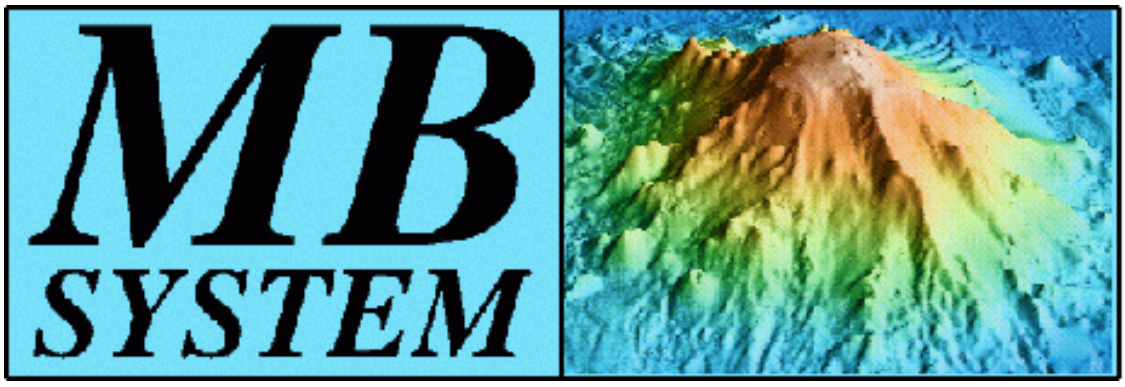


The MB-System™ Cookbook

Val Schmidt, Columbia University
Dale Chayes, Columbia University
Dave Caress, Monterey Bay Aquarium Research Institute

The MB-System Cookbook

by Val Schmidt, Dale Chayes, and Dave Caress

Table of Contents

List of Figures

Version

Version Information

$Id: mbcookbook.xml,v 1.15 2006/02/16 11:39:16 dale Exp $

Like the MB-System™ sources, this is a working document and as such is subject to change without notice.

```
$Log: mbcookbook.xml,v $
Revision 1.15  2006/02/16 11:39:16  dale
Change the "Version" section to show the RCS-style log file tags.

Revision 1.14  2006/02/16 11:06:32  dale
Removed the "pre-release" note

Revision 1.13  2004/02/20 14:10:58  vschmidt
Added pseudo-attribute to xml declaration to specifiy UTF-8 encoding. Required

Revision 1.12  2004/01/15 18:13:24  vschmidt
Fixed syntax to fully support xml/xslt processing.

Revision 1.11  2003/09/08 13:58:04  vschmidt
Added a new chapter "Statistics" and the relative entity entries.  Also modifie

Revision 1.10  2003/07/27 20:54:36  vschmidt
Added or fixed rcs/cvs keyword identiifers for Id, Name and Log

Revision 1.9  2003/07/27 20:33:50  vschmidt
Fixed rcs/cvs keyword identiifers for Id, Name and Log
```

Chapter 1. Introduction

1.1. What Is This Document About?

The MB-System™ Cookbook provides detailed examples regarding the processing of swath mapping data using the MB-System™ open-source software package. Step-by-step instructions, explanations, code (scripts), data files and processed results are provided for each example. The examples use real data files, with typical data irregularities.

MB-System™ supports many of the data formats and their variations that have evolved over the years. Accommodating these changes is a forever and ongoing process. The MB-System™ Cookbook is an attempt, in part, to document when data formats require special techniques in the processing.

1.2. How Do I Get The Most Current Version?

The most up to date version of this document can be obtained from the MB-System™ web page at

http://www.ldeo.columbia.edu/MB-System/MB-System.intro.html

where it can be found in both Portable Document Format (PDF) and Hypertext Markup Language (HTML). Alternatively, it can be downloaded via anonymous ftp from the Lamont-Doherty Earth Observatory at:

ftp.ldeo.columbia.edu

1.3. What is MB-System™

MB-System™ is a collection tools used to process research grade swath mapping sonar data in more than four-dozen formats from sonar equipment manufactured and operated around the world. MB-System™ is used in conjunction with the Generic Mapping Tools (GMT) created by Paul Wessel of the University of Hawaii and Walter Smith of NOAA. GMT is a powerful set of processes used to manipulate data and to create Encapsulated Post Script maps and charts.

1.4. What Kind of Data Sets Can MB-System™ be Used On?

MB-System™ rides atop a library of functions for reading and writing swath mapping sonar data files called MBIO,(Multi Beam Input-Output).While largely unseen by the user, MBIO is where the rubber meets the road. MBIO takes into account multitudes of details, including existence of side scan or and or amplitude data, interpolation of navigation to ping times, geometry of specific sonar models, and of course the various data formats themselves. While MBIO does not support every possible data type, it has grown to accommodate the bulk of the sonar data formats common the the multibeam community. Indeed, considerable development is ongoing to support ever greater variations in sonar data formats, created both by commercial vendors and research institutions.

In addition to the native data formats, MBIO defines many MB-System™-only data formats. These have been created out of necessity when vendor-native formats fail to accommodate the needs required of sonar processing, or are too unwieldy to store. For example, some Simrad sonars have traditionally stored navigation information separate from the bathymetry information, requiring interpolation of navigation to ping times each time the data is read. MBIO defines a data format stores the bathymetry and the interpolated navigation in a single composite file. Similarly, native Hydrosweep DS-2 data contain no less than 9 separate data files with multitudes of ancillary information regarding the sonar's operational settings, far more information than the sonar data itself. In this case, MBIO defines an alternate

format that condenses the necessary navigation, bathymetry and sidescan information into a single, more manageable and considerably smaller file. Many research institutions have found the MB-System™ formats preferable to the native sonar formats and perform a real time conversion of sonar data for their users.

Stating which sonar and sonar formats are supported by MB-System™ is something of a moving target, however as of the time of this writing the following sonars were supported:

- SeaBeam "classic" 16 beam multibeam sonar
- Hydrosweep DS 59 beam multibeam sonar
- Hydrosweep MD 40 beam mid-depth multibeam sonar
- SeaBeam 2000 multibeam sonar
- SeaBeam 2112, 2120, and 2130 multibeam sonars
- Simrad EM12, EM121, EM950, and EM1000 multibeam sonars
- Simrad EM120, EM300, EM1002, and EM3000 multibeam sonars
- Hawaii MR-1 shallow tow interferometric sonar
- ELAC Bottomchart 1180 and 1050 multibeam sonars
- ELAC/SeaBeam Bottomchart Mk2 1180 and 1050 multibeam sonars
- Reson Seabat 9001/9002 multibeam sonars
- Reson Seabat 8101 multibeam sonars
- Simrad/Mesotech SM2000 multibeam sonars
- WHOI DSL AMS-120 deep tow interferometric sonar
- AMS-60 interferometric sonar

1.5. Important Notes Regarding the Data Used in the MB-System™ Cookbook

In an effort to demonstrate MB-System™ in as realistic manner as possible we have provided several real datasets as examples. These are available in three separate archives that may be optionally download from the MB-System™ ftp site.

The standard MB-System™ distribution comes with an archive of example data sets and scripts generated using MB-System™ (MB-SystemExamples.X.Y.Z.tar.Z). These are referenced on occasion in the Cookbook, and since their size is small they can readily be downloaded. When uncompressed and unarchived, the resulting directory tree will resemble the lines below.

```
[vschmidt@val-LDEO mbexamples]$ ls
data mbbath mbgrid mbinfo mblist mbm_plot README xbt
```

The `~/data` directory contains several sample data files used in these examples. The other directories contain scripts that briefly demonstrate the use of several of the tools in MB-System™ .

For the purposes of the MB-System™ Cookbook, however these data sets were not sufficient to demonstrate the intricacies of various processing techniques, nor the art of managing a data set within a larger archive. Therefore, for the purposes of the examples contained herein, you will find two additional collections of data that are freely download-able such that you may follow along with each step we demonstrate.

The first collection of data has been added to the `mbexamples.tgz` archive file which is a superset of the standard set of examples described above. That is to say, this archive contains all the examples in the standard set, plus a collection of "other data sets" for illustration of svp corrections, roll bias corrections and general survey plotting. A directory called `~mbexamples/cookbook_examples` has been created which will serve as a home directory for calculations utilizing these data sets.

These smaller data sets do not illustrate the processing and management of data within a larger archive however. To this end, we have provided a completely separate archive containing several data sets mapping the volcanic seamount Loihi, south of the Big Island of Hawaii. Within the `loihi` data archive you will find four data sub directories, each with their own Loihi data sets. The entire archive is some 660 MB in size.

Chapter 2. Multibeam Sonar Basics

2.1. How Sound Travels Through Water

The basic measurement of most sonar systems is the time it takes a sound pressure wave, emitted from some source, to reach a reflector and return. The observed travel time is converted to distance using an estimate of the speed of sound along the sound waves travel path. Estimating the speed of sound in water and the path a sound wave travels is a complex topic that is really beyond the scope of this cookbook. However, understanding the basics provides invaluable insight into sonar operation and the processing of sonar data, and so we have included an abbreviated discussion here.

2.1.1. Speed of Sound

The speed of sound in sea water is roughly 1500 meters/second. However the speed of sound can vary with changes in temperature, salinity and pressure. These variations can drastically change the way sound travels through water, as changes in sound speed between bodies of water, either vertically in the water column or horizontally across the ocean, cause the trajectory of sound waves to bend. Therefore, it is imperative to understand the environmental conditions in which sonar data is taken.

2.1.1.1. Affects of Temperature, Salinity and Pressure

The speed of sound increases with increases in temperature, salinity and pressure. Although the relationship is much more complex, to a first approximation, the following tables provides constant coefficients of change in sound speed for each of these parameters. Since changes in pressure in the sea typically result from changes in depth, values are provided with respect to changes in depth.

- Change in Speed of Sound per Change in Degree C

  ---> + 3 meters/sec)

- Change in Speed of Sound per Change in ppt Salinity

  ---> + 1.2 meters/sec

- Change in Speed of Sound per Change in 30 Meters of Depth

  ---> + .5 meters/sec)

[1]

Temperature has the largest effect on the speed of sound in the open ocean. Temperature variations range from 28 F near the poles to 90 F or more at the Equator. Of course, relevant temperature differences with regard to multibeam sonar systems are the variations that occur over relatively short distances, in particular those that occur with depth. These are discussed further below.

While the salinity of the world's oceans varies from roughly 32 to 38 ppt. These changes are gradual, such that within the range of a multibeam sonar, the impact on the speed of sound in the ocean is negligible. However near land masses or bodies of sea ice, salinity values can change considerably and can have significant effects on the way sound travels through water.

While the change in the speed of sound for a given depth change is small, in depth excursions where temperature and salinity are relatively constant, pressure changes as a result of increasing depth becomes

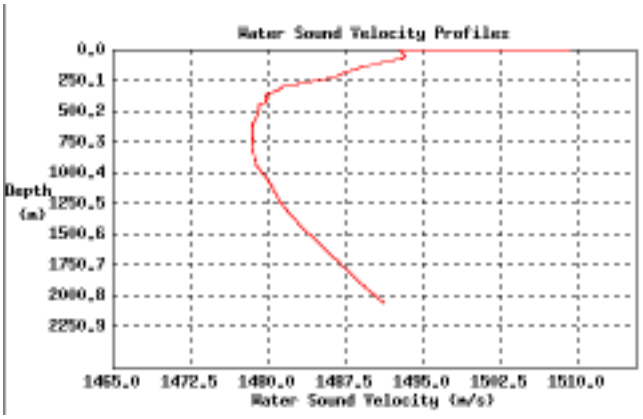
[1] Ulrich p 114.

the dominating factor in changes in the speed of sound.

2.1.1.2. The Sound Speed Profile

The composite effects of temperature, salinity and pressure are measured in a vertical column of water to provide what is known as the "sound speed profile or SSP . A typical SSP is shown below:

Figure 2.1. Typical SSP



From the SSP above, one can see an iso-speed layer from the surface down to a few tens of meters due to mixing of water from wave action. This layer is called the "mixed surface layer" and is characterized by a flat or slightly negative (slanting down to the right) sound speed profile. The iso-speed layer is followed by a seasonal thermocline down to about 250 meters. Below, a larger main thermocline exists. These variations in the SSP are almost entirely due to changes in temperature of the water. Below the main thermocline, the temperature becomes largely constant and changes in pressure due to depth have the dominant effect on the SSP causing it to gradually increase.

2.1.2. The Effects of Sound Speed Errors

There are two fundamental sound speed measurement inputs into multibeam sonar systems. These are 1) the speed of sound at the keel of the ship in the vicinity of the sonar array, and 2) the profile of sound speed changes vertically in the water column. The former is used in the sonar's beam forming while the latter is used more directly in the the bathymetry calculations.

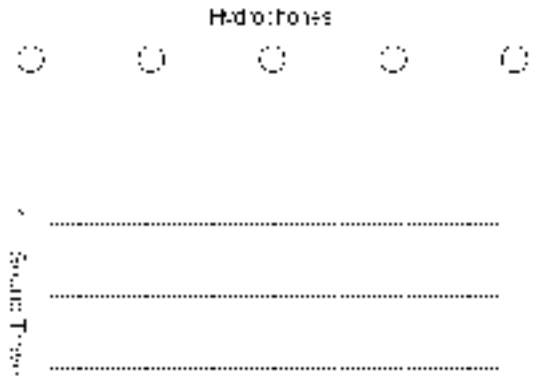
2.1.2.1. Sound Speed Errors at the Keel

In an effort to understand the effect on sonar performance of an incorrect sound speed at the keel, we must discuss the process of beam forming. This discussion surrounds a single simplified beam forming method, of which there are many. However it illustrates the effects well and the results can be applied to any sonar system.

The sound speed at the keel of the ship, local to the array, affects the directivity of the beams produced by the sonar. The result is that the sonar is not exactly looking in the direction we expect, introducing considerable error.

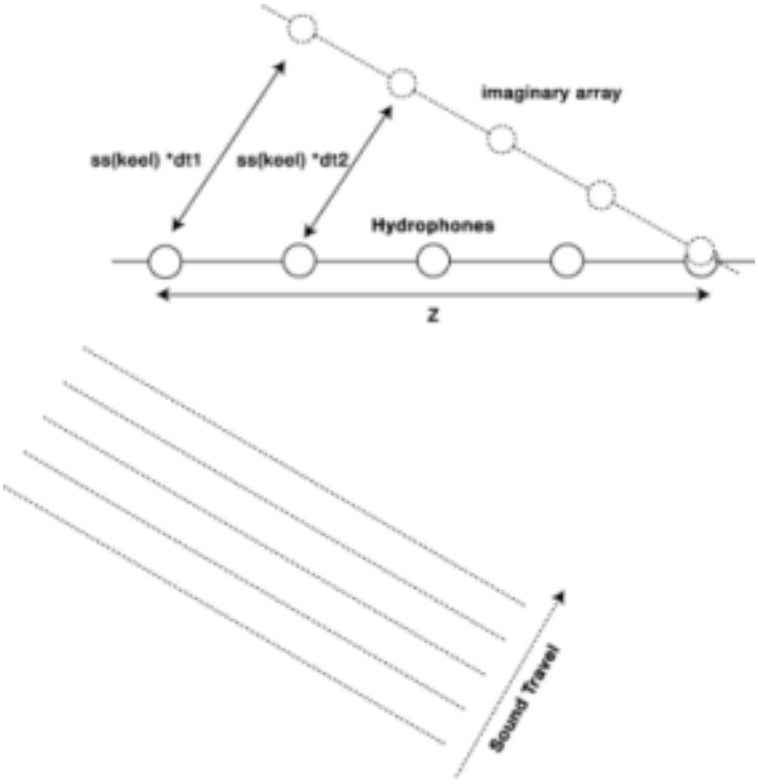
In multibeam sonar, a beam is formed by summing the sounds measured by multiple hydrophones time delayed by a specific amount with respect to each other. The following illustration depicts an array of hydrophones with an incident planar sound wave.

Figure 2.2. Diagram of Planar Soundwaves Orthogonally Incident on a Linear Hydrophone Array.



From this figure above, it is clear that the hydrophones will measure the incoming sounds simultaneously. However when the sound wave is incident from some angle , the hydrophones closer to the source will detect the sound prior to those farther away. Look at the figure below.

Figure 2.3. Diagram of Beam Forming Performed by A Linear Array



To listen in that direction, therefore, sonar systems sum the sounds measured from each hydrophone

after delaying its measurement by the amount of time it would take for the sound to travel from the closest hydrophone. The result is an imaginary hydrophone array shown in the figure above, where the angle with respect to the actual array might be considered the direction the sonar beam is pointing. In this way, sounds coming from this direction are measured at each hydrophone in the imaginary array simultaneously.

As an example, assume the array is attempting to *listen* in a direction 30 degrees from its orientation. The sonar calculates the distance between the first hydrophone and the last hydrophone in the direction it is listening. This distance, *X* is simply

X = Z Sin (30)

The sonar divides the speed of sound at the keel into this distance *X* to get the time it will take the sound return to travel from the closer hydrophone to the further one.

t=X/ss

The measurements of the closer hydrophone are then summed with those of the further hydrophone delayed by this interval such that the arrival of the sound signal at the last hydrophone coincides with the delayed measurement of this closer hydrophone.

If the sound speed used to calculate this delay time is too low, the calculated delay time will be too large. A larger delay time results in a beam that is pointed further away from orthogonal than has been assumed.

If all this is confusing, just try to remember:

SS(Keel) Too Low - > Beam Fan Pattern Too Wide

SS(Keel) Too High - > Beam Fan Pattern Too Narrow

It is important to realize that the beam perpendicular to the plane of the sonar array is directionally unaffected. Only the beams formed by delaying the signals from individual hydrophones look in the wrong direction when the speed of sound at the keel is in error. This implies, of course, that the effect a sound speed error has on any given data set are largely dependent on the orientation of the installed sonar.

Many sonar arrays are installed flush with the hull of the ship and parallel to the sea (nominal) floor. A sonar array installed parallel to the sea floor sees little or no error in the nadir beam due to errors in the keel sound speed. However, because of the high beam angles at the outer beams, the errors are exacerbated at the edges of the swath. When the value is too low, the beam fan shape is erroneously wide, causing the measured bathymetry at the outer beams to be too deep. (Sound travel times are erroneously longer for the wider beams.) This causes the cross track shape of a swath to "frown". The converse is true, of course, for keel sound speeds that are too high.

Other sonar arrays are installed on a V-shaped structure mounted to the bottom of the ship or towfish. For these sonars, the zero angle beam, whose direction is unaffected by the keel sound speed, is that which is formed perpendicular to the array at some angle from nadir. The effect on the beam pattern, is the same - keel sound speed too low -> beam pattern is too wide - keel sound speed too high -> beam pattern too narrow. However, in this case, the effects on the data set are slightly different. A wider beam pattern does indeed cause erroneously deep bathymetry values at the beams furthest from the ship's track, but the beams directly beneath the ship will be erroneously shallow. Similarly, a narrower beam pattern causes erroneously shallow bathymetry values at beams furthest from the ship's track and erroneously deep values directly beneath the ship.

In both sonar installation orientations positive keel sound speed errors result in "frowns" in the cross track swath profile, while negative errors result in "smiles". The distinction to make is that the beams unaffected by these errors are beneath the ship for the former and at an angle orthogonal to the sonar array for the latter.

One final effect to point out. Multibeam sonars use beam forming both for projection as well as for reception of sound pulses. Moreover, beam forming is done, not only in the athwartships direction to create a swath, but fore and aft to increase measurement resolution. Typically the beam footprint created by the projectors on the sea floor is large compared to that created by the receivers. In this way, errors in the receive footprint still fall within that of the projection footprint. However significant sound speed errors at the keel can upset this balance lowering the signal to noise ratio of the sonar and compromising data quality.

Here's the kicker. ERRORS IN KEEL SOUND SPEED ARE (typically) NOT RECOVERABLE! Let us say that again:

Warning

ERRORS IN KEEL SOUND SPEED ARE NOT RECOVERABLE!

Because sonar systems do not save wave forms from individual hydrophones, one cannot go back and apply corrected sound speed values to beam forming calculations. This value must be correct the first time. Therefore, and it cannot be stressed enough, *sonar operators must always be conscience of the correct operation of any device that provides the keel sound speed measurement to the sonar.*

2.1.2.2. Sound Speed Profile Errors

Bathymetric sonar systems calculate water depths by measuring the time it takes a sound pulse to travel to the bottom and back to the receiver. To translate these time measurements into distances, one must know the speed at which sound travels through water and the general trajectory the sound traveled. As we have seen, temperature, pressure, and salinity all contribute to the speed of sound in water. Moreover, differences in sound speed across the water column acts as a lens bending the path that sound travels. For these reasons, it is imperative to have accurate sound speed profiles for any data set.

Inaccurate sound speed profiles may be the single largest correctable cause of bathymetry errors in multibeam sonar data. Understanding the various effects errors in sound speed can have on sonar data is challenging and deserves careful consideration. Even more difficult, is recognizing these errors from other sources of error in the data. A solid theoretical understanding is essential.

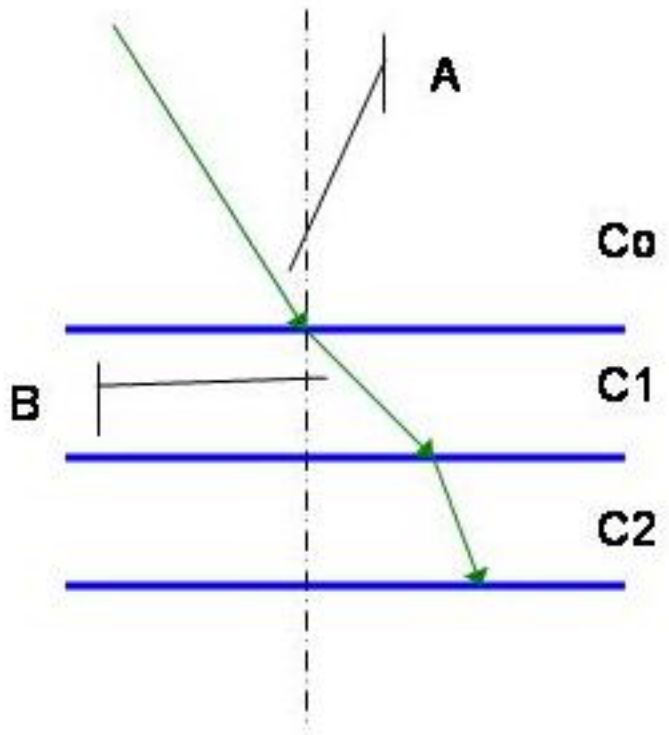
Errors in the sound speed profile produce predictable, although often confused, results in the data.

For example, a simple step offset in the sound speed profile will cause the calculated bathymetry to be shallower for higher sound speeds, and deeper for lower sound speeds. Not so obvious, is that relative changes in sound speed down through the water column cause sound to bend, a process called *refraction*. Therefore, errors in the relative changes in sound speed through the water column cause errors in the calculated sound trajectory. Because the bending is larger for sound traveling at oblique angles to the gradient, oblique beams (usually the outer ones) have more error. All of this is further complicated by the fact that sound speed profile errors high in the water column can exacerbate or offset the effects of those lower in the water column.

2.1.2.2.1. Refraction

It is convenient, when talking about sound and its travel path from a point, to consider the path as a ray. This is a common technique in wave theory, used in optics and other sciences. In a homogeneous medium sound does indeed travel in a straight line. However, when a sound wave passes between two mediums having different sound speeds its direction is bent. This is a property of waves more than sound itself, and many theories and explanations, with varying degrees of success, have been put forth over they years. While not true in all cases a simple one for illustration is offered here.

Figure 2.4. Snell's Law



Consider sound wave incident on a boundary between two bodies of water having differing sound speeds as shown in the figure above. Snell's Law states that the Cosine of the initial angle of incidence divided by the initial sound speed is a constant as the sound passes from regions of differing sound speeds.

Cos (A)/ Co = Constant

Therefore, if the new region has lower sound speed, the Cosine of the angle of incidence must be less than that of the previous region. Hence the new angle of incidence is less. So one might imagine the sound bending "toward" regions of lower sound speed, in the sense that the angle of sound travel is more steep, and "away" from regions of higher sound speed in the sense that the angle of incidence is more shallow.

Then if one knows the angle that a sound wave left its source, and the sound speed profile of the medium it passes through, one can calculate the path the wave takes over its entire trip.

A slightly more complicated example is one where the sound speed of a single medium changes linearly with depth rather than at discrete depth intervals. In this scenario, the sound ray bends continuously over its travel path. It can be shown that the path the sound travels is that of the arc of a circle whose radius, R, is the ratio of the initial sound speed and the slope of the gradient (R = Co/g).[2]

This process, of calculating the course of travel of a sound ray through a water column of varying sound speed, is called "ray tracing" and is essential to sonar performance. Since sound does not travel as a straight line though the ocean, sonar systems must calculate propagation paths for both the the ping and return to calculate the correct distance and direction of the reflecting object.

2.1.2.2.2. SSP Errors and the Resulting Bathymetry

[2]Ulrich pg 123-125.

To be sure, the most likely source of sound speed profile errors is simply not measuring it frequently enough. Sound speed profiles change as bodies of waters change and whether, due to insufficiently frequent measurements or plain old inattention, inevitably sound speeds change before sonar operators notice.

But assuming your monitoring the sonar performance, and measurements have been carefully planned, we can consider the likely sources of error. Since in the open ocean temperature and pressure are the largest contributors to changes in sound speed, errors in the sensors aboard XBT and CTD sensors are of central concern. Over the relatively small temperature variation seen by these devices, temperature measurement response is typically quite linear. However inexpensive and non-calibrated sensors common on XBT's and CTD's often have step offset errors (the measurement will be off by a fraction of a degree or more over the entire range). Conversely, because pressure sensors must operate over such a large range (over to 2000 psi), they are much more prone to non-linearity (the measurement error will typically increase with depth).

Consider how these two errors would affect a sound speed profile. An constant temperature error will most significantly affect the portions of the sound speed profile in the thermocline with a proportionally smaller affect in the other regions. Conversely, a depth sensor non-linearity will most significantly affect the deep depth areas and have less proportional affect on the regions where changes in temperature dominate.

2.1.3. Spreading Loss

The energy radiated from a omni-directional transducer spreads spherically through a body of water. Since all the energy is not directed in a single direction but in all directions, much of the energy is lost. This is called spreading loss. In deep water, to a first approximation, sound spreads spherically from the source, and the power loss due to this spreading increases with the square of the distance from the source. The classical equation is TL=10 Log (r^2) or 20 Log (r) where TL is the transmission loss and r is the distance from source.[3]

In shallow water beyond a certain distance the travel of sound is bounded by the surface and the sea floor resulting in cylindrical spreading. In this case, sound power loss increases linearly with the distance from the source. The classical equation in this case is TL= 10 Log (r), again where TL is the transmission loss and r is the distance from the source.[4]

2.1.4. Absorption

As sound travels through a body of water, some of the energy in the sound waves is absorbed by the water itself resulting in an attenuation of the amplitude of the original sound wave. The amount of energy that is attenuated in the water column is frequency dependent, larger frequencies exhibiting much larger attenuation than lower ones. [5]

To a first approximation, spherical spreading and absorption losses can be approximated by TL=20 Log (r) + ar where a is a frequency dependent constant with units of dB per unit distance.

2.1.5. Reverberation

Reverberation is a measure of the time it takes a the energy of a sound pulse to dissipate within the water column. Imagine yelling "Hellooooooooo!" from the center of large arena. You'd hear all kinds of echos from the surrounding walls and seats, and since they are distant, the echoes would be delayed from your initial shout. The result would be lots of echos that might last up to several seconds "HELLOOO, Hel-

[3]Ulrich p 101.
[4]Ulrich p 102.
[5]The topic of absorption is a very interesting one. It has been found be have a much more complex mechanism than a simple viscous heating. Ionic relaxations of MgSO4 and Boric Acid (which are themselves depth, temperature and pH dependent) have been shown to have effects on the absorption of sound (> ~ 5 kHz) in sea water. (Ulrich pgs 102-111.)

looo hellooooo." This is reverberation.

In music halls, reverberation is a desirable thing, as it reinforces the sound waves in a wonderful way to create a more full-sounding experience for the listeners. However in theaters reverberation is not desirable, as the repetitive echos tend to interfere with the clarity and understanding of speech.

As you might expect, the repetitive echos are also a problem for sonars. Sonars attempting to find the bottom can become confused by loud repetitive echos. The result is a drop in the signal-to-noise ratio which results in a higher concentration of false bottom detects. Of course, as the signal of interest becomes smaller the echoes become more of a problem. Hence high reverberation will tend to affect the outer beams somewhat more than the closer ones.

The conditions that cause high reverberation are similar in the ocean to those with which we are more familiar. Like a stairwell with brick walls, deep waters with acoustically hard sea floors act as good reflectors. These conditions cause the largest problems.

To be quite honest, most operators of multibeam sonar systems pay little attention to reverberation levels in the ocean. Indeed, sonars are not designed to provide any indication to the operator that reverberation levels are high. And while ocean sea floor acoustic hardness data is available these are not typically consulted prior to a cruise.

Instead, sonar operators see an increased noise level in their sonar and perhaps a narrowing of the effective swath width due to noise in the outer beams. Because these effects can be caused by any of a multitude of problems and tend to come and go, reverberation level is rarely identified as the cause.

2.2. How Sound Interacts with the Sea Floor

The amount of sound reflected from the sea floor is highly variable. It is dependent on the angle of incidence of the sound wave, (called the *grazing angle*), the smoothness of the sea floor, the sea floor composition, and the frequency of the sound.

Sound energy is well reflected when it bounces off a flat surface normal to the sound waves path of travel. However at an oblique angle, much of the sound is reflected at a complementary angle away from the receiver. Similarly rough surfaces tend to scatter the sound energy in directions away from the source. This generally dissipates the received sound level, but can enhance it when the angle of interception with the surface would otherwise reflect most of the sound energy away.

Some of the sound energy is lost into the sea floor itself. The amount sound energy will propagate into the sea floor is highly dependent on the frequency and bottom composition. For a typical bottom type and nominal source level, frequencies above 10kHz penetrate very little. From 1kHz to 10kHz sound often penetrates to several meters of depth. From 100 Hz to 1kHz sound can penetrate to several 10s of meters or more. Below 100 Hz sound waves have been detected traveling at various depths in the earth's crust around the globe.

2.3. Acoustic Interference

Acoustic interference is somewhat loosely defined as any unwanted acoustic source that increases the sound levels detect by the sonar with respect to the bottom signal.

2.3.1. Radiated Noise or Self Noise

For example, an increase in noise levels in and around the sonar transducers as a result of radiated noise from the host ship is one common type of interference. Typically sonar transducers are installed in the bow of a ship, well away from the engineering spaces and heavy machinery, to reduce the chances of interference problems. However sometimes the best attempts during installation to prevent radiated noise interference fail, resulting in a particularly poor performing sonar. After installation, these problems tend

to be systematic and are not easily fixed.

Also common are the occasional sound shorts that radiate noise into the water column and result in poor sonar performance. These occur from improper installation of new pumps or motors, poor maintenance of devices designed to acoustically isolate pumps and other machinery from the hull, or improper stowage of gear around pumps and motors.

While research ships occasionally undergo radiated noise acoustic measurements which might identify sound shorts, they are not common. More frequently the sonar will begin to produce poor, noisy data with little or no warning. While interference of this type may affect a handful of transducers, it is important to remember that the effect won't be seen in a corresponding number of beams. Each transducer contributes to each and every beam. Therefore the increased noise levels will be seen across the entire swath. (Or in the case of a split Port/Stbd array, maybe across half the swath.) In troubleshooting these kinds of symptoms, frequently the last thing considered is an interview of the ship's Captain, First Mate and Engineer for newly installed equipment or other changes that might cause the problem. However it should not be forgotten.

2.3.2. Flow Noise

Turbulent flow in the boundary layer around the hydrophone(s) can result in flow noise that can cause acoustic interference. Careful attention to detail at the design and installation can reduce flow noise.There is a good discussion of flow noise in Chapter 11 of Urick[1983].

2.3.3. Bubble Sweep Down

Bubbles originating at the sea surface and drawn under the hull along flow lines and bubbles that result from cavitation separation are a common problem. Bubbles generate noise which can cause reception problems and they absorb acoustic energy which can cause problems on both transmit and receive.

The effects of bubble sweep down can be reduced by careful attention to detail in the design of transducer installation location and in by minimizing sharp edges and projections that can can result in separation.

2.3.4. Cross Talk

One other form of interference that is worth mentioning, is cross talk. Cross talk occurs in sonars with separate port and starboard transducer arrays and is the effect seen when signal and returns produced from one side are inadvertently detected on the other. Considerable design and thought goes into preventing this, yet, under certain configurations it can still occur. Typically the effect is seen in beams near nadir.

2.4. Signal to Noise Ratio

In the preceding sections we have talked qualitatively about "signal to noise ratio" (SNR), but we have not yet defined it in any formal way. While our goal is to provide only a cursory introduction to acoustics as it applies to sonar systems, it is instructive to look at the equation for SNR, if only briefly.

First we can define a bunch of terms, each of which are represented in decibels (dB):

SL = *Source Level* or the amount of sound energy that we "ping" into the water.

TS = *Target Strength* or the amount of sound energy that is reflected from an object (in our case the sea floor).

PL = *Propagation Loss* or the amount of energy lost to absorption and spreading of sound energy in the water column.

N = *Noise* or the amount of sound energy from other sources including reverberation, other ships, own ship, sea life, you name it. This term also includes electronic noise that is not "acoustic" in origination.

Armed with these definitions we can then write an equation for SNR, but before we do, we could review some quick math so it makes more sense. Remember that these values are expressed in dB, which is the log of the ratio of the sound intensity level to some standard level. Our "ratio" is a ratio of sound intensities, so when writing these as logarithms, to multiply the sound intensities we add the logarithms and to divide the sound intensities we subtract the logarithms. Now for the equation:

SNR = SL - PL + TS - PL - N

Reading left to right this equation makes perfect sense. Our signal starts with the source level from the transducer array (SL). The level is then reduced during propagation to the sea floor target (PL). Some amount of signal is reflected from the target (TS). This target signal is then decreased again by propagation loss back to the transducers (PL). The received signal is then reduced by are ability to discern it from the surrounding noise (N).

Written more compactly:

SNR = (SL + TS) - (2PL + N)

It is helpful to consider this equation when considering the possible sources of poor sonar performance and data quality.

2.5. Swath Mapping Sonar Systems

Blackinton(199_) defines a useful terminology for describing swath mapping sonar systems that produce bathymetric data...

2.5.1. Multibeam Echo Sounders MBES

Something about MBES.

2.5.2. Sidescan Swath Bathymetric Sonars SSBS

Something about SSBS.

Chapter 3. Surveying Your Survey with MB-System™

As a start, we want to demonstrate the how to use MB-System™ to survey a data set, without any special processing. This chapter might by skipped by more experienced users, but it is a good place to start for those new to sonar data processing and MB-System™.

The standard MB-System™ distribution comes with an archive of example data sets and scripts generated using MB-System™ (MB-SystemExamples.X.Y.Z.tar.Z). When uncompressed and un-archived, the resulting directory tree will resemble the lines below.

```
[vschmidt@val-LDEO mbexamples]$ ls
data mbbath mbgrid mbinfo mblist mbm_plot README xbt
```

The `~/data` directory contains several sample data files used in some of the examples in this chapter an the next. The other directories contain scripts that demonstrate the use of several of the tools in MB-System™ .

For the purposes of the MB-Cookbook examples, a directory called `~/mbexamples/cookbook_examples` has been created. While some of the data sets provided in the mbexamples archive will be used in these illustrations, other data sets have been chosen to augment them. These data sets have been placed in `~/mbexamples/cookcook_examples/other_data_sets`.

3.1. Managing Your Data With *mbdatalist*

It is common when working within a survey, to want to plot the whole thing, and then take a closer look at a few particular points of interest. "Lets look at the data around this sea mount!" "How about a plot of yesterday's data." "What about a plot of the old data, with the new data." We can do all this with mbdatalist.

Note

While this chapter covers the processing of a single survey, Chapter 4 will explain how to organize the data from several surveys, or even several cruises, into a larger archive. Managing large data sets of this type is greatly simplified with recursive data lists created by mbdatalist.

We will first use mbdatalist to create a master file list of all the files in the survey. This list will have proper relative references, will contain the MBIO format of the files, and will have an appropriate grid weighting factor.

Note

Weighting, referred to in the paragraph above, is the process of assigning relative weights to collocated data sets, such that only those deemed most accurate or up to date are used for subsequent processing and plotting. This feature will be demonstrated more in Chapter 4: "Processing Multibeam Data".

We will then use mbdatalist to create three ancillary files for each data file. These ancillary files will greatly speed the processing of subsequent tasks. The three files are referred to as "info", "fast bathymetry" and "fast navigation." The info (".inf" suffix") contains meta-data and statistics about its parent data file. The information it contains is, in fact, identical to that created by mbinfo as we will see later.

The fast bathymetry (.fbt suffix) and fast navigation (.fnv) files contain bathymetry and navigation data, respectively, in a format that can be read and processed more quickly than the original swath file format.

We can use mbdatalist to create a geographically windowed list of the files we are interested in.

Finally, we can make some cool, quick plots.

Perhaps we need an example.

In the `~/mbexamples/cookbook_examples/other_data_sets/ew0204survey/` directory you will find a collection of data files recorded with the Atlas Hydrosweep DS2 sonar aboard the R/V Ewing. These files will be used for the following examples.

First we need an initial list of the data files. This list needs to be in the directory that contains the data files themselves. This is done easily enough with the following:

```
cd ew0203survey/
ls -1 | grep mb183$ > tmplist
```

Note

Sometimes the data files are in a write protected archive, that prevents you from just making your own local data list. What do you do then? Generate the file list with the following: find <pathtoarchivedir> -type f | grep mb183$ > tmplist. This will create a file list with relative path names to the data files, which will be carried through the subsequent steps below. Clever eh?

Now that we have a list of the files in our data directory, we can use mbdatalist to create a properly formatted data list for our subsequent MB-System™ processing.

```
mbdatalist -F-1 -I tmplist > datalist-1
```

Let us take a moment to peek inside and see what mbdatalist has done for us.

```
00020504090010.mb183 183 1.000000
00020504091010.mb183 183 1.000000
00020504092010.mb183 183 1.000000
00020504093010.mb183 183 1.000000
00020504094010.mb183 183 1.000000
00020504095010.mb183 183 1.000000
00020504100010.mb183 183 1.000000
...
```

Here we see, for each file, the file name, the format, and a default grid weight of 1. Perfect!

With our new data list in hand, we can go ahead and create the ancillary data files. Using mbdatalist again:

```
mbdatalist -F-1 -I other_data_sets/ew0204survey/filelist.124 -N
```

The data directory now looks like this:

```
00020504090010.mb183
00020504090010.mb183.fbt
00020504090010.mb183.fnv
00020504090010.mb183.inf
00020504091010.mb183
00020504091010.mb183.fbt
00020504091010.mb183.fnv
00020504091010.mb183.inf
00020504092010.mb183
00020504092010.mb183.fbt
00020504092010.mb183.fnv
00020504092010.mb183.inf
...
```

Now that we have the ancillary files created (particularly the statistics file which is used by mbdatalist for geographic windowing) we can create a new file list geographically windowed around a particular area. In this instance, I know from my exemplary notes during the cruise, that a small survey was taken of an area bounded by the following coordinates: (W/E/S/N) 170.133/170.35/42.2/42.4. The following line creates a list of data files within the geographic bounds of interest, with all the proper details required by other MB-System™ tools.

```
mbdatalist -F-1 -I datalist-1 -R170.133/170.35/42.2/42.4 \
 > survey-datalist
```

The resulting `survey-datalist` looks something like this:

```
00020504100010.mb183 183 1.000000
00020504101010.mb183 183 1.000000
00020504102010.mb183 183 1.000000
00020504103010.mb183 183 1.000000
00020504104010.mb183 183 1.000000
00020504105010.mb183 183 1.000000
....
```

As you can see, each data file is listed appropriately, each format type (183) is specified, and the default weighting factor of 1 is included as well.

Now lets make some plots.

3.2. Plotting Data

Now that we've got a list of the data files of interest, we want to make some quick plots to see what we've got. These will not be the prettiest plots, as none of the data has been edited. None-the-less, MB-System™ will create nicely formatted plots, even if the data is a little unpolished.

First, perhaps we'd like to look at just the navigation data and see the ship track for this survey. To do that we call mbm_plot as shown below:

```
mbm_plot -F-1 -I survey-datalist -N
```

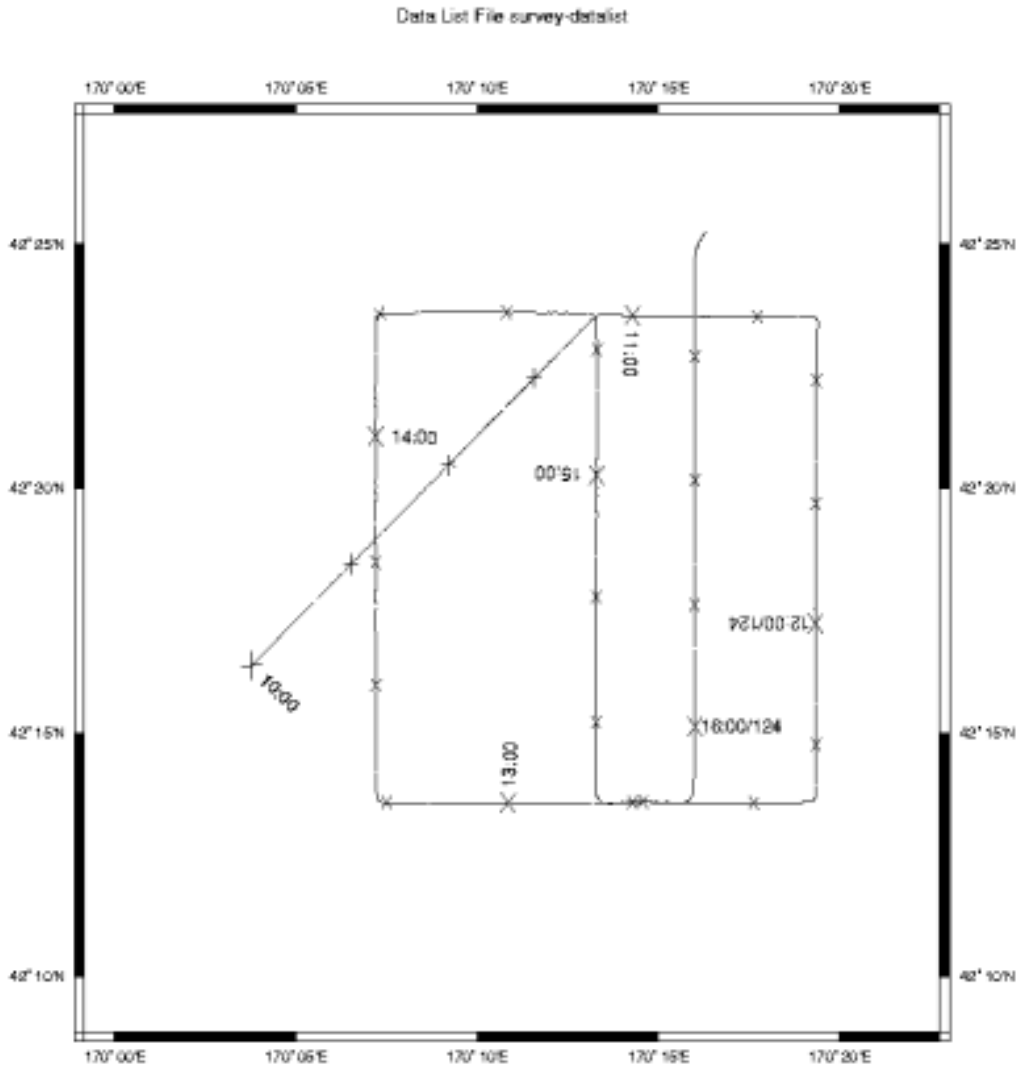
Here, *-F-1* specifies the format, in this case "-1" indicates that the input is a list of files rather than an individual file and that the format for each file is specified in the list. Of course, *-I survey-datalist* is our list of data files. A navigation plot is specified with *-N*. With this simple line, and nothing more, we get a script that will create a navigation plot with default annotations, grid lines, tick marks, etc.. The results

of executing the line above are shown below.

```
Plot generation shellscript <survey-datalist.cmd> created.

Instructions:
  Execute <survey-datalist.cmd> to generate Postscript plot
<survey-datalist.ps>.
  Executing <survey-datalist.cmd> also invokes ghostview to
  view the plot on the screen.
```

A script was created called `survey-datalist.cmd`. When executed the script will create the navigation plot as a post script file and then execute ghostview to view the plot immediately. When we execute the resulting script the following plot is created:

Figure 3.1. Survey Navigation Plot



Wasn't that easy? mbm_plot sets default tick marks, grid lines, and longitude and latitude annotations as well as track annotations. It takes care of centering your plot onto a single page and even adds a title. All of these details, of course, can be changed though the myriad of options to mbm_plot. Have a look at the man page for the details.

Note

For the remaining examples, only the initial mbm_plot line and the plot created from execution of the subsequent script will be shown, for the sake of brevity.

Now let us take a look at the bathymetry data itself. We might start with a color plot of the bathymetry data. One specifies plotting of the bathymetry data by indicating a "graphics" mode utilizing the "-G" flag to mbm_plot. Five graphics modes are supported:

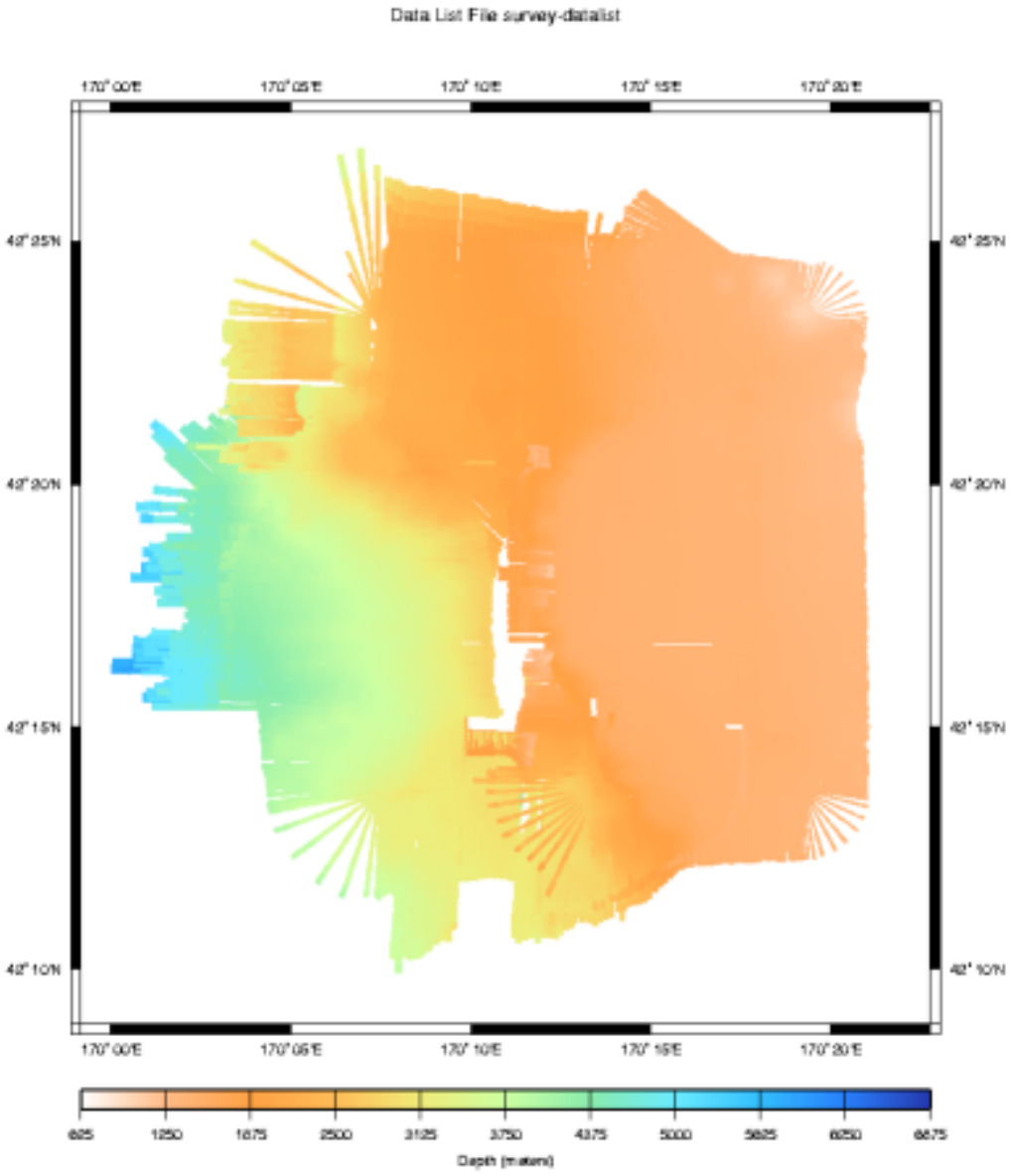
- mode = 1: Color fill of bathymetry data.
- mode = 2: Color shaded relief bathymetry.
- mode = 3: Bathymetry shaded using amplitude data.
- mode = 4: Grayscale fill of amplitude data.
- mode = 5: Grayscale fill of sidescan data.

