HIGHER-ORDERED SPECTRAL ANALYSIS OF WHALE SONGS

Swati Savkar
Anderson High School
Austin, Texas

Supervisor
Jack Shooter
Environmental Sciences Group
Applied Research Laboratories

Background

This project is an analysis of the whale data gathered by Dr. Loyd Hampton for the time period of July 23, 1988 - February 1, 1989. The data was collected on 57 ten inch tapes. In my analysis, I examined Hampton tape #31 (ARL tape 1092), time from 330:00:24:00 to 330:00:25:00 (comprised of 50 records). This was a recording of a bowhead whale burp. All of these data were shown in Stacy Mehevec's report in form of Lofargram #9, Alliant tape # 4946. Along with the analysis, this project demonstrates the ability of Spyglass as an effective analysis tool.

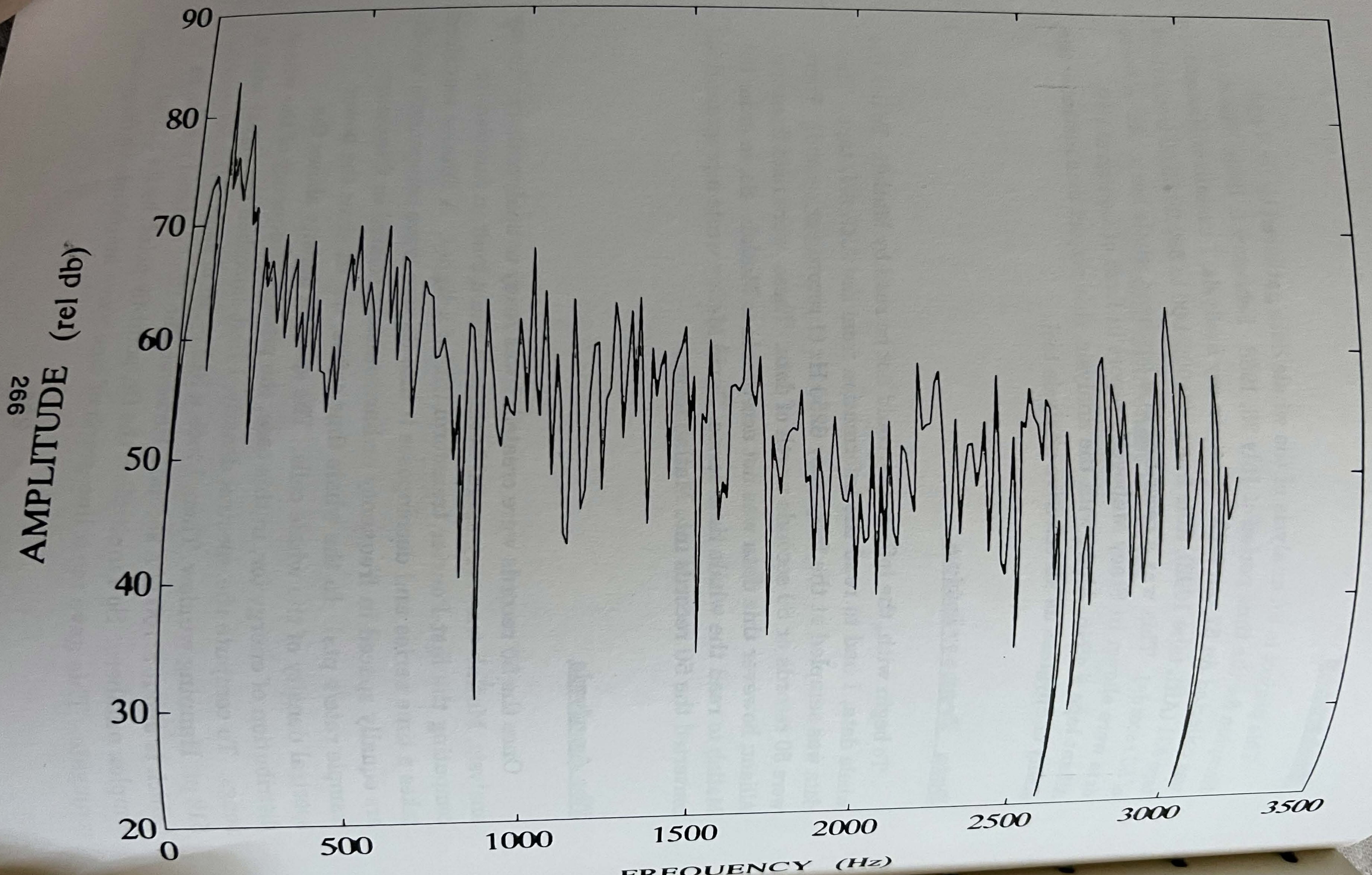
Data Transcription

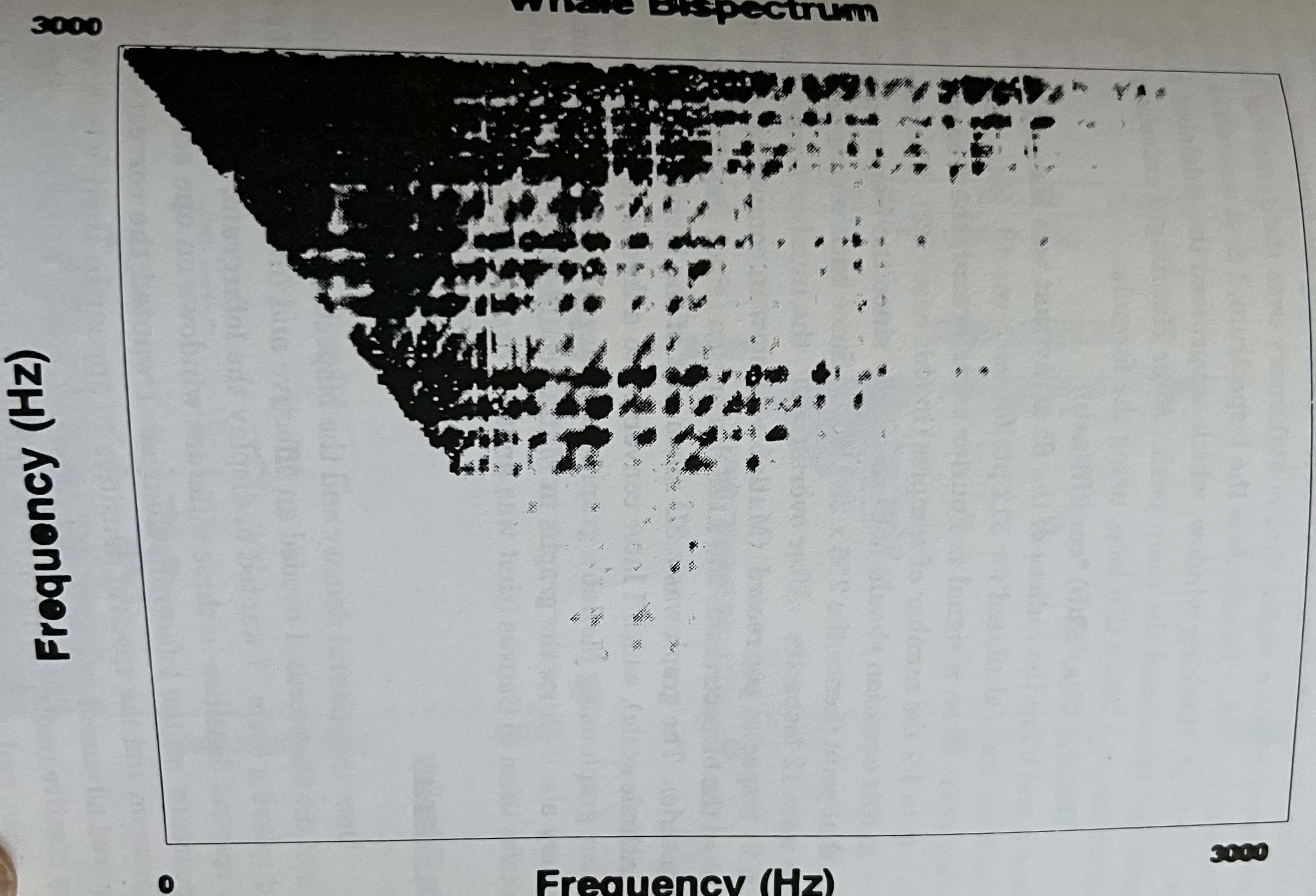
To begin with, the lofargrams could not be read by Matlab. To use the whale data, I had to read the information from the ARL A/D tape. This data was sampled at the frequency 6250 Hz (1 record per second). There were 50 records or 50 seconds worth of data. These were read from the Alliant; however this data was not acceptable to Matlab. So, in order for Matlab to read the whale time series, David McCoy wrote a program that converted the 50 records into Matlab form.

