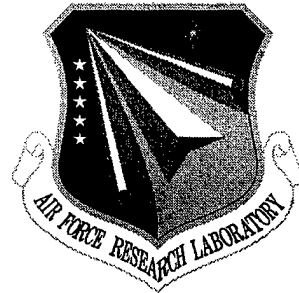


AFRL-IF-RS-TM-1999-6
In House Technical Memorandum
December 1999



A SIGNAL ENERGY DETECTION IMPLEMENTATION

Timothy M. Hughes

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

**AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK**

20000201 004

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

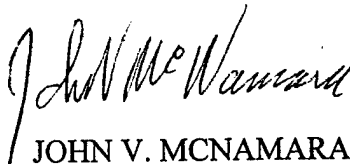
AFRL-IF-RS-TM-1999-6 has been reviewed and is approved for publication.

APPROVED:



GERALD C. NETERCOTT
Chief, Multi-Sensor Exploitation Branch
Info and Intel Exploitation Division
Information Directorate

FOR THE DIRECTOR:



JOHN V. MCNAMARA, Tech Advisor
Info and Intel Exploitation Division
Information Directorate

If your address has changed or if you wish to be removed from the Air Force Research Laboratory Rome Research Site mailing list, or if the addressee is no longer employed by your organization, please notify AFRL/IFEA, 32 Brooks Road, Rome, NY 13441-4114. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document require that it be returned.

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.				
1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE December 1999		3. REPORT TYPE AND DATES COVERED In House, May 99 - August 99
4. TITLE AND SUBTITLE A SIGNAL ENERGY DETECTION IMPLEMENTATION			5. FUNDING NUMBERS PE - 62702F PR - 4594 TA - 15 WU- 2F	
6. AUTHOR(S) Timothy M. Hughes*				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Air Force Research Laboratory/IFEC 32 Brooks Road Rome, NY 13441-4114			8. PERFORMING ORGANIZATION REPORT NUMBER AFRL-IF-RS-TM-1999-6	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Research Laboratory/IFEC 32 Brooks Road Rome, NY 13441-4114			10. SPONSORING/MONITORING AGENCY REPORT NUMBER AFRL-IF-RS-TM-1999-6	
11. SUPPLEMENTARY NOTES AFRL Project Engineer: Andrew J. Noga/IFEC/(315)330-2270. *Author performed this research while employed with AFRL under the Summer Engineering Aide Program.				
12a. DISTRIBUTION AVAILABILITY STATEMENT APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.			12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words) The Adjustable Bandwidth Concept (ABC) signal energy detector has been implemented in the Mathwork's MATLAB environment, with a graphical user interface. The ABC algorithm, US patent#5,257,211 can be applied to a variety of digitized signals, including both audio and basebanded RF communications signals. The implementation allows for easy re-configuration of the algorithm through graphical parameter control, which facilitates signal energy detection tests.				
14. SUBJECT TERMS Adjustable Bandwidth Concept, Signal Energy Detection, Graphical User Interface, GUI			15. NUMBER OF PAGES 36	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT UL	

Table of Contents

<u>Paragraph</u>	<u>Title</u>	<u>Page</u>
1.0	Introduction	1
2.0	Background	1
2.1	ABC Block Diagram	2
2.2	A Single Channel Signal Energy Detector	3
2.3	Wavelet Comparison	3
2.4	Existing Implementation	3
3.0	Overview of Enhanced Implementation	4
3.1	Parameter Adjustment	4
3.2	Directory Structure	6
3.3	Implementing Graphical User Interface	7
4.0	Overview of Graphical User Interface	7
4.1	Running Graphical User Interface	7
4.1.1	Last Stage Setup	9
4.1.2	First Stage Setup	10
4.1.3	Intermediate Stage Setup	11
4.2	Saving Parameters	12
4.3	Loading Parameters	12
4.4	Running ABC Algorithm	13
5.0	Conclusions	13
5.1	Future Developments	14
5.1.1	Delay Adjustment	14
5.1.2	Low Pass Filter Adjustment	15
5.1.3	Display Scaling	15
5.1.4	File Input	16
5.1.5	Detection	16
5.2	Acknowledgements	16
Appendix A	Sample Output	A-1

1.0 Introduction

This report summarizes the work performed under the 1999 Summer Engineering program in the area of signal detection and parameter estimation. The task was to develop and implement a MATLAB-based automated signal energy detection capability, based on the Adjustable Bandwidth Concept (ABC) Signal Energy Detector, U.S. Patent #5,257,211. The ABC algorithm can be applied to a variety of digitized signals, including both audio and basebanded RF communication signals.

The resulting implementation has a user-friendly graphical interface, which is designed to allow for easy alteration of device parameters, thereby facilitating tests of the algorithm. A synthetic multi-component signal was generated for initial test and evaluation of the ABC detector. The device was also tested and evaluated using a variety of audio .wav files. The test results demonstrate the ability of the ABC algorithm to perform the signal energy detection function.

2.0 Background

The purpose of this task is to further implement and test the existing ABC algorithm. An existing implementation was tested with a synthetic signal and fixed parameters. The algorithm appeared to be processing the signal as expected, but multiple inputs and parameters were necessary to completely test the algorithm's capabilities. Changes in the parameters were possible through the code, but a graphical user interface was preferred. A directory structure was also desired in order to organize and simplify the use of multiple files and functions. It was also desired to run this enhanced implementation on various input signals, and the .wav file format was chosen to facilitate this ability.

2.1 ABC Block Diagram

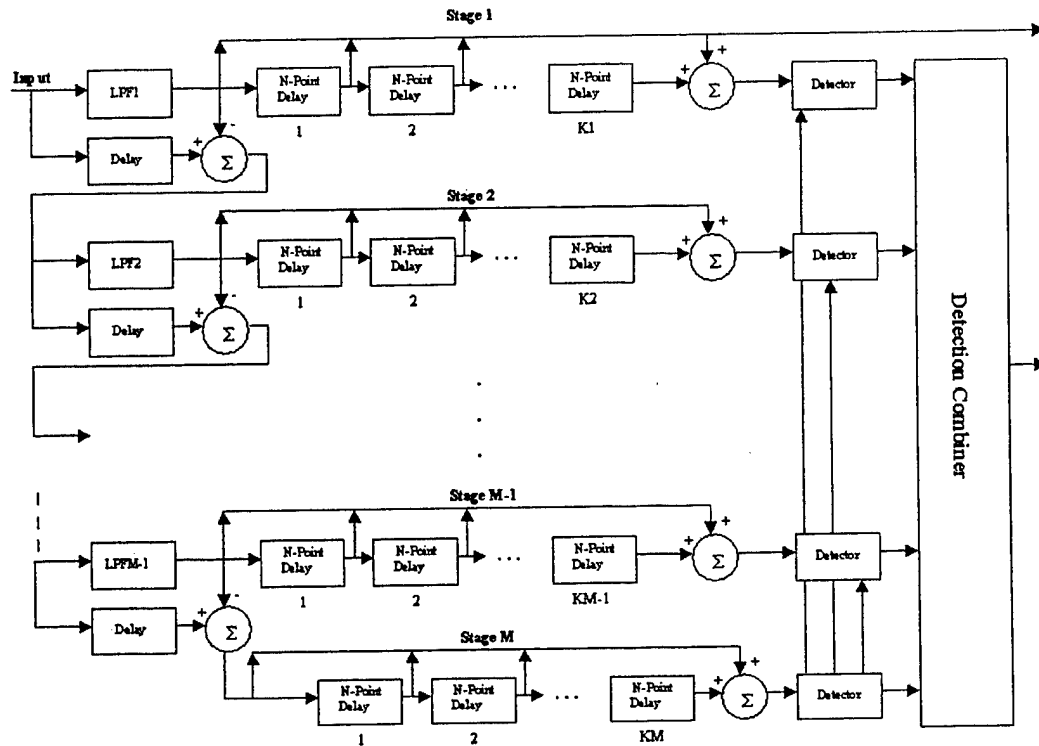


Figure 2.1-1. ABC Block Diagram

The above diagram (Figure 2.1-1) is the ABC algorithm block diagram. It shows the input to the system at the upper left corner of the diagram. The input consists of a series of N-point frequency bins, as from an FFT, in log scaled magnitude. This input passes through the first stage, and then continues on to be processed by the rest of the M stages. Each stage from one to M-1 contains the same components. These components are a lowpass filter, a filter delay, a set of N-point delays, two summers, and a detector. Each stage's components are independent from another stage's components. The last stage differs from the others because it does not contain a filter. This stage processes the

residue from the previous stages. The output from each stage is then sent to a detector, where the separated components are processed and analyzed.

