# Space-Time Correlation Fields

## Methods for Observing Wave-Velocities from a Sparsely Sampled Data Set

A thesis submitted in partial fulfillment of the requirement
for the degree of Bachelor of Science with Honors in
Physics from the College of William and Mary in Virginia,

By

Stephen Simons

Accepted for_____
(Honors, High Honors, or Highest Honors)

_____
Director

_____

_____

_____

**Table of Contents** **Page**

**Acknowledgements:**

I would like to thank Andrew Norman for his computer expertise, his general knowledge of physics, and his many hours of help without which I could not have finished this project. I would also like to thank Professor Gene Tracy for his understanding of the subject matter and his willingness to work with me even when his schedule was incredibly busy.

**Abstract:**

The hypothesis presented by Kaufman *et al.* is that there is resonance between an equatorially trapped Yanai mode and a coastal Kelvin mode in the Gulf of Guinea. The TOPEX/POSEIDON merged geophysical data records are the data set proposed to be used to observe the eastern Atlantic region in consideration. Using satellite altimetry data to observe this propagating waves presents several interesting data analysis challenges. This paper deals with the issues of the maximum bandwidth, minimum signal to noise ratio, maximum sparsity, and smallest sample size allowed to created statistically significant evidence of signal propagation in the ocean. To create a controlled experiment, a model Yanai wave will be used as the wave field for this analysis.

## 1. Introduction:

The problem presented is one of observation. Can propagating signals be detected using sparsely sampled data? Can causality be maintained using auto-correlation functions of discrete sets in space and time? These questions are of particular relevance to the physical problem at hand. The hypothesis under consideration (Kaufman *et al.*) is that there is resonance between an equatorially trapped Yanai wave and a coastal Kelvin wave in the Gulf of Guinea (see figure 1.1).

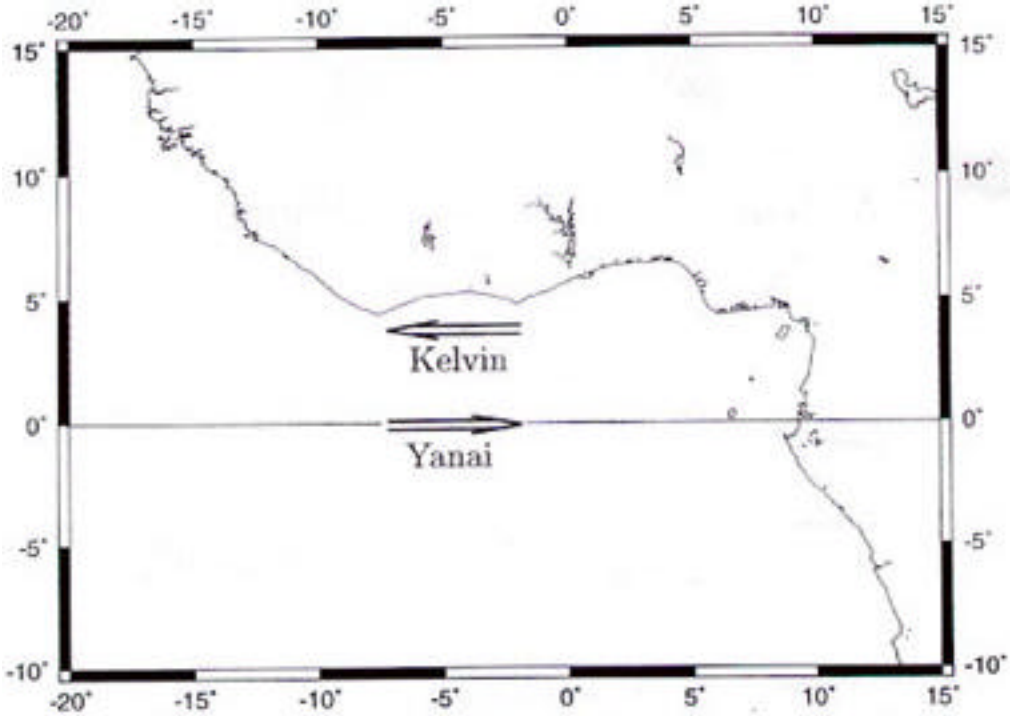


Figure 1.1: The Gulf of Guinea with an eastward propagating Yanai mode and a westward propagating Kelvin mode (Kaufman *et al.*).