Then to create a color fill of the bathymetry data, we execute the following:

```
mbm_plot -F-1 -I survey_filelist.124 -G1
```

Here is the resulting plot.

Figure 3.2. Color Bathymetry Plot

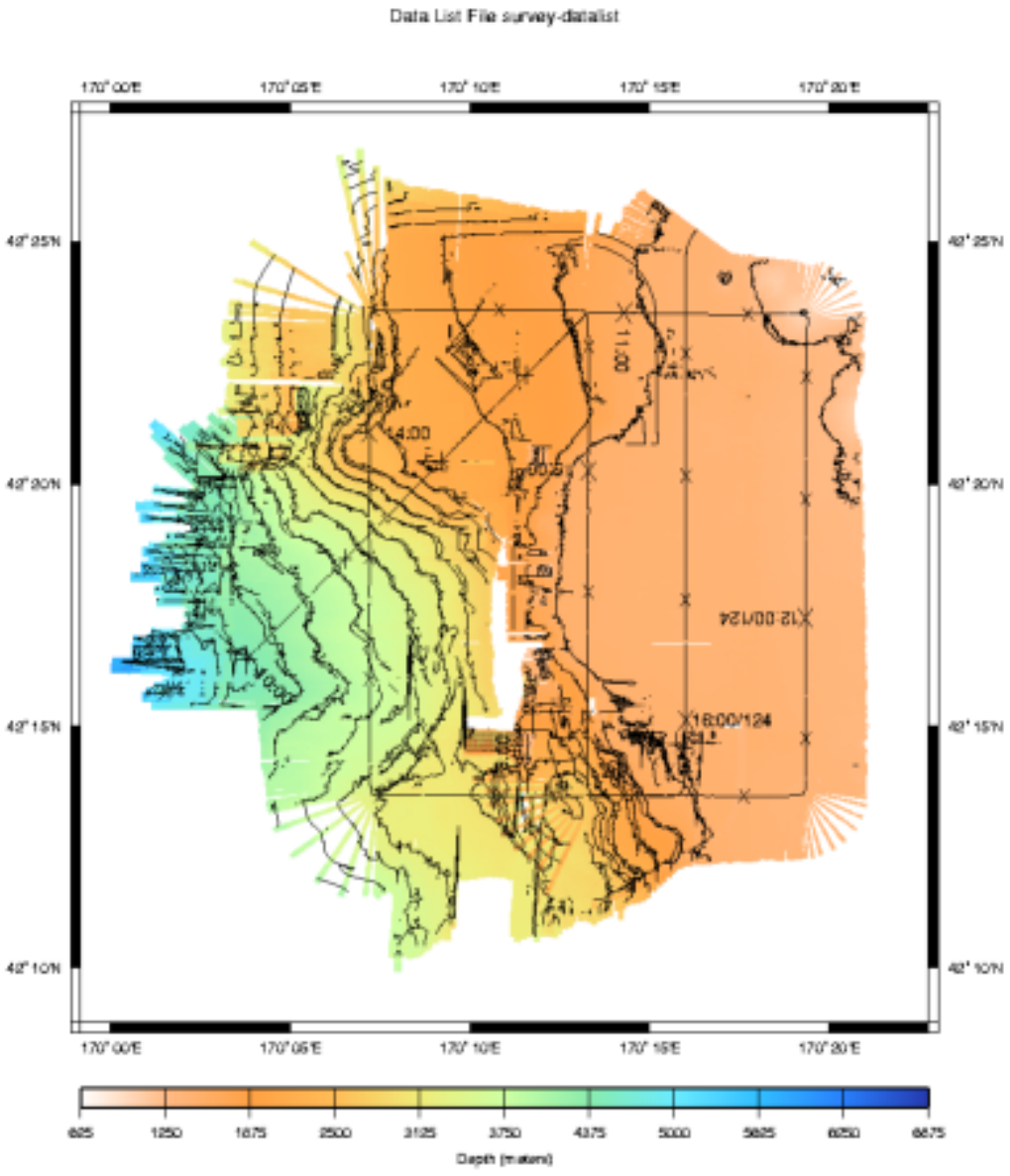


Now perhaps we'd like to add some contours to our color plot, and maybe add the navigation back in as well. A quick contour plot with default parameters can be specified with the "-C" flag.

```
mbm_plot -F-1 -I survey_filelist.124 -G1 -N -C
```

The resulting plot is shown below:

Figure 3.3. Color Bathymetry Plot with Contours



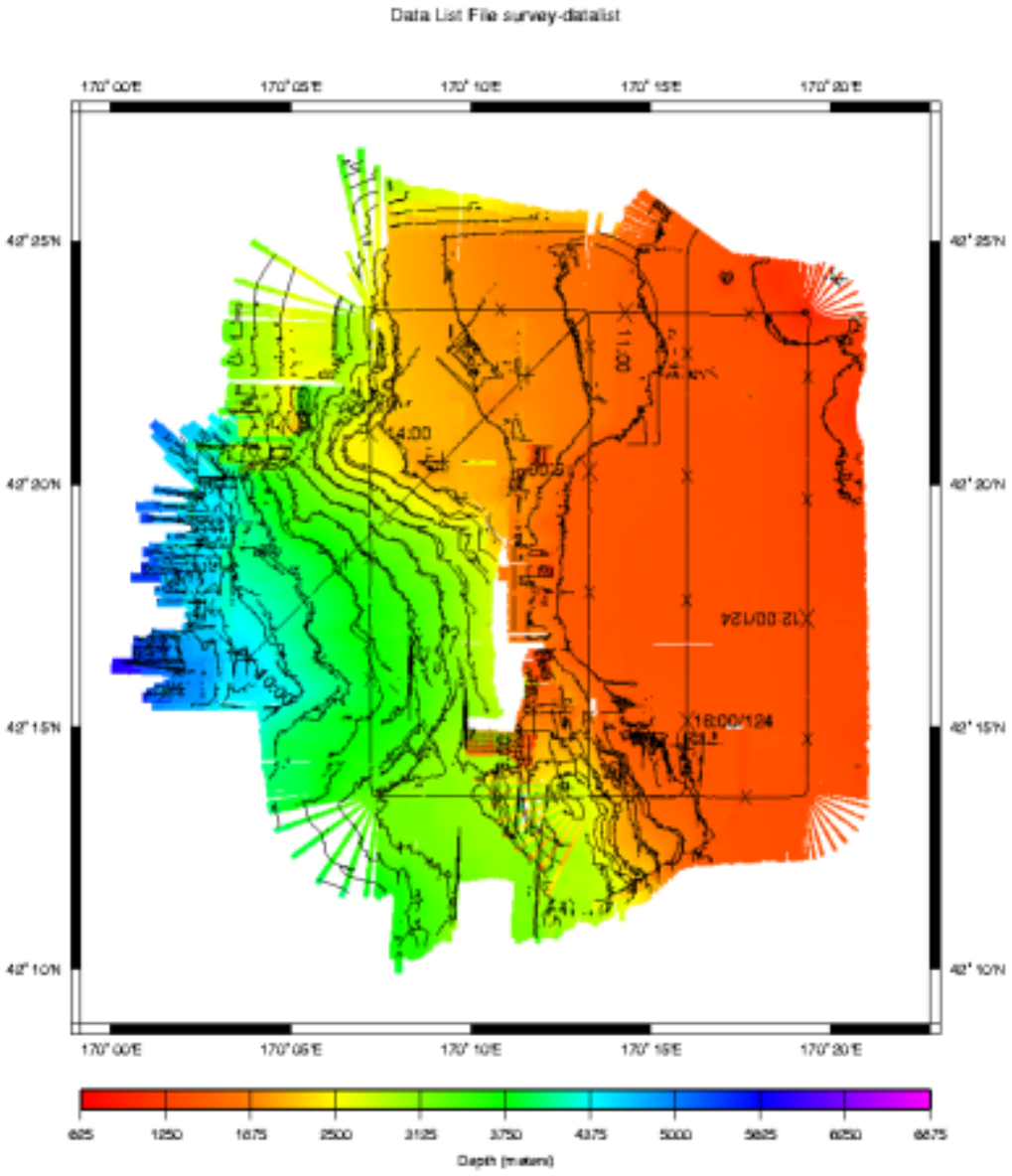
What a nice plot of the data set! We can quickly see the lay of the sea floor, maximum and minimum contours, the ship track's coverage, and oddly shaped contours here and there already give hints as to where we we might need to spend special attention in our data editing.

Just for fun, lets create the same plot with a different color scheme. mbm_plot has a few predefined palettes that do not require us to create our own color map. These can be specified with the "-W" flag followed by the color style (continuous or discrete intervals), and optionally the palette (1-5) and number of colors(11 default).The following will create the same plot with mbm_plot's "high intensity" color palette.

```
mbm_plot -F-1 -I survey_filelist.124 -G1 -N -C -W1/2
```

Here's the resulting plot:

Figure 3.4. High Intensity Color Bathymetry Plot with Contours

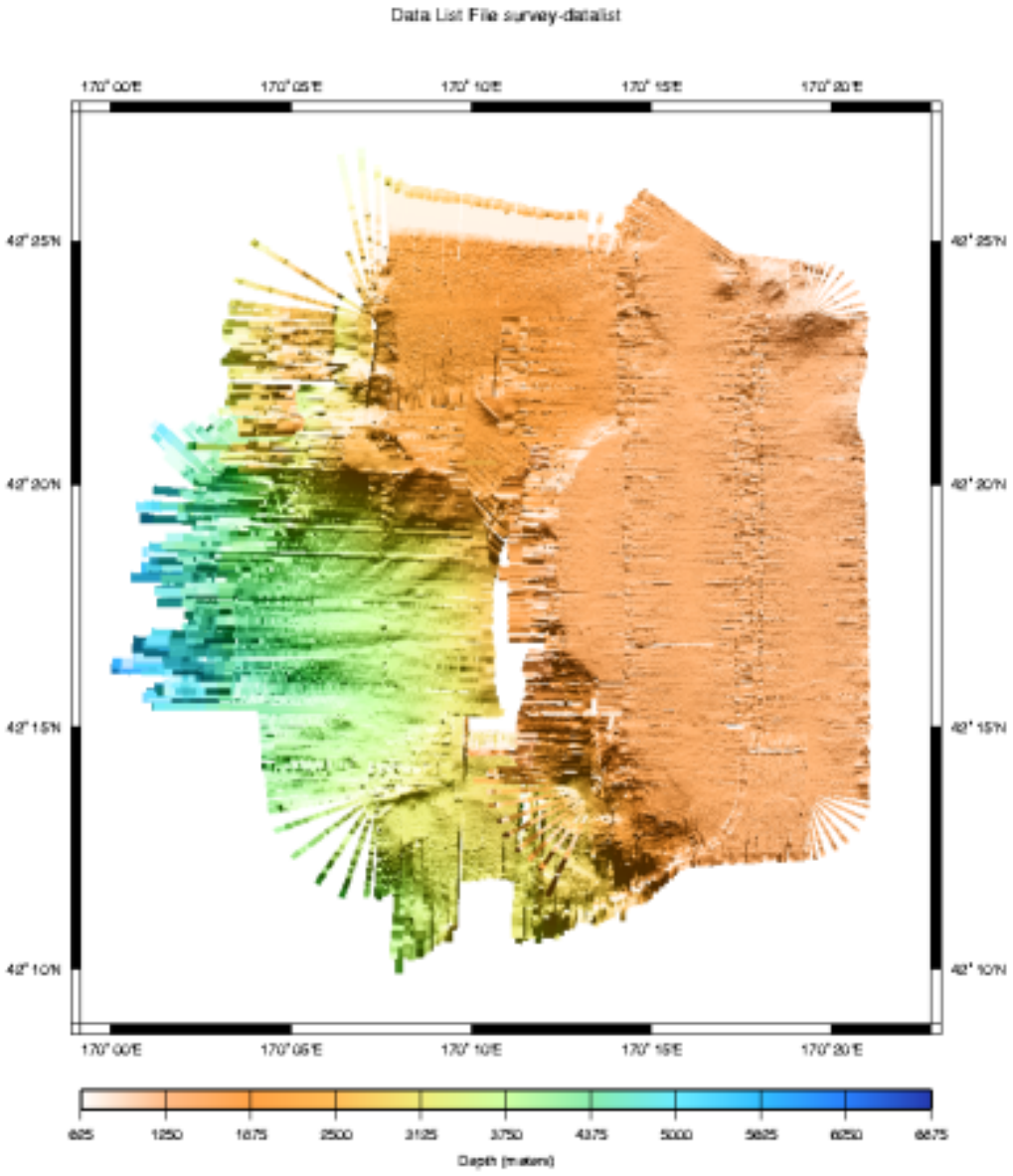


mbm_plot can also quickly create shaded relief maps. These are specified with graphics mode "2" listed above, and by default produce a plot with artificial illumination from the north. The direction of illumination may be modified with the *-A* flag. See "COMPLETE DISCRIPTION OF OPTIONS" in the man page for more details.

```
mbm_plot -F-1 -I survey_filelist.124 -G2
```

Here's the resulting plot.

Figure 3.5. Shaded Relief Color Bathymetry Plot

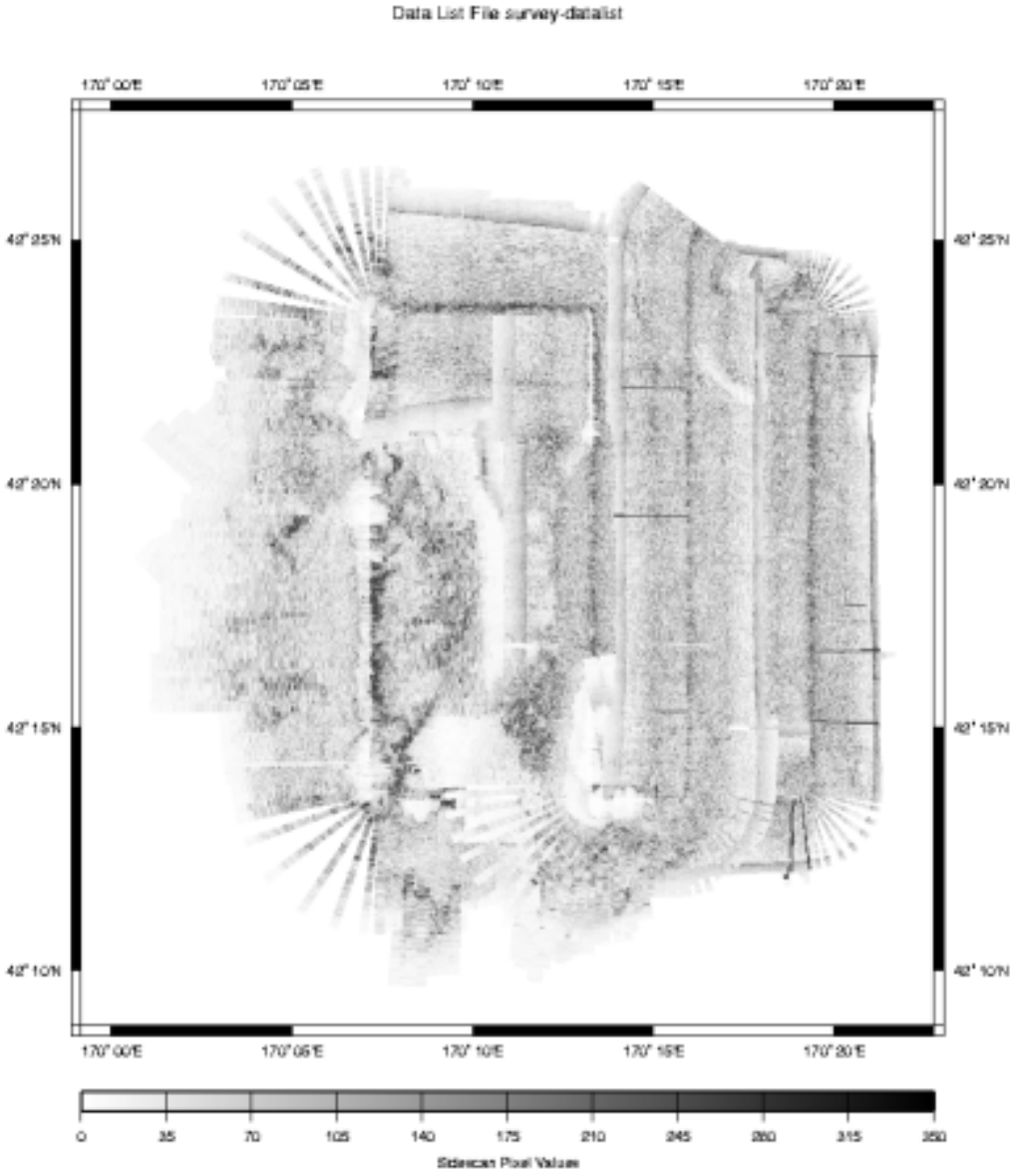


Finally we might wish to see a plot of the side scan data. Sidescan plotting is also specified by a graphics mode, in this case mode "5".

```
mbm_plot -F-1 -I survey_filelist.124 -G5
```

Here is the plot:

Figure 3.6. Sidescan Plot



The plot above is not as nice as we might like. Much of shading is lost on erroneously high sidescan values which have incorrectly weighted the grayscales. The result is a fuzzy, washed out side scan image. This particular data set covers such a large range of side scan values (due to a change of several 1000's of meters in depth plus normal noise) that the side scan image looks quite poor. In this case, we will have to edit the data to remove the erroneous data spikes and then renormalize the gray scaling to get a good side scan image. We leave that to another chapter.

Now that we've explored the data sets graphically one might be interested to extract some statistics about the data set. This leads us to the next section.

3.3. Extracting Statistics

MB-System™ provides mbinfo as a simple tool to extract statistics about a data set. Actually we've already used mbinfo and didn't know it. It was called in the background on our first call to mbdatalist to create the ".inf" files for each data file. However we could have executed mbinfo on each file individually as shown below.

```
mbinfo -F 183 -I 00020504090010.mb183
```

The results are sent to STDOUT by default rather than an ".inf" file. One can optionally redirect the results to a file of the same name as the original with ".inf" appended to the end with the *-O* flag. The resulting file will be automatically read and utilized by several of the other MB-System™ processes.

Here is the result of executing the mbinfo line above.

```
Swath Data File:      00020504090010.mb183
MBIO Data Format ID:  183
Format name:          MBF_HSDS2LAM
Informal Description: L-DEO HSDS2 processing format
Attributes:           STN Atlas multibeam sonars,
                      Hydrosweep DS2, Hydrosweep MD,
                      Fansweep 10, Fansweep 20,
                      bathymetry, amplitude, and sidescan,
                      up to 1440 beams and 4096 pixels,
                      XDR binary, L-DEO.

Data Totals:
Number of Records:              37
Bathymetry Data (140 beams):
  Number of Beams:            5180
  Number of Good Beams:       5106     98.57%
  Number of Zero Beams:         74      1.43%
  Number of Flagged Beams:       0      0.00%
Amplitude Data (140 beams):
  Number of Beams:            5180
  Number of Good Beams:       5106     98.57%
  Number of Zero Beams:         74      1.43%
  Number of Flagged Beams:       0      0.00%
Sidescan Data (2180 pixels):
  Number of Pixels:          73815
  Number of Good Pixels:     55276     74.88%
  Number of Zero Pixels:     18539     25.12%
  Number of Flagged Pixels:      0      0.00%

Navigation Totals:
Total Time:            0.1593 hours
Total Track Length:    3.4493 km
Average Speed:        21.6521 km/hr (11.7038 knots)
```

```
Start of Data:
Time:  05 04 2002 08:59:56.460000  JD124
Lon:  169.8792     Lat:   42.1343     Depth:  5057.7613 meters
Speed: 23.1530 km/hr (12.5151 knots)  Heading:  46.1206 degrees
Sonar Depth:    5.8000 m  Sonar Altitude: 5051.9613 m

End of Data:
Time:  05 04 2002 09:09:29.968000  JD124
Lon:  169.9084     Lat:   42.1563     Depth:  4950.8452 meters
Speed: 21.0345 km/hr (11.3700 knots)  Heading:  43.9014 degrees
Sonar Depth:    5.1000 m  Sonar Altitude: 4945.7452 m

Limits:
Minimum Longitude:     169.8268   Maximum Longitude:     169.9605
Minimum Latitude:       42.0966   Maximum Latitude:       42.1970
Minimum Sonar Depth:     4.6000   Maximum Sonar Depth:     6.7000
Minimum Altitude:     4817.2718   Maximum Altitude:     5051.9613
Minimum Depth:        4770.9467   Maximum Depth:        5347.7265
Minimum Amplitude:       7.0000   Maximum Amplitude:     241.0000
Minimum Sidescan:        2.0000   Maximum Sidescan:      255.0000
```

We could also have executed mbinfo on a list of data files to see summary statistics for all of them. As an example we'll use our survey list created earlier:

```
[vschmidt@val-ldeo cookbook_examples]$ mbinfo -F-1 -I survey-datalist
```

And the results...

```
...
Data Totals:
Number of Records:             2412
Bathymetry Data (140 beams):
  Number of Beams:           337680
  Number of Good Beams:      329580     97.60%
  Number of Zero Beams:        8100      2.40%
  Number of Flagged Beams:        0      0.00%
Amplitude Data (140 beams):
  Number of Beams:           337680
  Number of Good Beams:      329580     97.60%
  Number of Zero Beams:        8100      2.40%
  Number of Flagged Beams:        0      0.00%
Sidescan Data (4094 pixels):
  Number of Pixels:         8746103
  Number of Good Pixels:    7680282     87.81%
  Number of Zero Pixels:    1065821     12.19%
  Number of Flagged Pixels:       0      0.00%

Navigation Totals:
Total Time:             7.0001 hours
Total Track Length:   128.0750 km
Average Speed:         18.2963 km/hr ( 9.8899 knots)

Start of Data:
Time:  05 04 2002 09:59:44.485000  JD124
Lon:  170.0626     Lat:   42.2726     Depth:  4508.8810 meters
Speed:  0.0000 km/hr ( 0.0000 knots)  Heading:  44.6704 degrees
Sonar Depth:    6.0000 m  Sonar Altitude: 4502.8810 m

End of Data:
```

```
Time:  05 04 2002 16:59:44.722000  JD124
Lon:  170.2718     Lat:   42.4209     Depth:  1446.6716 meters
Speed: 21.4741 km/hr (11.6076 knots)  Heading:  35.9473 degrees
Sonar Depth:    5.4000 m  Sonar Altitude: 1441.2716 m

Limits:
Minimum Longitude:     170.0040   Maximum Longitude:     170.3616
Minimum Latitude:       42.1619   Maximum Latitude:       42.4469
Minimum Sonar Depth:     3.6000   Maximum Sonar Depth:     7.9000
Minimum Altitude:      990.7620   Maximum Altitude:     4740.1799
Minimum Depth:         973.2481   Maximum Depth:        6223.7055
Minimum Amplitude:       0.0000   Maximum Amplitude:     250.0000
Minimum Sidescan:        1.0000   Maximum Sidescan:      255.0000
```

Here mbinfo has created summary statistics for the entire survey . We can see that the survey lasted some seven hours covering some 128 km at an average ship speed of about 10 knots. We can see the number of beams recorded, and had any processing been done on these data files, we would see the number of beams that had been flagged for removal. We can also see the data bounds, in latitude, longitude, minimum and maximum water depth, and start and stop times.

Now that we've seen how to quickly make some nice plots and extract some basic meta-data about our data files, we can turn to the topic of processing that data.

Chapter 4. Processing Multibeam Data with MB-System™

Now that we have seen how one can quickly survey a data set, both graphically and statistically, we now turn to the task of post processing sonar data with MB-System™.

4.1. Overview

4.1.1. Many Types of Sonar

MB-System™ supports many types of sonar, each of which as its own data file format. The details of processing data from each are slightly different, however, in general, the core set of steps is the same. We have attempted to capture this core set of steps in the ensuing chapter.

That said, we should quickly note that the devil is in the details. Sonar data formats, vary between sonar models - even from the same vendor. Occasionally the same model sonar differs from one ship installation to the next. In as much as possible, MB-System™ rides atop an input/output layer, MBIO, which handles these differences in the form of *format identifiers* as we have described. However it is a constant struggle to cater to the evolving needs of the scientific and sonar development communities, and in some instances special processing techniques are required. We have added an appendix entitled "Shipboard Multibeam Sonar Installations" in an attempt to document the varied idiosyncrasies of particular sonar systems we know about and, when appropriate, provide hints regarding how to handle their data.

4.1.2. Strategy

The processing strategy is to first survey the data and organize it in some meaningful way. Then, when they have not been provided for us by the survey ship, we calculate physical constants such as roll and pitch bias of the sonar system from sections of our own carefully taken data. Next we conduct automated and manual editing of the data to remove unwanted artifacts and erroneous segments. We edit the navigation data to ensure continuity and "reasonableness". We identify high quality sound speed profiles for regions in the data set. For sonars that produce sidescan data, we generate amplitude and grazing angle tables. Finally we apply all our physical constants, bathymetry and navigation editing changes, sound speed profiles and grazing angle tables to the data to create the final, processed, data set.

Rather that creating intermediate incremental files with each processing step, changes are tabulated in a processing parameter file. These changes are applied in the last step to create the final post processed data file. The next section is describes the parameter file in detail.

Interactive tools provided by MB-System™ utilize the editing already recorded in the parameter file to display the current state of the edited data. In this way, while the final processed file is not produced until the final step, one always works with the most recent changes applied. Because the changes are not irrevocably applied to the data files, one can conduct these steps in any order or return to any intermediate step and make changes.

A summary of the processing steps might look like the following:

1. Determine the make and model of the sonar that recorded the data as well as the format of the data. The combination of these two will map to an MB-System™ format type, explicit delineation of which is required in many of the subsequent steps.

2. Survey the data to get an idea of its coverage and quality. Organize the data files into directories and create data lists to aid in processing. Also make a qualitative assessment of the quality of the

navigation data.

3. Determine the sonar system characteristics, including the roll and pitch bias. Typically these values are measured periodically and kept on record by the ship's science officer, however it is never a bad idea to verify their values through independent measurement. It is a common misconception that these values do not change over the life of a ship or sonar.

4. Identify bathymetry data that is erroneous. This is a several step process using both automated and interactive techniques to identify processing errors in the data and data plagued by excessive noise or other interference.

5. SSP information is assumed by the sonar when calculating bathymetry from its measurements. The SSP used by the sonar may come from historical data of the survey area, direct measurements via a CTD or an XBT, or may be pulled out of thin air by the science party or ship's crew. The next step is to identify the correct SSP 's for the data set. Rarely is a single SSP sufficient for a survey. The SSP should be reevaluated on periodic intervals, with changes in bodies of water, and when the quality of the calculated data indicates a change in the SSP .

6. Smooth incongruities from the navigation data.

7. For sidescan data, determine amplitude and grazing angle functions.

8. Recalculate bathymetry and sidescan data from the system constants, SSP(s), smoothed navigation, amplitude and grazing angle functions and flagged data errors to produce a final processed data file.

Remember that each of these steps are autonomous in that one may go back and recalculate, for example, the system's roll bias, creating a revised parameter file which may be then used to reprocess the original data without recalculating any of the other steps.

The following sections illustrate each of the steps above, providing examples and results along the way. While most of the examples utilize a single data set for continuity, we have included other data sets in the sound speed profile and roll bias discussions. We hope no loss of clarity results.

From Section 1.5, you will recall that these examples will primarily utilize datasets mapping the volcanic seamount Lo'ihi off the southern coast of the Big Island of Hawaii.

Now on to the processing!

4.2. Identifying the MB-System™ Data Format

The first step in processing sonar data files with MB-System™ is to identify the type of sonar that was used to record the data and the format the data files are in. MB-System™ uses this information not only for successfully reading from and writing to the data files, but behind the scenes details regarding these sonars are used in reprocessing the data files.

Hopefully this information has been recorded by the science party when the data was taken and is readily available. However at times those with the task of post processing the data are not always the ones who participated in the cruise.

MB-System™ uses a standard set of libraries for reading sonar data files. MB-System™ version 5 supports the following sonar systems, among others:

- SeaBeam "Classic" 16 Beam Multibeam Sonar
- Hydrosweep DS 59 Beam Multibeam Sonar

- Hydrosweep MD 40 Beam Mid-depth Multibeam Sonar
- SeaBeam 2000 Multibeam Sonar
- SeaBeam 2112, 2120, and 2130 Multibeam Sonars
- Simrad EM12, EM121, EM950, and EM1000 Multibeam Sonars
- Simrad EM120, EM300, EM1002, and EM3000 Multibeam Sonars
- Hawaii MR-1 Shallow Tow Interferometric Sonar
- ELAC Bottomchart 1180 and 1050 Multibeam Sonars
- ELAC/SeaBeam Bottomchart Mk2 1180 and 1050 Multibeam Sonars
- Reson Seabat 9001/9002 Multibeam Sonars
- Reson Seabat 8101 Multibeam Sonars
- Sim-rad/Mesotech SM2000 Multibeam Sonars
- WHOI DSL AMS-120 Deep Tow Interferometric Sonar
- AMS-60 Interferometric Sonar

MB-System™ supports several file formats unique to each sonar type - more than can be easily listed here. However they are all clearly delineated in the man page for *MBIO* allowing one to easily determine the MB-System™ format for the specified sonar and file type.

You can use the mbformat tool provided by MB-System™ to determine the sonar data Format ID. Each sonar writes data files with standard naming conventions. mbformat, typically uses these naming conventions to identify the sonar data format from which it looks up the numerical *Format ID*.

In some cases, the naming convention doesn't provide enough clues to determine the correct format ID. For example, Simrad recently changed their sonar data format, although their file naming convention is the same (*.raw). mbformat must open these files and begin to read the data to determine if the correct format ID is 51 - for the old format, or 56 - for the new format.

If the files are renamed with some naming conventions other than that provided by the sonar system or the standard MB-System™ convention, mbformat will not recognize them. In this case one must know details regarding the sonar system and sift through the MBIO man page for the appropriate format.

For example, suppose it is known, from notes provided with the data set, that the data was taken using the SeaBeam™ - 19 beam sonar on the R/V Thomas. I might further know that the data is bathymetry only (no sidescan) and is stored in binary format. I could then look in the man page for *MBIO* and see the following:

```
...
MBIO Data Format ID:  16
Format name:          MBF_SBSIOSWB
Informal Description: SIO Swath-bathy SeaBeam
Attributes:           Sea Beam, bathymetry, 19 beams,
                      binary, centered, SIO.
...
```

There we have it - "MBIO Data Format ID: 16".

4.3. Format Conversion

4.3.1. Background

Many sonar systems provide data in unwieldy or un-editable formats. Hydrosweep DS2™ data in its raw format, produces no less than 9 data files for each 10 minute segment. Simrad data files encode navigation information at non-periodic intervals within the data such that there is not a specific position associated with each ping. Some sonars do not embed automated and or manual comments into their data files. Some do not have data structures to allow the flagging of beams for editing to occur. To cope with these differences MB-System™ provides for the conversion the the "raw" data formats into one of several alternative formats defined by MBIO . These formats are native only to MB-System™ but provide, in many cases, the only way to manage, manipulate, and process the data.

Since each sonar vendor format is somewhat different, we are forced to provide specific instructions for each. While not all raw sonar data formats supported by MB-System™ are discussed here, most will fall into one of the following sections.

Note

Typically, vendor provided sonar data viewing and editing software will not be able to read the MB-System™ -only formats. If there is a chance that you will need the data in its raw format, it is best to keep backup copies of the raw data before converting it. In the case of Hydrosweep DS2™ data, the nine files are such a burden that most ships convert the data to an alternate format immediately as an automated process and provide only the condensed MB-System™ file format to scientists unless the raw data is requested.

4.3.2. SeaBeam 2100

Lucky for us, SeaBeam 2100 series multibeam sonars provide data in an amenable format right out of the box. This format is MB-System™ format 41. Typical raw data files however are named with a date-time stamp followed by ".rec". All that is required is a renaming of the files to "date-time_stamp.mb41" to facilitate operations with MB-System™.

4.3.3. Converting Your Data

If it hasn't been done already, early on in the post processing we will want to convert our sonar data to one of the condensed MB-System™ data formats. This is done with the mbcopy process. mbcopy reads data files of a specific format, optionally windows them in time or space, may average specified numbers of pings, adds comments specifying when and by whom the data was converted (if the output format supports comments), and finally converts the data to a new MB-System™ format. mbcopy can read and write from STDIN and to STDOUT making it ideal for real-time automated processes. Alternatively, it may read and write to and from specific files. The general form of mbcopy with no ping averaging looks something like the following:

```
mbcopy -F182/183 -I 00011209011010.fsw -O 00011209011010.mb183
```

Here the input and output file formats are specified (182/183), as well as the names of the files from which to read and write (-I 00011209011010.fsw -O 00011209011010.mb183). Notice that we've renamed the file with MB-System™ 's standard naming conventions of .mb*ID#*. This will allow interactive tools such as mbedit to automatically identify the data format.

At the time of this writing, mbcopy did not yet have a built-in way to convert multiple data files. Therefore I've provided a short script I call multi-copy that can be quite helpful with larger data sets.

```
#!/usr/bin/perl
#
#
# Usage: ls -1 | multi-copy
#
# Val Schmidt
# Lamont Doherty Earth Observatory
#
# This script takes a directory full to 182 format multi
# beam files and converts them to 183 format files.

while (<>){
        if (/.*fsw$/){
            chomp;
            s/\.fsw//;
            $cmd="mbcopy -F182/183 -I$_"."\.fsw"." -O".$_."\.mb183";
            #print $cmd;
            system($cmd);
        }
    }
```

This script works as it is to convert Hydrosweep DS2™ data in the 182 format to the 183 format. You'll have to modify it to suite your particular data format and naming conventions. For example, raw Hydrosweep files end in "fsw", so I screen the file list with the statement "if (/.*fsw$/)". If your files end in something else you'll have to edit that line and the others appropriately.

Before we move on, we should take am moment to discuss some other details regarding translation between data formats. With few exceptions, mbcopy will translate between any two data formats - even to those of other sonars. When the input and output formats are associated with the same sonar system the complete data stream is copied to the output file. When they are not, however, only the time stamp, navigation, and beam values are copied. This allows conversion of the bulk of the data, even though some formats do not make provisions for comments or other details in the data stream.

Note

Note however, typically data that has been edited cannot be translated back into the raw sonar format to be further viewed/edited in their proprietary software.

You will find that the data files in our example data set, (`loihi/MBARI1998EM300`) have already been converted to MB-System™ Format ID 57.

AUTHOR'S NOTE: THERE IS MORE TO BE ADDED HERE REGARDING THE DETAILS OF INDIVIDUAL FILE AND SONAR TYPES. INVESTIGATE THE DETAILS SURROUNDING PARTICULAR SONAR SYSTEMS AND DOCUMENT THEM HERE (I.E. WHAT SUPPORTS EDITING, WHEN IT WILL BE LOST, WHAT'S RECOMMENDED FOR EACH SONAR TYPE, ETC.)

4.4. Survey and Organize the Data

With the MB-System™ Format ID identified and the files copied to a manageable format, the next step is to take a look at the data. The things we want to accomplish are:

- Organize the data set into a larger body of data you may already have.
- Organize the data set internally, identifying specific areas that may require special processing.
- Get a qualitative feel for the data to better understand what we are up against. Is the data generally free of errors or are there frequent outliers in the contours? Are there any longitudinal striations that

might be caused by excessive ship noise? Do the outer beams exhibit depth excursions that might be indicative of large SSP errors? The idea is not to conduct an exhaustive examination, but to pick a few files through the data set to get a rough feel for how things look.

- Extract the Latitude and Longitude as well as depth bounds of the data set. These values aid in future calculations and provide a framework for splitting up the processing of the data into chunks based on similar environmental conditions.

- Evaluate the navigation information embedded in the data set. Is the navigation information smooth, or are there frequent outliers? Do data sets measured over the same area result in similar bathymetry, or are they offset due to navigation errors?

4.4.1. Integrating Your Data into a Larger Set

Often research institutions may maintain an single archive of several multibeam data sets from various cruises and various surveys. We might quickly consider how one might organize such an archive, to ease the task of processing and data retrieval.

4.4.1.1. The Method

Consider our example Lo'ihi data set. The top level directory of the Lo'ihi archive provided is entitled `loihi`. This is actually an unfortunate name on our part. Data sets should not be organized based on geographic location, as we shall explain. It might have been more appropriately entitled `multi-beam_archive`. But for the time being we'll stick with `loihi`.

Organization below the top level directory can be quite varied, however should follow a single axiom - *Data sets should not be organized by parameters that are easily managed by MB-System*™. Two particular examples come to mind: time and space.

Managing data sets manually, by the time or location, is difficult, if not impossible to do. A common way to organize data sets is by cruise. (Alternatively, some institutions will organize data sets by principle investigator to allow administrative control over "ownership" and release of the data.) Cruises can be further broken down into segments, such that data is grouped in portions that require like processing or are simply in smaller chunks.

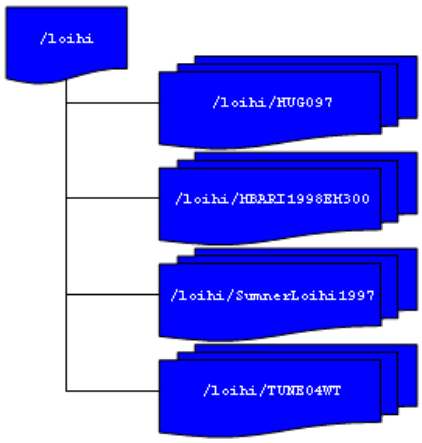
An example follows with segment directories shown for just one of the cruises.

```
archive/ewing_0101
archive/ewing_0101/segment_001
archive/ewing_0101/segment_002
archive/ewing_0101/segment_003
archive/ewing_0102
archive/ewing_0103
archive/knorr_0101
archive/knorr_0102
archive/knorr_0103
archive/knorr_0104
archive/thom_0101
archive/thom_0102
archive/thom_0103
```

The granularity of the file system within the archive will differ greatly from implementation to implementation, but the general idea is the same. With the hierarchy of directories and data files in place, we then create a series of recursive *datalists* within each directory.

Now consider our example data. Within our top archive directory, `loihi` we have subdirectories for each cruise. We might have broken these down further into segments if the data warranted, but in this case it did not. A diagram of the structure and a listing of the loihi directory are provided below.

Figure 4.1. Lo'ihi Archive Data Structure




```
>ls -1 loihi
datalist.mb-1
datalistp.mb-1
HUGO97
MBARI1998EM300
SumnerLoihi1997
TUNE04WT
```

Note

For the moment disregard the file `datalistp.mb-1. This data list will be used to specify that processed files are to take precedence over their unprocessed originals. The utility of this file will be covered in detail later on.`

In the `loihi` directory we create the archive data list - `datalist.mb-1`. This file is a text file containing the relative path and name of each of the data lists in the successive lower level subdirectories. With each list file name is a special MB-System™ Format ID ("-1") denoting that formats will be specified in the lower recursed data lists. Also included a data weighting factor or "grid weight". The con-

tents of `datalist.mb-1` are shown below:

```
>cat datalist.mb-1
HUGO97/datalist.mb-1 -1 100.0
MBARI1998EM300/datalist.mb-1 -1 1.0
SumnerLoihi1997/datalist.mb-1 -1 0.1
TUNE04WT/datalist.mb-1 -1 0.001
```

Before we continue, consider the weighting factors above. The grid weighting factor is a way to merge multiple data sets over the same geographic area and prioritize which data will be extracted for display. Larger weighting factors take priority over smaller ones. While occasionally mitigating factors affect individual sonar performance, usually the difference in sonar quality between sonar vendor and even by vendor models are quite significant. Therefore, weighting data sets by "sonar model" is often a good practice. One might further refine the weighting factor by sonar model and ship, since differences in ship size and construction can significantly affect sonar data results. Increments between weights should be large to allow further refinement without added hassle. Toward that end, some organizations weight their datasets logarithmically as we have done here.

If we now consider the lower level data lists we see the format is similar, except now, instead of referring to lower level data lists, we refer directly to the data files themselves. For example, `MBARI1998EM300/datalist.mb-1` contains a list of the data files and their format:

```
cat MBARI1998EM300/datalist.mb-1
mbari_1998_53_msn.mb57 57
mbari_1998_54_msn.mb57 57
mbari_1998_55_msn.mb57 57
mbari_1998_56_msn.mb57 57
mbari_1998_57_msn.mb57 57
mbari_1998_58_msn.mb57 57
mbari_1998_59_msn.mb57 57
mbari_1998_60_msn.mb57 57
mbari_1998_61_msn.mb57 57
```

Here no grid weight is included, although we could have specified one if a further level of granularity was required.

Note

The default behavior is for any grid weight in a particular datalist entry to override values derived from higher levels in the recursive structure. This behavior can be reversed if the text `$NOLOCALWEIGHT` is inserted in the data list (or in a datalist higher up in the structure).

In our Lo'ihi archive, we have only two layers of subdirectories, however we could have more. Recursive data lists would be created in a similar way, referring down through the structure until the actual data files are referenced. Typically upper level data lists are maintained manually, as they are small and easy to manipulate. However, in the lowest levels, creating a data list for a directory of hundreds of files can be cumbersome. mbdatalist can be used to generate these lists.

As an example, we'll create another data list for the `MBARI1998EM300` data directory. First we need a text file with a list of the data files in the directory. To do that execute the following:

```
ls -1 | grep mb57 > list
```

We can then use this list with mbdatalist to create a data list as follows:

```
mbdatalist -F-1 -Ilist
mbari_1998_53_msn.mb57 57 1.000000
mbari_1998_54_msn.mb57 57 1.000000
mbari_1998_55_msn.mb57 57 1.000000
mbari_1998_56_msn.mb57 57 1.000000
mbari_1998_57_msn.mb57 57 1.000000
mbari_1998_58_msn.mb57 57 1.000000
mbari_1998_59_msn.mb57 57 1.000000
mbari_1998_60_msn.mb57 57 1.000000
mbari_1998_61_msn.mb57 57 1.000000
```

The results of the above command are sent to STDOUT, however the could have been redirected to a text file named datalist.mb-1 and we would essentially have the list we looked at earlier. We have now added a default grid weight of 1. Remember, the weights assigned here take precedence over those assigned higher in the recursion structure.

Note

Note that mbdatalist does not currently provide the ability to specify a weighting factor other than the default. It is primarily a data list parsing tool as we shall see in future examples. One can either manually specify the weight at a higher level in the data list recursion structure, or adjust the default grid weight via a simple shell script. The following example changes the default value to 100.

```
cat datalist_old | sed 's/1\.000000/100/' > datalist_new
```

Finally we should remove our interim list

```
rm list
```

4.4.1.2. The Benefit

So what is the benefit of having our data in this archive, with all these recursive data lists?

Suppose you were interested in the highest quality data you have covering Lo'ihi and the surrounding area. You can now quickly make a mosaic of the various data sets, simply by referring to the archive data list rather than sorting through a myriad of data files to decide the best quality data to then refer to the files directly. Where two data sets are collocated, the data with the highest grid weight will be extracted.

As we continue with data processing, we'll create post-processed data files adjacent to the raw data files in the archive. mbdatalist allows us to insert a flag ("`$PROCESSED`") into any data list causing the processed versions of the data in that list, and those beneath it in the recursion structure, to be extracted rather than the raw, unprocessed data. In this way we may selectively extract the processed or non-processed data files within our data structure. As an illustration, you'll find a secondary data list in the highest level directory called `datalistp.mb-1` which contains only two entries - the `$PROCESSED` flag and the original archive data list `datalist.mb-1`.

Note

Alternatively, mbdatalist can be called with the `-P` flag which sets the `$PROCESSED` flag and overrides all others.

Similar to the discussion above, a "`$RAW`" flag in a data list or the `-U` flag on the command line can be specified to explicitly extract only the unprocessed data files, either by data list or universally as above.

Suppose we want to know the geographic bounds of the entire Lo'ihi archive. We simply run mbinfo and refer to the highest level data list to get statistics regarding the entire archive.

```
 mbinfo -F-1 -I datalist.mb-1
...

Data Totals:
Number of Records:           191764
Bathymetry Data (135 beams):
  Number of Beams:          12119082
  Number of Good Beams:      8745488     72.16%
  Number of Zero Beams:      1590395     13.12%
  Number of Flagged Beams:   1783199     14.71%
Amplitude Data (135 beams):
  Number of Beams:           7285751
  Number of Good Beams:      5519648     75.76%
  Number of Zero Beams:       608084      8.35%
  Number of Flagged Beams:   1158019     15.89%
Sidescan Data (1024 pixels):
  Number of Pixels:         10155008
  Number of Good Pixels:     4023419     39.62%
  Number of Zero Pixels:     6131589     60.38%
  Number of Flagged Pixels:        0      0.00%

Navigation Totals:
Total Time:          -49797.0534 hours
Total Track Length: 36092.8118 km
Average Speed:           0.0000 km/hr ( 0.0000 knots)

Start of Data:
Time:  06 21 1997 19:01:07.393000  JD172
Lon: -155.2214     Lat:   18.8823     Depth:   301.0800 meters
Speed:  1.9631 km/hr ( 1.0611 knots)  Heading: 229.1400 degrees
Sonar Depth:    0.3000 m  Sonar Altitude:  300.7800 m

End of Data:
Time:  10 16 1991 21:57:55.000000  JD289
Lon: -157.8782     Lat:   21.2853     Depth:    33.0000 meters
Speed: 20.5175 km/hr (11.0906 knots)  Heading:  31.2014 degrees
Sonar Depth:    0.0000 m  Sonar Altitude:   33.0000 m

Limits:
Minimum Longitude:    -157.8801   Maximum Longitude:    -153.3829
Minimum Latitude:       18.5684   Maximum Latitude:       21.2853
Minimum Sonar Depth:    -0.4100   Maximum Sonar Depth:  1530.3800
Minimum Altitude:      -92.9500   Maximum Altitude:     5482.8000
Minimum Depth:          33.0000   Maximum Depth:        5605.0000
Minimum Amplitude:     -60.0000   Maximum Amplitude:     255.0000
Minimum Sidescan:        0.1600   Maximum Sidescan:       90.5000
```

In these results we see that information regarding track length, speed, and duration are incorrect. mbinfo as they become meaningless when considering multiple overlapping data sets taken on different ships at different times. However the geographic and depth "Limits" are reported correctly as are the beam statistics. For example, here we see the archive has a minimum depth of 33 meters and maximum depth of 5604 meters.