2.2 A Single Channel Signal Energy Detector

The ABC algorithm allows for averaging to be varied dependent on the situation. The algorithm allows for wide-band signals to be averaged more over frequency and less over time, while narrowband signals can be averaged less over frequency and more over time. This unique averaging process gives a better separation and detection of the signals. For further details, reference is made to the patent itself.

2.3 Wavelet Comparison

The output of the ABC algorithm can be likened to that of a wavelet-based time-frequency transformation, but the processes are different in each case. In the ABC process the time-frequency representation is log-scaled in amplitude, and is treated as if it were a time sequence. Thus, time-domain filter processes are applied to the frequency-domain input sequence, to effect averaging over both time and frequency. This input sequence is composed of a time-ordered series of N-point power spectral density (PSD) segments, as for example from a sliding FFT.

2.4 Existing Implementation

The existing MATLAB implementation of the ABC algorithm was developed in March of 1998, by Dr. Andrew Noga. This implementation consisted of fixed parameters and system setup. The system was fixed as a three-stage detection process with static

filter coefficients and numbers of N-point delays. The signal being processed was synthetically generated and consisted of three example tones, a wideband signal, a mediumband signal, and random noise. These were then combined to form one signal to be processed by the ABC algorithm. After the algorithm completed processing of the signal, the output was a collection of plots and images. The output consisted of a time vs. frequency image of the input, plots of each stage's average, and time vs. frequency images of each stage's output. This implementation appeared to be working with the given signal and setup, and facilitated further development.

3.0 Overview of Enhanced Implementation

The core algorithm for the enhanced implementation of the ABC process does not vary from the existing implementation. All of the calculations and procedures remain unchanged. The additional enhancements that were made to the existing implementation were parameter adjustment, the use of a directory structure, and a graphical user interface.

3.1 Parameter Adjustment

The first enhancement to the existing implementation was to allow for parameters and settings to be adjusted. The algorithm was only able to run one signal on fixed settings in the existing implementation. Therefore, being able to adjust the settings used by the algorithm was a major enhancement. These included the number of stages that the algorithm would use, the number of N-point delays that each stage would use, and the filter coefficients for each stage.

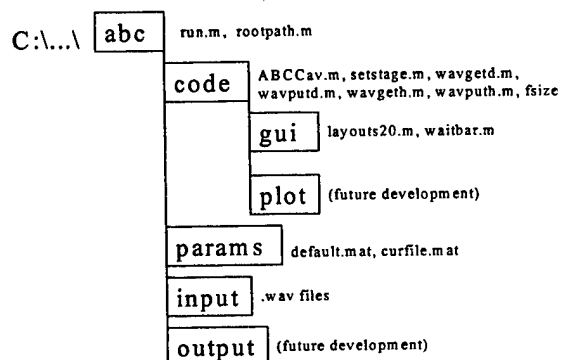
In the existing implementation, sections of MATLAB code were run separately for each of the three stages. The enhanced implementation was able to change the number of stages by using a For- loop to run the algorithm for as many stages as were needed. This not only was more efficient, but also made the code a great deal more compact.

In the existing implementation, the code used three fixed values for the number of N-point delays for each stage. The number of N-point delays contained in a stage, determines the amount of time-averaging performed in that stage. For example, one N-point delay results in the averaging of two consecutive PSD segments. Two N-point delays will result in the average of three PSD segments, and so on. The boundary values from the existing implementation were used as the basis for defaulting the number of N-point delays per stage in the enhanced implementation. Therefore, the number of stages determined how many equal sized segments the difference between the boundary values would be divided into. For example, if the boundary values are 1 N-point delay for stage 1 and 10 for stage 4, then the default number of delays for stages one through four would be 1, 4, 7, and 10, respectively.

In the existing implementation, the algorithm also used fixed values for the filter coefficients for each stage. A similar approach was used to generate default filter coefficient sets as was used to generate default numbers of N-point delays. Lower stages are assigned longer filter delays for more effective low-pass filtering, up to stage M, which does not employ a filter and therefore has a delay of zero. For simplicity, the current enhancement has used equal-valued filter coefficients, resulting in moving averages of varying lengths occurring at each stage input.

3.2 Directory Structure

The second enhancement to the existing implementation was to develop a directory structure. This structure has two principle objectives. The first is to organize the contents and accessories to the ABC algorithm. As a result of the enhancements to the existing implementation, more source code files were generated. They needed to be organized in a way that would make sense and ease code maintenance. The directory was made with folders and subfolders, each containing certain files. With relevant names and order, files can be found using the directory. The second objective of the directory structure was to ease the running of the ABC algorithm. This structure stores all relevant variables together. For example, when the ABC algorithm starts to run and calls a .mat file to get variables for the parameters and settings, it goes to folder “params” which contains all of the saved and default settings. Having similar files logically grouped together eases code maintenance and allows for further enhancements. A diagram of the directory structure used for the enhanced implementation is shown below.



3.3 Implementing Graphical User Interface

The third enhancement to the existing implementation was to incorporate a graphical user interface (GUI). This GUI has two principle objectives. The first is to make the algorithm a user-friendly program. Using this GUI makes the algorithm much easier to run. Also, the visualization of the block diagram and active controls make the system easier to comprehend. The second objective of the GUI is to utilize the enhanced implementation's flexibility. With the ability to change parameters and settings for the algorithm, the GUI organizes, displays, and accesses all of the variables. This is a very powerful enhancement in running the ABC algorithm.

4.0 Overview of Graphical User Interface (GUI)

The GUI is very important in running the ABC algorithm. This is the base for the ABC algorithm to run off of in the enhanced implementation. With the GUI, all settings and parameters are displayed, and any or all can be changed and/or saved. Once the settings are to the user's liking, data can be processed through the configured ABC detection process. The graphical user interface utilizes several functions. These include `rootpath.m`, `run.m`, `layouts20.m`, and `waitbar.m`.

4.1 Running the Graphical User Interface

To start the process of the GUI and the ABC algorithm, change the directory at the MATLAB command line to `C:\...\abc` (type `cd C:\...\abc`), then type `run`. (The '...' represents any parent directories that may exist.) This will bring up the last stage of an

ABC system. The stage number is labeled on the screen in bold font. The parameters and settings of this system will be as they were when the previous program was closed.

The GUI runs from one function, layouts20.m. This function loads the parameters for the desired system, and then loads the information on the input signal. The function then creates all of the menus, buttons, and properties shared by each stage. At this point, the function continues only with portions of the code pertaining to the given stage. The background picture, created in Microsoft PowerPoint, is displayed accordingly, along with any additional buttons or menus needed for the given stage. Each time a change is made to the system while the program is running, the change is saved and this function calls itself (referred to as a “Callback”). The function runs again, this time with whatever changes have been made to the previous system. Thus, GUI effectively controls the parameters associated with the input signal and the ABC device such that changes can easily be made prior to any processing session.

