Evaluation of Frequency Modulation for Reducing Acoustic Feedback in Hearing Aids

by

Kinuko Masaki

Submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 1997

[June 1997]

Author ....
Department of Electrical Engineering and Computer Science
May 23, 1997

Certified by ....
Patrick Zurek
Principal Research Scientist
Thesis Supervisor

Accepted by ....
Arthur C. Smith
Chairman, Department Commitee on Graduate Theses
Electrical Engineering and Computer Science

Evaluation of Frequency Modulation in Reducing Acoustic Feedback in Hearing Aids

by

Kinuko Masaki

Submitted to the Department of Electrical Engineering and Computer Science on May 23, 1997, in partial fulfillment of the requirements for the degree of Master of Engineering in Electrical Engineering and Computer Science

Abstract

One of the major problems faced by hearing aid users when they desire more gain than the hearing aid can deliver is high-intensity oscillation called "whistling." This problem is due to acoustic feedback of the input signal to the microphone. In this thesis, the ability of frequency modulation to reduce this acoustic feedback was investigated. A real-time implementation of the algorithm was done on a DSP chip and both electroacoustic and psychoacoustic tests were made. It was found that this algorithm delivered a maximum additional stable gain of 7 dB.

Thesis Supervisor: Patrick Zurek
Title: Principal Research Scientist

Acknowledgements

This work would not have been possible without the support and guidance from many in the Sensory Communication Group of the Research Laboratory of Electronics. I would especially like to thank my supervisor, Patrick Zurek, for being very patient with me and always guiding me in the correct direction. Thank you very much for editing this thesis about a hundred times. I would also like to thank Julie Greenberg without whose help I would never have finished this thesis on time. Thank you very much for letting me call you even at your home for debugging tips. I am also grateful to Jay Desloge for always being there to answer some Matlab and signal processing questions.

I would also like to thank my family very much for always being very supportive of me throughout my life especially through my thesis project. Thank you very much for letting me stay on campus so many nights so I could finish my thesis. Rosanne, thank you very much for being the best roommate and friend I could ever wish to have. You were always there to cheer me up when I was down or when I needed someone to talk to. Thank you Sonny for putting up with me all the time. I can't count the number of times I came to distract you from your work. Special thanks to Evan for staying up with me until the early morning just to keep me company. I don't think I would have survived those long nights at Athena without you. Thank you also for always encouraging me and helping me keep a smile on my face. Thanks again to everyone for your support and guidance.

Contents

1 Introduction **7**

1.1 Acoustic Feedback . . . 7

1.1.1 Hearing Aids . . . 7

1.1.2 Nyquist Stability Criterion . . . 8

1.2 Review of Different Methods . . . 8

1.2.1 Method of Evaluation . . . 8

1.2.2 Gain-Reduction Method . . . 9

1.2.3 Frequency-Shifting Method . . . 9

1.2.4 Adaptive Feedback Cancellation . . . 9

1.3 Frequency Modulation . . . 10

2 Implementation of Frequency Modulation **12**

2.1 System Layout . . . 12

2.1.1 Filtering . . . 12

2.2 Frequency Shifting - The Simplified Case . . . 13

2.3 Frequency Modulation . . . 17

2.4 DSP Chip Implementation . . . 19

3 Methods of Evaluating the Algorithm **22**

3.1 Electroacoustic Test . . . 22

3.2 Psychoacoustic Tests . . . 23

4 Results **25**

5 Discussion **29**

5.1 Summary of Analysis 29

5.2 Suggestions for FutureWork . . . 29

A1 Electroacoustic and Psychoacoustic Results **30**

A2 DSP Assembly Language Programs **33**

References **47**

List of Figures

1.1 Feedback System Function 7

2.1 Block diagram of the frequency modulation system 12

2.2 Bandpass filter with fc = 3150 Hz 13

2.3 x[n] with fx/fs = 1/12 and A = 1 14

2.4 y[n] with c = -0.5 15

2.5 y[n] when c = -0.1 15

2.6 y[n] when c = 0.1 16

2.7 Implementation of interpolation 17

2.8 A period of the function with Am = 1 and fm = 5. 17

2.9 Relationship between n and n' when Am = 1 and fm = 5 18

2.10 Result of frequency modulating a 500 Hz sinewave with Am = 1 and fm = 5 . . . 19

3.1 Setup of Feedback Simulator. 23

3.2 Relationship between maximum additional stable gain and Am and fm 26

3.3 Relationship between speech quality and gain for Fm = 1 27

3.4 Relationship between speech quality and gain for Fm = 2 28

3.5 Relationship between speech quality and gain for Fm = 5 28

3.6 Relationship between speech quality and gain for Fm = 10 28

Chapter 1

Introduction

1.1 Acoustic Feedback

One of the major problems faced by hearing aid users when they desire more gain than the hearing aid can provide is high-intensity oscillation know as "whistling." This problem is not only annoying to the user but it also prevents low-level signals from being audible. "Whistling" is caused by acoustic feedback of the signal which leaks from the hearing aid receiver back to the microphone. To better understand this problem, *in situ* hearing aids can be modeled using control theory. Then by using Nyquist's stability criterion the conditions under which acoustic feedback occurs can be established.

1.1.1 Hearing Aids

The microphone for the hearing aid is usually placed near or inside of the pinna of the ear. This microphone converts the sound pressure at this location to an electrical signal that is amplified and then drives the receiver, resulting in sound pressure at the tympanic membrane. This, however, is not the complete signal path since sound can be lost through the vents in the hearing aid mold and then fed back to the microphone.

By modeling both the forward and the feedback pathway as individual systems described by system functions G and H, respectively, a control system model of the hearing aid can be established as shown Figure 1.1. Nyquist's stability criterion can then be applied to the *in situ* hearing aid.

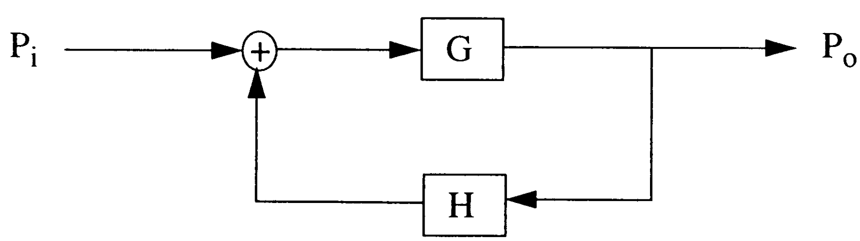

Figure 1.1: Feedback System Function

1.1.2 Nyquist Stability Criterion

Given a control system that can be modeled as Figure 1.1, the value of the output after n passes through the loop is given as:

$$P_0 = G[1 + (GH) + (GH)^2 + \ldots + (GH)^n]P_i \tag{1.1}$$

where GH is the open-loop transfer function. If the magnitude of the open-loop transfer function (=|GH|) is less than 1 and the number of times through the loop approaches infinity, the output can be calculated as

$$P_0 = \left(\frac{G}{1-GH}\right)P_i \tag{1.2}$$

The system function is stable if $|GH| < 1$. However, if the magnitude of the open-loop transfer function is greater than or equal to one and the phase of the open-loop transfer function is a multiple of $360°$ then the output approaches infinity and the system becomes unstable. In an unstable system, an input P_i is only needed to initiate the process which continues afterwards without any input. Therefore, to terminate the feedback process, the amplifier gain, G, must be reduced.

1.2 Review of Different Methods

Several signal processing methods to mitigate the feedback problem have been proposed, especially for public address systems. Egolf [1] reviewed some of the acoustic feedback literature in that field and suggested that some of these algorithms could be adapted for reducing acoustic feedback in an *in situ* hearing aid.

1.2.1 Method of Evaluation

The amount of attenuation in the feedback path of a hearing aid is related to the maximum stable gain of the hearing aid. Maximum stable gain is the maximum gain of the hearing aid before it becomes unstable. However, in comparing different methods, it is important not only to compare the stable gain of each but also their effects on sound quality and the amount of annoyance it causes the listener.

1.2.2 Gain-Reduction Method

As discussed above, one of the most direct ways to reduce acoustic feedback is by reducing the amplifier gain. Boner and Boner [2] inserted a notch filter before the amplifier to eliminate the peak amplitude in the system function which caused the instability. However, this requires that the magnitude of the open-loop spectrum be measured so that the peaks in the magnitude can be found. The problem with this algorithm is that the open-loop spectrum is dependent on the speaker and microphone arrangement and also the acoustic environment. Therefore, Maxwell and Zurek [3] examined an adaptive notch filter in which the center frequency of the notch is adapted to reduce the largest spectral peak in the environment. They however found that feedback-reduction techniques such as the single adaptive notch filter which directly reduces the gain of the forward path are effective only if the feedback path is relatively narrowband. This implies that the magnitude of the feedback path can only have one prominent peak that is narrower than the notch width [1].

1.2.3 Frequency-Shifting Method

In this method, the signal is frequency shifted by a given amount, say 5 Hz, so that the output would be a frequency shifted version of the input [4]. Even though a maximum additional stable gain of 10-12 dB was obtained, the intelligibility of the speech was sacrificed. The subjects heard "audible beating" when the gain was greater than 6 dB. Therefore, the maximum additional stable gain obtainable while retaining good speech quality is only about 6 dB [1].