The Analysis

Once the 50 records were created, I was ready to analyze them. For my analysis, Matlab was very useful because it had a built-in function for computing the fast-Fourier transform (fft) of a signal. A Fourier transform takes a timė series and decomposes it into its sine wave components which are equally spaced in frequency or harmonically related in frequency (sample rate/# pts). In the whale time series, the fft gives the power spectral density of the whale calls. The spectral density shows the distribution of energy (or, in this case, the pitch) vs frequency of the whale songs. To compute the spectral density, I first smoothed the signal with a 512 pt. Hanning window. Then, I took a 512 pt. fft of each of the 50 records (which is about 8/100 of a second). However, the fft of a signal results in complex numbers. So I multiplied the fft, point-by-point, by its 512 pt conjugate. This gave me a line graph of sine wave amplitude vs frequency.

WHALE SPECTRAL DENSITY




Whale Bispectrum
3000
Frequency (Hz)
0
Frequency (Hz)
3000

Once the spectral density was calculated, the next step in my analysis was to take the bispectrum of each record. The bispectrum is a tool for studying the interaction of the frequency components within a signal. It is "the two-dimensional Fourier transform of the expected value of the signal at three time points. This expected value is the third-order cumulant function. The bispectrum is a tool for measuring the frequency structure of a nonlinear random signal that is stationary over the sampling period" (Brockett, p. 1386). To calculate the bispectrum, I first smoothed the signal with a 512 pt. Hanning window, which suppressed the sidelobes of the signal and broadened the main peak. After Hanning, I calculated the bispectrum, in a 256 x 256 loop, using the formula:

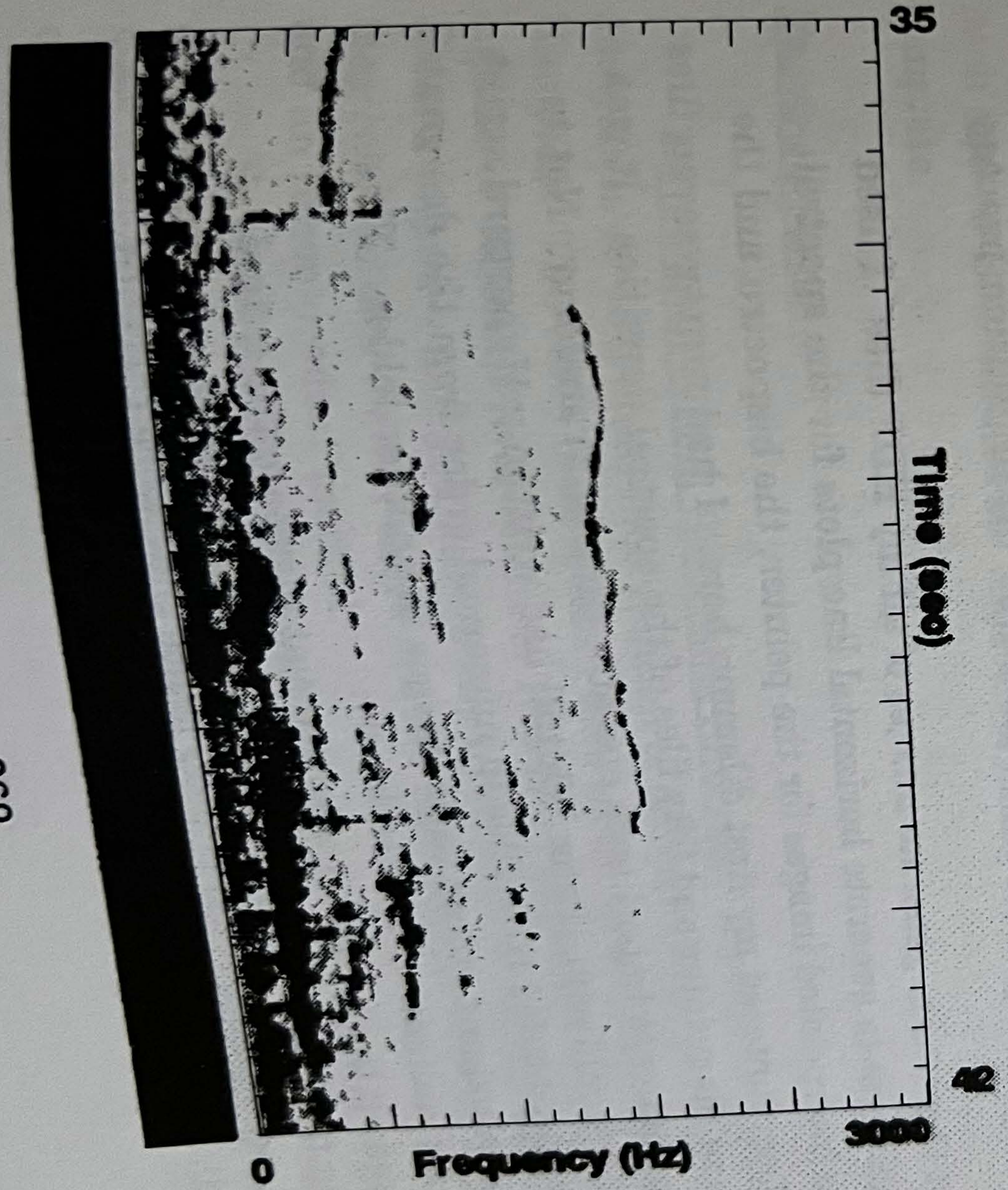
B(a,b) = fft(a)*fft(b).*conj(fft(a+b))