4.1.1 Last Stage Layout

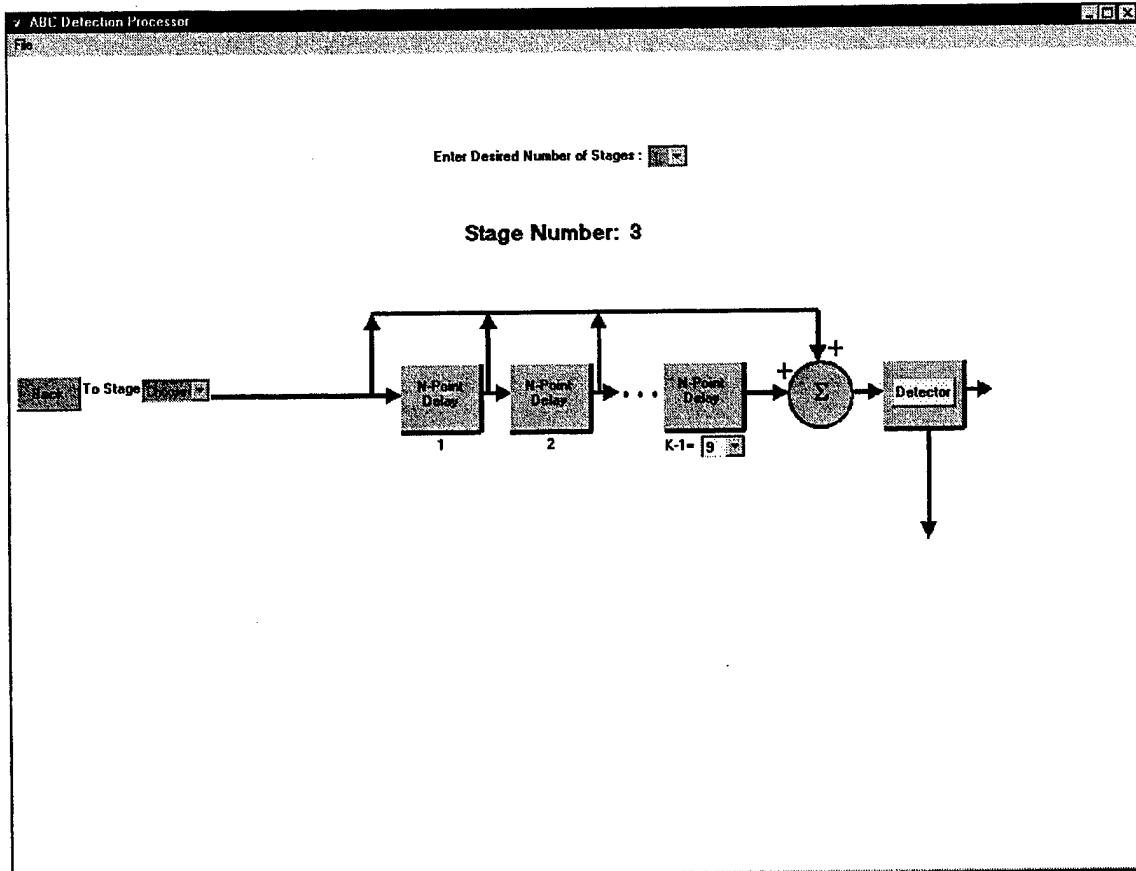


Figure 4.1.1-1. Last Stage Layout.

At the top of the screen of the last stage, there is a pull down menu that allows for the changing of the number of stages used by the current system (See Figure 4.1.1-1). If a number is chosen from this menu, the GUI changes the current number of stages to the selected number of stages. This is now stored as the default, and parameters are set to default values. Choosing a number from this menu will also bring you to the first stage of this new system, in preparation for input file selection.

4.1.2 First Stage Layout

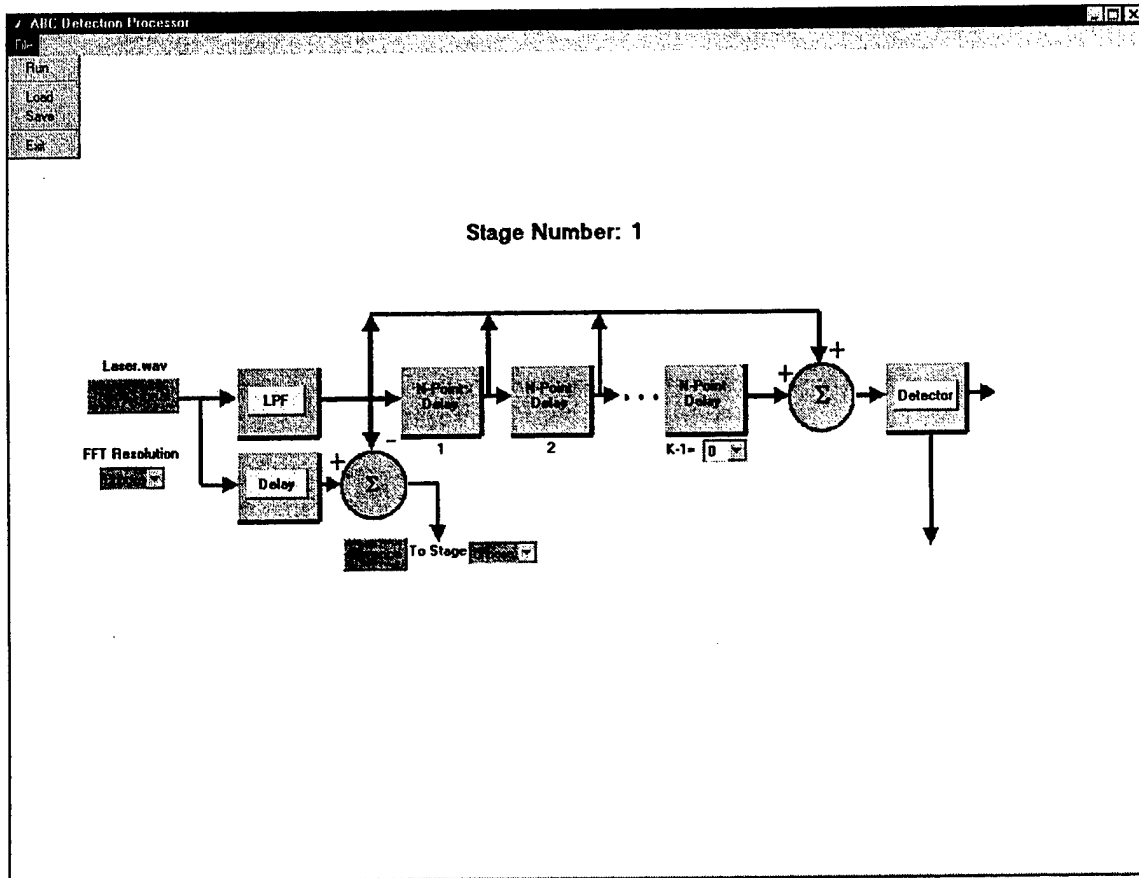


Figure 4.1.2-1. First Stage Layout

At the first stage, there are some different controls (See Figure 4.1.2-1). The button labeled "Input Signal" has a file name above it. This is the current .wav file. If a different file is desired for the ABC detection process, press the "Input Signal" button. Highlight the file desired and click on open. Underneath this button, there is a pull down menu labeled "FFT Resolution". Choose a number from this menu to be the frequency resolution in the processing and graphs of the input and output from the ABC detection process. Also, as shown above, the "File" menu is at the top of each stage screen. This menu includes commands described in Paragraph 4.2, 4.3, and 4.4. Underneath the stage

layout is a pull down menu labeled “Choose”. If a different stage is desired, choose the according number from this menu. To advance to that stage, press the “Advance” button.