1.2.4 Adaptive Feedback Cancellation

Adaptive feedback cancellation methods have been studied by a number of investigators [3, 5, 6, 7]. These methods attempt to prevent oscillation due to acoustic feedback by cancelling the feedback path. The output of the hearing aid is filtered with an estimate of the feedback transfer function, H. The resultant signal becomes the estimated feedback signal and this is subtracted from the input signal of the hearing aid. However, the exact method of estimation and adaptation of these feedback signals depends on the implementation chosen and this also affects the maximum additional stable gain achievable. According to measurements from several reports [3], the maximum additional stable gain achievable with this method is approximately 12 dB. Therefore, adaptive feedback cancellation systems seem to allow substantial increase in the wideband system gain. Maxwell and Zurek found that continuously-adapting systems distorted the input signal and also were inherently unstable. Maxwell and Zurek proposed a quiet-interval adaptation method that attempts to interrupt the signal not only when oscillation is detected but also when the input signal is estimated to be low. According to Maxwell and Zurek [3], this system performed significantly better than other adaptive feedback cancellation systems in providing maximal feedback cancellation with minimal disturbance to the user. They achieved a maximum additional stable gain comparable to that of the adaptive feedback cancellation system (e.g. 12 dB) but the quality of the sound was nearly perfect.

1.3 Frequency Modulation

Finally, another plausible procedure for reducing feedback in hearing aids proposed by Nishinomiya [8] is frequency modulation. In this method, the output signal is frequency modulated so that the stationary feedback relationship between the receiver and the microphone is broken. The modulation will prevent the feedback signal from being continuously in phase with the incoming signal. According to Egolf [1], Nishinomiya obtained 7 dB additional stable gain using this method. Nishinomiya pointed out that only frequency ranges where feedback is most likely should be modulated to prevent listener annoyance; frequencies below 500 Hz should be passed untouched through the system to prevent "warbling." It was also found by Engebretson et al. [6] that typical feedback paths in hearing aids are much stronger in higher frequencies and therefore a feedback-reduction method should not be rejected just because it is unacceptable in lower frequencies. Egolf also suggested in his paper that this method should be tested to examine the effect it has on speech perception.

Therefore, in this thesis, the frequency modulation algorithm will be evaluated for reducing acoustic feedback in hearing aids. Specifically, the algorithm will first be implemented in Matlab to specify the details of the algorithm and to test if this algorithm is even promising. Then the algorithm will be implemented on a Motorola DSP96002 DSP chip and the acoustic feedback path will by simulated with an electrical feedback path. An electrical feedback path will be used to ensure repeatability. The output of this system will then be evaluated for not only its added stable gain but also its effect on speech quality.

Chapter 2

Implementation of Frequency Modulation

2.1 System Layout

Figure 2.1 shows a block diagram of the major signal-processing components. The system inside the dotted lines was implemented on a DSP chip. X is the input speech signal. G is the gain of the hearing aid. As can be seen in the figure, only a certain band of frequencies centered around a given frequency, f_c, are frequency modulated to disrupt the feedback path. Ideally, f_c will be very close to the frequency where there is a peak magnitude in the feedback path. Frequencies above and below the cutoff frequency of the bandpass filter are passed through the system without modification and are later summed with the frequency-modulated output. The arrow from the output to the input represents the feedback path. This path was simulated with an electronic bandpass filter, as described below.

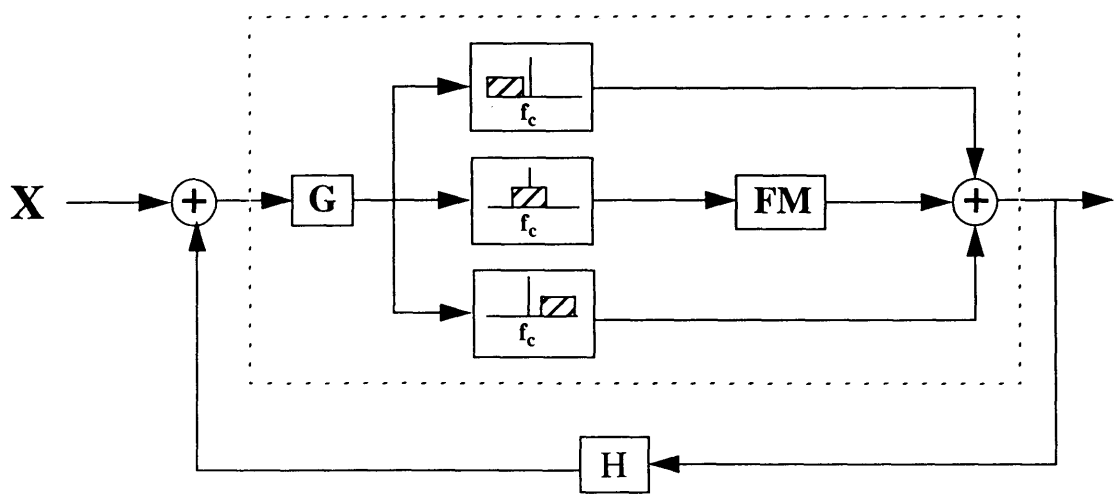

Figure 2.1: Block diagram of the frequency modulation system

2.1.1 Filtering

For the lowpass, bandpass, and highpass filters, the Parks-McClellan optimal finite impulse response (FIR) filter design was used. Also in creating the filters, the design criterion was to create a 35 dB difference between the passband and the stopband using the minimum number of coefficients to maximize the computation speed. Finally, the width of the bandpass filter was chosen to be one octave since we can expect to be able to determine the likely oscillation feedback frequencies to within this precision. Figure 2.2 shows the bandpass filter that was used.

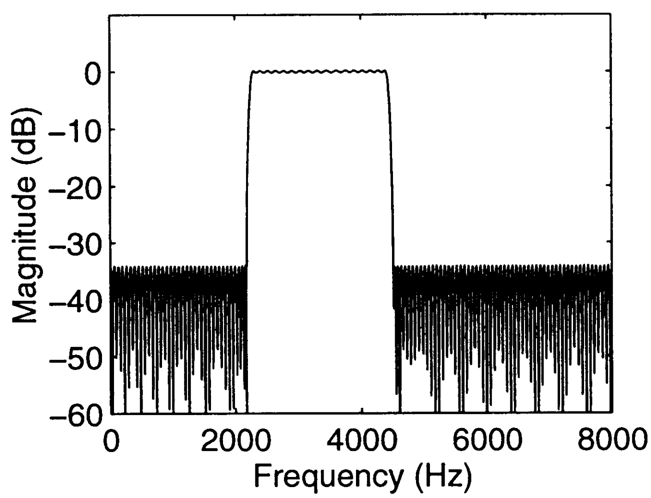

Figure 2.2: Bandpass filter with $f_c = 3150$ Hz

2.2 Frequency Shifting - The Simplified Case

The algorithm we are implementing is slightly different from traditional radio frequency modulation. Normally, a sinusoid is modulated with an information-carrying signal, but in this case, the signal will be modulated with a sinusoid. For explanation purposes, the signal is assumed to be a simple sinusoid. Therefore,

$$x[n] = A\sin\left(\frac{2\pi f_x n}{f_s}\right) \tag{2.1}$$

where A represents the magnitude and f_x is the frequency of the input and f_s is the sampling frequency. An example is shown in Figure 2.3.

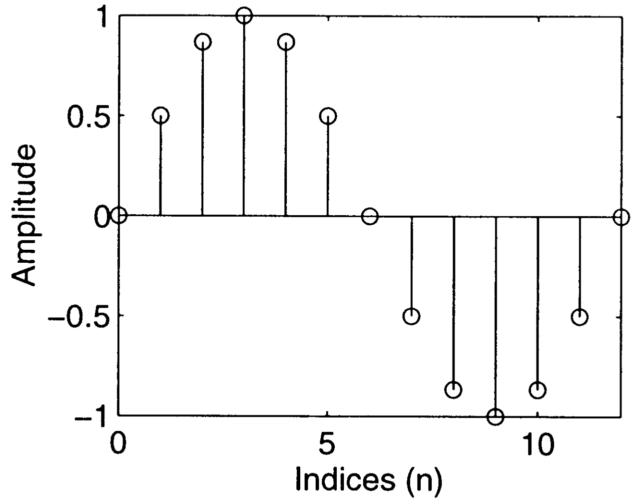

Figure 2.3: x[n] with $f_x/f_s = 1/12$ and A = 1

A speech signal, as is the case for all signals, can be represented as a sum of many sinusoids in a Fourier series. The output signal is then the frequency modulated version of the input signal. However, we will initially consider a simpler case in which we only want a simple frequency shift of (1 + c). Therefore, the desired output signal can be represented as a signal with the same amplitude as the input but with a different frequency, $f_x(1 + c)$. Specifically, assuming that values of x in-between samples are available, the output would be:

$$y[n] = A\sin\left(\frac{2\pi f_x(1+c)n}{f_s}\right) = x[(1+c)n] = x[n'] \tag{2.2}$$

Therefore, if we wanted to reduce the frequency of the input by a factor of 2, c would equal -0.5. Then, the output would look like Figure 2.4. The filled circles represent the original data points and the dotted lines show how the interpolated points are mapped to the new indices.