where a and b are the indices of the fft array that was Hanned. The bispectrum was calculated for 512 pts for each record. Since there is a folding point when a signal is Fourier transformed, the frequencies are valid up to 1/2 the number of samples (in this case 512). Therefore, the third term in this equation should not exceed 256 since the bispectrum folds over after that point (hence the 256 x 256 loop). Since each record had 6250 pts, there were 12 bispectra. After overlapping, the number doubled, and there were 24 bispectra per record. (Matlab was slow in computing this). I graphed the bispectrum in 256 x128 dimension (since the other 128 pts are symmetric). The graph was a 3D mesh plot of the bispectrum (in a logarithmic scale), which I later converted to a frequency x frequency contour graph using Matlab's graphic abilities (the contour took time and memory also). Since the graphs in Matlab couldn't be saved, I had to transfer them to Canvas (that was another drawback of Matlab).

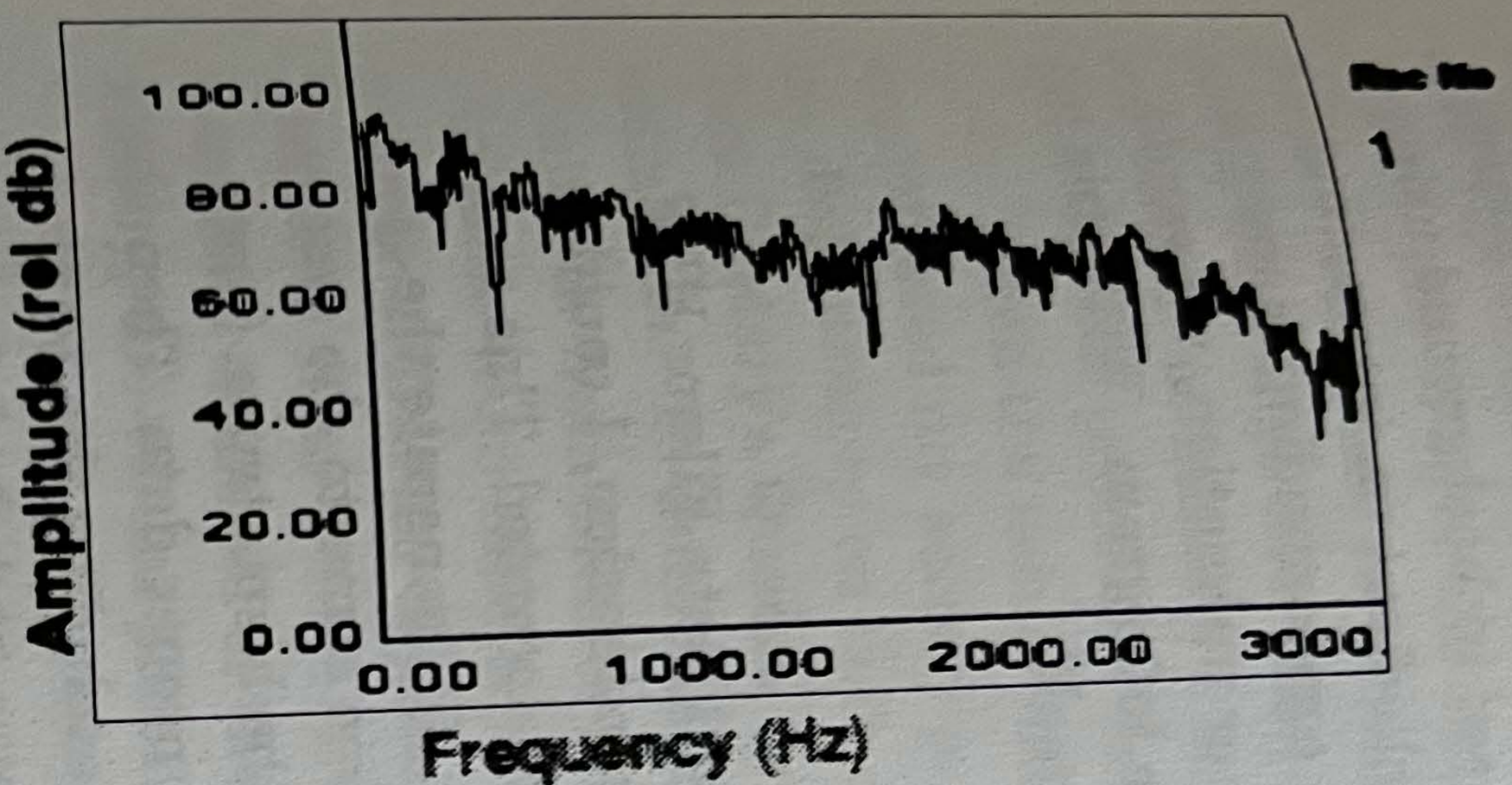
The Display

Once the spectral density and the bispectrum had been calculated for each of the 50 records, I needed an effective and interesting way of plotting or displaying them. I wanted to display the lofargram, the bispectra, and the spectral densities in three different windows on one screen. As the appearence on the lofargram changed, I wanted the corresponding bispectrum and the spectral densities to appear in their windows. This required advanced graphics and constant change of frames. Spyglass is very creative in displaying diverse graphs, using color to denote third dimension, and, most of all, it's famous for its animation. So I