4.1.3 Intermediate Stage Layout

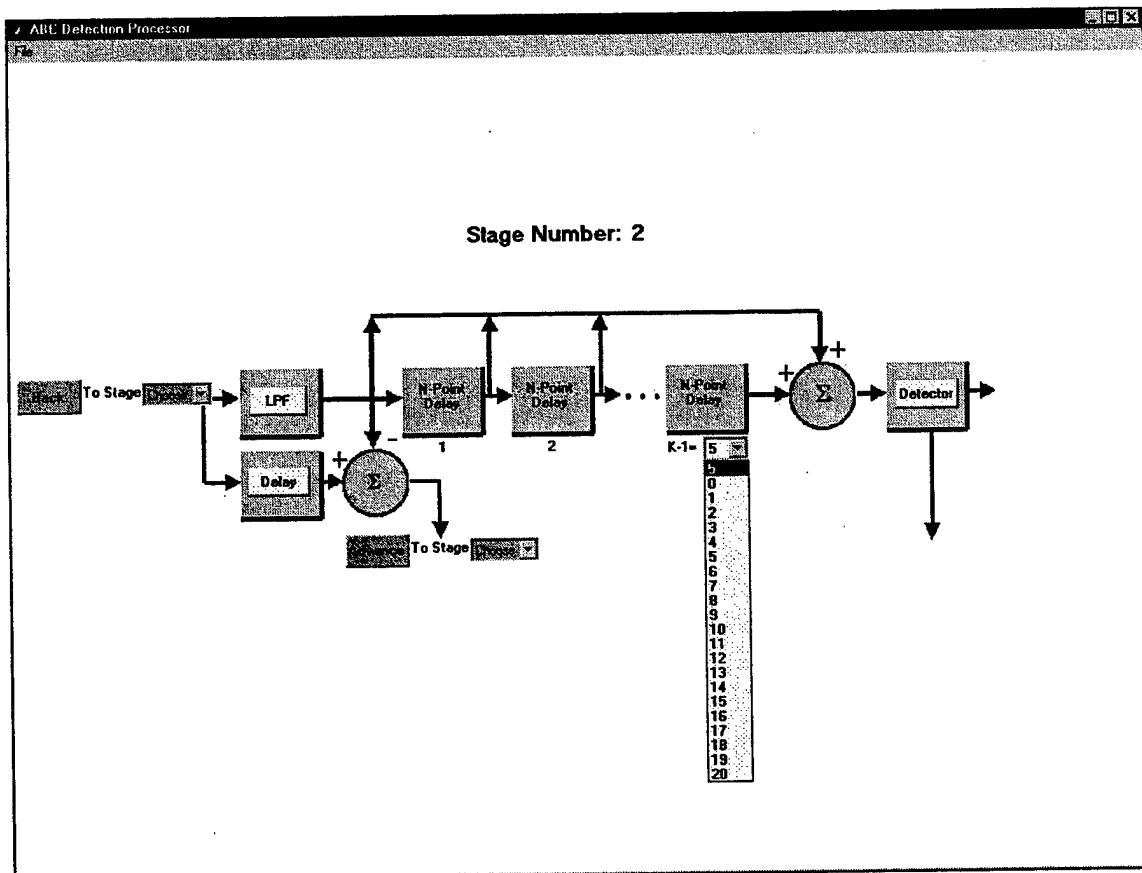


Figure 4.1.3-1. Intermediate Stage Layout.

The intermediate stage layout contains two similar functioning objects (See Figure 4.1.3-1). Just as in the first stage layout, there is a pull down menu with an “Advance” button preceding it. This works in the same manner as described in the preceding section. The intermediate stage layout also has another pull down menu with a

“Back” button preceding it. This works in a similar manner as the “Advance” button, but instead brings up a previous stage layout.

Also included in the intermediate stage layout is a pull down menu underneath the last box labeled “N-Point Delay”. The number displayed in this box is the number of delays for the current stage. Selecting another number from this menu will change the number of N-point delay segments for this stage. This pull down menu is included in each stage layout.

4.2 Saving Parameters

When the system is set in a way that is useful to the user for a specific signal, the parameters and settings of the current system can be saved. This includes the number of stages and the number of delays for each stage.

To save a system, open the “File” menu. Then click on “Save”. This will open another window that allows you to choose a path and name to save the system as. Once the desired path is displayed, type the name the system will be saved as. Then click “OK”. This system has been saved and is still the current setup for the program.

4.3 Loading Parameters

When a different system setup than the current is desired, a previously saved system may be loaded. Any system that has been saved previously using this program can be loaded to be the current system for the ABC detection process.

To load a system, open the “File” menu. Then click on “Load”. This will open another window that allows you to choose a path and name for the program to load.

Once the desired path is displayed, choose a file to load. Then click “Open”. The system has now been loaded. This is now the current setup for the program.

4.4 Running ABC Algorithm

When a system setup is desired to be run as the configured ABC algorithm, it can be done through the GUI. The only system that can be run is the current system displayed by the GUI.

To run a system through the ABC algorithm, open the “File” menu. Then click “Run”. This will run the current system and display the input and output images. While the signal is being processed by the algorithm, a display will show the progress and percentage completed. Once this figure displays completion, the results will be displayed. The first image is the input signal. After this, the output from each stage is displayed. These windows must be closed manually after viewing is complete.

5.0 Conclusions

The enhanced ABC algorithm and graphical user interface utilize the basis of the algorithm’s calculation capabilities to make a user-friendly program with both good flexibility and useful display of results. To date, the program is functional and efficient. In the process of getting the program to this point, useful knowledge and skills were gained. The two main areas in which the most information was gained by the author were signal processing and MATLAB usage.

Working with the ABC algorithm proved to be a very good learning experience. In researching other methods of computation and display, a broad range of knowledge

was gained in the field of signal processing. This knowledge allowed for the optimal performance of the enhanced algorithm and its graphical user interface.

The commands and processes utilized by the ABC algorithm generated for the author an in depth knowledge base for MATLAB. One example of this knowledge is the introduction to Singleton dimensions. In the process of setting up matrices, matching sizes became a problem. This led to the discovery that matrices were set up by default to be three dimensional, with single dimensionality where applicable. For example, a one-by-one matrix is represented as being a one-by-one-by-one. This and other knowledge led to skills that were able to aid in the layout and implementation of the algorithm and its graphical user interface. MATLAB resources and capabilities proved to be important tools during this research.

5.1 Future Developments

Although the ABC algorithm and its accompanying graphical user interface are capable of performing desired tasks on signals, there are many future developments that could greatly enhance the overall program. These developments include the selection of N-point delays for each stage, the changing of filter coefficients for each stage, the scaling of the ABC algorithm output images, detector controls, and varying file input.

5.1.1 Delay Value Adjustment

The graphical user interface for the ABC detection process contains the capability to change the number of N-point delay segments for each stage. Although these changes can be saved and are used as the new value in that system setup, the estimation and

effects of the resulting time averaging were not looked into. For now, the adjustments can be made without a known effect. In the future, these parameters will be studied, and effects on the system will be interpreted. This will allow full implementation of this capability.

5.1.2 Low Pass Filter Adjustment

The graphical user interface for the ABC detection process also contains the capacity to change the low pass filter setup for each stage. At this date, there is a button on the system setup that is labeled "LPF", but it has no function. Research went into how an implementation might be used, but its function has not yet been put into action. The filter coefficients were derived from the original three-stage setup and generated signal. These may or may not be suitable values for all cases. For now, the filter values are representative of moving averages of shorter lengths as the stage number increases. More general filter implementation could possibly improve the performance and output of the ABC algorithm.