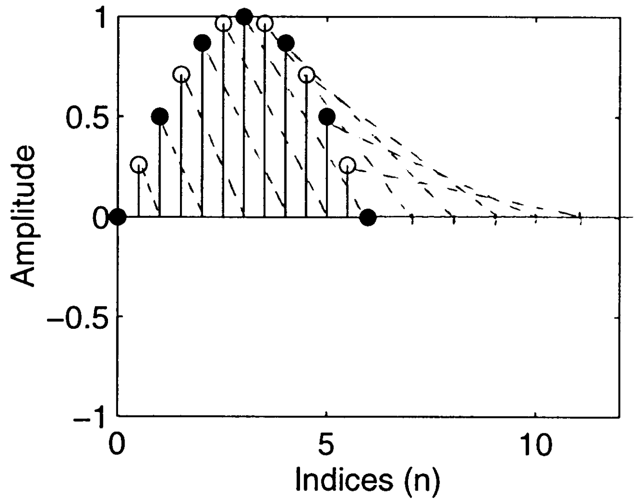

Figure 2.4: y[n] with c = -0.5

As long as c is less than 0, then only past sample values are used. For example, if c = -0.1 then the output would look like:

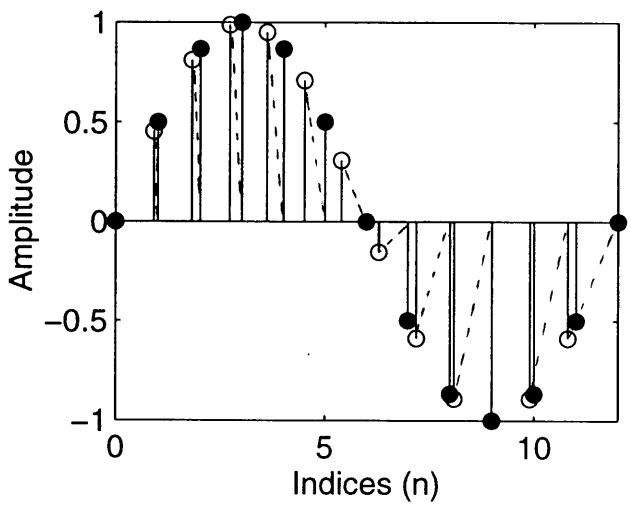

Figure 2.5: y[n] when c = -0.1

However, if c is greater than 0 then this algorithm requires future samples. For example, if c is equal to 0.1 then the output would look like Figure 2.6 again with the filled circles representing the original data points.

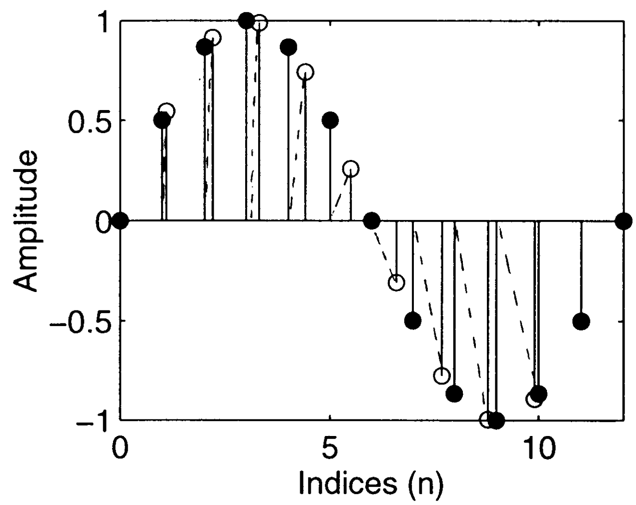

Figure 2.6: y[n] when c = 0.1

To obtain values of x[n] when n' is no longer an integer, we do an interpolation between two sample points. For example, if we want to get a sample value for an intermediate time (1+c)n between two samples, a simple technique is to interpolate linearly between them to estimate the value, x', of the function at (1+ c)n. To do this let

$$n1 = \text{floor}((1+c)n) \tag{2.3}$$

$$n2 = \text{ceil}((1+c)n) \tag{2.4}$$

$$r = (1+c)n - n1 \tag{2.5}$$

where floor is a function which rounds to the nearest integer toward minus infinity and ceil is a function which rounds to the nearest integer toward positive infinity. After these variable are calculated, next compute the slope between the two sample points:

$$m = x[n2] - x[n1] \tag{2.6}$$

Then, the linearly-interpolated x' is:

$$x' = mr + x[n1] = x[n1] + (x[n2]-x[n1])[(1+c)n - n1] \tag{2.7}$$

This is illustrated in Figure 2.7.

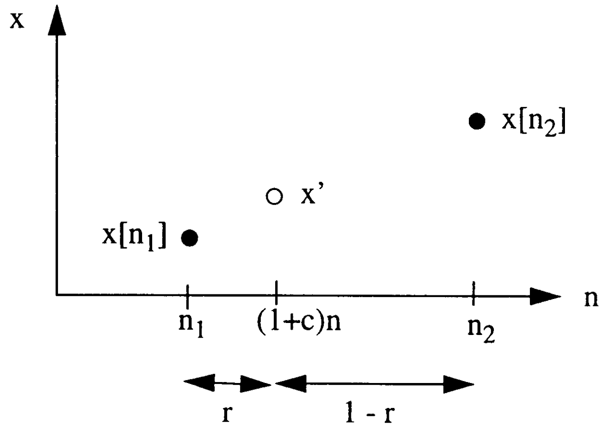

Figure 2.7: Implementation of interpolation

2.3 Frequency Modulation

Time-varying frequency modulation is very similar to the frequency shifting algorithm discussed above except that we now have to consider the rate that samples are outputted. This new sample release rate is expressed by the following equation:

$$n' = \Delta n\left[1 + (-A_m)\sin\left(\frac{2\pi f_m n}{f_s}\right)\right] \tag{2.8}$$

where A_m is the maximum degree of frequency modulation and f_m is the frequency of modulation. A period of this function is illustrated in Figure 2.8.

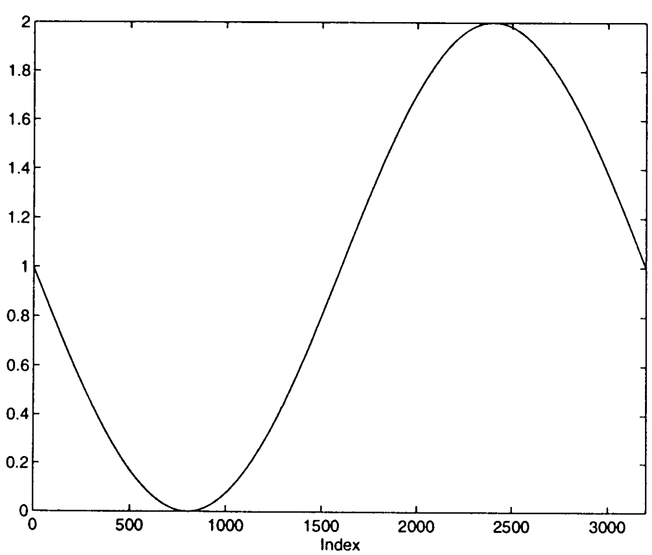

Figure 2.8: A period of the function with $A_m = 1$ and $f_m = 5$.

This implies that input frequencies will be modulated up by a factor of $(1 + A_m)$ at the peak of the modulation cycle and downward by a factor of $(1 - A_m)$ at the minimum of the cycle. To obtain the modified sample time as a function of the original sample time, summation is done over all the previous rate changes starting at index 0 as shown in Equation 2.9.

$$n'(n) = \sum_{k=0}^{n}\left(1 - A_m \sin\left(\frac{2\pi f_m k}{f_s}\right)\right) = n - \sum_{k=0}^{n} A_m \sin\left(\frac{2\pi f_m k}{f_s}\right) \tag{2.9}$$

A period of the modified sample time, n', is shown below.

