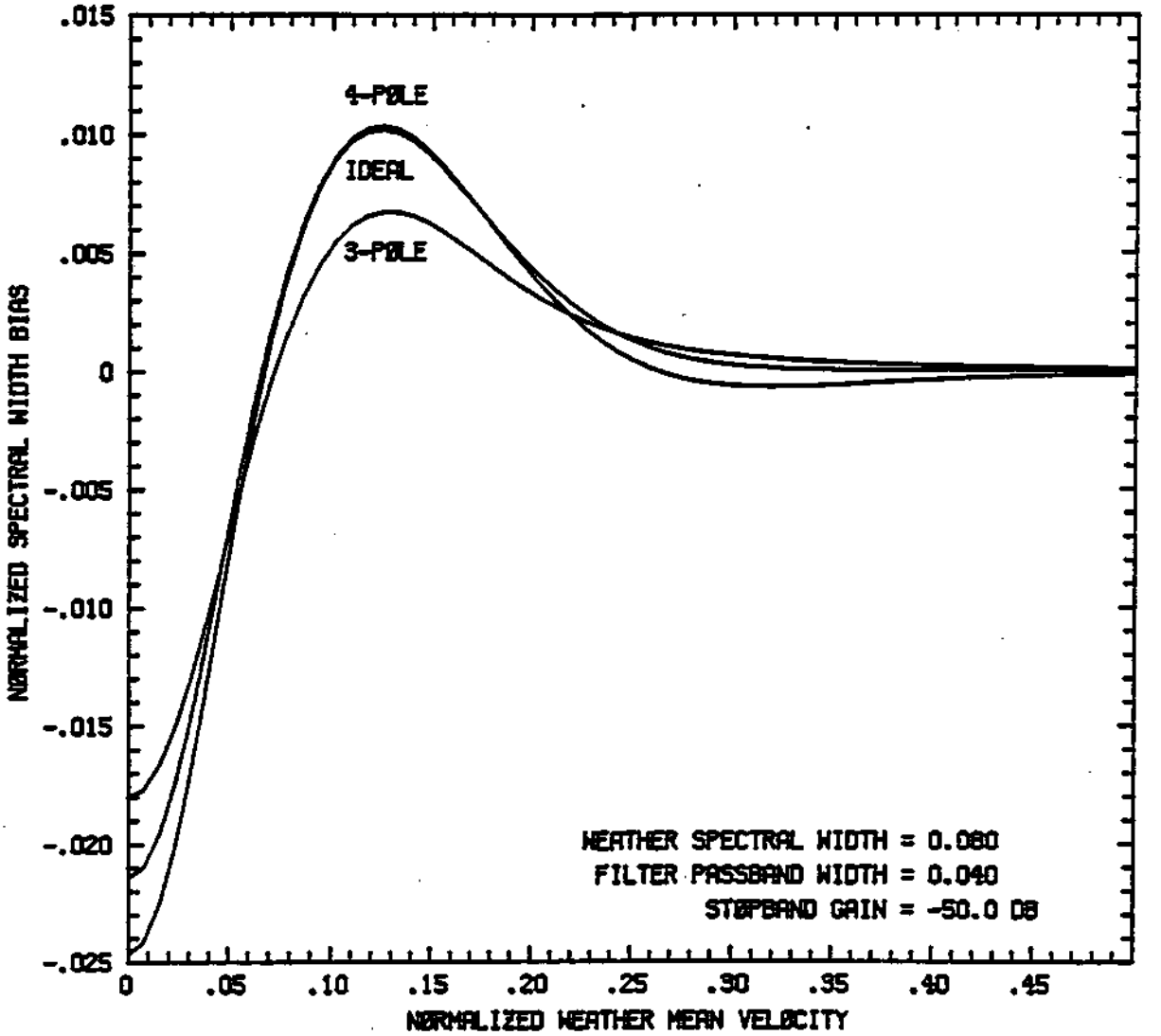


Figure 4.11(c). Same as Fig. 4.11(a) but spectral width biases.

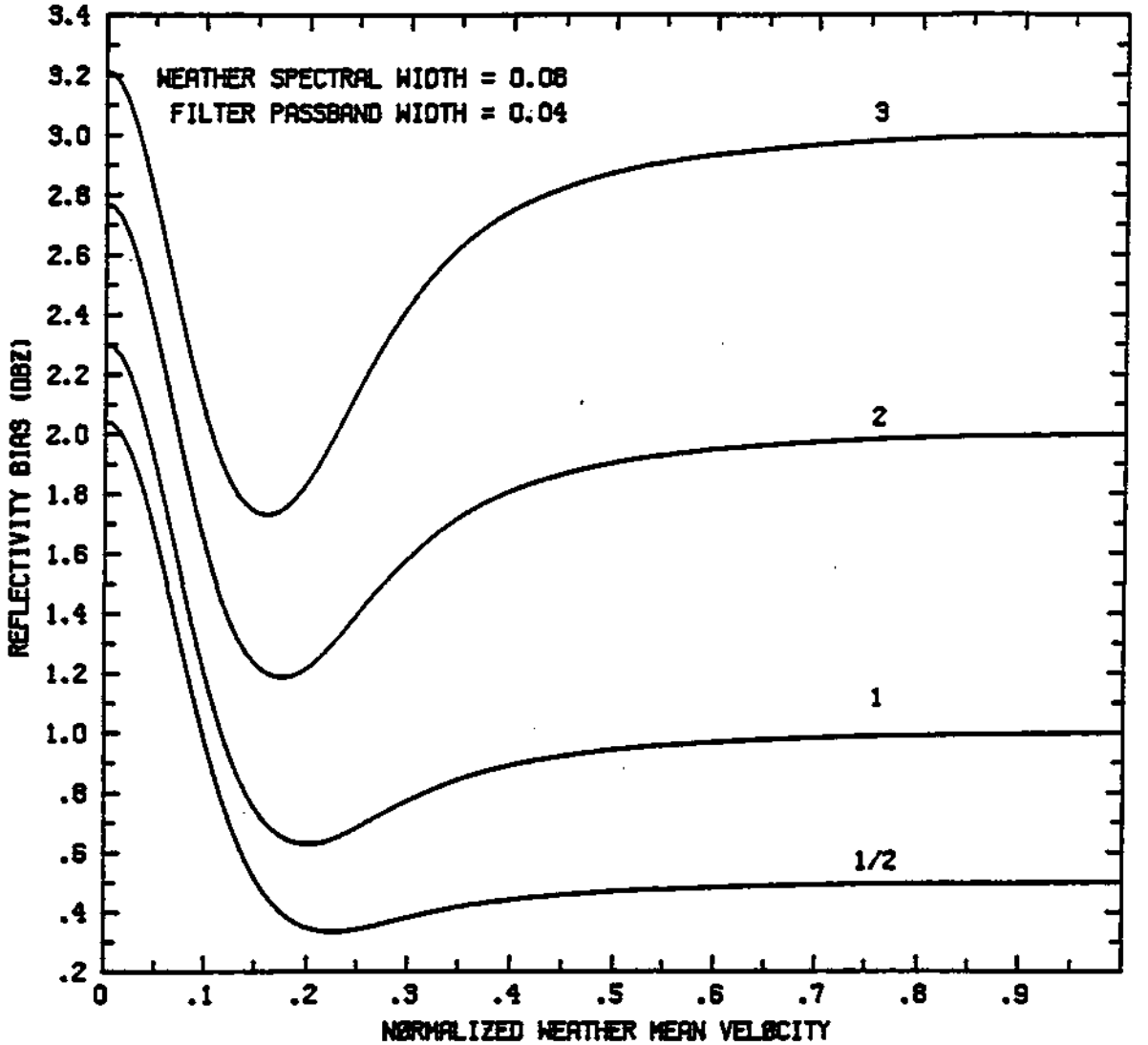


Figure 4.12(a). Reflectivity biases introduced by filtering a weather signal by a four-pole filter having a -50 dB stopband gain. Line labels are filter passband ripples expressed in decibels.

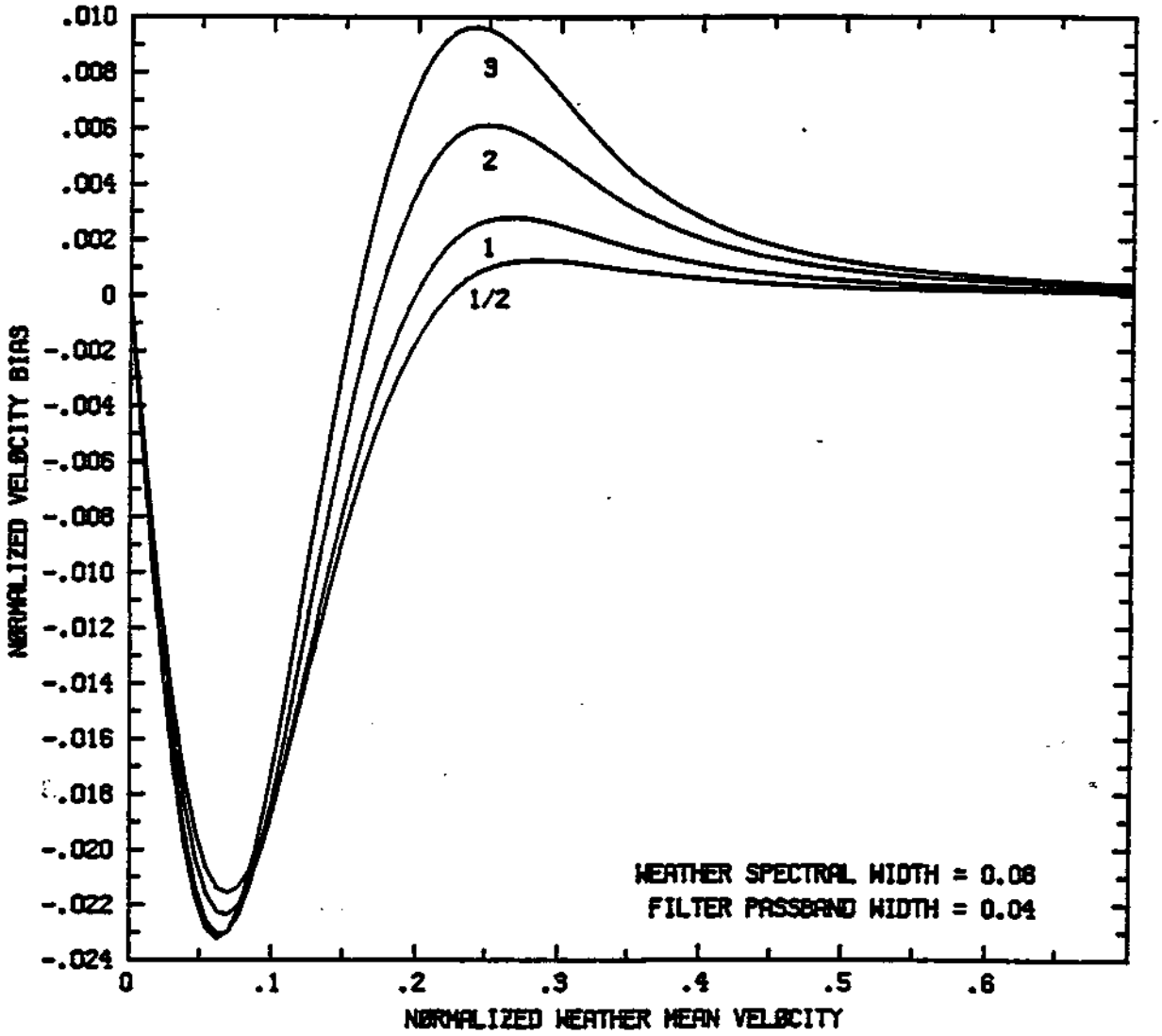


Figure 4.12(b). Same as Fig. 4.12(a) but velocity biases.

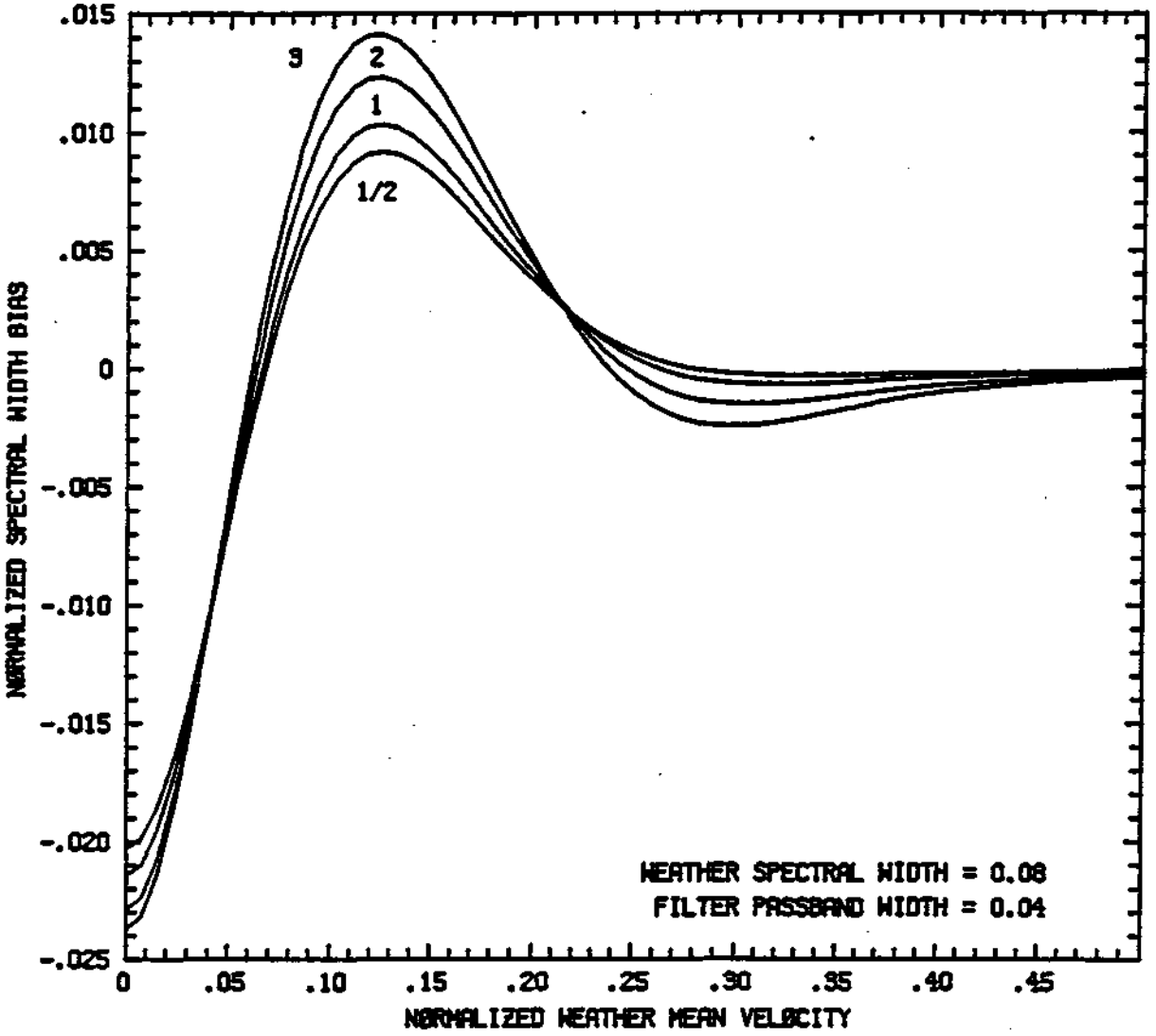


Figure 4.12(c). Same as Fig. 4.12(a) but spectral width biases.

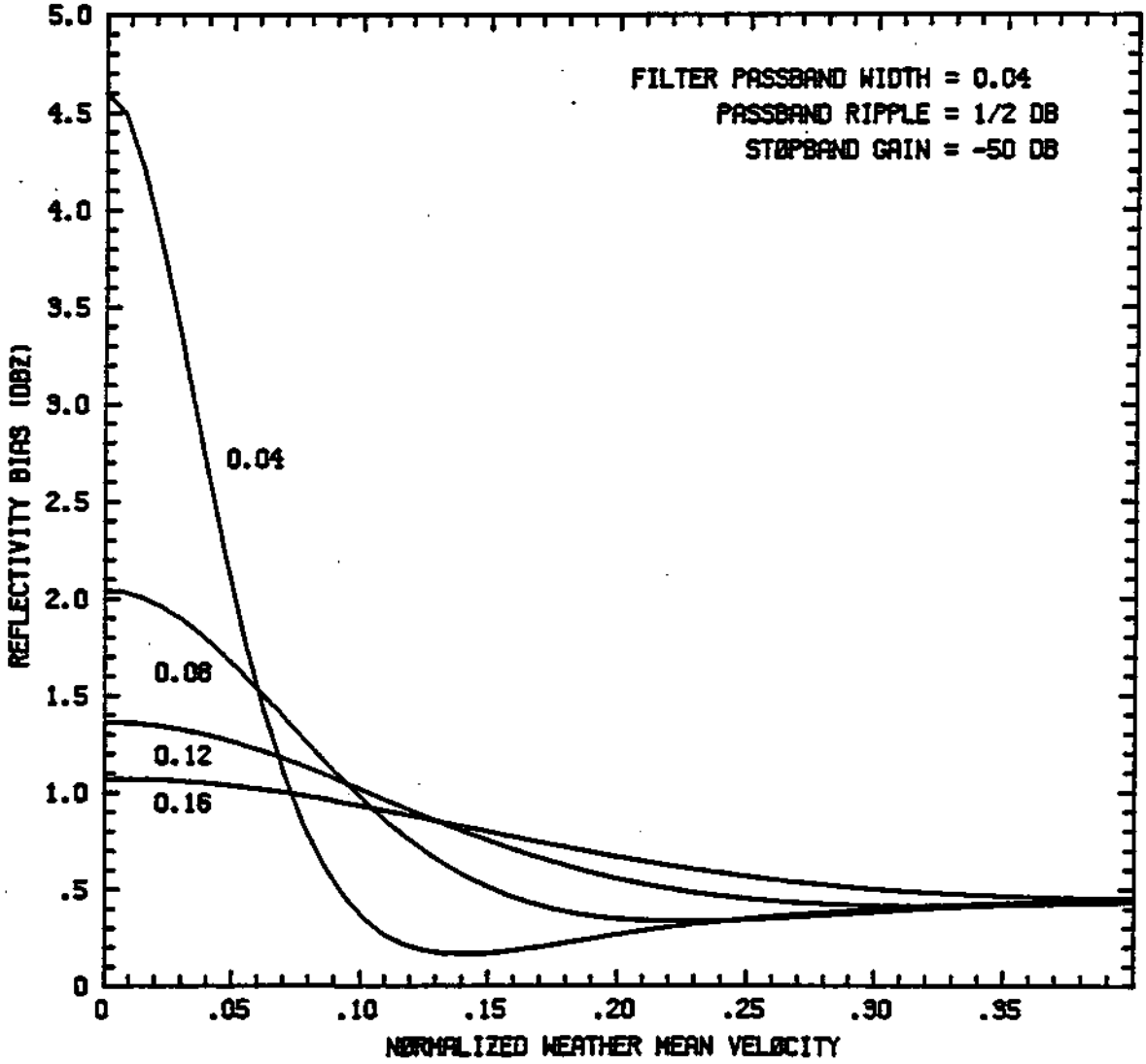


Figure 4.13(a). Reflectivity biases introduced by filtering a weather signal of various spectral widths. Line labels are weather spectral widths.

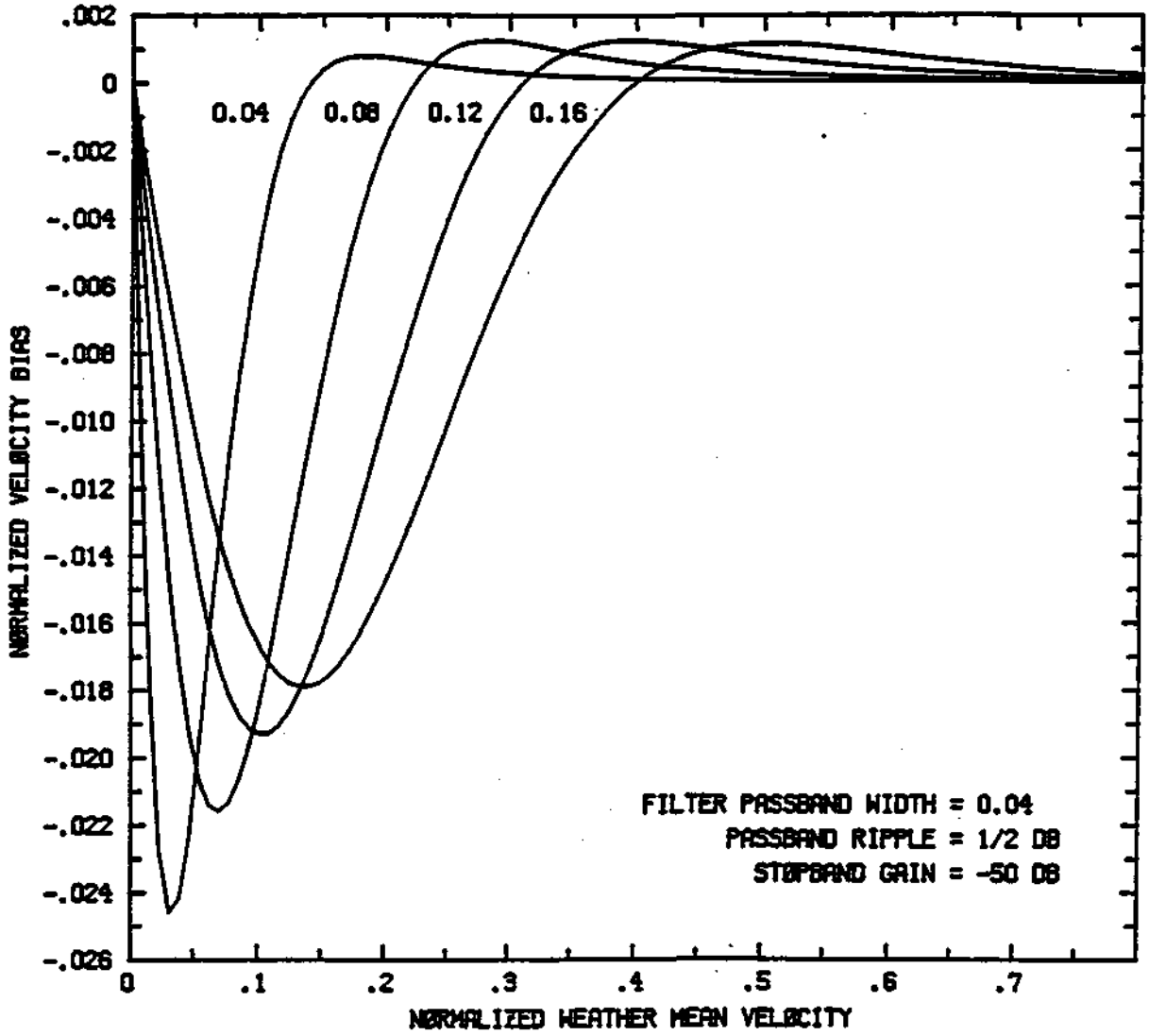


Figure 4.13(b). Same as Fig. 4.13(a) but velocity biases.