5.1.3 Output Image Scaling

After the ABC algorithm runs the given system, its output consists of time vs. frequency images of both the input and output for each stage. The scaling for the intensity is preset. It is thought that if the highest signal value is set to be the most intense color, while the rest is scaled accordingly, images could be more informative. On some of the outputs, a tone is visible, but might be more easily distinguished if a scaling was done. This scaling, though, could also potentially cause problems with making

background noise more predominant, thus making the tone harder to distinguish. For now, scaling is statically fixed. Implementation of scaling may or may not prove useful.

5.1.4 File Input

The ABC detection process at the moment is limited to .wav file input. Future implementation could include capabilities to input, process, and output various file types. This implementation would eliminate the need to convert data to a .wav file format.

5.1.5 Detection

The graphical user interface for the ABC detection process also contains a button labeled "Detector". The implementation may include a detection process in the future. This will take the information and capabilities gained to date and use them to generate an automatic signal locator. This detection process can then lead to parameter estimation and concise reporting methods, rather than relying on visual interpretation of the images.

5.2 Acknowledgements

The enhanced implementation of the Adjustable Bandwidth Concept (ABC) algorithm could not have been possible without the guidance of Dr. Andrew Noga. His ideas and suggestions were many times the basis for the coding and functions used in the enhanced ABC algorithm. His guidance also allowed the understanding and proficiency of MATLAB needed to use its capabilities and resources to accomplish the given task. Gratefulness is also due for the opportunity to participate in the Summer Engineering Aide Program at the Rome Research Site.

Appendix A:
Sample Output

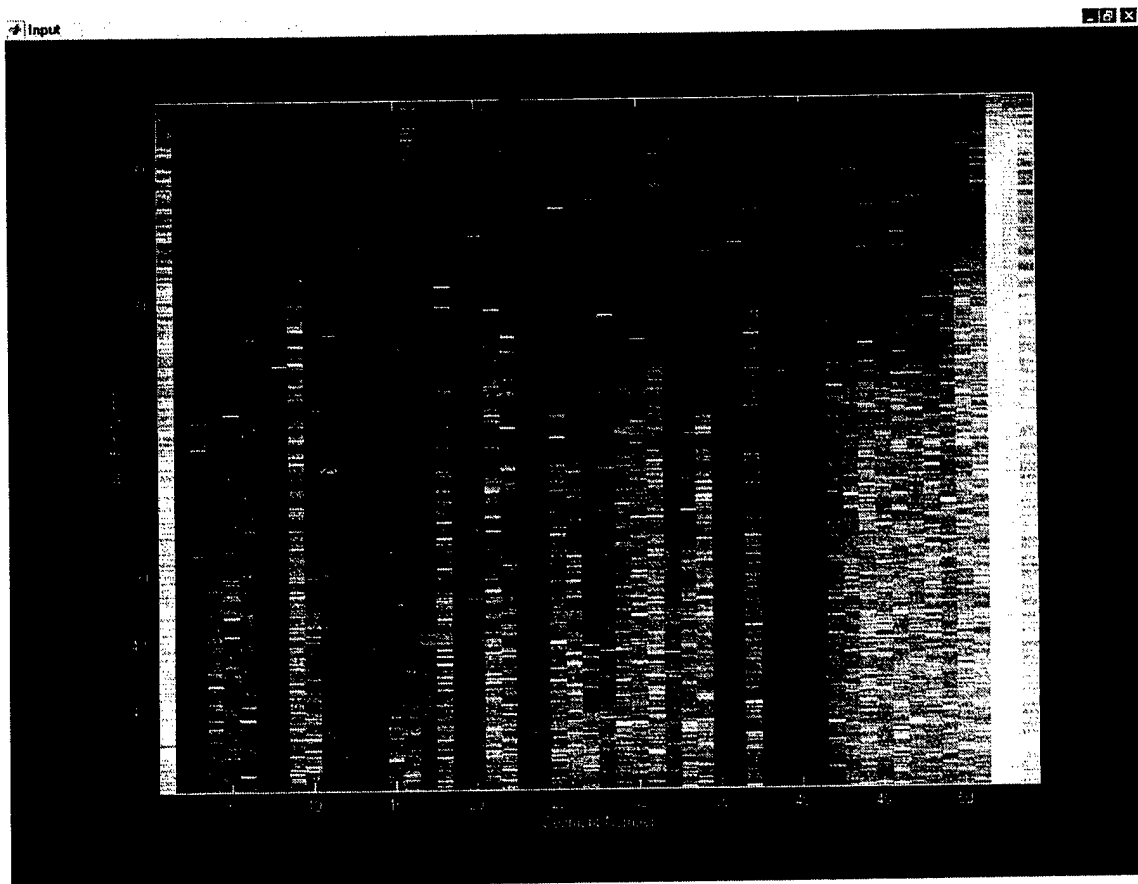


Figure A-1. Input from speech signal.

The above image (Figure A-1) is the input from a file that consisted of speech. This signal is eight-bit mono with a sampling rate of 22.050 kHz. This image is representative of the input to an example three-stage ABC configuration, with default numbers of N-point delays, and a frequency resolution of 512. (Note that this and all subsequent images show stronger components as darker pixels, and weaker components as lighter pixels.) This image shows the signal containing large amounts of low frequency content, and some high frequency content as well. The input image proves useful because many times it shows where possible components can be found. These possibilities may be confirmed by outputs from each stage.

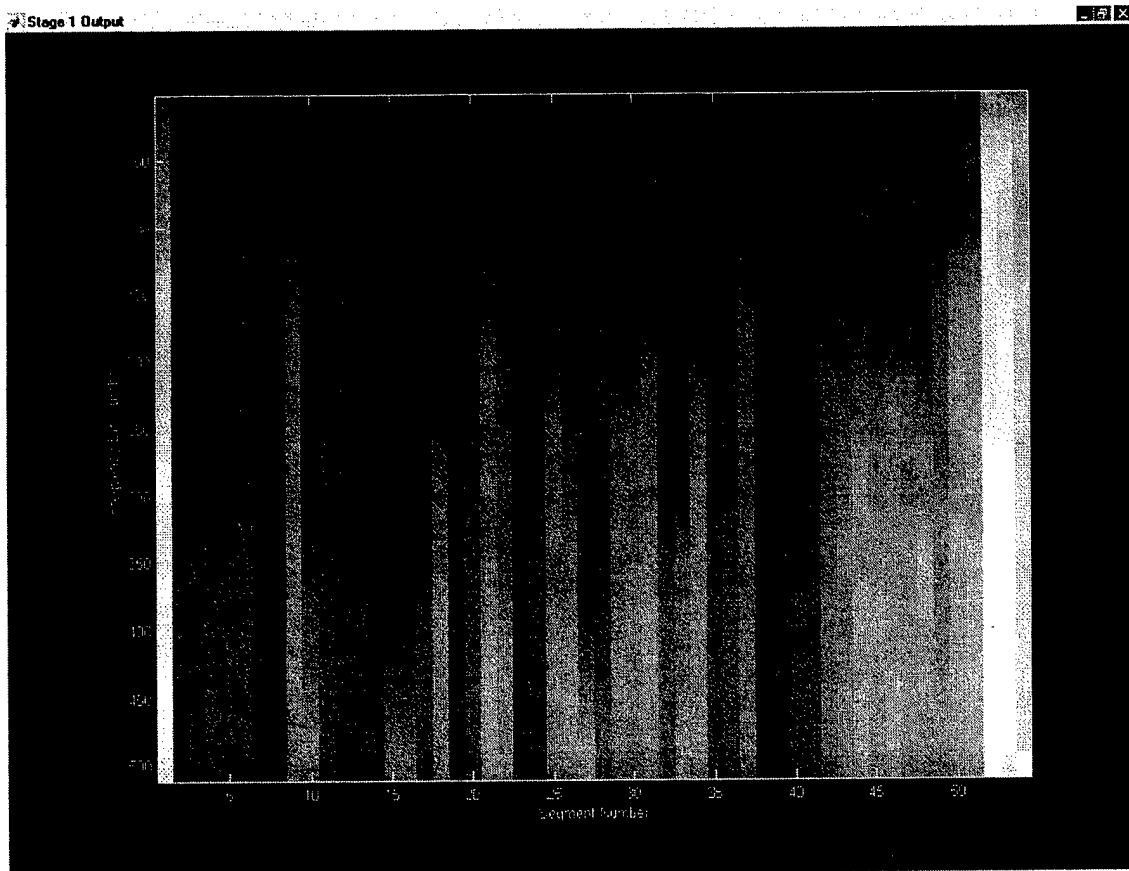


Figure A-2. First stage output from speech signal.