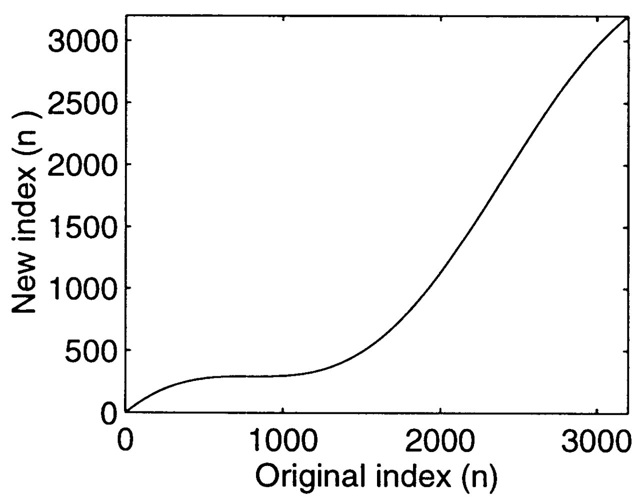

Figure 2.9: Relationship between n and n' when $A_m = 1$ and $f_m = 5$

From this figure, it can be seen that when the slope of this function is less than one, the input signal is being stretched because the function is indexing at a lower rate compared to the original. On the other hand, when the slope is greater than one, the input signal is being compressed. However, one thing we have to assure is that future samples are never required. Therefore, $n' \le n$. This implies that

$$n - \sum_{k=0}^{n} A_m \sin\left(\frac{2\pi f_m k}{f_s}\right) \le n \tag{2.10}$$

Therefore, if Equation 2.11 is satisfied then we can be assured that no future samples are needed.

$$\sum_{k=0}^{n} A_m \sin\left(\frac{2\pi f_m k}{f_s}\right) \geq 0 \tag{2.11}$$

If the summation can be approximated by an integration, then the above equation is guaranteed to be satisfied. Figure 2.10 shows the result of frequency modulating a 500 Hz sinewave with $A_m = 1$ and $f_m = 5$.

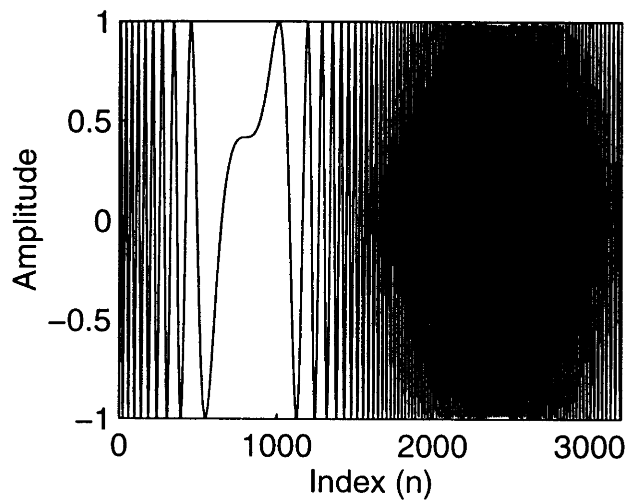

Figure 2.10: Result of frequency modulating a 500 Hz sinewave with $A_m = 1$ and $f_m = 5$

This figure correlates with the result showed in Figure 2.7. Near the index value of 800, the input is stretched maximally and near the index value of 2400 it is being maximally compressed.

2.4 DSP Chip Implementation

The frequency modulation algorithm was implemented on a digital signal processing (DSP) microprocessor, specifically, a Motorola DSP96002 (Ariel D96). The real-time implementation was important in evaluating the algorithm's performance under dynamic conditions.

To decrease the amount of real-time processing that had to be done on-line, all variables that were independent of the input signal were precomputed in Matlab and loaded onto the DSP board at the beginning of the processing. Therefore, the frequency modulation algorithm was initiated by first running the Matlab function **fmf6.m** which computed and then loaded all the constants into the appropriate buffers on the DSP board. This included not only the frequency and amplitude of modulation (f_m and A_m) but also all the filter coefficients and many variables needed for the frequency modulation algorithm. The variables f_m and A_m were defined in Matlab so that they could be easily changed and implemented without having to recompile the DSP code. When loading the scaling and indexing functions for the frequency modulation algorithm, it was very important to make sure that a full period of these functions was saved. A full period corresponded to the closest integer value of f_s/f_m where f_s is the sampling frequency which in this case was 16 kHz and f_m was the frequency of modulation (or warbling).

Of course, the DSP processing had to be accomplished in real-time, using only past (stored) samples. At the start, the DSP chip was called when all the constants were loaded into the appropriate DSP registers by Matlab. The board then captured a single sample from the A/D converter and appended it to the input buffer. This buffer was then filtered so that only the bandpass filtered input centered at the frequency of the maximum feedback was frequency modulated and the other frequency ranges were just passed through. The algorithm to do frequency modulation was identical to that described above. However, the indexing scheme for the DSP chip had to be altered. First of all, since there were a limited number of registers, two registers had to be split and used to store two variables. Therefore, one had to always keep track of the distance between the different variables in a given register and where the pointer for a variable was in respect to the other variable. When implementing the frequency modulation index, the bandpass filtered input was indexed by keeping one pointer constantly pointed to the beginning of the bandpass filtered input and the other pointer was incremented with respect to the initial pointer by means of the value specified in the indexing buffer. When the second pointer reached the end of the indexing function, it was again forced to point to the beginning of the indexing function. The output of the frequency modulation processing was summed with the high-pass and lowpass filtered signals and put in the output buffer. This whole process was repeated continuously until the code was terminated.

Chapter 3

Methods of Evaluating the Algorithm

The frequency modulation algorithm was evaluated electroacoustically and psychoacoustically. In particular, the maximum stable gain and sound quality were measured for different combination of values for the frequency and amplitude of modulation.

3.1 Electroacoustic Tests

To initially test the assembly code written to implement the frequency modulation code discussed above, a storage oscilloscope was used to analyze the output when a sinewave was used as input to the system. The result was then compared to the theoretically predicted output shown in Figure 2.10.

Once it was confirmed that the DSP board was implementing the desired algorithm, a third-octave bandpass filter centered at 3.15 kHz with gain of 20 dB was used to model the frequency response of the feedback path (see Figure 3.1). The gain in the forward path was varied to determine the maximum stable gain achievable with the system. The input to this system was the processed output of the DSP board, which was monitored both auditorily (via the earphone driver) and visually (with an oscilloscope). The unprocessed signal from the Ariel DSP board was similarly monitored. This was done so that the unprocessed signal could be compared both visually and auditorily with the processed signal.

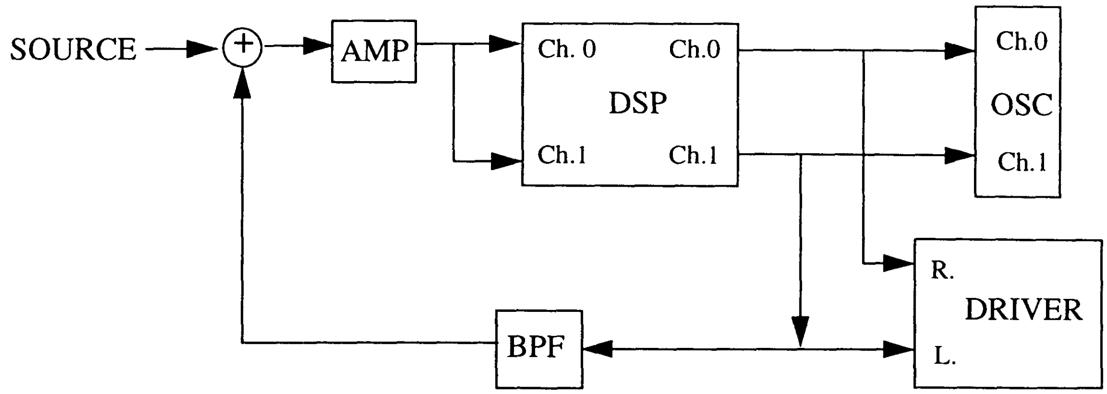

Figure 3.1: Setup of Feedback Simulator.

The additional stable gain provided by the frequency modulation algorithm was measured by first obtaining the maximum stable gain with no frequency modulation as a baseline. Specifically, the gain was gradually increased until the system became unstable. Next, the maximum stable gain with the frequency modulation algorithm in the forward path was measured in a similar fashion. The additional stable gain provided by the algorithm is then the difference between the stable gain with and without frequency modulation. In other words, it is a measure of how much processing improves the gain achievable by the hearing aid. The frequency modulation algorithm was implemented using different combinations of f_m, and A_m to determine which combination of parameters gives the best result.

3.2 Psychoacoustic Tests

For hearing aid applications, it is not only the maximum stable gain provided by an algorithm that is important but also any effects on the quality of speech. For example, even if the algorithm achieves a high value of added stable gain, if it distorts the speech signal beyond recognition then it would be practically useless. Therefore, measurements of sound quality were made at three different gain levels: at the maximum stable gain, 3 dB below the maximum stable gain, and finally 6 dB below the maximum stable gain. The sound quality was assessed by a rating between 1 and 10 where a value of 1 corresponded to speech quality that was completely unacceptable and 10 corresponded to quality as good as that of the original speech input. The speech input was a CD recording of the Rainbow Passage (Q/MASS Speech Audiometry, Volume 3).

Three normal-hearing young adults served as subjects for rating sound quality. Each subject made one rating of each combination of f_m and A_m.