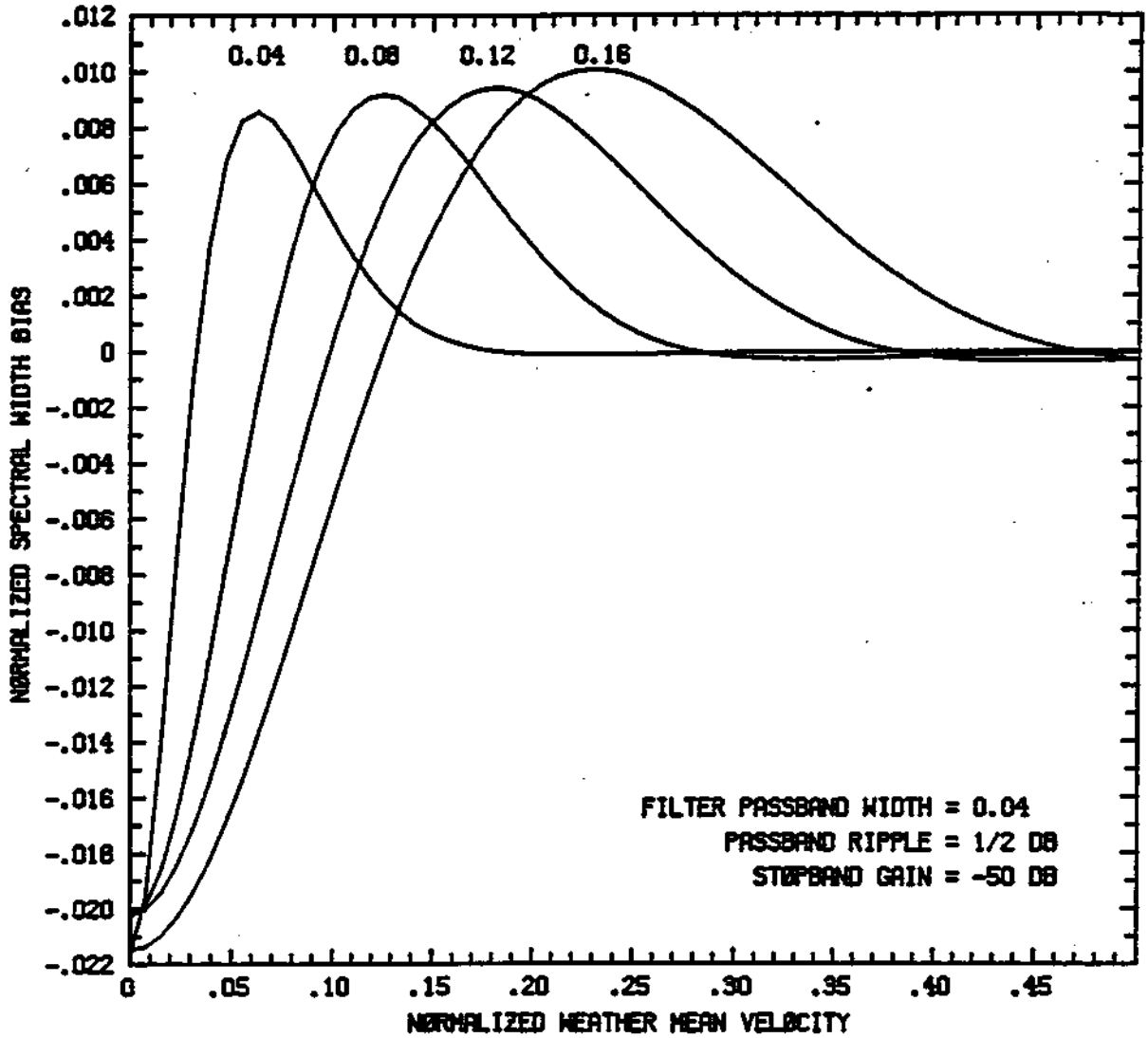


Figure 4.13(c). Same as Fig. 4.13(a) but spectral width biases.

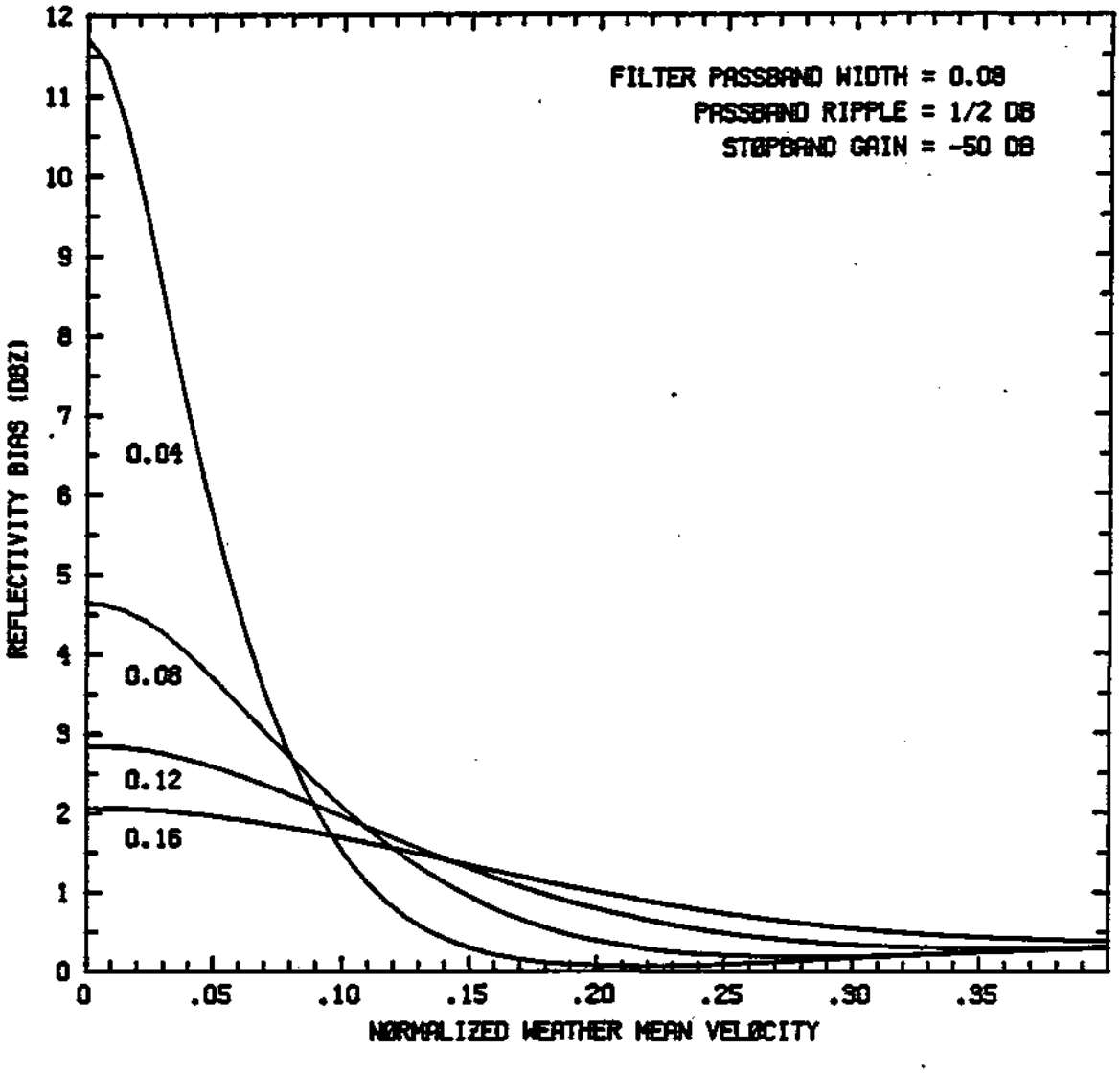


Figure 4.14(a). Same as Fig. 4.13(a) except filter width is 0.08.

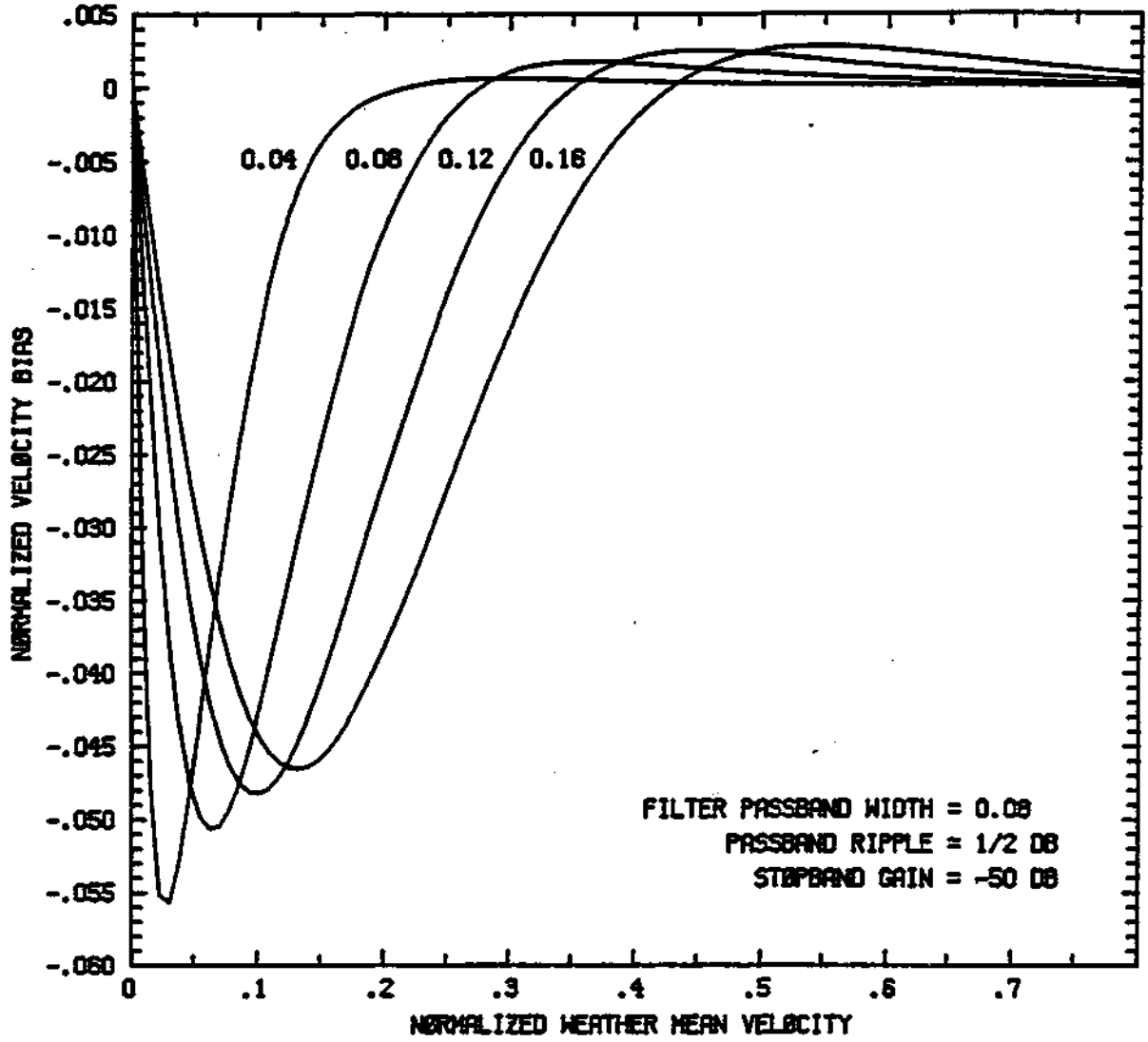


Figure 4.14(b). Same as Fig. 4.13(b) except filter width is 0.08.

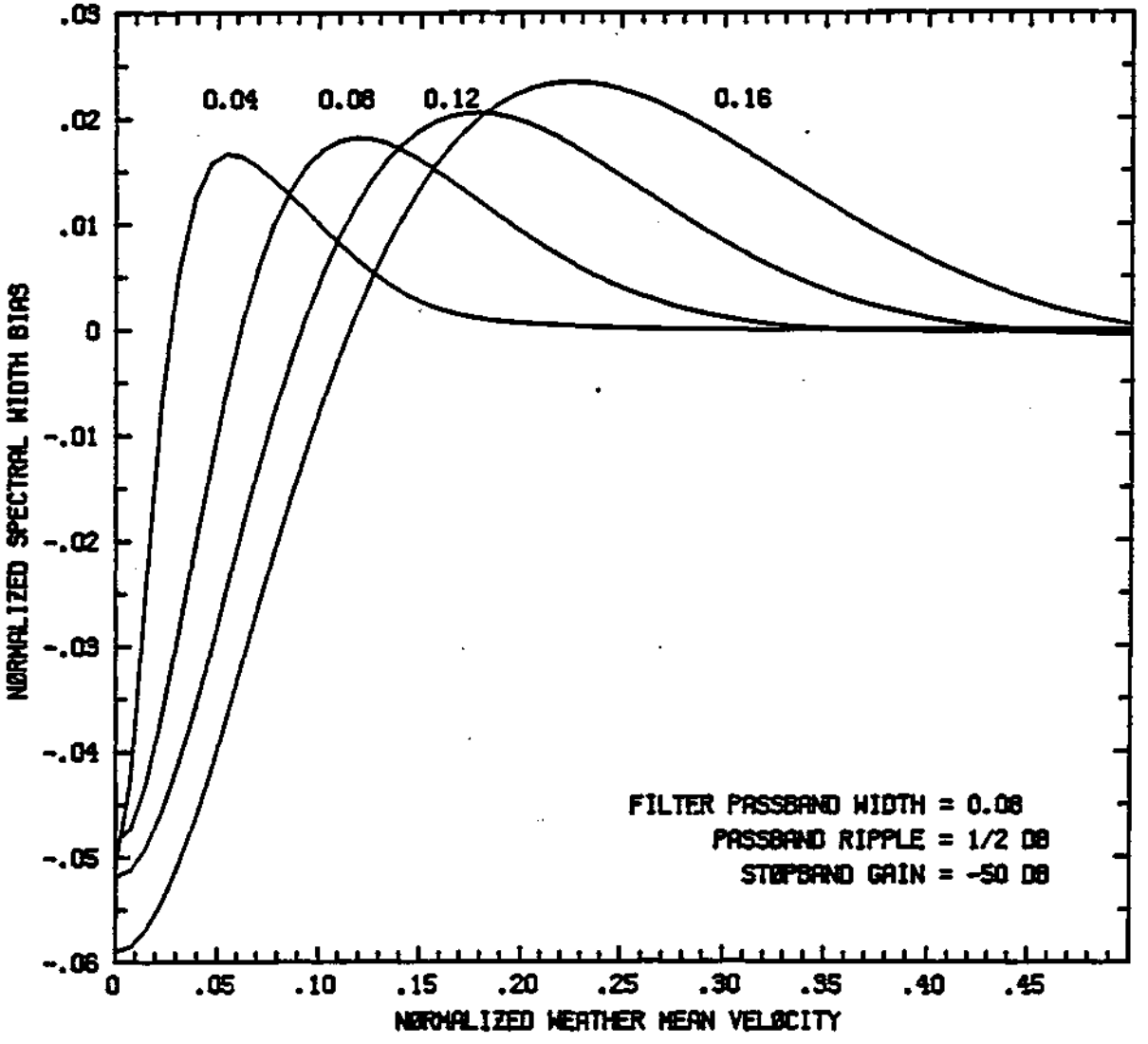


Figure 4.14(c). Same as Fig. 4.13(c) except filter width is 0.08.

procedure for generating these errors has five steps:

(1) Create desired weather spectrum $S_w(v)$

(2) Form filtered weather spectrum $S_f(v) = |H(v)|^2 S_w(v)$

(3) Generate the filtered weather autocorrelation $R_f(t)$ by taking the inverse Fourier transform of $S_f(v)$

(4) Form the pulse-pair estimates

(5) Subtract pulse-pair estimates from the true value to give biases.

Figure 4.11 is a comparison of errors between an ideal filter, a three-pole elliptic filter and a four-pole elliptic filter. The ideal filter's magnitude is zero in the stopband, one in the passband, and has an infinitely narrow transition band. The elliptic filters used have passband ripples of one dB and stopband gains of -50 dB. All the filters have passband cutoff velocities of 0.04. The simulated weather signal is gaussian with a spectral width of 0.08. The reflectivity biases of Fig. 4.11(a) show little difference between filters. Less than one dB separates the filters at any mean weather velocity. The velocity biases of Fig. 4.11(b) also show little difference between filters. The total errors are quite small. The maximum error introduced by the four-pole filter is only 0.022. The spectral width biases are shown in Fig. 4.11(c). As expected, errors are greatest when the weather signal is centered in the filter notch. Maximum width errors are in the neighborhood of 0.02. Figure 4.12 shows the effects of filter passband ripple on the parameter estimator biases. The basic filter is a four-pole elliptic filter with a

stopband gain of -50 dB and a passband cutoff velocity of 0.04. These figures show that the errors increase as the filter passband ripples become larger. Larger passband ripples allow sharper transition bands which translate into more rejection in the filter stopband. This means that not only will more weather signal be rejected in the stopband, but so will more clutter signal. The relative increases in stopband rejection attained by increasing the passband ripple are actually quite minor. The difference in transition band width attained by allowing some passband ripple with the elliptic filter, as opposed to no passband ripple with a Tchebychev-type filter, is more significant. In order to keep weather estimate biases down while keeping transition bands narrow, some passband ripple is needed but it should be kept relatively small, 1/2 or 1 dB.

Figures 4.13 and 4.14 show biases introduced by filtering weather signals of various spectral widths. The filter used in Fig. 4.13 was a four-pole elliptic with a stopband gain of -50 dB, a passband ripple of 1/2 dB and a passband cutoff of 0.04. Once again, power estimates have maximal errors when the weather signals have zero mean velocity. The maximal errors are the same as the suppressions predicted by Fig. 4.5. The velocity errors are maximum when the mean weather velocity is approximately equal to the weather spectral width. These maxima are still fairly small, in the 0.02 range. Maximum spectral width errors are fairly independent of weather width. The wider weather spectra do tend to spread the errors over a larger range of mean weather velocities. Figure 4.14 is identical

to Fig. 4.13 except that a filter with the wider notch width of 0.08 was used. The results are almost identical to those of Fig. 4.13 except that all the biases are approximately doubled. Although not pictured, comparisons of parameter estimate biases for filters of varying stopband gains were made. The results showed virtually no bias difference for stopband gains in the range of -30 to -70 dB.

4.4 Filter Initialization

As mentioned at the end of Section 4.1, the fact that one must consider initialization of recursive filters has traditionally been a reason to avoid their use. When the delay elements of a recursive filter are set randomly, the filter will generally ring when turned on. Since during block processing the data blocks are not contiguous, the filter must be restarted every data block. Typically, block lengths are on the order of the dwell time. For a typical beamwidth of one degree, antenna rotation rate of 15 degrees per second, and a pulse repetition time of 1 ms, the dwell time given by (2.11) is about 66 pulses. Transient responses of the recursive filters used in ground clutter reduction may be significant for a hundred pulses or more. For accurate parameter estimates, this ringing must be greatly reduced. Fortunately, this is not a problem at the CHILL since samples are processed in real time as they arrive. This enables a continuous data stream to flow through the filter. Initialization is only a problem when the filters

are first started. If estimates are produced every dwell time, the filter ringing will probably corrupt only the first few estimates.

Even though filter initialization is not a problem at the CHILL, a few techniques can be used to try to minimize the filter ringing. The first technique involves weighting the first samples into the filter with the front half of the Hamming window,

$$W(n) = 0.54 - 0.46\cos[2\pi n/(N-1)],\ 0 \le n < (N-1)/2. \quad (4.14)$$

The second initialization technique, called one-pulse initialization, involves loading the filter delay elements with their anticipated steady state values [6]. If the input to the filter is a constant value, this initialization can be done perfectly with the first sample. For the three-pole filter shown in Fig. 4.2(a), this initialization can be achieved by setting the delay element closest to the filter input to

$$gU/(1-k_4), \quad (4.15)$$

where U is the first sample into the filter. Both of the other delay elements must be set to zero. For the four-pole filter shown in Fig. 4.2(b) this initialization can be achieved by setting the two delay elements closest to the filter input to

$$gU/(1-k_3+k_4), \quad (4.16a)$$

and two delays closest to the filter output to

$$gU(2-k_1)/[(1-k_3+k_4)(1-k_5+k_6)]. \qquad (4.16b)$$

To obtain a quantitative measure of the effectiveness of these initializations, simulations were run. Some of the results are displayed in Figs. 4.15 through 4.17. Blocks of simulated complex time series data 256 samples long were constructed using the method outlined by Zrnic in [27] and [30]. The spectra of this data consisted of a gaussian weather signal and a gaussian ground clutter signal. The ground clutter signal had a spectral width of 0.02 and was 10 dB stronger than the weather signal. The weather signals had various mean velocities and spectral widths. The time series data were applied to filters with passband velocities of 0.08, stopband gains of -50 dB, and total passband ripples of 1/2 dB. Both the initialization techniques mentioned above were used along with an uninitialized run. The outputs of these filters were broken into four contiguous blocks of 64 samples each for pulse-pair parameter estimation. The whole process was repeated for fifty independent trials and the resulting parameter estimates were averaged to give the values in the plots.