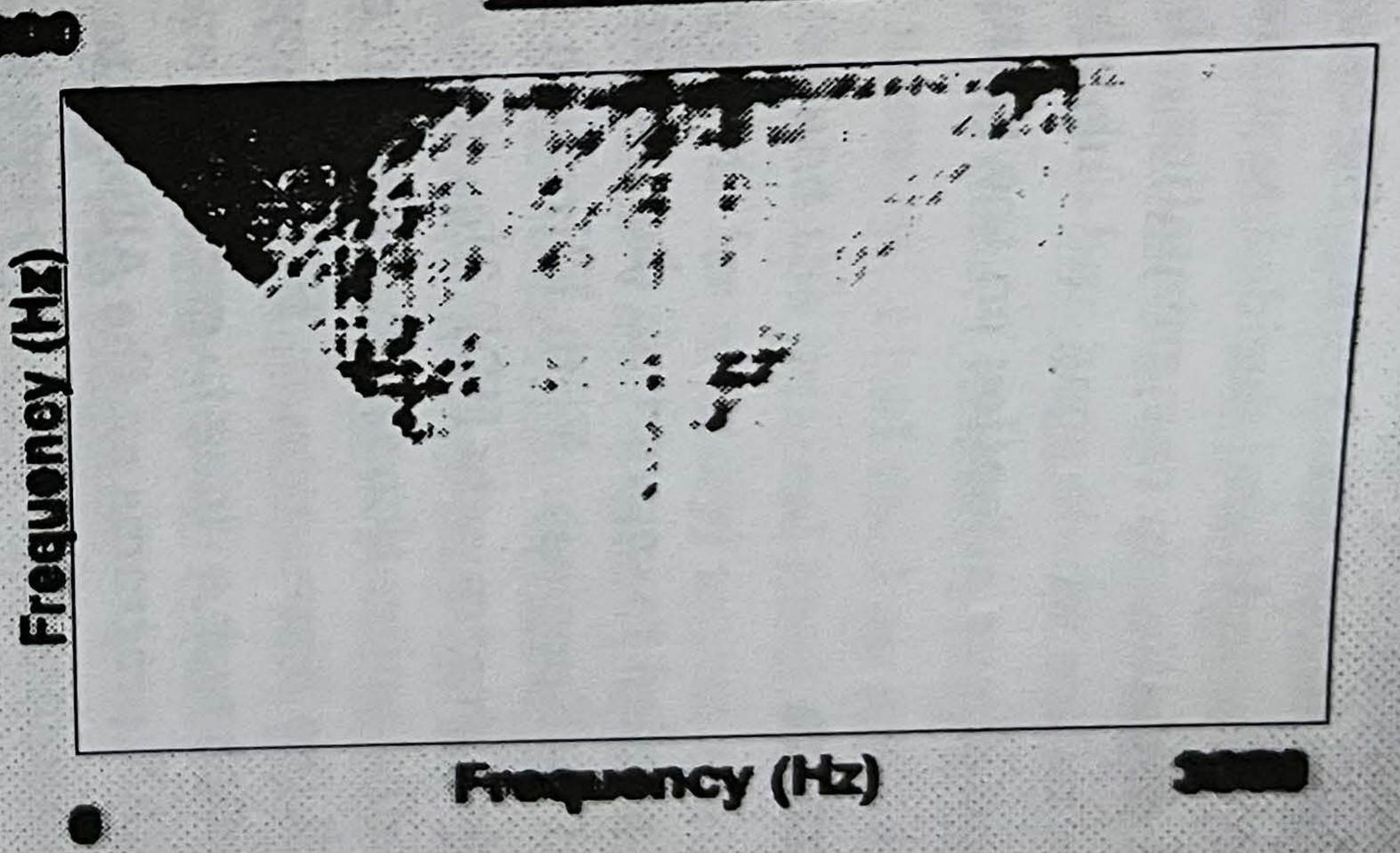
LOFARGRAM OF A WHALE CALL



SPECTRAL DENSITY



BISPECTRUM



combined View, Format, and Transform's graphing and animating abilities to put the two together with a lofargram, accompanied by a pointer.

Along with visual effects, I decided to use the whale song to accompany the animation. Jason Heuring wrote a Matlab script to convert the whale time series to sound. This could be played in background using SoundEdit Pro. In our whale song, Stacy and I picked out an interesting part of about 7 seconds and decided to animate its corresponding bispectra and spectral densities in one animation. In another animation, we would just have the entire whale song and only the entire lofargram. With this in mind, we focused our attention on the animation itself.

The Animation

First, I needed to convert all data to Hierarchical Data File or HDF, which is used by Spyglass. With Elizabeth Miller's extension, I could convert my Matlab arrays to HDF files that Spyglass accepted. This was used to make the animation for the spectral density. To generate the lofargram and the bispectrum in HDF, Stacy Mehevec came to the rescue.

Since Matlab took so long to generate and plot the bispectrum, Stacy wrote FORTRAN programs on the Alliant to mount proper data. Then she converted the data from binary to two-byte integers (excluding the 14 pt. header). Next, she generated the pointer, the bispectra and the lofargram in HDF. Once I transferred the HDF files over, my animation process began.

The first step was to generate images of all my HDF files. I used Spyglass Transform to generate horizontal line plots for the spectral densities and interpolated images for the pointer, the bispectra and the lofargram. I encountered my first dilemma here. I had ninety-seven files of bispectra and the pointer and fifty files of the spectral densities. It was very time consuming to import each one-at-a-time in Transform. Not to worry! (As Jack would say). Peter Whitfill had written a Hypercard script that automatically loaded HDF files in numerical order from the designated folder and generated and stored their images in another folder. This helped speed my animation tremendously.

Once I had all the images, I used Format to create the actual display. Using the "Place sequence" command, I loaded the spectral density, the bispectrum, and the pointer (since they were the ones to be animated). Next, using the "Place" command, I loaded the

lofargram. After everything had been loaded properly and the counter had been set, I added color, labels, and tick marks to liven up the display. Once I had the satisfactory display, I only needed to save my animation on my cartridge. Before I could save anything, I had to create two new empty folders- one to save the template and one to save all the animation images. Then I went back to Format, and, first, using the "Save as" command, I saved the image as a template in one folder. Next, using the "Save animation" command, I saved the animation in my other empty folder. I followed this procedure for both of my animations.