Chapter 4

Results

F_m	A_m	Max. Gain	Thres. rating	-3dB Rating	-6dB Rating
1	0.001	0.9	1	8.00	9.67
1	0.002	1.0	1	8.17	9.83
1	0.005	1.2	1	8.00	9.33
1	0.01	1.7	2	8.00	9.67
1	0.02	2.5	2	7.00	9.00
1	0.05	4.2	2.33	6.00	8.00
1	0.1	6.3	2	4.00	6.33
2	0.001	1.0	1	8.33	9.50
2	0.002	0.9	1	8.00	9.50
2	0.005	1.0	1.67	8.00	9.5
2	0.01	1.2	1	8.33	9.67
2	0.02	1.8	1.33	8.17	9.33
2	0.05	3.3	2.33	7.83	9.00
2	0.1	7	2	4.33	6.67
5	0.001	0.8	1	8.00	10.00
5	0.002	0.8	1	8.17	9.83
5	0.005	0.9	1	7.67	9.33
5	0.01	1.0	1	7.67	9.5
5	0.02	1.1	1	7.5	9.33
5	0.05	1.8	1	7.67	9.5
5	0.1	7.1	2	4.67	6.83
10	0.001	1.0	1	7.67	10.00
10	0.002	0.8	1	7.67	9.67
10	0.005	1.0	1	7.33	9.33
10	0.01	0.9	1	7.67	9.83
10	0.02	1.0	1	8.17	9.83
10	0.05	1.2	1	8.00	9.67
10	0.1	1.9	1	7.33	8.67

Table 4.1: Summary of electroacoustic and psychoacoustic tests

The maximum additional stable gain and the subjective ratings at the threshold, 3 dB below threshold, and 6 dB below threshold are summarized in Table 4.1. The ratings are a result of averaging over the values given by 3 different listeners. (Individual subject's results are included in the Appendix). Figure 4.1 shows how maximum additional gain varies as a function of both A_m and f_m.

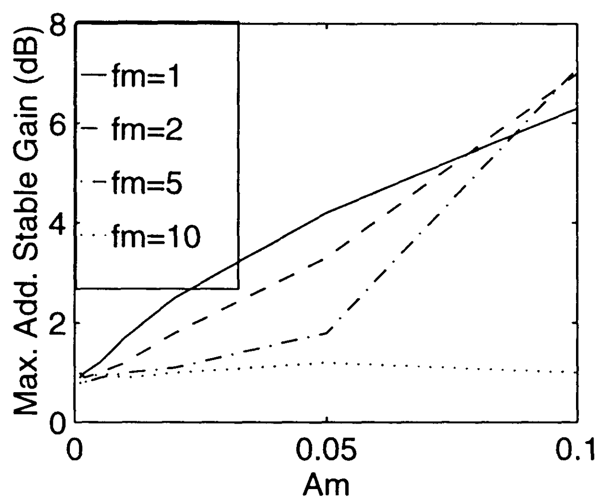

Figure 4.1: Relationship between maximum additional stable gain and A_m and f_m

From this figure it can be seen that when f_m is equal to 1, 2, and 5 Hz, the maximum additional stable gain increases as a function of A_m. But when f_m is equal to 10 the gain is almost independent of A_m and is very small.

The maximum gain of a system is usually defined as the maximum gain possible without the system becoming unstable. Instability is usually defined as the point at which the output of the system continues to grow indefinitely even when the input is kept constant. This definition works satisfactorily for A_m less than 0.02. However, when A_m is greater than 0.02 there can be extremely large thresholds. However, the quality of the speech is so poor over much of this range of gains that the definition seems overly restrictive. Therefore, in the tests done above, instability was defined as the point where the output signal is substantially prolonged after termination of the input.

The trade-off between speech quality and gain (re. the instability point with no processing) is shown in Figure 4.2 through Figure 4.5. Each of these figures is a plot for one value of frequency modulation, f_m, with A_m as the parameter. In the four figures below, the solid line represents the relationship between speech quality and gain when no processing is done. The ". . . ." line is for $A_m = 0.001$; "o" symbol is for $A_m= 0.002$; "+" symbol is for $A_m = 0.005$; " * " symbol is for $A_m = 0.01$; "x" symbol is for $A_m = 0.02$; "- - - -" line is for $A_m = 0.05$, and "- . - ." line is for $A_m = 0.1$.

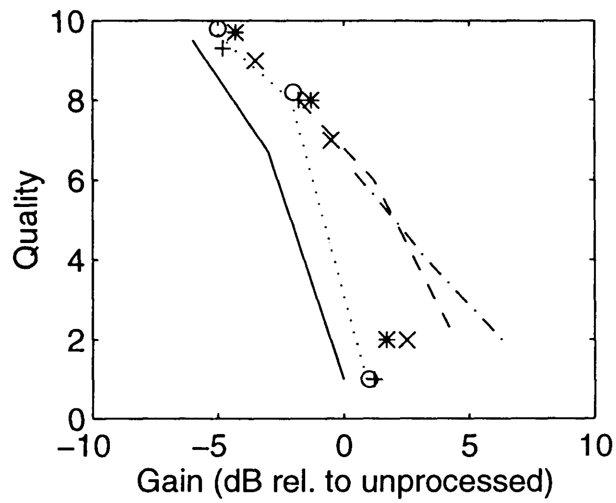

Figure 4.2: Relationship between speech quality and gain for $F_m = 1$

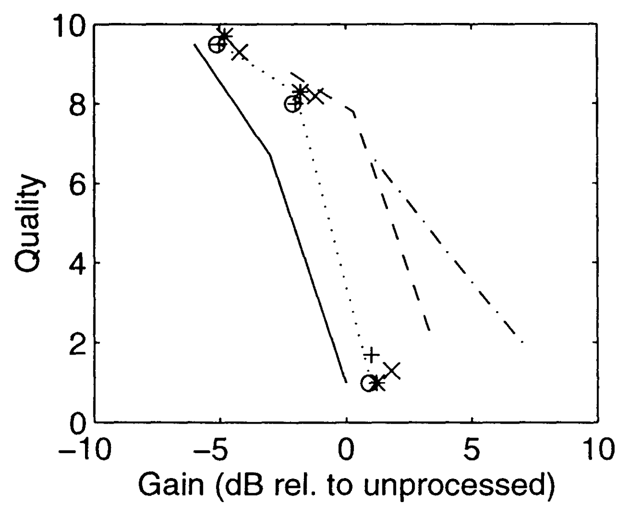

Figure 4.3: Relationship between speech quality and gain for $F_m = 2$

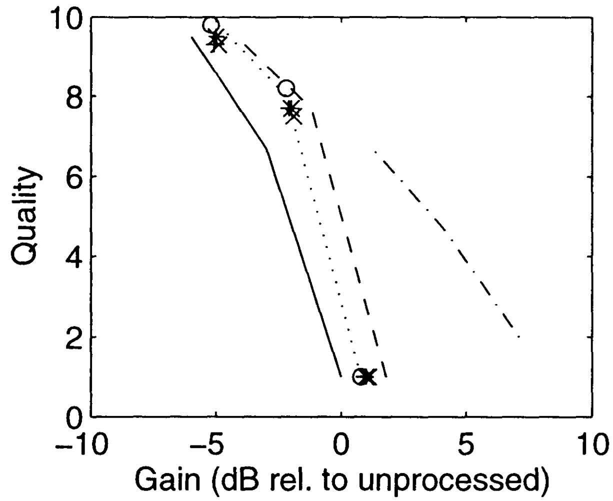

Figure 4.4: Relationship between speech quality and gain for $F_m = 5$

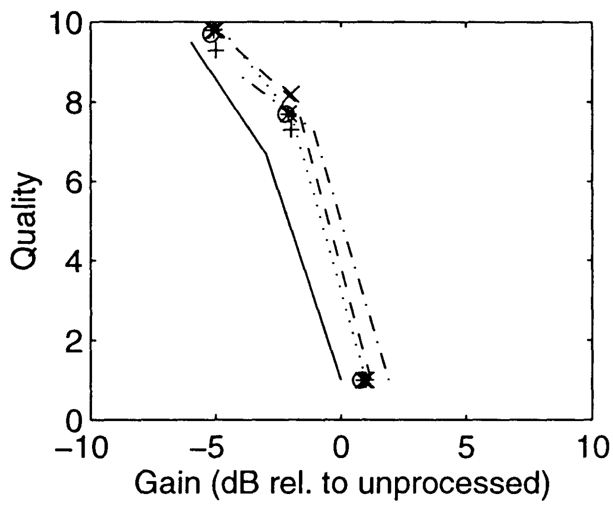

Figure 4.5: Relationship between speech quality and gain for $F_m = 10$

Figures 4.2 - 4.5 show that as we increase the amount of gain, the speech quality deteriorates. However, as we increase the amount of frequency modulation (i.e. increase A_m), the speech quality improves for any given gain value. From these figures, it can be seen that a modulation frequency of 5 Hz gives the best tradeoff between speech quality and gain.

Chapter 5

Discussion

5.1 Summary of Analysis

The method of using band-limited frequency modulation to reduce acoustic feedback in hearing aids was evaluated. This was done by implementing the algorithm on a DSP board, modeling the feedback path electrically, and then measuring the additional stable gain and sound quality that could be achieved. The present results confirmed those of Nishinomiya [8], who found only a 7 dB additional stable gain is possible with this algorithm. However, our results also showed that for the A_m and f_m values needed to achieve 7 dB gain, the quality of speech is degraded even when the gain is 6 dB below the threshold value. This study, therefore, like that of Maxwell and Zurek [3] shows the importance of not only doing electroacoustic but also psychoacoustic tests.