Figure 4.15 shows the mean weather velocity estimates for the different initialization types. Ideal performance would result in a line connecting the lower-left corner and the upper-right corner of the grid. These estimates were formed from the first data block out of the filter. Even the estimates obtained when the filters were uninitialized are quite good. In this case filter initialization was not really needed. If the clutter-to-signal power ratio were

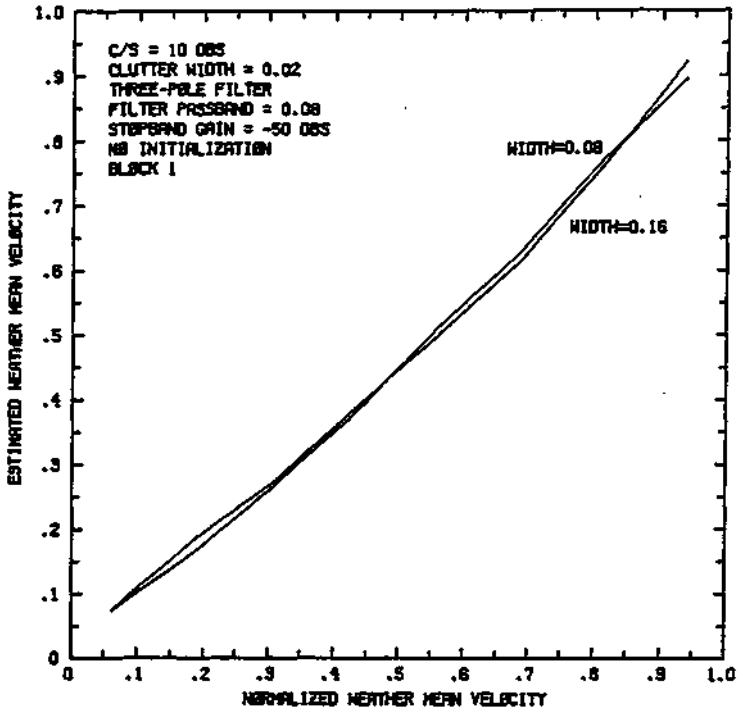


Figure 4.15(a). Estimated mean weather velocities from the first block of data out of an uninitialized three-pole filter. The line labels indicate weather spectral widths.

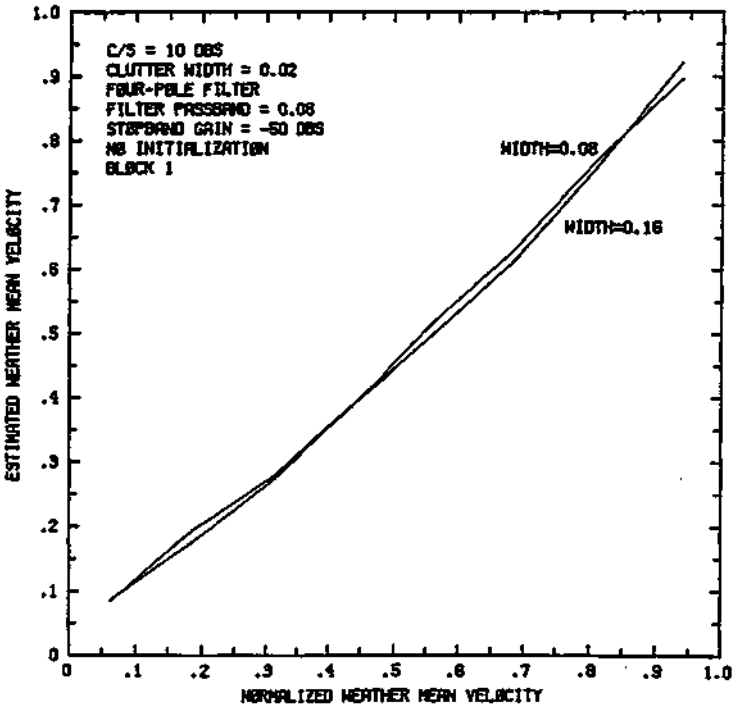


Figure 4.15(b). Same as Fig. 4.15(a) except a four-pole filter was used.

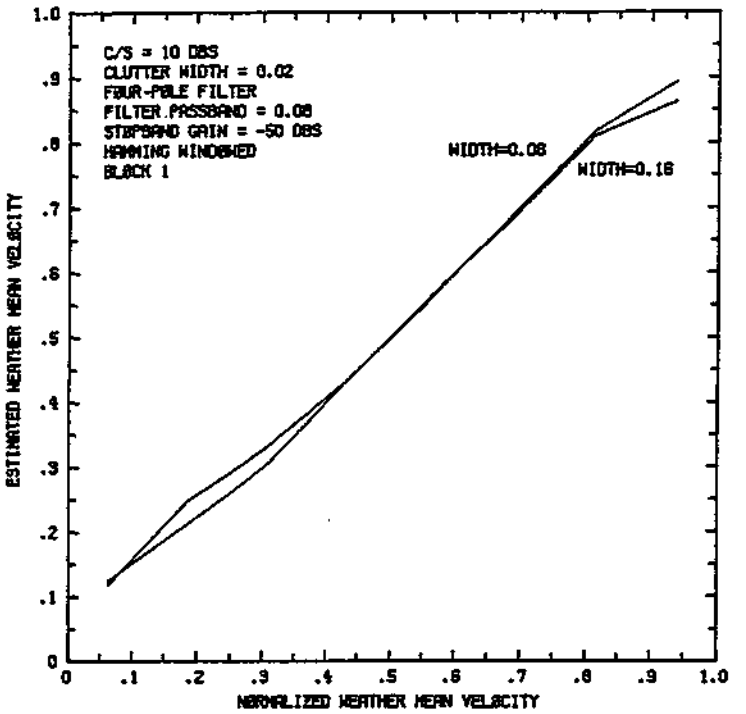


Figure 4.15(c). Same as Fig. 4.15(b) except the Hamming window initialization was employed.

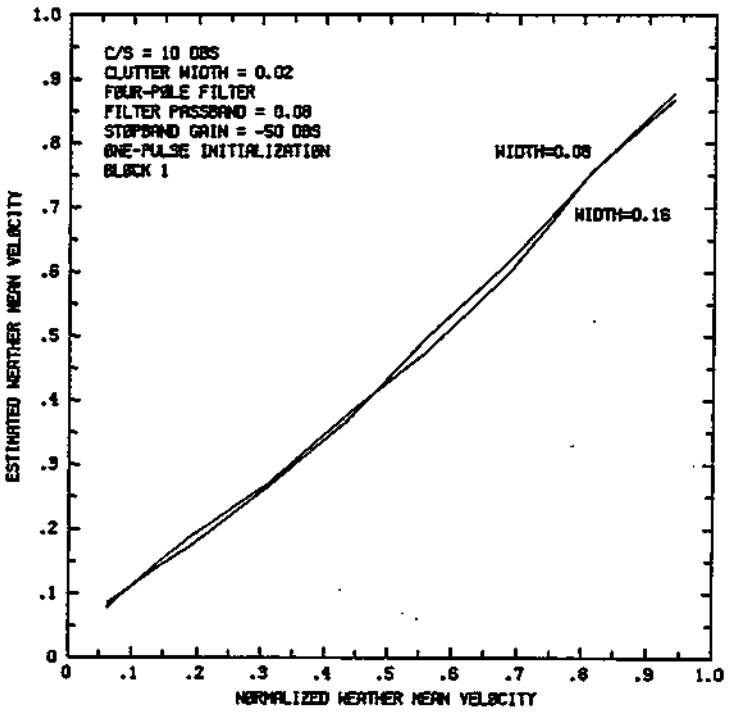


Figure 4.15(d). Same as Fig. 4.15(b) except the one-pulse initialization was employed.

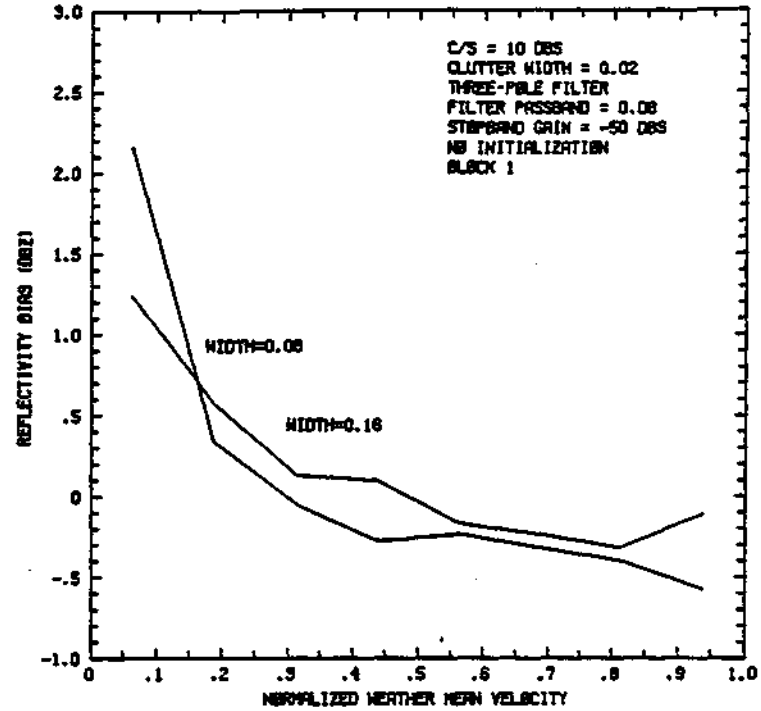


Figure 4.16(a). Same as Fig. 4.15(a) except reflectivity biases. The line labels indicate weather spectral widths.

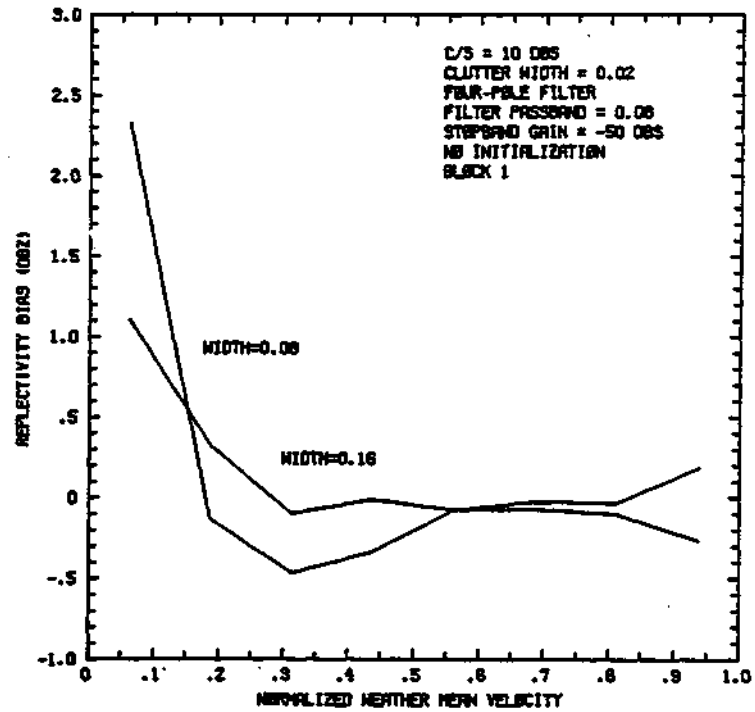


Figure 4.16(b). Same as Fig. 4.16(a) except a four-pole filter was used.

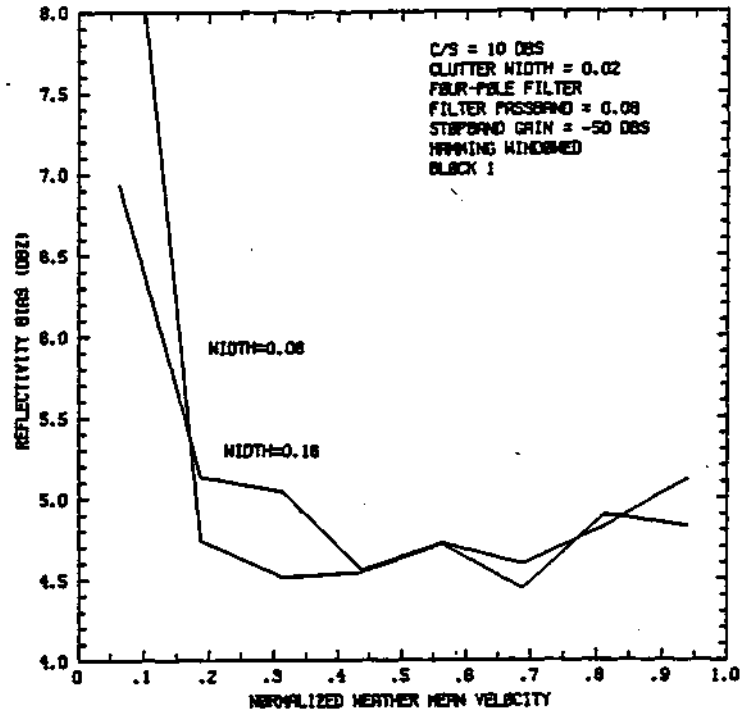


Figure 4.16(c). Same as Fig. 4.16(b) except the Hamming window initialization was employed.

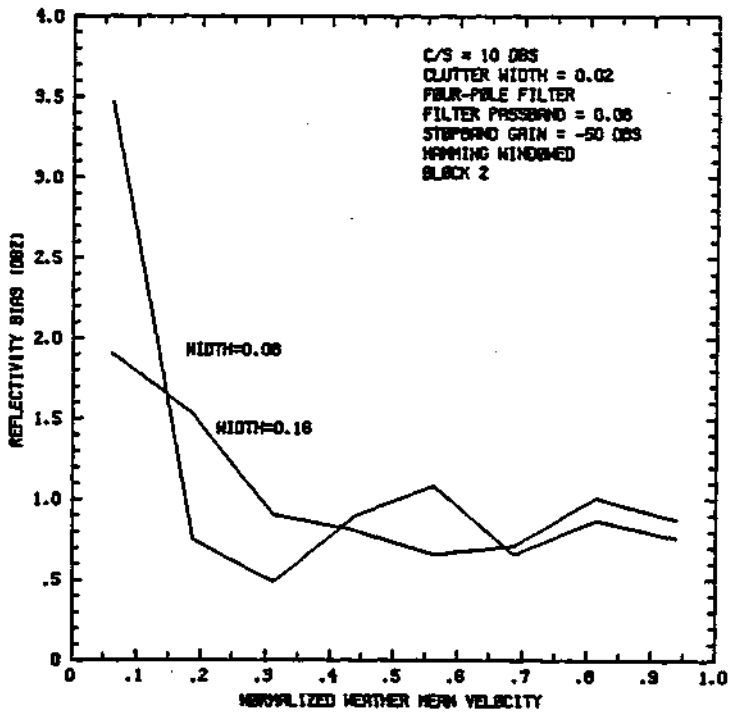


Figure 4.16(d). Same as Fig. 4.16(c) except the second block of data is used to form the estimates.

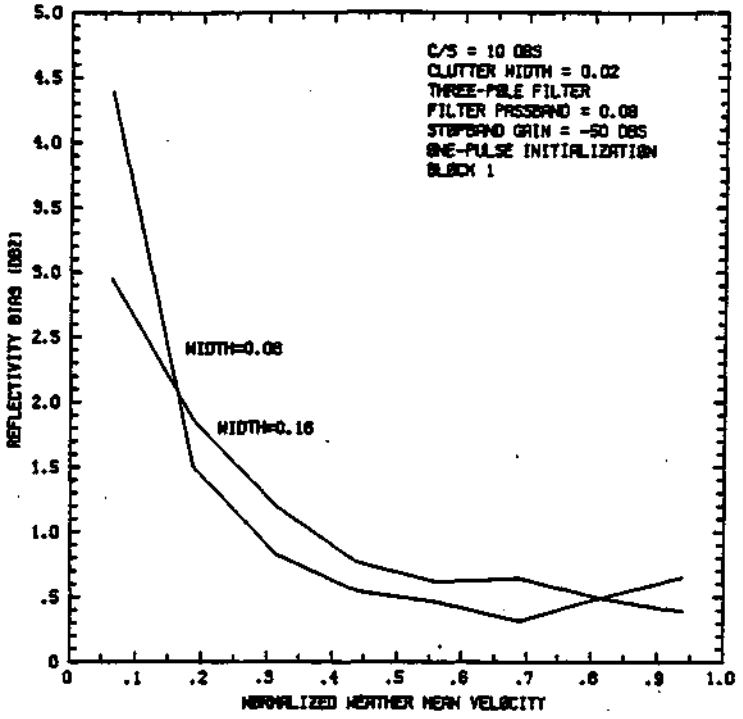


Figure 4.16(e). Same as Fig. 4.16(a) except the one-pulse initialization was employed.

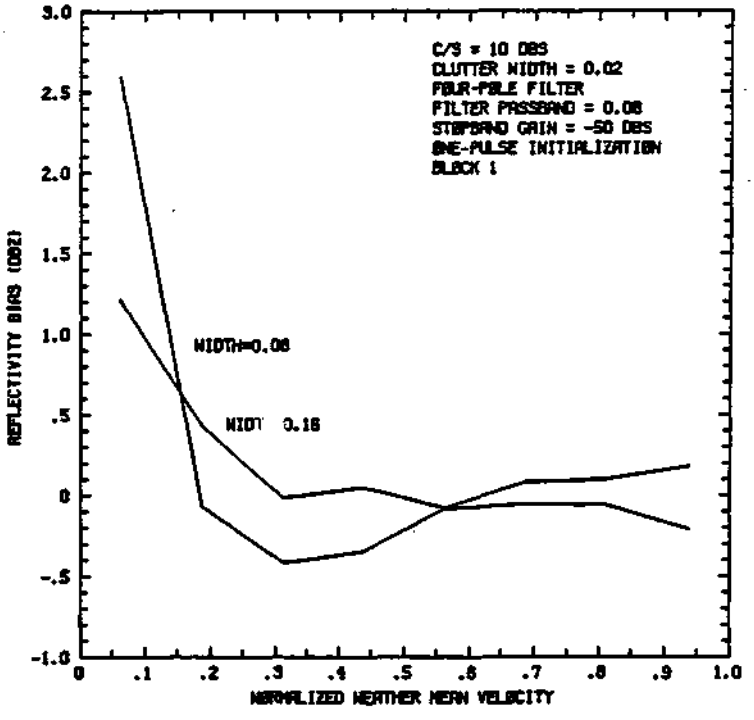


Figure 4.16(f). Same as Fig. 4.16(b) except the one-pulse initialization was employed.

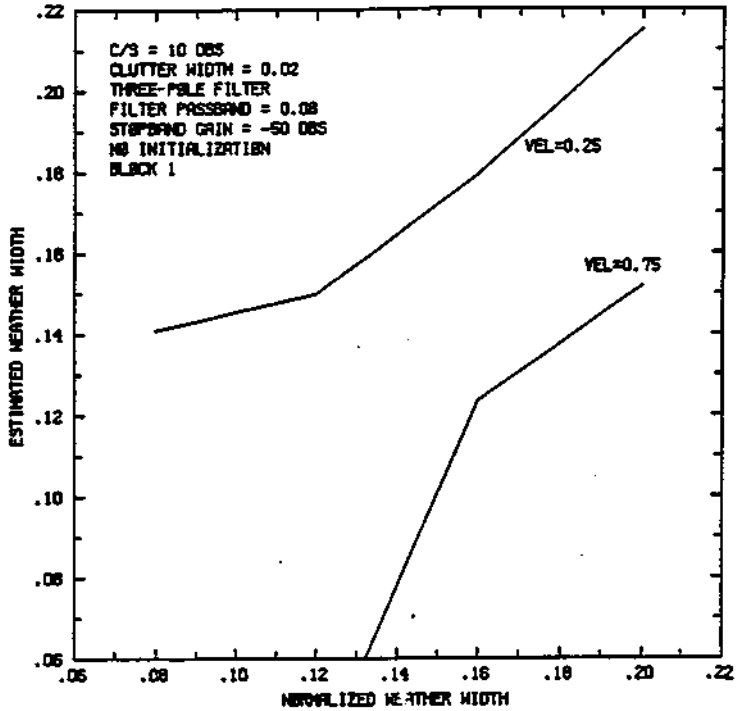


Figure 4.17(a) Same as Fig. 4.16(a) except estimated weather widths. The line labels indicate mean weather velocities.

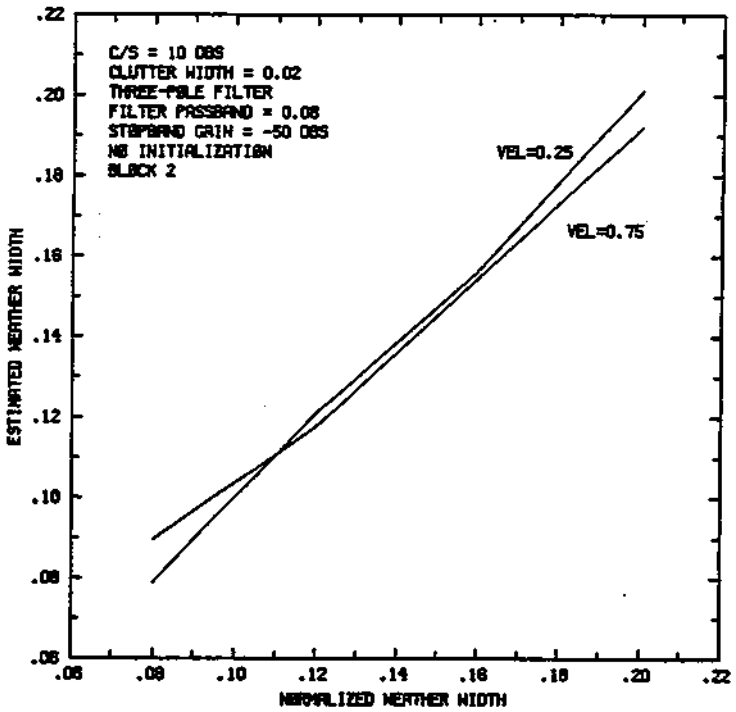


Figure 4.17(b) Same as Fig. 4.17(a) except the second output data block was used to form estimates.

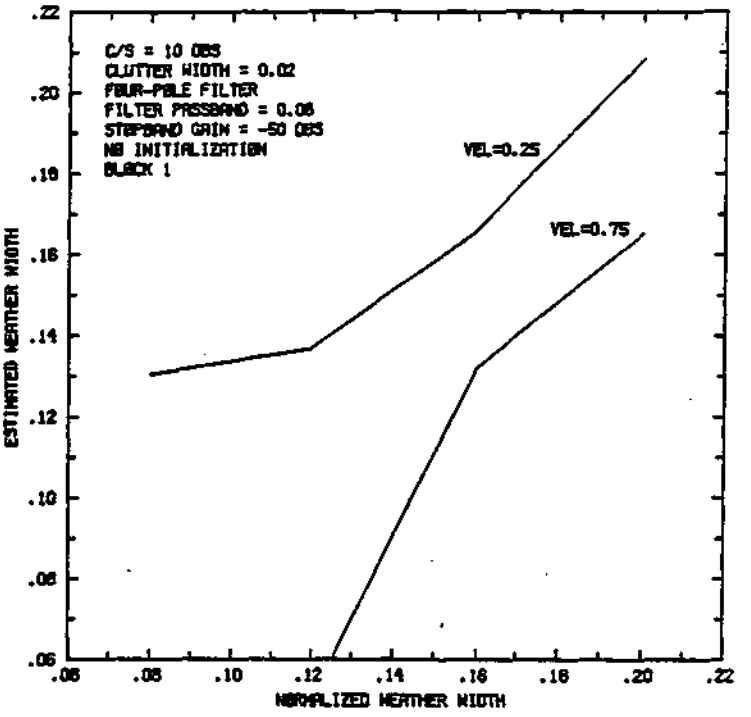


Figure 4.17(c) Same as Fig. 4.17(a) except a four-pole filter was used.