Suppose you are interested in which data files contain data covering the top of Lo'ihi. We can use mbda-

talist to create an extraction data list that covers only data files containing data within specified geographic bounds, for example:

```
mbdatalist -F-1 -I datalist.mb-1 -R-155.5/-155/18.7/19.2
```

results in the following data list containing with data within the bounds -155.5/-155/18.7/19.2. :

```
HUGO97/mba97172L05.mb121 121 100.000000
HUGO97/mba97174L01.mb121 121 100.000000
HUGO97/mba97174L02.mb121 121 100.000000
HUGO97/mba97174L04.mb121 121 100.000000
HUGO97/mba97174L05.mb121 121 100.000000
HUGO97/mba97174L06.mb121 121 100.000000
HUGO97/mba97174L07.mb121 121 100.000000
HUGO97/mba97175L01.mb121 121 100.000000
HUGO97/mba97175L03.mb121 121 100.000000
MBARI1998EM300/mbari_1998_53_msn.mb57 57 1.000000
MBARI1998EM300/mbari_1998_54_msn.mb57 57 1.000000
MBARI1998EM300/mbari_1998_55_msn.mb57 57 1.000000
MBARI1998EM300/mbari_1998_56_msn.mb57 57 1.000000
MBARI1998EM300/mbari_1998_57_msn.mb57 57 1.000000
MBARI1998EM300/mbari_1998_58_msn.mb57 57 1.000000
MBARI1998EM300/mbari_1998_59_msn.mb57 57 1.000000
MBARI1998EM300/mbari_1998_60_msn.mb57 57 1.000000
MBARI1998EM300/mbari_1998_61_msn.mb57 57 1.000000
SumnerLoihi1997/61mba97043.d01 121 0.100000
SumnerLoihi1997/61mba97043.d02 121 0.100000
SumnerLoihi1997/61mba97044.d01 121 0.100000
SumnerLoihi1997/61mba97044.d02 121 0.100000
TUNE04WT/TUNE04WT.SWSB.91oct08 16 0.001000
TUNE04WT/TUNE04WT.SWSB.91oct09 16 0.001000
TUNE04WT/TUNE04WT.SWSB.91oct10 16 0.001000
TUNE04WT/TUNE04WT.SWSB.91oct11 16 0.001000
```

At the time of this writing, mbcopy does not support the datalist feature. However, we can write a simple script to extract the files in our data list to the local directory. As an example, a quick script is included here:

```
#!/usr/bin/perl
#
# Program Name: archiveExt
#
# Usage: cat datalist | ./archiveExt
#
# Val Schmidt
# Lamont Doherty Earth Observatory
#
#
# This script takes a datalist of files in an archive
# and extracts a copy of them to the local directory.
# No change in format is applied, however that is easily modified.
# Files not using the standard MB-System suffix (mb###) will have
# one appended.

while (<>){
        ($file,$format,$gridweight)=split(" ",$_);
        $infile=$file;
        # look for standard suffix and remove it if it exists
```

```
        if (($file=~m/mb...$/) || ($file=~m/mb..$/)){
                $file=~s/\.mb.*//;      # remove suffix
        }
        $file=~s/^.*\///g;      # remove leading path
        $outfile=$file;
        $cmd="mbcopy -F$format/$format -I$infile"." -O".$outfile."\.mb"."$format";
#           print "$cmd\n";
            system($cmd);

    }
```

With the data files extracted from the archive we can write them to some portable media and port them where ever we like.

As you can see data archives with systematic recursive data lists, combined with the MB-System™ tool mbdatalist provide a powerful tool for managing your data set within a larger archive.

4.4.2. Organize The Data Set Internally

Now that our MBARI1998EM300 data set has a home within our larger archive, we turn to looking at the data itself to determine if there is a need to further segment the data - "Divide and Conquer!" In general, it is usually helpful to segment the data into portions that require unique processing. One might place each subset into its own directory with its own recursive data list. This is a good time to take a close look at the notes that you have painstakingly taken during the cruise. They can be invaluable in providing clues for portions of the data that might be troublesome. This is also where you typically curse yourself, or someone else, for a lack of detail.

For example, suppose on the 31st day of a 45 day cruise, the ship's heading reference fails. Six hours later, a marine tech has suspended a magnetic needle from a string. And with a laser diode from a CD player a reflector, and a series of photo-transistors managed to restore the ship's heading to the sonar system. While far from perfect, the sonar has a modest idea of the ship's heading. But where before the multibeam data looked as though you'd painted the sea floor with a roller, it now looks like a Jackson Pollack painting. Post processing this data is not impossible, but certainly different than the typical multibeam data set and should be conducted separately from the rest of the data.

Or more simply, suppose your mapping near the edge of the Gulf Stream, and you've seen from the many XBT's shot during the cruise that there was a somewhat dramatic change in sound speed profile across what you suspect was a large eddy. Since you expect to be applying different sound speed profiles to each subset of the data, you might decide to segment the data around this change in sound speed profile.

Caution

When participating in a cruise it is imperative to consider in advance the kinds of things that could go wrong in a multibeam data set. Train watchstanders to look for and log changes in sound speed profile, operation of the ship's heading and attitude reference, GPS receiver operation, changes in sea state, and *MOST IMPORTANTLY* operation of the temperature instruments that provide the speed of sound at the keel. This final factor is the one element that cannot be corrected in post processing. Good notes provide lots of clues regarding how to fix data sets and provide immeasurable peace of mind when you find yourself presenting your data before your peers.

Clearly segmenting the data is something of an iterative process. As you look more closely at your data set and get further into the post processing, you may rearrange you plan of attack and change your organization structure. One must not get frustrated, but simply sigh and understand that it is part and parcel of the process.

4.4.3. Ancillary Files

Before we continue, it seems as good a time as any to describe the various ancillary data files generated and used by MB-System™. After a brief introduction to them we'll generate a few standard ones for our data set automatically with mbdatalist.

The ancillary data files provide statistics, metadata and abbreviated data formats for the manipulation, display and post processing of the multibeam data. A summary description for each is provided below.

- *.inf - file statistics*

  These files contain metadata about the multibeam data, including geographic, temporal and depth bounds, ship track information, and numbers of flagged or zero beams. They are created by mbinfo, often via mbdatalist and automatically via mbprocess.

- *.fbt - fast bathymetry*

  These files are bathymetry only format 71 files that are intended to provide faster access to swath bathymetry data than in the original format. These are created only for data formats that are slow to read. These are generated using mbcopy typically via mbdatalist or automatically through mbprocess.

- *.fnv - fast navigation*

  These files are simply ASCII navigation files that are intended to provide faster access to swath navigation than data in the original format. These are created for all swath data formats except single beam, navigation, XYZ. These are generated using mblist, typically via mbdatalist and automatically by mbprocess.

- *.par - parameter files*

  These files specify settings and parameters controlling how mbprocess generates a processed swath data file from a raw swath data file. These are generated or updated by all of the data editing and analysis tools, including mbedit, mbnavedit, mbvelocitytool, mbclean, mbbackangle, and mbnavadjust. They are also directly altered by mbset.

- *.esf - bathymetry edit flags*

  These files contain the bathymetry edit flags output by mbedit and/or mbclean.

- *.nve - edited navigation*

  These files contain the edited navigation output by mbnavedit.

- *.na0, .na1, .na2, .na3 ... - edited navigation*

  These files contain adjusted navigation output by mbnavadjust. These navigation files generally supersede *.nve* files output by mbnavedit. The latest (highest number) *.nv#* files are used so *.na2* will supersede *na1*.

- *.svp - sound velocity profile*

  These files contain a sound velocity profile used in recalculating bathymetry. The *.svp* files may derive from mblevitus, mbvelocitytool, mbm_xbt, or other sources.

We can now use mbdatalist to generate, for each data file, summary statistics and special bathymetry and navigation data that is more easily read by MB-System™ tools. This excerpt from the mbdatalist man page is helpful in understanding how they fit into the big picture:

MB-System™ makes use of ancillary data files in a number of instances. The most prominent ancillary files are metadata or "inf" files (created from the output of mbinfo). Programs such as mbgrid and mbm_plot try to check "inf" files to see if the corresponding data files include data within desired areas. Additional ancillary files are used to speed plotting and gridding functions. The "fast bath" or "fbt" files are generated by copying the swath bathymetry to a sparse, quickly read format (format 71). The "fast nav" or "fnv" files are just ASCII lists of navigation generated using mblist with a -OtMXYHSc option. Programs such as mbgrid, mbswath, and mbcontour will try to read "fbt" and "fnv" files instead of the full data files whenever only bathymetry or navigation information are required.

To generate these ancillary data files, from within the `MBARI1998EM300` directory, we execute mbdatalist with the *-N* flag:

```
mbdatalist -F-1 -I datalist.mb-1 -N
```

Now the directory looks like the following:

```
ls
datalist.mb-1                mbari_1998_57_msn.mb57.fnv
mbari_1998_53_msn.mb57       mbari_1998_57_msn.mb57.inf
mbari_1998_53_msn.mb57.fbt   mbari_1998_58_msn.mb57
mbari_1998_53_msn.mb57.fnv   mbari_1998_58_msn.mb57.fbt
mbari_1998_53_msn.mb57.inf   mbari_1998_58_msn.mb57.fnv
mbari_1998_54_msn.mb57       mbari_1998_58_msn.mb57.inf
mbari_1998_54_msn.mb57.fbt   mbari_1998_59_msn.mb57
mbari_1998_54_msn.mb57.fnv   mbari_1998_59_msn.mb57.fbt
mbari_1998_54_msn.mb57.inf   mbari_1998_59_msn.mb57.fnv
mbari_1998_55_msn.mb57       mbari_1998_59_msn.mb57.inf
mbari_1998_55_msn.mb57.fbt   mbari_1998_60_msn.mb57
mbari_1998_55_msn.mb57.fnv   mbari_1998_60_msn.mb57.fbt
mbari_1998_55_msn.mb57.inf   mbari_1998_60_msn.mb57.fnv
mbari_1998_56_msn.mb57       mbari_1998_60_msn.mb57.inf
mbari_1998_56_msn.mb57.fbt   mbari_1998_61_msn.mb57
mbari_1998_56_msn.mb57.fnv   mbari_1998_61_msn.mb57.fbt
mbari_1998_56_msn.mb57.inf   mbari_1998_61_msn.mb57.fnv
mbari_1998_57_msn.mb57       mbari_1998_61_msn.mb57.inf
```

Each of the tree types of data ancillary data files have been created for each data file in the original directory. This is a good time to create ancillary data files for all the data you will be processing as they will be greatly speed the process.

4.4.4. Surveying your Survey - Revisited

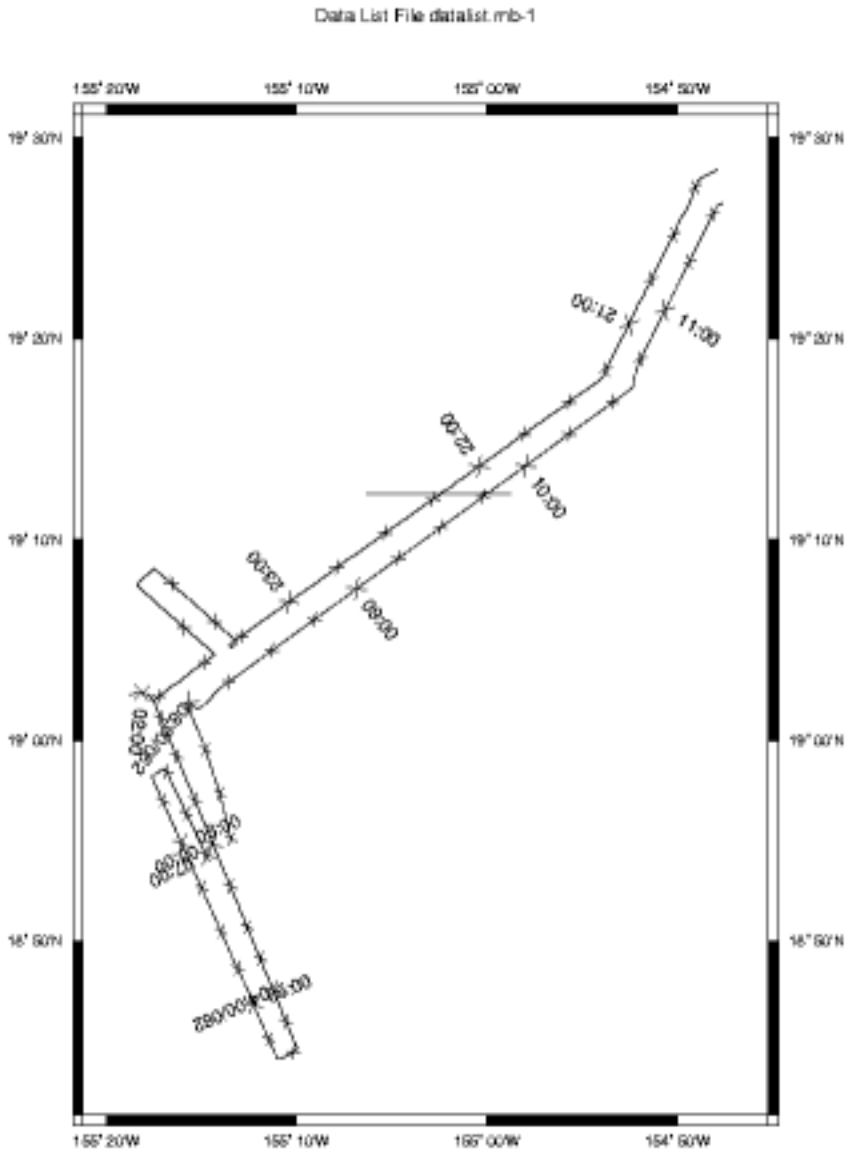
Now we need to take a quick look at our data and get a feel for what we're up against. This is much the same process described in the "Surveying your Survey" chapter. The idea is to get an idea of its quality and the lay of the sea floor.

First lets generate a navigation plot:

```
mbm_plot -F-1 -I datalist.mb-1 -N
```

And here's the resulting plot:

Figure 4.2. Lo'ihi Survey Navigation Data



Here we can see the ship's track coming in from the North East, a small mapping pattern over the summit of Lo'ihi and the return trip.

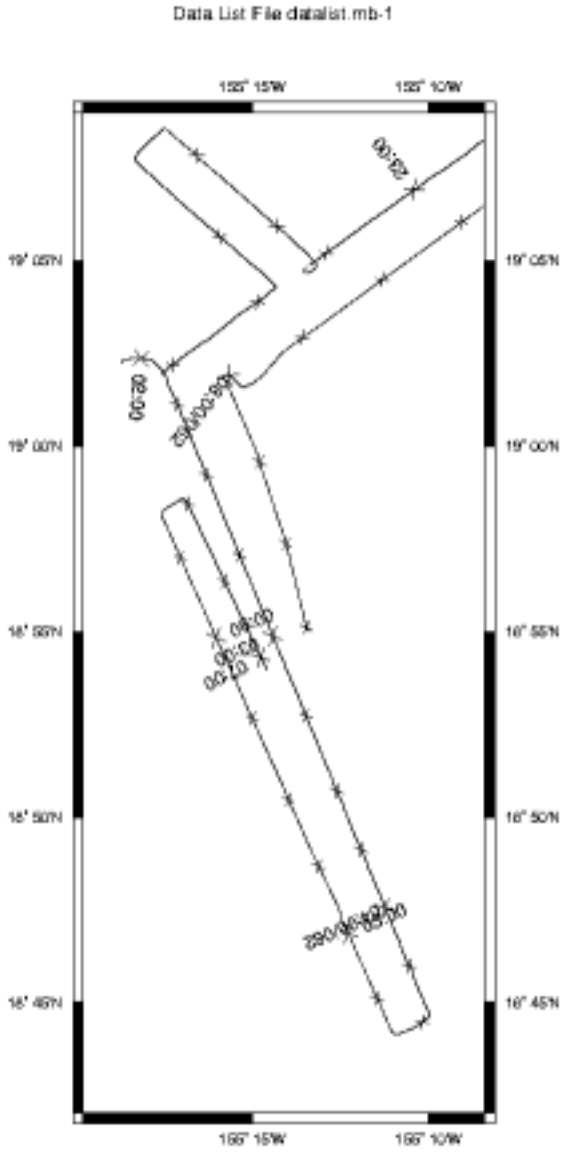
We can also see what appears to be an extraneous line on the plot at roughly 19'12''N. We note this irregularity as something to investigate.

If we look close there appears to be discontinuities along the ship's track at roughly 19'02''N 155'16W

(0200) and 18'55"N 155'15"W (0700). So we look focus a our plot a bit, using the *-R* flag to specify geographic bounds to mbm_plot, and look again.

```
mbm_plot -F-1 -I datalist.mb-1 -N -R-155.33/-155.14/18.7/19.15
```

Figure 4.3. Lo'ihi Survey Navigation Plot



We clearly see two discontinuities in the navigation data. Have we lost our navigation source? Was there

a problem with the sonar? Is there a portion of the data set missing? Anyone who has spend much time processing multibeam data has spent multitudes of time trying to answer these kinds of questions, however, it shouldn't be a process of data forensics. The answer should be in the cruise notes - a lesson that cannot be stressed enough.

In this case the answer is "none of the above". The notes say that this data set was taken by a commercial survey outfit. Commercial ships tend to only take data when they are absolutely required. Therefore, they frequently secure their data systems when outside the contracted survey area, or during turns, when a poor ship's vertical reference will make the data all but unusable. The "missing" data segments shown in the plot are a result and we know this because we took good notes during the cruise. Good notes are *essential* to post processing.

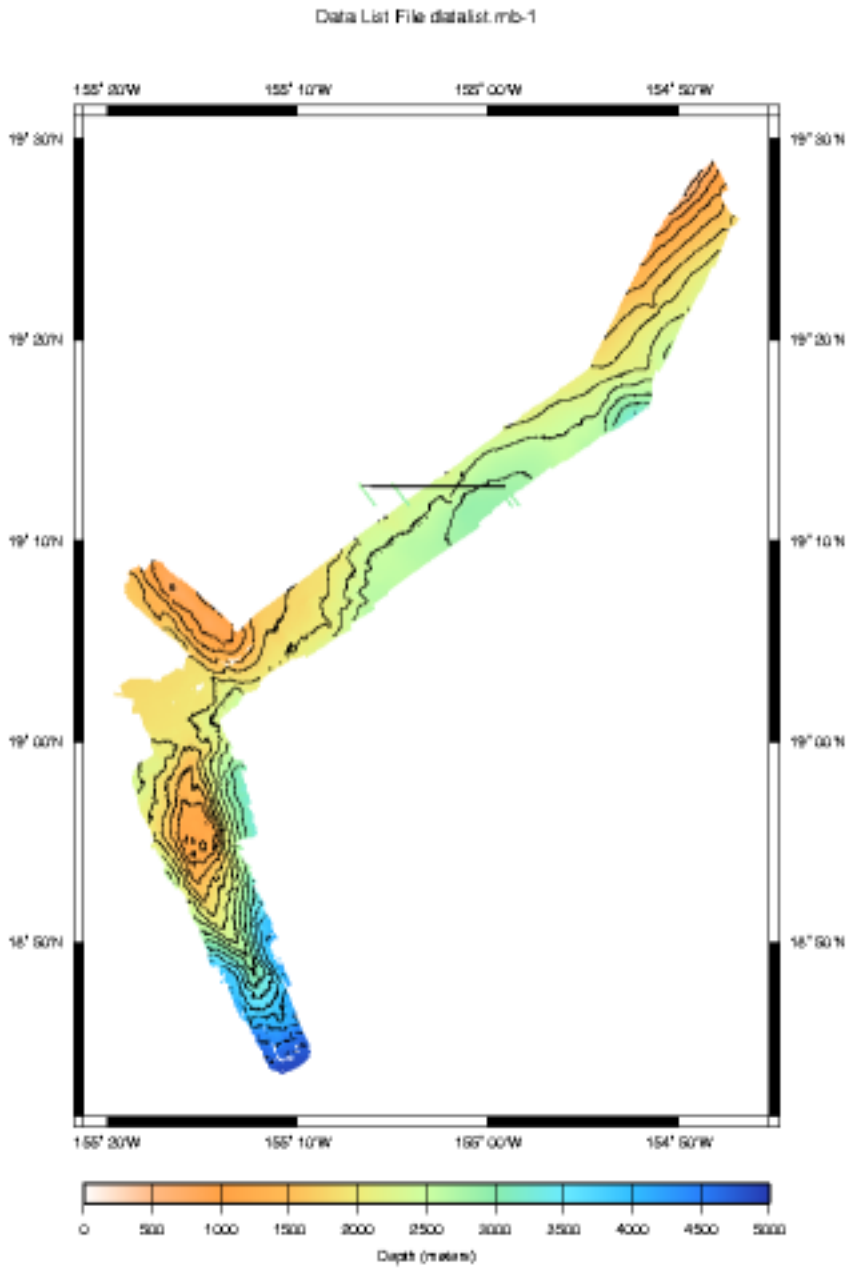
Note

The practice of securing sonars during turns often seems stingy to most scientists, who prefer to err on the side of collecting all the data they can. This comes with a price however. Considerable amounts of time and money are often then required to flag beams during turns or other events that prevent the taking of usable data.

Now we should take a look at a contour plot, as it can give us a better idea of the noise in the data set. Here we look primarily for contours that are jagged and irregular to signify data that requires editing.

```
mbm_plot -F-1 -I datalist.mb-1 -C -G1
```

Figure 4.4. Lo'ihi Survey Contour Plot



In the plot above we see that, in general, things look pretty good with two exceptions. The large line at 19'N is still apparent and appears to be more than a plotting artifact. Bathymetry data have been associated with the discontinuity placing small colored lines on our plot well outside our track and expected coverage. The deep water bathymetry at the bottom of the plot is also troublesome. Multibeam sonars radiate a fan shape from the bottom of the vessel, hence we expect the bathymetry map to widen with increasing water depth. Here however, the swath width narrows in a non-uniform way and data is lost near

the center beam. These effects can be caused by a sonar that is not appropriately adjusting its power and gain settings dynamically with depth. Or perhaps something else is afoot. At any rate, it will no doubt require careful consideration.

We won't segment this small data set any further, however it is useful to know the bounds in time and space of the entire data set so we run mbinfo

```
mbinfo -F-1 -I MBARI1998EM300/datalist.mb-1
```

... and the results:

```
Data Totals:
Number of Records:              9917
Bathymetry Data (135 beams):
  Number of Beams:           1338795
  Number of Good Beams:      1207888     90.22%
  Number of Zero Beams:        79143      5.91%
  Number of Flagged Beams:     51764      3.87%
Amplitude Data (135 beams):
  Number of Beams:           1338795
  Number of Good Beams:      1207888     90.22%
  Number of Zero Beams:        79143      5.91%
  Number of Flagged Beams:     51764      3.87%
Sidescan Data (1024 pixels):
  Number of Pixels:         10155008
  Number of Good Pixels:     4023419     39.62%
  Number of Zero Pixels:     6131589     60.38%
  Number of Flagged Pixels:        0      0.00%

Navigation Totals:
Total Time:            15.4770 hours
Total Track Length:   286.9084 km
Average Speed:         18.5377 km/hr (10.0204 knots)

Start of Data:
Time:  03 02 1998 20:06:01.740000  JD61
Lon: -154.7988     Lat:   19.4734     Depth:  1000.6700 meters
Speed: 18.0000 km/hr ( 9.7297 knots)  Heading: 243.2100 degrees
Sonar Depth:    6.0700 m  Sonar Altitude:  988.5300 m

End of Data:
Time:  03 03 1998 11:34:38.985000  JD62
Lon: -154.7938     Lat:   19.4458     Depth:  1502.0900 meters
Speed: 18.0000 km/hr ( 9.7297 knots)  Heading:  59.4300 degrees
Sonar Depth:    5.0100 m  Sonar Altitude: 1492.0700 m

Limits:
Minimum Longitude:    -155.3265   Maximum Longitude:    -154.7814
Minimum Latitude:       18.7261   Maximum Latitude:       19.4817
Minimum Sonar Depth:     3.8700   Maximum Sonar Depth:     7.4600
Minimum Altitude:      695.0300   Maximum Altitude:     4861.8900
Minimum Depth:         331.8400   Maximum Depth:        4919.0700
Minimum Amplitude:       2.0000   Maximum Amplitude:      58.5000
Minimum Sidescan:        0.1600   Maximum Sidescan:       90.5000
```

Gaining some knowledge about our data set and its potential problems is just the first step in preparing to process the data. Before we can conduct our final processing we need to eliminate other errors. Among these are the ship and sonars roll and pitch biases. So we move on to that topic next.

4.5. Determine Roll and Pitch Bias

4.5.1. Roll Bias

4.5.1.1. Roll Bias Introduction

When a ship is at sea, the sonar array is continuously moving with respect to the sea floor due to the ship's motion. The sonar must then continuously correct the sonar data for the ship's motion by comparing the orientation of the ship at any instance with some some known vertical reference. Most sonars take input from separate gyroscopic or inertial navigation systems which provide these corrections in real time.

Roll bias is a measure of the difference between the athwartships alignment of the ship's multibeam hydrophone array, and that of its vertical reference source. Although the installation of both the hydrophone array and the inertial navigation system are done with the utmost care to minimize these differences, some error always exists. Moreover, over the course of the life of a ship the values will change and should be periodically remeasured.

Note

Roll bias is frequently misunderstood to be a measure of the difference between the sonars alignment and the true vertical. This is not the case, and can cause considerable confusion when troubleshooting and trying to understand abnormal tilts in a data set. The measurement of roll bias, as we shall see, is between the ships vertical reference and the sonar. The assumption is made that errors in the ships vertical reference from the true vertical average out over time such that the reference approximates the true vertical, however this is not always the case.

A correction to the current roll bias value is measured by taking two sets of data over a planar bottom while sailing nearly exact reciprocal courses. (Note that the sea floor need not be level, but must be as planar as possible, ideally with any slope perpendicular to the course of the ship.) The two data sets are then fit to planes and an angular athwartships deviation is calculated between them. This value is then applied as a *correction* to the current roll bias in the system.

The example that follows is a narrative discussion of a roll bias calculation performed aboard the R/V Ewing in the spring of 2002.

4.5.1.2. A Real Live Roll Bias Example

4.5.1.2.1. Data Collection

Data was taken for the roll bias calculations during the transit between Guam and Dutch Harbor, Alaska. A possible flat bottom area was found by chart review. During the transit over this area, the Hydrosweep was carefully watched to find the a stretch of planar sea floor suitable to conduct the test.

The ship was in water of about 5700 meters. This translated into a ping interval of about 20 sec. We therefore decided on a data set 45 min long in each direction to give us some 150 ping returns. To reduce the ship's noise in the data set we slowed to 8 knots during the runs.

After identifying a good 45 min stretch of sea floor we noted it's beginning latitude and longitude and went to the bridge. We asked the helm to mark our current position, conduct a Williamson turn and slow to 8 knots, retrace our track to our beginning point, conduct a second Williamson to resume base course, and finally notify us when they had reached the starting point, after which they could resume their transit speed.

Our resulting data files covered 200204282310 - 20020429010010 GMT.

4.5.1.2.2. Data Manipulation

The Data Manipulation strategy is to convert the "raw" data files to a more manageable format, plot the data such that exact start and stop times can be identified for each course, and finally extract data for each track using these start and stop times to create composite data files for each reciprocal course track. These composite data files can then be directly input into the MB-System™ mbrollbias process.

The raw data files, retrieved directly from the sonar workstation, each contain 10 minutes of data and are in the "182" MB-System™ format.

A 10 minute file set in the "182" format contains 9 files. While the bathymetry is primarily contained in the `.fsw` files the remaining 8 files are also required.

Once copied to a working directory the files were converted to a more manageable 183 format using mbcopy. A small script was written to do them all in one step.

```
#!/usr/bin/perl
#
#
# Usage: ls -1 | ./multi-copy
#
# Val Schmidt
# Lamont Doherty Earth Observatory
#
# This script takes a directory full to 182 format multi
# beam files and converts them to 183 format files.

while (< >){
        if (/.*fsw$/){
            chomp;
            s/$_/\.fsw$//;
            $cmd="mbcopy -F182/183 -I$_"."\.fsw"." -O"."$_"."\.mb183";
            #print $cmd;
            system($cmd);
        }
    }
```

Note

Subsequent beta versions of MB-System™ (5b29 and later) have included the new process mbm_copy which allows one to copy large numbers of data files between formats and obsoletes the need for this script. See, feedback works!

MB-System™ 's 183 format is compact, with only a single file for the bathymetry, navigation and other information. The results of the above script are single data files with the ".mb183" suffix appended to the end of their root filename. This suffix is recognized by MB-System™ interactive utilities such as mbedit and mbvelocitytool.

Times annotated in our notes during the rollbias data taking runs allowed us identify the files containing the bathymetry data for each track. A text file containing a list of these files and their format was created to facilitate the subsequent steps as shown below.

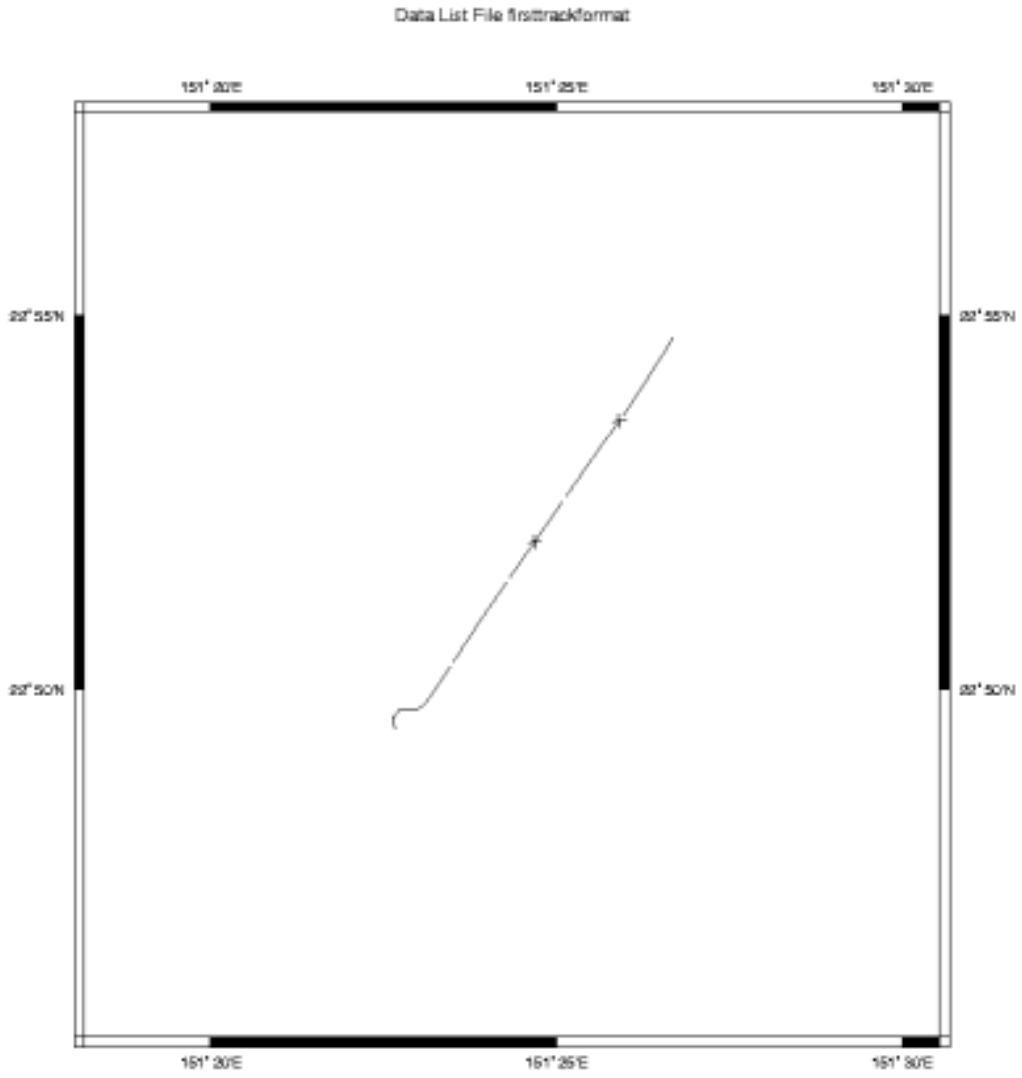
```
[vschmidt@val-ldeo rollbias]$ cat firsttrackformat
00020428232010.mb183 183
00020428233010.mb183 183
00020428234010.mb183 183
00020428235010.mb183 183
00020429000010.mb183 183
```

```
[vschmidt@val-ldeo rollbias]$ cat secondtrackformat
00020429001010.mb183 183
00020429002010.mb183 183
00020429003010.mb183 183
00020429004010.mb183 183
00020429005010.mb183 183
00020429010010.mb183 183
```

These data sets were then plotted to ensure they cover the reciprocal tracks and do not contain any extraneous data particularly the turns at the end of each leg.

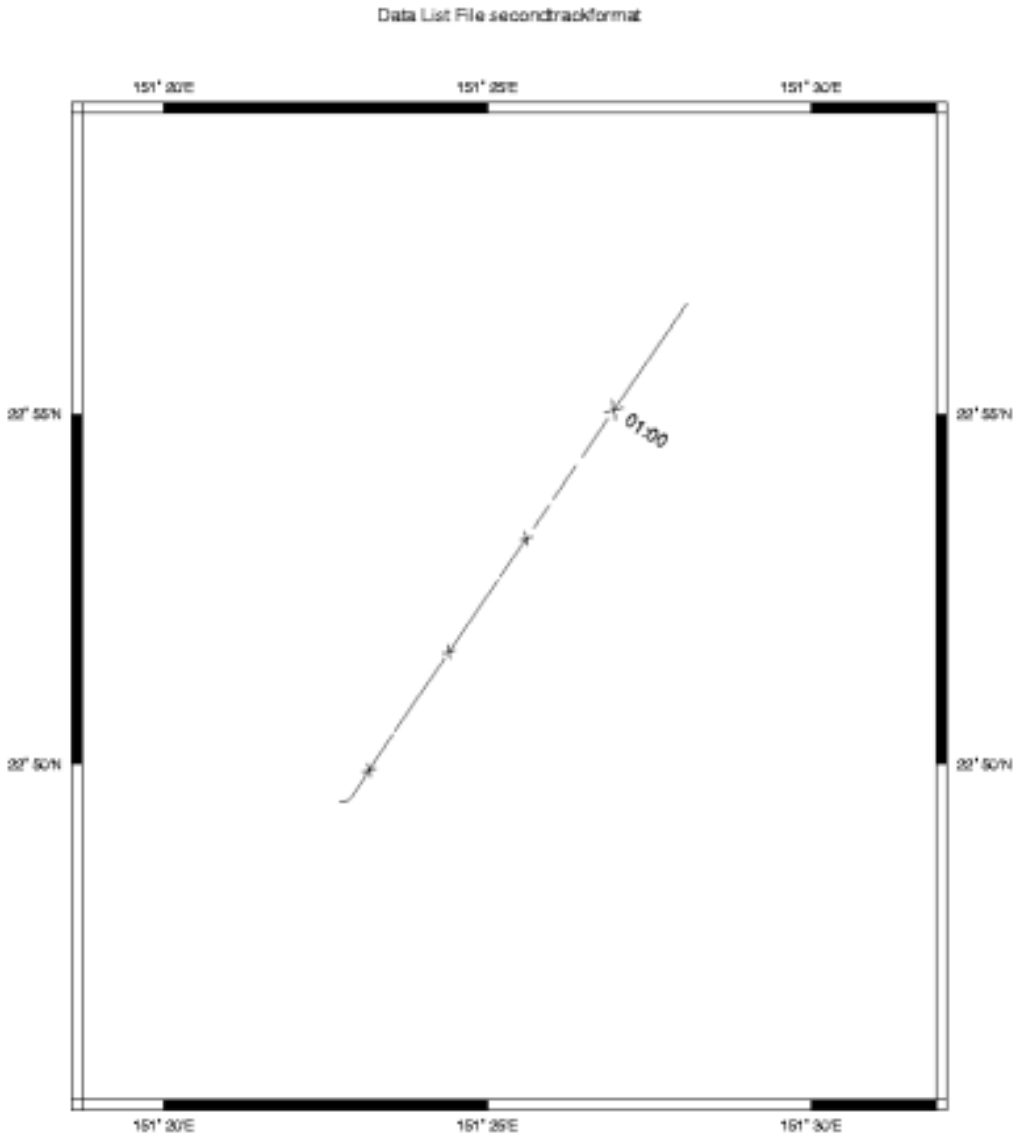
```
mbm_plot -F-1 -I firsttrackformat
```

Figure 4.5. Plot of First Data Leg for Roll Bias Calculation




```
mbm_plot -F-1 -I secondtrackformat
```

Figure 4.6. Plot of Second Data Leg for Roll Bias Calculation



As luck would have it, our first track contains a portion of the turn. We therefore have to window the data file to remove the unwanted section. The last data file in the first track list was the one that required windowing. By noting the time at which the turn started, a copy of the straight portion of the track could be extracted using mbcopy.

```
mbcopy -F183/183 -Iinputfilename -Ooutputfilename -BYYYY/MM/DD/HH/MM/SS -EYYYY/MM/
```

We then augmented our "firsttrack" file list as shown below.

```
vschmidt@grampus:~/ewing/data/cruises/ew0204/rollbias% less firsttrack
00020428232010.mb183
00020428233010.mb183
00020428234010.mb183
00020428235010.mb183
00020429000010.copy.mb183
```

The mbrollbias process requires a single set in input files for each leg of the track. Armed with the *firsttrack* and *secondtrack* file list we created single composite data files for each track with the following lines shown below

```
cat firsttrack | xargs cat >> compositefirsttrack.mb183

cat firsttrack | xargs cat >> compositesecondtrack.mb183
```

Note

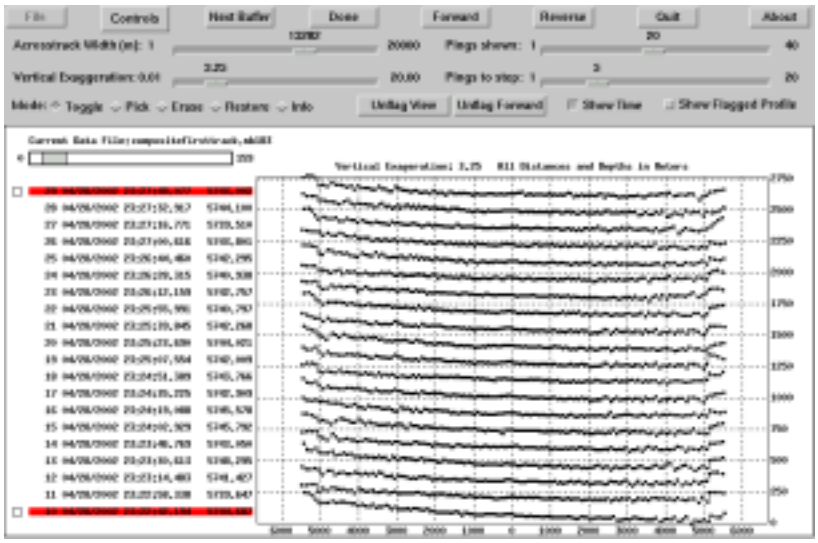
The concatenation of data files as shown below does not typically work with the "native" sonar formats.

So now we'd looked at the data from each track, windowed them appropriately to ensure we had exactly the data we wanted, and concatenated the data into single composite data files for each of the two reciprocal courses.

4.5.1.2.3. Data Processing

Now that we have extracted the data files, we needed to clean them up a bit. This was a straight forward manual process of running *mbedit*. From within mbedit we loaded the first data file.

Figure 4.7. mbedit Screen Shot



We used the *Filters...* selection under the *Control* button to flag the outer 10 beams on either side, as

they are typically quite noisy. Then after adjusting the vertical scaling to something around 3 and increasing the number of beams displayed in each screen to something around 30, we carefully stepped through the entire data set. Using the "toggle" setting and mouse we flagged individual unruly or inconsistent beams. Occasionally an entire ping would be removed by first flagging a single beam and then typing *z*. When complete with the first file we clicked *Quit* which closed mbedit and saved a *.par* file for later processing of the flagged beams. The editing process was repeated for the second composite data file.

With the flagging of poor data complete for both data sets we ran mbprocess to apply the corrections and create new edited data sets.

```
mbprocess -Icompositefirsttrack.mb183 -F183
mbprocess -Icompositesecondtrack.mb183 -F183
```

These lines result in processed data files named compositefirsttrackp.mb183 and compositesecondtrackp.mb183 respectively. For illustration they are plotted below.

Figure 4.8. Plot of *compositefirsttrackp.mb183*

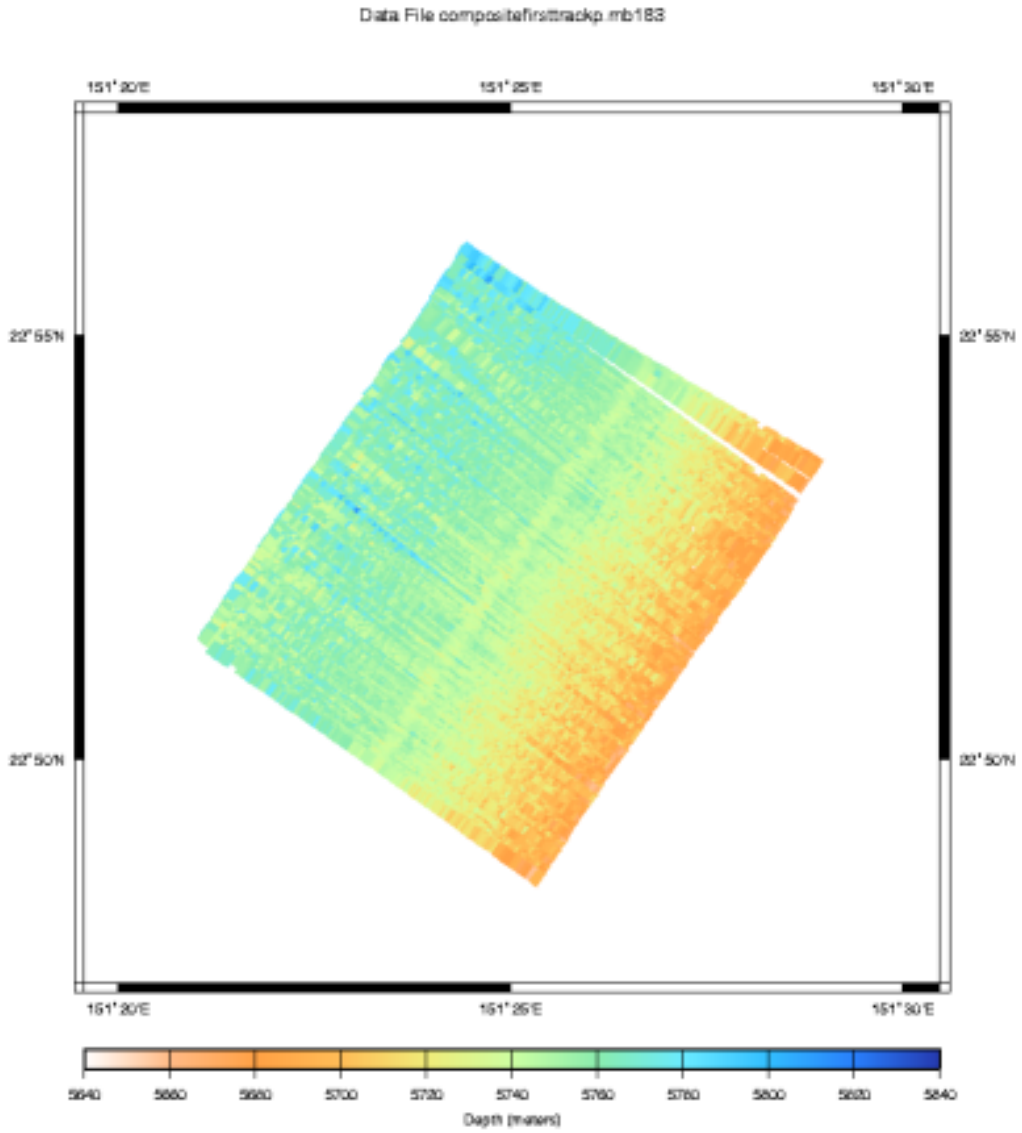
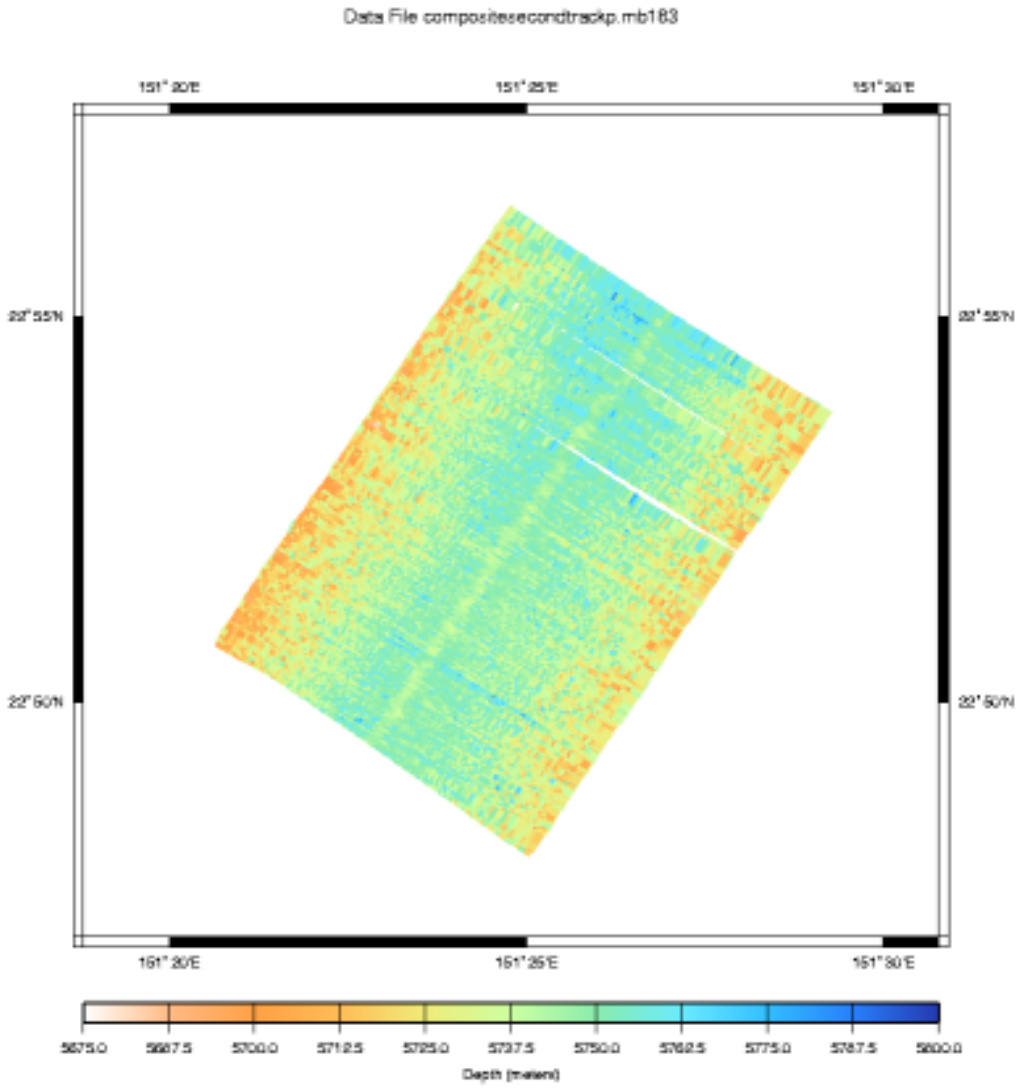


Figure 4.9. Plot of *compositesecondtrackp.mb183*



4.5.1.2.4. Calculating Rollbias

With clean data sets for each reciprocal track the set remaining is use mbrollbiasto conduct the final calculations. mbrollbias takes as arguments, the two input data file and their formats, as well as a region in latitude and longitude over which to calculate. mbrollbias splits this region into a grid and calculates roll bias values for each region. The number of grid squares can be specified with the *-Dx/y* argument. This can be helpful when the data only covers some small swath of a larger region. Alternatively mbrollbias can be forced to conduct the calculations over the composite whole returning only a single value using *-D1/1*. This is what we have done, our results are below.

```
mbrollbias -F183/183 -I compositefirsttrackp.mb183 \
-J compositesecondtrackp.mb183 -D1/1 -R151.4997/151/3344/22.7970/22.9520

MBROLLBIAS Parameters:
Input file 1:     compositefirsttrackp.mb183
Input file 2:     compositesecondtrackp.mb183
Region grid bounds:
  Longitude:  151.3344  151.4997
  Latitude:    22.7970   22.9520
Region grid dimensions: 1 1
Longitude interval: 0.165300 degrees or 16.962596 km
Latitude interval:  0.155000 degrees or 17.165048 km
Longitude flipping:   0

17902 depth points counted in compositefirsttrackp.mb183
19747 depth points counted in compositesecondtrackp.mb183
17902 depth points read from compositefirsttrackp.mb183
19747 depth points read from compositesecondtrackp.mb183

Region 0 (0 0) bounds:
    Longitude:  151.3344  151.4997
    Latitude:    22.7970   22.9520
First data file:    compositefirsttrackp.mb183
    Number of data: 17902
    Mean heading:   217.036742
    Plane fit:      5.750465 -0.006331 0.005092
Second data file:   compositesecondtrackp.mb183
    Number of data: 19747
    Mean heading:   31.696466
    Plane fit:      5.735079 0.000629 0.000129
Roll bias:   0.004220 (0.241800 degrees)
Roll bias is positive to starboard, negative to port.
A positive roll bias means the vertical reference used by
    the swath system is biased to starboard,
    giving rise to shallow bathymetry to port and
    deep bathymetry to starboard.
```

With the roll bias correction calculations complete, one may either set MB-System™ to apply these corrections to the data, or preferentially, the correction is entered into the multibeam sonar system such that corrections are made on the fly. Let us first look at how to apply the roll bias to the sonar data in post processing using MB-System™, as entering the value into the sonar sounds easy, but this process is sufficiently confusing it is well worth belaboring the details.