5.2 Suggestions for Future Work

If the frequency range of likely oscillation could be narrowed down to a smaller frequency range, then a possible way of improving this algorithm would be to frequency modulate over a narrower frequency range compared to the one octave frequency band that was used in this implementation. This would reduce the amount of "warbling" heard by the listeners.

To understand how this algorithm compares to other algorithms, a direct comparison should be made amongst the different algorithms using the same conditions (e.g. feedback path). This would ensure that no subtle differences are missed.

Appendix A

Electroacoustic and Psychoacoustic Results

F_m	A_m	Max. Gain	Thres. rating	-3dB Rating	-6dB Rating
1	0.001	0.9	1	8	10
1	0.002	1.0	1	8	10
1	0.005	1.2	1	8	9
1	0.01	1.7	1	8	10
1	0.02	2.7	1	7	9
1	0.05	4.1	1	6	8
1	0.1	6.6	1	2	6
2	0.001	1.0	1	8	9.5
2	0.002	0.9	1	7	9
2	0.005	1.0	1	7	9
2	0.01	1.2	1	8	10
2	0.02	1.8	1	8	9
2	0.05	3.6	1	7	8
2	0.1	7.6	1	2	4
5	0.001	0.8	1	8	10
5	0.002	0.8	1	8	10
5	0.005	0.9	1	7	9
5	0.01	1.0	1	7	9
5	0.02	1.1	1	7	9
5	0.05	1.6	1	8	10
5	0.1	7.5	1	3	5
10	0.001	1.0	1	8	10
10	0.002	0.8	1	7	9
10	0.005	1.0	1	7	9
10	0.01	0.9	1	7	10
10	0.02	1.0	1	8	10
10	0.05	1.2	1	8	10
10	0.1	2.0	1	7	9

Table A.1: Electroacoustic and psychoacoustic results from Subject 1

F_m	A_m	Max. Gain	Thres. rating	-3dB Rating	-6dB Rating
1	0.001	0.9	1	9	10
1	0.002	1.1	1	9	10
1	0.005	1.2	1	9	10
1	0.01	1.6	3	9	10
1	0.02	2.6	3	8	10
1	0.05	4.6	3	7	9
1	0.1	5.2	3	7	8
2	0.001	1.0	1	9	10
2	0.002	1.0	1	9	10
2	0.005	1.0	1	9	10
2	0.01	1.2	1	9	10
2	0.02	1.8	1	9	10
2	0.05	3.4	3	9	10
2	0.1	6.5	3	7	9
5	0.001	0.8	1	9	10
5	0.002	0.8	1	9	10
5	0.005	0.9	1	9	10
5	0.01	1.0	1	9	10
5	0.02	1.1	1	9	10
5	0.05	1.9	1	8	10
5	0.1	6.7	3	7	9
10	0.001	1.0	1	9	10
10	0.002	0.8	1	9	10
10	0.005	1.0	1	9	10
10	0.01	0.9	1	9	10
10	0.02	1.0	1	9	10
10	0.05	1.2	1	9	10
10	0.1	1.9	1	9	10

Table A.2: Electroacoustic and psychoacoustic results from Subject 2

F_m	A_m	Max. Gain	Thres. rating	-3dB Rating	-6dB Rating
1	0.001	0.8	1	7	9
1	0.002	1.0	1	7.5	9.5
1	0.005	1.1	1	7	9
1	0.01	1.9	2	7	9
1	0.02	2.3	2	6	8
1	0.05	3.9	3	5	7
1	0.1	7.0	2	3	5
2	0.001	0.9	1	8	9
2	0.002	0.9	1	8	9.5
2	0.005	1.1	3	8	9.5
2	0.01	1.2	1	8	9
2	0.02	1.7	2	7.5	9
2	0.05	3.0	3	7.5	9
2	0.1	7	2	4	7
5	0.001	0.8	1	7	10
5	0.002	0.8	1	7.5	9.5
5	0.005	0.8	1	7	9
5	0.01	1.0	1	7	9.5
5	0.02	1.0	1	6.5	9
5	0.05	1.8	1	7	8.5
5	0.1	7.2	2	4	6.5
10	0.001	1.0	1	6	10
10	0.002	0.8	1	7	10
10	0.005	1.0	1	6	9
10	0.01	0.9	1	7	9.5
10	0.02	1.0	1	7.5	9.5
10	0.05	1.1	1	7	9
10	0.1	1.9	1	6	7