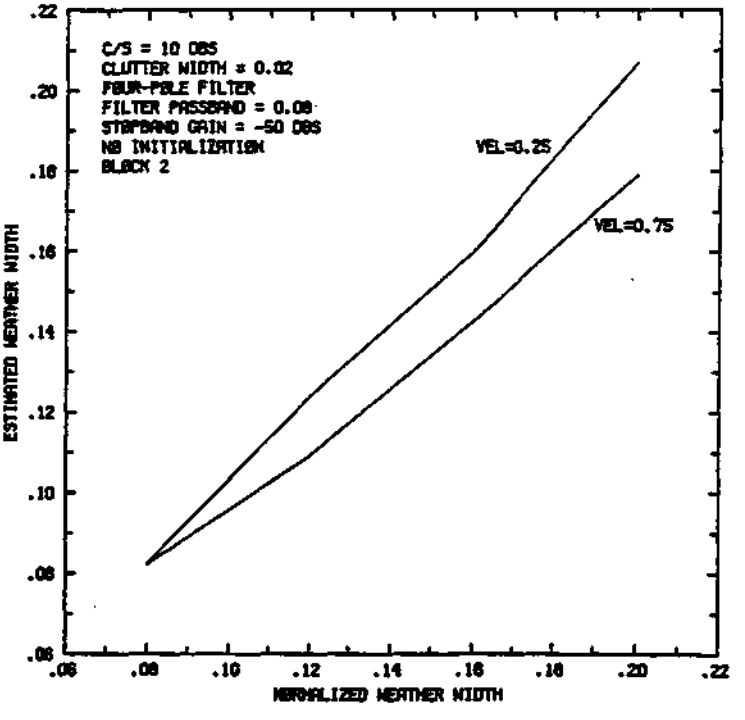


Figure 4.17(d) Same as Fig. 4.17(b) except a four-pole filter was used.

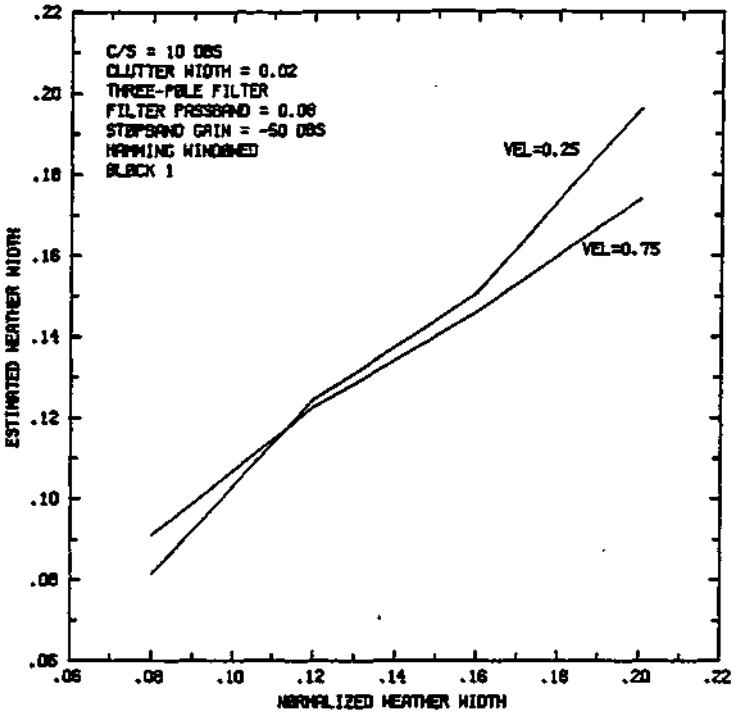


Figure 4.17(e) Same as Fig. 4.17(a) except Hamming window initialization was employed.

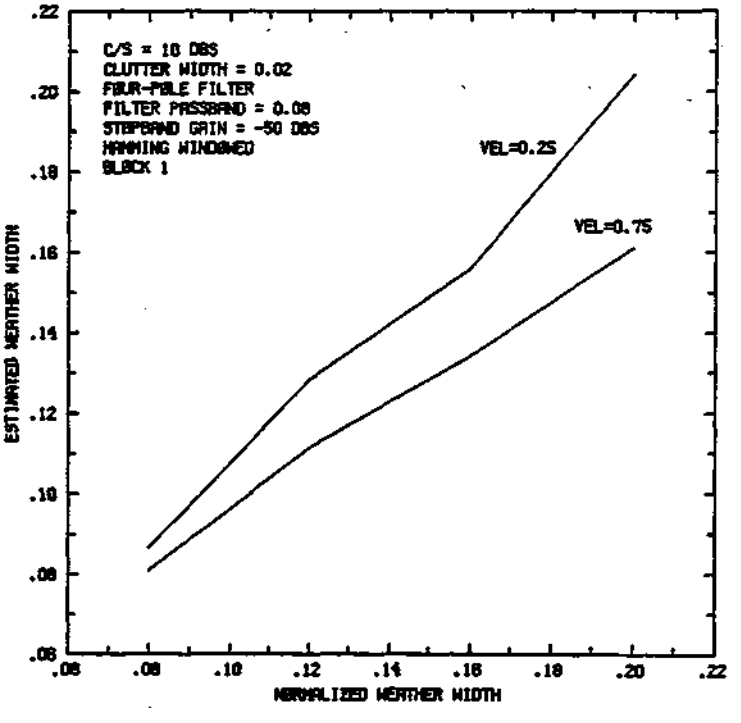


Figure 4.17(f) Same as Fig. 4.17(c) except Hamming window initialization was employed.

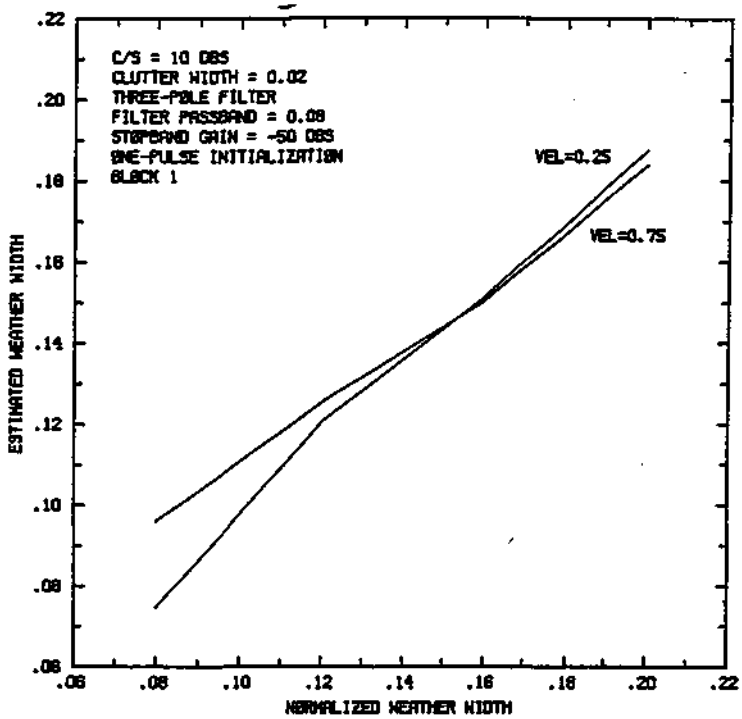


Figure 4.17(g) Same as Fig. 4.17(e) except one-pulse initialization was employed.

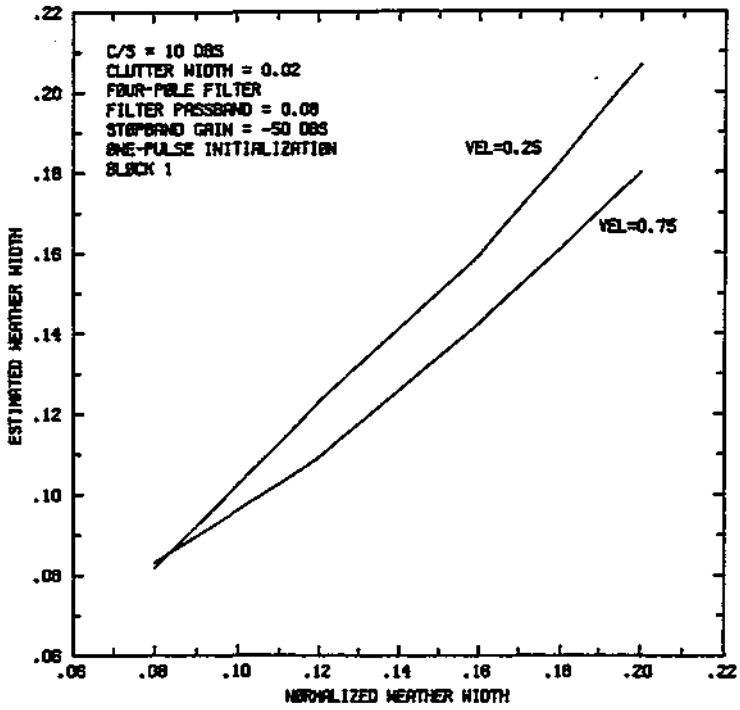


Figure 4.17(h) Same as Fig. 4.17(f) except one-pulse initialization was employed.

much larger, the initializations would make a difference.

Figure 4.16 shows the reflectivity biases in the data output from the filters. Notice that the general form of the curves follow those found in Fig. 4.11(a). Once again, initializations did not really help the parameter estimates. The Hamming window initialization even introduced a 5 dB greater error on the block it was applied to, Fig. 4.16(c). The errors settled down to their steady state values by the next block, Fig. 4.16(d). The uninitialized filter reflectivities are actually quite good.

Figure 4.17 displays the spectral width estimates obtained from the output of the filters for the different initialization techniques. As Figs. 4.17(a) through (d) show, the uninitialized filters needed at least a block to settle before reasonable estimates were obtained. Figures 4.17(e) and (f) show that fairly good results can be obtained from the first block out of the filter when the Hamming window initialization technique is used. Figures 4.17(g) and (h) show the results from the one-pulse initializations. When comparing these results to those from the Hamming window, these are seen to be a little better. Since the one-pulse initialization is more computationally efficient than the Hamming window initialization, it seems to be a better choice.

The ringing effects of recursive filters can cause a problem in addition to initialization, because of the automatic gain control used to obtain floating-point samples (see Appendix II). Since AGC exponent A/D converters do not sample as fast as the mantissa A/D converters, a large

ground return may saturate the mantissa A/D converters when the AGC has the attenuators switched to a small value. This may happen when the large ground return is surrounded by much smaller echoes. When this occurs, the sample corresponding to the range gate containing the large ground return will have a mantissa that is at its maximum, but its exponent will be too small. If subsequent pulses set the attenuators to the same value, the swamped echo will appear as a constant and be filtered out. The problem arises when a subsequent pulse changes the attenuator settings. When the attenuators switch, the saturated echo sample will have its exponent changed. This appears as a step to the input of the filter, causing it to ring. This ring may corrupt subsequent echoes.

Since the antenna is rotating relatively slowly, the AGC should not change often between subsequent pulses at the same range gate. Furthermore, since the beam is illuminating much the same area from pulse to pulse, the AGC should not change the exponent by more than one bit at a time between pulses at the same range gate. This corresponds to changing the saturated sample by a factor of two. Thus, the magnitude of the filter ring will be less than 3 dB while it lasts. This is a legitimate problem, but not really a large one. If the AGC is working well, the saturated samples should not occur too often.

CHAPTER 5

CONCLUSIONS

The contamination of weather echo spectral features by ground returns is a significant problem at the CHILL radar site. Low elevation scans at close ranges around the CHILL were observed to contain ground targets with average reflectivities in the neighborhood of 40 dBZ. A few structures were seen with reflectivities 20 dBZ stronger than this. The spectra of the ground returns are narrow and centered around zero velocity. Depending on the source of the echoes, spectral widths of ground returns were observed to have mean values ranging from 0.1 to 0.7 m/s. These values are similar to the 0.1 to 0.5 m/s range observed by Zrnic and Hamadi surrounding the NSSL radar in Norman, Oklahoma ([31];pp.10-15).

The biasing effects that these clutter returns introduce to weather parameter estimates can be effectively minimized by filtering the incoming echo samples with short recursive dc notch elliptic filters. For accurate weather parameter estimation, the power of the ground returns must be 15 to 20 dB below that of the desired weather returns. For weak weather returns and strong ground returns, the level of clutter suppression needed can be in excess of 70 dB. This level of suppression is not always possible at the CHILL since its antenna scan spectrum has side lobes that are only 48 dB down. These side lobes tend to smear the

clutter spectrum such that no more than about 48 dB of clutter suppression can be achieved. Good results were obtained using filters with 50 dB of suppression in the stopband and variable notch widths.

Determination of the notch width needed for a desired performance involves a number of factors. The first is determining the spectral width of the clutter to be suppressed. This can be done by first determining the spectral width of the type of clutter to be suppressed. Typical values of spectral width for most ground targets are in the 0.3 to 0.5 m/s range while crops, giving much weaker returned powers, have spectral widths closer to 1 m/s. To this spectral width value must be added broadening due to antenna rotation and any secondary factors such as wind speed. Once the width of the clutter is determined, the level of desired suppression needs to be known. As already mentioned, clutter powers need to be 15 to 20 dB below the power of the weakest weather echo for which accurate weather parameter estimates are to be obtained. The next step is to determine the weakest weather echoes that are to be analyzed. Once this is accomplished, the clutter power level for the desired range and elevation must be estimated. This can be done from curves such as those found in Fig. 3.10. The desired level of suppression is then that which will put the clutter echo power 15 to 20 dB below the weakest weather echo power. The notch width can then be set so that the filter removes enough of the clutter spectrum's spectral skirts to achieve the desired level of suppression. For example, the notch width needs to be 4.8 times the clutter's

spectral width for 50 dB of suppression. This calculation was made using (4.13). The notch widths must be kept as narrow as possible to prevent unnecessarily removing weather echoes.

The whole subject of clutter characteristics needs more development than was possible for this study. By taking much larger data sets with a better knowledge of the clutter environment, better spectral width estimates could be obtained. Secondary spectral width broadening mechanisms such as wind speed are known to be small, around 0.2 m/s, but a good model should be developed. Also, by using a larger data set, average clutter powers for different elevations and ranges could be fit with curves. Then maps could be used to automatically compensate, by narrowing the filter notch widths, for the decrease in clutter power as the beam elevates. The current filters at the CHILL can be independently set for each range gate but the processor in which the filtering takes place is ignorant of the antenna elevation. This means that the filters can not be turned off when the elevation angles are large enough that the ground returns are negligible.

Another area for further work in ground clutter reduction involves the use of dual polarization. During normal operation, all of the pulses leave the antenna with a horizontal polarization. However, the CHILL is capable of operating in a dual polarization mode where every fourth pulse leaves the antenna with a vertical polarization. The horizontal pulses are used for velocity and spectral width estimations, while reflectivity estimates are calculated for

each polarization. By comparing the two reflectivities, information can be deduced about the scatterer's shape.

The horizontally and vertically polarized pulses must be processed separately. This means that the horizontal pulses are non-contiguous, every fourth pulse being missed. This is a problem for the linear filters. Some method of inserting an estimated pulse for the missing one is needed. This may possibly be done with maximum-entropy methods. A problem also exists in processing the vertical pulse train since its unambiguous velocity is only a fourth of that of the horizontal pulse train. This means weather spectra may easily alias into a filter's notch.

Concerning the issue of spectral width estimation in the time domain, two estimators were studied. The first involved the ratio of the magnitudes of the zero and first lags of the echo's correlation. The second estimator involved the ratio of the magnitudes of the first and second lags. In addition to the previously documented superior performance of the second estimator's performance in the presence of white noise, two additional performance measures were examined. The first analyzed the estimator's performance in the presence of a gain imbalance between the in-phase and quadrature components. The second performance measure compared the two estimators when weather echoes were contaminated with ground clutter. The second estimator proved superior in both of these instances. Therefore, unless it is computationally infeasible, the second estimator is preferable to the first for determining spectral width.

REFERENCES

[1] Anderson, J. R., "Evaluating Ground Clutter Filters for Weather Radars," *Preprints 20th Conf. Radar Meteor.,* Am. Meteor. Soc., Boston, MA, pp. 314-318, Nov. 1981.

[2] Berkowitz, R. S. (ed.), *Modern Radar Analysis, Evaluation,and System Design.*New York, NY: John Wiley & Sons, 1965, pp. 6-8, 478-480.

[3] Doviak, R. J., Zrnic, D. S., and Sirmans, D. , "Doppler Weather Radar," *Proc. IEEE,* vol. 67, pp. 1522-1553, Nov. 1979.

[4] Doviak, R. J., and Zrnic, D. S., *Doppler Radar and Weather Observations.* New York, NY: Academic Press, 1984, pp. 49-51, 81-87, 94-113, 441, 445-447.

[5] Evans, J. E., "Ground Clutter Cancellation for the NEXRAD System," Project Rep. ATC-122, Lincoln Laboratory, MIT, Lexington, MA, Oct. 1983, pp. 6.1-6.8, 7.1-7.23.

[6] Fletcher, R. H., and Burlage, D. W., "An Initialization Technique for Improved MTI Performance in Phased Array Radars," *Proc. IEEE,* vol. 60, pp. 1551-1552, Dec. 1972.

[7] Groginsky, H. L., "Pulse-Pair Estimation of Doppler Spectrum Parameters," *Preprints 15th Conf. Radar Meteor.,* Am. Meteor. Soc, Boston, MA, pp. 233-236, Oct. 1972.

[8] Groginsky, H. L., and Glover, K., "Weather Radar Canceler Design," *Preprints 19th Conf. Radar Meteor.,* Am. Meteor. Soc, Boston, MA, pp. 192-201, April 1980.

[9] Haykin, S., Kesler, S., and Currie, B., "An Experimental Classification of Radar Clutter," *Proc. IEEE,* vol. 67, pp. 332-333, Feb. 1979.

[10] Haykin, S., Kesler, S., and Currie, B., "Maximum-Entropy Spectral Analysis of Radar Clutter," *Proc. IEEE,* vol. 70, pp. 953-962, Sept. 1982.

[11] Hildebrand, P. H., and Sekhon, R.S., "Objective Determination of the Noise Level in Doppler Spectra," *J. Appl. Meteor.,* vol. 13, pp. 808-811, Oct. 1974.

[12] Long, M., *Radar Reflectivity of Land and Sea.* Lexington, MA: Lexington Books, 1975, pp. 237-238.

[13] Ludeman, L., *Fundamentals of Digital Signal Processing.* New York, NY: Harper & Row, 1986. pp. 149-160.

[14] Natheson, F. E., *Radar Design Principles.* McGraw-Hill, New York, NY: McGraw-Hill, 1969. pp. 345-347.

[15] Papoulis, A., *The Fourier Integral and Its Applications.* New York, NY: McGraw-Hill, 1962, p. 14.

[16] Papoulis, A., *Probability, Random Variables, and Stochastic Processes.* New York, NY: McGraw-Hill, 1984, pp. 134-141.

[17] Roberts, R. A., and Mullis, C. T., *Digital Signal Processing.* Reading, MA: Addison-Wesley, 1987, pp. 290-297.

[18] Rummler, W. D., "Introduction of a New Estimator for Velocity Spectral Parameters," Tech. Memo MM-68-4121-5, Bell Telephone Labs., Whippany, NJ, 1968.

[19] Rummler, W. D., "Accuracy of Spectral Width Estimators Using Pulse-Pair Waveforms," Tech. Memo MM-68-4121-14, Bell Telephone Labs., Whippany, NJ, 1968.

[20] Rummler, W. D., "Two Pulse Spectral Measurements," Tech. Memo MM-68-4121-14, Bell Telephone Labs., Whippany, NJ, 1968.

[21] Sekine, M., Ohitani, S., Musha, T., Irabu, T. Kiuchi, E., Hagisawa, T., and Tomita, Y., "Weibull-Distributed Ground Clutter," *IEEE Trans. Aerosp. Electron. Syst.,* vol AES-17, pp. 596-598, July 1981.

[22] Sirmans, D., and Bumgarner, B., "Numerical Comparisons of Five Mean Frequency Estimators," J. *Appl. Meteor.,* vol. 14, pp. 991-1003, Sept. 1975.

[23] Sirmans, D., and Bumgarner, B., "Estimation of Spectral Density Mean and Variance by Covariance Argument Techniques," *Preprints 16th Conf. Radar Meteor.,* Am. Meteor. Soc, Boston, MA, pp. 6-13, Apr. 1972.

[24] Skolnik, M. I., *Introduction to Radar Systems.* New York,NY: McGraw-Hill, 1980, pp. 450-456.

[25] Srivastava, R. C., Jameson, A. R., and Hidebrand, P. H., "Time-Domain Computation Of Mean and Variance of Doppler Spectra," *J. Appl. Meteor.,* vol. 18, pp. 189-194, Feb. 1979.

[26] Wallace, J. M., and Hobbs, P. V., *Atmospheric Science, An Introductory Survey.* New York, NY: Academic Press, 1977. pp. 215-274.

[27] Zrnic, D. S., "Simulation of Weatherlike Doppler Spectra and Signals," *J. Appl. Meteor.,* vol. 14, pp. 619-620, June 1975.

[28] Zrnic, D. S., "Spectral Moment Estimates from Correlated Pulse Pairs," *IEEE Trans. Aerosp. Electron. Syst.,* vol. AES-13, pp. 344-354, July 1977.

[29] Zrnic, D. S., "Estimation of Spectral Moments for Weather Echoes," *IEEE Trans. Geosci. Electron.*, vol. GE-17, pp. 113-128, Oct. 1979.

[30] Zrnic, D. S., "Spectral Statistics for Complex Colored Discrete-Time Sequences," *IEEE Trans. Acoust., Speech, Signal Processing,* vol. ASSP-28, pp. 596-599, Oct. 1980.

[31] Zrnic, D. S., and Hamadi, S., "Considerations for the Design of Ground Clutter Cancelers for Weather Radar," Interm Rep. DOT/FAA/RD-81/72, NSSL, Norman, OK, Feb. 1981, pp. 10-41.