The above image (Figure A-2) is the first stage output from the speech signal input shown in Figure A-1. This output is from the same 3-stage system previously referred to. This image shows “wide-band” components of the speech signal. They are apparent throughout most of the frequencies displayed, but predominantly at low frequencies.

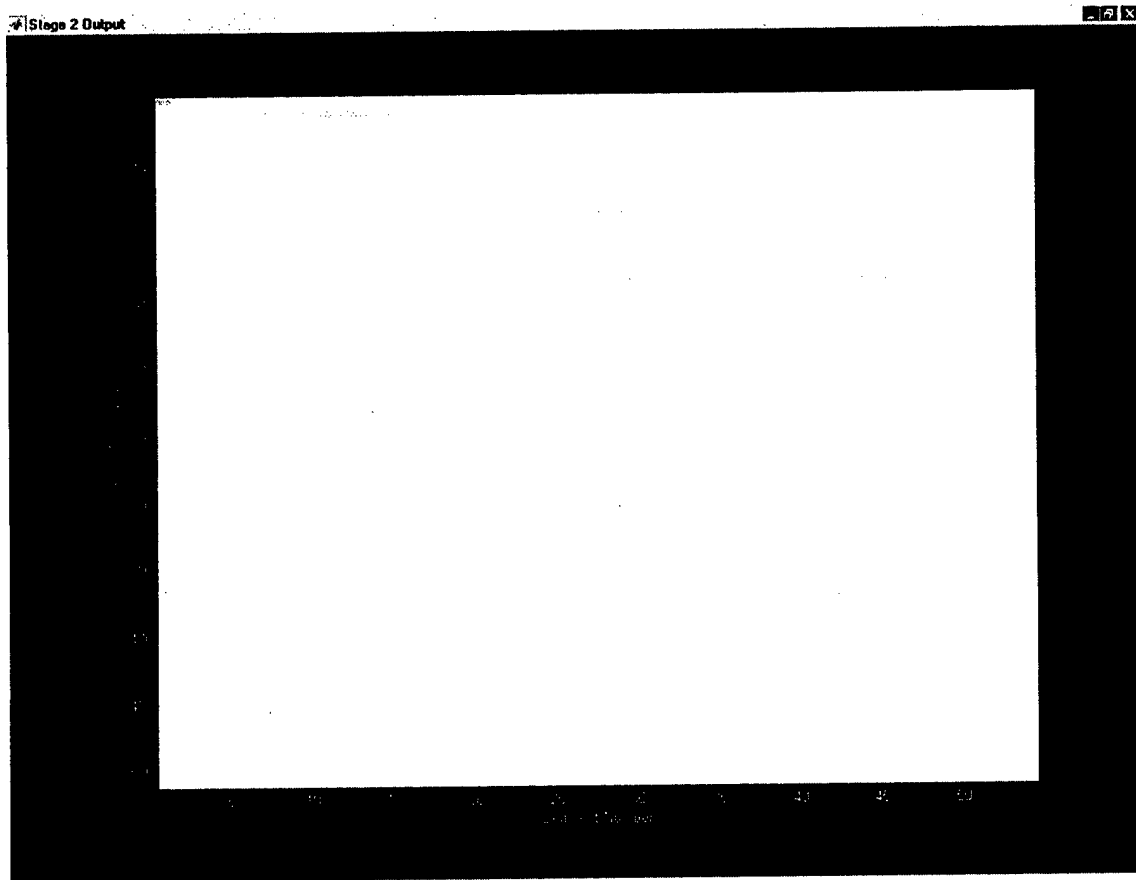


Figure A-3. Second stage output from speech signal.

The above image (Figure A-3) is the second stage output from the speech signal input shown in Figure A-1. This output is from the same system as used previously. This image shows medium-band components of the speech signal. They are not very apparent at any location in time. If anywhere, medium-band components are located about halfway through the time segments at medium to low frequencies.

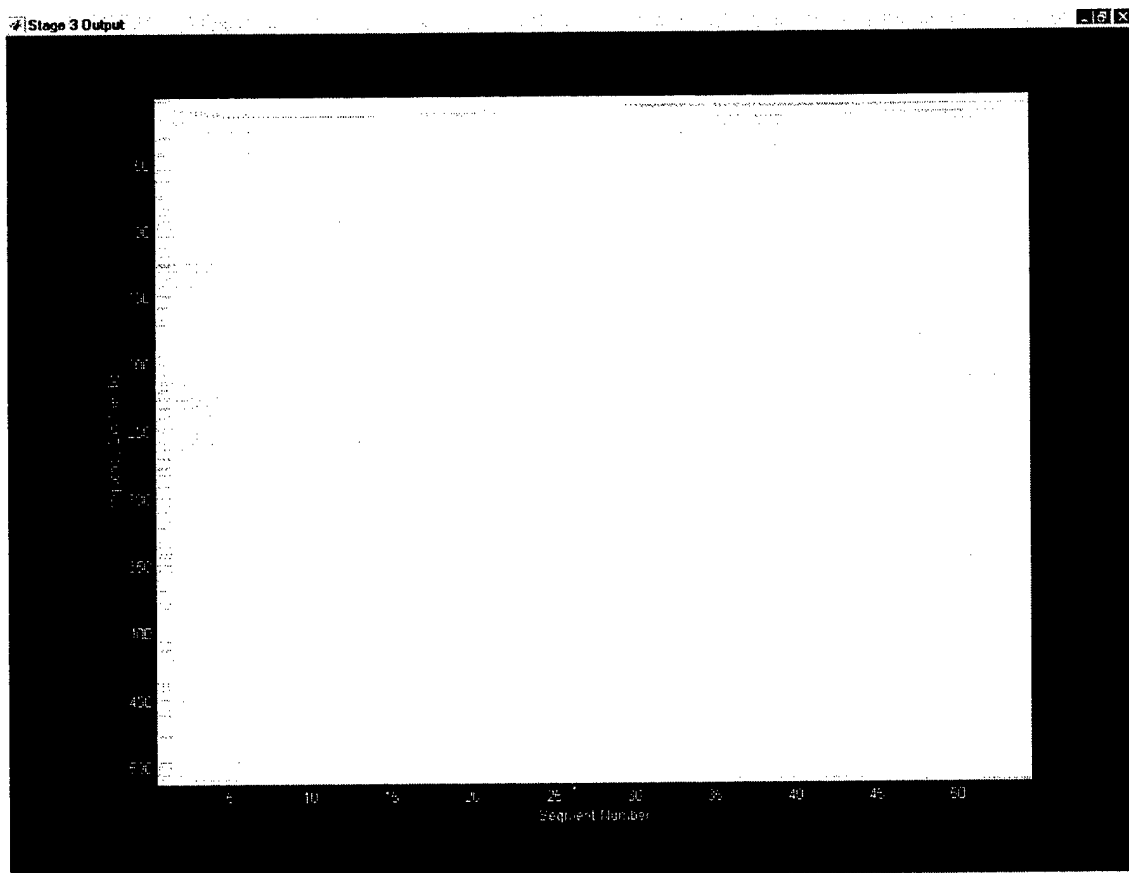


Figure A-4. Third stage output from speech signal.