Applying roll bias corrections to sonar data only requires that we specify, in the ancillary parameter file (".par"), the roll bias to apply. This is done most easily using mbset. The syntax for such a statement might look like the following:

```
mbset -F-1 -Idatalist-1 -PROLLBIASMODE:1 -PROLLBIAS:rollbias_value
```

In the line above, all the data files listed in datalist-1 will be affected. Parameters to be set are specified with the "-P" flag to mbset. A rollbias mode of 1 specifies that roll bias corrections should be processed, and that they should be applied to both the port and starboard beams equally. (Alternatively roll bias correction processing can be turned off - Mode 0, or applied with different biases for port and starboard beams - Mode 2) Finally, the roll bias value to apply is specified in the last argument. With the line above executed, subsequent execution of mbprocess will apply the roll bias corrections to that data.

Now we can move on to the delecate process of entering roll bias values directly into the sonar, such that the data is correct "right out of the box". The details of this should be routine to any ship with a multi-

beam sonar, and asking the right questions from knowledgeable people should illicit the correct answers. However often that is not the case, so an in depth discussion is worth while, in the event one has to figure it out on their own.

First one must know the polarity convention the sonar uses for roll - is a roll to STBD a positive value, or a negative one? Most sonars consider a roll to STBD as positive, but many have the ability to change this polarity convention via software or dip switch settings, so you'll have to investigate. It is typically not sufficient to look at the convention of the vertical reference. Vertical reference inputs often have user-selectable polarity for roll values as well. So while the sonar may think of a roll to STBD as positive, it may actually take a negative input for a STBD roll from the vertical reference. The true-blue test is to fake the roll value entered into the sonar and watch the cross track profile of the sonar swath. Many newer inertial navigation units have the ability to do just that. A simulated roll to port will have the effect of slanting the cross track profile down to STBD.

Next one must understand how the sonar applies the roll bias settings. For example, does the sonar's interface have the operator enter the sonar's measured roll bias value, which it then applies as a correction to (subtracts from) the real time roll, or does it have one enter the value as a the roll bias *correction* which it then simply applies (adds to) the real time roll. Unfortunately, sonar operator manuals often do not make these essential details clear, and one has to experiment with large values, inspect the sonar data, and convince yourself that you've got it right.

The final hurdle, is to understand that the value provided by mbrollbias is the actual roll bias as measured from the current settings in the sonar. If one were to "zero" the roll bias settings before collecting the data, the values provided by mbrollbias would be the absolute roll bias. However since we did not, in this example, the values provided are *adjustments* to the current values. That is, if the current roll bias setting is +.10 degrees, the new roll bias setting would be .10 - .24 = - .14 .

4.5.2. Pitch Bias

Determine Pitch Bias

4.6. Determine Correct SSP's

4.6.1. Variations in Sound Speed and Your Data

As has been discussed in other sections, determining the correct sound speed profile for your data sets is crucial to high quality multibeam sonar data. Understanding the science of sound travel and how it can affect your data will give you clues to finding the best sound speed profile, however it can still be something of an art rather than a science.

4.6.1.1. Sources of SSP's

There can never be too many sources of sound speed profile data.

Ideally, SSP's are measured directly using CTD's or XBT's during the cruise. A conscientious investigator will measure sound speed profiles in a periodic manner, perhaps daily, and will take additional measurements based on bathymetric and meteorological variations. Still further measurements should be taken when the data itself shows signs of an erroneous profile.

This direct measurement method is most accurate and provides the best results. Typically results from these measurements, in the form of a table of depths and sound speeds, are input directly into the sonar. Subsequent data taken is processed by the sonar in real time with the correct sound speed profile, until of course ambient conditions change. The sound speed profiles should be retained in the event that later analysis requires adjustments.

Note

> Never forget a good quality sound speed profile is never a substitute for a poor keel water temperature input to the sonar. No amount of post processing can correct the beam forming errors that result from an incorrectly measured water temperature at the hydrophone array.

Unfortunately, not all investigators are as conscientious as the post processor might like. XBT's can be expensive and because cruises are full of activity, the quality of the sonar data is not always carefully watched.

Alternatively, sound speed profiles can be calculated from data collected in areas where the bottom is flat and the depth is known and that are in close proximity to the rest of the data set. Investigators often build these calibration tracks into their data to allow for an additional input for SSP processing.

SSP's can also be calculated from historical temperature, salinity and depth (pressure) tables. Environmental parameters of some ocean regions change little of the course of the year and for these areas historical data can often provide good estimates. In other ocean regions where there is considerable variability historical data is less helpful.

4.6.1.2. MBLevitus

The MB-System™ package provides a historical database of temperature and salinity values with depth as well as the mblevitus function for extracting sound speed profiles from this data. It's described well in the mblevitus man page:

```
mblevitus  generates  a  mean water sound velocity profile
for a specified location using  temperature  and  salinity
data from the 1982 Climatological Atlas of the World Ocean
[Levitus, 1982].  The water velocity profile is  representative
  of  the  mean  annual water column structure for a
specified 1 degree by 1 degree  region.   The  profile  is
output  to  a specified file which can be read and used by
programs (e.g. mbbath or mbvelocitytool)  which  calculate
swath sonar sonar bathymetry from travel times by ray tracing
 through a water velocity model.
```

More on mblevitus later.

4.6.1.3. An Example

Like the roll bias example above, the data sets used in this example are taken from a Pacific Ocean transit aboard the R/V Ewing, rather than the example Lo'ihi data set we have been using to illustrate examples. The data and sound speed profiles can found in the mbexamples/cookbookexamples/other_data_sets/ssp/ directory.

4.6.1.3.1. Collecting Sound Speeds

In practice SSP's are gathered from as many sources as possible. These are then compared and manipulated while simultaneously observing the effects on a portion of the sonar data. In this way, the best composite profile can be determined. MB-System™ provides mbvelocitytool as an interactive way to accomplish this as we will see.

Our example data set from the R/V Ewing taken in the spring of 2002 in the North Pacific during a non-science transit. Although every ship is different, the general procedure for the measurement and analysis of data to determine the sound speed profile is essentially the same. A simple procedure follows:

1. Review the chart to find a planar portion of sea floor for data collection. Analysis of the impact of changes to a sound speed profile on a data set is greatly eased, if the sea floor is in fact flat (not just

planar), although this is not imperative.

2. Brief the Captain and the Mate. If the ship's speed adversely affects the quality of data you may want to request a slower speed. If prevailing winds or seas have adverse affects on certain headings, you may want to request an alternate heading to mitigate these effects. Make sure the Captain and the Mate are aware of where and when you plan to launch the XBT or CTD as they have ship's speed requirements as well.

3. Ensure any software/hardware preparations required for XBT launch are complete.

4. Then simply note the start and stop times of the data run and the time/Latitude and Longitude of the XBT/CTD launch.

For this data set, an XBT was launched over a planar section of sea floor in the North Pacific, and a small 10 minute section of data coinciding with that location was chosen.

Here are the first few lines of the text file result produced by our Sparton™ XBT launch:

```
SOC BT/SV PROCESSOR
01/05/102
08:45
3
2
R/V Ewing
ew0204 4may z
41:01:00N
168:23:0E
5700 m.
5 kts.
10.4 c
0.00    15.639
0.68    13.126
1.37    11.467
2.05    10.945
2.73    10.801
3.41    10.749
4.10    10.743
...
```

The XBT provides depth and temperature, but depth and sound speed values are required. MB-System™ provides the script mbm_xbt to convert depth and temperature values to depth and sound speed.

Note

XBT's providing only depth and temperature make the assumption that salinity effects on sound speed are due to a bulk difference in salinity rather than gradients in the vertical water column. In the open ocean away from ice or shorelines, this is a good assumption. However in polar regions or near shoreline runoff, CTD's should be used.

mbm_xbt supports data from two types of XBT's, the NBP MK12 and the Sparton. It is a sophisticated little script that optionally takes a local salinity for bodies of water outside the nominal 35 ppt as well as the ship's latitude to correct for pressure differences with depth that occur as a result. (Unless otherwise specified mbm_xbt assumes a salinity of 35 ppt and a latitude of 0.) Our results are below:

```
# Sparton XBT data processed using program mbm_xbt
# SOURCE:              SOC BT/SV PROCESSOR
# DATE:                01/05/102
```

```
# TIME:                 08:45
# UNKNOWN PARAMETER:    3
# UNKNOWN PARAMETER:    2
# SHIP:                 R/V Ewing
# CRUISE:               ew0204 4may z
# LATITUDE:             41:01:00N
# LONGITUDE:            168:23:0E
# DEPTH:                5700 m.
# SPEED:                5 kts.
# SURFACE TEMPERATURE: 10.4 c
0.00 1508.67
0.68 1500.60
1.37 1494.98
2.05 1493.17
2.73 1492.68
...
```

To complement the XBT measurement it is always worth while to calculate SSP's from the Levitus database. If you have installed the Levitus database correctly as part of your MB-System™ installation (usually LevitusAnnual.dat in your $MBSYSTEM/share directory), one SSP will be calculated for you automatically when you load your data file into the mbvelocitytool application. However, it is best to calculate several profiles around the data set to ensure the data set does not fall in a historical thermal front. We have used +/- 1 degree increments to provide several sound profiles around the center location of the data file. To get a center Lat and Long we look at the "Limits" section of the mbinfo process executed on the data file for which we will conduct our sound speed analysis:

```
Limits:
Minimum Longitude:     168.6078   Maximum Longitude:     168.7523
Minimum Latitude:       41.1646   Maximum Latitude:       41.2759
Minimum Sonar Depth:     4.8000   Maximum Sonar Depth:     7.0000
Minimum Altitude:     5667.3852   Maximum Altitude:     5689.2592
Minimum Depth:        5556.8645   Maximum Depth:        5920.7173
Minimum Amplitude:       9.0000   Maximum Amplitude:     246.0000
Minimum Sidescan:        3.0000   Maximum Sidescan:      255.0000
```

The first few lines of a typical result from mblevitus is shown below:

```
mblevitus -R168.5/41.5 -O168E41N.svp -V

# Water velocity profile created by program MBLEVITUS version $Id: 168E41N.svp,v
 1.1.1.1 2002/10/31 22:07:08 dale Exp $
# MB-system Version 5.0.beta12
# Run by user <vschmidt$gt; on cpu <grampus> at <Sat May  4 07:33:49 2002>
# Water velocity profile derived from Levitus
# temperature and salinity database.  This profile
# represents the annual average water velocity
# structure for a 1 degree X 1 degree area centered
# at 168.5000 longitude and 41.5000 latitude.
# This water velocity profile is in the form
# of discrete (depth, velocity) points where
# the depth is in meters and the velocity in
# meters/second.
# The first 32 velocity values are defined using the
# salinity and temperature values available in the
# Levitus database; the remaining 14 velocity values are
# calculated using the deepest temperature
# and salinity value available.
0.000000 1498.003662
10.000000 1497.996826
```

```
20.000000 1497.213379
30.000000 1496.007935
50.000000 1492.968628
...
```

So with several sound speed profiles in hand, from the Levitus database and those we've measured directly, we use mbvelocitytool to assess the quality of these sound speed profiles.

4.6.1.3.2. Mbvelocitytool

mbvelocitytool is an interactive program that allows one to load several sound speed profiles and a data set to see how manipulations to those sound speed profiles affect the data in real time.

mbvelocitytool is launched simply with:

```
>mbvelocitytool
```

An interactive application will launch as shown here:

Figure 4.10. MB-VelocityTool

The strategy of mbvelocitytool is to compare an editable SSP with the measured and calculated SSP's we previously collected, and see the effects of our editing in real time on a real data set.

Note

There's bound to be a bit of confusion with regards to terminology used in this manual and that used in the MB-System™ software as they are slightly different. For example, while the term

"Sound Speed Profile" is technically more correct, and is used in this manual exclusively, "Sound Velocity Profile" has been used historically. Even the MB-System™ tools refer to "velocity" - mbvelocitytool for example, and "SVP" rather than SSP. The use of SSP over SVP is a pet peeve of the author, but please don't be confused by this. You'll see both used interchangeably in many places and certainly in MB-System™ - something to live with for the moment.

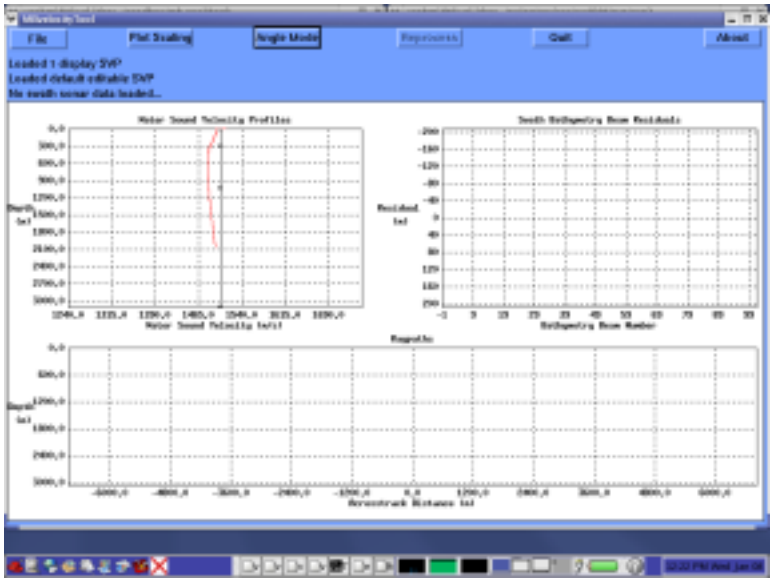
If we first select "File->New editable profile" we'll see a line a line with four (one is hidden somewhat at the top) handles (dots) appear on the "Water Sound Velocity Profile" plot. The handles allow graphical editing of the profile by the "drag and drop" method on the file handles.

If we select "File->Open editable profile" we'll see a dialog box that will allow us to browse to the location of our sound speed profiles. Find /mbexamples/cookbookexamples/other_data_sets/ssp and load the XBT SSP identifiable by an "sv" suffix. See the figure below:

Figure 4.11. MB-VelocityTool

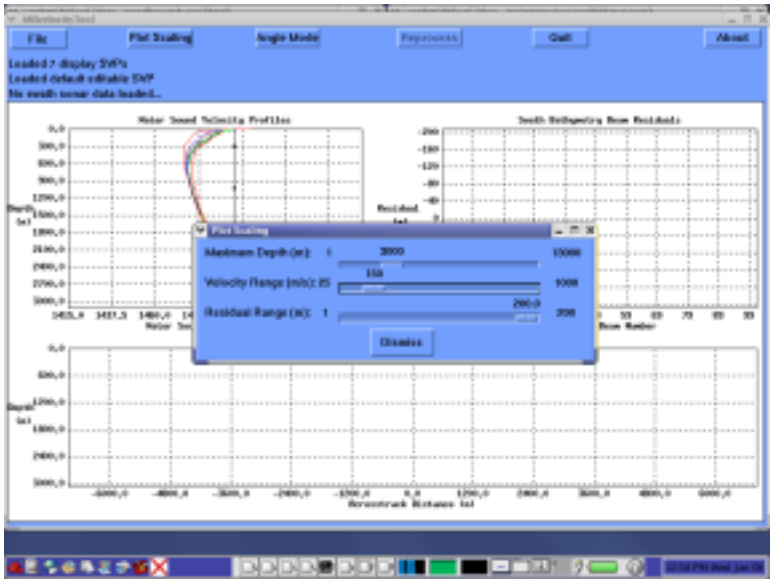
You'll find the XBT sound speed profile plotted in red on the "Water Sound Velocity Profiles" plot. Why does the data extend down only 2000 meters of water depth? The model of XBT used in this example is good only to that depth, but more importantly, changes in the SSP below the thermocline are entirely due to changes in pressure with depth - so we need not measure much below the thermocline to have a good idea of the shape of the SSP at deeper depths. Here's an illustration:

Figure 4.12. MB-VelocityTool with SSP's Loaded



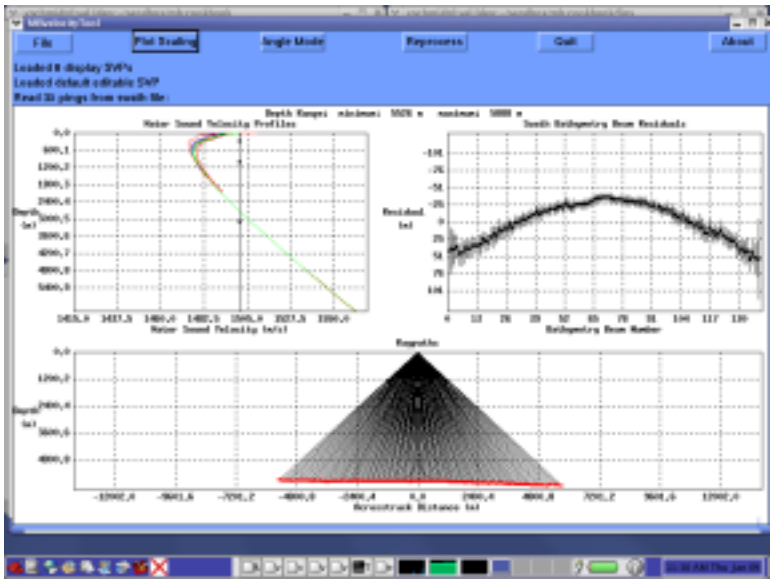
In the same manner as above, load the remaining SSP's generated by mblevitus. They will be added to the plot in various colors to make them easily distinguishable. In an effort to better see the differences between them, we can adjust the scale of the plot by selecting the "Plot Scaling" button. Slide the "Velocity Range" scale to 150. You'll probably be able to see the plot change scale in the background as you let go the mouse. Click "Dismiss" to close the Plot Scaling window.

Figure 4.13. MB-VelocityTool with Plot Scaling Dialog



With out SSP's loaded we can turn to loading our data set. The process is much the same - select "File->Open swath sonar data". Browse to the file, and ensure the correct format is specified in the dialog box before clicking Open. One additional SSP will be calculated for your swath data from the Levitus database automatically. Before we discuss the details of the various plots, let us adjust the plot scaling,"Maximum Depth" to something near 6000 meters. Here is the result:

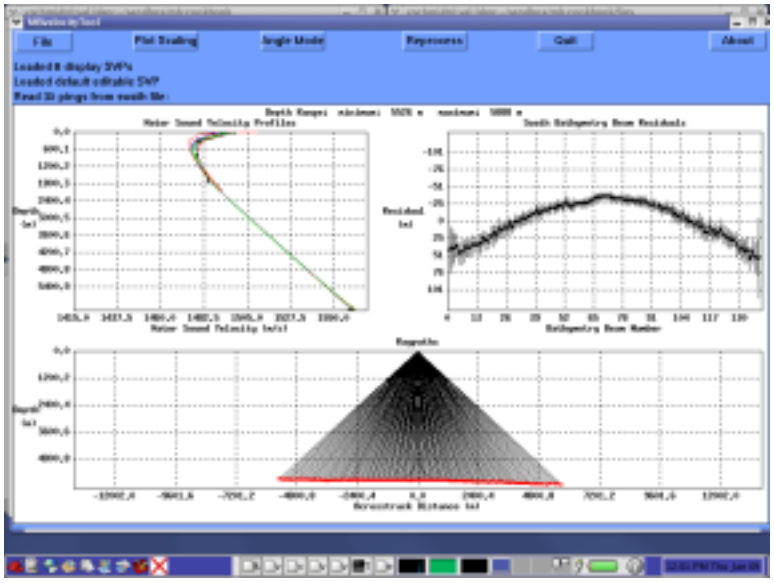
Figure 4.14. MB-VelocityTool with Sonar Data Loaded



When the loaded sonar data contains raw beam angles and travel times, the path the sonar traveled to the sea floor is calculated and displayed in the "Raypaths" plot. The first good data value in each beam is displayed from the data file (most from the first ping) to allow one to see realtime results of interactive editing. Based on the current SSP a line is fit to the across track profile of each ping and the difference between each bathymetry value and the linear fit is calculated. These are then averaged across the loaded data file and plotted as "residuals" in the upper right hand plot. Bathymetry values that are shallower than the fit are negative while those deeper are positive and the standard deviation of the averages are plotted as error bars.

Next we adjust our editable sound speed profile to approximate the loaded profiles. Place the cursor over one of the handles (dots), click with the left button and drag the profile as is appropriate. Often you will find that the four handles provided are not enough to adequately shape the profile. Additional handles can be added by placing the cursor over the portion of the profile that a handle is desired and clicking the center mouse button. Below the profile has been edited as described.

Figure 4.15. MB-VelocityTool with Edited Profile



You may have noticed that the ray tracing and residuals plots did not change while we edited our profile. These are updated with a click of the "Reprocess" button. Before we do that however, it will be helpful if we adjust our residual plot scaling. Because our example data set is of good quality, a value of 50 meters is appropriate. Now reprocess the data and note the changes in the Residuals plot.

Here on out, one iteratively adjusts the profile to both flatten the residual plot and minimize the error bars. Meanwhile, the raytracing profile should present a creditable representation of the sea floor.

With the sound speed profile adjusted to your liking, on can save the "editable" sound speed profile by selecting the appropriate choice under the "File" button.

REVISITED WITH A MORE DETAILED DISCUSSION OF THE EDITING PROCESS.

INSERT A DISCUSSION REGARDING SAVING RESIDUALS AND APPLYING THEM TO A DATA SET AND WHEN IT IS APPROPRIATE TO DO SO.

4.7. Smooth The Navigation Data

Multibeam sonar systems derive their position information from external navigation systems. These are typically some combination of GPS unit(s) and inertial navigation system(s). Towed sonars and AUV/ROV based systems must use inertial nav systems exclusively (with the exception of an initializing position which is typically correlated to a GPS source). All of these navigation systems are prone to various kinds of error. Multibeam data can only be as good as the reference navigation information the sonar receives.

In an effort to assess the quality of the navigation data, and to minimize its adverse affects on the the sonar data, MB-System™ provides an interactive navigation tool, mbnavedit.

4.7.1. MBnavedit

mbnavedit is an interactive tool that loads sonar data files, extracts navigation information, plots that information, and provides a "clickable" tool for editing the data. When swath data navigation has been loaded, mbnavedit displays auto-scaled plots of longitude, latitude, speed, heading, sonar depth time-series and a plot of the difference in time stamps between navigation values. Using these plots, one may clearly identify and correct poor fixes, a loss of heading reference, and even a loss of navigation information altogether.

Note

Navigation of deep-water hull-mounted swath data was poor before GPS and thus required adjustment as well as artifact removal to make overlapping swathes match. Nowadays, the navigation of hull mounted surveys is usually adequate from the start, but navigation of surveys from submerged platforms is almost always initially inadequate.

For surveys from hull mounted sonars, navigation is typically provided by high quality GPS, often augmented by an internal navigation system to provide increased accuracy in heading and smoothing of the fix data.

In contrast, underwater navigation systems generally involve one or a combination of:

1. Long baseline (LBL): position determined by ranging to an array of transponders.
2. Ultra-Short-Baseline (USBL): position determined by ranging to a transponder on the ship.
3. Inertial navigation system (INS): position determined by integrating motion over time using all available data, including acceleration measurements.

There are generally three types of modifications made to the navigation:

1. *Editing:* Under most circumstances, the navigation is merely edited. Isolated, obviously wrong values are selected and then replaced by interpolation or by adoption of alternate values (e.g. Course made good for heading or speed made good for speed). Because navigation systems provided to modern hull mounted sonars typically provide very good navigation, a cursory removal of outliers using the direct editing method is usually all that is required.
2. *Modeling:* If the position values are totally useless, but good heading and speed values are available, the Dead Reckoning navigation modeling function provided in mbnaveditcan generate a reasonable and self consistent approximation to the navigation. This is sometimes required for deep-towed sonar navigation. If the navigation values are very, very noisy but follow a correct navigation, then the smooth inversion navigation modeling function can generate reasonable navigation that is consistent with the original values. This method is frequently used with USBL-derived ROV navigation.
3. *Adjustment:* If the navigation is reasonable, but features don't match where bathymetry and sidescan swathes overlap, then the navigation needs to be adjusted so that these features do match. mbnavadjust is typically used to accomplish this as will be discussed in another section. This step will be required with old, pre-GPS deep water hull mounted data and with modern data from submerged platforms (sometimes even with LBL navigation).

Complicating the sonar-data--navigation-integration problem is that various sonar systems use differing methodologies for storing navigation and time information in their native data files. Some sonar data formats store a navigation record and time stamp with each ping. "Synchronous" data files of this type are the standard for MB-System™ defined formats. Other, sonar data formats, for example those created by Simrad sonars, do not store navigation information with the sonar data files at all. Instead, separate files are recorded for navigation information from up to three sources. Typically recorded at regular intervals, these "asynchronous" data records require interpolation of time and navigation information to the ping times. Still other data formats, such as those produced by Sea Beam 2100 family of sonars produce both synchronous navigation values (associated with each ping), and separate files with asynchronous navigation (at regular intervals).

Caution

Since data file formats such as Format 56 (native Simrad data) have no internal place to put navigation data, converting from other file formats to Format 56 will result in a loss of the navigation data.

To make sense of all of this MB-System™ identifies a "primary" navigation source for each data format type. When synchronous (associated with each ping) navigation data is available it is used. For asynchronous data files, a primary navigation source is chosen and per-ping positions are interpolated from their corresponding time stamps. These distinctions are worth remembering when using mbnavedit. For example, consider the time difference plot which plots differences in successive time stamps for each recorded position. When navigation information is recorded with each ping (synchronously), differences in time intervals correspond to differences in the ping rate. Most sonars automatically adjust the ping rate based on water depth, so one would expect this plot to produce a smooth varied plot (perhaps with jumps where ping rates have been manually manipulated) [Bet you don't have a log of THAT!]. In contrast, when navigation information is recorded separately at regular intervals, one would expect this plot to be, well, regular (constant).

Another distinction worth keeping in mind is the difference between navigation generated from a shipboard sonar and that generated from a transponder or inertial navigation system aboard a towed system, ROV or AUV. In the former, navigation is provided largely by GPS or some other similar navigation source. As such, navigation editing is done to the position data itself, and typically only old sonar data contains navigation so poor that one must smooth through noisy navigation. In contrast, navigation from transponders and inertial systems is generated through integration of acceleration, speed and heading data to generate positional data. In this case, editing is primarily done to the speed and heading data, such that when a dead reckoning algorithm is applied, a more accurate position map is generated.

In addition to the positional Latitude and Longitude information, sonar data formats may (or may not) also store speed and heading. These values are recorded from separate navigation inputs to the sonar system, (in contrast to "speed-made-good" and "heading-made-good" which are calculated from the Latitude and Longitude data). If the speed and heading data are available, mbnavedit plots them, and they may be edited interactively to correct errors. Editing these values and merging them back into the sonar data, however does not change independent position data. It is only when the dead-reckoning algorithm is used to generate position information that the positions recorded for each ping are changed. Editing the "raw" speed and heading data when NOT using the dead reckoning algorithm has no affect on the sonar data grid and is generally a waste of time.

.That said, plotted in blue the heading and speed plots, are speed-made-good and heading-made-good. These curves are generated from successive position information and as such are very sensitive to outliers in the independently generated navigation. They are not directly editable data points, but are very helpful in identifying unruly position data when the dead reckoning algorithm is NOT used.

OK, enough already, let us see some examples.

4.7.1.1. Reviewing and Editing Sonar Navigation with MBnavedit

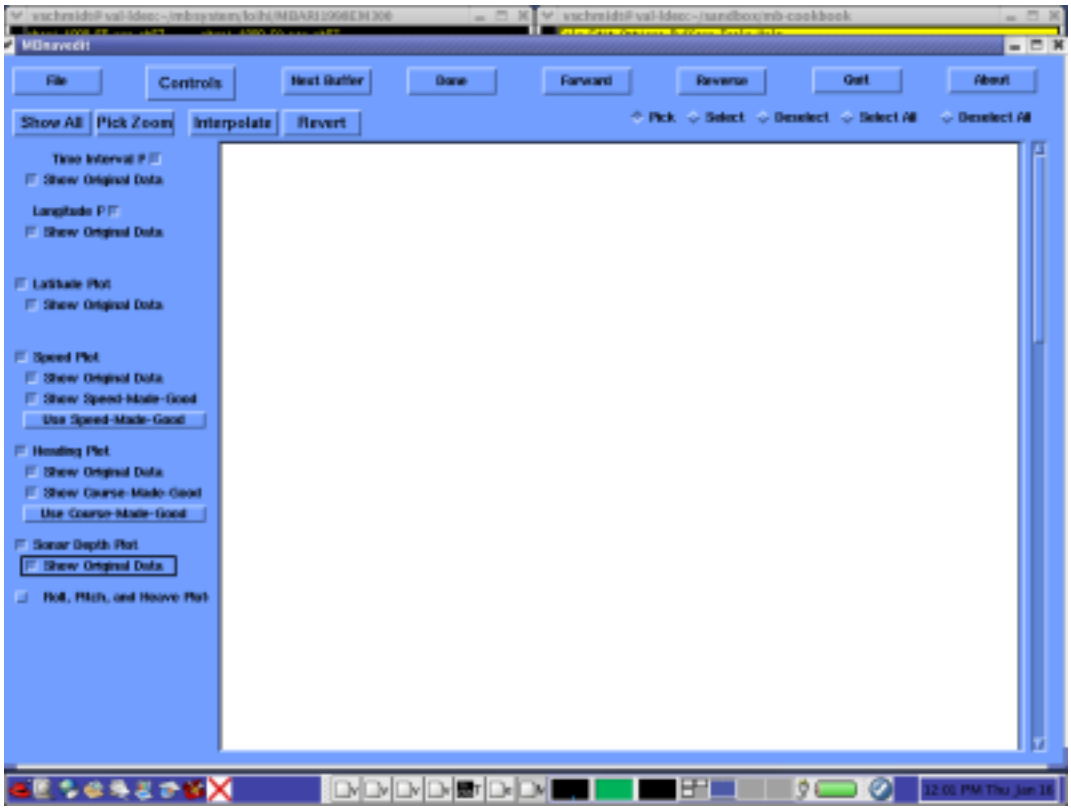
For illustration purposes, we consider two data files from the Lo'ihi example data set. These files were selected to show navigation typical of both a hull mounted sonar, and a towed sonar. The former is shown to illustrate the application, as there is little editing that is required. The latter is shown to illustrate a particularly difficult navigation problem. First we consider the hull mounted sonar data file.

4.7.1.1.1. Navigation from Hull-Mounted Synchronous Sonar Data

The mbnavedit process is executed simply by typing the following on the command line:

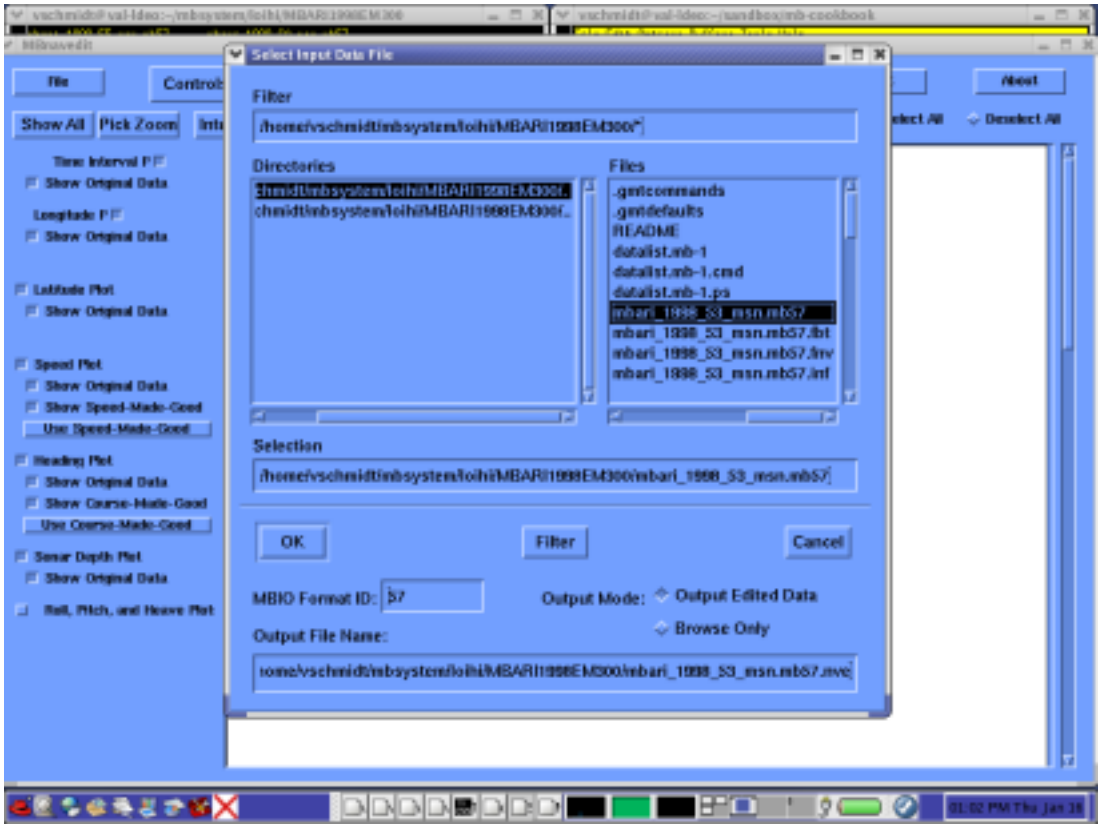
```
>mbnavedit
```

Figure 4.16. MBnavedit



To load a data set, click the "File" button. A dialog box is shown that allows one to browse to find the desired bathymetry or navigation file (as appropriate).

Figure 4.17. MBnavedit



MBIO Format ID is specified in the dialog box at the bottom, as done in mbvelocitytool. Again, this value will be filled in automatically when files are named with the MB-System™ standard *.mb##.

One may also specify, at the bottom right of the dialog box, whether the loaded data will be edited for processing or simply browsed. The former results in creation, on exit, of an edited navigation file as well as a *parameter* file that will be used by mbprocess to apply the changes. If a parameter file already exists, it will be edited to reference the newly edited navigation file.

The edited navigation file will be created with the base name of the loaded data file plus the ".nve" suffix by default. This naming convention can be changed in the bottom blank where any path and navigation file name may be specified.

If we select the first EM300 data file of our Lo'ihi data set as is shown selected in the figure above, mbnavedit will load the navigation and produce the plots.

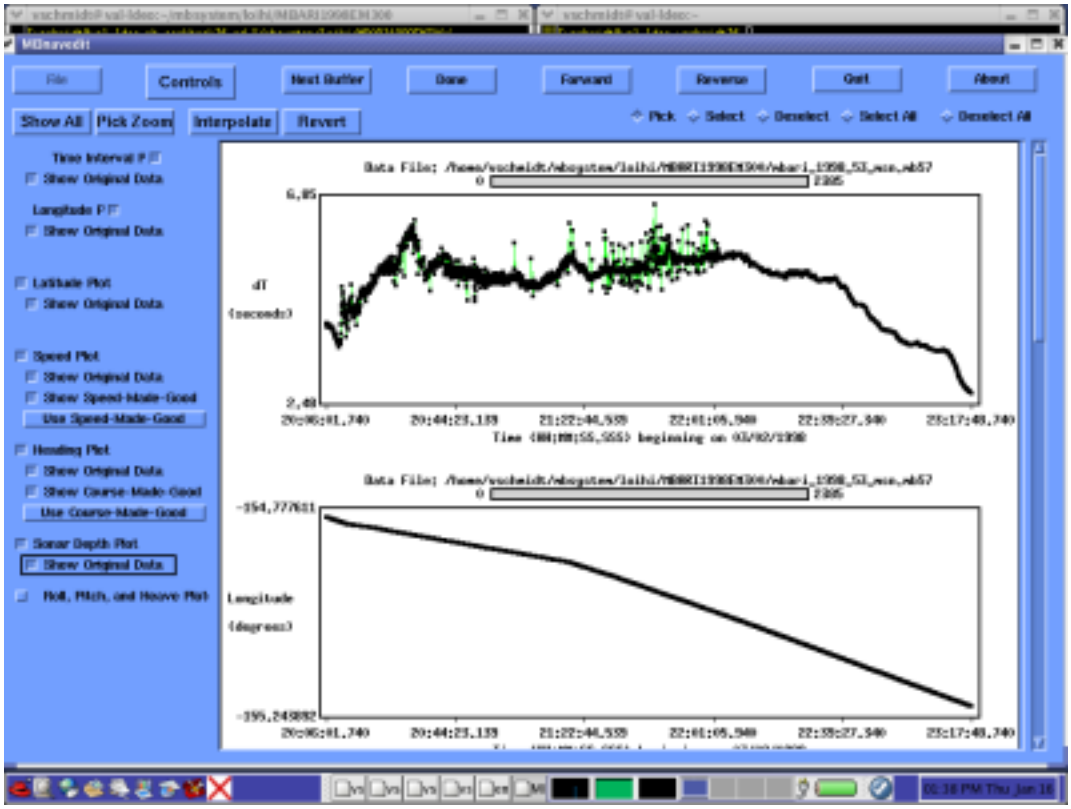
mbnavedit will load all of the navigation information in the file up to a maximum of 25000 records. In the event the file exceeds this number of records, successive chunks of data can be processed by loading successive "buffers" of data without exiting the application and reloading data.

Now lets take a quick look at the plots generated by mbnavedit.

4.7.1.1.1.1. The MBnavedit Plots

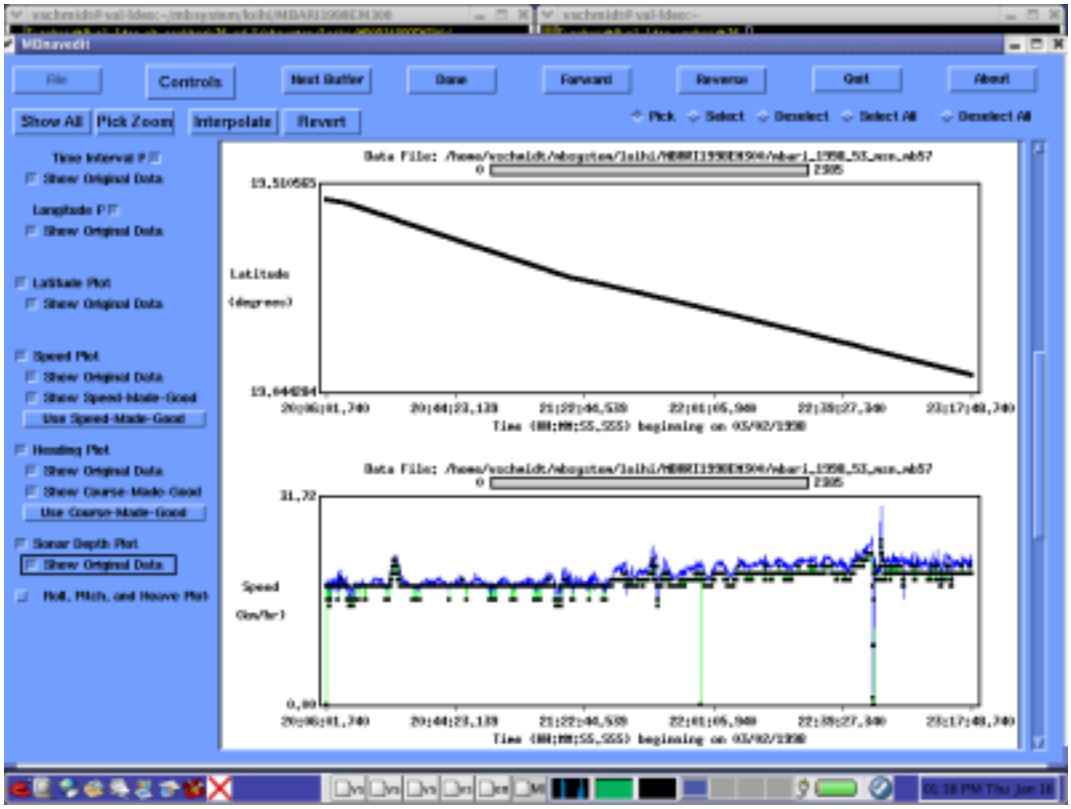
The first are the time interval plot followed by the longitude vs. time plot.

Figure 4.18. MBnavedit Time Interval and Longitude Plots



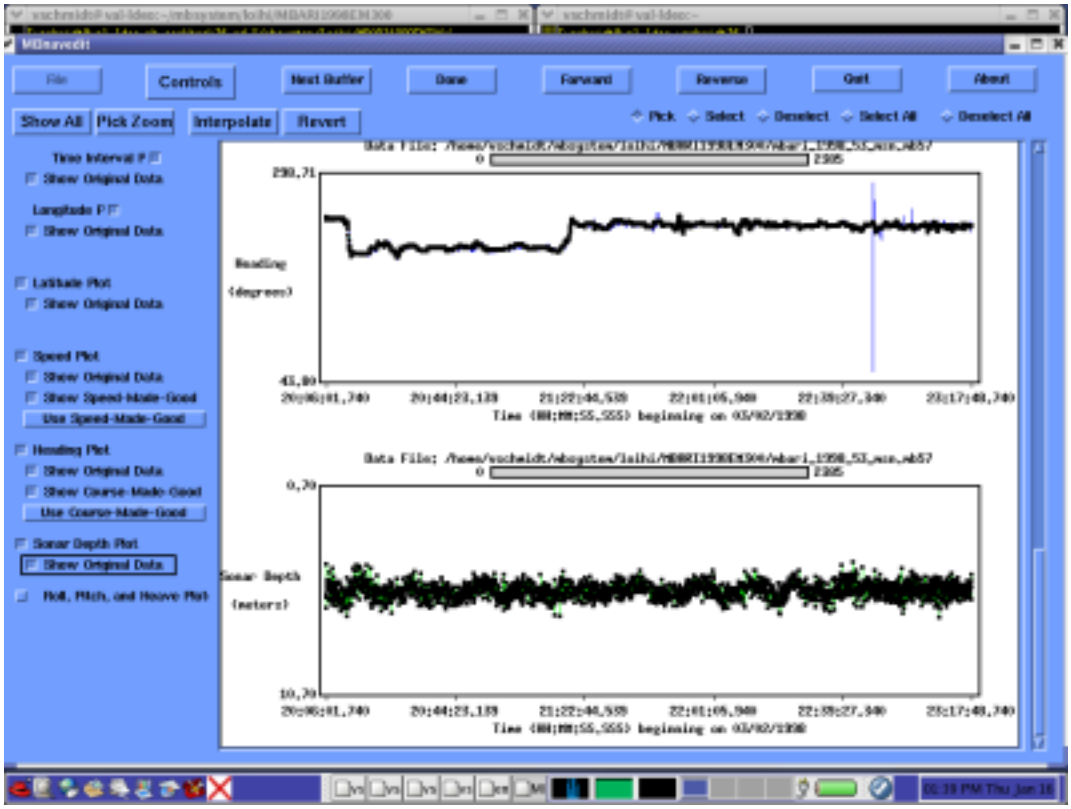
These are followed by the latitude and speed plots:

Figure 4.19. MBnavedit Latitude and Speed Plots



Scrolling further one finds the heading and sonar depth plots:

Figure 4.20. MBnavedit Heading and Sonar Depth Plots



As you scan down through the list of plots on the right you'll notice not selected, by default, are plots for pitch, roll and heave. These plots are provided for additional consideration, and can be displayed by clicking on the appropriate button. However unlike the others they are not editable (at the time of this writing). These values are used in the beam forming algorithms of the sonar system and can be helpful in determined causes of irreconcilable errors.

4.7.1.1.1.2. Editing the Navigation

As stated before, editing of navigation data requires different strategies depending on 1) whether the navigation values are from an inertial navigation system (e.g. a towed system, ROV or AUV) or a constant navigation source (e.g. GPS) and 2) whether the sonar stores time information at periodic intervals or with each ping. In the EM300 sonar data file we've chosen to illustrate MBnavedit the primary navigation source is stored synchronously, and since its a hull mounted sonar with a GPS input, we can expect the navigation to be quite good. Just the same, we can use it to illustrate the process.

Let us review the plots shown above and point a few things out.

First consider the time difference or "dT" plot. Remember that time stamps in this data set are integral to the ping data - "synchronous" if you will. Here we see a varying time difference plot that corresponds largely to changes in the ping rate. Spikes in the time interval plot typically reflect abnormal ping cycles rather than a loss of navigation data.

In this particular data file, we have quite a lot of spikes for the first half of the data file, after which the time difference values change in a smooth expected way. There is, in fact, nothing wrong with this navigation data, but these irregular changes in ping rate might be indicative of a different problem that is worth pointing out - an improper configuration of the sonar. Sonars typically require an operator to provide it with a range of depth values (or at least a minimum depth value) in which it should expect to find the bottom. If the sonar had difficulty in reconciling the ping results with this setting (i.e. the actual depth was outside the range) it might attempt to adjust the sonar ping rate frequently, resulting in corres-

ponding fluctuations in this time interval plot. This is likely the cause of the spikes in this plot. A well kept sonar operator's log might provide clues that would alert the sonar data processor for this type of behavior.

As one would expect for a modern, hull-mounted sonar, there is nothing particularly interesting in the Longitude and Latitude plots. They are smooth and continuous. Great! There is nothing to do. Not much of an example. In fact, it is difficult to find hull mounted sonars that require navigation editing. Let us next consider a towed sonar example.

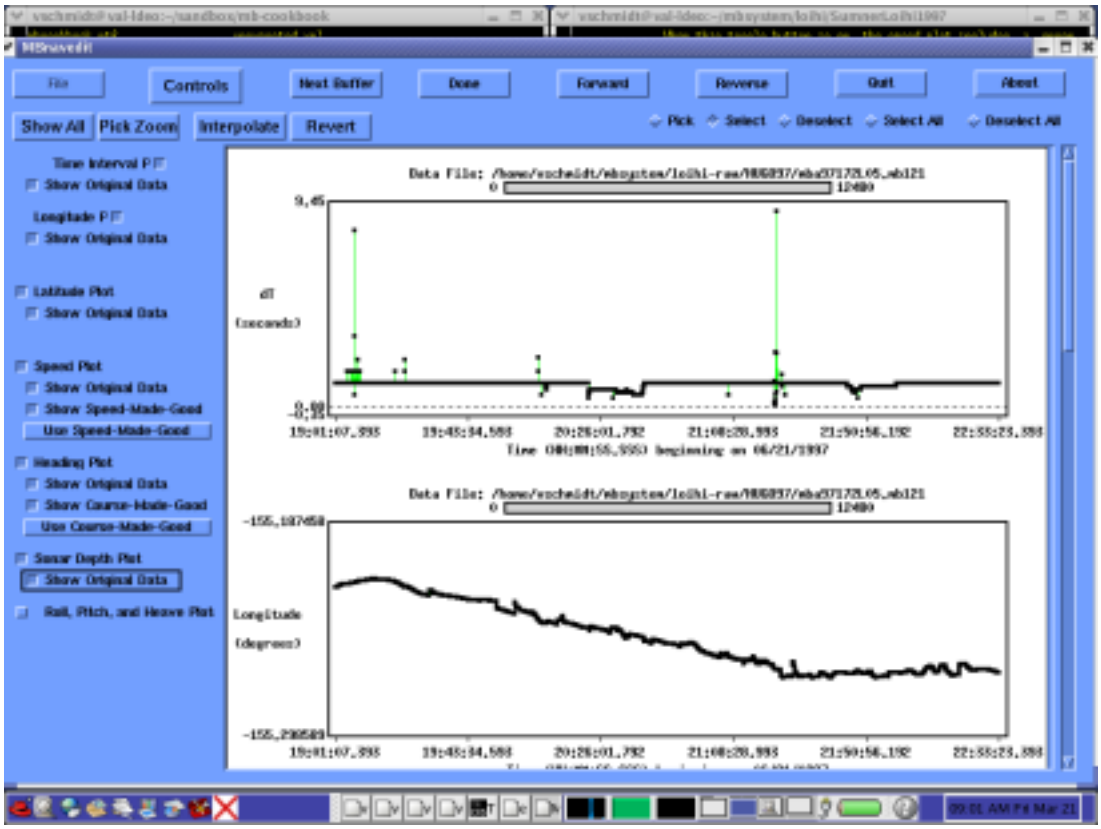
4.7.1.1.2. Navigation from a Towed Sonar

For this illustration we have selected and loaded the first data file from the "HUGO" Loi'hi data set. Let us first review the plots.