The ocean's vast expanse prevents the present day observer from obtaining a complete data set, with high spacial and temporal resolution. Due to this limitation, various methods have been developed to track the circulation, temperature, wind fields, and many other useful observables for the study of the fluid dynamics involved with our earth's oceans. One of the most promising techniques for acquiring more complete and uniformly sampled data sets is satellite altimetry. Using radar, laser range finders, and the global positioning satellite network, radar altimeters can calculate the surface height

of the ocean to a precision of the order of one centimeter. This level of precision is actually not only due to the hardware, but is imposed by the sophisticated set of data flags and correction factors that are applied to the raw data in the calculation.

Of particular interest is the TOPEX/POSEIDON satellite, a joint effort of NASA, the French Space Agency, Jet Propulsion Laboratories (JPL), and NOAA. The TOPEX/POSEIDON satellite uses a dual band radar altimeter (Ku and C band radar, 13.6 GHz and 5.3 GHz respectively). The satellite has been in its observation phase since February of 1993. The satellite emits RF radiation toward the earth's surface. It then receives and processes the back-scattered radiation. The onboard computer then has "the height above the earth's surface (pulse transmit time), ocean significant wave height (via return pulse shape characteristics), and surface radar backscatter coefficient (via received energy)" as raw data for its calculations (Brooks *et al*). Its average altitude is 1339 km, which provides a sampled footprint with a radius of 11.0 km. The ground track velocity of the satellite is 5.8 km/s and the groundtrack pattern repeats within $\pm 1$ km every 9.92 days creating a grid of 254 groundtracks on the earth's surface.

The difficulty in using satellite data to study wave motion becomes evident in figure 1.2.

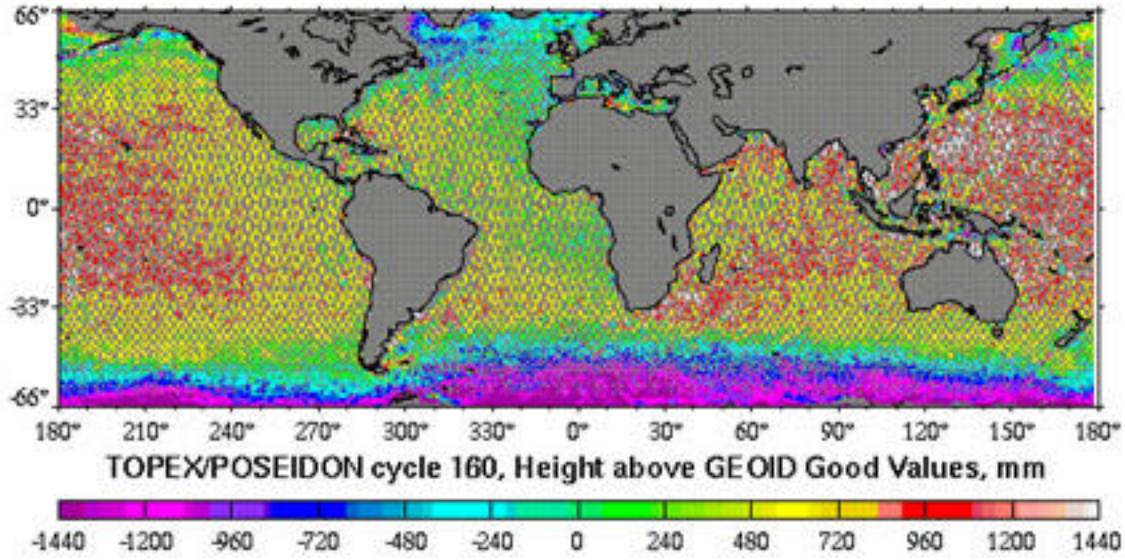


Figure 1.2: Graphic mapping of sea surface height data from the TOPEX/POSEIDON satellite