APPENDIX I

GENERAL CHARACTERISTICS OF THE CHILL RADAR SYSTEM

Antenna

Shape	Parabolic
Diameter	8.5 m
One-Way Two-Sided Half-Power Beamwidth	0.96°
Gain	43.3 dB
First Side-Lobe Level	-25 dB
Polarization	Horizontal and vertical on a pulse by pulse basis
Azimuthal Antenna Rotation Rate	0 - 30 degrees/sec

Transmitter

Wavelength	11.0 cm
Frequency	2.73 GHz
Peak Power	1 MW
Pulse Width	1/4, 1/2 or 1 μs
Pulse Repetition Time	800 - 2500 μs
Maximum Unambiguous Range	375 km
Maximum Unambiguous Velocity	34.4 m/s

Receiver

Noise Figure	4.0 dB
Transfer Function	Linear
Dynamic Range	90 dB
3 dB, Bandwidth	Varies with pulse width
Minimum Detectable Signal (SNR = 1)	-110

Data Acquisition

# of Range Gates	1024 - 4096
Range Gate Spacing	1/4, 1/2 or 1 μs
Recorded Word Length	
Velocity	8 bits (2's complement)
Width	8 bits (binary)
Intensity	8 bits (binary)
# of Samples in Estimate	Arbitrary

APPENDIX II

SPECIFICS OF THE CHILL SIGNAL FLOW

A simplified overview of signal flow at the CHILL radar is shown in Fig. AII.1. The IF signal enters the system through the coherent quadrature receiver in the upper-left corner of the diagram. This receiver is expanded upon in Fig. AII.2. Since weather echoes can span a large dynamic range, use of only a linear receiver is usually insufficient. The 10 bit A/D converters in the linear channel of the CHILL receiver give it a dynamic range of about 30 dB. By using the incoherent logarithmic channel to switch attenuators into the linear channel, the dynamic range of the total receiver is increased to over 100 dB. Since these attenuators change the gain of the linear channel, the logarithmic channel along with the switchable attenuators are sometimes referred to as the automatic gain control circuitry (AGC). The delay line in the linear channel enables the attenuators to be switched in before the signal reaches the coherent quadrature demodulator.

The three-bit unsigned samples of the incoherent logarithmic channel provide the common exponent for the floating point I and *Q* samples. The 10-bit samples of the linear channel provide the I and *Q* signs and mantissas. Before leaving the receiver section, a word must be said about A/D sample times. Although the mantissa A/D samples every 250 ns, the exponent A/D samples only every 1 μs. This

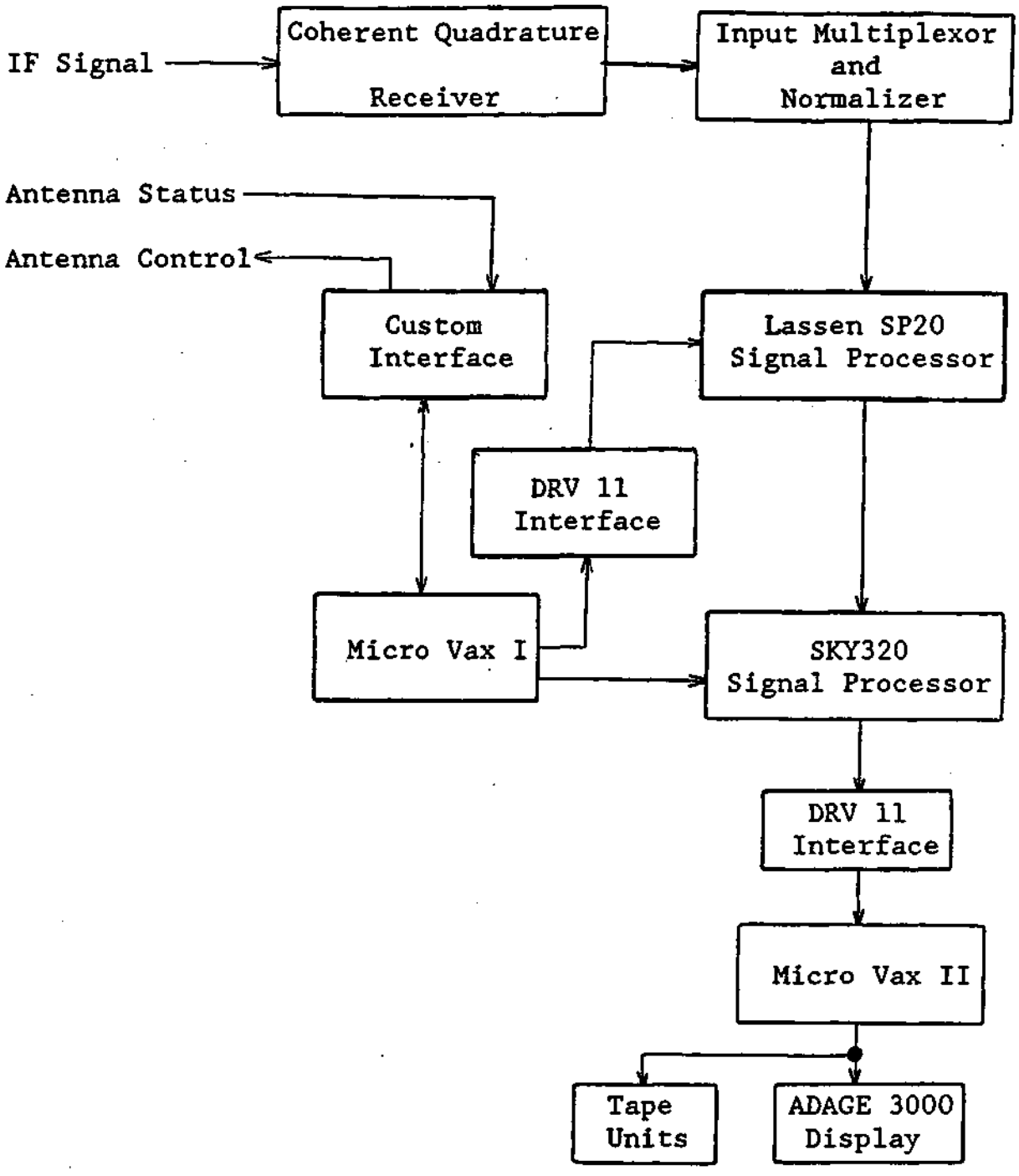


Figure AII.1. Block diagram of the CHILL's signal flow.

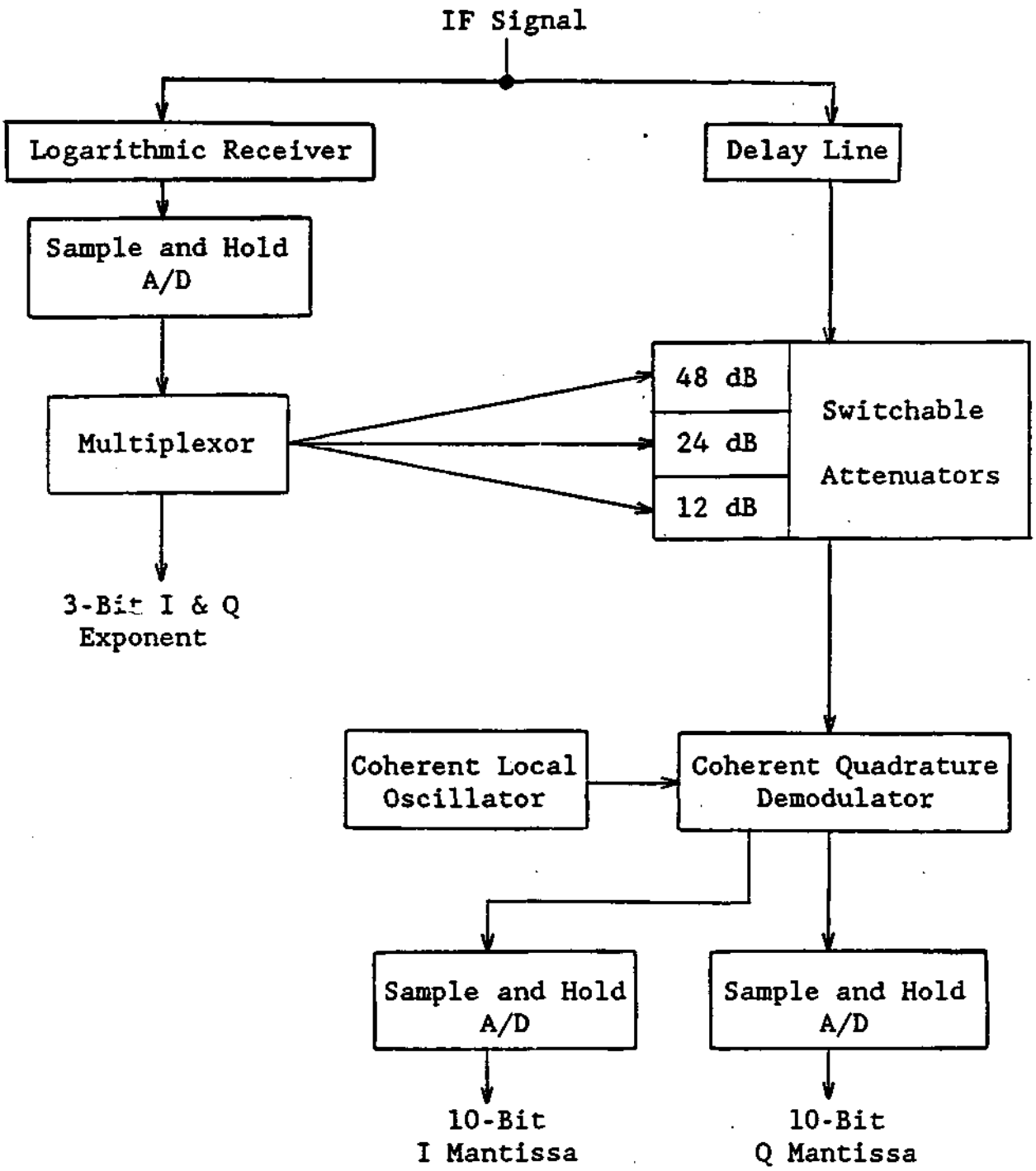


Figure AII.2. Block diagram of the CHILL's receiver.

can be a problem when adjacent 250 ns samples span too large a range. When this happens, the exponent cannot be correct for both samples.

Once samples exit the receiver, they enter the input multiplexor and normalizer. Here the samples are converted into the IEEE Standard 754 floating-point format used by the SP20 signal processor. The normalizer will also block an average of two or four samples to facilitate 1/2 and 1 μs range gating.

From the normalizer, the data are sent to the SP20 signal processor. The SP20 signal processor consists of four independent parallel processors connected by three common 32-bit busses. Each processor contains a 32-bit floating-point multiplier, a 32-bit floating-point adder, 64K of 32-bit data RAM, 8 32-bit accumulators and 8K of program RAM. Each processor is capable of 10-million floating-point multiplications and 10-million floating-point additions every second. The processors are programmable by a low level assembly language unique to the SP20. Programs may be downloaded from the Micro Vax I.

Two of the processor cards are allocated for ground clutter cancellation. The remaining two cards are used for pulse-pair processing. A fifth card exists without the multiplier or adder chips, which is used for interfacing the SP20 with the outside world.

After data are processed in the SP20, pulse-pair estimates leave and enter the SKY320. The SKY320 is a 16 bit fixed-point signal processor. Its main purpose is to merge housekeeping data, acquired from the Micro Vax I, with the

pulse-pair estimates. The housekeeping data contain date, time and antenna position.

When the data leave the SKY320, thay are routed to the Micro Vax II. The Micro Vax II buffers the data to tape units and the ADAGE 3000 color display system. A computer in the ADAGE 3000 has several functions. One is that it unpacks the incoming data and performs a conversion from polar to Cartesian coordinates. It also strips pulse-pair power estimates of their range-squared dependence and performs the necessary reflectivity calibrations.