4.7.1.1.2.1. MBnavedit plots

First the Time Difference and the Longitude plots:

Figure 4.21. MBnavedit Towed Sonar Time Difference and Longitude Plots

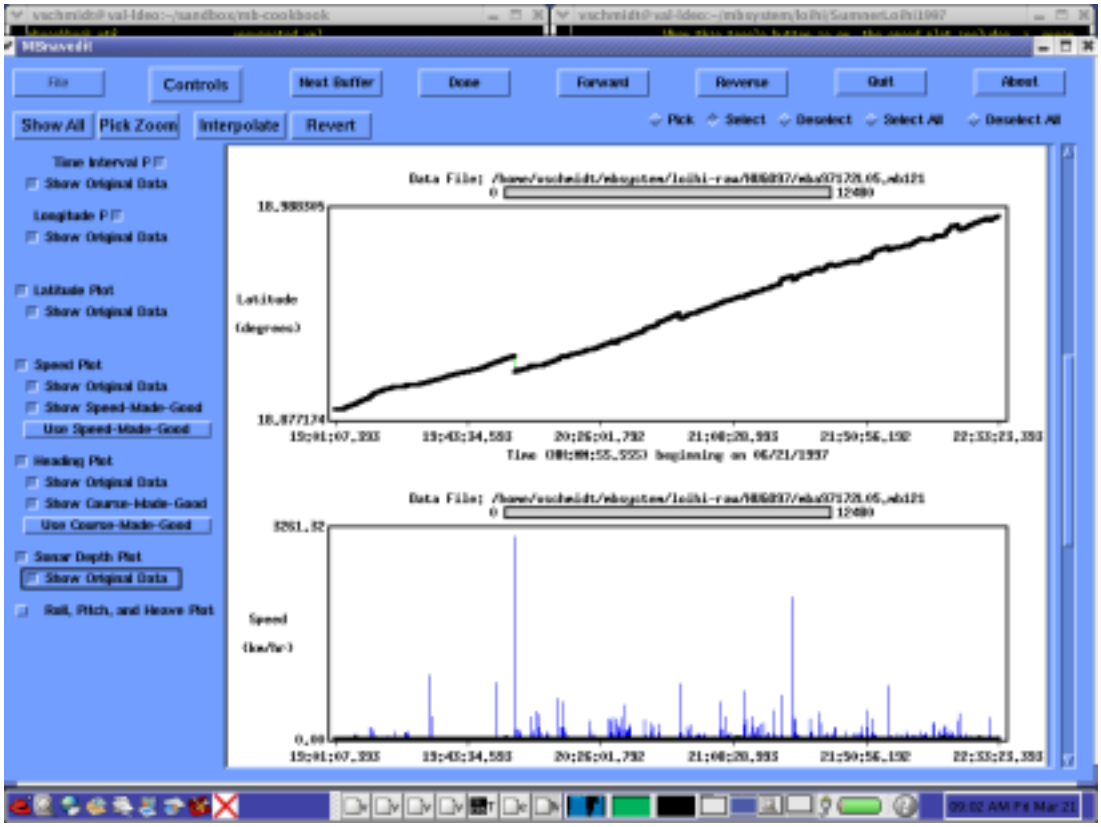


Here we see a nearly constant time difference plot. Navigation is recorded "asynchronously" and ping navigation points are interpolated from the their associated time stamps. A few spikes show blips in the recorded navigation and are inconsequential.

The Longitude plot shows both a "waviness" that we know to be incorrect and more than a few discontinuities. These will have to be addressed in our editing process.

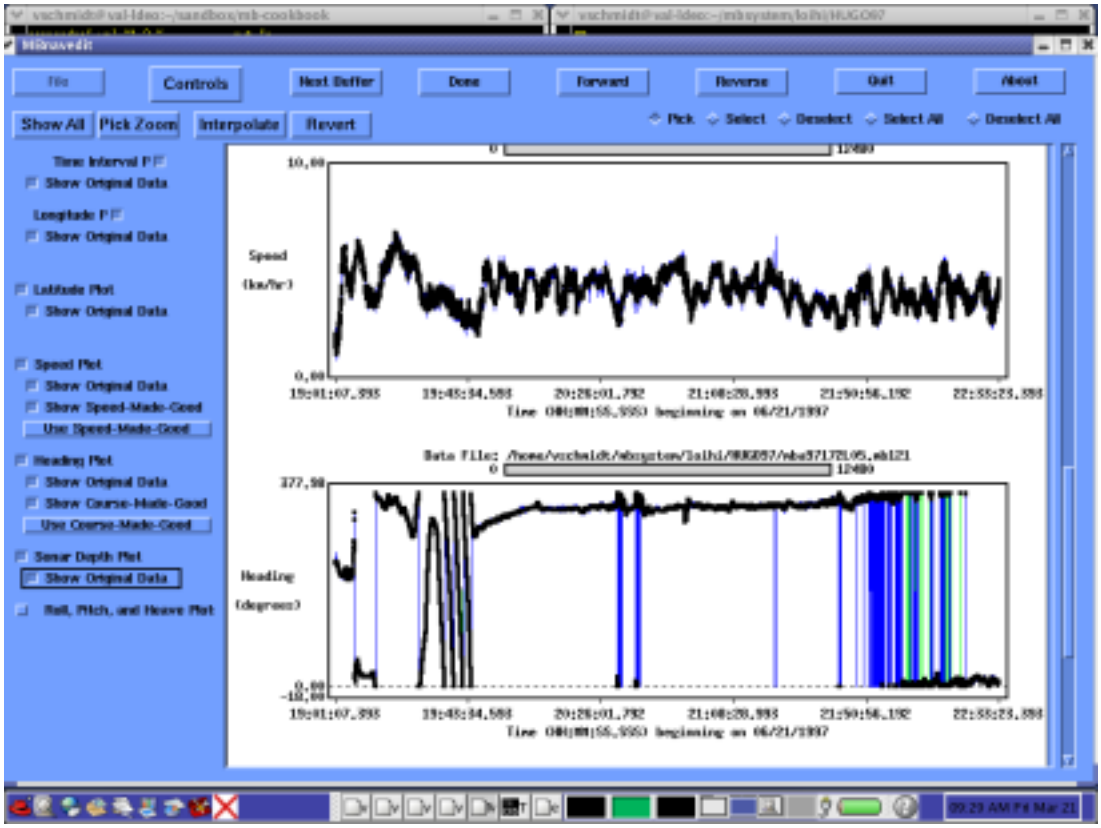
Next the Latitude and Speed plots:

Figure 4.22. MBnavedit Towed Sonar Latitude and Speed Plots



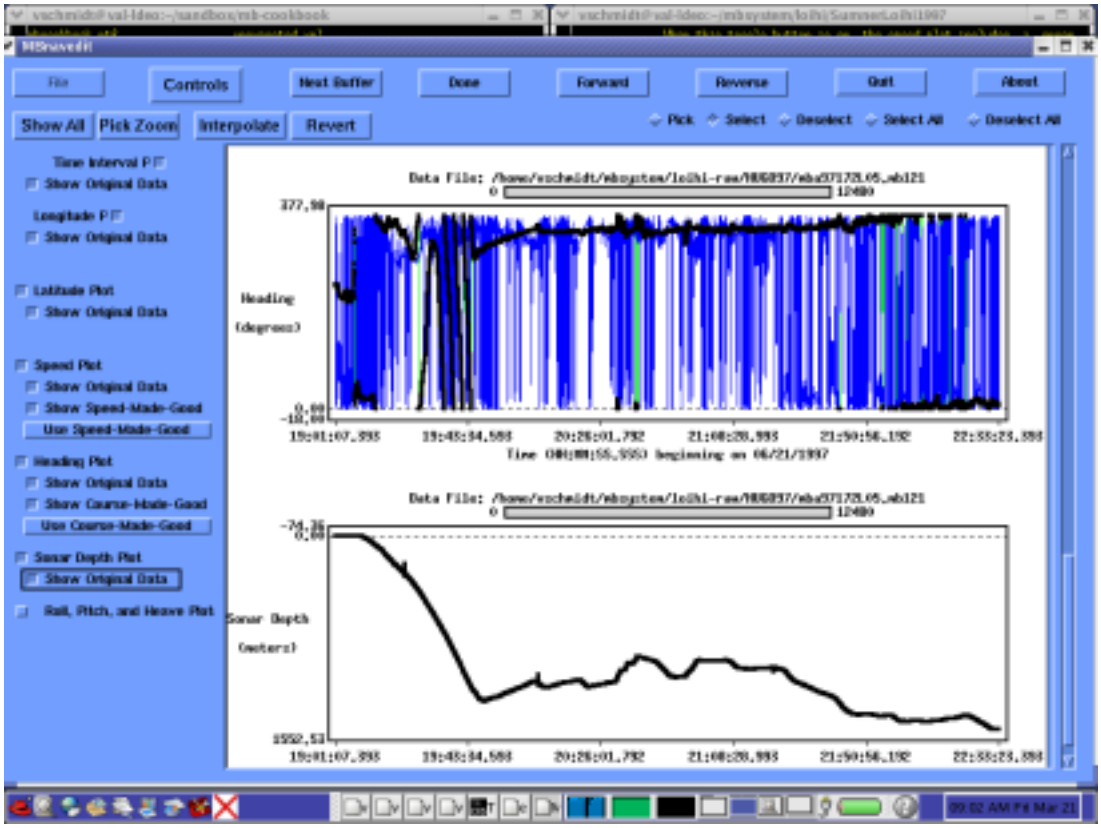
In the Latitude plot, we again see the "waviness" and discontinuities that must be addressed in the editing process. Hints to the errors can be seen in the Speed plot as well. Spikes in the "speed-made-good" plot flag erroneous position data. If we "unselect" the "Show Speed-Made-Good" button we can see that the speed values themselves are quite noisy and are most likely the source of the positional problems.

Figure 4.23. MBnavedit Towed Sonar Speed Plot



Finally the Heading and Sonar Depth plots:

Figure 4.24. MBnavedit Towed Sonar Heading and Depth Plots



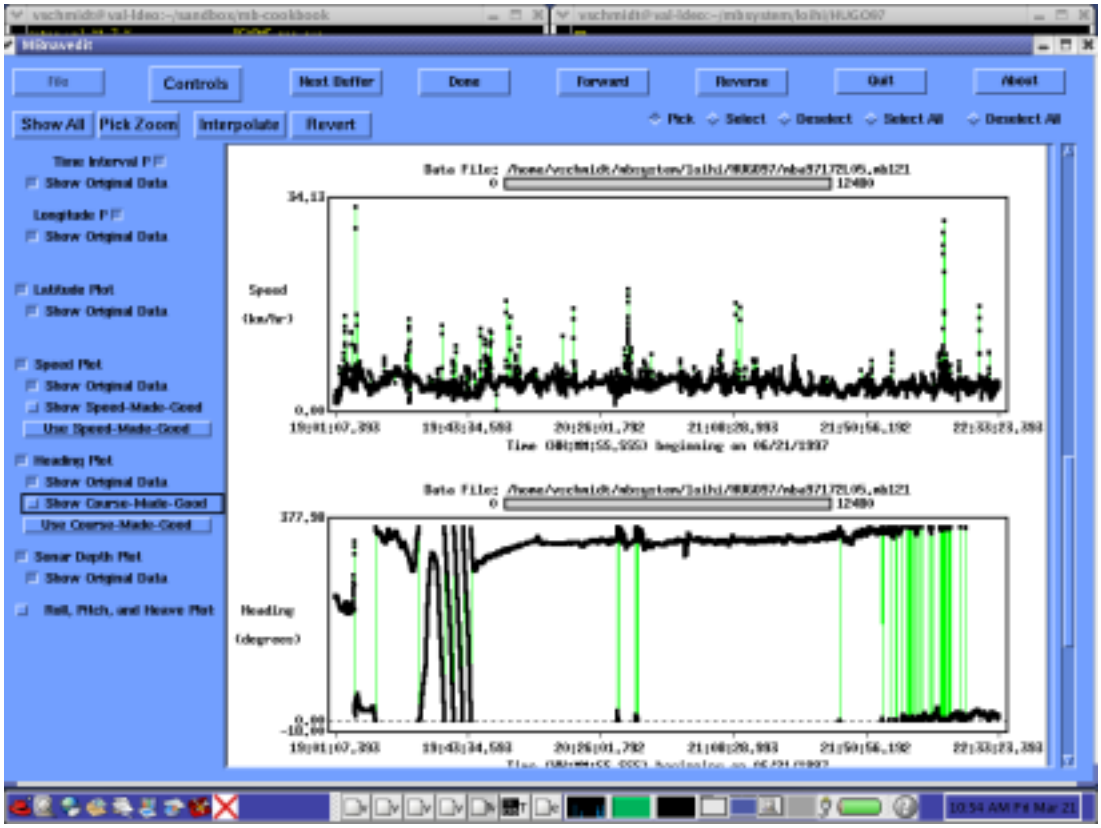
Here we can see considerable noise in the heading plot. Plus, the blue, heading-made-good plot is a jumbled mess, due to the noise in the position data that we noted earlier. The depth plot is smooth and reasonable and need not be considered at this time.

4.7.1.1.2.2. Editing the Navigation and Applying The Algorithms

For a towed sonar of this type, with such poor position data, the general strategy is to edit the speed and heading data, and then recalculate the position data using the Dead Reckoning Algorithm.

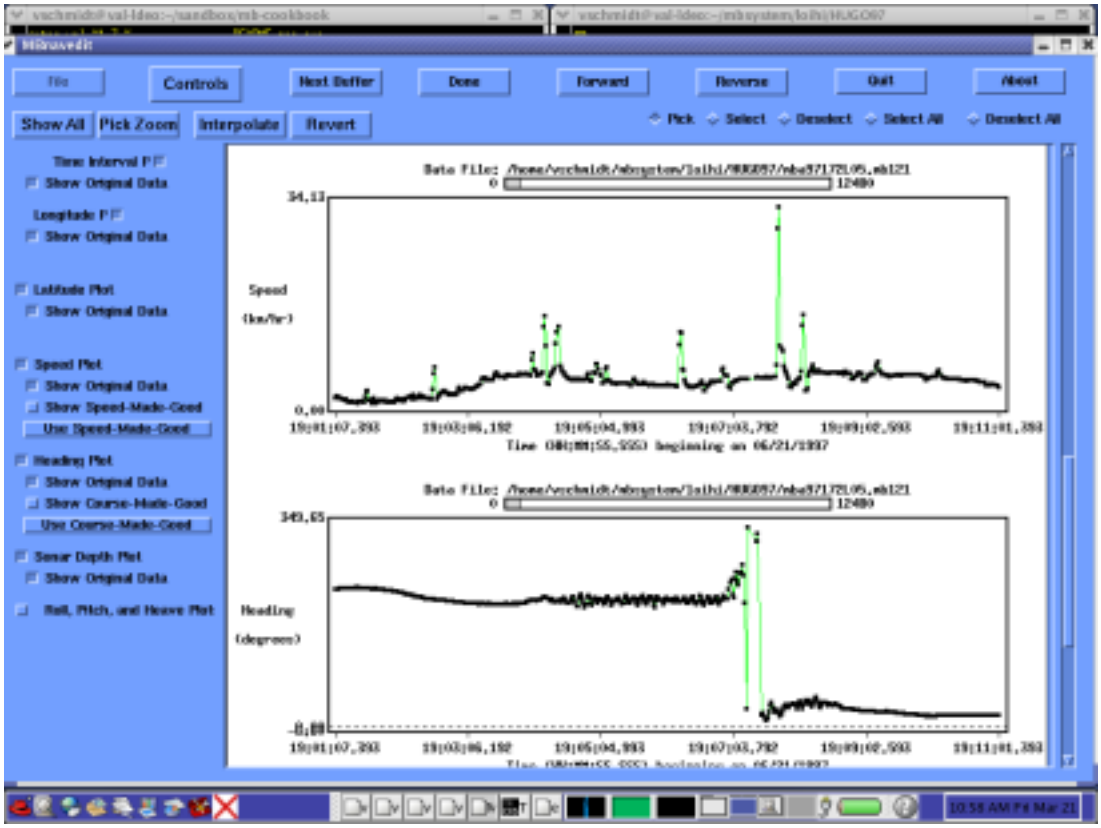
First to see the data more clearly, we can remove the speed-made-good and heading-made-good lines from the plot. Simply deselect the "Show Speed-Made-Good" and "Show Heading-Made-Good" toggle boxes on the left.

Figure 4.25. MBnavedit Towed Sonar Heading and Depth Plots with "Made-Good" plots Removed



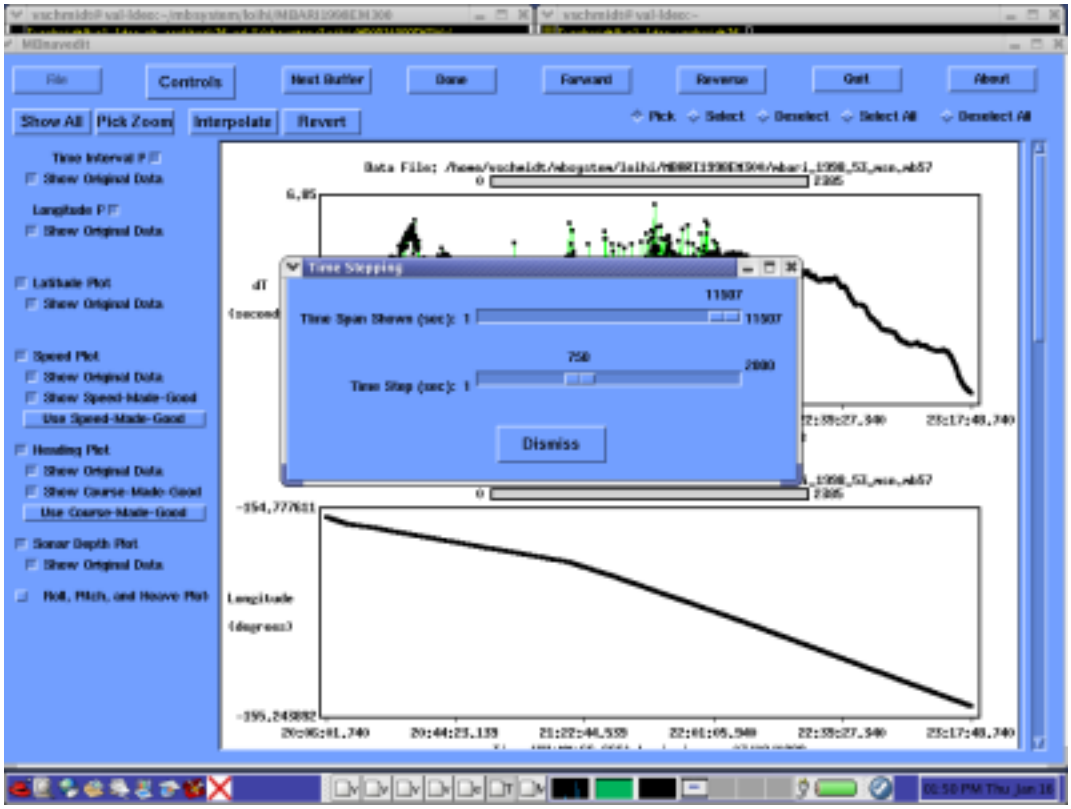
We can inspect the speed and heading plots more closely using the "Pick Zoom" feature. Clicking the "Pick Zoom" button in the upper left hand corner of the screen changes the cursor to a "+". One may then position the cursor over the plot, aligning it with the left edge of the desired area to zoom and clicking the left mouse button. A red line denotes the edge of the area to zoom. Then one may position the cursor over the right edge of the desired area and click the middle mouse button. A second red lines denotes the right boundary. Positioning the cursor between the two red lines and clicking the right mouse button will zoom the plot to the selected boundaries.

Figure 4.26. Speed and Heading Plots Zoomed



Alternatively, one can manually adjust the horizontal scale. A click of the "Controls" button brings up a drop down menu with "Time Stepping", "Nav Modeling" and "Time Interpolation" options. Selecting "Time Stepping" brings the following dialog:

Figure 4.27. MBnavedit Heading and Sonar Depth Plots



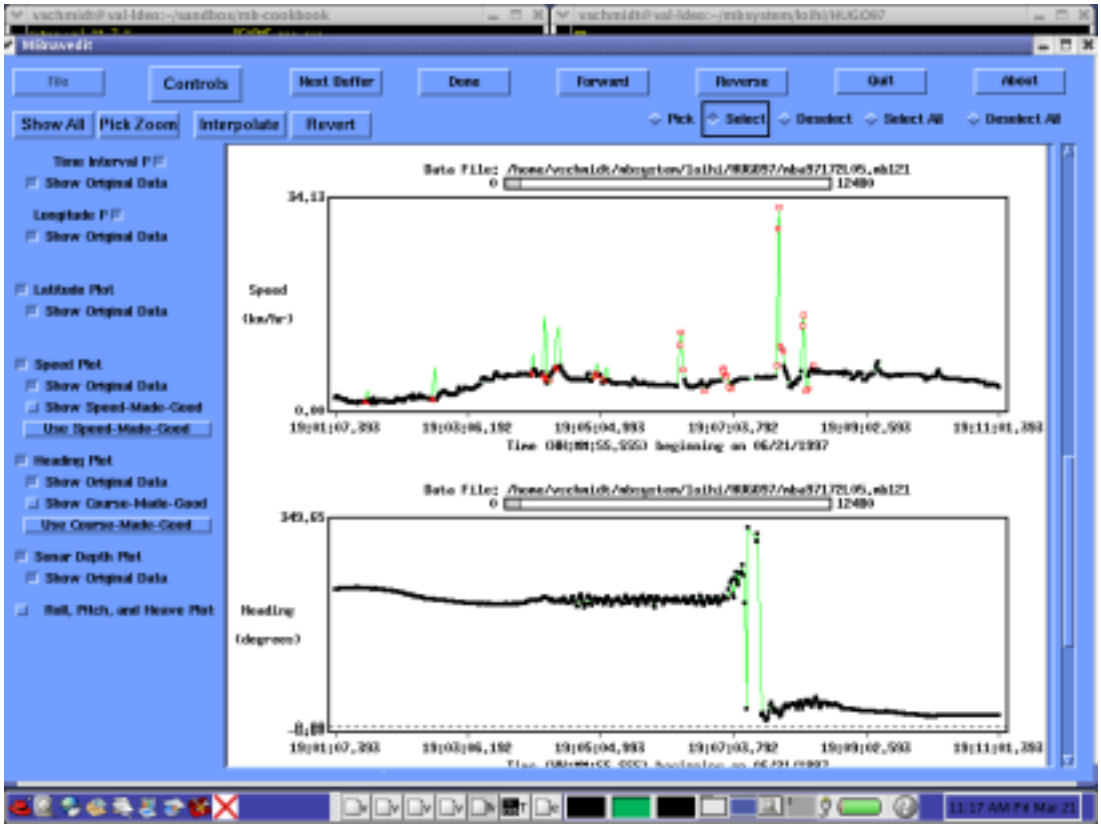
Here the "Time Span Shown" slider allows one to adjust the amount of data that will be plotted in each plot by choosing the time span during which data will be plotted. The default is to plot the entire data set, however for even moderately sized data files this makes editing difficult and one will usually want to adjust this value to a smaller value. Note that this does not translate into the number of data points in any plot, as the ping interval is longer than a second in this data set.

As one "time span" of data is edited one will want to move forward to successive time spans to edit them as well. The shifting of the plots is done by clicking the "Forward" button in the main window, and the "Time Step" slider in this window allows one to specify the amount the plots are shifted with each click. A good rule of thumb is to select a value of about 2/3 of the "Time Span" setting to allow successive shifts in the plot to overlap.

One can now click the "Forward" and "Reverse" buttons in the main window which has the effect of "scrolling" through the plots in time. A white bar with a gray box allows one to see where the current plot is in the larger data set.

To edit the speed and heading data, we simply select the point or points we wish to edit and click the "Interpolate" button. The original data points are removed and new data points are shown whose values are interpolated from the surrounding remaining data. When "Pick" is selected (at the top right of the mbnavedit window), data points are selected one click and one point at a time. When "Select" is chosen,all the points in the general area of the cursor are chosen and one may "mow" over points by moving the cursor with the left mouse button depressed, selecting everything in the cursor's path. The following illustration shows several points on the speed plot that have been selected and interpolated (on the left) and several more that have been selected, but the Interpolate button has not yet been pressed.

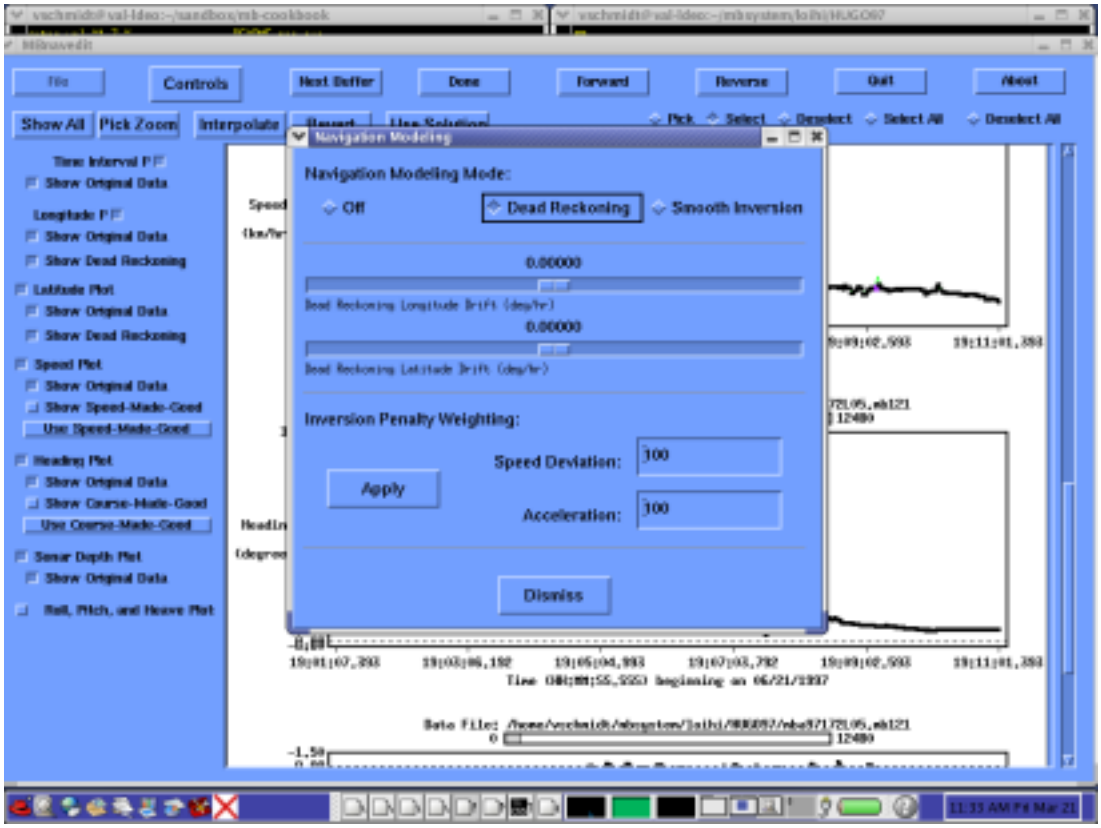
Figure 4.28. Interpolated Points



In this way, one continues through the speed and heading plots, removing outliers and smoothing through noisy data. When the process is complete, we then apply the Dead Reckoning Algorithm.

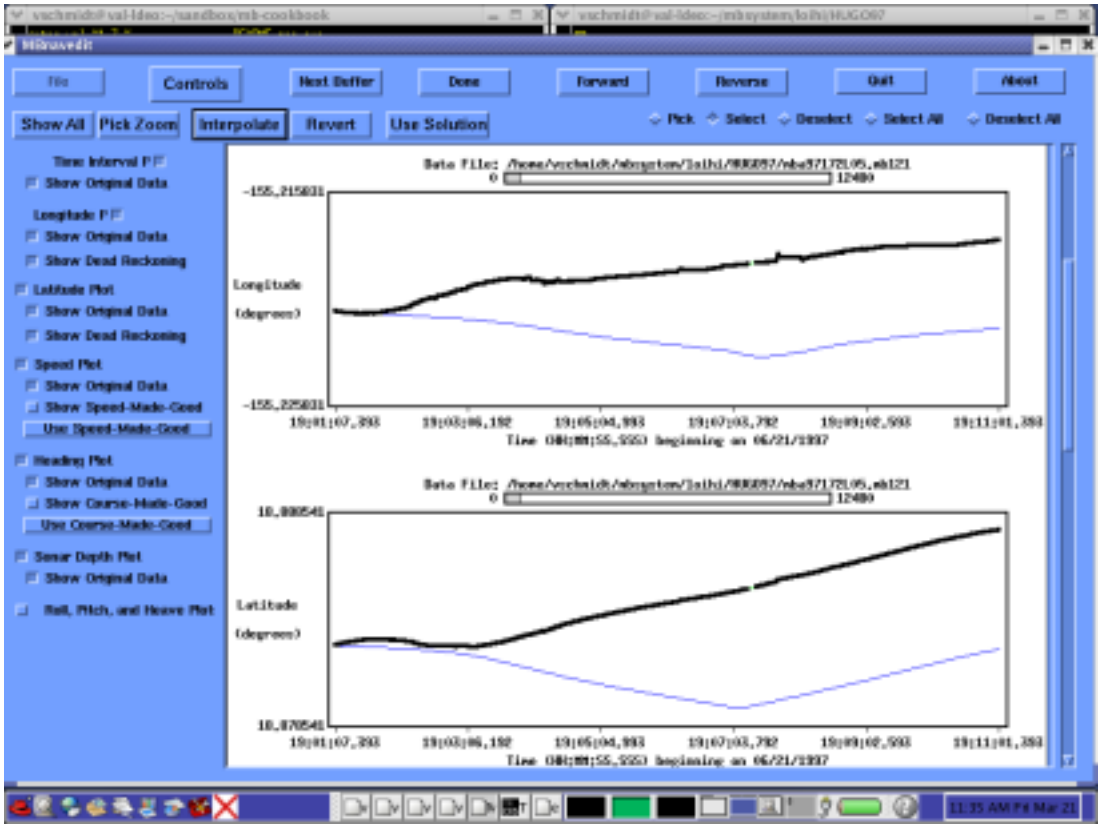
To apply the Dead Reckoning Algorithm, select "Controls" menu, and then "Nav Modeling". At the top of the resulting window, one chooses the kind of Nav Modeling desired, or none at all. If Dead Reckoning was selected, once can select values for Latitude and Longitude drift rates, if known. If Smooth Inversion was selected, one can set weighting factors for speed and acceleration. In this case, we select Dead Reckoning. We will leave the drift rate values blank for this example, HOWEVER, operators of AUV or ROV based sonar systems must maintain a log of local currents such that these corrections can be applied.

Figure 4.29. Nav Modeling Window



When "Dismiss" is selected to close the window, the "solution" generated by the Dead Reckoning algorithm is plotted in blue on the Latitude and Longitude plots.

Figure 4.30. Dead Reckoning Position Plots



Also shown in the mbnavedit is a "Use Solution" button. Clicking this button tells mbnavedit to save the Dead Reckoning solution when exiting such that it may be used as the primary navigation source by mbprocess.

4.8. Flag Erroneous Bathymetry Data

Now that we have done some preliminary work regarding SSP's, physical constants for the sonar system, and navigation corrections, we can begin automated and interactive editing of the bathymetry data itself. Editing of bathymetry in MB-System™ consists simply of "flagging" data points that are erroneous such that they can be ignored. In this way, tools that will later grid the data set, can interpolate from the surrounding data without the interference of poor data points.

It may seem odd that MB-System™ does not allow the user to manually manipulate data points. One cannot look at a piece of sonar data with $mbs; and create one's own sea floor profile through an otherwise noisy data set, as can be done with some sonar data editing systems. Rather, one can only chose existing points to ignore, and allow standard interpolation algorithms to fill in the blanks. The distinction is a subtle, but important difference in philosophy regarding sonar processing on the part of the authors of $mbs;.

Sonar editing, whether automated or manual, creates an ancillary data file, typically with an ".esf" suffix. This binary file contains the list of the data points flagged in the editing process. The depths associated with these data points are given negative values (or zero) during the final processing such that GMT will disregard and interpolate around them.

$mbs; editing processes also check to see if an associated parameter file exits (with the ".par" suffix). If not, one is created. In either case, the parameter file is made to specify the file name of the edits ancillary file, so that it may be applied by mbprocess.

First let us consider automated flagging of Bathymetry.

4.8.1. Automated Flagging of Bathymetry

Lucky for us, many errors in multibeam sonar systems are predictable, such that we can use an automated tool to flag them as incorrect. Toward that end, MB-System™ provides the process mbclean.

mbclean is an extremely useful tool to automate a portion of the bathymetry editing process. Often mbclean can be used in automated scripting to modify sonar data in near real-time such that reasonable quality data can be produced right from the get-go. However, even when processing data from hull mounted, modern sonars, no automated algorithms can approach the discretionary ability of the human eye. Interactive editing with mbedit is recommended to augment the automated editing for publication quality data sets.

mbclean provides seven ways to automatically flag bad beams in the bathymetry data. In the order they are applied, they are:

1. Flag specified number of outer beams. (-X option)
2. Flag soundings outside specified acceptable depth range. (-B option)
3. Flag soundings outside acceptable depth range using fractions of local median depth. (-G option)
4. Flag soundings outside acceptable depth range using deviation from local median depth. (-A option)
5. Flag soundings associated with excessive slopes (-C option or default)
6. Zap "rails". (-Q option)
7. Flag all soundings in pings with too few good soundings. (-U option)

The first flags a specified number of outer most beams from each side of the sonar. These are commonly poor beams, as SNR is low and sound speed profile errors are magnified.

The second flags sounding outside a specified depth range. This can be helpful when, for example, the sea floor is relatively flat and the range of depths is well known.

Next mbclean optionally flags beams with depths that are outside bounds generated by some specified fraction of the local median depth. This is helpful, when the sea floor is not particularly flat, and depth range is not well known. The local median depth is generated as the median of the current beam and those immediately before and after it.

Then mbclean flags beams that are outside some exact specified range of the local median depth. This option is usually preferable to the previous one when the total depth is quite deep, resulting in larger than desirable bounds using the previous method.

The next method employed by mbclean is flagging based on excessive slopes. This is actually the most common way to identify faulty bathymetry. Four optional methods of flagging beams based on slope are provided. The first flags the beam that is furthest from the local median depth. The second flags both beams associated with the excessive slope. The third method zeroes the beam furthest from the local median depth rather than flagging it. The final method zeroes both beams associated with the excessive slope.

If specified, mbclean will next try to identify groups of along track beams that have abnormally short across track distances and shallow depths. These kinds of errors cause ridges or "rails" along the track of the ship.

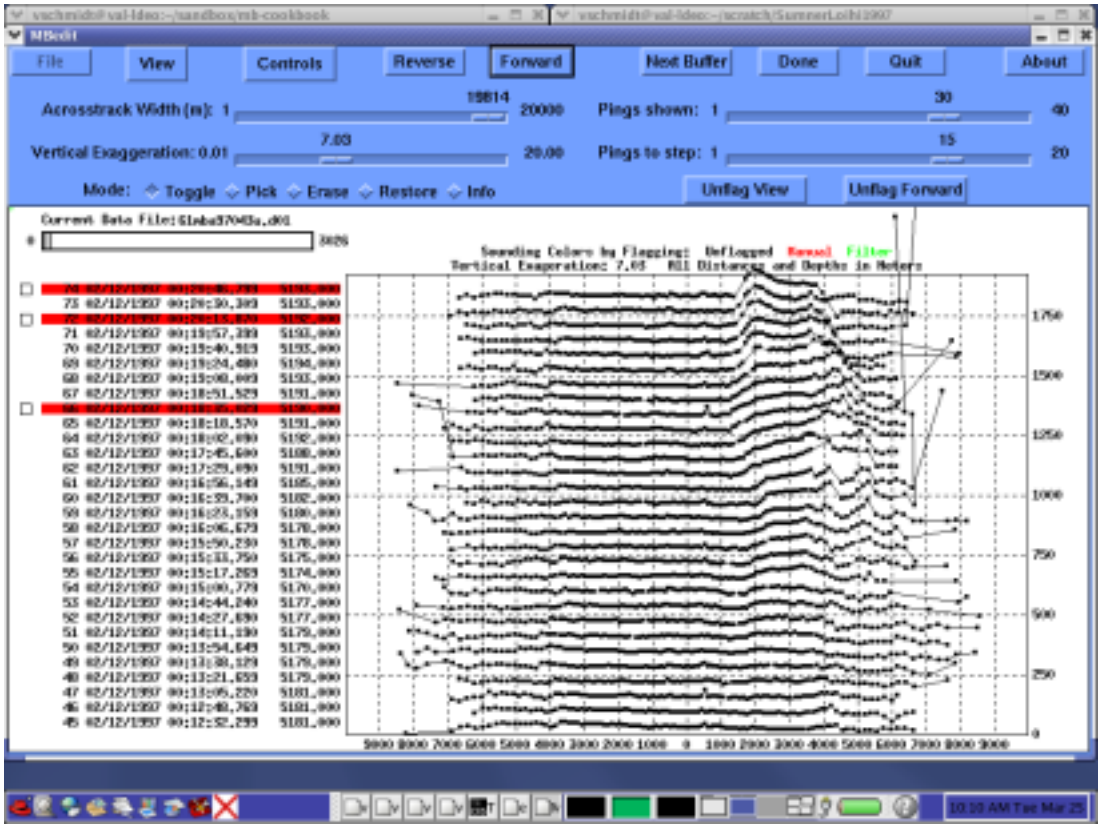
Finally, mbclean will toss out entire pings whose quality is so poor that flagging by other methods has resulted in fewer than a specified number of good beams.

4.8.1.1. An MBclean Example

We can apply mbclean to a portion of our Loi'hi data set and see how it works. mbedit is such a nice tool for visualizing (and editing as we will see) sonar data, we can use it in this example to see the effects of mbclean. However, the details regarding mbedit are saved for the next section, so for now just bare with me and look at the pictures.

Here is a screen-shot of mbedit with the first data file from the SumnerLoihi data loaded. All flagging of data has been removed the scaling has been adjusted and the data has been scrolled such that erroneous data is easy to make out.

Figure 4.31. Sumer Loihi Unedited Data



In the screen shot above it is easy to see data that needs editing. In particular, spikes in the outer few beams on either side are apparent. Now lets see if we can clean things up a bit with mbclean.

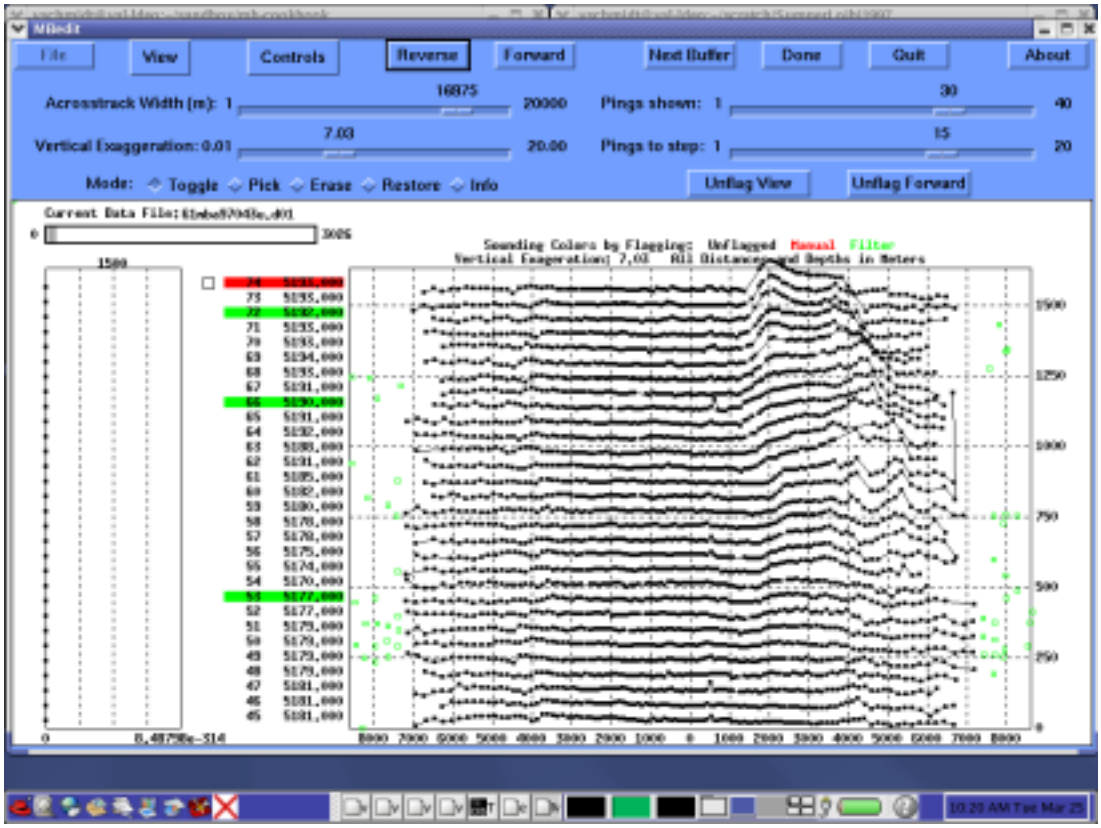
Normally this would be done in a single step, but for the purposes of illustration we can do them separately to see the results. First let us run mbclean to flag the outer five beams on either side of the swath. This we do with the *-X* flag.

```
mbclean -F121 -I 61mba97043u.d01 -X5

Processing 61mba97043u.d01
```

```
3026 bathymetry data records processed
14613 outer beams zapped
0 beams zapped for too few good beams in ping
0 beams out of acceptable depth range
0 beams out of acceptable fractional depth range
0 beams exceed acceptable deviation from median depth
0 bad rail beams identified
0 excessive slopes identified
14613 beams flagged
0 beams unflagged
0 beams zeroed

MBclean Processing Totals:
-------------------------
1 total swath data files processed
3026 total bathymetry data records processed
0 total beams flagged in old esf files
0 total beams unflagged in old esf files
0 total beams zeroed in old esf files
14613 total outer beams zapped
0 total beams zapped for too few good beams in ping
0 total beams out of acceptable depth range
0 total beams out of acceptable fractional depth range
0 total beams exceed acceptable deviation from median depth
0 total bad rail beams identified
0 total excessive slopes identified
14613 total beams flagged
0 total beams unflagged
0 total beams zeroed
```

From the text output, we can see that there were 3026 pings in this data file (each has some 151 beams), and that as a result of mbclean, 14613 beams are now flagged. We can take a look at the file again in mbedit and see what has been flagged.

Figure 4.32. Sumer Loihi Data with Outer Beams Flagged from MBclean



Above we see that the much of the noise in the outer beams has been removed by mbclean.

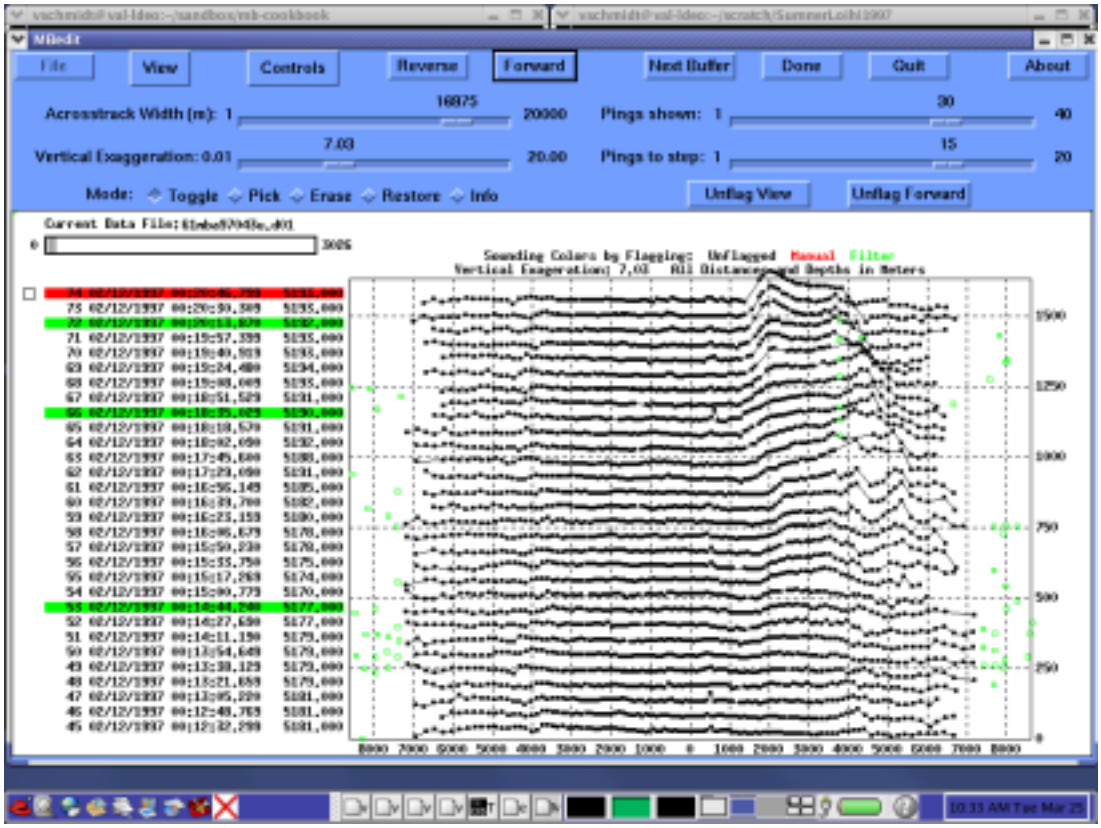
Perhaps the most common way to use mbclean to automatically flag bathymetry data is to employ the slope checking algorithm. This method is common because sea floor slope is relatively predictable across a wide array of total sea depths. Flagging based on depth limits, set as a fraction or constant portion of the local median depth, often do not produce good results across a large change in the depth of the sea floor, and require flagging in a piece meal way. Slope flagging is specified with the *-C* flag to mbclean as shown below.

```
mbclean -F121 -I 61mba97043u.d01 -C1

Processing 61mba97043u.d01
Sorting 14613 old edits...
10000 of 14613 old edits sorted...
3026 bathymetry data records processed
14613 beams flagged in old esf file
0 beams unflagged in old esf file
0 beams zeroed in old esf file
0 outer beams zapped
0 beams zapped for too few good beams in ping
0 beams out of acceptable depth range
0 beams out of acceptable fractional depth range
0 beams exceed acceptable deviation from median depth
0 bad rail beams identified
66280 excessive slopes identified
80893 beams flagged
0 beams unflagged
0 beams zeroed

MBclean Processing Totals:
```

```
-------------------------
1 total swath data files processed
3026 total bathymetry data records processed
14613 total beams flagged in old esf files
0 total beams unflagged in old esf files
0 total beams zeroed in old esf files
0 total outer beams zapped
0 total beams zapped for too few good beams in ping
0 total beams out of acceptable depth range
0 total beams out of acceptable fractional depth range
0 total beams exceed acceptable deviation from median depth
0 total bad rail beams identified
66280 total excessive slopes identified
80893 total beams flagged
0 total beams unflagged
0 total beams zeroed
```

Here we see that mbclean initially processed the flagging we had already applied to the data set. Then the new rules based on a slope of 1 were applied. We can see the results in the following screen shot of mbedit.