The above image (Figure A-4) is the third stage output from the speech signal input in Figure A-1. The output is from the same system as used in the speech signal input. In the last stage, narrow-band tones are apparent. In this image, a strong narrow-band tone is not detected. If anywhere, weak narrow-band tones are apparent at low frequencies.

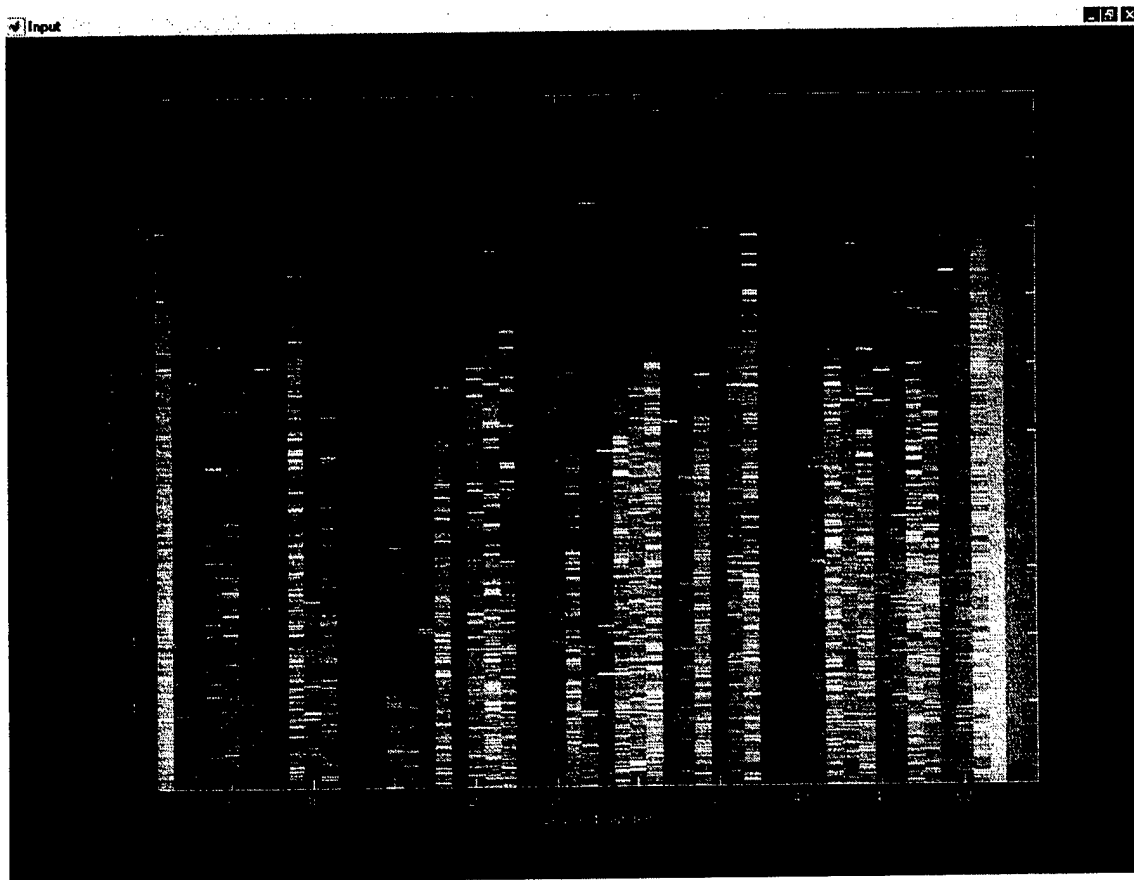


Figure A-5. Input from speech signal altered with a tone.

The above image (Figure A-5) is the input from a speech signal similar to the first example, but altered with a tone. This signal was 16-bit mono with a sampling rate of 22.050 kHz. This image was generated using the same frequency resolution, settings, and parameters as in Figure A-1, namely a three-stage setup with default delay values and frequency resolution of 512.

The speech signal appears the same as shown in Figure A-1. A tone was added to this signal, which is evident from this image. This tone appears as the dark line across every segment number.

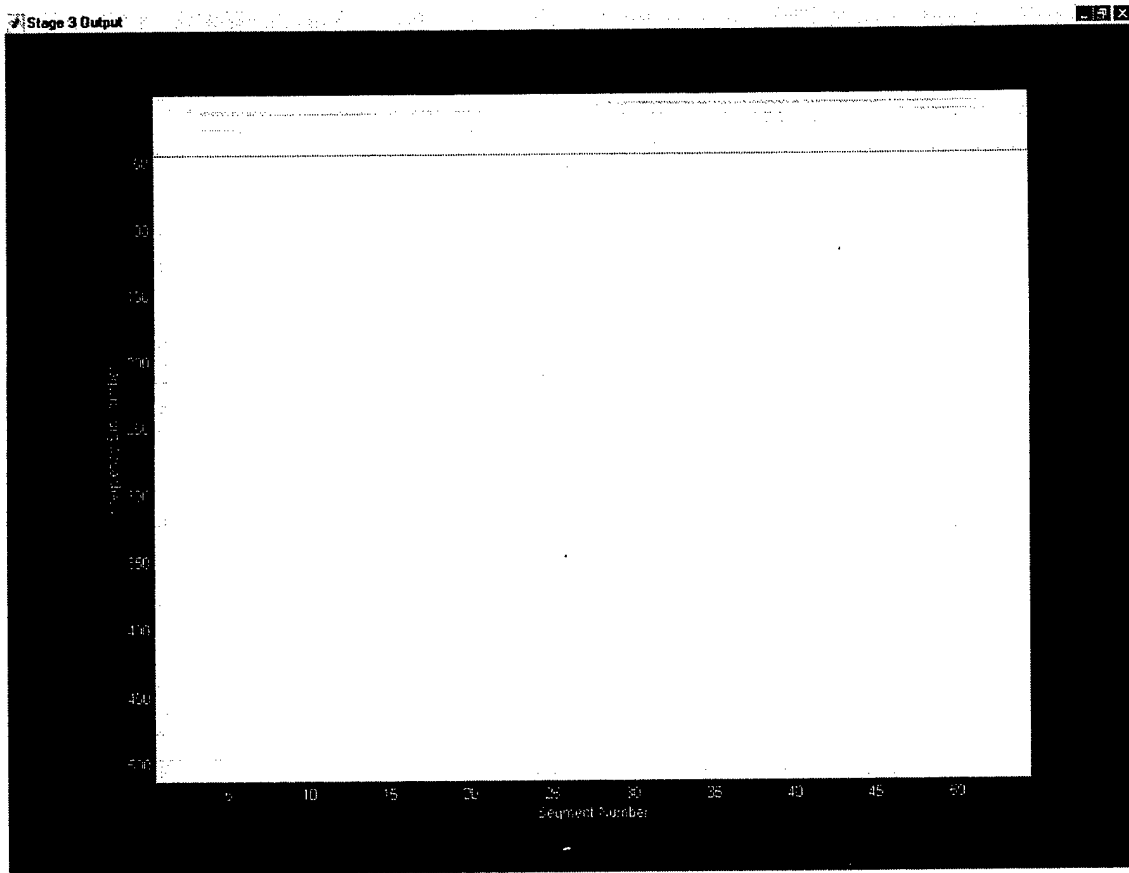


Figure A-6. Third stage output from speech signal altered with a tone.

The above image (Figure A-6) is the third stage output from the previously used speech signal altered with a tone. The settings and parameters of the system remain unchanged from Figure A-5.