Table A.3: Electroacoustic and psychoacoustic result from Subject 3

Appendix B

DSP Assembly Language Programs

```
;*****************************************************************************
;
;fmf6.asm       Frequency Modulation and Feedback Algorithm
;               final version with out0 pass-thru result
;               and out1 processed result
;
;     x -----> G --------> hp filter ---------------> + ----->
;              |                        |
;              ---> bp filter ---> freq mod --->
;              |                        |
;              ---> lp filter ----------------->
;
;
;*****************************************************************************
;          Register Usage
;
;     r0 - pointer to newest data in input tapped delay line (IN0)
;     m0 - length of input TDL
;     r1- pointer to current location in proc tapped delay line (PROC0)
;     r2 - pointer to curent location in lp0 tapped delay line (LP0)
;     n2 - offset between lp0/hp0 tapped delay line
;     m2 - length of lp/hp tapped delay line (LPHPLEN- 1)
;     r3 - pointer to index (IND)
;     n3- offset between ind and r
;     r4 - pointer to BP0
;     r5 - pointer to lowpass/highpass filter coeff (LPCOEF)
;     n5 - offset between lp/hp filter coeff
;     m5 - length of highpass filter (HPLEN - 1)
;     r7 - reserved for Janus
;
;*****************************************************************************

     section main

     xdef LPMID,LPLEN,TEMP4,TEMP5,TEMP6,HPCOEF,LPCOEF
     xref lphp

     nolist
     include 'm96equ.a'          ;include the Ariel equates
     include 'jgequ.asm'
     include 'mem6.asm'          ;include memory defintions
```

```
    list
    include 'jgmacros.asm'        ;include Julie's macros
    org    pl:

    move   #1,d0.l              ;at the very beginning . . .
    move   d0.l,x:RESTART         ;set restart flag to wait for matlab
mainstart
    brset  #0,x:RESTART,mainstart  ;wait here for matlab
    jsr    init
mainloop
    move   (r0)+                ;update pointer to input TDL
    jsr    get_data             ;get data and do preprocessing
    move   (r2)+                ;increment pointer to LP/HP TDL
    jsr    lphp                 ;do lowpass/highpass filtering
    move   (r1)+                ;increment pointer to processed TDL
    jsr    fmf0                 ;debugging subroutine
    jsr    put_data             ;postprocess data
    brclr  #0,x:RESTART,mainloop  ;continue if no restart flag
    bra    mainstart

;*****************************************************************************
;
; init        INITIALIZATION SUBROUTINE
;
;*****************************************************************************

init
    ori    #$30,MR              ;disable interrupts
    bclr   #5,x:MI_HCR           ;enable inner port interface to host
    move   #0,d0.s              ;for clearing FP locations
    move   d0.s,x:DCVALS         ;clear DC offset for channel
    move   #STD_BCR,d0.l
    move   d0.l,x:MI_BCR         ;clear inner and outer bus control
    move   d0.l,x:MO_BCR         ;registers for zero wait states
    clr    d0.l
    move   d0.l,x:IOSTAT         ;clear status flags for ISR
    move   d0.l,x:PASSFLG        ;clear passthru flag
    move   #IN0,r0              ;set up address registers
    move   #PROC0,r1
    move   #LP0,r2
    move   #BP0,r4
    move   #LP0,d0.l
    move   #HP0,d1.l
    sub    d0.l,d1.l
    move   d1.l,n2
    move   x:<Nmax,d0.l
```

```
    dec   d0
    move  d0.l,m2
    move  d0.l,m4
    move  d0.l,m3
    move  #IND,r3
    move  #IND,d0.l
    move  #R,d1.l
    sub   d0.l,d1.l
    move  d1.l,n3
    move  x:LPLEN,d0.l
    dec   d0.l
    move  d0.l,m5
    move  d0.l,m0
    move  #LPCOEF,d1.l
    move  #HPCOEF,d2.l
    sub   d1.l,d2.l
    move  d2.l,n5


    clr   d0.l
    move  d0.l,y:DAU_CR              ;suggested by David Lum
    movep #ADA_DACEN|ADA_N16,y:DAU_CR    ;set 16 kHz sample rate
    bset  #0,y:DAU_INTEN         ;enable interrupts from analog I/O
    bset  #B_IAL0,x:M_IPR        ;IRQA priority level 0
    bclr  #B_IAL1,x:M_IPR        ;IRQA priority level 0
    bset  #B_IAL2,x:M_IPR        ;IRQA edge triggered
    bset  #BO_HBL0,x:M_IPR         ;also David Lum's
    bset  #BO_HBL1,x:M_IPR         ;also David Lum's
    andi  #$cf,MR            ;enable level 0 interrupts for I/O
    rts

;***************************************************************************
;
; get_data     DATA ACQUISITION SUBROUTINE
;
;***************************************************************************

get_data
    move  #DCLPB,d8.s              ;set up coeffs in advance
    move  #DCLPA,d9.s
wait_here
    brclr #IOREADY,x:IOSTAT,wait_here    ;wait here 'til new data ready
    move  x:<RAWLOC,d1.l             ;get new data from this board
    bclr  #IOREADY,x:IOSTAT          ;clrbit to tell ISR we got it

    split d1.l,d0.l     #A2D,d7.s     ;extract channel 1
```

```
    DCNULL  0,0,7,8,9               ;do DC nulling ch0
    move   d0.s,y:(r0)             ;move new data to IN0 (r0)

    move   d0.s,x:<PASS0            ;store passthru data
    rts

;******************************************************************************
;
; put_data    DATA OUTPUT SUBROUTINE
;
; Takes two floating point values from memory locations OUT0 and OUT1,
; applies digital gain, does software clipping, and then converts them
; to stereo integer format for ISR to write to D/As.
;
; UPDATES:    OUTDATA
; READS:      OUT0, OUT1
; USES:       d0-d2
;
;******************************************************************************

put_data
    brclr  #0,x:<PASSFLG,get_out  ;test passthru flag
    move   x:<PASS0,d0.s         ;get stereo passthru data samples
    bra    after_out
get_out
    move   x:<OUT0,d0.s          ;get stereo output data samples
    move   x:<OUT1,d1.s
after_out
    move   #DGAIN,d2.s
    fmpy.s d2,d0,d0             ;apply digital gain
    fmpy.s d2,d1,d1
    move   #1.0,d2.s            ;check ch0 magnitude
    fcmpm  d2,d0
    fble   noclip0
    fcopys.s d0,d2              ;software clip ch0
    move   d2.s,d0.s
noclip0
    move   #1.0,d2.s
    fcmpm  d2,d1
    fble   noclip1
    fcopys.s d1,d2
    move   d2.s,d1.s
noclip1
    move   #D2A,d2.s            ;scale samples to D/A range
    fmpy.s d2,d0,d0
    fmpy.s d2,d1,d1
```

```
        intrz  d0                ;convert to 16 bit integers
        intrz  d1
        join   d0.l,d1.l         ;combine in one 32 bit word
        move   d1.l,x:<OUTDATA        ;store in location for ISR
        rts

;*************************************************************************
;
; fmf0       Debugging subroutine to test code without freq mod.
;
; out = hp filtered + lp filtered + bp filtered
;
;*************************************************************************


fmf0
        move   r2,r6
        move   y:(r2)+n2,d0.s         ;retrieve lp result
        move   y:(r2)+n2,d1.s         ;retrieve hp result
        move   y:(r2),d2.s          ;retrieve the bp result

        ;start frequency modulation
        move   x:(r3),d5.l          ;ind -> d5
        move   #1.0,d4.s            ;1 -> d4
        move   x:(r3+n3),d3.s         ;r -> d3
        move   d5.l,n4
        fsub.s d3,d4                ;1-r -> d4
        move   y:(r4+n4),d6.s         ;x(ind) -> d6
        inc    d5                 ;ind -> d5
        move   x:<Nmax,d7.l
        cmp    d7,d5
        bne    neq
        move   #0,d5.l
neq
        move   d5.l,n4
        fmpy.s d4,d6,d6             ;(1-r) * x(ind) -> d6
        move   y:(r4+n4),d4.s         ;x(ind+1) -> d4
        fmpy.s d3,d4,d4             ;(r * x(ind+1)) -> d4
        fadd.s d4,d6                ;d4 + d6 -> d6
        move   (r3)+
        fadd.s d1,d0              ;lp + hp --> d0.l;
        fadd.s d0,d2              ;lp + hp + bp --> d2.l (nonprocessed)
        fadd.s d6,d0              ; hp + lp + bp  --> d0.l  (processed)
        move   d2.s,x:<OUT0          ;nonprocessed --> OUT0
        move   d0.s,x:<OUT1          ;processed --> OUT1
        move   r6,r2
        rts
```

```
;****************************************************************************
;
; io_isr      INTERRUPT VECTOR (jump to ISR for long interrupt)
;
;****************************************************************************

    org   p:P_IRQA
    jsr   io_isr

;****************************************************************************
;
; io_isr      INTERRUPT SERVICE ROUTINE
;
;****************************************************************************

    org   pl:
io_isr
    movep   y:DAU_DATA,x:RAWLOC      ;get stereo input from A/Ds
    movep   x:OUTDATA,y:DAU_DATA     ;send stereo output to D/As
    bset    #IOREADY,x:IOSTAT        ;set new data flag
    rti

    endsec
```

```
;*************************************************************************
;
; mem6.asm     MEMORY ALLOCATION FOR BOARD 0
;
;*************************************************************************

LPHPMAX equu   2048            ;maximum length of lp/hp/bp TDL
LPHPLEN equ    LPHPMAX         ;length of TDLs for storing lp/hp results
SINMAX  equ    32768           ;FM min is 0.488 assuming 16 kHz

    org    x:$0                ;on-chip SRAM
RAWLOC  ds     1               ;stereo data from this board
OUTDATA ds     1               ;stereo data for D/A output
IOSTAT  ds     1               ;status flags for ISR
RESTART ds     1               ;restart flag set&cleared via matlab
PASSFLG ds     1               ;passthru flag set&cleared via matlab
DCVALS  ds     1               ;DC values for channel 0
SCALES  dc     1.0             ;scale factors for mic correction
OUT0    ds     1               ;output for channel 0
OUT1    ds     1
PASS0   ds     1               ;pass thru output for channel 0
LPHPOFF ds     1               ;relative offset of TDLs
LPLEN   ds     1               ;length of lowpass filter, must be odd
LPMID   ds     1               ;midpoint of lowpass filter
TEMP1   ds     1
TEMP2   ds     1
TEMP3   ds     1
TEMP4   ds     1
TEMP5   ds     1
TEMP6   ds     1
TEMP7   ds     1
Nmax    ds     1

    org    y:$0                ;on-chip SRAM

    org    x:SRAM1             ;inner bus SRAM in x memory
PROC0   dsm    LPHPMAX         ;processed data TDLs
LPCOEF  dsm    LPHPMAX
HPCOEF  dsm    LPHPMAX
IND     dsm    SINMAX


    org    y:SRAM1             ;outer bus SRAM in y memory
IN0     dsm    LPHPMAX         ;TDLs for input data
```

```
      org    x:DRAM
R       dsm    SINMAX

      org    y:(DRAM+$20000)
LP0    dsm    SINMAX            ;TDLs for LP/HP results
HP0    dsm    SINMAX            ;for OTH, IN0 is left outer
BP0    dsm    SINMAX
```

```
;****************************************************************************
; file kfilter1.asm containts lphp
;
;****************************************************************************
    section kfilter1

    xdef   lphp
    xref   LPMID,LPLEN,TEMP4,TEMP5,TEMP6,HPCOEF,LPCOEF

    nolist
    include 'm96equ.a'          ;include the Ariel equates
    include 'jgequ.asm'         ;include my equates
    list
    include 'jgmacros.asm'      ;include my macros
    org    pl:

;****************************************************************************
;
; lphp      LOWPASS/HIGHPASS/BANDPASS FILTERING SUBROUTINE
;
; Gets data samples from input tapped delay lines and performs
; lowpass, highpass, and bandpass filtering.
;
; UPDATES:    LP0,HP0,BP0
; USES:       d0-d3
;             d7 - repeat counter
;             r3 - index to input data TDLs
;             n3 - copy of n0 or n6, note dual use of this register
;             m3 - copy of m0
;             r4 - index to lpfilter coefficients
;             r6 - copy of r2 to restore at end
;             n6 - offset to midpoint of filter
; READS:      r0,m0,n0
;
;****************************************************************************
lphp
    move   x:<LPMID,d0.l
    dec    d0.l
    neg    d0.l   x:LPLEN,d7.l
    dec    d7.l   d0.l,n6
    move          d0.l,n0
    move   #LPCOEF,r5
    move   r2,r6
    FIR1   0,y,5,x              ;filter IN0  with LPfilter
    move   d0.s,y:(r2)+n2            ;store lp result in LP0
    move   #HPCOEF,r5
```

```
	move	d0.s,d2.s
	FIR1	0,y,5,x		;filter IN0 with HPfilter	move	d0.s,y:(r2)+n2		;store
hp result in HP0
	move	y:(r0+n0),d1.s		;get original input
	fsub.s	d0,d1		;IN0 - HP0 = BP0'
	fsub.s	d2,d1		;BP0' - LP0 = BP0
	move	d1.s,y:(r2)+n2		;store result in BP0
	move	r6,r2
	rts
	endsec
```

```
;****************************************************************************
;
; FIR1        MACRO   data,datasp,coef,coefsp
;
;             d0 = h[n] * x[n]
;
; READS:      d7 is length of filter minus one
;             r\coef is pointer to h[0], not corrupted if m\coef=d7
;             m\coef is length of filter minus one
;             r\data is pointer to x[n], not corrupted if m\data=d7
;             m\data is length of filter minus one
;             \csp and \dsp must be X and Y or Y and X
;
; MODIFIES:   d0,d1,d4,d5 (result returned in d0)
;
;****************************************************************************

FIR1   MACRO   data,dsp,coef,csp
    fclr   d0
    fclr   d1                   \dsp:(r\data)-,d4.s \csp:(r\coef)+,d5.s
    rep    d7.l
    fmpy   d4,d5,d1  fadd.s d1,d0  \dsp:(r\data)-,d4.s \csp:(r\coef)+,d5.s
    fmpy   d4,d5,d1  fadd.s d1,d0
    fadd.s d1,d0
    ENDM

;****************************************************************************
;
; IIR1        MACRO   newdata,yout,bcoef,acoef,temp
;
;             y[n] = b x[n] * a y[n-1]
;
; UPDATES:    yout - input is y[n-1], returns y[n], both FP in dn.s
; READS:      newdata - x[n] is newest sample as FP in dn.s
;             bcoef - first order IIR b coefficient as FP in dn.s
;             acoef - first order IIR a coefficient as FP in dn.s
; MODIFIES:   temp
; oldout:     y[n-1] is previous output as FP value in dn.s, CORRUPTED
; result:     y[n] is FP value in dn.s
;
;****************************************************************************

IIR1   MACRO   newdata,yout,bcoef,acoef,temp
    fmpy.s  d\bcoef,d\newdata,d\temp
    fmpy.s  d\acoef,d\yout,d\yout
    fadd.s  d\temp,d\yout
```

```
    ENDM

;*****************************************************************************
;
; DCNULL      MACRO   i,data,sf,dcb,dca
;
; Converts integer data sample to FP format, scales, and DC nulls.
; Updates running DC value and stores results.
;
; UPDATES:    DCVALS+i - computed DC value of ith channel
;             data - newest data sample in dn; input as integer
;                  returned as scale FP value with DC nulled
; READS:      SCALES+i - scale factors for microphone corrections
;             i - channel number and index into DCVALS and SCALES
;             sf - scale factor for FP conversion in dn.s
;             dcb - first order IIR b coefficient as FP in dn.s
;             dca - first order IIR a coefficient as FP in dn.s
; MODIFIES:   d4, d5, d6
; CALLS:      IIR1
;
;
;*****************************************************************************
DCNULL  MACRO   i,data,sf,dcb,dca
    float.s d\data          x:DCVALS+i,d5.s ;convert to floating point
    fmpy.s  d\sf,d\data,d\data  x:SCALES+i,d4.s  ;scale A/D data to +/-1
    IIR1    data,5,dcb,dca,6            ;macro for first order IIR
    fsub.s  d6,d\data       d6.s,x:DCVALS+i ;d6 = DC value
    fmpy.s  d4,d\data,d\data            ;scale for mic correction
    ENDM

;*****************************************************************************
;           Macro for Division -- Calculates d0/d5--> d0
;
; taken from Daniel Welker's Master's Thesis
;
; NOTE:  Uses d0,d2,d3,d4,d5
;
;*****************************************************************************
DIV   MACRO
    fseedd  d5,d4
    fmpy.s  d5,d4,d5                    #2.0,d2.s
    fmpy    d0,d4,d0    fsub.s d5,d2        d2.s,d3.s
    fmpy.s  d5,d2,d5                    d2.s,d4.s
    fmpy    d0,d4,d0    fsub.s d5,d3
    fmpy.s  d0,d3,d0
    ENDM
```

```
%*************************************************************************
% fmf6.m
%
%*************************************************************************

% Memory locations on board 0
restart0_addr = '3';
restart0_space = 'X';
lplen_addr = 'B';
lplen_space = 'X';
lpmid_addr = 'C';
lpmid_space = 'X';
lpcoef_addr = '100800';
lpcoef_space = 'X';
hpcoef_addr = '101000';
hpcoef_space = 'X';
passflg_addr = '4';
passflg_space = 'X';
ind_addr = '108000';
ind_space = 'X';
r_addr = '20000000';
r_space = 'X';
Nmax_space = 'X';
Nmax_addr = '14';

% constants
FM = 1;
AM = 0.1;
FS = 16000;
GAIN = 1;
N = 252;     % LPLEN-1

% variables
K = 2*pi*FM/FS;

% Set up board, code will run but wait for restart flag
ch0 = ddeinit('m96serv','0');
initdsp(ch0,'fmf6');

% Design lowpass/highpass/bandpass filter
fsamp = 16000;
fc = 3150;
flo = fc/sqrt(2);
fhi = fc*sqrt(2);
ftrans = 100;


%lowpass
fparml = [0 2*flo-ftrans 2*flo+ftrans fsamp] / fsamp;
mparml = [1 1 0 0];
lplen = 253;            % Length must be odd for highpass design
lpmid = (lplen+1)/2;    % Compute midpoint
lpcoef = remez(lplen-1,fparml,mparml);
```

```
%highpass
fparmh = [0 2*fhi-ftrans 2*fhi+ftrans fsamp] / fsamp;
mparmh = [0 0 1 1];
hpcoef = remez(lplen-1,fparmh,mparmh);

% Frequency modulation variables
Nmax = ceil(FS/FM)          %(fs/fm) - to get one period
m = 1 - AM*sin(K*(1:Nmax));

sm(1) = m(1);
for i=2:Nmax
sm(i) = sm(i-1) + m(i);
end

ind = floor(sm);
r = rem(sm+1,ind+1);

huge = max(ind);
small = min(ind);
indsize = size(ind);
rsize = size(r);

% Download filter parameters
mat2dsp(ch0,lplen,lplen_addr,lplen_space,'ulong');
mat2dsp(ch0,lpmid,lpmid_addr,lpmid_space,'ulong');
mat2dsp(ch0,lpcoef,lpcoef_addr,lpcoef_space,'float');
mat2dsp(ch0,hpcoef,hpcoef_addr,hpcoef_space,'float');
mmat2dsp(ch0,ind,ind_addr,ind_space,'ulong');
mmat2dsp(ch0,r,r_addr,r_space,'float');
mat2dsp(ch0,Nmax,Nmax_addr,Nmax_space,'ulong');

% Start both boards by clearing restart flag
input('Press return to clear restart flag');
a=0;
mat2dsp(ch0,a,restart0_addr,restart0_space,'ulong');

temp1 = dsp2mat(ch0,1,'C','X','ulong');
temp2 = dsp2mat(ch0,1,'D','X','ulong');
temp3 = dsp2mat(ch0,1,'E','X','ulong');
temp4 = dsp2mat(ch0,1,'F','X','ulong');
temp5 = dsp2mat(ch0,1,'10','X','ulong');
temp6 = dsp2mat(ch0,1,'11','X','ulong');

indtesta = mdsp2mat(ch0,Nmax,ind_addr,ind_space,'ulong');
figure(2)
plot(indtesta)

nmaxtesta = mdsp2mat(ch0,1,Nmax_addr,Nmax_space,'ulong')

input('Press return to reset DSPs and close connection');
initdsp(ch0,'');
ddeterm(ch0);
```