APPENDIX III

FOUR-POLE ELLIPTIC FILTER PROGRAM LISTING

SP20 Microcode Assembier V 2.00

Input file name: filter4pole.ass
Mod file name: filter4pole.mod

```

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!                                                                          !
!       Filename   : Fourpole.ass                                          !
!       Author     : James T. Peltier                                      !
!       Date       : December 20, 1988                                     !
!                                                                          !
!       Four Pole Elliptical Filter                                        !
!       Cascade Realization for Use on SP20.                               !
!       Set up for 2048 Hits (maximum), one every 100 nS.                  !
!       Quadrature term 100 nS after in-phase term.                        !
!       Input on TGC bus. Output on TGB bus.                               !
!                                                                          !
!       System Equations:                                                  !
!                                                                          !
!         R(n) = GU(n) + K3R(n-1) - K4R(n-2)                               !
!         S(n) =  R(n) - K1R(n-1)  +  R(n-2) + K5S(n-1) - K6S(n-2)         !
!         Y(n) =  S(n) - K2S(n-1)  +  S(n-2)                               !
!                                                                          !
!       A note on comments:                                                !
!                                                                          !
!         Instructions are numbered in a data loop form.                   !
!           Data arrives on instructions -30,-20,-10,0,10.                 !
!           Main processing occurs on 0 - 9 loop.                          !
!           Terms calculated from data obtained in current                 !
!             "gate" is denoted with a 0, terms calculated                 !
!             for data arriving in the next gate are denoted               !
!             with a 1 etc.                                                !
!                                                                          !
!             e.g.   S(-1,n-2)   denotes S(n-2) for data that              !
!                                arrived last cycle (gate).                !
!                                                                          !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

! Labels:

:R_1_AD:      = 0X1000
:R_2_AD:      = 0X2000
:S_1_AD:      = 0X3000
:S_2_AD:      = 0X4000
:K1_AD:       = 0X5000
:K2_AD:       = 0X6000
:K3_AD:       = 0X7000
:K4_AD:       = 0X8000
:K5_AD:       = 0X9000
:K6_AD:       = 0XA000
:G_AD:        = 0XB000
:BANK:        = 0XC000

:GATES:  => 1024
:FILTER: => GATES/8
:WIDTH:  => 1

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!                                                                          !
!       In-Phase Filter                                                    !
!                                                                          !
!         Input commences 500 nS after flag falls.                         !
!         Output lags input by 3 uS.                                       !
```

```
!                                                                    !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!                                                                    !
!       Memory Map                                                   !
!                                                                    !
!               1000-17FF : R(n-1)  Address Generator BG4 - READ     !
!                                                     AG7 - WRITE    !
!               2000-27FF : R(n-2)  Address Generator BG1 - READ     !
!                                                     AG6 - WRITE    !
!               3000-37FF : S(n-1)  Address Generator BG0 - READ     !
!                                                     AG3 - WRITE    !
!               4000-47FF : S(n-2)  Address Generator BG2 - READ     !
!                                                     AG5 - WRITE    !
!               5000-57FF : K1      Address Generator AG1 - READ     !
!               6000-67FF : K2      Address Generator AG2 - READ     !
!               7000-77FF : K3      Address Generator BG3 - READ     !
!               8000-87FF : K4      Address Generator AG4 - READ     !
!               9000-97FF : K5      Address Generator BG5 - READ     !
!               A000-A7FF : K6      Address Generator BG6 - READ     !
!               B000-B7FF : G       Address Generator AG0 - READ     !
!               C000-     : Various Filter Coefficients              !
!                                                                    !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!


#CARD 3

*0010

!       Instruction P0 (Preliminary)

SCONT,GARST,GBRST;                        ! Reset address generators
;                                         !
;                                         !
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0010 0000 0000 0000 0000 0003 0042 0000 013b 0303 7878 0000 2222 0000 8080


!       Instruction P1

SJSAU<FILTER_GENESIS_I>;                  ! Dump banks of filter coefficients
;                                         !  into memory
;                                         !
          SDATA=0094  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0011 7000 0094 0000 0000 0000 0042 0000 013b 0303 7878 0000 2222 0000 8080


!       Instruction P2

SCONT,GAFYRTR0<BANK+((WIDTH-1)*7)>;       ! Load AG0 with pointer to a specific
;                                         !  set of filter coefficients
;                                         !
          SDATA=0000  GADATA=C000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0012 0000 0000 0000 0006 1000 8842 0000 013b 0303 7878 0000 2222 0000 8080


!       Instruction P3

SCONT,GAYINC00;                           !
;                                         !
MA;                                       !
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
```

```
0013 0000 0000 0000 0000 0088 8042 0000 013b 0303 7878 0000 2222 0000 8080

0112
0113  !       Instruction P4
0114
0115  SCONT,GAYINC00;                          !
0116  A00000A,MAR,LM;                          ! G -> A0
0117  MA;                                      !
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0014 0000 0000 0000 0000 0088 8042 0000 013b 0303 78b8 0000 2222 0000 8000

0118
0119  !       Instruction P5
0120
0121  SCONT,GAYINC00;                          !
0122  A10000A,MAR,LM;                          ! K1 -> A1
0123  MA;                                      !
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0015 0000 0000 0000 0000 0088 8042 0000 013b 0303 78b8 0000 2223 0000 8000

0124
0125  !       Instruction P6
0126
0127  SCONT,GAYINC00;                          !
0128  A20000A,MAR,LM;                          ! K2 -> A2
0129  MA;                                      !
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0016 0000 0000 0000 0000 0088 8042 0000 013b 0303 78b8 0000 2222 0080 8000

0130
0131  !       Instruction P7
0132
0133  SCONT,GAYINC00;                          !
0134  A30000A,MAR,LM;                          ! K3 -> A3
0135  MA;                                      !
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0017 0000 0000 0000 0000 0088 8042 0000 013b 0303 78b8 0000 2223 0080 8000

0136
0137  !       Instruction P8
0138
0139  SCONT,GAYINC00;                          !
0140  A40000A,MAR,LM;                          ! K4 -> A4
0141  MA;                                      !
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0018 0000 0000 0000 0000 0088 8042 0000 013b 0303 78b8 0000 2222 0040 8000

0142
0143  !       Instruction P9
0144
0145  SCONT,GAYINC00;                          !
0146  A50000A,MAR,LM;                          ! K5 -> A5
0147  MA;                                      !
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0019 0000 0000 0000 0000 0088 8042 0000 013b 0303 78b8 0000 2223 0040 8000

0148
0149  !       Instruction P10
0150
0151  SCONT;                                   !
```

```
0152  A60000A,MAR,LM;                          ! K6 -> A6_
0153  ;                                        !
                SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9    10   11   12   13   14
001a 0000 0000 0000 0000 0000 0042 0000 013b 0303 78b8 0000 2222 00c0 8000

0154
0155  !       Instruction P11
0156
0157  SCONT,GAFYRTR0<G_AD>,GBFYRTR0<S_1_AD>;  ! Initialize address generators
0158  ;                                        !
0159  ;                                        !
                SDATA=0000  GADATA=B000  GBDATA=3000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9    10   11   12   13   14
001b 0000 0000 0030 001a 9004 c842 0000 013b 0303 7878 0000 2222 0000 8080

0160
0161  !       Instruction P12
0162
0163  SCONT,GAFYRTR1<K1_AD>,GBFYRTR1<R_2_AD>; !
0164  ;                                        !
0165  ;                                        !
                SDATA=0000  GADATA=5000  GBDATA=2000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9    10   11   12   13   14
001c 0000 0000 0020 0014 9007 c842 0000 013b 0303 7878 0000 2222 0000 8080

0166
0167  !       Instruction P13
0168
0169  SCONT,GAFYRTR2<K2_AD>,GBFYRTR2<S_2_AD>; !
0170  ;                                        !
0171  ;                                        !
                SDATA=0000  GADATA=6000  GBDATA=4000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9    10   11   12   13   14
001d 0000 0000 0040 000c 9304 c842 0000 013b 0303 7878 0000 2222 0000 8080

0172
0173  !       Instruction P14
0174
0175  SCONT,GAFYRTR4<K4_AD>,GBFYRTR3<K3_AD>;  !
0176  ;                                        !
0177  ;                                        !
                SDATA=0000  GADATA=8000  GBDATA=7000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9    10   11   12   13   14
001e 0000 0000 0070 0002 9605 c842 0000 013b 0303 7878 0000 2222 0000 8080

0178
0179  !       Instruction P15
0180
0181  SCONT,GBFYRTR4<R_1_AD>;                  !
0182  ;                                        !
0183  ;                                        !
                SDATA=0000  GADATA=0000  GBDATA=1000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9    10   11   12   13   14
001f 0000 0000 0010 0000 8804 4042 0000 013b 0303 7878 0000 2222 0000 8080

0184
0185  !       Instruction P16
0186
0187  SCONT,GBFYRTR5<K5_AD>;                   !
0188  ;                                        !
0189  ;                                        !
                SDATA=0000  GADATA=0000  GBDATA=9000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9    10   11   12   13   14
0020 0000 0000 0090 0000 8805 4042 0000 013b 0303 7878 0000 2222 0000 8080
```

```
0190
0191  !       Instruction P17
0192
0193  SCONT,GBFYRTR6<K6_AD>;                          !
0194  ;                                               !
0195  ;                                               !
            SDATA=0000  GADATA=0000  GBDATA=A000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0021 0000 0000 00a0 0000 8a04 4042 0000 013b 0303 7878 0000 2222 0000 8080

0196
0197  !       Instruction P18
0198
0199  SWDC0<GATES-1>;                                 ! Set up Cntr0 for GATES number
0200                                                  !  of loops
0201  ;                                               !
0202  ;                                               !
            SDATA=03FF  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0022 3800 03ff 0000 0000 0000 0042 0000 013b 0303 7878 0000 2222 0000 8080

0203
0204  :INIT_STOR_1:
0205
0206  !       Instruction P19
0207
0208  SJDC0<DONE_STOR_1>,GBYINC04;                    !
0209  LL00000000;                                     !
0210  MBL;                                            !
            SDATA=0028  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0023 4c00 0028 0000 0000 0810 4042 0000 017b 0b01 7c78 0000 2222 0000 8080

0211
0212  !       Instruction P20
0213
0214  SCONT,GBYINC01;                                 !
0215  LL00000000,MBW;                                 ! 0 -> R(n-1)
0216  MBL;                                            !
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0024 0000 0000 0000 0000 0011 4042 0000 017b 0b09 7c7a 0000 2222 0000 8080

0217
0218  !       Instruction P21
0219
0220  SCONT,GBYINC00;                                 !
0221  LL00000000,MBW;                                 ! 0 -> R(n-2)
0222  MBL;                                            !
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0025 0000 0000 0000 0000 0010 4042 0000 017b 0b09 7c7a 0000 2222 0000 8080

0223
0224  !       Instruction P22
0225
0226  SCONT,GBYINC02;                                 !
0227  LL00000000,MBW;                                 ! 0 -> S(n-1)
0228  MBL;                                            !
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0026 0000 0000 0000 0000 0210 4042 0000 017b 0b09 7c7a 0000 2222 0000 8080

0229
0230  !       Instruction P23
0231
```

```
0232  SJDAU<INIT_STOR_I>;                              !
0233  MBW;                                             ! 0 -> S(n-2)
0234  ;                                                !
            SDATA=0023  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0027 7300 0023 0000 0000 0000 0042 0000 013b 030b 787a 0000 2222 0000 8080

0235
0236  :DONE_STOR_I:
0237
0238  !       Instruction P24
0239
0240  SWDC0<FILTER-1>;                                 ! Set Cntr0 for # of filter loops
0241  ;                                                !
0242  ;                                                !
            SDATA=007F  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0028 3800 007f 0000 0000 0000 0042 0000 013b 0303 7878 0000 2222 0000 8080

0243
0244  !       Instruction P25
0245
0246  SWDC1<GATES-FILTER-1>;                           ! Set Cntr1 for # of unfilter loops
0247  ;                                                !
0248  ;                                                !
            SDATA=0381  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0029 3900 0381 0000 0000 0000 0042 0000 013b 0303 7878 0000 2222 0000 8080

0249
0250  :INIT_NTCH_COEF_I:
0251
0252  !       Instruction P26
0253
0254  SJDC0<INIT_AP_COEF_I>,GAYINC00;                  ! If Cntr < 0 then done
0255  A00000,LD;                                       ! G -> LB
0256  MAL;                                             ! G -> IML
            SDATA=0032  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
002a 4c00 0032 0000 0000 0088 8042 0000 013b 0302 7c78 0000 2222 0000 8040

0257
0258  !       Instruction P27
0259
0260  SCONT,GAYINC01;                                  !
0261  A10000,LD,MAW;                                   ! K1 -> LB, G -> M
0262  MAL;                                             ! K1 -> IML
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
002b 0000 0000 0000 0000 008a 8042 0000 013b 0302 7c7a 0000 2223 0000 8040

0263
0264  !       Instruction P28
0265
0266  SCONT,GAYINC02;                                  !
0267  A20000,LD,MAW;                                   ! K2 -> LB, K1 -> M
0268  MAL;                                             ! K2 -> IML
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
002c 0000 0000 0000 0000 0188 8042 0000 013b 0302 7c7a 0000 2222 0080 8040

0269
0270  !       Instruction P29
0271
0272  SCONT,GBYINC03;                                  !
0273  A30000,LD,MAW;                                   ! K3 -> LB, K2 -> M
```

```
0274  MBL;                                          ! K3 -> IML
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
002d 0000 0000 0000 0000 0211 4042 0000 017b 0b02 7c7a 0000 2223 0080 8040

0275
0276  !       Instruction P30
0277
0278  SCONT,GAYINC04;                               !
0279  A40000,LD,MAW;                                ! K4 -> LB, K3 -> M
0280  MAL;                                          ! K4 -> IML
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
002e 0000 0000 0000 0000 0488 8042 0000 013b 0302 7c7a 0000 2222 0040 8040

0281
0282  !       Instruction P31
0283
0284  SCONT,GBYINC05;                               !
0285  A50000,LD,MAW;                                ! K5 -> LB, K4 -> M
0286  MBL;                                          ! K5 -> IML
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
002f 0000 0000 0000 0000 0811 4042 0000 017b 0b02 7c7a 0000 2223 0040 8040

0287
0288  !       Instruction P32
0289
0290  SCONT,GBYINC06;                               !
0291  A60000,LD,MAW;                                ! K6 -> LB, K5 -> M
0292  MBL;                                          ! K6 -> IML
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0030 0000 0000 0000 0000 0a10 4042 0000 017b 0b02 7c7a 0000 2222 00c0 8040

0293
0294  !       Instruction P33
0295
0296  SJDAU<INIT_NTCH_COEF_I>;                      !
0297  MAW;                                          ! K6 -> M
0298  ;                                             !
            SDATA=002A  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0031 7300 002a 0000 0000 0000 0042 0000 013b 0303 787a 0000 2222 0000 8080

0299
0300  :INIT_AP_COEF_I:
0301
0302  !       Instruction P34
0303
0304  SCONT,GAYDEC00;                               ! Adjust G pointer
0305  ;                                             !
0306  ;                                             !
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0032 0000 0000 0000 0000 0088 0042 0000 013b 0303 7878 0000 2222 0000 8080

0307
0308  :NEXT_AP_COEF_I:
0309
0310  !       Instruction P35
0311
0312  SJDC1<NEW_RAY_I>,GAYINC00;                    ! If Cntr < 0 then done
0313  LF1.000000;                                   ! G -> LB
0314  MAL;                                          ! G -> IML
            SDATA=003B  GADATA=0000  GBDATA=0000  LDATA=3F800000
```

```
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0033 4d00 003b 013f 0000 0088 8042 0000 013b 0301 7c78 0000 2222 0000 8080

0315
0316  !       Instruction P36
0317
0318  SCONT,GAYINC01;                          !
0319  LL00000000,MAW;                          ! K1 -> LB, G -> M
0320  MAL;                                     ! K1 -> IML
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0034 0000 0000 0000 0000 008a 8042 0000 013b 0301 7c7a 0000 2222 0000 8080

0321
0322  !       Instruction P37
0323
0324  SCONT,GAYINC02;                          !
0325  LL00000000,MAW;                          ! K2 -> LB, K1 -> M
0326  MAL;                                     ! K2 -> IML
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0035 0000 0000 0000 0000 0188 8042 0000 013b 0301 7c7a 0000 2222 0000 8080

0327
0328  !       Instruction P38
0329
0330  SCONT,GBYINC03;                          !
0331  LL00000000,MAW;                          ! K3 -> LB, K2 -> M
0332  MBL;                                     ! K3 -> IML
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0036 0000 0000 0000 0000 0211 4042 0000 017b 0b01 7c7a 0000 2222 0000 8080

0333
0334  !       Instruction P39
0335
0336  SCONT,GAYINC04;                          !
0337  LF1.000000,MAW;                          ! K4 -> LB, K3 -> M
0338  MAL;                                     ! K4 -> IML
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=3F800000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0037 0000 0000 013f 0000 0488 8042 0000 013b 0301 7c7a 0000 2222 0000 8080

0339
0340  !       Instruction P40
0341
0342  SCONT,GBYINC05;                          !
0343  LL00000000,MAW;                          ! K5 -> LB, K4 -> M
0344  MBL;                                     ! K5 -> IML
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0038 0000 0000 0000 0000 0811 4042 0000 017b 0b01 7c7a 0000 2222 0000 8080

0345
0346  !       Instruction P41
0347
0348  SCONT,GBYINC06;                          !
0349  LF1.000000,MAW;                          ! K6 -> LB, K5 -> M
0350  MBL;                                     ! K6 -> IML
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=3F800000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0039 0000 0000 013f 0000 0a10 4042 0000 017b 0b01 7c7a 0000 2222 0000 8080

0351
0352  !       Instruction P42
0353
```

```
0354  SJDAU<NEXT_AP_COEF_I>;                          !
0355  MAW;                                            ! K6 -> M
0356  ;                                               !
            SDATA=0033  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
003a 7300 0033 0000 0000 0000 0042 0000 013b 0303 787a 0000 2222 0000 8080

0357
0358  :NEW_RAY_I:
0359
0360  !       Instruction S0 (Set up)
0361
0362  SWDC0<GATES-5>,GAFYRTR0<G_AD>,GBFYRTR0<S_1_AD>; ! Set Cntr0 for Gates-4 loops.
0363                                                  ! Note that 3 inputs arrive
0364                                                  !  before main loop, and 1
0365                                                  !  while exiting main loop.
0366  ;                                               !
0367  ;                                               !
            SDATA=03FB  GADATA=B000  GBDATA=3000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
003b 3800 03fb 0030 001a 9004 c842 0000 013b 0303 7878 0000 2222 0000 8080

0368
0369  !       Instruction S1
0370
0371  SCONT,GAFYRTR1<K1_AD>,GBFYRTR1<R_2_AD>;          !
0372  ;                                               !
0373  ;                                               !
            SDATA=0000  GADATA=5000  GBDATA=2000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
003c 0000 0000 0020 0014 9007 c842 0000 013b 0303 7878 0000 2222 0000 8080

0374
0375  !       Instruction S2
0376
0377  SCONT,GAFYRTR2<K2_AD>,GBFYRTR2<S_2_AD>;          !
0378  ;                                               !
0379  ;                                               !
            SDATA=0000  GADATA=6000  GBDATA=4000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
003d 0000 0000 0040 000c 9304 c842 0000 013b 0303 7878 0000 2222 0000 8080

0380
0381  !       Instruction S3
0382
0383  SCONT,GAFYRTR3<S_1_AD>,GBFYRTR3<K3_AD>;          !
0384  ;                                               !
0385  ;                                               !
            SDATA=0000  GADATA=3000  GBDATA=7000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
003e 0000 0000 0070 0018 9307 c842 0000 013b 0303 7878 0000 2222 0000 8080

0386
0387  !       Instruction S4
0388
0389  SCONT,GAFYRTR4<K4_AD>,GBFYRTR4<R_1_AD>;          !
0390  ;                                               !
0391  ;                                               !
            SDATA=0000  GADATA=8000  GBDATA=1000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
003f 0000 0000 0010 0002 9c04 c842 0000 013b 0303 7878 0000 2222 0000 8080

0392
0393  !       Instruction S5
0394
0395  SCONT,GAFYRTR5<S_2_AD>,GBFYRTR5<K5_AD>;          !
```

```
0396  ;                                                !
0397  ;                                                !
            SDATA=0000  GADATA=4000  GBDATA=9000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0040 0000 0000 0090 0004 9c07 c842 0000 013b 0303 7878 0000 2222 0000 8080

0398
0399  !       Instruction S6
0400
0401  SCONT,GAFYRTR6<R_2_AD>,GBFYRTR6<K6_AD>; !
0402  ;                                                !
0403  ;                                                !
            SDATA=0000  GADATA=2000  GBDATA=A000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0041 0000 0000 00a0 0008 9f04 c842 0000 013b 0303 7878 0000 2222 0000 8080

0404
0405  !       Instruction S7
0406
0407  SCONT,GAFYRTR7<R_1_AD>;                          !
0408  ;                                                !
0409  ;                                                !
            SDATA=0000  GADATA=1000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0042 0000 0000 0000 0010 1502 8842 0000 013b 0303 7878 0000 2222 0000 8080

0410
0411  !       Instruction S8
0412
0413  SCONT,GAYINC01,GBYINC04;                         !
0414  ;                                                !
0415  MA;                                              !
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0043 0000 0000 0000 0000 089a c042 0000 017b 0303 7878 0000 2222 0000 8080

0416
0417  !       Instruction S9
0418
0419  SCONT;                                           !
0420  A50000A,MBR,LM;                                  ! K1(1) -> A5
0421  A20050A,MAR,LM;                                  ! R(1,n-1) -> A2, K1(1) -> X
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0044 0000 0000 0000 0000 0000 0042 0000 013b 030b b8b8 0000 2223 a840 0000

0422
0423  !       Instruction S10
0424
0425  SCONT,GBYINC00,XSA0B1S10;                        !
0426  A00020;                                          ! R(1,n-1) -> X
0427  MA;                                              !
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0045 0000 0000 0000 0000 0010 4183 0000 017b 0303 7878 0000 2222 0010 8080

0428
0429  !       Instruction S11
0430
0431  SCONT,GAYINC04,GBYINC03,XSS01;                   ! Form K1R(1,n-1)
0432  MB;                                              !
0433  A10000A,MBR,LM;                                  ! S(1,n-1) -> A1,
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0046 0000 0000 0000 0000 0699 c140 0000 017b 0b0b b878 0000 2322 0000 0080
```

```
0434
0435  !       Instruction S12
0436
0437  SCONT;                                          !
0438  A00000A,MAR,LM;                                 ! K3(1) -> A0
0439  A30000A,MAR,LM;                                 ! K4(1) -> A3
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0047 0000 0000 0000 0000 0000 0042 0000 013b 0303 b8b8 0000 2322 8000 0000

0440
0441  !       Instruction S13
0442
0443  SCONT;                                          !
0444  A06000B,XMSW;                                   ! K1R(1,n-1) -> A6,
0445  ;                                               !
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0048 0000 0000 0000 0000 0000 0042 0000 013b 0303 7878 0003 2242 0000 8080

0446
0447  !       Instruction WAIT
0448
0449  SWAITN,GBYRTR1;                                 !
0450  ;                                               !
0451  A00000;                                         ! K3(1) -> X
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0049 3500 0000 0000 0000 8001 4042 0000 013b 0303 7878 0000 2222 0000 8080

0452
0453  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
0454  !
0455  !       QUADRATURE FILTER TIMING
0456  !
0457  !               Instruction WAIT
0458  !
0459  !       SWAITN;                                         !
0460  !       ;                                               !
0461  !       ;                                               !
0462  !
0463  !       Instruction -34
0464  !
0465  !       SCONT,GBYRTR1;                                  !
0466  !       ;                                               !
0467  !       A00000;                                         ! K3(1) -> X
0468  !
0469  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
0470
0471  !       Instruction -33
0472
0473  SCONT,GAYINC02,GBYINC05,XSA0B0S11;      !
0474  MB;                                     !
0475  A00036A,MBR,LM;                         ! R(1,n-1) -> A0, K4(1) -> X,
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
004a 0000 0000 0000 0000 0999 c184 0000 017b 0b0b b878 0000 2222 3300 0080

0476                                            ! K1R(1,n-1) -> ALU,
0477
0478  !       Instruction -32
0479
0480  SCONT,GAYINC00,GBYINC06,XSA1B0S01,UA0B0S13NOP;  ! Form K3R(1,n-1)
0481  A00000A,MAR,LM;                                 ! K5(1) -> A0,
0482                                                  ! R(1,n-2) -> X & ALU
0483  A60000A,MAR,LM;                                 ! K2(1) -> A6, K5(1) -> X
```

```
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
004b 0000 0000 0000 0000 0a98 c164 0003 1373 0303 b8b8 0000 2222 c000 0000

0484
0485  !       Instruction -31
0486
0487  SCONT,GBYINC02,XSA0B1S10,UA1B3S00SSUBA; ! Form K4R(1,n-2) & [R(1,n-2) -
0488                                          ! K1R(1,n-1)]
0489  A30010A,MBR,LM;                         ! G(1) -> A3, S(1,n-1) -> X,
0490  A60060A,MBR,LM;                         ! K6(1) -> A6, K2(1) -> X
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
004c 0000 0000 0000 0000 0210 4183 0738 857b 0b0b b8b8 0000 2223 d8a0 0000

0491
0492  !       Instruction -30
0493
0494  SCONT,GBYINC01,XSA1B0S01,02;             ! Form K5S(0,n-1)
0495                                          ! Light green LED
0496  A45000AB,XMSW,MAR,LM;                   ! S(0,n-2) -> A4, K3R(0,n-1) -> A5,
0497  A00064,TRC;                             ! K6(0) -> X, S(0,n-2) -> ALU,
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
004d 0000 0000 0000 0000 0011 4164 0000 017a 0303 78b8 0802 22c2 1940 8000

0498                                          ! U(0,n) -> TGCIL (TGBus C In Latch)
0499
0500  !       Instruction -29
0501
0502  SCONT,GAYINC06,XSA0B0S11,UA2B0S11NOP;   ! Form K2S(0,n-1)
0503  A04640ABC,XMSW,UMSW,LC;                 ! U(0,n) -> LB & A0, K4R(0,n-2) -> A4,
0504                                          ! R(0,n-2) - K1R(0,n-1) -> A6
0505                                          ! S(0,n-2) -> X
0506  A20035,LD;                              ! R(0,n-1) -> LB, G(0) -> X,
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
004e 0000 0000 0000 0000 0588 8184 0001 1b3b 0202 7839 0002 224c b508 4000

0507                                          ! K3R(0,n-1) -> ALU
0508
0509  !       Instruction -28
0510
0511  SCONT,GAYINC01,GBYINC04,XSA1B1S00,UA0B0S11NOP;  ! Form K6S(0,n-2)
0512  A04004B,XMSW,MAL;                               ! K5S(0,n-1) -> A4,
0513                                                  ! U(0,n) -> X, K4R(0,n-2) -> ALU
0514                                                  ! R(0,n-1) -> IML (Into
0515                                                  ! Memory Latch)
0516  MAW;                                            ! R(0,n-1) -> M
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
004f 0000 0000 0000 0000 089a c163 0003 137b 0303 7a7c 0002 2242 0001 8080

0517
0518  !       Instruction -27
0519
0520  SCONT,XSS11,UA1B1S00SSUBB;               ! Form GU(0,n) & K3R(0,n-1) -
0521                                          ! K4R(0,n-2)
0522  A57000AB,XMSW,MBR,LM;                   ! K1(1) -> A5, K2S(0,n-1) -> A7
0523  A20050A,MAR,LM;                         ! R(1,n-1) -> A2, K1(1) -> X
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0050 0000 0000 0000 0000 0000 0100 053c 253b 030b b8b8 0003 22c3 a840 0000

0524
0525  !       Instruction -26
```

```
0526
0527  SCONT,GAYINC05,GBYINC00,XSA0B1S10,UA0B0S11NOP;  !
0528  A17027B,XMSW,LD;                                !  S(0,n-1) -> LB,
0529                                                  !  K6S(0,n-2) -> A7,
0530                                                  !  R(1,n-1) -> X,
0531                                                  !  K2S(0,n-1) -> ALU
0532  A00004,MAL;                                     !  K5S(0,n-1) -> ALU
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0051 0000 0000 0000 0000 049a c183 0003 137b 0302 7c78 0003 22c3 0117 8040

0533                                                  ! S(0,n-1) -> IML
0534
0535  !       Instruction -25
0536
0537  SCONT,GAYINC04,GBYINC03,XSS01,UA1B1S20SSUBB;    ! Form K1R(1,n-1) &
0538                                                  ! S(0,n-2) - K2S(0,n-1)
0539  A05707BC,XMSW,UMSW,MBW;                         ! GU(0,n) -> A5,
0540                                                  ! K3R(0,n-1) - K4R(0,n-2) -> A7,
0541                                                  ! K6S(0,n-2) -> ALU
0542                                                  ! S(0,n-1) -> M
0543  A10007A,MBR,LM;                                 ! S(1,n-1) -> A1,
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0052 0000 0000 0000 0000 0699 c140 053c 247b 0b0b b87b 0002 23dc 0707 0080

0544                                                  ! K3R(0,n-1) - K4R(0,n-2) -> ALU
0545
0546  !       Instruction -24
0547
0548  SCONT,GBYINC01,UA0B0S11SSUBB;           ! Form K5S(0,n-1) - K6S(0,n-2)
0549  A00005A,MAR,LM;                         ! K3(1) -> A0, GU(0,n) -> ALU
0550  A00000A,MAR,LM;                         ! K4(1) -> A0, K3(1) -> X,
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0053 0000 0000 0000 0000 0011 4042 053f 137b 0303 b8b8 0000 2222 0005 0000

0551
0552  !       Instruction -23
0553
0554  SCONT,GAYINC02,GBYINC05,XSA0B0S11,UA1B2S00SADD; ! Form R(0,n)
0555  A067068C,XMSW,UMSW,MB;                          ! K1R(1,n-1) -> A6,
0556                                                  ! S(0,n-2) - K2S(0,n-1) -> A7,
0557                                                  ! R(0,n-2) - K1R(0,n-1) -> ALU
0558  A00006A,MBR,LM;                                 ! R(1,n-1) -> A0, K4(1) -> X,
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0054 0000 0000 0000 0000 0999 c184 0538 457b 0b0b b879 0003 225c 0303 0080

0559                                                  ! K1R(1,n-1) -> ALU,
0560
0561  !       Instruction -22
0562
0563  SCONT,GAYINC00,GBYINC06,XSA1B0S01,UA0B0S13NOP;  ! Form K3R(1,n-1)
0564  A00600AC,UMSW,MAR,LM;                           ! K5(1) -> A0,
0565                                                  ! K5S(0,n-1) - K6S(0,n-2) -> A6,
0566                                                  ! R(1,n-2) -> X & ALU
0567  A60006A,MAR,LM;                                 ! K2(1) -> A6, K5(1) -> X
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0055 0000 0000 0000 0000 0a98 c164 0003 1373 0303 b8b9 0000 222c c300 0000

0568                                                  ! K5S(0,n-1) - K6S(0,n-2) -> ALU
0569
0570  !       Instruction -21
0571
```

```
0572  SCONT,GBYINC02,XSA0B1S10,UA1B3S00SSUBA;          ! Form K4R(1,n-2) & [R(1,n-2) -
0573                                                   ! K1R(1,n-1)]
0574  A30717AC,UMSW,MBR,LM;                           ! G(1) -> A3, R(0,n) -> A7,
0575                                                   ! S(1,n-1) -> X,
0576                                                   ! S(0,n-2) - K2S(0,n-1) -> ALU
0577  A60060A,MBR,LM;                                 ! K6(1) -> A6, K2(1) -> X
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0056 0000 0000 0000 0000 0210 4183 0738 857b 0b0b b8b9 0000 223d d8a7 0000

0578
0579  !       Instruction -20
0580
0581  SCONT,XSA1B0S01,US12SADD,O2;                     ! Form K5S(0,n-1) & [R(-1,n-2) -
0582                                                   ! K1R(-1,n-1) + K5S(-1,n-1) -
0583                                                   ! K6S(-1,n-2)]
0584                                                   ! Light green LED
0585  A45000AB,XMSW,MAR,LM;                           ! S(0,n-2) -> A4, K3R(0,n-1) -> A5,
0586  A00064,TRC;                                     ! K6(0) -> X, S(0,n-2) -> ALU,
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0057 0000 0000 0000 0000 0000 0164 0538 0332 0303 78b8 0802 22c2 1940 8000

0587                                                   ! U(0,n) -> TGCiL (TGBus C In Latch)
0588
0589  !       Instruction -19
0590
0591  SCONT,GAYINC06,XSA0B0S11,UA2B0S11NOP;            ! Form K2S(0,n-1)