In figure 1.2 the diagonal colored lines are the satellite ground tracks. The gray spaces in between these lines reveal the incompleteness of the collected data. This spacing produces a less than ideal resolution in the data. For instance, the section of the Atlantic Ocean that is proposed to have resonance between coastal Kelvin modes and equatorially trapped Yanai modes in Kaufman *et al.,* is the Gulf of Guinea. Comparing figure 1.1 to figure 1.2, it becomes obvious that only a total of $\approx 20$ ground tracks lie in the area of interest. So for any given latitude, there will only be on the order of 20 data points. This, needless to say, does not even begin to approach the size of a data set needed to perform a Fourier transformation to confirm the dispersion relations of the two wave modes and calculate their phase speeds and time lags to prove the hypothesis of resonance between the two (Kaufman *et al*). Therefore, an alternative

method of data analysis must be used in order to obtain a meaningful result using satellite altimetry.

The proposed alternative technique for data analysis is to use the TOPEX/POSEIDON sea surface height data as a displacement field in space and time. Using statistical methods presented in the paper by Sciremamano, auto-correlation functions will be found independently in space and time to determine the separation in space and time between statistically independent data points. Once this decorrelation time and distance are found, correlation functions can be found for space and time lags together. It is the object of this study to use a model wave signal to test the limits of this analysis process. Through examination of the correlation functions produced, the ability to detect a wave propagating, the maximum bandwidth that is still detectable, and the minimum signal to noise ratio and data sample needed for detection will be estimated.

## 2. Background:

Kaufman *et al.* presents a hypothesis that there is resonance between the Kelvin mode propagating westward along the coast of the Gulf of Guinea and the incoming (eastward propagating) mixed-Rossby gravity mode (Yanai). The theory is developed using $\beta$-plane and $f$-plane models (see Appendix 1) for the Yanai mode and Kelvin mode respectively. These models are developed using the shallow water approximation. This approximation is valid because the wavelength of the waves involved and the length scale of the Atlantic Ocean are large with respect to the depth of the ocean. Table 2.1 provides a general idea of the depths of the oceans covering the Earth.

TABLE 2.1

| Ocean | Mean Depth (km) | Maximum Depth (km) |
|---|---|---|
| Pacific | 3.94 | 11.022 |
| Atlantic | 3.575 | 8.605 |
| Indian | 3.84 | 7.45 |
| Arctic | 1.117 | 4.6 |

The physics involved will be dealt with more thoroughly in a moment. Theoretical considerations suggest that the wavelength of the resonant modes is on the order of 100 kilometers, at least one order of magnitude greater than the maximum depth of the Atlantic Ocean.

The essential equations of any fluid motion are two conservation laws: conservation of mass and conservation of momentum. In a fluid continuum, conservation of mass is

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \qquad (2.1)$$

where $\rho$ is the density of the fluid and $\mathbf{u}$ is the velocity field. Conservation of momentum in a fluid is,

$$\rho\left[\frac{d\mathbf{u}}{dt} + 2\Omega \times \mathbf{u}\right] + \nabla p + \rho \nabla \Phi = 0\,, \qquad (2.2)$$

the sum of external forces with $\nabla p$ as the pressure gradient force, $\rho \nabla \Phi$ is the body force with $\Phi$ as the gravitational potential energy per unit mass, and $\Omega = 2\pi / 24 hours$ is the rotational frequency of the earth. It is of note that at these scales of motion, viscous

effects are negligible. Fluids are also subject to conservation of energy and the first and second laws of thermodynamics, but these will not play significant roles in this discussion.

With the conservation equations in place we are interested in developing the mathematics of the Kelvin and Yanai modes. There are two length scales that will determine whether or not shallow water theory is a valid approximation for these two modes: the Rossby radii of the two modes

$$R_e = (c/\beta)^{1/2} \tag{2.3}$$

(where the Coriolis parameter of the $\beta$-plane approximation is $f(y) \equiv \beta y$ and $c = (g'H)^{1/2}$ is the wave speed in terms of the reduced gravity $g' \equiv (\Delta\rho/\rho)g$ with $\Delta\rho$ equal to the change in density over the thermocline (Kaufman *et al.*)(see Appendix 1)) and