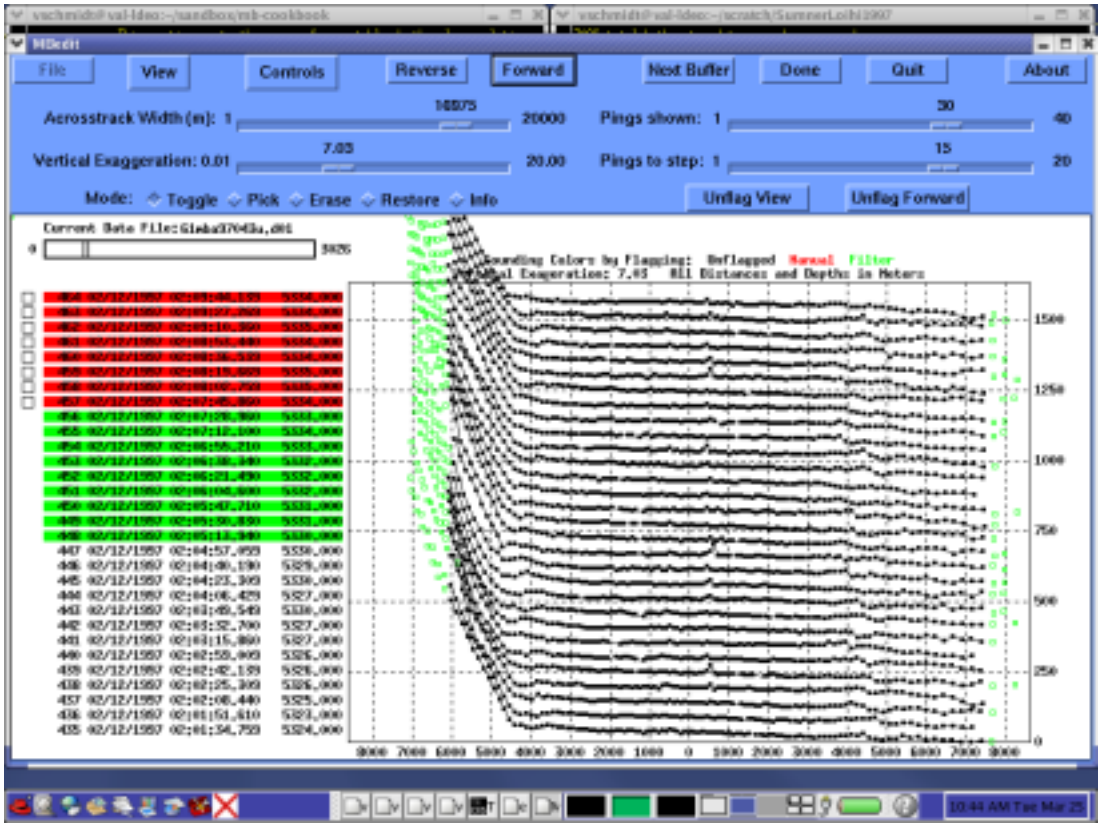
Figure 4.33. Sumner Loihi Data with Outer Beams and Slope Greater than 1 Flagged from MBclean



Much is the same as before, but close examination of the small ridge in the upper right hand corner of the data window shows that a few new points have been flagged.

Now there is an error in what we have done in this last step that has been left to illustrate a point. The slope limit of 1 we have chosen in this last step is actually too small (and would be for most data sets). It is not particularly apparent in the screen shot above, however further into the data set, as the Lo'ihi sea mount is approached, the sea floor rises dramatically. The slope limit of 1 flags otherwise reasonable data. Take a look at the screen shot below.

Figure 4.34. Sumner Loihi Data with Slope Greater than 1 Flagged and the Loss of Good Data



There is a fine line between aggressively flagging of poor data and accidentally throwing out the baby with the bath water. Any setting will suffer from either missing clearly erroneous bathymetry, or flagging data that is actually quite good. It is up to the data processor (YOU!) to assess the data set and choose flagging algorithm parameters wisely. This general method, of trying various slopes or other algorithm parameters with mbclean, and then inspecting the results through the interactive editor, is a good way to quickly get a feel for what parameters might be most appropriate. Experience will bring a better intuition for what works best, and there is comfort in knowing that nothing is permanent. One can always remove the flagging, using mbunclean, and start anew. Again we see that mbclean cannot completely take the place of interactive editing with mbedit.

4.8.2. Interactive Flagging of Bathymetry

One of the most useful tools provided by MB-System™ is mbedit. It provides an easy-to-use interface for the interactive editing of sonar data, and several built-in algorithms for the semi-automated editing of data (similar to mbclean). What is more, numerous plots of sonar inputs such as heading, speed, center beam depth, pitch, roll and many others can be viewed right along with the sonar data. The utility of this feature cannot be under stated, as it can be extremely helpful in understanding the causes of poor sonar

data and how to fix them.

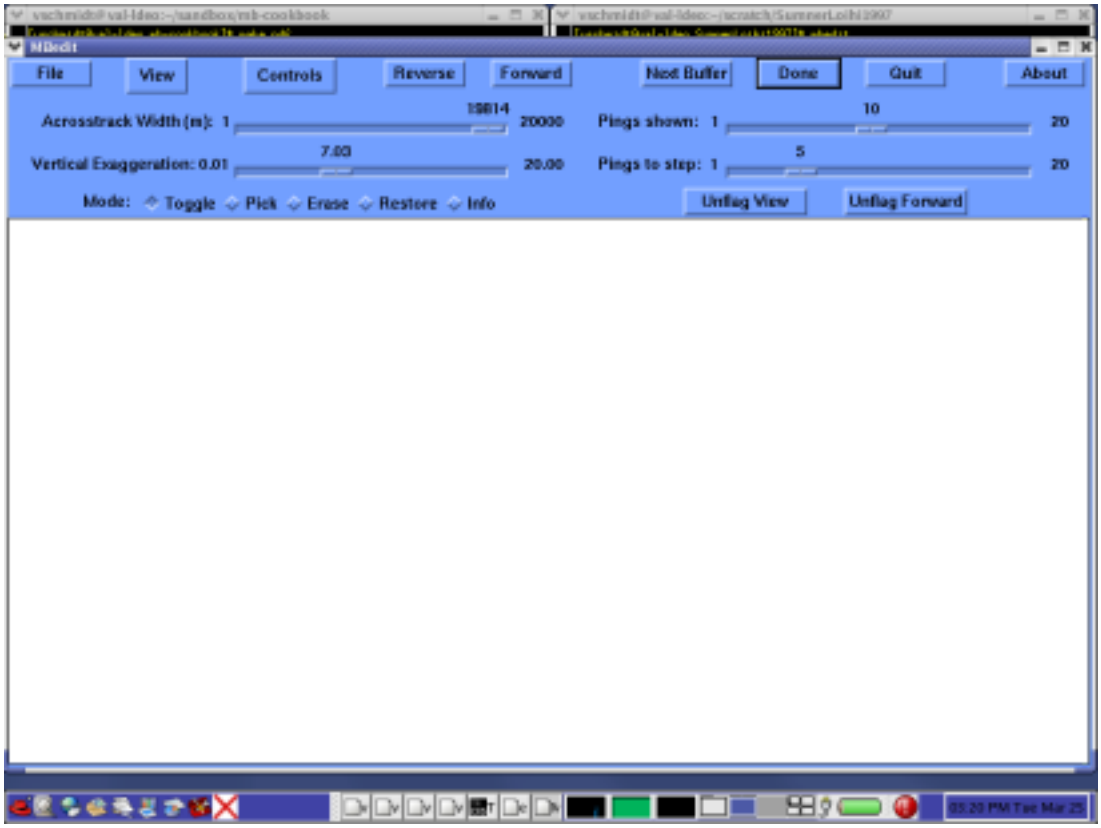
4.8.2.1. Getting Started with MBedit

Cranking up mbedit is quite straight forward:

```
mbedit
```

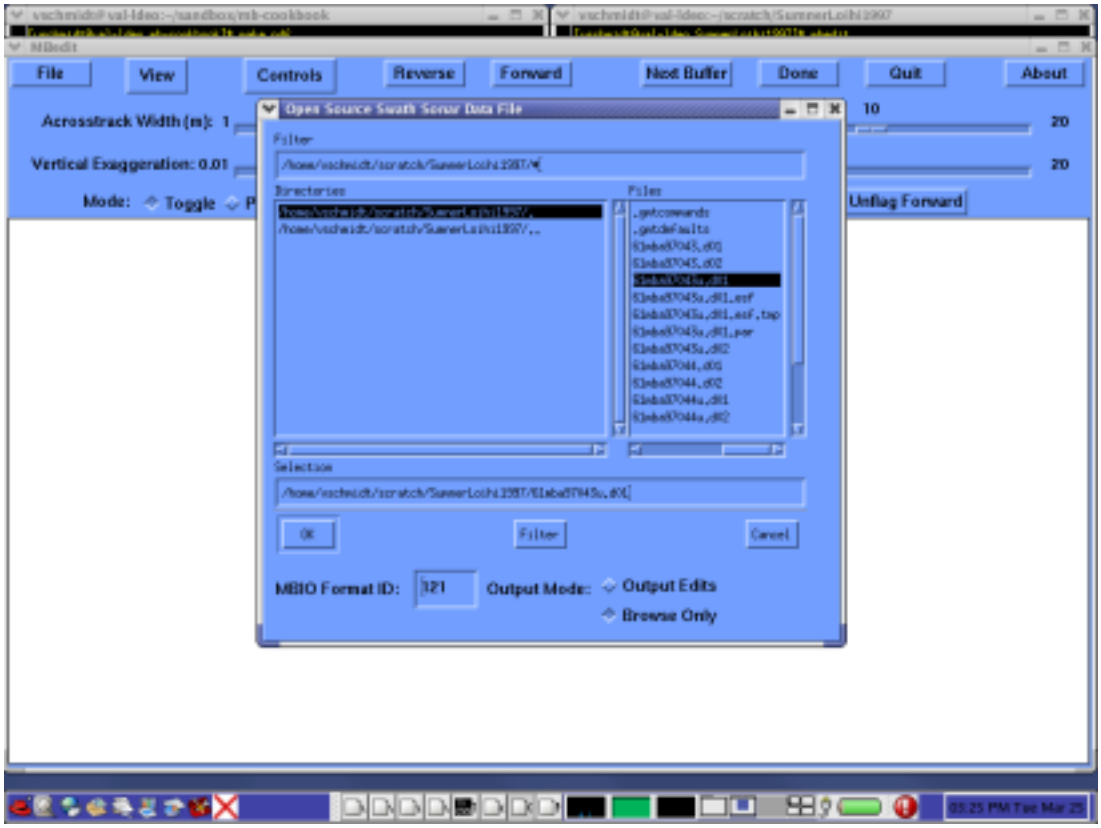
As we've seen in other screen shots, the GUI is much the same as other MB-System™ interactive applications.

Figure 4.35. MBedit GUI



Like mbvelocitytool and mbnavedit, the process for loading a data file is much the same. Select the "File" button to generate the "Open Sonar Swath File" dialog box.

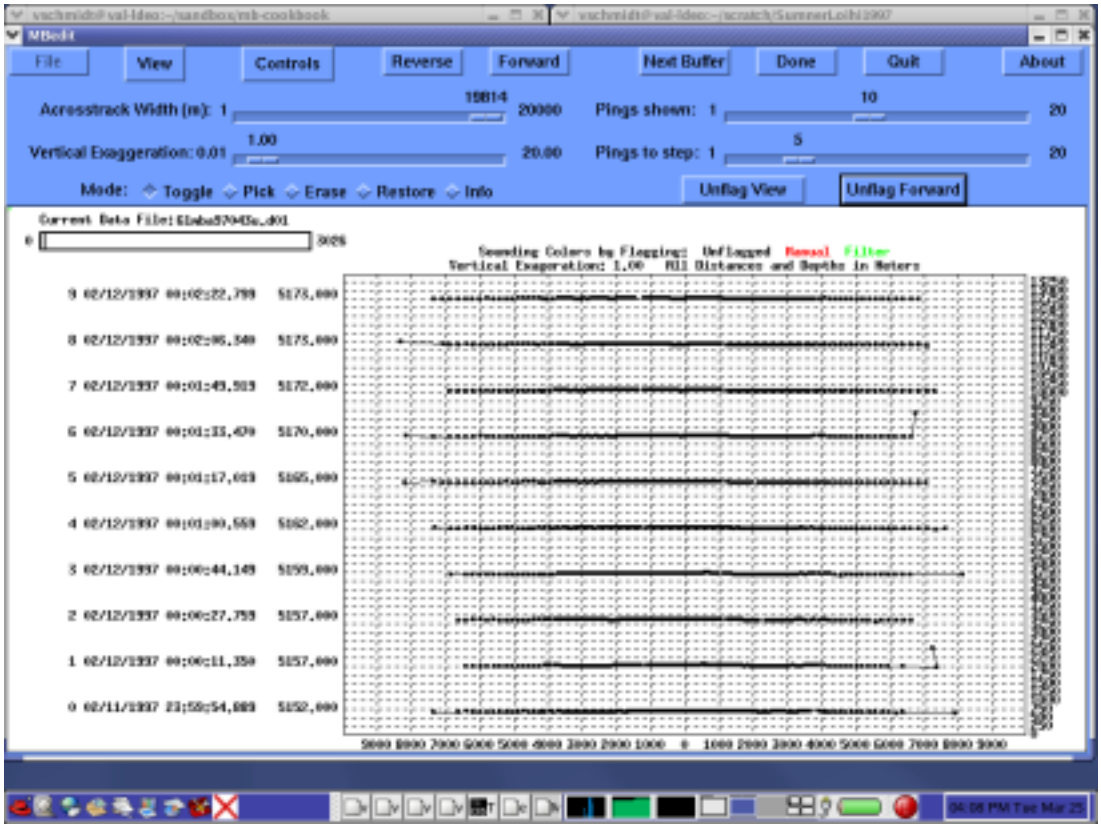
Figure 4.36. MBedit Open Sonar Swath File Dialog Box



Then simply navigate to the appropriate file to load and select it. Remember to specify the appropriate MB-System™ format ID, and whether you would like to save your editing, or simply browse the data set.

Once loaded the data set will be displayed, one ping at a time, with time stamps to the right as shown below.

Figure 4.37. MBedit With Sonar Data Loaded



4.8.2.2. The MBedit Display and Adjustments

Ok, let us look around a bit and make some adjustments to our environment before we get to editing. Plotted on the screen are the first 9 individual pings. In this case, each ping contains 120 beams, and each plot contains 120 corresponding black boxes, plotted on a horizontal scale of athwartships distance and a vertical scale of depth and along track distance. Each box along a ping is connected with thin black lines, as it is difficult to determine their order in noisy data. In this way the plot creates the effect of looking down and forward at the sea floor from the perspective of the sonar.

Across the bottom we see the across-track distance from centerline in meters. Along the left hand side are ping numbers (running from bottom to top), time stamps, and center line depths for each ping. On the right are horizontal distances in meters. At the top left corner of the plot area is a graphical representation of the viewed portion of current data file.

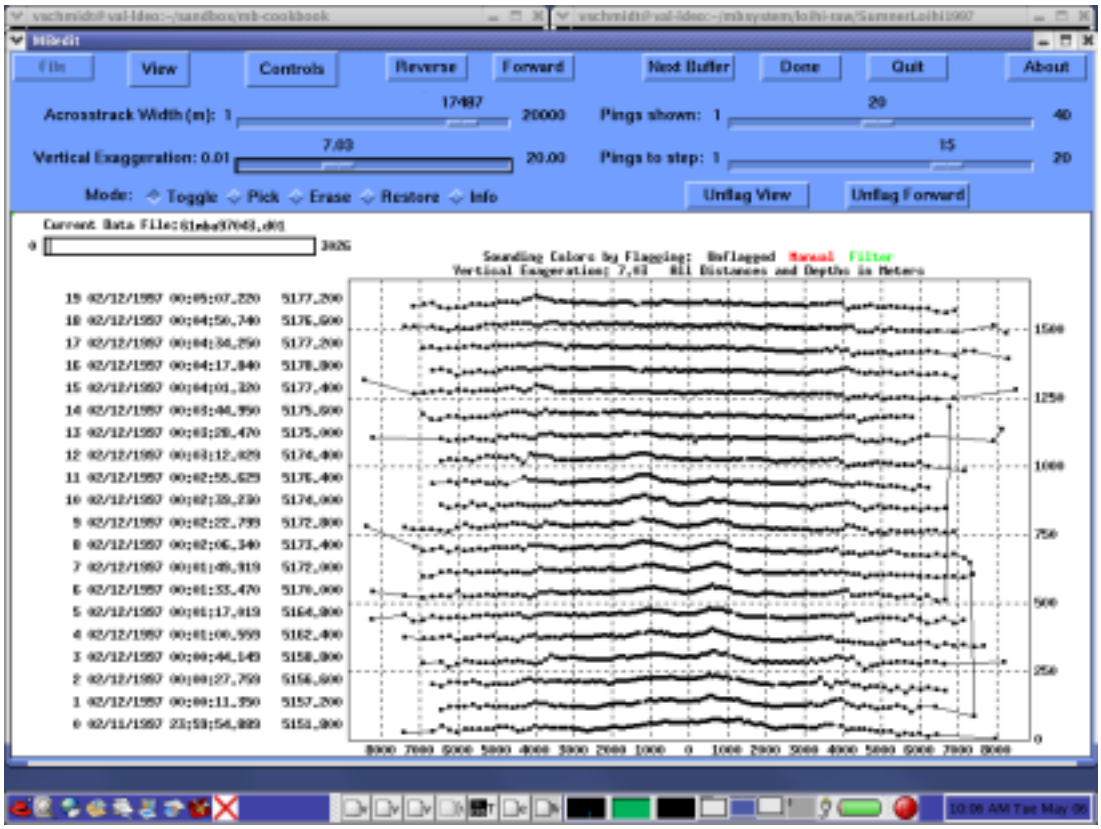
Note that the ping number, timestamp, and depth for the sixth ping in this window are highlighted in red. This indicates that at least one of the beam data points in this ping falls outside the current plot window parameters. We can see this point at the far STBD side of the ping, all the way up in the title of the plot. One may readjust the plot parameters to display all the data, or one may click the small box to the left of the highlighted text to flag the data points falling outside the plot area.

We can adjust the parameters of our plot such that we can more clearly see erroneous data and more efficiently move through the data set. Across the top we have a standard set of buttons which we'll come back to in a moment. Under them are our sliders that allow one to adjust the scaling of the plotted data. "Acrosstrack Width" allows one to adjust he horizontal zoom of the sonar plot, by selecting the maximum distance in the athwartships direction that data will be plotted. Similarly, "Pings Shown" allows one to select the number of pings that are plotted in the window. "Pings to step" simply determines the number of pings that are paged through with each click of the "Forward" or "Reverse" buttons. Finally, "Vertical Exaggeration" amplifies the vertical distances in each ping such that noisy data can be more easily

seen.

Below I have adjusted the "Pings Shown" slider to 20, the "Pings to Step" slider to 15, and the "Vertical Exaggeration" to 7.

Figure 4.38. MBedit With Display Adjustments

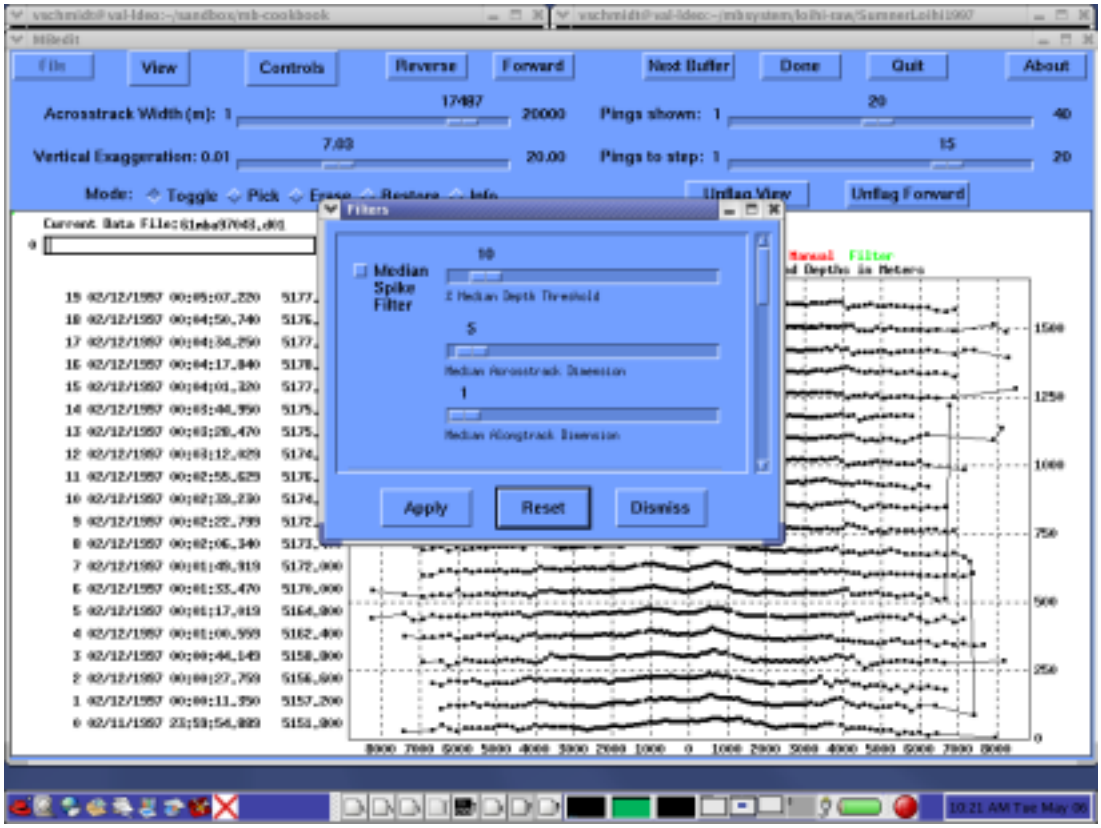


Now let us get on to editing the data.

4.8.2.3. Automated Flagging of Data in MBedit

Much like mbclean, mbedit provides several algorithms to flag erroneous bathymetry data. These can be found by selecting "Controls -> Filters..." at the top of the mbedit window. The "bathymetry filters" window is shown as below.

Figure 4.39. MBedit Bathymetry Filters

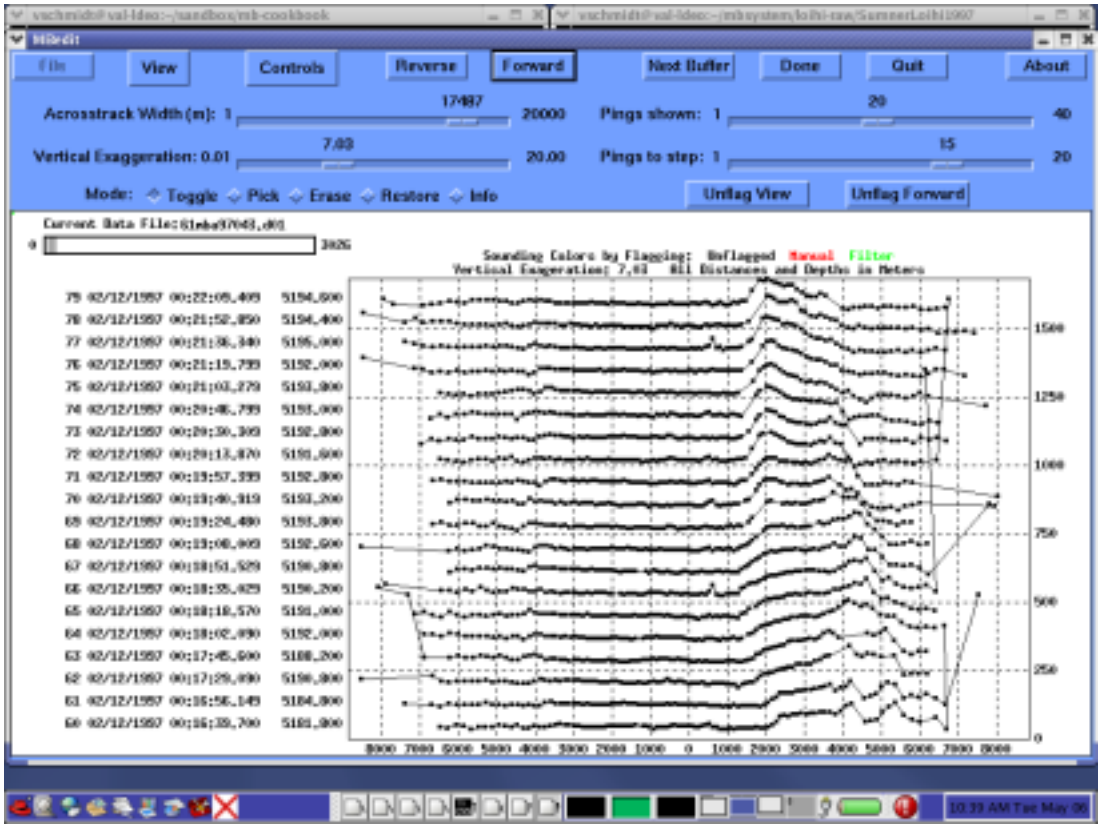


Here we can scroll through several filters that will apply some of the same algorithms provided by mbclean as well as a few new ones. In each sliders are provided that allow one to specify the parameters for the filter.

For example, the "Median Spike Filter" allows one to flag beams whose depths are outside the bounds set as some fraction of the local median depth. (the -G option to mbclean, and the "Flag by Beam Number Filter" allows one to flag a specified number of outer beams (the -X option to mbclean). Appropriate sliders are provided to specify these parameters. "Flag by Distance Filter" allows one to flag data based on their distance from the center beam, and "Flag by Beam Angle" allows one to flag data based on its angle from nadir. The "Wrong Side Filter" numbers each beam from port to starboard and then flags beams whose beam number is higher than the center beam, but sea floor position is to port of it. Beams with the converse scenario are similarly flagged. The slider provided for this filter allows one to specify how many beams surrounding the center beam are ignored by the filter.

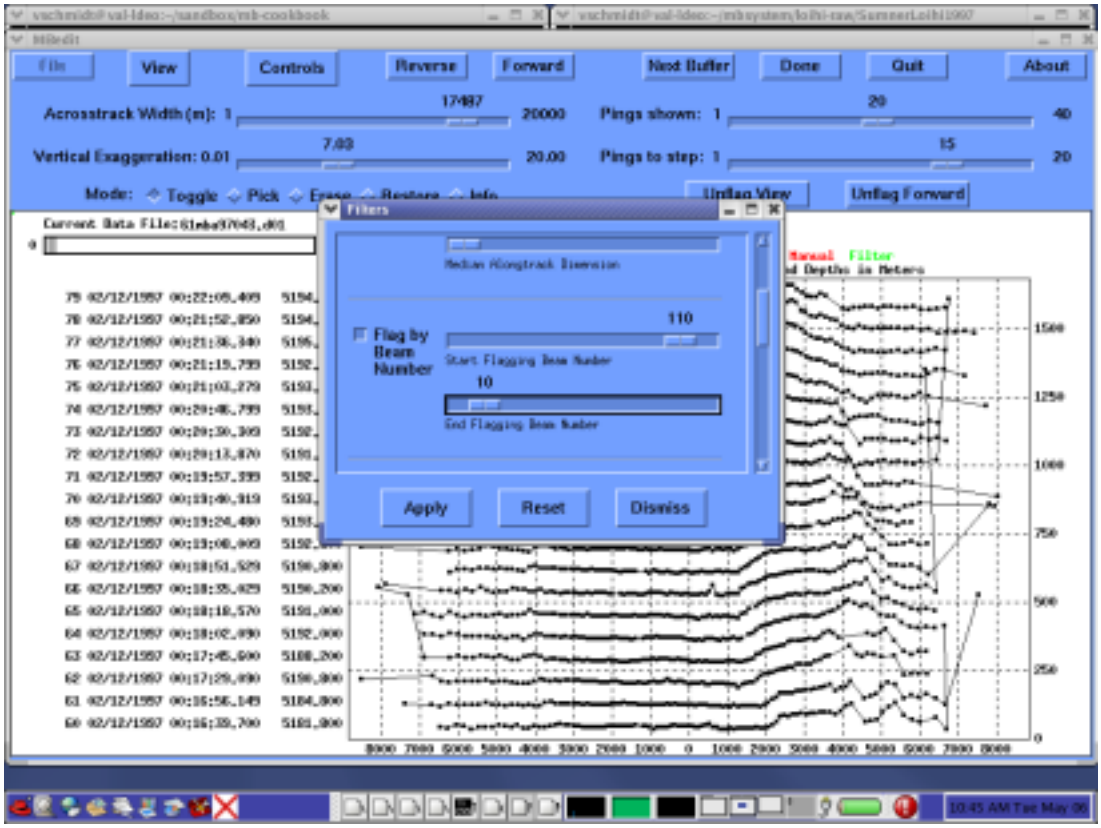
To illustrate how it works, I've forward along into the data set so we can see some new data. Here's how it looks:

Figure 4.40. MBedit Bathymetry



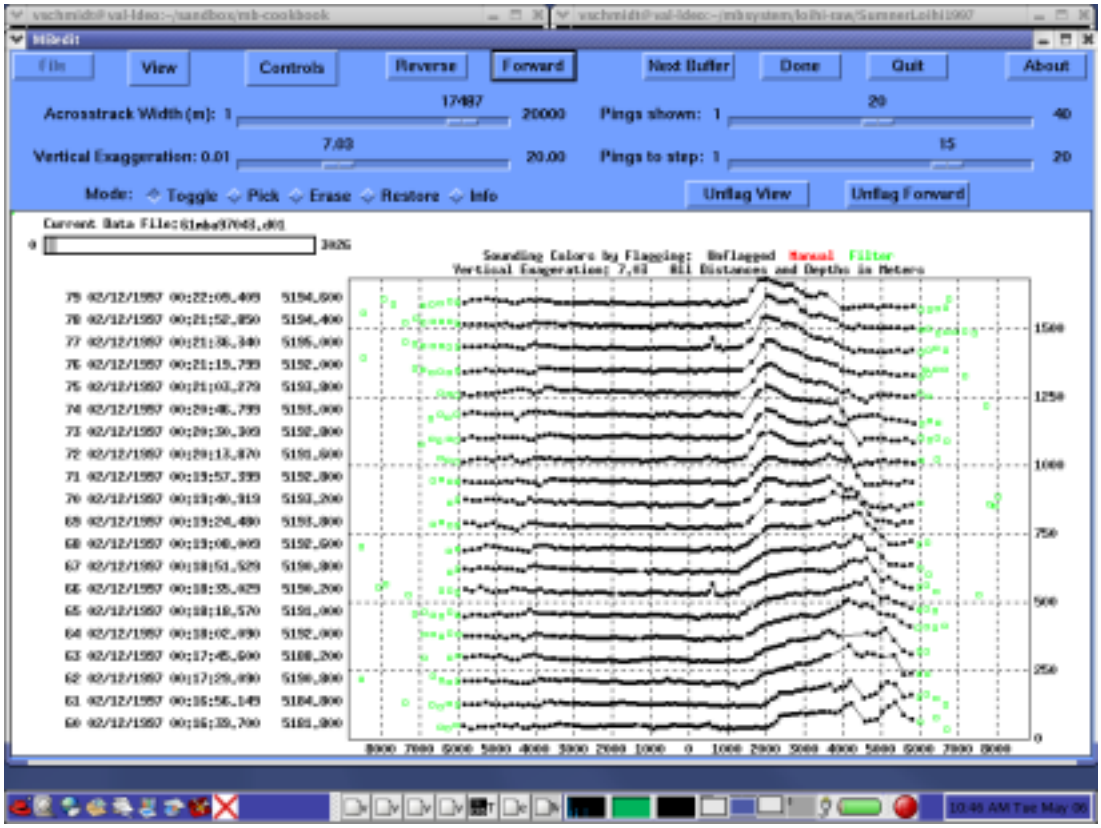
Since the nominal depth is quite deep - 5000m it doesn't make sense to use a fraction of the median depth for our filter criteria. A one percent filter would flag data as high as 50 meters, well within the resolution of most modern hull mounted sonars. So we'll apply the "Flag by Beam Number" instead. For this sonar, the beams are numbered from STBD to PORT, so we slide the sliders appropriately as shown below.

Figure 4.41. MBedit Filter by Beam Number



We then click "Apply" and "Dismiss" and the flagged beams are highlighted in green on the display.

Figure 4.42. MBedit Filter Results



In this case I've flagged 10 beams on either side. This might be too many in this particular case, and will certainly be different from sonar to sonar and environment to environment. Use your best judgment.

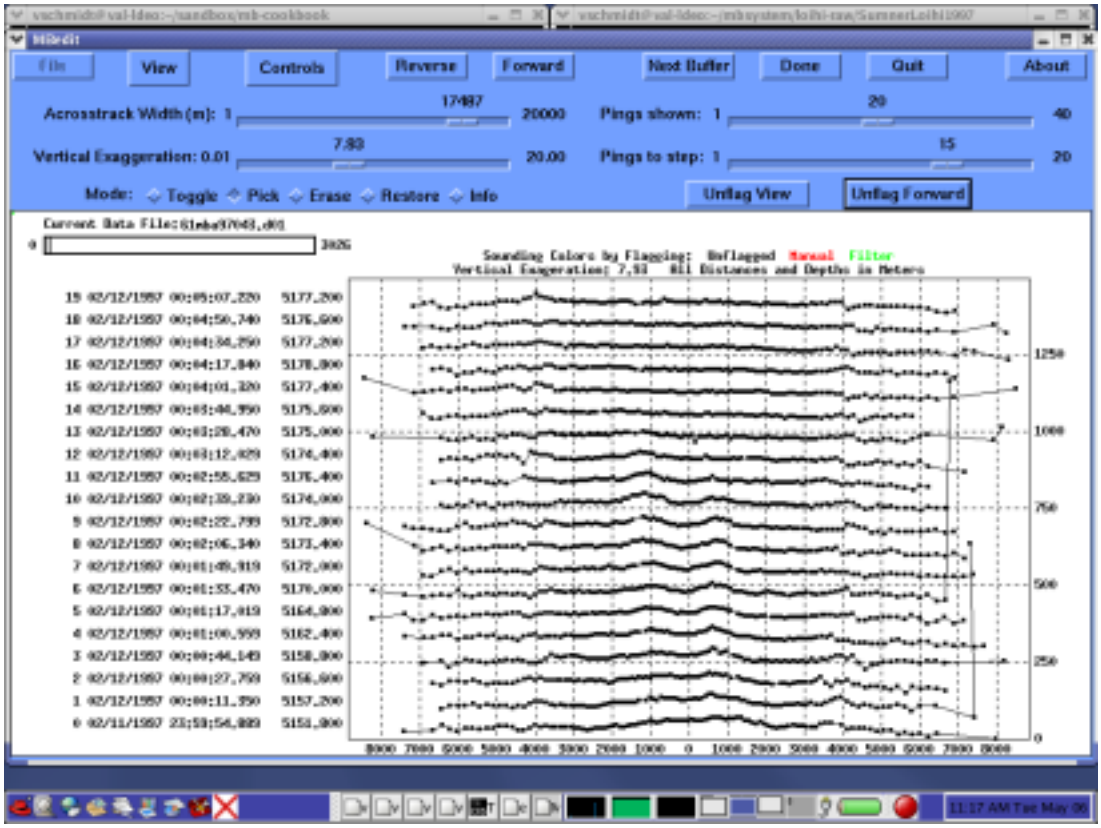
One final point to make, the filtering algorithms are applied to data in the current window and to all later data in the file. However they are not applied to data "behind" the current window. If we click "Reverse" we see that the outer 10 beams in the earlier pings have not been flagged by the filter. So if you want to filter the entire file, one must execute the filter with the first pings showing.

4.8.2.4. Point and Click Editing

There are several modes that we can place the cursor in for "point and click" editing. These are specified using the toggle buttons next to "Mode" in the mbedit window. The first is "Toggle" mode. In this mode the cursor can be placed over any data point and a click will flag the point. A second click will "unflag" the point. This mode is great for precise editing of individual beams. Similarly, "Pick" will also flag an individual beam, however, a second click will not "unflag" the beam. This allows greater control of beam flagging without accidentally undoing your work. "Erase" mode allows one to drag the cursor over data points to flag them, rather than clicking each time. While not as exacting as the earlier modes, "Erase" mode can be much less tedious. "Restore" is similar to "Erase" in that flagged beams can be "unflagged" by dragging the cursor across them. And finally, "Info" is not an editing mode at all, but displays loads of information about each selected ping.

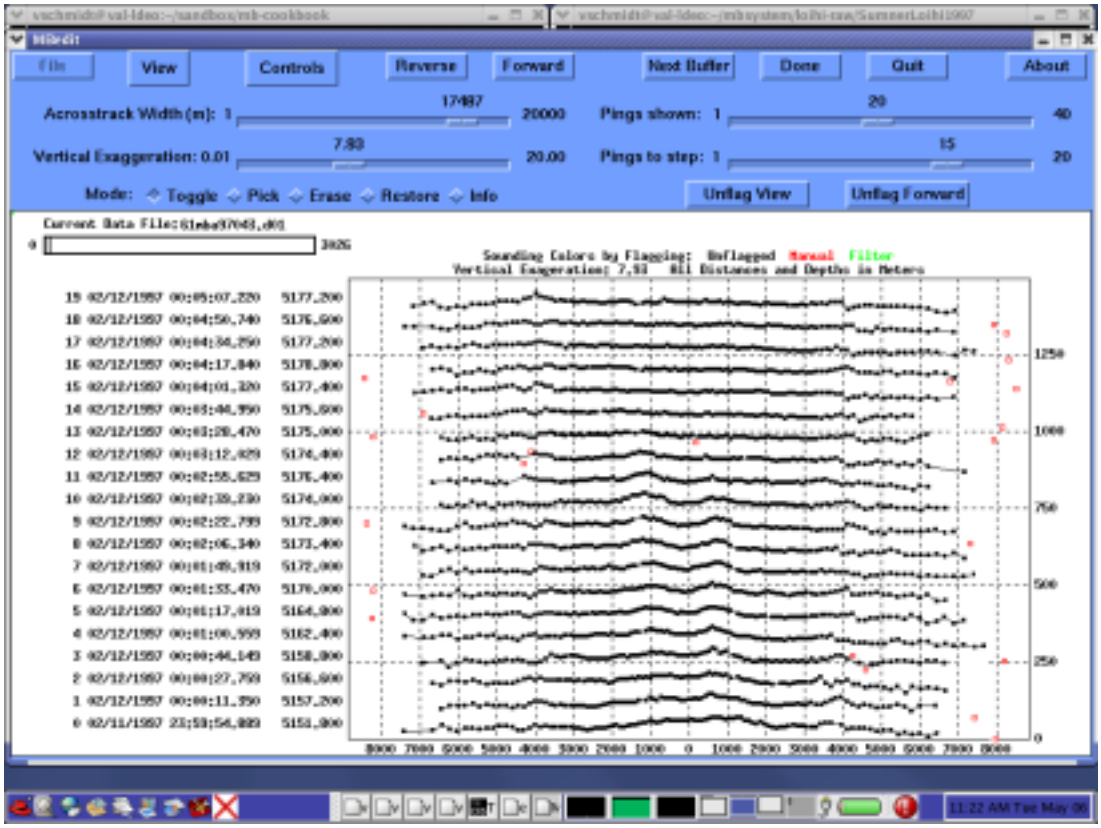
To demonstrate, I've scrolled backwards in the data set to an "unfiltered" portion. The unedited data is shown below.

Figure 4.43. Unedited Data for Demonstration



Now using the Toggle Mode, I can select several beams that need editing. In comparing the previous figure and the following one you will find that the lines connecting across track beams interpolate around flagged data points. You'll also see the flagged data points in read, indicating that they have been manually flagged.

Figure 4.44. Interactive Editing Results



In this way, one can page through each data file, inspecting the sonar data and throwing out data that appears unreasonable.

When exiting the mbedit, an ancillary ".esf" file is created that contains the edited data points. To ensure that this new ".esf" file is used in the final processing step, the ancillary processing parameter file is automatically edited to reference it. If the ".par" file does not yet exist from other preprocessing steps, it is created.

With the automatic and interactive editing of the bathymetry complete, one can turn to preparations for processing of the sidescan data.

4.9. Determine Amplitude and Grazing Angle Functions for Side Scan

Using edited data. Ran mbbackangle -F-1 -I filelist -N137/75 -P25 where 137 = the number of beams(rounded to odd number) and 75 = the angular swath width /2. Then ran process, but had to force the files to be processed as there were no changes in the "esf" files. This didn't produce noticeable results in the sidescan plot. I decided I need to filter the amplitude and sidescan values to remove the spikes. For this I need to use mbfilter, but mbfilter isn't yet capable of utilizing datalists. Therefore, I've written a script, similar to mult-copy, that will read a data list and create the mbfilter commands with output files having an appropriate naming convention.

4.10. The Final Step: Reprocessing the Bathymetry and Sidescan

After all this discussion, let us take a moment to review what we've done. We first set out to identify

what MB-System™ format identified our sonar data. With that in hand, we went on to quickly survey our data to get a good idea for the types of problems we were likely to see. We analyzed the data with the most accurate sound speed profiles available, making adjustments as necessary to produce the best profile for each portion of the data set. We then carefully analyzed selected portions of the data to determine the roll and pitch biases of the sonar. We reviewed the navigation data and may have smoothed it or even decided that a second source of navigation data must be applied altogether. Finally we began the task of flagging bad data, pitching outliers and excessively noisy data. First this was done using automated tools, and then, more laboriously, using interactive tools. We also might have applied filters and grazing angle functions to the side scan data to bring out features in the data set.

All of this has been preparatory work for the final step of applying these changes to the sonar data. This final step is completed using the application mbprocess.

Note

It is quite common to have used mbprocess within the processing of the earlier steps. For example, you may have measured and applied a SSP to the data used to calculate the roll bias. Or you may have run mbprocess to see the effects of your sidescan filters such that you could apply them iteratively to produce quality pictures. mbprocess allows changes to the parameter file at any time, and subsequent executions will incorporate those changes. In this way, mbprocess can be executed iteratively such that you can see the effects with each applied change and reprocess new changes at a later date.

As we have gone along, using various tools, they have automatically created or edited the parameter files (ending in ".par") to identify what options are to be applied to the sonar data. mbprocess uses these parameters to complete the final step of applying the sound speed profiles, biases, flagged bathymetry, smoothed navigation, and a host of other options.

In addition to the tools thus far described, there are a host of other processing options that can applied to a sonar data set via mbprocess. Many of these have thus far, not been mentioned, but are extremely useful to the processing of sonar data. These include corrects for constant errors in heading, offsets from the ship's navigation source to the sonar array (if not already taken into account), correction of the ship's draft and many others. Application of these corrections require editing of the parameter file. This man be done manually, or with mbset.

4.10.1. Mbset

mbset is a tool for creating or modifying MB-System™ parameter files. Typically the "-P" flag to mbset is simply followed by a series "PARAMETERNAME:value" separated by a comma, which are written to the parameter files of the associated data.

As an example, assume you find out that in middle of a survey, the primary GPS receiver providing position information to the sonar fails, and a backup is place online. Everything seems OK, until you realize that the new GPS antenna is mounted on the aft end of the ship, while the former was mounted on the pilot house, and no professional survey numbers are available to correct for this difference in position. Some months after the cruise, you receive an email from the ship's Science Officer that they have had the new GPS position professionally surveyed and the offsets from the previous position are 80 meters aft, and 2.5 meters to STBD. We could apply these values to the sonar data using mbset to specify these offsets in the appropriate parameter files.

```
mbset -F-1 -I DATALISTOFAFFECTFILES -PSONAROFFSETX:2.5 \
-PSONAROFFSETY:-80
```

The preceding line would write the correction values into the parameter files for those bathymetry files specified in the data list and the corrections would be applied with mbprocess.

Now that we have seen how to edit the parameter files, we can take a more in-depth look at types of things in the parameter files.

4.10.2. The Parameter File

Any post processing you would like to do to a multibeam data file is specified in its associated parameter file, and a handful of ancillary data files which accompany it. As we have seen, the parameter file will be automatically generated for you by the MB-System™ tools as it is required. So, for example, when you edit bathymetry data in mbedit, on exiting you will see that a parameter file has been generated with the same basename as the original data file, and a *.par* suffix. In this case you will also see a *.esf* file which contains a list of the bathymetry edit flags from your work in mbedit. Similarly, the application mbset allows one to set specific processing parameters within the parameter file, and will generate a parameter file if one does not already exist.

The parameter file is simply a text file containing a host of options for the post processing of the sonar data. Its general form is of commented out headings followed by parameter and value pairs. Each section designates a kind of processing and usually has a "Mode" parameter associated with it that turns the recalculation of that section "ON" or "OFF". Each parameter is quite well described in the mbprocess man page which is reproduced at the end of this section.

Caution

New processing options are being added to the parameter file all the time, often before they are fully documented in the man page or this document. It is well worth while to survey the parameter files generated by any of the automated processing tools to see what might be currently available. In the mean time, we'll do our best to keep the documentation up to date.

There is truly an extraordinary number of processing tools that may be specified in the parameter file and applied with subsequent use of mbprocess. While documented in the man pages for mbprocess and mbset, there are many fantastic, time saving details that are little known. A few in particular that are worth pointing out:

- *EXPLICIT:*

  When this flag is set (which is the default), mbprocess parses the parameter file and makes assumptions regarding what sections require processing based on the parameter values and ancillary data files specified. For example, if a roll bias is to be applied, mbprocess will automatically also turn on the ray-tracing processing mode such that the new incident angles are appropriately applied to the bathymetry.

- *Navigation Merging:*

  The collection of parameters that may be specified in this section are some of the most helpful and powerful tools in MB-System™.

  One may specify, smoothed or altogether new navigation files to be merged with the bathymetry. If you think your navigation source has been incorrectly surveyed with respect to your sonar or vessel reference, you can apply vessel-referenced fore/aft and athwartships offsets. Perhaps you find that the time standard provided for your sonar has an offset from GMT. The Navigation section allows you to set a time offset to be applied. Similarly, if you fear your heading reference has an offset, you can specify that as well, and all the necessary lever arm translations will be applied appropriately.

- *Bathymetry Editing:*

  This section allows one to apply new sound speed profiles to the data file. When the default EXPLICIT parameter is set (as described above) simply specifying a new sound speed profile in this section will turn on ray tracing such that new bathymetry values are calculated appropriately.

- *Metadata Insertion:*

  This section allows one to add comments to processed data files including the vessel, institution, cruise ID and principle investigator among others.

- *Processing Kludges:*

  This section enables us to make special provisions for particularly finicky sonar data.