0592  A04640ABC,XMSW,UMSW,LC;                         ! U(0,n) -> LB & A0, K4R(0,n-2) -> A4,
0593                                                   ! R(0,n-2) - K1R(0,n-1) -> A6
0594                                                   ! S(0,n-2) -> X
0595  A20035,LD;                                      ! R(0,n-1) -> LB, G(0) -> X,
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0058 0000 0000 0000 0000 0588 8184 0001 1b3b 0202 7839 0002 224c b508 4000

0596                                                   ! K3R(0,n-1) -> ALU
0597
0598  !       Instruction -18
0599
0600  SCONT,GAYINC01,GBYINC04,XSA1B1S00,UA0B0S11NOP;   ! Form K6S(0,n-2)
0601  A04504BC,XMSW,UMSW,MAL;                         ! K5S(0,n-1) -> A4, [R(-1,n-2) -
0602                                                   ! K1R(-1,n-1) + K5S(-1,n-1) -
0603                                                   ! K6S(-1,n-2)] -> A5,
0604                                                   ! U(0,n) -> X, K4R(0,n-2) -> ALU
0605                                                   ! R(0,n-1) -> IML (Into
0606                                                   ! Memory Latch)
0607  A70005,MAW,LD;                                  ! R(-1,n) -> LB, [R(-1,n-2) -
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0059 0000 0000 0000 0000 089a c163 0003 137b 0203 7a7d 0002 2354 c501 4080

0608                                                   ! K1R(-1,n-1) + K5S(-1,n-1) -
0609                                                   ! K6S(-1,n-2)] -> ALU,
0610                                                   ! R(0,n-1) -> M
0611
0612  !       Instruction -17
0613
0614  SCONT,GAYINC07,XSS11,UA1B1S00SSUBB;              ! Form GU(0,n) & K3R(0,n-1) -
0615                                                   ! K4R(0,n-2)
0616  A57007AB,XMSW,MBRL,LM;                          ! K1(1) -> A5, K2S(0,n-1) -> A7,
0617                                                   ! R(-1,n) -> ALU & IML
0618  A20050A,MAR,LM;                                 ! R(1,n-1) -> A2, K1(1) -> X
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
005a 0000 0000 0000 0000 058a 8100 053c 253b 030b b8bc 0003 22c3 a847 0000
```

```
0619
0620  !       Instruction -16
0621
0622  SCONT,GAYINC05,GBYINC00,XSA0B1S10,UA0B0S11SADD; ! Form S(-1,n)
0623  A17027B,XMSW,MAW,LD;                           ! S(0,n-1) -> LB,
0624                                                 ! K6S(0,n-2) -> A7,
0625                                                 ! R(1,n-1) -> X,
0626                                                 ! K2S(0,n-1) -> ALU
0627                                                 ! R(-1,n) -> M
0628  A00004,MAL;                                    ! K5S(0,n-1) -> ALU
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
005b 0000 0000 0000 0000 049a c183 053b 137b 0302 7c7a 0003 22c3 0117 8040

0629                                                 ! S(0,n-1) -> IML
0630
0631  !       Instruction -15
0632
0633  SCONT,GAYINC04,GBYINC03,XSS01,UA1B1S20SSUBB;   ! Form K1R(1,n-1) &
0634                                                 ! S(0,n-2) - K2S(0,n-1)
0635  A05707BC,XMSW,UMSW,MBW;                        ! GU(0,n) -> A5,
0636                                                 ! K3R(0,n-1) - K4R(0,n-2) -> A7,
0637                                                 ! K6S(0,n-2) -> ALU
0638                                                 ! S(0,n-1) -> M
0639  A10007A,MBR,LM;                                ! S(1,n-1) -> A1,
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
005c 0000 0000 0000 0000 0699 c140 053c 247b 0b0b b87b 0002 23dc 0707 0080

0640                                                 ! K3R(0,n-1) - K4R(0,n-2) -> ALU
0641
0642  !       Instruction -14
0643
0644  SCONT,GBYINC01,UA0B0S11SSUBB;                  ! Form K5S(0,n-1) - K6S(0,n-2)
0645  A00505AC,UMSW,MAR,LM;                          ! K3(1) -> A0, S(-1,n) -> A5,
0646                                                 ! GU(0,n) -> ALU
0647  A00005A,MAR,LM;                                ! K4(1) -> A0, K3(1) -> X,
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
005d 0000 0000 0000 0000 0011 4042 053f 137b 0303 b8b9 0000 2234 0505 0000

0648                                                 ! S(-1,n) -> ALU
0649
0650  !       Instruction -13
0651
0652  SCONT,GAYINC02,GBYINC05,XSA0B0S11,UA1B2S00SADD; ! Form R(0,n)
0653  A56706BC,XMSW,UMSW,MB,LD;                      ! S(-1,n) -> LB,
0654                                                 ! K1R(1,n-1) -> A6,
0655                                                 ! S(0,n-2) - K2S(0,n-1) -> A7,
0656                                                 ! R(0,n-2) - K1R(0,n-1) -> ALU
0657  A00006A,MBRL,LM;                               ! R(1,n-1) -> A0, K4(1) -> X,
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
005e 0000 0000 0000 0000 0999 c184 0538 457b 0b0a bc79 0003 225d 0343 0040

0658                                                 ! K1R(1,n-1) -> ALU,
0659                                                 ! S(-1,n) -> IML
0660
0661  !       Instruction -12
0662
0663  SCONT,GAYINC00,GBYINC06,XSA1B0S01,UA0B0S13SADD; ! Form K3R(1,n-1) & Y(-1,n)
0664  A00600AC,UMSW,MAR,LM;                          ! K5(1) -> A0,
0665                                                 ! K5S(0,n-1) - K6S(0,n-2) -> A6,
0666                                                 ! R(1,n-2) -> X & ALU
0667  A60006A,MAR,LM;                                ! K2(1) -> A6, K5(1) -> X
```

```
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
005f 0000 0000 0000 0000 0a98 c164 053b 1373 0303 b8b9 0000 222c c300 0000

0668                                                    ! K5S(0,n-1) - K6S(0,n-2) -> ALU
0669
0670  !       Instruction -11
0671
0672  SCONT,GAYINC03,GBYINC02,XSA0B1S10,UA1B3S00SSUBA;! Form K4R(1,n-2) & [R(1,n-2) -
0673                                                    ! K1R(1,n-1)]
0674  A30717AC,UMSW,MBR,LM;                            ! G(1) -> A3, R(0,n) -> A7,
0675                                                    ! S(1,n-1) -> X,
0676                                                    ! S(0,n-2) - K2S(0,n-1) -> ALU
0677  A60060A,MBR,LM;                                  ! K6(1) -> A6, K2(1) -> X
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0060 0000 0000 0000 0000 039a c183 0738 857b 0b0b b8b9 0000 223d d8a7 0000

0678
0679
0680  !       Instruction -10
0681
0682  SCONT,XSA1B0S01,US12SADD,02;                      ! Form K5S(0,n-1) & [R(-1,n-2) -
0683                                                    ! K1R(-1,n-1) + K5S(-1,n-1) -
0684                                                    ! K6S(-1,n-2)]
0685                                                    ! Light green LED
0686  A45000ABC,XMSW,UMSW,MAR,LM;                      ! S(0,n-2) -> A4, K3R(0,n-1) -> A5,
0687                                                    ! Y(-2,n) -> A0
0688  A00064,MAW,LD,TRC;                               ! K6(0) -> X, S(0,n-2) -> ALU,
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0061 0000 0000 0000 0000 0000 0164 0538 0332 0203 7ab9 0802 22c0 1940 4000

0689                                                    ! Y(-2,n) -> LB (Local Bus),
0690                                                    ! S(-2,n) -> M (Memory)
0691                                                    ! U(0,n) -> TGCIL (TGBus C In Latch)
0692
0693  !       Instruction -9
0694
0695  SCONT,GAYINC06,XSA0B0S11,UA2B0S11NOP;             ! Form K2S(0,n-1)
0696  A04640ABC,XMSW,UMSW,LC,TWB;                      ! U(0,n) -> LB & A0, K4R(0,n-2) -> A4,
0697                                                    ! R(0,n-2) - K1R(0,n-1) -> A6
0698                                                    ! S(0,n-2) -> X, Y(-2,n) -> TGBOL
0699                                                    ! (TGBus Outgoing Latch)
0700  A20035,LD;                                        ! R(0,n-1) -> LB, G(0) -> X,
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0062 0000 0000 0000 0000 0588 8184 0001 1b3b 0202 7839 0012 224c b508 4000

0701                                                    ! K3R(0,n-1) -> ALU
0702
0703  !       Instruction -8
0704
0705  SCONT,GAYINC01,GBYINC04,XSA1B1S00,UA0B0S11NOP;    ! Form K6S(0,n-2)
0706  A04504BC,XMSW,UMSW,MAL;                          ! K5S(0,n-1) -> A4, [R(-1,n-2) -
0707                                                    ! K1R(-1,n-1) + K5S(-1,n-1) -
0708                                                    ! K6S(-1,n-2)] -> A5,
0709                                                    ! U(0,n) -> X, K4R(0,n-2) -> ALU
0710                                                    ! R(0,n-1) -> IML (Into
0711                                                    ! Memory Latch)
0712  A70005,MAW,LD;                                   ! R(-1,n) -> LB, [R(-1,n-2) -
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0063 0000 0000 0000 0000 089a c163 0003 137b 0203 7a7d 0002 2354 c501 4080

0713                                                    ! K1R(-1,n-1) + K5S(-1,n-1) -
```

```
0714                                                    ! K6S(-1,n-2)] -> ALU,
0715                                                    ! R(0,n-1) -> M
0716
0717  !       Instruction -7
0718
0719  SCONT,GAYINC07,XSS11,UA1B1S00SSUBB;       ! Form GU(0,n) & K3R(0,n-1) -
0720                                            ! K4R(0,n-2)
0721  A57007AB,XMSW,MBRL,LM;                    ! K1(1) -> A5, K2S(0,n-1) -> A7,
0722                                            ! R(-1,n) -> ALU & IML
0723  A20050A,MAR,LM;                           ! R(1,n-1) -> A2, K1(1) -> X
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0064 0000 0000 0000 0000 058a 8100 053c 253b 030b b8bc 0003 22c3 a847 0000

0724
0725  !       Instruction -6
0726
0727  SCONT,GAYINC05,GBYINC00,XSA0B1S10,UA0B0S11SADD; ! Form S(-1,n)
0728  A17027B,XMSW,MAW,LD;                      ! S(0,n-1) -> LB,
0729                                            ! K6S(0,n-2) -> A7,
0730                                            ! R(1,n-1) -> X,
0731                                            ! K2S(0,n-1) -> ALU
0732                                            ! R(-1,n) -> M
0733  A00004,MAL;                               ! K5S(0,n-1) -> ALU
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0065 0000 0000 0000 0000 049a c183 053b 137b 0302 7c7a 0003 22c3 0117 8040

0734                                                    ! S(0,n-1) -> IML
0735
0736  !       Instruction -5
0737
0738  SCONT,GAYINC04,GBYINC03,XSS01,UA1B1S20SSUBB;    ! Form K1R(1,n-1) &
0739                                                   ! S(0,n-2) - K2S(0,n-1)
0740  A05707BC,XMSW,UMSW,MBW;                          ! GU(0,n) -> A5,
0741                                                   ! K3R(0,n-1) - K4R(0,n-2) -> A7,
0742                                                   ! K6S(0,n-2) -> ALU
0743                                                   ! S(0,n-1) -> M
0744  A10007A,MBR,LM;                                  ! S(1,n-1) -> A1,
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0066 0000 0000 0000 0000 0699 c140 053c 247b 0b0b b87b 0002 23dc 0707 0080

0745                                                    ! K3R(0,n-1) - K4R(0,n-2) -> ALU
0746
0747  !       Instruction -4
0748
0749  SCONT,GBYINC01,UA0B0S11SSUBB;             ! Form K5S(0,n-1) - K6S(0,n-2)
0750  A00505AC,UMSW,MAR,LM;                     ! K3(1) -> A0, S(-1,n) -> A5,
0751                                            ! GU(0,n) -> ALU
0752  A00005A,MAR,LM;                           ! K4(1) -> A0, K3(1) -> X,
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0067 0000 0000 0000 0000 0011 4042 053f 137b 0303 b8b9 0000 2234 0505 0000

0753                                            ! S(-1,n) -> ALU
0754
0755  !       Instruction -3
0756
0757  SCONT,GAYINC02,GBYINC05,XSA0B0S11,UA1B2S00SADD; ! Form R(0,n)
0758  A56706BC,XMSW,UMSW,MB,LD;                 ! S(-1,n) -> LB,
0759                                            ! K1R(1,n-1) -> A6,
0760                                            ! S(0,n-2) - K2S(0,n-1) -> A7,
0761                                            ! R(0,n-2) - K1R(0,n-1) -> ALU
0762  A00006A,MBRL,LM;                          ! R(1,n-1) -> A0, K4(1) -> X,
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
```

```
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0068 0000 0000 0000 0000 0999 c184 0538 457b 0b0a bc79 0003 225d 0343 0040

0763                                                    ! K1R(1,n-1) -> ALU,
0764                                                    ! S(-1,n) -> IML
0765
0766  !       Instruction -2
0767
0768  SCONT,GAYINC00,GBYINC06,XSA1B0S01,UA0B0S13SADD; ! Form K3R(1,n-1) & Y(-1,n)
0769  A00600AC,UMSW,MAR,LM;                           ! K5(1) -> A0,
0770                                                  ! K5S(0,n-1) - K6S(0,n-2) -> A6,
0771                                                  ! R(1,n-2) -> X & ALU
0772  A60006A,MAR,LM;                                ! K2(1) -> A6, K5(1) -> X
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0069 0000 0000 0000 0000 0a98 c164 053b 1373 0303 b8b9 0000 222c c300 0000

0773                                                    ! K5S(0,n-1) - K6S(0,n-2) -> ALU
0774
0775  !       Instruction -1
0776
0777  SCONT,GAYINC03,GBYINC02,XSA0B1S10,UA1B3S00SSUBA;! Form K4R(1,n-2) & [R(1,n-2) -
0778                                                  ! K1R(1,n-1)]
0779  A30717AC,UMSW,MBR,LM;                           ! G(1) -> A3, R(0,n) -> A7,
0780                                                  ! S(1,n-1) -> X,
0781                                                  ! S(0,n-2) - K2S(0,n-1) -> ALU
0782  A60060A,MBR,LM;                                ! K6(1) -> A6, K2(1) -> X
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
006a 0000 0000 0000 0000 039a c183 0738 857b 0b0b b8b9 0000 223d d8a7 0000

0783
0784  :MAIN_LOOP_1:
0785
0786  !       Instruction 0
0787
0788  SJDC0<FINISH_UP_1>,XSA1B0S01,US12SADD,02;       ! Form K5S(0,n-1) & [R(-1,n-2)-
0789                                                  ! K1R(-1,n-1) + K5S(-1,n-1) -
0790                                                  ! K6S(-1,n-2)], Light green LED
0791  A45000ABC,XMSW,UMSW,MAR,LM,TTB;                 ! S(0,n-2) -> A4,
0792                                                  ! K3R(0,n-1) -> A5,
0793                                                  ! Y(-3,n) -> TGBus B
0794                                                  ! Y(-2,n) -> A0
0795  A00064,MAW,LD,TRC;                              ! K6(0) -> X, S(0,n-2) -> ALU,
          SDATA=0075  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
006b 4c00 0075 0000 0000 0000 0164 0538 0332 0203 7aa9 0802 22c0 1940 4000

0796                                                    ! Y(-2,n) -> LB (Local Bus),
0797                                                    ! S(-2,n) -> M (Memory)
0798                                                    ! U(0,n) -> TGCIL
0799                                                    !  (TGBus C In Latch)
0800
0801  !       Instruction 1
0802
0803  SCONT,GAYINC06,XSA0B0S11,UA2B0S11NOP;    ! Form K2S(0,n-1)
0804  A04640ABC,XMSW,UMSW,LC,TWB;              ! U(0,n) -> LB & A0, K4R(0,n-2) -> A4,
0805                                           ! R(0,n-2) - K1R(0,n-1) -> A6
0806                                           ! S(0,n-2) -> X, Y(-2,n) -> TGBOL
0807                                           ! (TGBus Outgoing Latch)
0808  A20035,LD;                               ! R(0,n-1) -> LB, G(0) -> X,
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
006c 0000 0000 0000 0000 0588 8184 0001 1b3b 0202 7839 0012 224c b508 4000

0809                                           ! K3R(0,n-1) -> ALU
```

```
0810
0811  !       Instruction 2
0812
0813  SCONT,GAYINC01,GBYINC04,XSA1B1S00,UA0B0S11NOP;  ! Form K6S(0,n-2)
0814  A04504BC,XMSW,UMSW,MAL;                         ! K5S(0,n-1) -> A4, [R(-1,n-2) -
0815                                                  ! K1R(-1,n-1) + K5S(-1,n-1) -
0816                                                  ! K6S(-1,n-2)] -> A5,
0817                                                  ! U(0,n) -> X, K4R(0,n-2) -> ALU
0818                                                  ! R(0,n-1) -> IML (Into
0819                                                  ! Memory Latch)
0820  A70005,MAW,LD;                                  ! R(-1,n) -> LB, (R(-1,n-2) -
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
006d 0000 0000 0000 0000 089a c163 0003 137b 0203 7a7d 0002 2354 c501 4080

0821                                                  ! K1R(-1,n-1) + K5S(-1,n-1) -
0822                                                  ! K6S(-1,n-2)] -> ALU,
0823                                                  ! R(0,n-1) -> M
0824
0825  !       Instruction 3
0826
0827  SCONT,GAYINC07,XSS11,UA1B1S00SSUBB;      ! Form GU(0,n) & K3R(0,n-1) -
0828                                           ! K4R(0,n-2)
0829  A57007AB,XMSW,MBRL,LM;                   ! K1(1) -> A5, K2S(0,n-1) -> A7,
0830                                           ! R(-1,n) -> ALU & IML
0831  A20050A,MAR,LM;                          ! R(1,n-1) -> A2, K1(1) -> X
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
006e 0000 0000 0000 0000 058a 8100 053c 253b 030b b8bc 0003 22c3 a847 0000

0832
0833  !       Instruction 4
0834
0835  SCONT,GAYINC05,GBYINC00,XSA0B1S10,UA0B0S11SADD; ! Form S(-1,n)
0836  A17027B,XMSW,MAW,LD;                            ! S(0,n-1) -> LB,
0837                                                  ! K6S(0,n-2) -> A7,
0838                                                  ! R(1,n-1) -> X,
0839                                                  ! K2S(0,n-1) -> ALU
0840                                                  ! R(-1,n) -> M
0841  A00004,MAL;                                     ! K5S(0,n-1) -> ALU
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
006f 0000 0000 0000 0000 049a c183 053b 137b 0302 7c7a 0003 22c3 0117 8040

0842                                                  ! S(0,n-1) -> IML
0843
0844  !       Instruction 5
0845
0846  SCONT,GAYINC04,GBYINC03,XSS01,UA1B1S20SSUBB;    ! Form K1R(1,n-1) &
0847                                                  ! S(0,n-2) - K2S(0,n-1)
0848  A05707BC,XMSW,UMSW,MBW;                         ! GU(0,n) -> A5,
0849                                                  ! K3R(0,n-1) - K4R(0,n-2) -> A7,
0850                                                  ! K6S(0,n-2) -> ALU
0851                                                  ! S(0,n-1) -> M
0852  A10007A,MBR,LM;                                 ! S(1,n-1) -> A1,
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0070 0000 0000 0000 0000 0699 c140 053c 247b 0b0b b87b 0002 23dc 0707 0080

0653                                                  ! K3R(0,n-1) - K4R(0,n-2) -> ALU
0854
0855  !       Instruction 6
0856
0857  SCONT,GBYINC01,UA0B0S11SSUBB;            ! Form K5S(0,n-1) - K6S(0,n-2)
0858  A00505AC,UMSW,MAR,LM;                    ! K3(1) -> A0, S(-1,n) -> A5,
0859                                           ! GU(0,n) -> ALU
```

```
0860  A00005A,MAR,LM;                              ! K4(1) -> A0, K3(1) -> X,
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9    10   11   12   13   14
0071 0000 0000 0000 0000 0011 4042 053f 137b 0303 b8b9 0000 2234 0505 0000

0861                                             ! S(-1,n) -> ALU
0862
0863  !       Instruction 7
0864
0865  SCONT,GAYINC02,GBYINC05,XSA0B0S11,UA1B2S00SADD; ! Form R(0,n)
0866  A56706BC,XMSW,UMSW,MB,LD;                    ! S(-1,n) -> LB,
0867                                               ! K1R(1,n-1) -> A6,
0868                                               ! S(0,n-2) - K2S(0,n-1) -> A7,
0869                                               ! R(0,n-2) - K1R(0,n-1) -> ALU
0870  A00006A,MBRL,LM;                             ! R(1,n-1) -> A0, K4(1) -> X,
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9    10   11   12   13   14
0072 0000 0000 0000 0000 0999 c184 0538 457b 0b0a bc79 0003 225d 0343 0040

0871                                             ! K1R(1,n-1) -> ALU,
0872                                             ! S(-1,n) -> IML
0873
0874  !       Instruction 8
0875
0876  SCONT,GAYINC00,GBYINC06,XSA1B0S01,UA0B0S13SADD; ! Form K3R(1,n-1) & Y(-1,n)
0877  A00600AC,UMSW,MAR,LM;                        ! K5(1) -> A0,
0878                                               ! K5S(0,n-1) - K6S(0,n-2) -> A6,
0879                                               ! R(1,n-2) -> X & ALU
0880  A60006A,MAR,LM;                              ! K2(1) -> A6, K5(1) -> X
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9    10   11   12   13   14
0073 0000 0000 0000 0000 0a98 c164 053b 1373 0303 b8b9 0000 222c c300 0000

0881                                             ! K5S(0,n-1) - K6S(0,n-2) -> ALU
0882
0883  !       Instruction 9
0884
0885  SJDAU<MAIN_LOOP_1>,GAYINC03,GBYINC02,XSA0B1S10,UA1B3S00SSUBA;
0886                                             ! Form K4R(1,n-2) & [R(1,n-2) -
0887                                             ! K1R(1,n-1)]
0888  A30717AC,UMSW,MBR,LM;                        ! G(1) -> A3, R(0,n) -> A7,
0889                                               ! S(1,n-1) -> X,
0890                                               ! S(0,n-2) - K2S(0,n-1) -> ALU
0891  A60060A,MBR,LM;                              ! K6(1) -> A6, K2(1) -> X
            SDATA=006B  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9    10   11   12   13   14
0074 7300 006b 0000 0000 039a c183 0738 857b 0b0b b8b9 0000 223d d8a7 0000

0892
0893  !       Instruction 10 (Really instruction 0, but for orders sake)
0894
0895  :FINISH_UP_1:
0896
0897  !       Instruction 11
0898
0899  SCONT,GAYINC06,XSA0B0S11,UA2B0S11NOP;        ! Form K2S(0,n-1)
0900  A04640ABC,XMSW,UMSW,LC,TWB;                  ! U(0,n) -> LB & A0, K4R(0,n-2) -> A4,
0901                                               ! R(0,n-2) - K1R(0,n-1) -> A6
0902                                               ! S(0,n-2) -> X, Y(-2,n) -> TGBOL
0903                                               ! (TGBus Outgoing Latch)
0904  A20035,LD;                                   ! R(0,n-1) -> LB, G(0) -> X,
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9    10   11   12   13   14
0075 0000 0000 0000 0000 0588 8184 0001 1b3b 0202 7839 0012 224c b508 4000

0905                                             ! K3R(0,n-1) -> ALU
```

```
0906
0907  !       Instruction 12
0908
0909  SCONT,XSA1B1S00,UA0B0S11NOP;                ! Form K6S(0,n-2)
0910  A04504BC,XMSW,UMSW,MAL;                     ! K5S(0,n-1) -> A4, [R(-1,n-2) -
0911                                              ! K1R(-1,n-1) + K5S(-1,n-1) -
0912                                              ! K6S(-1,n-2)] -> A5,
0913                                              ! U(0,n) -> X, K4R(0,n-2) -> ALU
0914                                              ! R(0,n-1) -> IML (into
0915                                              ! Memory Latch)
0916  A70005,MAW,LD;                              ! R(-1,n) -> LB, [R(-1,n-2) -
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9    10   11   12   13   14
0076 0000 0000 0000 0000 0000 0163 0003 133b 0203 7a7d 0002 2354 c501 4080

0917                                              ! K1R(-1,n-1) + K5S(-1,n-1) -
0918                                              ! K6S(-1,n-2)] -> ALU,
0919                                              ! R(0,n-1) -> M
0920
0921  !       Instruction 13
0922
0923  SCONT,GAYINC07,XSS11,UA1B1S00SSUB8;         ! Form GU(0,n) & K3R(0,n-1) -
0924                                              ! K4R(0,n-2)
0925  A07007B,XMSW,MBL;                           ! K2S(0,n-1) -> A7,
0926                                              ! R(-1,n) -> ALU & IML
0927  MA;                                         !
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9    10   11   12   13   14
0077 0000 0000 0000 0000 058a 8100 053c 253b 030b 787c 0003 22c2 0007 8080

0928
0929  !       Instruction 14
0930
0931  SCONT,GAYINC05,UA0B0S11SADD;                 ! Form S(-1,n)
0932  A17007B,XMSW,MAW,LD;                        ! S(0,n-1) -> LB,
0933                                              ! K6S(0,n-2) -> A7,
0934                                              ! K2S(0,n-1) -> ALU
0935                                              ! R(-1,n) -> M
0936  A00004,MAL;                                 ! K5S(0,n-1) -> ALU
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9    10   11   12   13   14
0078 0000 0000 0000 0000 048a 8042 053b 133b 0302 7c7a 0003 22c3 0107 8040

0937                                              ! S(0,n-1) -> IML
0938
0939  !       Instruction 15
0940
0941  SCONT,UA1B1S20SSUB8;                         ! Form S(0,n-2) - K2S(0,n-1)
0942  A05707BC,XMSW,UMSW,MBW;                     ! GU(0,n) -> A5,
0943                                              ! K3R(0,n-1) - K4R(0,n-2) -> A7,
0944                                              ! K6S(0,n-2) -> ALU
0945                                              ! S(0,n-1) -> M
0946  A00007;                                     ! K3R(0,n-1) - K4R(0,n-2) -> ALU
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9    10   11   12   13   14
0079 0000 0000 0000 0000 0000 0042 053c 243b 030b 787b 0002 22dc 0707 8080

0947
0948  !       Instruction 16
0949
0950  SCONT,UA0B0S11SSUB8;                         ! Form K5S(0,n-1) - K6S(0,n-2)
0951  A00505C,UMSW;                               ! S(-1,n) -> A5, GU(0,n) -> ALU
0952  A00005;                                     ! S(-1,n) -> ALU
          SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9    10   11   12   13   14
007a 0000 0000 0000 0000 0000 0042 053f 133b 0303 7879 0000 2234 0505 8080
```

```
0953
0954  !       Instruction 17
0955
0956  SCONT,UA1B2S00SADD;                          ! Form R(0,n)
0957  A50706C,UMSW,LD;                            ! S(-1,n) -> LB,
0958                                              ! S(0,n-2) - K2S(0,n-1) -> A7,
0959                                              ! R(0,n-2) - K1R(0,n-1) -> ALU
0960  MBL;                                        ! S(-1,n) -> IML
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
007b 0000 0000 0000 0000 0000 0042 0538 453b 0b02 7c79 0000 223d 0043 8040

0961
0962  !       Instruction 18
0963
0964  SCONT,US13SADD;                             ! Form Y(-1,n)
0965  A00600C,UMSW;                               ! K5S(0,n-1) - K6S(0,n-2) -> A6,
0966  A00006;                                     ! K5S(0,n-1) - K6S(0,n-2) -> ALU
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
007c 0000 0000 0000 0000 0000 0042 0539 0333 0303 7879 0000 222c 0300 8080

0967
0968  !       Instruction 19
0969
0970  SCONT,GAY1NC03,UA1B3S00NOP;                 !
0971  A00707C,UMSW;                               ! R(0,n) -> A7,
0972                                              ! S(0,n-2) - K2S(0,n-1) -> ALU
0973  ;                                           !
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
007d 0000 0000 0000 0000 018a 8042 0000 853b 0303 7879 0000 223c 0007 8080

0974
0975  !       Instruction 20
0976
0977  SCONT,US12SADD;                             ! Form [R(-1,n-2) - K1R(-1,n-1) +
0978                                              ! K5S(-1,n-1) - K6S(-1,n-2)]
0979  A00000C,UMSW,MA,TTB;                        ! Y(-3,n) -> TGBus B
0980                                              ! Y(-2,n) -> A0
0981  A00064,MAW,LD;                              ! Y(-2,n) -> LB (Local Bus),
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
007e 0000 0000 0000 0000 0000 0042 0538 0333 0203 7a69 0000 2220 1900 4080

0982                                              ! S(-2,n) -> M (Memory)
0983
0984  !       Instruction 21
0985
0986  SCONT;                                      !
0987  TWB;                                        ! Y(-2,n) -> TGBOL
0988  ;                                           !
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
007f 0000 0000 0000 0000 0000 0042 0000 013b 0303 7878 0010 2222 0000 8080

0989
0990  !       Instruction 22
0991
0992  SCONT;                                      !
0993  A00500C,UMSW;                               ! [R(-1,n-2) - K1R(-1,n-1) +
0994                                              ! K5S(-1,n-1) - K6S(-1,n-2)] -> A5
0995  A70005,LD;                                  ! R(-1,n) -> LB, [R(-1,n-2) -
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
```

```
0080 0000 0000 0000 0000 0000 0042 0000 013b 0203 7879 0000 2334 c500 4080

0996                                                ! K1R(-1,n-1) + K5S(-1,n-1) -
0997                                                ! K6S(-1,n-2)] -> ALU
0998
0999  !       Instruction 23
1000
1001  SCONT,GAYINC07,UA1B1S00NOP;                   !
1002  A00007,MBL;                                   ! R(-1,n) -> ALU & IML
1003  MA;                                           !
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9    10   11   12   13   14
0081 0000 0000 0000 0000 058a 8042 0000 253b 030b 787c 0000 2222 0007 8080

1004
1005  !       Instruction 24
1006
1007  SCONT,US11SADD;                               ! Form S(-1,n)
1008  MAW;                                          ! R(-1,n) -> M
1009  ;                                             !
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9    10   11   12   13   14
0082 0000 0000 0000 0000 0000 0042 0539 033b 0303 787a 0000 2222 0000 8080

1010
1011  !       Instruction 25
1012
1013  SCONT;                                        !
1014  ;                                             !
1015  ;                                             !
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9    10   11   12   13   14
0083 0000 0000 0000 0000 0000 0042 0000 013b 0303 7878 0000 2222 0000 8080

1016
1017  !       Instruction 26
1018
1019  SCONT;                                        !
1020  A00500C,UMSW;                                 ! S(-1,n) -> A5
1021  A00005;                                       ! S(-1,n) -> ALU
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9    10   11   12   13   14
0084 0000 0000 0000 0000 0000 0042 0000 013b 0303 7879 0000 2234 0500 8080

1022
1023  !       Instruction 27
1024
1025  SCONT,UA1B2S00NOP;                            !
1026  A50000,LD;                                    ! S(-1,n) -> LB
1027  MBL;                                          ! S(-1,n) -> IML
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9    10   11   12   13   14
0085 0000 0000 0000 0000 0000 0042 0000 453b 0b02 7c78 0000 2223 0040 8040

1028
1029  !       Instruction 28
1030
1031  SCONT,US13SADD;                               ! Form Y(-1,n)
1032  ;                                             !
1033  ;                                             !
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9    10   11   12   13   14
0086 0000 0000 0000 0000 0000 0042 0539 0333 0303 7878 0000 2222 0000 8080

1034
1035  !       Instruction 29
```

```
1036
1037  SCONT,GAYINC03;                                  !
1038  ;                                                !
1039  ;                                                !
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0087 0000 0000 0000 0000 018a 8042 0000 013b 0303 7878 0000 2222 0000 8080

1040
1041  !       Instruction 30
1042
1043  SCONT;
1044  A00000C,UMSW,MA,TTB;                            ! Y(-3,n) -> TGBus B
1045                                                   ! Y(-2,n) -> A0
1046  A00064,MAW,LD;                                   ! Y(-2,n) -> LB (Local Bus),
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0088 0000 0000 0000 0000 0000 0042 0000 013b 0203 7a69 0000 2220 1900 4080

1047                                                 ! S(-2,n) -> M (Memory)
1048
1049  !       Instruction 31
1050
1051  SCONT;                                           !
1052  TWB;                                             ! Y(-2,n) -> TGBOL
1053  ;                                                !
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0089 0000 0000 0000 0000 0000 0042 0000 013b 0303 7878 0010 2222 0000 8080

1054
1055  !       Instruction 32
1056
1057  SCONT;                                           !
1058  ;                                                !
1059  ;                                                !
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
008a 0000 0000 0000 0000 0000 0042 0000 013b 0303 7878 0000 2222 0000 8080

1060
1061  !       Instruction 33
1062
1063  SCONT;                                           !
1064  ;                                                !
1065  ;                                                !
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
008b 0000 0000 0000 0000 0000 0042 0000 013b 0303 7878 0000 2222 0000 8080

1066
1067  !       Instruction 34
1068
1069  SCONT;                                           !
1070  ;                                                !
1071  ;                                                !
            SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
008c 0000 0000 0000 0000 0000 0042 0000 013b 0303 7878 0000 2222 0000 8080

1072
1073  !       Instruction 35
1074
1075  SCONT;                                           !
1076  ;                                                !
1077  ;                                                !
```

```
                SDATA=0000   GADATA=0000   GBDATA=0000   LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
008d 0000 0000 0000 0000 0000 0042 0000 013b 0303 7878 0000 2222 0000 8080


 !        Instruction 36

 SCONT;                                      !
 ;                                           !
 ;                                           !
                SDATA=0000   GADATA=0000   GBDATA=0000   LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
008e 0000 0000 0000 0000 0000 0042 0000 013b 0303 7878 0000 2222 0000 8080


 !        Instruction 37

 SCONT;                                      !
 ;                                           !
 ;                                           !
                SDATA=0000   GADATA=0000   GBDATA=0000   LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
008f 0000 0000 0000 0000 0000 0042 0000 013b 0303 7878 0000 2222 0000 8080


 !        Instruction 38

 SCONT;                                      !
 ;                                           !
 ;                                           !
                SDATA=0000   GADATA=0000   GBDATA=0000   LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0090 0000 0000 0000 0000 0000 0042 0000 013b 0303 7878 0000 2222 0000 8080


 !        Instruction 39

 SCONT;                                      !
 ;                                           !
 ;                                           !
                SDATA=0000   GADATA=0000   GBDATA=0000   LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0091 0000 0000 0000 0000 0000 0042 0000 013b 0303 7878 0000 2222 0000 8080


 !        Instruction 40

 SCONT;                                      !
 TTB;                                        ! Y(-3,n) -> TGBus B
 ;                                           !
                SDATA=0000   GADATA=0000   GBDATA=0000   LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0092 0000 0000 0000 0000 0000 0042 0000 013b 0303 7868 0000 2222 0000 8080


 !        Instruction 41

 SJDAU<NEW_RAY_I>;                           !
 ;                                           !
 ;                                           !
                SDATA=003B   GADATA=0000   GBDATA=0000   LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0093 7300 003b 0000 0000 0000 0042 0000 013b 0303 7878 0000 2222 0000 8080


 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