From this image, the tone is clearly shown. The wide and medium-band components have been filtered out, leaving only the narrow-band tone to be displayed. This third stage output image can be viewed relative to the third stage output image of the original speech signal, and comparisons can be made. The tone appearing in the above image was not present in the original signal. Note that this tone could be easily detected through a threshold-based detection process.

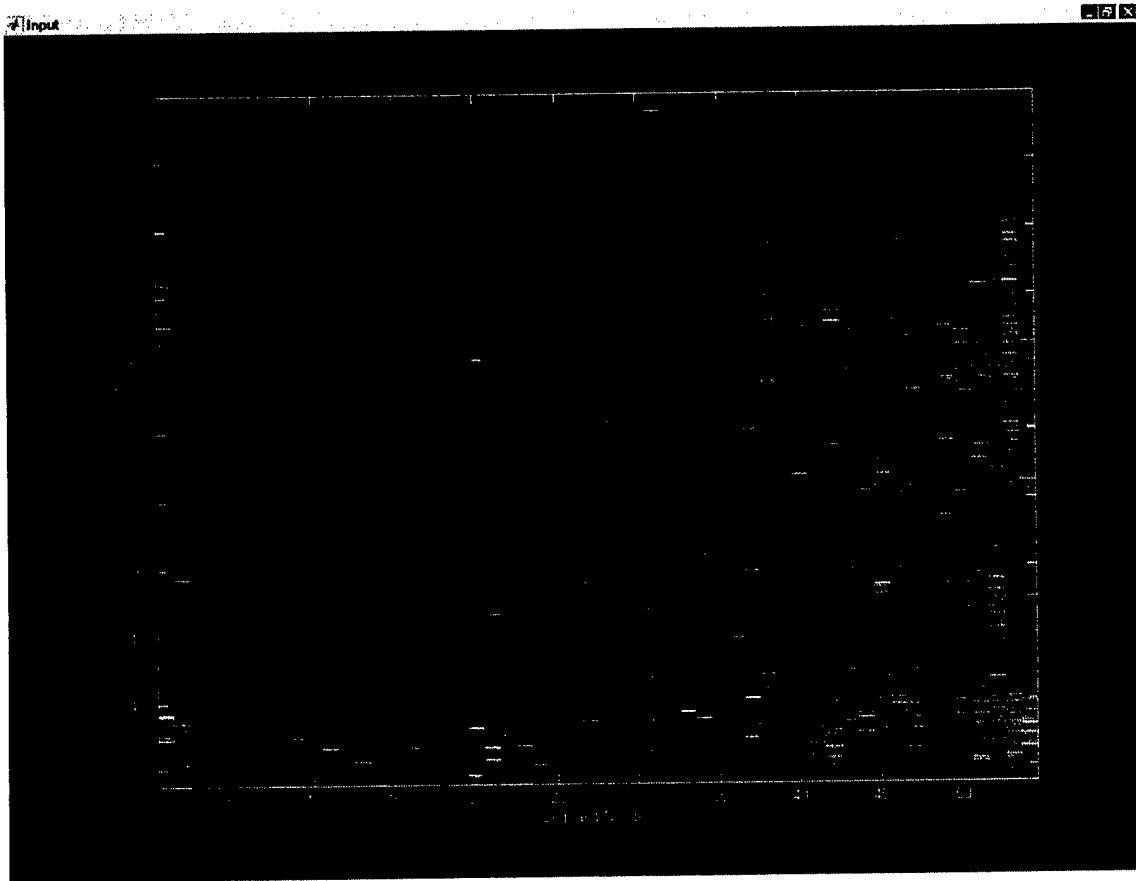


Figure A-7. Input from speech signal altered with a tone and converted from 16 to 8-bit.

The above image (Figure A-7) shows the input image of a speech signal. The speech signal that was altered with a tone was converted from sixteen-bit to eight-bit, producing this signal. A quantization effect occurred, resulting in the additional tones apparent in the image.

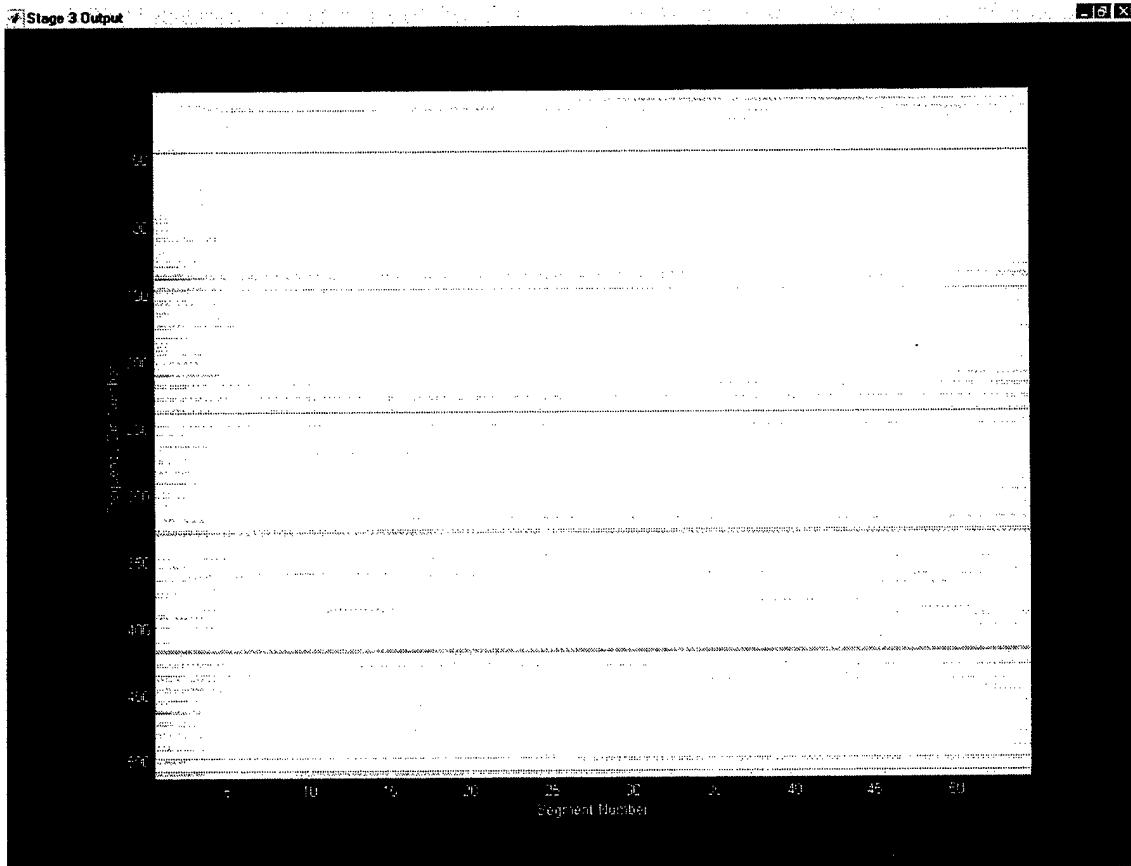


Figure A-8. Third stage image from speech signal altered with a tone.

The above image (Figure A-8) is the third stage output from the previously used speech signal altered with a tone that was converted from sixteen to eight-bit. Both the tone and quantization noise are clearly visible in this image.

**MISSION
OF
AFRL/INFORMATION DIRECTORATE (IF)**

The advancement and application of information systems science and technology for aerospace command and control and its transition to air, space, and ground systems to meet customer needs in the areas of Global Awareness, Dynamic Planning and Execution, and Global Information Exchange is the focus of this AFRL organization. The directorate's areas of investigation include a broad spectrum of information and fusion, communication, collaborative environment and modeling and simulation, defensive information warfare, and intelligent information systems technologies.