It is good to remember that while the MB-System™ tools will manage and edit your parameter files for you, one can edit them by hand. The syntax is straight forward and the allowable options are specified in the man pages.

```
MBPROCESS PARAMETER FILE COMMANDS
       The mbprocess commands found in parameter files are:

        GENERAL PARAMETERS:
          EXPLICIT
               causes mbprocess to set modes implicitly
               - e.g. the SVPFILE command will also set
                 raytracing on even if the RAYTRACE command
                 is not given [explicit mode commands required]
          FORMAT constant
               sets format id [no default]
          INFILE filename
               sets input file path [no default]
          OUTFILE filename
               sets output file path [no default]

        NAVIGATION MERGING:
          NAVMODE boolean
               sets navigation merging [0]
                 0: navigation merge off
                 1: navigation merge on
          NAVFILE filename
               sets navigation file path [no default]
          NAVFORMAT constant
               sets navigation file format [9]
          NAVHEADING boolean
               sets heading to be merged from navigation file
               - note: heading merged from navigation before
                 heading correction applied
                 0: heading not changed
                 1: heading merged from navigation file
          NAVSPEED boolean
               sets speed to be merged from navigation file
                 0: speed not changed
                 1: speed merged from navigation file
          NAVDRAFT boolean
               sets draft to be merged from navigation file
               - note: draft merged from navigation before
                 draft correction applied
                 0: draft not changed
                 1: draft merged from navigation file
          NAVINTERP boolean
               sets navigation interpolation algorithm [0]
                 0: linear interpolation (recommended)
                 1: spline interpolation

        ADJUSTED NAVIGATION MERGING:
          NAVADJMODE boolean
```

```
            sets navigation merging from mbnavadjust [0]
             - longitude and latitude only
               0: adjusted navigation merge off
               1: adjusted navigation merge on
        NAVADJFILE filename
             sets adjusted navigation file path
             - this file supersedes navigation file for
               lon and lat only
             - uses mbnavadjust output
        NAVADJINTERP boolean
             sets adjusted navigation interpolation algorithm [0]
               0: linear interpolation (recommended)
               1: spline interpolation

      DATA CUTTING:
        DATACUTCLEAR
             removes all existing data cutting commands
        DATACUT kind mode min max
             adds new data cutting command, where:
               kind = 0 : cut applied to bathymetry data
               kind = 1 : cut applied to amplitude data
               kind = 2 : cut applied to sidescan data
               mode = 0 : min and max indicate start and end
                          beam/pixel numbers between which data
                          are flagged or zeroed
               mode = 1 : min and max indicate start and end
                          acrosstrack distance (m) between which
                          data are flagged or zeroed
        BATHCUTNUMBER min max
             adds new bathymetry data cutting command where
             min and max are the start and end beam numbers
             between which data are flagged (note that
             flagging bathymetry also flags amplitude data)
        BATHCUTDISTANCE min max
             adds new bathymetry data cutting command where
             min and max are the start and end acrosstrack
             distance (m) between which data are flagged
             (note that flagging bathymetry also flags
             amplitude data)
        BATHCUTSPEED min max
             adds new bathymetry data cutting command where
             all beams are flagged for pings with a ship
             or vehicle speed less than min or greater than
             max (note that flagging bathymetry also flags
             amplitude data)
        AMPCUTNUMBER min max
             adds new amplitude data cutting command where
             min and max are the start and end beam numbers
    between which amplitude data are zeroed (note
             that zeroing amplitude data has no impact on
             bathymetry data)
        AMPCUTDISTANCE min max
             adds new amplitude data cutting command where
             min and max are the start and end acrosstrack
             distance (m) between which amplitude data are
             zeroed (note that zeroing amplitude data has
             no impact on bathymetry data)
        AMPCUTSPEED min max
             adds new amplitude data cutting command where
             all amplitude values are zeroed for pings with
             a ship or vehicle speed less than min or greater
             than max (note that zeroing amplitude data has
             no impact on bathymetry data)
        SSCUTNUMBER min max
```

```
        adds new sidescan data cutting command where
        min and max are the start and end pixel numbers
        between which sidescan data are zeroed  (note
        that zeroing sidescan data has no impact on
        bathymetry data)
  SSCUTDISTANCE min max
        adds new sidescan data cutting command where
        min and max are the start and end acrosstrack
        distance (m) between which sidescan data are
        zeroed  (note that zeroing sidescan data has
        no impact on bathymetry data)
  SSCUTSPEED min max
        adds new sidescan data cutting command where
        all sidescan values are zeroed for pings with
        a ship or vehicle speed less than min or greater
        than max (note that zeroing sidescan data has
        no impact on bathymetry data)

BATHYMETRY EDITING:
  EDITSAVEMODE boolean
        turns on reading edit save file (from mbedit) [0]
  EDITSAVEFILE filename
        sets edit save file path (from mbedit) [none]

BATHYMETRY RECALCULATION:
  SVPMODE mode
        sets usage of a water sound speed model (sound
        velocity profile, or SVP) [0]
         0: bathymetry recalculation by raytracing off
         1: bathymetry recalculation by raytracing on
         2: translate depths from corrected to uncorrected
            or vice versa depending on SOUNDSPEEDREF
            command
  SVPFILE filename
        sets SVP file path [no default]
  SSVMODE boolean
        sets surface sound velocity (SSV) mode [0]
         0: use SSV from file
         1: offset SSV from file (set by SSV command)
         2: use constant SSV (set by SSV command)
  SSV constant/offset
        sets SSV value or offset (m/s) [1500.0]
  ANGLEMODE mode
        sets handling of beam angles during
        raytracing [0]
         0: angles not changed before raytracing
         1: angles adjusted using Snell's Law for
            the difference between the surface sound
            velocity (SSV) and the sound speed at
            the sonar depth in the SVP.
         2: angles adjusted using Snell's Law and
            the sonar array geometry for the
            difference between the surface sound
            velocity (SSV) and the sound speed at
            the sonar depth in the SVP.
  TTMULTIPLY multiplier
        sets value multiplied by travel times [1.0]
  SOUNDSPEEDREF boolean
        determines the handling of the sound
        speed reference for bathymetry [1]
        - note: if raytracing is turned off then
          this command implies correcting or
          uncorrecting using the SVP specified
          with the SVPFILE command
```

```
          0: produce "uncorrected" bathymetry
             referenced to a uniform 1500 m/s
             water sound speed model.
          1: produce "corrected" bathymetry
             referenced to a realistic water
             sound speed model.

STATIC BEAM BATHYMETRY OFFSETS:
  STATICMODE mode
       sets offsetting of bathymetry by
       per-beam statics [0]
         0: static correction off
         1: static correction on
  STATICFILE filename
       sets static per-beam file path [no default]
       - static files are two-column ASCII tables
         with the beam # in column 1 and
         the depth offset in m in column 2

DRAFT CORRECTION:
  DRAFTMODE mode
       sets draft correction [0]
       - note: draft merged from navigation before
         draft correction applied
         0: no draft correction
         1: draft correction by offset
         2: draft correction by multiply
         3: draft correction by offset and multiply
         4: draft set to constant
  DRAFT constant
       sets draft value (m) [0.0]
  DRAFTOFFSET offset
       sets value added to draft (m) [0.0]
  DRAFTMULTIPLY multiplier
       sets value multiplied by draft [1.0]

HEAVE CORRECTION:
  HEAVEMODE mode
       sets heave correction [0]
       - note: heave correction by offset and/or
         multiplication is added to any lever
         heave correction, and then either used in
         bathymetry recalculation or added to
         existing bathymetry
         0: no heave correction
         1: heave correction by offset
         2: heave correction by multiply
         3: heave correction by offset and multiply
  HEAVEOFFSET offset
       sets value added to heave (m)
  HEAVEMULTIPLY multiplier
       sets value multiplied by heave

LEVER CORRECTION:
  LEVERMODE mode
       sets heave correction by lever calculation [0]
       - note: lever heave correction is added to
         any heave correction by offset and/or
         multiplication, and then either used in
         bathymetry recalculation or added to
         existing bathymetry
         0: no lever calculation
         1: heave correction by lever calculation
  VRUOFFSETX constant
```

```
        sets athwartships offset of attitude sensor (m)
        - note: positive to starboard
  VRUOFFSETY constant
        sets fore-aft offset of attitude sensor (m)
        - note: positive forward
  VRUOFFSETZ constant
        sets vertical offset of attitude sensor (m)
        - note: positive down
  SONAROFFSETX constant
        sets athwartships offset of sonar receive array (m)
        - note: positive to starboard
  SONAROFFSETY constant
        sets fore-aft offset of sonar receive array (m)
        - note: positive forward
  SONAROFFSETZ constant
        sets vertical offset of sonar receive array (m)
        - note: positive down

ROLL CORRECTION:
  ROLLBIASMODE mode
        sets roll correction [0]
          0: no roll correction
          1: roll correction by single roll bias
          2: roll correction by separate port and
             starboard roll bias
  ROLLBIAS offset
        sets roll bias (degrees)
  ROLLBIASPORT offset
        sets port roll bias (degrees)
  ROLLBIASSTBD offset
        sets starboard roll bias (degrees)

PITCH CORRECTION:
  PITCHBIASMODE mode
        sets pitch correction [0]
          0: no pitch correction
          1: pitch correction by pitch bias
  PITCHBIAS offset
        sets pitch bias (degrees)

HEADING CORRECTION:
  HEADINGMODE mode
        sets heading correction [no heading correction]
        - note: heading merged from navigation before
          heading correction applied
          0: no heading correction
          1: heading correction using course
             made good
        2: heading correction by offset
          3: heading correction using course
             made good and offset
  HEADINGOFFSET offset
        sets value added to heading (degrees)

TIDE CORRECTION:
  TIDEMODE mode
        sets tide correction [0]
        - note: tide added to bathymetry after
          all other calculations and corrections
          0: tide correction off
          1: tide correction on
  TIDEFILE filename
        sets tide file path
  TIDEFORMAT constant
```

```
        sets tide file format [1]
        - tide files can be in one of four ASCII
          table formats
          1: format is >time_d tide<
          2: format is >yr mon day hour min sec tide<
          3: format is >yr jday hour min sec tide<
          4: format is >yr jday daymin sec tide<
        - time_d = decimal seconds since 1/1/1970
        - daymin = decimal minutes start of day

AMPLITUDE CORRECTION:
  AMPCORRMODE  boolean
        sets correction of amplitude for
        amplitude vs grazing angle function
          0: amplitude correction off
          1: amplitude correction on
  AMPCORRFILE filename
        sets amplitude correction file path      [no default]
  AMPCORRTYPE mode
        sets sidescan correction type [0]
          0: correction by subtraction (dB scale)
          1: correction by division (linear scale)
  AMPCORRSYMMETRY boolean
        forces correction function to be symmetric [1]
  AMPCORRANGLE constant
        sets amplitude correction reference angle      (deg) [30.0]

SIDESCAN CORRECTION:
  SSCORRMODE  boolean
        sets correction of sidescan for
        amplitude vs grazing angle function
          0: sidescan correction off
          1: sidescan correction on
  SSCORRFILE filename
        sets sidescan correction file path      [no default]
  SSCORRTYPE mode
        sets sidescan correction type [0]
          0: correction by subtraction (dB scale)
          1: correction by division (linear scale)
  SSCORRSYMMETRY boolean
        forces correction function to be symmetric [1]
  SSCORRANGLE constant
        sets sidescan correction reference angle      (deg) [30.0]

SIDESCAN RECALCULATION:
  SSRECALCMODE  boolean
        sets recalculation of sidescan for
        Simrad multibeam data
          0: sidescan recalculation off
          1: sidescan recalculation on
  SSPIXELSIZE constant
        sets recalculated sidescan pixel size (m) [0.0]
        - a zero value causes the pixel size to
          be recalculated for every data record
  SSSWATHWIDTH  constant
        sets sidescan swath width (degrees) [0.0]
        - a zero value causes the swath width
          to be recalculated for every data record
  SSINTERPOLATE  constant
        sets sidescan interpolation distance
        (number of pixels)

METADATA INSERTION:
  METAVESSEL string
```

```
        sets mbinfo metadata string for vessel
    METAINSTITUTION string
        sets mbinfo metadata string for vessel
        operator institution or company
    METAPLATFORM string
        sets mbinfo metadata string for sonar
        platform (ship or vehicle)
    METASONAR string
        sets mbinfo metadata string for sonar
        model name
    METASONARVERSION string
        sets mbinfo metadata string for sonar
        version (usually software version)
    METACRUISEID string
        sets mbinfo metadata string for institutional
        cruise id
    METACRUISENAME string
        sets mbinfo metadata string for descriptive
        cruise name
    METAPI string
        sets mbinfo metadata string for principle
        investigator
    METAPIINSTITUTION string
        sets mbinfo metadata string for principle
        investigator
    METACLIENT string
        sets mbinfo metadata string for data owner
        (usually PI institution)
    METASVCORRECTED boolean
        sets mbinfo metadata boolean for sound
        velocity corrected depths
    METATIDECORRECTED boolean
        sets mbinfo metadata boolean for tide
        corrected bathymetry
    METABATHEDITMANUAL boolean
        sets mbinfo metadata boolean for manually
        edited bathymetry
    METABATHEDITAUTO boolean
        sets mbinfo metadata boolean for automatically
        edited bathymetry
    METAROLLBIAS constant
        sets mbinfo metadata constant for roll bias
        (degrees + to starboard)
    METAPITCHBIAS constant
        sets mbinfo metadata constant for pitch bias
        (degrees + forward)
    METAHEADINGBIAS constant
        sets mbinfo metadata constant for heading bias
    METADRAFT constant
        sets mbinfo metadata constant for vessel draft (m)

PROCESSING KLUDGES:
    KLUGE001 boolean
        enables correction of travel times in Hydrosweep DS2
        data from the R/V Maurice Ewing in 2001 and 2002.
    KLUGE002 boolean
        processing kludge 002 (not yet defined)
        - occasional odd processing problems will
          occur that are specific to a particular
          survey or sonar version
        - mbprocess will allow one-time fixes to
          be defined as "kludges" that can be turned
          on through the parameter files.
    KLUGE003 boolean
```

```
     processing kludge 003 (not yet defined)
KLUGE004 boolean
     processing kludge 004 (not yet defined)
KLUGE005 boolean
     processing kludge 005 (not yet defined)
```

Organization is key to speeding the reprocessing of sonar data. With your data in an archive as described in earlier chapters, one can utilize the data lists and mbset to apply changes to the parameter files in bulk. However it is instructive to make a metadata map of the files involved to make sure you do not miss any detail along the way.

"What do you mean by "metadata map"?, you say. Make a list of all the data files, and identify for each, all the appropriate corrections that you have identified along the way. For example, identify which SSP's are to be applied to which portions of the survey, what roll bias values are to be applied (typically to all of the data), and what navigation corrections should be applied and where. You may have organized your data into subdirectories already, and you may decide to reorganize your data at this point based on the corrections you need to apply.

With the map in hand, you can then ensure that the parameter files for each data file are correct and make appropriate changes by hand or in bulk using data lists and mbset.

4.10.3. Mbprocess

To apply the settings in the parameter file and the other ancillary files (navigation, flagged bathymetry, etc.) we call mbprocess.

Calling mbprocess is quite simple, the following is usually sufficient.

```
mbprocess -Idatalist -F-1
```

The line above will apply the changes specified in the respective parameter files to the data files listed in the "datalist". Unless specified explicitly with the "-O" flag, mbprocess automatically creates names for the processed data files by appending a "p" to the basename of each file. Hence 00020504133000.mb183 becomes 00020504133000p.mb183. mbprocess keeps tabs, in the parameter file, of the name and modification time of the processed data file, the parameter file, and the ancillary data files and only reprocessed data if changes have been made. One can force reprocessing of the data, however, with the "-P" flag to mbprocess. Typically mbprocess will include comments from the original data file as well as embed the meta-data items specified in the parameter file, into the processed sonar data. The "-N" flag prevents this.

It is helpful to remember that changes can be made piece-meal, in any order, and indeed iteratively, to the parameter file. mbprocess can be called after each change to see the effects, or only after all the changes have been made.

That's it. The rest is up to you. If you have taken good notes during the survey, and are methodical in your methods for post processing the data, it can be relatively straight forward. If you are post processing data that is several years old, is poorly documented, and is from a cruise in which you did not participate, it can be quite a challenge. In either case, MB-System™ will be an invaluable aid.

Best of Luck!

Chapter 5. Advanced Sonar Data Statistics

Extracting and plotting sonar statistics is one one the most helpful tools available to both a sonar engineers and scientists. So much so, that we've decided to tuck an entire chapter dedicated to the topic here in the back. Whether you are trying to discern the amount of noise inherent in your sonar, or how good the coverage is in your survey, MB-System™ has many tools to help.

5.1. Advanced Statistics with mbinfo

At the end of the "Surveying your Survey" chapter we saw that mbinfo quickly produces a standard set of statistics about a data set. mbinfo can also provide statistics on a "per beam" rather than a "per file" basis. These statistics include the mean, variance and standard deviation of the bathymetry, sidescan and amplitude data in a given beam over the length of the data file. Actually what is provided by mbinfo is the ability to filter the data set somewhat by specifying a number of pings to average. The resulting data are then used to calculate the mean variance and standard deviation.

Let me try to explain with an example. Suppose you have a data set with just 5 pings of data. One invokes the mean, variance and standard deviation calculations with the *-Ppings* flag of mbinfo where *pings* specifies the averaging or filtering interval. If we invoke `mbinfo -F183 -I datafile -P3`, for the first beam, averages will be calculated for pings 1,2 and 3; then averages for pings 2,3 and 4; and finally averages for pings 3,4 and 5. For the first beam, instead of the 5 original data points, we now have 3 data points which are the results of our filtering. The mean, variance, and standard deviation of these 3 points are then calculated and reported by mbinfo.

Below mbinfo is executed on a piece of data used elsewhere in the cookbook for roll bias calculations with these statistics calculations turned on. Below the normal mbinfo data are the "Beam Bathymetry Variances". A subset of the results are shown.

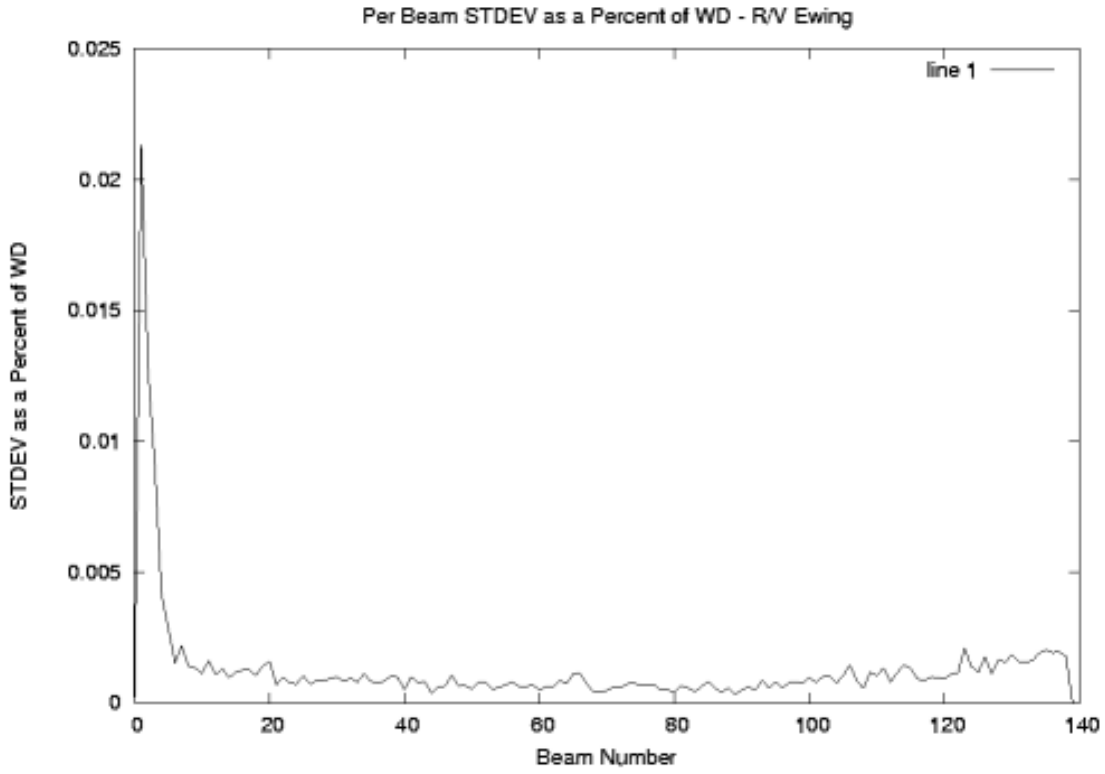
```
mbinfo -F183 -I other_data_sets/rollbias/compositefirsttrack.mb183 -P3
...
Beam Bathymetry Variances:
Pings Averaged: 3
 Beam     N      Mean     Variance    Sigma
 ----     -      ----     --------    -----
   0      0       0.00        0.00     0.00
   1     18    5495.23    14800.32   121.66
   2     36    5566.27     5284.35    72.69
   3     36    5616.40     2704.20    52.00
   4     36    5630.93      552.92    23.51
   5     36    5672.55      268.41    16.38
   6     36    5673.54       75.28     8.68
   7     36    5674.83      155.70    12.48
   8     36    5676.09       62.31     7.89
   9     36    5682.15       58.47     7.65
 ...
```

Calculating these statistics over a flat piece of sea floor provides a quick way to assess the noisiness of a sonar array. Noise is heavily depth dependent, therefore, in order to compare this data with other data sets taken in different depths of water, we can divide our standard deviation values by the average water depth to get something akin to percent error. Over a flat bottom, one would expect the standard deviation of the data in each beam to be quite small.

Not shown here, we have manually extracted the bathymetric standard deviation data output by mbinfo

above. We then loaded into a favorite math program, divided the standard deviation data by a the nominal water depth for the file, and plotted the results.

Figure 5.1. Flat Bottom - STDEV vs Beam Number



Here we see that the sonar has a noise level that creates a standard deviation of depth errors of far less than 2% of the water depth. We also see some problems in the outer beams of the port side - perhaps something to investigate. Noise of this kind can be caused by several factors, including own-ship noise, sea state, and electronic interference of various kinds.

5.2. Advanced Statistics with mblist

Often we would like to look at other details about the data set that mbinfo does not provide. MB-System™ provides two tools to both extract and display these statistics, mblist and mbm_xyplot. We should consider a few examples.

While mbinfo provides nice summary statistics, it is frequently useful to look at the nadir beam depth for a data set. Often sonars use the initial bottom detection as a hint for where the sonar might find the bottom in subsequent beams moving out to either side of the ship. Therefore, the quality of nadir beam bottom detects is often of concern. Moreover, linear profiles of the sea floor often show morphological features that are not readily apparent in a 3D plot. Among many other details, mblist can extract the nadir beam depth and mbm_xyplot can create a quick plot for us. Here's how:

For this example, we can use the survey taken aboard the R/V Ewing that was referenced in the "Surveying Your Survey" chapter. The data can be found in mbexample/cookbookexamples/otherdatasets/ew0204survey/

To extract the along track distance in kilometers, and the nadir beam depth in meters, we can execute:

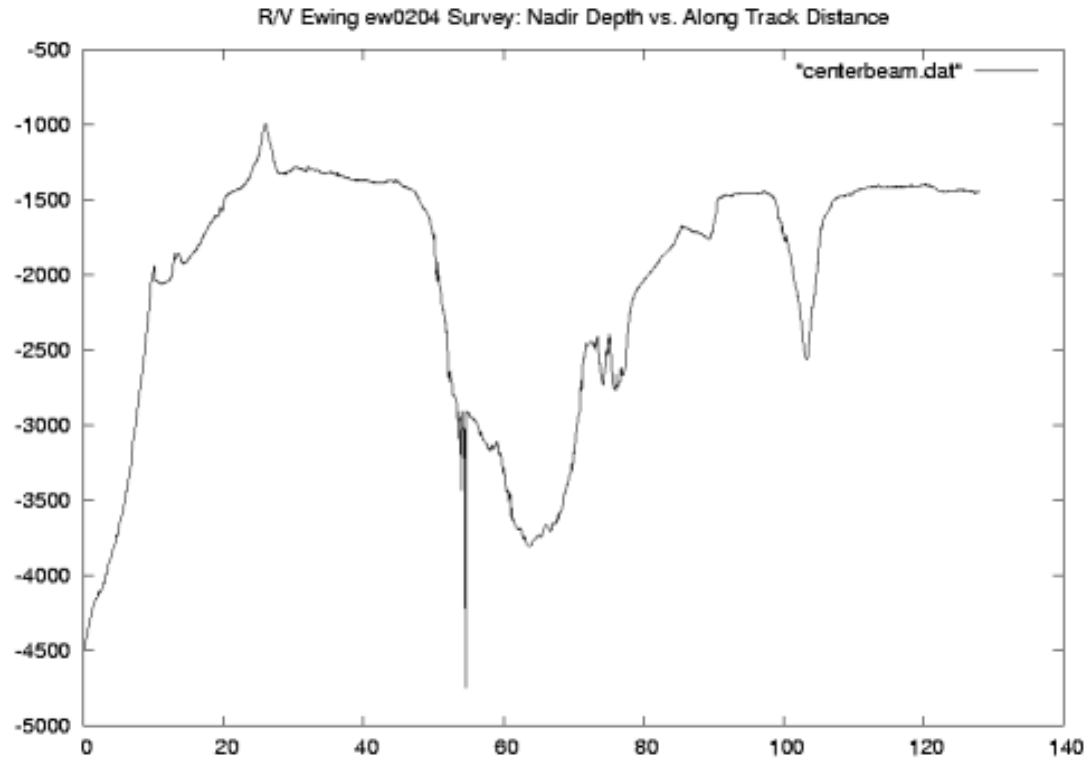
```
mblist -F-1 -I survey-datalist -OLZ > nadir.dat
```

Similar to other conventions in MB-System™ the "-1" argument to the format flag (-F) signifies a data list rather than a single file. That datalist is provided - "survey-datalist". The output format is specified by the "-O" flag and any of combination of a series of letters signifying what data to extract and list. In this case, L signifies along track distance in kilometers, and Z signifies "topography" (measured with negative numbers from sea level) in meters. Here are the results:

```
more nadir.dat
  0.000 -4508.881
  0.000 -4498.527
  0.079 -4473.954
  0.167 -4471.477
  0.253 -4449.490
  0.335 -4422.259
  0.426 -4411.923
...
```

The plot has been created with gnuplot:

Figure 5.2. Nadir Beam Depth Plot



In the plot above we have a nicely scaled profile of the depth directly beneath the ship for the duration of the survey. We see a data-discontinuity near the center - a detail that would be remedied if the survey

data has been cleaned up at all prior.

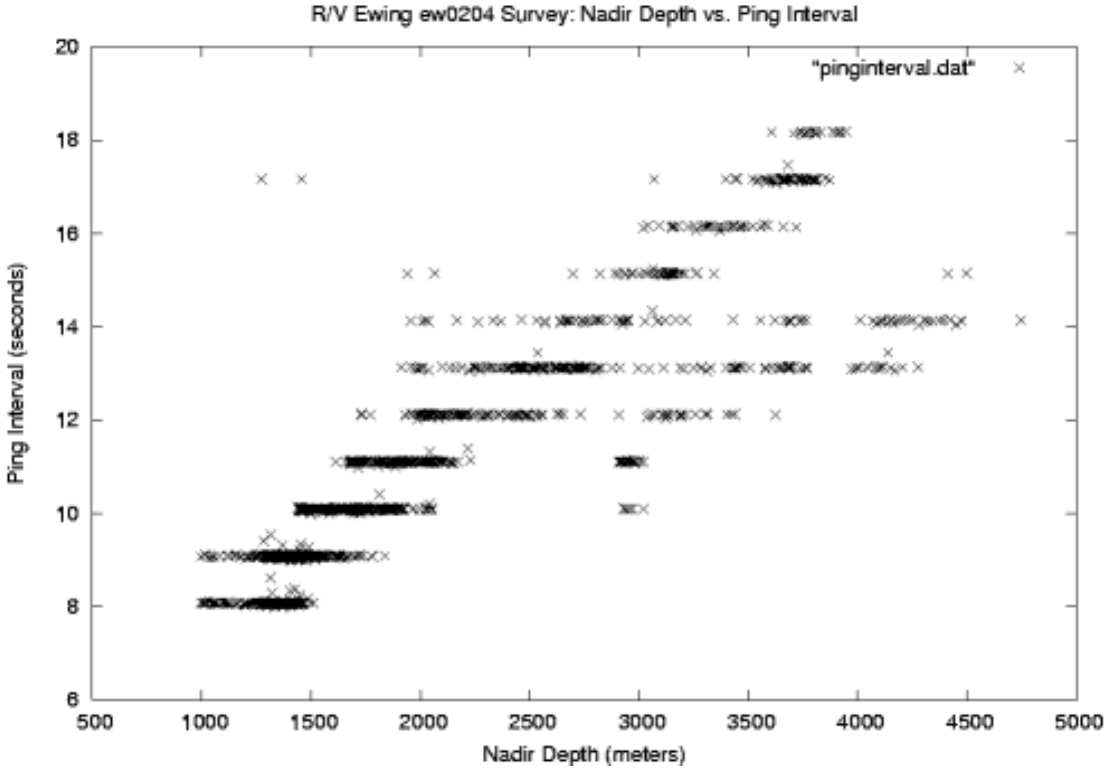
mblist is not limited to just nadir depth. Oh no, mblist can extract many others, among them speed heading, pitch, roll, draft, ping interval, course made good - all for each ping. It can also extract Latitude and Longitude, bathymetry, amplitude, across track distance and grazing angle per beam. It can calculate across track slopes per ping, and can extract and convert time into three formats.

Another interesting example is to create a plot of ping interval vs. nadir depth a sample of sonar data that varies considerably with depth. Well behaving sonars should show a clear relationship between ping interval and bottom depth. When the sonar is operating correctly, it will gate around the bottom in a orderly (typically linear) way. We can generate the data for such a plot with a line like the following:

```
mblist -F-1 -I survey-datalist -OzV > pinginterval.dat
```

Then plotting the results:

Figure 5.3. Ping Interval vs. Bottom Depth



In the plot above, one can quite clearly make out the linear relationship between ping interval and bottom depth. One can also see that ping intervals are adjusted in whole second intervals for this sonar. The plot also seems to show that the automatic gating mechanism adjusts the ping intervals at roughly 250 meter intervals. Considering the amount of data and the 3000 meter (plus) depth changes in this survey, the scatter we see here is not unusual. None-the-less, the data points outside the linear band are a product of the sonar bottom tracking mechanism loosing the bottom for a time. These can be of concern if they increase in number and persist over flat sea floors where one would otherwise expect very regular behavior.

5.3. Advanced Statistics with mbgrid

When working with multiple data sets, overlapping swath files, and new survey data, repeatability is a helpful measure of the performance of the sonar and the quality of the data set. When preparing data for publication, swath data is often first processed by a gridding algorithm to create a regular distribution of data points with gaps filled by interpolation and errors removed. It is therefore helpful to know something about the density and regularity of a gridded data set. mbm_grid can help us to do this.

For this example we can use the R/V Ewing survey data found in mbexamples/cookbookexamples/otherdatasets/ew0204survey/.

To start, we create a grid with mbm_grid much like has been demonstrated elsewhere in the text, however this time we can specify the "-M" option. This option causes mbgrid to output two additional scripts which will, in turn, produce two additional grids. The first grid will display the number of data points located within each bin, and the second, the standard deviation of those data points. Lets try it out.

```
mbm_grid -F-1 -I survey-datalist -M

Grid generation shellscript <survey-datalist_mbgrid.cmd> created.

Instructions:
  Execute <survey-datalist_mbgrid.cmd> to generate grid file <survey-datalist.grd>
```

Here mbm_grid has done all the difficult work in figuring out appropriate bounds and grid spacing and then creating script for us to execute. So now we can simply execute it:

```
./survey-datalist_mbgrid.cmd

Running mbgrid...
```

A load of information will go by here before you get the chance to look at it closely. First you'll see details regarding the input files, the type of gridding that's been chosen, the grid bounds and dimensions, and the input data bounds -- all details that mbm_grid has taken care of for you automatically. Then you'll see processing of the swath bathymetry data files, and some information regarding the number of bins and their statistics. After that, you'll see details regarding a secondary script that has been generated to create the bathymetry grid. This first section will end with something like this:

```
Plot generation shellscript <survey-datalist.grd.cmd> created.

Instructions:
  Execute <survey-datalist.grd.cmd> to generate Postscript plot <survey-datalist.g
  Executing <survey-datalist.grd.cmd> also invokes ghostview to view the plot on t
```

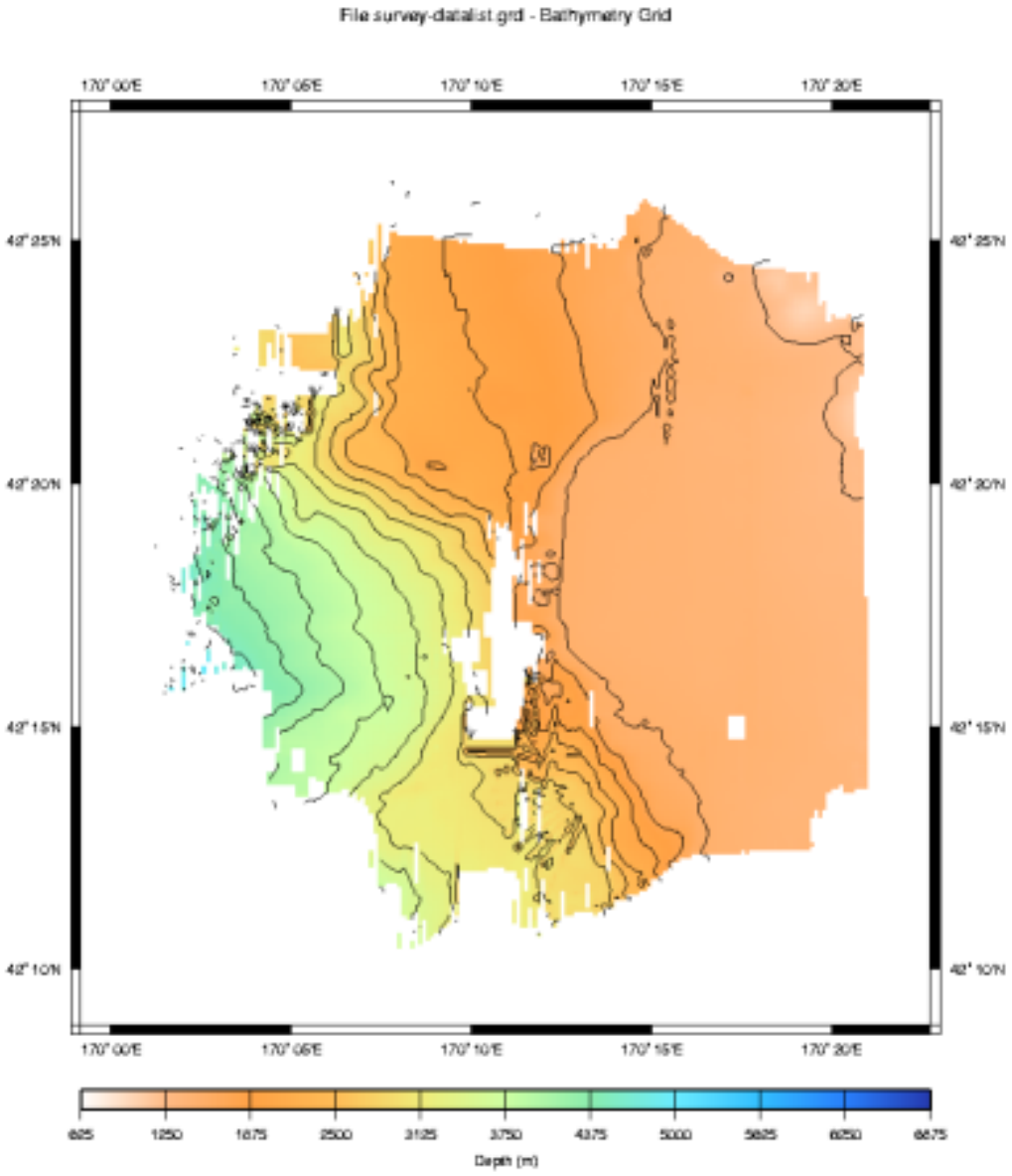
But wait, there is more! You'll see two more sets of plot details and instructions to run two more scripts that will generate the "number of data points per bin" and "standard deviation" plots. These will all likely speed pass you, so the "Instructions" lines are reproduced here:

```
...
Plot generation shellscript <survey-datalist_num.grd.cmd> created.

Instructions:
  Execute <survey-datalist_num.grd.cmd> to generate Postscript plot <survey-datali
  Executing <survey-datalist_num.grd.cmd> also invokes ghostview to view the plot
```

```
...

Plot generation shellscript <survey-datalist_sd.grd.cmd> created.

Instructions:
  Execute <survey-datalist_sd.grd.cmd> to generate Postscript plot <survey-datalis
  Executing <survey-datalist_sd.grd.cmd> also invokes ghostview to view the plot o
```

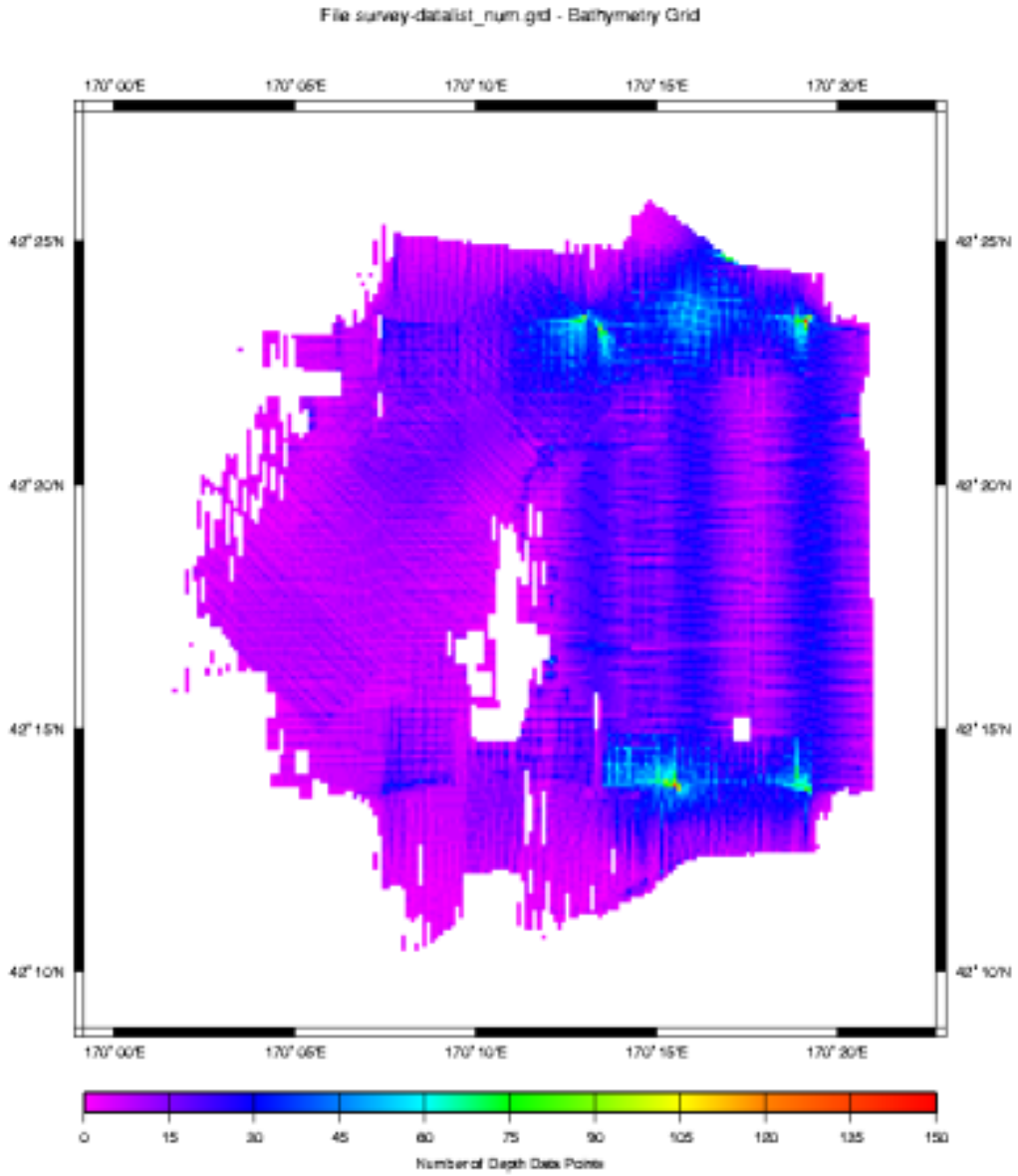
Now we can execute the three new scripts. For the sake of brevity, and because we've seen this before, we will just show the results. First the gridded data set:

Figure 5.4. R/V Ewing Survey: Gridded Data



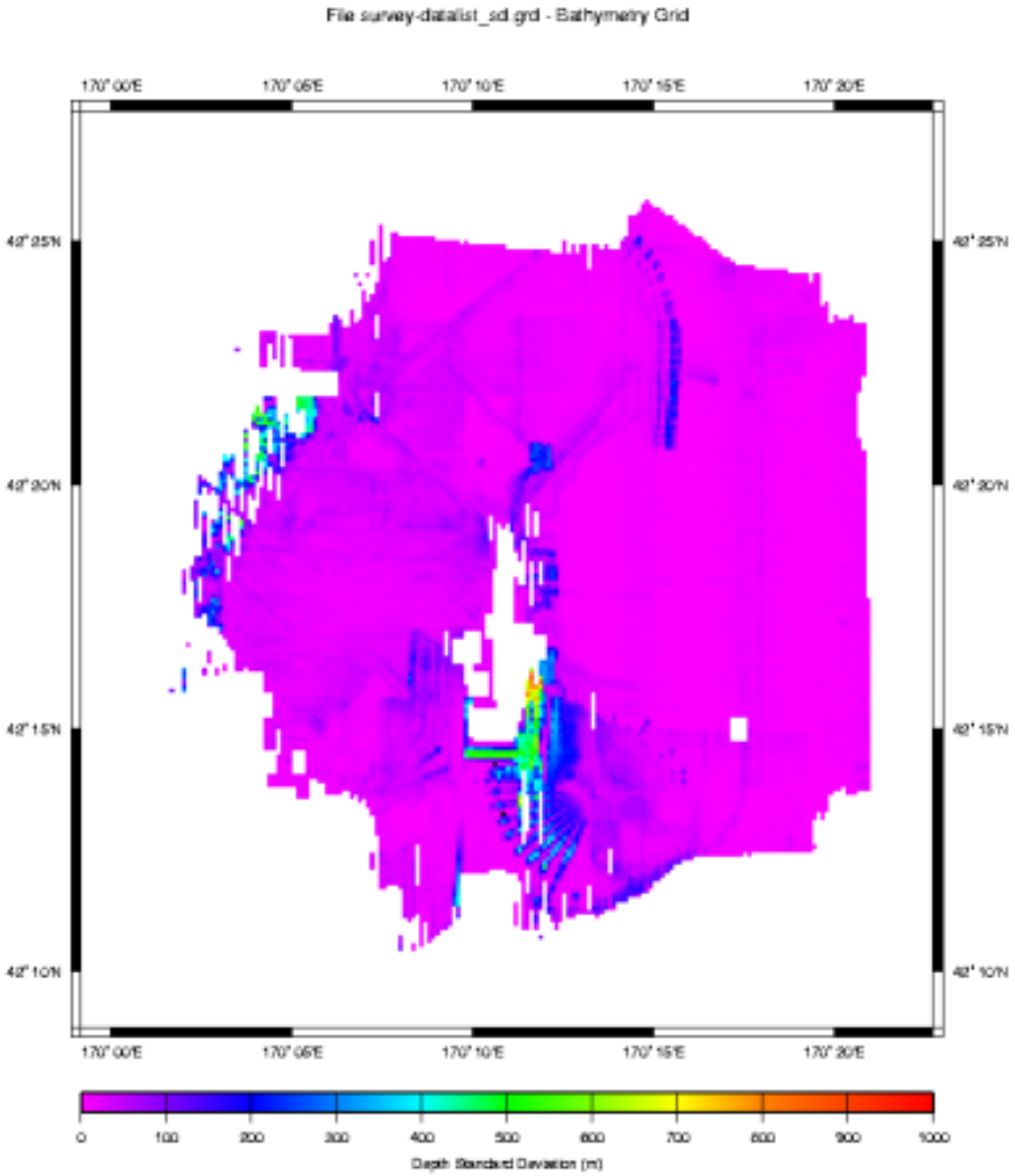
The default selections made by mbm_grid have produced a nice grid as we would expect. Now let us look at the data density plot.

Figure 5.5. R/V Ewing Survey: Data Density Plot



Here we see a density plot representing the number of data points in each bin. The highest data density is invariably directly beneath the ship, in shallow water and where turns cause subsequent pings to overlap. Conversely, the lowest data density occurs in the deepest water where the beam angles spread the data points so far apart that even an overlapping ship track does not greatly increase the data density in any given bin. Let us turn now to the standard deviation of the data in each bin.

Figure 5.6. R/V Ewing Survey: Gridded Standard Deviation



This is a fascinating plot, and there are loads of things to point out.

First, we must consider that this data has not at all been post processed. So much of the large variability is due to outliers that would normally be eliminated. But keeping them in for this example is instructive, because it makes obvious where sonars have a difficult time. In our plot we have a large variability where the sonar is attempting to look over the edge and down a large escarpment. The real bottom has likely been shadowed by the higher edge, and that combined with the fact that most of the sound energy is deflected away from the ship at such steep angles,\ results in poor depth estimations by the sonar sys-

tem. In this case some of the data points differ by as much as 1000 meters. The left side of the plot shows a similar effect, this time in deeper water, but also where the sonar is looking down a large slope. Here the data points appear to differ by as much as 500 meters.

Also notice the blue streak down the center of the plot. You'll note that it abruptly ends at about 42:21N Latitude. This kind of abrupt change results from watchstander adjustments of the sonar's operating parameters. In this case, the ship had just passed over a 3000 meter sea cliff. It is likely that some combination of the bottom detect window and source level were adjusted to produce better quality data in the shallower water. If you are lucky, when you see this you will have a watchstander's log book that will tell you exactly what changes were made.

It is also apparent from this plot that there is less variability in the sonar data for beams that are more directly beneath the keel. We know to expect this from the theory chapter, as errors in the sound speed at the keel and the sound speed profile will be exacerbated in the outer beams.

The plot also demonstrates that the generally produces poor data during turns. This is particularly evident in the fan shaped figures near the bottom center of the plot.

Since we did not edit this data before gridding it and conducting this analysis, it would be interesting to look more closely at the variability of the data points closer to zero. This would give us a view of the variability in the data had we done some post processing and removed the outliers that result in such large variability in this plot.

To do this we need to create a new color table to cover a smaller range of values. We can then edit the script that created this plot to use the new table and create a new plot. In this case, we'd like a new color table to cover the range of points from 0 to 100 meters in 5 meter increments. We can create the new color table with a GMT command makecpt. Here's how it works:

```
makecpt -Chaxby -T0/200/10 > survey-datalist_sd.grd.cpt
```

Here we've specified the "Haxby" master color pallette. We've then specified bounds (0-200) and an increment (10) which will be used to map the master color pallette to our new color table. We've redirected the output to a file of the same name that the `survey-datalist_sd.grd.cmd` script uses for its color table.