```
1116  !                                                                      !
1117  !       Subroutine to fill memory with filter coefficients.            !
1118  !         The first filter coefficients start at BANK and are          !
1119  !         sequentially listed as G,K1,K2,K3,K4,K5,K6. The next         !
1120  !         filter coefficients start at BANK+7 and are sequentially     !
1121  !         listed as G,K1,K2,K3,K4,K5,K6 etc.                           !
1122  !                                                                      !
1123  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
1124
1125  :FILTER_GENESIS_1:
1126
1127  SCONT,GAFYRTRO<BANK>;
1128  ;
1129  ;
              SDATA=0000  GADATA=C000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0094 0000 0000 0000 0006 1000 8842 0000 013b 0303 7878 0000 2222 0000 8080

1130
1131  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
1132  !                                                                      !
1133  !       Insert below the code generated by FILTBANK4.FOR that          !
1134  !         stores filter coefficients into memory.The code for the      !
1135  !         first filter is listed here as an example.                   !
1136  !                                                                      !
1137  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
1138
1139  SCONT,GAYINC00;
1140  LF+0.8927224E+00;
1141  MAL;
              SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=3F648975
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0095 0000 0000 263f ae23 0088 8042 0000 013b 0301 7c78 0000 2222 0000 8080

1142  SCONT,GAYINC00;
1143  LF+0.1999922E+01,MAW;
1144  MAL;
              SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=3FFFFD72
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0096 0000 0000 ff3f 4e7f 0088 8042 0000 013b 0301 7c7a 0000 2222 0000 8080

1145  SCONT,GAYINC00;
1146  LF+0.1999585E+01,MAW;
1147  MAL;
              SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=3FFFF267
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0097 0000 0000 ff3f e69e 0088 8042 0000 013b 0301 7c7a 0000 2222 0000 8080

1148  SCONT,GAYINC00;
1149  LF+0.1901770E+01,MAW;
1150  MAL;
              SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=3FF36D33
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0098 0000 0000 cf3f cc6d 0088 8042 0000 013b 0301 7c7a 0000 2222 0000 8080

1151  SCONT,GAYINC00;
1152  LF+0.9065588E+00,MAW;
1153  MAL;
              SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=3F68143D
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
0099 0000 0000 163f bc50 0088 8042 0000 013b 0301 7c7a 0000 2222 0000 8080

1154  SCONT,GAYINC00;
1155  LF+0.1985222E+01,MAW;
1156  MAL;
              SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=3FFE1BC1
```

```
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
009a 0000 0000 7f3f 83b1 0088 8042 0000 013b 0301 7c7a 0000 2222 0000 8080

1157  SCONT,GAY!NC00;
1158  LF+0.9871305E+00,MAW;
1159  MAL;
              SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=3F7CB496
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
009b 0000 0000 3e3f 695a 0088 8042 0000 013b 0301 7c7a 0000 2222 0000 8080

1160  SCONT;
1161  MAW;
1162  ;
              SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
009c 0000 0000 0000 0000 0000 0042 0000 013b 0303 787a 0000 2222 0000 8080

1163
1164  SRTNU;
1165  ;
1166  ;
              SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000
ADDR    1    2    3    4    5    6    7    8    9   10   11   12   13   14
009d 5300 0000 0000 0000 0000 0042 0000 013b 0303 7878 0000 2222 0000 8080

1167
1168  $
              SDATA=0000  GADATA=0000  GBDATA=0000  LDATA=00000000

24 Symbols Defined, 299 bytes used, space allows (512,5120)
7 Modify-able expressions tagged,  maximum permitted is 511

---------- Summary of References saved in MOD file ------------------

References to symbol GATES:
REF #1: Type=T Addr=D Expression= GATES/8, Next=3
REF #3: Type=S Addr=22 Expression=GATES-1, Next=5
REF #5: Type=S Addr=29 Expression=GATES-FILTER-1, Next=7
REF #7: Type=S Addr=3B Expression=GATES-5, Next=0

References to symbol FILTER:
REF #4: Type=S Addr=28 Expression=FILTER-1, Next=6
REF #6: Type=S Addr=29 Expression=GATES-FILTER-1, Next=0

References to symbol WIDTH:
REF #2: Type=A Addr=12 Expression=BANK+((WIDTH-1)*7), Next=0


24 SYM structures written
8 REF structures written
8 saved microcode words written
299 bytes of names and expressions written
```

APPENDIX IV

FILTER COEFFICIENT GENERATION
PROGRAM LISTING

```
0001          PROGRAM FILTBANK4
0002
0003    ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
0004    c                                                                       c
0005    c       Authored by : James T. Peltier                                  c
0006    c       Date        : January 12, 1989                                  c
0007    c                                                                       c
0008    c       This program generates fourpole filter coefficients             c
0009    c               and the SP20 code to load them into memory.             c
0010    c               The SP20 code is written to a file. This file           c
0011    c               can then be inserted in the appropriate point           c
0012    c               at the end of the SP20 program. An index                c
0013    c               pointer can be used to access the appropriate           c
0014    c               filter if it increments in multiples of seven.          c
0015    c                                                                       c
0016    ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
0017
0018
0019            IMPLICIT NONE
0020
0021            REAL*4        k1,k2,k3,k4,k5,k6,gain
0022            REAL*4        a1(5),a2(5),b1(5),b2(5)
0023            REAL*4        b3(5),b4(5),ag(5),wr(5)
0024            REAL*4        wu,t1,t2
0025            REAL*8        pi
0026            INTEGER*2     i,j
0027
0028
0029    ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
0030    c                                                                       c
0031    c         Coefficients for Analog Prototypes                            c
0032    c                                                                       c
0033    c       (Passband Ripple, Stopband Attenuation) - dBs                   c
0034    c                                                                       c
0035    c           (1,30) (1,40) (1,50) (1,60) (1,70)                          c
0036    c  Index -    1      2      3      4      5                             c
0037    c                                                                       c
0038    ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
0039
0040
0041            data a1 /5.72672,8.20047,11.40195,15.57508,21.05659/
0042            data a2 /1.37628,1.70946,2.18639,2.84628,3.74387/
0043            data b1 /0.682880,0.591849,0.510902,0.441175,0.381317/
0044            data b2 /0.348111,0.238719,0.169815,0.123525,0.091036/
0045            data b3 /0.148347,0.171046,0.173001,0.163210,0.148233/
0046            data b4 /0.803477,0.658897,0.520870,0.402671,0.307249/
0047            data ag /0.0316286,0.0100002,0.00316227,0.000999992,0.000316225/
0048            data wr /1.25040,1.51549,1.90819,2.46079,3.21902/
0049
0050            pi=4.0*atan(1.0)
0051
0052            OPEN(UNIT=16,FILE='COEF.DMP',STATUS='NEW')
0053
0054            do i = 1 , 5
0055              do j = 1 , 25
0056                wu   = (j/100.0)*pi          ! filter passband
0057                wu= tan(wu/2.0)/sqrt(wr(i))
0058                t1= wu
0059                t2= t1**2
0060                k1   = 2*(a1(i)-t2)/(a1(i)+t2)
0061                k2   = 2*(a2(i)-t2)/(a2(i)+t2)
0062                k3   = 2*(b2(i)-t2)/(b2(i)+b1(i)*t1+t2)
0063                k4   = (b2(i)-b1(i)*t1+t2)/(b2(i)+b1(i)*t1+t2)
0064                k5   = 2*(b4(i)-t2)/(b4(i)+b3(i)*t1+t2)
```

```
0065                  k6   = (b4(i)-b3(i)*t1+t2)/(b4(i)+b3(i)*t1+t2)
0066                  gain = ag(i)*(a1(i)+t2)*(a2(i)+t2)/
0067         1                ((b2(i)+b1(i)*t1+t2)*(b4(i)+b3(i)*t1+t2))
0068                  write(16,100) gain,k1,k2,k3,k4,k5,k6
0069               end do
0070            end do
0071
0072     100    format('SCONT,GAYINCOO;'/'LF',SPE14.7,';'/'MAL;'/
0073          1         'SCONT,GAYINCOO;'/'LF',SPE14.7,',MAW;'/'MAL;'/
0074          2         'SCONT,GAYINCOO;'/'LF',SPE14.7,',MAW;'/'MAL;'/
0075          3         'SCONT,GAYINCOO;'/'LF',SPE14.7,',MAW;'/'MAL;'/
0076          4         'SCONT,GAYINCOO;'/'LF',SPE14.7,',MAW;'/'MAL;'/
0077          5         'SCONT,GAYINCOO;'/'LF',SPE14.7,',MAW;'/'MAL;'/
0078          6         'SCONT,GAYINCOO;'/'LF',SPE14.7,',MAW;'/'MAL;'/
0079          7         'SCONT;'/'MAW;'/';')
0080
0081            close(unit=16,status='keep')
0082            stop
0083            end
```

PROGRAM SECTIONS

	Name	Bytes	Attributes
0	$CODE	350	PIC CON REL LCL SHR EXE RD NOWRT LONG
1	$PDATA	312	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2	$LOCAL	224	PIC CON REL LCL NOSHR NOEXE RD WRT QUAD
	Total Space Allocated	886	

ENTRY POINTS

Address	Type	Name
0-00000000		FILTBANK4

VARIABLES

Address	Type	Name	Address	Type	Name
**	R*4	GAIN	2-000000A0	I*2	I
2-000000A2	I*2	J	**	R*4	K1
**	R*4	K2	**	R*4	K3
**	R*4	K4	**	R*4	K5
2-000000B0	R*4	K6	2-000000A8	R*8	PI
**	R*4	T1	**	R*4	T2
**	R*4	WU			

ARRAYS

Address	Type	Name	Bytes	Dimensions
2-00000000	R*4	A1	20	(5)
2-00000014	R*4	A2	20	(5)
2-00000078	R*4	AG	20	(5)
2-00000028	R*4	B1	20	(5)
2-0000003C	R*4	B2	20	(5)
2-00000050	R*4	B3	20	(5)
2-00000064	R*4	B4	20	(5)
2-0000008C	R*4	WR	20	(5)

```
LABELS

    Address   Label

  1-00000009  100'

FUNCTIONS AND SUBROUTINES REFERENCED

  Type  Name

        FOR$CLOSE
        FOR$OPEN
  R*4   MTH$ATAN
  R*4   MTH$SQRT
  R*4   MTH$TAN

COMMAND QUALIFIERS

  FOR/LIS FILTBANK4.FOR

  /CHECK=(NOBOUNDS,OVERFLOW,NOUNDERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /STANDARD=(NOSYNTAX,NOSOURCE_FORM)
  /SHOW=(NOPREPROCESSOR,NOINCLUDE,MAP,NODICTIONARY,SINGLE)
  /WARNINGS=(GENERAL,NODECLARATIONS)
  /CONTINUATIONS=19 /NOCROSS_REFERENCE /NOD_LINES /NOEXTEND_SOURCE /F77
  /NOG_FLOATING /I4 /NOMACHINE_CODE /OPTIMIZE

COMPILATION STATISTICS

  Run Time:           2.30 seconds
  Elapsed Time:       5.03 seconds
  Page Faults:        283
  Dynamic Memory:     360 pages
```