When the actual animation was done, I needed to play it back. Here, I encountered my second problem- memory. I had used up just about all of my disk memory on both cartridges and the internal hard drive. So I used Tom DeMary's Quadra (with a gigabyte for memory) to run the actual animation. To animate, I used Spyglass View. I simply used the "Animate from Memory" or "Animate from Disk" commands. The "Animate from memory" command didn't always load all the images (the number of frames depends on the available RAM). So I decided to go with "Animate from Disk". Even though the animation is very slow, this command loads all of the frames (depending on the memory on your disk). Finally, the visual part of my display was complete.

For the final step, we combined the sound with the animation. All it took was a little speed adjusting. This could be done both on the animation itself and on SoundEdit Pro. Once everything was together, the display was complete.

Future Plans

1. In Depth Signal Processing Analysis : A possible future step in this analysis is to do a cross-bicorrelation of different hydrophones used by Dr. Hampton. The cross-bicorrelation tells us the time delay between different signals of different hydrophones. This is an important tool in locating a specific signal when you have many hydrophones. To calculate the cross-bicorrelation of a signal, there are two methods: an inverse Fourier method and an inner product method.

First, in the inverse Fourier method, the bispectrum of the three frequencies from three hydrophones is calculated using the standard formula (see "The Analysis"). Since the " *bispectrum* is the two-dimensional Fourier transform of the cross bicorrelation"

(Pflug 2763), take the two-dimensional *inverse* Fourier transform of the bispectrum to get the **cross-bicorrelation** of the three signals. The result should show a peak at the time delay factor.

Next, in the inner product method, the bicorrelation equals the two-dimensional inner product of the three signals. To calculate this, first perform a inner product on two time shifted arrays. Then multiply this product by your third time shift arrays (see program by Jason Manning). The main problem with this method is keeping track of the indices of the three time shift arrays. The graph should, of course, be the same as that of the inverse Fourier method. An interesting study would be to compare the results of the two methods and to determine which method is more efficient.

2. Further experiments with the Bispectrum: Since the Hampton hydrophones are so widely spaced, it may not make sense to attempt a cross-bicorrelation. Therefore, we concoct a simulation to test our system. Go to the big tank. Lay out three hydrophones in different places & generate a noise signal (a spectrum analyzer is a good source for generating noise). Perform the bispectrum and further analyze this signal. In addition, cross-bicorrelate only 1/30 th of a second of data using the two methods and compare the results against the known time delay.

3. Using Spyglass as an analysis tool : There is a promising use of Spyglass for examining various aspects of data at once. Spyglass animation is a versatile tool for giving a visual image of how the data and its aspects vary and evolve with time. Interesting experiments and applications with animations could be done not only in Signal Processing, but for analysis in other fields as well.

Appreciation

First, I deeply appreciate Stacy Mehevec's help, dedication, and support in this project. I want to thank Tim Scoggins and Peter Whitfill for their help with Spyglass and this animation. I also would like to thank David McCoy, Jason Manning, Elizabeth Miller, and Jason Heuring for their contributions to this project. I appreciate Tom DeMary allowing me to use his computer. I would like to thank my supervisor, Jack Shooter, for his support and guidance on this project. I would also like to thank Dr.

References

Brockett, Patrick L., Hinich, Melvin, and Wilson, Gary R. " Nonlinear and non-Gaussian ocean noise." The Journal of the Acoustical Society of America. October 1987. pp. 1386-1388.

Hinich, Melvin J. and Wilson, Gary R. "Detection of Non-Gaussian Signals in Non-Gaussian Noise Using the Bispectrum." IEEE Transactions on Acoustics, Speech and Signal Processing. July 1990. pp. 1126-1130.

APPENDIX

Matlab Programs :

1. Spectral Density
2. Bispectrum
3. Correlation: Inner Product Method
4. Cross-Bicorrelation: FFT Method

CALCULATING SPECTRAL DENSITY IN MATLAB

1) Results in Frequency vs Sine wave amplitude graph

```
%sme='my cart:Old HDF files:spec files:spec50'
A='H07920_S01490_M348'
B =['load ',A]
eval(B)
C=['sqre=',A,"";'];
eval(C);
S=fft(sqre,1025:1536);
P=S.*conj(S)/512;
Pyy=10*log10(P(1:256));
f=6250/512*(0:255);
%pw=MatlabtoHDF(sme,Pyy,1,f);
S=[]; sqre=[];
plot(f,Pyy(1:256));
%title('POWER SPECTRAL DENSITY OF A WHALE BURP');
%xlabel('FREQUENCY    (Hz)');ylabel('AMPLITUDE');
```