The resulting color table file contains:

```
#       cpt file created by: makecpt -Chaxby -T0/100/5
#COLOR_MODEL = RGB
#
0       10      0       121     5       10      0       121
5       40      0       150     10      40      0       150
10      0       10      200     15      0       10      200
15      0       25      212     20      0       25      212
20      26      102     240     25      26      102     240
25      25      175     255     30      25      175     255
30      50      190     255     35      50      190     255
35      97      225     240     40      97      225     240
40      106     235     225     45      106     235     225
45      138     236     174     50      138     236     174
50      205     255     162     55      205     255     162
55      223     245     141     60      223     245     141
60      247     215     104     65      247     215     104
65      255     189     87      70      255     189     87
70      244     117     75      75      244     117     75
75      255     90      90      80      255     90      90
80      255     124     124     85      255     124     124
85      245     179     174     90      245     179     174
```

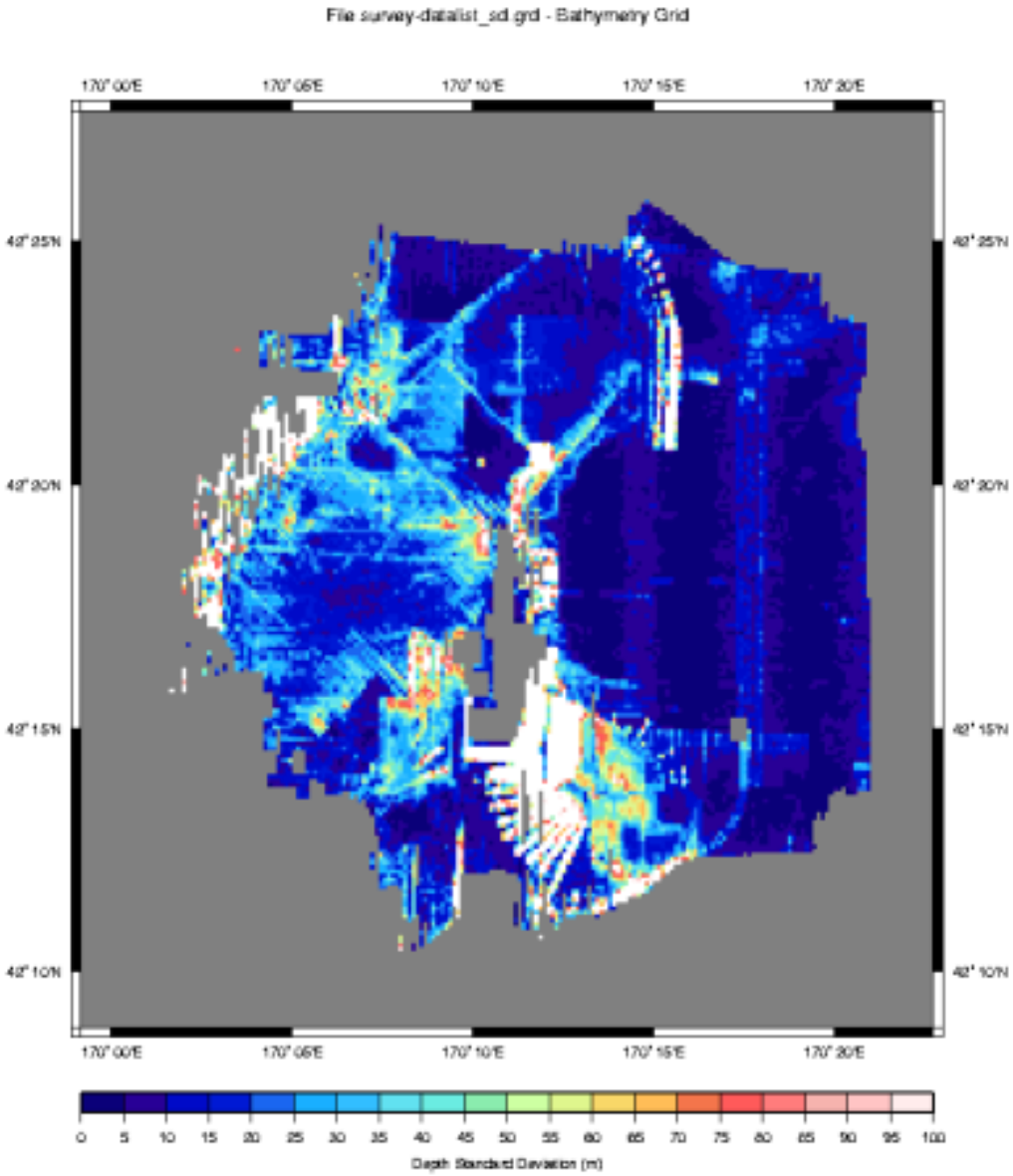
```
90      255     196     196     95      255     196     196
95      255     235     235     100     255     235     235
B       0       0       0
F       255     255     255
N       128     128     128
```

Our automatically generated plot script creates its own color table, and then deletes the file when it is done. So we need to comment out the lines that create the color table, and if we think we'll use our new color table again, we might want to save a copy so it does not inadvertently get deleted. Here are the lines to comment out:

```
# Make color pallette table file
echo Making color pallette table file...
echo      0 255   0 255    100 128   0 255 > $CPT_FILE
echo    100 128   0 255    200   0   0 255 >> $CPT_FILE
echo    200   0   0 255    300   0 128 255 >> $CPT_FILE
echo    300   0 128 255    400   0 255 255 >> $CPT_FILE
echo    400   0 255 255    500   0 255   0 >> $CPT_FILE
echo    500   0 255   0    600 128 255   0 >> $CPT_FILE
echo    600 128 255   0    700 255 255   0 >> $CPT_FILE
echo    700 255 255   0    800 255 128   0 >> $CPT_FILE
echo    800 255 128   0    900 255  64   0 >> $CPT_FILE
echo    900 255  64   0   1000 255   0   0 >> $CPT_FILE
#
```

Place "#" symbols before each echo statement above and save the file. Then give it a spin. The results will look like this:

Figure 5.7. R/V Ewing Survey: Gridded Standard Deviation With New Color Map



By rescaling our color table, we have omitted the standard deviations that are greater than 100 meters, in this way we can more clearly see the variation less than 100 meters, and values greater than 100 meters produce white space.

Of the values that are plotted, these have been scaled against the Haxby color pallette shown in the scale bar at the bottom. This new scale allows us to see the changes in variability, if any, in the 0 to 100 range.

Here we can see that indeed, there is less absolute variability in the data collected in the shallow water.

We know to expect this, since data accuracy is depth dependent - usually expressed as a percentage of absolute water depth. At depths of 4000 meters, even in very rough terrain, this plot rarely shows variability greater than 50 meters. That translates to about .1% error - a very low figure.

That brings up another step that we might have taken to more quantitatively compare this data. GMT provides a function called gridmath that would have allowed us to calculate the standard deviation as a percent of water depth. Without explaining the details, here is how it would be done:

```
# Do the math
grdmath survey-datalist_sd.grd survey-datalist.grd DIV 100 MUL \
   = survey-datalist_sd-percent.grd

# Generate the plot script.
mbm_grdplot -F survey-datalist_sd-percent.grd -G1 -W1/2 -V \
   -L'Standard Deviation as a Percent of Water Depth'
```

One final note, we would be remiss if we didn't point out that the lower data density at greater depths skews these statistics somewhat. To more accurately conduct this analysis, a data set with a more uniform depth profile should be selected with sufficient overlapping runs such that the data density per bin is large. Then the standard deviation of the binned data will provide a more statistically significant measure of the sonar's performance. None-the-less, our plot depicts a reasonable approximation.

Appendix A. Acknowledgments

MB-System was originally developed at the Lamont-Doherty Earth Observatory of Columbia University (L-DEO) and is now a collaborative effort between the Monterey Bay Aquarium Research Institute (MBARI) and L-DEO.

The National Science Foundation (NSF) has been the primary supporter of MB-System™ development. NSF initially provided grants to the authors at the Lamont-Doherty Earth Observatory in 1993 (two years), and 1995 (five years). SeaBeam Instruments also contributed significant effort from 1994 to 1999. NSF support has been renewed through a new five year grant (2001-2005) to Caress at the Monterey Bay Aquarium Research Institute (MBARI) and to Chayes at Lamont. MBARI is matching the NSF support as part of its seafloor mapping program. In addition to Dave and Dale, the following people of contributed code and/or ideas to MB-System™ :

- Suzanne O'Hara (Lamont-Doherty Earth Observatory)
- Daniel Scheirer (Brown University)
- Paul Cohen (SeaBeam Instruments, now Fidelity Investments)
- Steve Dzurenko (SeaBeam Instruments, now University of Texas)
- Peter Lemmond (Woods Hole Oceanographic Institution)
- David Brock (formerly with Antarctic Support Associates)
- Alberto Malinverno (Lamont-Doherty Earth Observatory, later at Schlumberger-Doll)
- Russ Alexander (formerly with UCSB)
- Roger Davis (University of Hawaii)

Additionally, vendors and users, too numerous to mention, have provided invaluable information regarding data file formats and sonar system specific characteristics.

Appendix B. MB-System™ Command Reference

- hsdump

```
hsdump  [ -F format  -I infile  -O kind  -V -H ]
```

  hsdump lists the information contained in data records on Hydrosweep DS data files, including survey, calibrate, water velocity and comment records. The default input stream is stdin.

- mbanglecorrect

```
mbanglecorrect  [ -A kind/scale  -B yr/mo/da/hr/mn/sc
```

  mbanglecorrect corrects the sidescan data by dividing by a model of the backscatter vs grazing angle function to produce a flat image which shows geology better than the raw data

- mbbackangle

```
mbbackangle  [ -A kind
```

  mbbackangle generates tables of the average amplitude and/or sidescan values as a function of the grazing angle with the seafloor. These tables can be used by the program mbprocess to correct the sidescan or amplitude data for the variation with grazing angle.

- mbbackangleold

  mbbackangleold generates a table of the average amplitude or sidescan values as a function of the grazing angle with the seafloor.This program replicates the functionality of the mbbackangle program included in MB-System version 4 distributions. The current version of mbbackangle generates a series of average sidescan tables in an "amplitude vs grazing angle" or ".aga" file that can be used by mbprocess. See the mbbackangle and mbprocess manual pages for details.

- mbbath

  mbbath is a utility for recalculating swath bathymetry data using new values for some of the fundamental sonar operational parameters. Users may apply changes to the water sound velocity profile, roll bias, pitch bias, and draft parameters. Static corrections can also be applied to the depth values.

- mbclean

  mbclean identifies and flags artifacts in swath sonar bathymetry data. Several algorithms are available for identifying artifacts; multiple algorithms can be applied in a single pass. The most commonly used approach is to identify artifacts based on excessive bathymetric slopes.

- mbcleanold

  mbcleanold identifies and flags artifacts in swath sonar bathymetry data. This program replicates the functionality of the mbclean program included in MB-System™ version 4 distributions. The input swath data is read into a buffer, processed, and then written directly into an output swath data file.

The current version of mbclean generates an "edit save file" like that of mbedit rather than directly outputting a swath file. See the mbclean manual page for details.

- mbcontour

  mbcontour is a utility for swath contouring of swath bathymetry data using Postscript. Like mb-swath, mbcontour is fully compatible with the GMT package version 3.

- mbcopy

  mbcopy is a utility for copying swath sonar data files which may be used to change formats, average pings, or window in time and space. mbcopy can be used as a filter from STDIN to STDOUT, or it may do i/o to and from files.

- mbcut

  mbcut removes swath data values that lie in user-specified ranges of beam numbers, pixel numbers, or across track distances.

- mbdatalist

  mbdatalist is a utility for parsing datalist files. Datalist files, or lists of swath data files and their format ids, are used by a number of MB-System™ programs. These lists may contain references to other datalists, making them recursive. See the MB-System™ manual page for details on the format and structure of datalists.

- mbdefaults

  mbdefaults is a utility for setting or listing the default read parameters used by some MBIO utilities such as mbcopy. If a user wishes to set default parameters different from those set by the MBIO library, then these specialized default values must be stored in the file .mbio_defaults in the users home directory. If any option flag is given in invoking mbdefaults, then a new .mbio_defaults file will be written incorporating the newly set parameters along with any old parameters which have not been reset.

- mbedit

  mbedit is an interactive editor used to identify and flag artifacts in swath sonar bathymetry data. Once a file has been read in, MBedit displays the bathymetry profiles from several pings, allowing the user to identify and flag anomalous beams.

- mbeditold

  mbeditold is the old version of an interactive editor used to identify and flag artifacts in swath sonar bathymetry data. The current version of mbedit works in conjunction with the mbprocess utility and uses a different i/o scheme. This version has been retained for compatibility with previous releases of MB-System™, but may be dropped from future releases.

- mbfilter

  mbfilter applies one or more simple filters to the specified swath data (sidescan, beam amplitude, and/or bathymetry).

- mbformat

  mbformat is a utility which identifies the swath sonar data formats associated with mbio format ids. If no format id or input filename is specified, mbformat lists all of the currently supported formats.

- mbgetesf

mbgetesf is a utility to extract swath bathymetry beam flags into tan edit save file such as that produced by mbedit or mbclean.

- mbgetmask

  mbgetmask and mbmask allow the user to extract the flagging information in the form of a "mask" file from the edited file and then to apply the flagging mask to another file containing a different version of the same data. The first utility, mbgetmask, is used to extract the mask from the edited swath bathymetry file; the output is written to STDOUT. The second utility, mbmask, reads in the mask file and the unedited swath data file and flags or unflags all beams indicated by the mask.

- mbgrdtiff

  mbgrdtiff generates a TIFF image from a GMT grid. The image generation is similar to that of the GMT program grdimage. In particular, the color map is applied from a GMT CPT file, and shading overlay grids may be applied. The output TIFF file contains information allowing the ArcView and ArcInfo GIS packages to import the image as a geographically located coverage. The image is 8 bits per pixel if the color map is a grayscale, and 24 bits per pixel otherwise. In order to automatically generate a reasonable image of a grid, use the macro mbm_grdtiff.

- mbgrid

  mbgrid is a utility used to grid bathymetry, amplitude, or sidescan data contained in a set of swath sonar data files. This program uses one of four algorithms to grid regions covered by swath sonar swathes and then can fill in gaps between the swathes (to the degree specified by the user) using a thin plate spline interpolation.

- mbhistogram

  mbhistogram reads a swath sonar data file and generates a histogram of the bathymetry, amplitude, or sidescan values. Alternatively, mbhistogram can output a list of values which break up the distribution into equal sized regions. These values can be used to construct a color table, producing an image which is histogram equalized.

- mbinfo

  mbinfo is a utility for reading a swath sonar data file or files and outputting some basic statistics. The table generated shows the filename, the data format id, a brief description of the format, any metadata that have been inserted into the data, data counts, navigation totals, time and navigation of the first and last data records, minimum and maximum data values, and the geographic bounding box of the data. The data counts include the total number of bathymetry, amplitude, and sidescan values read. Also reported are the numbers and percentages of good, zero, and flagged data values (good values are those which are neither zero nor flagged as bad).

- mblevitus

  mblevitus generates a mean water sound velocity profile for a specified location using temperature and salinity data from the 1982 Climatological Atlas of the World Ocean [Levitus, 1982].

- mblist

  mblist is a utility to list the contents of a swath data file or files to STDOUT. By default, mblist produces ASCII files in spreadsheet style, with data columns separated by tabs. Alternatively, the output can be binary, with each field represented as a double precision float (A option).

- mbm_arc2grd

  mbm_arc2grd is a macro to convert a ArcView ASCII grid to an GMT grid file in the GMT NetCDF

grid format. This allows users to import the grid into GMT.

- mbmask

  mbmask and mbmask allow the user to extract the flagging information in the form of a "mask" file from the edited file and then to apply the flagging mask to another file containing a different version of the same data. The first utility, mbgetmask, is used to extract the mask from the edited swath bathymetry file; the output is written to STDOUT. The second utility, mbmask, reads in the mask file and the unedited swath data file and flags or unflags all beams indicated by the mask.

- mbm_dslnavfix

  mbm_dslnavfix is a macro to translate WHOI DSL AMS 120 navigation from UTM eastings and northings to longitude and latitude. The WHOI DSL group typically provides processed AMS 120 navigation is files separate from the bathymetry and sidescan data. Moreover, the navigation is generally sampled less frequently than the sonar pings, and is reported in UTM easting and northing meters. The mbm_dslnavfix macro is used to translate the eastings and northings into the geographic coordinates (longitude and latitude) used by MB-System™ programs. The program mbmerge can be used to merge the translated navigation in with the bathymetry and sidescan.

- mbmerge

  mbmerge is a utility for merging navigation with swath sonar echosounder data.Many data formats include the navigation information in the same data records as the ping or survey data. In this case, mbmerge replaces the position values in the file by interpolation of the values in the navigation file. Some data formats use asynchronous navigation, which means that the navigation is contained in data records separate from the data records containing sonar ping data. The time stamps and frequency of the navigation records are in general different from those of the ping data records. Examples include all of the Simrad vendor data formats and all of the "UNB" formats for Reson SeaBat and Elac Bottomchart sonars. For data formats with asynchronous navigation, MB-System™ programs calculate the positions of ping data by interpolation and/or extrapolation of the navigation data.

- mbm_fmtvel

  mbm_fmtvel is a macro that scans a Hydrosweep swath sonar data file using the program hsdump and generates a list in columnar format with time, date, latitude, longitude, C mean, and C keel entries.

- mbm_grd2arc

  mbm_grd2arc is a macro to convert a GMT grid file in the GMT NetCDF grid format to an ArcView ASCII grid. This allows users to import the grid into Arc/Info and ArcView.

- mbm_grd2geovrml

  mbm_grd2geovrml is a macro that takes as input a GMT geographic coordinate bathymetry grid file (bath_file) and from it generates a shaded image which is paired with the elevation data in the grid file to create a TerraVision tile set. This tile set is then used to generate a GeoVRML quadtree multi resolution set of files.

- mbm_grd3dplot

  mbm_grd3dplot is a macro to generate a shellscript of GMT commands which, when executed, will generate a 3D perspective Postscript plot of gridded data. Several styles of plots can be generated, including color fill views, color shaded relief views, mesh plot views, and text labels.

- mbm_grdcut

mbm_grdcut is a macro to painlessly cut out a region from a GMT GRD grid file. The GMT program grdcut requires that one specify bounds which exactly match grid cell boundaries. Frequently, one just wants to extract an approximate region quickly, without calculating grid cell sizes and boundary locations. This macro does the the calculations and extracts the subregion closest to that specified by the user.

- mbm_grdinfo

  mbm_grdinfo is a macro to get information regarding a GMT GRD file when the region of interest is a subset of the area covered in the input file. If no bounds are specified, the program grdinfo is called directly. If longitude and latitude bounds are specified, then the specified region is cut from the input file using the macro mbm_grdinfo, and the information is obtained from the subset temporary GRD file using grdinfo.

- mbm_grdplot

  mbm_grdplot is a macro that generates a shellscript of GMT commands which, when executed, will generate a Postscript plot of gridded data. Several styles of plots can be generated, including color fill maps, contour maps, color fill maps overlaid with contours, shaded relief color maps, slope magnitude maps, coastline maps, text labels, and xy data in lines or symbols.

- mbm_grdtiff

  mbm_grdtiff is a macro to generate a shellscript which, when executed, will generate a geographically located TIFF image of gridded data. The primary purpose of this macro is to allow the simple, semi-automated production of a nice looking image with a few command line arguments. This image can be loaded into the ArcInfo and ArcView GIS packages as geographically located coverages.

- mbm_grid

  mbm_grid is a macro to generate a shellscript of MB-System™ commands which, when executed, will generate a grid or mosaic of the specified swath sonar data. The primary purpose of this macro is to allow the simple, semi-automated production of grids and mosaics with a few command line arguments.

- mbmosaic

  mbmosaic is a utility used to mosaic amplitude or sidescan data contained in a set of swath sonar data files. This program allows users to prioritize data according to the associated grazing angle and according to look azimuth. Individual mosaic bin values can be either the value of the highest priority sample in the bin or the Gaussian weighted mean of the highest priority samples in the neighborhood of the bin (the samples used here are those with priorities within a specified range of the highest priority sample found). Users can thus construct mosaics which are dominantly from a particular part of the swath (e.g. prioritize the outer swath higher than the inner swath) or which are dominantly from a particular look azimuth (e.g. construct an east looking mosaic by specifying a preferred look azimuth of 90 degrees).

- mbm_plot

  mbm_plot is a macro to generate a shellscript of MB-System™ and GMT commands which, when executed, will generate a Postscript plot of the specified swath sonar data. The plot may include bathymetry color fill, bathymetry color shaded relief, bathymetry shaded with amplitudes, greyshade fill amplitude, greyshade fill sidescan, contoured bathymetry, or annotated navigation. The plot may also include text labels, xy data in lines or symbols, and coastlines.

- mbm_rollerror

  mbm_rollerroris a macro which generates cross track slope corrections from the apparent vertical

reference noise of swath sonar data.

- mbm_stat

  mbm_stat mbm_stat is a Perl shellscript that extracts beam statistics from the output of mbinfo. The output of contains the base name of the data file, then number of hours of data in the data file, and the percentage of dropped and flagged beams. One line of output for each run of mbinfo is generated. This allows mbm_stat to be run on accumulated data such as might be generated by logging the daily mail statistics. Columns are tab delimited.

- mbm_utm

  mbm_utm is a macro to perform forward and inverse UTM projections of ASCII data triples using the GMT program mapproject.

- mbm_vrefcheck

  mbm_vrefcheck is a macro which generates a plot of high pass filtered apparent cross track seafloor slope.

- mbm_xbt

  mbm_xbt is a Perl shellscript to translate various XBT data sets from depth and temperature into depth and sound velocity.

- mbm_xyplot

  mbm_xyplot is a macro to generate a shellscript of GMT commands which, when executed, will generate a Postscript plot of xy data. Axes may be linear, log, or any of several geographic projections. Data may be plotted as symbols or lines. The plot will be scaled to fit on the specified page size or, if the scale is user defined, the page size will be chosen in accordance with the plot size. The primary purpose of this macro is to allow the simple, semi-automated production of nice looking plots with a few command line arguments.

- mbnavadjust

  mbnavadjust is an interactive graphical program used to adjust swath data navigation by matching bathymetric features in overlapping and crossing swathes. The primary purpose of mbnavadjust is to eliminate relative navigational errors in swath data obtained from poorly navigated sonars.

- mbnavedit

  mbnavedit is an interactive editor used to identify and fix problems with the navigation of swath sonar data. In the default mode the edited navigation is output to a file with the suffix ".nve" appended to the input swath data filename. The program can also be operated in a "browse" mode where no navigation is output. If saved, the edited navigation can be merged with the swath data using the program mbprocess, which outputs a processed swath data file.

- mbnaveditold

  mbnaveditold is an interactive editor used to identify and fix problems with the navigation of swath sonar data. The current version of mbnavedit works in conjunction with the mbprocess utility and uses a different i/o scheme. This version has been retained for compatibility with previous releases of MB-System™, but may be dropped from future releases.

- mbnavlist

  mbnavlist prints the specified contents of navigation records in a swath sonar data file to STDOUT.

By default, mbnavlist produces ASCII files in spreadsheet style, with data columns separated by tabs. Alternatively, the output can be binary, with each field represented as a double precision float.

- mbprocess

  mbprocess is a tool for processing swath sonar bathymetry data. This program can perform a variety of swath data processing functions in a single step (producing a single output swath data file), including:

  - Merge edited navigation generated by mbnavedit.
  - Apply bathymetry edit flags from mbedit and mbclean
  - Recalculate bathymetry from raw travel time and angle data by ray tracing through water sound speed models from mbvelocitytool or mbsvplist.
  - Apply changes to roll bias, pitch bias, heading bias, and draft values.
  - Recalculate sidescan from raw backscatter samples (Simrad multibeam data only).
  - Apply corrections to sidescan based on amplitude vs grazing angle tables obtained with mbbackangle.
  - Insert metadata.

- mbps

  mbps is a utility to generate an (almost correct) perspective view of a piece of swath sonar data. It is especially useful to get a detailed view of the quality of the data (which is not very well assessed in a contour plot) and to make pretty pictures of features that fit within a swath.

- mbrollbias

  mbrollbias is a utility used to assess roll bias of swath sonar sonar systems using bathymetry data from two swathes covering the same seafloor in opposite directions. The program takes two input files and calculates best fitting planes for each dataset. The roll bias is calculated by solving for a common roll bias factor which explains the difference between the seafloor slopes observed on the two swathes.

- mbset

  mbset is a utility for creating and modifying mbprocess parameter files.

- mbsimradmakess

  mbsimradmakess is a utility for regenerating sidescan imagery from the raw amplitude samples contained in data from Simrad EM300 and EM3000 multibeam sonars. This program ignores amplitude data associated with flagged (bad) bathymetry data, thus removing one important source of noise in the sidescan data.

- mbsmooth

  mbsmooth applies a spatial domain Gaussian filter to swath sonar bathymetry data in order to smooth out noise in the data. The width of the filter can be varied as a function of beam number. The default input and output streams are STDIN and STDOUT.

- mbstripnan

mbstripnan is a utility for removing NaN nodes that are produced by the GMT utilities grd2xyz and blockmean with the bo option. The standard output of mbstripNaN may be fed into the standard input of surface with the bi option. This is used by the mbm_grd2geovrml utility in order to spline fill areas of no data so that irregular bathymetry may be effectively visualized using GeoVRML which has no concept of NaN.

- mbsvplist

  mbsvplist lists all water sound velocity profiles (SVPs) within swath data files. Swath bathymetry is calculated from raw angles and travel times by ray tracing through a model of the speed of sound in water. Many swath data formats allow SVPs to be embedded in the data, and often the SVPs used to calculate the data will be included.

- mbswath

  mbswath is a utility for plotting swath sonar data in color fill or color shaded relief using Postscript. Like mbcontour, mbswath is fully compatible with the GMT package version 3, including the use of GMT style color palette (cpt) files to control the color table.

- mbtide

  mbtide is a utility for correcting swath bathymetry for tides. The input tide data can be in one of several ASCII table formats. A simple spline interpolation of the tide values is used to calculate the tide correction for each ping.

- mbunclean

  mbunclean is a utility for reversing previous attempts to edit swath sonar bathymetry data. Bad data is conventionally flagged by setting depth values negative; mbunclean sets negative depths positive. All flagged depths will be unflagged unless depth range checking is specified and the depth value in question is outside the acceptable range. The default input and output streams are STDIN and STDOUT.

- mbvelocitytool

  mbvelocitytool is an interactive water sound velocity profile (SVP) editor used to examine multiple SVPs, to create new SVPs, and to model the impact of SVP modification on swath bathymetry data. SVPs created using MBvelocitytool can be used by the program mbprocess to recalculate swath bathymetry from raw travel time and angle data.

Appendix C. Installing MB-System™

C.1. Overview

Installing MB-System™ is generally straight forward. A brief summary of the required steps follows:

- Download the MB-System™ package.
- Install GMT and netCDF if you haven't already. If you have, locate these packages on your system. Verify that you have Perl installed on your system as well.
- Create a home directory structure for MB-System™ and unpack the tar ball.
- Edit a copy of the "install_makefiles" script, specifying your parameters.
- Run "install_makefiles" to modify the default Makefile's for your parameters.
- Build the source with *make*, and install it with *make install*.

In general you do not need root privileges to compile MB-System™. (This is somewhat dependent on where you chose to put package, of course.) However if you are starting from scratch, you may need them to compile and install *GMT*.

C.2. Downloading MB-System™

The MB-System™ home page is at

http://www.ldeo.columbia.edu/MB-System/MB-System.intro.html

and the software package can be downloaded from

ftp://ftp.ldeo.columbia.edu/pub/mbsystem

Typically, you will find the latest full release of MB-System™ and several beta versions containing bug fixes and incremental increases in functionality. In general it is usually safe and recommended to get the latest beta version of the software, rather than the last full version. Beta versions often contain considerable increases in functionality, while full versions are issued with much reluctance and may be more than a year old.

Compressed and tar'd packages might have a naming convention similar to

```
MB-System.5.0.beta31.tar.gz
```

Be sure to get a copy of the associated README. And you'll also want a copy of the Levitus temperature and salinity database, which you'll find in "annual.gz" or "annual.Z".

C.3. Installing GMT and netCDF

The *Generic Mapping Tools* and *netCDF* libraries are required to be installed before installing MB-System™. You'll find the GMT package at

http://gmt.soest.hawaii.edu/.

Perhaps the easiest way to install GMT and netCDF is using GMT's automated install script. The script needs lots of details about your system and your preferences regarding the installation, so to make answering these questions more straight-forward, the folks at GMT have created a web form. One answers the questions in the web page, and then clicks "GET PARAMETERS". The web page creates a properly formatted parameter file that can be saved and referenced from the GMT install script to semi-automate the rest of the install. The web form can be found at http://gmt.soest.hawaii.edu/gmt/gmt_install_form.html.

With the parameters correctly specified, one can download and install GMT with execution of the GMT-install script (also available for download from the web form page above):

```
./install_gmt GMTparams.txt > install.log 2>&1
```

This gives you a nice log of everything that's going on during the install which you can monitor with

```
tail -f install.log
```

C.4. Creating a Directory Structure for MB-System™

Different system administrators and users have different philosophies regarding how to install and maintain software packages on Unix systems. Some will invariably disagree with elements of this strategy, but it has worked well in the past so I will present it here.

In general, when installing MB-System™ it is good to create a parent directory. This might be located in /usr/local or /usr/local/packages/ or in your $HOME/packages if you haven't the requisite permissions to do otherwise. Typically this parent directory is simply called *mbsystem*.

Then download the MB-System™ package and move it to this parent directory. Uncompress and un-tar this package with something like

```
tar -xvpzf MB-System.5.0.beta31.tar.gz
```

The resulting extracted files will be in a subdirectory called *mbsystem*. Then make a new directory in the parent directory that will uniquely identify this release of MB-System™, perhaps *mbsystem5b31*.

Finally, move the contents of the extracted package to the new sub-directory and remove the old generic "*mbsystem* sub-directory

```
mv mbsystem/* mbsystem5b31/
rmdir mbsystem
```

Now the directory structure will look something like the following:

```
/usr/local/packages/mbsystem/
/usr/local/packages/mbsystem/mbsystem5b31/
```

Then subsequent installations of new releases can be placed in similar, uniquely named subdirectories within /usr/local/packages/mbsystem/, e.g. mbsystem5b32, mbsystem5b33, etc. In this way, administrators can easily maintain several working versions of the software, and users need only change their PATH variable to revert to an older version if required.

One final thing is to uncompress the Levitus temperature and salinity database, name it something appropriate, and copy it to an appropriate place. A common place to put it is in /usr/local/packages/mbsystem5b31/share/ or if you'd prefer something more generic, perhaps /usr/local/data/. Either way, note where you've put it, as you'll need the path later on.

C.5. Editing "install_makefiles"

Assuming you have downloaded and unpacked the MB-System™ package, you will find the contents to include a perl script called install_makefiles. When properly configured, this script will rewrite the various "Makefile" files for proper compilation and installation of the software with the make utility.

Note

If you're not already somewhat familiar with the make utility this might all seem very confusing. From the make man page:

```
The purpose of the make utility is
to determine automatically which pieces of a large program need
to be recompiled, and issue the commands to recompile
them.
```

The make utility takes a file of instructions (called "Makefile" by default), as an argument. Information in this file provides detailed instructions for compiling portions or all of a software package, and often also includes instructions for installing the software or optionally removing installed software to prepare for a new compilation and install.

Several templates of these "Makefile" files exist in various parts of the source directory for MB-System™. Execution of the install_makefiles script simply rewrites these files to customize them for your installation.

It is best to copy the script to a new file and edit the new version. This allows one to keep a customized copy of your install_makefiles script that can be used with subsequent upgrades, usually without further editing.

```
cp install_makefiles install_makefiles.yourlaptop
```

There is only a portion of the install_makefiles script that one should edit, and unfortunately, it is somewhat buried in the middle. Much of the front matter is commented out and provides several examples. The portion to edit is well marked and looks something like the following:

```
# **** EDIT THE PARAMETERS HERE ****
# Set the environment parameters:
$MBSYSTEM_HOME = "/usr/local/mbsystem/mbsystem-5.0.4";
$OS = "LINUX";
$CC = "gcc";
$BYTESWAPPED = "YES";
$GRAPHICAL = "YES";
$MOTIFINCDIR = "/usr/X11R6/include";
$MOTIFLIBS = "-lXm -L/usr/X11R6/lib -lXt -lX11";
$OPENGLLIBS = "-lGLw -lGLU -lGL";
$GMTVERSION = "4.0";
$GMTLIBDIR = "/usr/local/gmt/GMT4.0/lib";
$GMTINCDIR = "/usr/local/gmt/GMT4.0/include";
$NETCDFLIBDIR = "/usr/local/gmt/netcdf-3.5.0/lib";
$NETCDFINCDIR = "/usr/local/gmt/netcdf-3.5.0/include";
```

```
$LEVITUS = "$MBSYSTEM_HOME/share/LevitusAnnual82.dat";
$PROJECTIONS = "$MBSYSTEM_HOME/share/Projections.dat";
$CFLAGS = "-g -w";
$LFLAGS = "-lm -L$NETCDFLIBDIR -lnetcdf";

# **** LEAVE EVERYTHING ELSE BELOW ALONE ****
```

A description of these parameters is included in the script and is reproduced here:

```
# The important parameters to be set are:
#       MBSYSTEM_HOME   Absolute path to the directory containing
#                               this file.
#       OS              Operating system (SUN, IRIX, LINUX, LYNX,
#                               SOLARIS, HPUX, or OTHER).
#       BYTESWAPPED     If set to "YES" will enable byte swapping
#                               of input and output binary data.
#                               Otherwise byte swapping is disabled.
#                               Byte swapping needs to be enabled
#                               when installing on "little endian"
#                               machines such as PCs or VAXs.
#       CC              C compiler to be used (optional).
#       GRAPHICAL       If set to "YES" will enable the installation
#                       of the Motif based graphical utilities mbedit,
#                       mbnavedit, and mbvelocitytool. If Motif
#                       is not available but the nongraphical utilities
#                       are desired then disable with "NO".
#                       (optional, default is "YES")
#       MOTIFINCDIR     Location of Motif include files.
#       MOTIFLIBS       X and Motif libraries required for graphical tools.
#       OPENGLLIBS      OpenGL libraries required for new graphical tools,
#.                              including MBgrdviz. If this parameter is
#                               not set,
#       GMTVERSION      Version of GMT to be used and linked with. GMT
#                       versions prior to 4.0 are no longer supported, so
#                       under most circumstances this parameter need not
#                       be set.
#                               The default is 4.0.
#
#       GMTLIBDIR       Location of GMT libraries libgmt.a and libpsl.a
#       GMTINCDIR       Location of GMT include files gmt.h, grd.h,
#                               and pslib.h
#       NETCDFLIBDIR    Location of NetCDF library libnetcdf.a
#       NETCDFINCDIR    Location of NetCDF include file netcdf.h
#       LEVITUS         Path for Levitus data file annual
#                       This is usually $MBSYSTEM_HOME/share/annual.
#       CFLAGS          Compile flags for C source files.
#       LFLAGS          Load flags for all object files.
```

From the above, one can see that the parameters to be specified include compilation optimization options, paths for libraries to be linked, and references to installed software such as GMT, and NetCDF. Perhaps the most difficult details to get correct are the Motif libraries and compile flags (CFLAGS and LFLAGS). Look to the examples for guidance and write to the Dave, Dale or Val if you have questions.

C.6. Run install_makefiles and Compile MB-System™

The final two steps are to run the install_makefiles.yourlaptop script, and to compile the software. Run install_makefiles.yourfile simply with

```
./install_makefiles.yourfile

Parameters defined for Makefiles:
Mb-System Home:            /usr/local/mbsystem/mbsystem-5.0.7
Operating System:          LINUX
Byte swapping:             enabled
C Compiler:                gcc
C Compile Flags:           -g -w -DLINUX -DBYTESWAPPED -DGMT4_0
Load Flags:                -lm -L/usr/local/gmt/netcdf-3.5.0/lib -lnetcdf -L/usr/l
Library Archiver:          ar rcv
Graphical Utilities:       enabled
Motif Include Location:    /usr/X11R6/include
Motif Libraries:           -lXm -L/usr/X11R6/lib -lXt -lX11
OpenGL Libraries:          -lGLw -lGLU -lGL
GMT Library Location:      /usr/local/gmt/GMT4.0/lib
GMT Include Location:      /usr/local/gmt/GMT4.0/include
NetCDF Library Location:   /usr/local/gmt/netcdf-3.5.0/lib
NetCDF Include Location:   /usr/local/gmt/netcdf-3.5.0/include
Levitus Data Location:     /usr/local/mbsystem/LevitusAnnual82.dat
Projections Data Location: /usr/local/mbsystem/mbsystem-5.0.7/share/Projections.da

Makefile Template: Makefile.template
Output Makefile:   ./Makefile

Makefile Template: src/gmt/Makefile.template
Output Makefile:   src/gmt/Makefile

Makefile Template: src/gsf/Makefile.template
Output Makefile:   src/gsf/Makefile

Makefile Template: src/macros/Makefile.template
Output Makefile:   src/macros/Makefile

Makefile Template: src/Makefile.template
Output Makefile:   src/Makefile

Makefile Template: src/mbaux/Makefile.template
Output Makefile:   src/mbaux/Makefile

Makefile Template: src/mbedit/Makefile.template
Output Makefile:   src/mbedit/Makefile

Makefile Template: src/mbeditold/Makefile.template
Output Makefile:   src/mbeditold/Makefile

Makefile Template: src/mbio/Makefile.template
Output Makefile:   src/mbio/Makefile

Makefile Template: src/mbnavadjust/Makefile.template
Output Makefile:   src/mbnavadjust/Makefile

Makefile Template: src/mbnavadjustsave/Makefile.template
Output Makefile:   src/mbnavadjustsave/Makefile

Makefile Template: src/mbnavedit/Makefile.template
Output Makefile:   src/mbnavedit/Makefile

Makefile Template: src/mbnaveditold/Makefile.template
Output Makefile:   src/mbnaveditold/Makefile

Makefile Template: src/mbvelocitytool/Makefile.template
Output Makefile:   src/mbvelocitytool/Makefile
```

```
Makefile Template: src/mbview/Makefile.template
Output Makefile:   src/mbview/Makefile

Makefile Template: src/mr1pr/Makefile.template
Output Makefile:   src/mr1pr/Makefile

Makefile Template: src/proj/Makefile.template
Output Makefile:   src/proj/Makefile

Makefile Template: src/surf/Makefile.template
Output Makefile:   src/surf/Makefile

Makefile Template: src/utilities/Makefile.template
Output Makefile:   src/utilities/Makefile

All done!
```

If your output looked something like above (of course with your specific parameters) then you are ready to compile. To compile MB-System™ execute

```
make
```

Now go get a cup of coffee. The make command will descend recursively down through the directory structure, compiling source files, linking object files, and copying executables and manual pages to the appropriate directories. The executables will be placed in mbsystem/bin, the libraries in mbsystem/lib, and the manual pages in mbsystem/man/man1.

If there is an error, scroll back through the output and see if you can find the source. If there are no overt errors when the process completes, you are probably ready to take things for a spin. It's a good idea to try things out with mbm_plot, and mbedit as these will utilize enough of the software that you are likely to catch any major problems.

Now that you've compiled (and installed) MB-System™ you should edit your PATH variable, typically in .cshrc for c-shell or .bash_profile for the bash shell.

Finally, you may also want to specify the default viewer for postscript files (which is called automatically from utilities such as mbm_plot). This can be set as an environment variable (PS_VIEWER), or with the mbdefaults utility, which will store the value (and any other changes from the installed default) in their $HOME/.mbdio_defaults. The default for the postscript viewer is ghostview, usually executed with gv on unix machines.

Appendix D. Other Useful Tools

Words about the other useful tools

Appendix E. References

The references listed below were used, in part, in the creation of the MB-System Cookbook as well as MB-System itself. We hope that they prove as helpful to you as they have to us.

E.1. Acoustic References

Mackenzie, K. V. (1981). "Nine-term Equation for Sound Speed in the Oceans." J. Acoust. Soc. Am. 70(807).

Manley, P. L. and D. W. Caress (1994). "Mudwaves on the Gardar Sediment Drift,NE Atlantic." Paleoceanography 9(6): 973-988.

Medwin, H. (1975). "Speed of Sound in Water for Realistic parameters." J. Acoust. Soc. Am. 58(1318).

Miller, S. P. (1991). "3-D Bathymetry Imaging: State of the Art Visualization."Sea Technology June(June): 27-31.

Nielson, R. O. (1991). Sonar Signal Processing. Norwood, MA, Artech House.

Phillips, J. C., J. L. Abbott, et al. (1988). Multiple Sound Source synchronizer for Seafloor Surveying. Offshore Technology Conference, Houston, TX., OTC.

Robinson, A. R. and D. Lee, Eds. (1994). Oceanography and Acoustics Prediction and Propagation Models. Modern Acoustics and Signal Processing. Woodbury, NY, AIP Press.

Spiesberger, J. L. and K. Metzger (1991). "A new algorithm for sound speed in seawater." Jour. Acoust. Soc. Amer. 89: 2677-2688.

Spiesberger, J. L. and K. Metzger (1991). "New estimates of sound speed in water." Jour. Acoust. Soc. Amer. 89: 1697-1700.

Teague, W. J., M. J. Carron, et al. (1990). "A comparison between the Generalized Digital Environmental Model and Levitus climatologies." J. Geophys. Res. 95(C5): 7167-7183.

Urick, R. J. (1983). Principles of Underwater Sound. New York, McGraw-Hill BookCompany.

Wilson, W. (1960). "Equation for the Speed of Sound in Sea Water." Jour. Acoust. Soc. Amer. 32(10): 1357.

Lurton, X. (2002). An Introduction to Underwater Acoustics: Principles and Applications. Chichester, UK, Springer/Praxis.

Lewis, E. L. (1980). "The Practical Salinity Scale 1978 and Its Antecedents." IEEE Journal of Oceanic Engineering OE-5(1): 3-8.

Jensen, F. B., W. A. Kuperman, et al. (1994). Computational Ocean Acoustics. Woodbury, NY, AIP Press.

Grosso, V. A. D. and C. W. Mader (1972). "Speed of Sound in Pure Water,." Jour. Acoust. Soc. Amer. 52: 1442-1446.

Grosso, V. A. D. (1974). "New Equation for the Speed of Sound in

Natural Waters(with Comparison to Other Equations)." Jour. Acoust. Soc. Amer. 56(4): 1084-1091.

Glenn, M. F. (1970). "Introducing an Operational Multibeam Array Sonar." International Hydrographic Review XLVII(1): 35-36.

Foote, K. and T. Lassen (2003). Developing Acoustic Methods for Surveying Groun dfish. Rockport, Maine, Alliance for Coastal Technologies: 22.

Faulconbridge, I. (2002). Radar Fundamentals. Red Hill ACT 2603 Australia, Argos Press.

Fofonoff, N. P. and R. C. Millard (1983). "Algorithms for Computation of Fundamental Properties of Sea-Water." UNESCO Tech. Pap. in Mar. Sci. 44: 53.

Etter, P. C. (1992). Underwater Acoustic Modeling: Principles, Techniques and Applications. New York, Elsevier.

Dushaw, B. D., P. F. Worcester, et al. (1991). "On equations for the speed of sound in seawater." Jour. Acoust. Soc. Amer. 89: 1697-1700.

Denbigh, P. N. (1994). "Signal processing strategies for a bathymetric sidescan sonar." IEEE J. Ocean. Eng. 19(3): 382-390.

Denbigh, P. N. (1989). "Swath bathymetry: Principles of operation and an analysis of errors." IEEE J. Ocean. Eng. 14: 289-298.

de Moustier, C., P. F. Lonsdale, et al. (1990). "Simultaneous Operation of the Sea Beam Multibeam Echo-Sounder and the SeaMARC II Bathymetric Sidescan Sonar System." IEEE J. of Oceanic Engineering 15(2): 84-94.

de Moustier, C. and M. C. Kleinrock (1986). "Bathymetric artifacts in Sea Beam data: how to recognized them and what causes them." J. Geophys. Res. 91: 3407.

de Moustier, C., J. C. Charters, et al. (1994). "Processing and Display Techniques for Sea Beam 2000 Bathymetry and Acoustic Backscatter Amplitude Data Collected Aboard R.V. Melville." EOS, Transactions, American Geophysical Union 75(3):24.

de Moustier, C. (1988). "State of the art in swath bathymetry survey systems." Int. Hydrog. Rev. 65(25).

Clay, C. S. and H. Medwin (1977). Acoustical Oceanography: principles and applications. New York, Wiley-Interscience.

Chen, C. T. and F. J. Millero (1977). "Sound Speed in Seawater at High Pressures." Jour. Acoust. Soc. Amer. 62(5): 1129-1135.

Chakraborty, B. (1995). "Studies on a 120^o Segmented circular array for multi-beam multi-frequency bathymetric application." J. Sound and Vibration 179(1): 1-12.

Caruthers, J. W. (1977). Fundamentals of marine Acoustics, Elsevier.
Asada, A. (1992). "Sea Beam 2000: Bathymetric Surveying with Interferometry." Sea Technology(June): 10-15.

E.2. Sonar References

Ali, H. B. (1993). "Oceanographic Variability in Shallow-Water Acoustis and the Dual Role of the Sea Bottom." J. IEEE Oceanic Eng. 18(1): 31-41.

Asada, A. (1992). "Sea Beam 2000: Bathymetric Surveying with Interferometry." Sea Technology(June): 10-15.

Babb, R. J. (1989). "Feasibility of interferometric swath bathymetry using GLORIA, a long-range sidescan." IEEE J. Oceanic Eng. 14: 299-305.

Barilko, S. I., V. N. Kuznetsov, et al. (1989). "A towed multi-beam sonar." Oceanology 29: 762-765.

Bergersen, D. D. (1991). A synopsis of SeaMARC II side-scan processing techniques. OCEANS '91, Honolulu, HI, IEEE.

Blackinton, J. G., D. M. Hussong, et al. (1983). First results from a combination side-scan sonar and seafloor mapping system (SeaMARC II). Offshore Technology Conference.

Blackinton, J. G. (1986). Bathymetric Mapping with SeaMARC II: An Elevation-Angle Measuring Side-Scan Sonar System. Ocean Engineering. Honolulu, University of Hawaii: 145.

Blackinton, J. G. (1991). Bathymetric Resolution, Precision and Accuracy Considerations for Swath Bathymetry Mapping Sonar Systems. IEEE Oceans '91, Honolulu Hawaii, IEEE.

Calder, B. R. and L. A. Mayer (2001). Robust Automatic Multi-beam Bathymetric Processing. U.S. Hydro '2001 Conference, The Hydrographic Society of the US.

Carara, W. G., R. J. Goodman, et al. (1995). Spotlight Synthetic Aperture Radar, Signal Processing Algorithms. Boston, MA, Artech House.

Caress, D. W. and D. N. Chayes (1995). New Software for Processing Sidescan Data from Sidescan-Capable Multibeam Sonars. IEEE Oceans '95, San Diego, CA., IEEE.

Caress, D. W. and D. N. Chayes (1996). "Improved Processing of Hydrosweep Multibeam Data on the R/V Maurice Ewing." Marine Geophysical Researches 18: 631-650.

Chayes, D. N. and D. W. Caress (1993). Processing and Display of Multibeam Echosounder Data on the R/V Maurice Ewing. Fall Meeting, San Francisco, CA, AGU.

Cole, F. W. (1968). A Familiarization with lateral or side-scanning sonars. Dobbs Ferry, NY, Hudson Labs.

Collot, J.-Y., J. Delteil, et al. (1995). "Sonic Imaging Reveals New Plate Boundary Structures Offshore New Zealand." EOS 76(11).

D. Alexandrou and C. deMoustier (1988). "Adaptive noise canceling applied to SeaBeam sidelobe interference rejection." J. Ocean. Eng. 13: 70-76.

Davis, T. M. (1974). Theory and Practice of Geophysical Survey Design. Department of Geosciences. Washington DC, Pennsylvania State University: 137.

Davis, E. E., R. G. Currie, et al. (1986). "The Use of Bathymetric and Image Mapping Tools for Marine Geosciences." Marine Technology Society Jour. 20(4): 17-27.

Appendix F. History of MB-System™

The development of MB-System began in 1990 as part of ongoing research at L-DEO involving swath bathymetry data collected with SeaBeam multibeam sonars. Development was accelerated in 1991 as part of the effort to support the STN-Atlas Hydrosweep DS multibeam sonar on L-DEO's ship, the R/V Maurice Ewing. The National Science Foundation provided support in 1993 and 1994 to improve and extend MB-System. The intent of this initial grant was to provide a standard generic set of tools for processing and display of swath sonar data that could be used by the U.S. academic community. The first generally released version of MB-System (3.0) was made available in the Spring of 1993. This was followed by versions 3.1 and 3.2 in July, 1993, version 3.3 in November, 1993, and version 3.4 in December 1993. All of these early releases supported only SeaBeam and Hydrosweep data.

SeaBeam Instruments and Antarctic Support Associates provided additional support in 1994 for the development of MB-System, with particular emphasis on capabilities related to the new SeaBeam 2100 series of sonars. A considerably enhanced MB-System version 4.0 was released on October 22, 1994; this release followed an almost complete rewrite of the underlying source code. The new capabilities included support for sidescan as well as bathymetry data and support for data from a number of very different sonars.

The National Science Foundation funded a five year effort begun in 1995 to maintain and further develop MB-System. From 1994 to 1997, SeaBeam Instruments (a major multibeam sonar manufacturer and, at the time, the principle employer of David W. Caress) provided significant support for MB-System development and maintenance. Similarly, the Newport, RI office of the Science Applications International Corporation (SAIC) supported some MB-System development during 1997-1998, when David W. Caress worked there. Version 4.1 was released in November, 1994, followed by 4.2 in February 1995, 4.3 on March 12, 1996, 4.4 on August 27, 1996, and 4.5 on September 23, 1997.

David W. Caress joined the Monterey Bay Aquarium Research Institute (MBARI) in September, 1998. Version 4.6 was released on April 16, 1999. The final update to version 4.6 (4.6.10) was announced on March 8, 2000. The primary innovations during this period included support for the new generation of Simrad multibeam sonars and tools for generating data products that could be imported directly into GIS software packages.

The National Science Foundation funded a second five year grant to MBARI and L-DEO which support the MB-System project from 2001-2005. The version 5.0 release, which was the first version to include this cookbook, incorporates yet another substantial rewrite of the underlying code as well as providing significant new capabilities.

MB-System 5.0.8 was released in February, 2006

Appendix G. Shipboard Multi-Beam Sonar Installations

Words about the various specific sonar installations.