References

[1] D.P. Egolf, "Review of the acoustic feedback literature from a control theory point of view," *The Vanderbilt Hearing-Aid Report,* Studebaker and Bess, Eds. Upper Darby, PA: Monographs in Contemporary Audiology, 1982, pp. 94-103.

[2] C.P. Boner, C.R. Boner, "Minimizing feedback in sound systems and room-ring modes with passive networks," *J. Acoust. Soc. Am.* 37, 1965, pp. 131-135.

[3] J.A. Maxwell, P.M. Zurek, "Reducing acoustic feedback in hearing aids," *IEEE Trans. on Speech and Audio Processing,* Vol. 3. No. 4, 1995, pp. 304-313.

[4] M.R. Schroeder, "Improvement of acoustic feedback stability in public address systems," *Proc. Third Int. Cong. Acoust.* Vol. 2, 1961, pp. 771-775.

[5] D.K. Bustamante, T.L. Worrall, and M.J. Williamson, "Measurement of adaptive suppression of acoustic feedback in hearing aids," *Proc. 1989 Int. Conf. Acoustics, Speech, Signal Processing,* Glasgow, U.K., 1989, pp. 2017-2020.

[6] A.M. Engebretson, M.P. O'Connell, and F. Gong, "An adaptive feedback equalization algorithm for the CID digital hearing aid," *Proc. IEEE Ann. Int. Conf. Eng. Med. and Biol. Soc.,* Vol. 12, 1990, pp. 2286-2287.

[7] J.M. Kates, "Feedback cancellation in hearing aids: Results from a computer simulation," *IEEE Trans. Signal Processing,* Vol. 39, pp. 553-562.

[8] G. Nishinomiya, "Improvement of acoustic feedback stability of public address system by warbling." *Proc. Sixth Int. Cong. Acoust.,* Vol. 3, 1968, pp. 93-96.