2) Results in Time vs Frequency graph

```
load  H07920_S01477_M319;
sqre=H07920_S01477_M319';
%plot(sqre);
Length=512;
x=zeros(1:256);
y=zeros(1:45)';
B=y*x;
for  n=1:45
      y=zeros(1:45)';
      y(n)=1;
      N=((n-1)/4)*(Length)+1;
      M=N+(Length)-  1;
      sig=sqre(N:M);
      trans=fft(sig);
      Pwr=   trans(1:256).*conj(trans(1:256));
      x=10*log10(Pwr);
      B=B+y*x;
end
plot(mesh(B'));
```

CALCULATING THE BISPECTRUM IN MATLAB

```
 sampno=[1:512];
 load  H07920_S01477_M319;
 sqre=H07920_S01477_M319';
 han=.5*(1-cos(2*pi*(sampno-1)/512));
 window=sqre(1:512).*han;
a=fft(window);
for  num=1:256
   for k=1:256
        b(k,num)=a(k)*a(num).*conj(a(k+num));
    end
  end
 lb=abs(b);
 lc=.01*ones(lb);
 lb=max(lb,lc);
 lb=10*log10(lb);
%try='The Lil" (Big) Disk:Samptry'
%Trial=MatlabtoHDF(try,lb);
```

PERFORMING SIMPLE CORRELATION USING INNER PRODUCT METHOD

```
%Generate random noise-take portions of it to
%form 1x128 signals.
t=[0:0.01:1.37];
rand('normal')
hydro= 2*rand(t);
y= hydro(1:128);
x= hydro(10:137);

%Correlate one chunk of y against the x.
a=x;
b=y(20:108);
for ind=1:40
  m=88+ind;
  X=a(ind:m);
  c(ind)=b*X'/sqrt((X*X')*(b*b'));
end

%Plot Correlation against time axis.
n=1:40;
lag(n)=(n-20);
plot(lag,c);
```

program by Jason Manning

CROSS-BICORRELATION USING FOURIER TRANSFORM METHOD

```
% Perform bispectrum on random noise with known time delay. The
inverse FFT of     % this bispectrum should give us the cross
correlation of the three signals.

% Create random noise and use portions of it to create
% three signals with time delay factor.

%hdf='The Lil" (Big) Disk:Corr'
t=[0:0.01:2.27];
rand('normal')
hydro=  2*rand(t);
H1=  hydro(1:128);
H2=  hydro(10:137);
H3=  hydro (25:152);
%subplot(221),plot(hydro);title('Big   Plot');
%subplot(222),plot(H1);title('H1');
%subplot(223),plot(H2);title('H2');
%subplot(224),plot(H3);title('H3');

% Smooth out the noise with a 128 pt. Hanning window.
sampno=[1:128];
han=0.5*(1-cos(2*pi*(sampno-1)/128));
w1=H1(1:128).*han;
w2=H2(1:128).*han;
w3=H3(1:128).*han;
%subplot(221),plot(w1);title('W1');
%subplot(222),plot(w2);title('W2');
%subplot(223),plot(w3);title('W3');

% Add 128 zeros to the Hanning window & FFT- will be 256 pt. FFT.
T=zeros(1,128);
F1=[w1,T];
F2=[w2,T];
F3=[w3,T];
f1=fft(F1);
f2=fft(F2);
f3=fft(F3);

% Perform the  256x256 bispectrum- for the three hydrophones.
```

```
% The bispec. is symmetric when the index is greater than 256.
for num=1:256,
for k=1:256,
   if (num+k)<=256,
      b(k,num)=f2(k)*f3(num).*conj(f1(k+num));
   elseif (num+k)>256
      b(k,num)=f2(k)*f3(num).*conj(f1(k+num-256));
   end
 end
end

% The two-dimentional inverse FFT should give us the bicorrelation
% of the three hydrophones.
CC= ifft2(b);
%lb=abs(CC);
%lc=.01*ones(lb);
%lb=max(lb,lc);
%lb=10*log10(lb);
lb=fftshift(CC);
plot(mesh(lb));
%s=MatlabtoHDF(hdf,lb);
```