$$\delta = \frac{D}{L} \tag{2.4}$$

with D as the characteristic depth and L as the characteristic length scale.
As it turns out, the Rossby radius of the Kelvin mode is $\approx 70km$ and of the Yanai mode is $\approx 190km$. Both of these are at least one order of magnitude larger than the mean depth of the Atlantic Ocean (3.575km). This fulfils the first condition for the shallow water approximation to be valid; the second is that $\delta = D/L << 1$. With a depth scale of 3.575 kilometers and a length scale on the order of 1000 kilometers, the Atlantic Ocean obviously fulfills $\delta = D/L << 1$.
Seeing that these two conditions are met, the shallow water approximation can be used to model the Kelvin and Yanai modes in the Gulf of Guinea. For the purpose of this discussion, six assumptions will be made to simplify the shallow water equations:

- The fluid is incompressible: $\nabla \cdot \mathbf{u} = 0$.
- The fluid has constant and uniform density. This eliminates internal waves and restricts the theory to a description of thermocline dynamics. Although the equations presented are derived for a simple shallow water system, they can be used to describe the dynamics of the thermocline under the rigid lid approximation.
- The fluid is shallow. Described above, this is the essential condition on which this theory is based.
- $W \le O(\delta U)$. The vertical scale of the velocity field, W, is much smaller than the horizontal scale of the velocity field, U, by a factor on the order of $\delta << 1$.
- For this solution, there is no background fluid flow. In other words, the perturbation will be about a rest state.
- A columnar model of the ocean is assumed where the pressure, p, at any point in the field is

  $$p = \rho g(h - z) + p_0 \tag{2.6}$$

  where $p_0$ is the pressure at the surface atmospheric pressure, and h is the height of the surface from some reference level below the seafloor. Note that this equation for pressure establishes the independence of the vertical and horizontal pressure gradient:

  $$\frac{\partial p}{\partial x} = \rho g \frac{\partial h}{\partial x} \tag{2.7a}$$

  $$\frac{\partial p}{\partial y} = \rho g \frac{\partial h}{\partial y} \tag{2.7b}$$

This also establishes that the horizontal accelerations must be independent of z. This means that it is consistent to say that the horizontal velocities remain z-independent if they initially are so.

With these six assumptions in place, we are interested in deriving the dispersion relations of the Kelvin and Yanai modes. To do this we will manipulate the momentum equation (2.2). The linearized forms of the momentum equations for shallow water theory with no background flow disregard any quadratic terms in $u, v, \eta$ (see Appendix 1 for a more complete derivation of shallow water equations).

$$\frac{\partial u}{\partial t} - fv = -g\frac{\partial \eta}{\partial x} \tag{2.8a}$$

$$\frac{\partial v}{\partial t} + fu = -g\frac{\partial \eta}{\partial y} \tag{2.8b}$$

$$\frac{\partial \eta}{\partial t} + \frac{\partial}{\partial x}(uH_0) + \frac{\partial}{\partial y}(vH_0) = 0 \tag{2.8c}$$

where $H_0$ is the constant depth about which the perturbation is created. These three equations can then be manipulated to obtain an equation in one variable

$$\frac{\partial}{\partial t}\left[\left(\frac{\partial^2}{\partial t^2} + f^2\right)\eta - \nabla\cdot\left(C_0^2\nabla\eta\right)\right] - gfJ(H_0,\eta) = 0 \tag{2.9}$$

where J is the Jacobian of two functions

$$J(A,B) \equiv \frac{\partial A}{\partial x}\frac{\partial B}{\partial y} - \frac{\partial A}{\partial y}\frac{\partial B}{\partial x} \tag{2.10}$$

and the squared phase velocity of the wave is

$$C_0^2 = gH_0. \tag{2.11}$$

The eigenvalue relation that arises from imposing the boundary condition (in y) of an infinite (in x) channel of width L is

$$\left(\omega^2 - f^2\right)\left(\omega^2 - C_0^2k^2\right)\sin\left[\left(\frac{\omega^2 - f^2}{C_0^2} - k^2\right)L\right] = 0 \tag{2.12}$$

Taking the second factor of this relation and assuming a local boundary in the horizontal plane of motion gives us the Kelvin mode with the dispersion relation

$$k = -\frac{\omega}{C_0} \tag{2.13}$$

This result will be seen to be particularly interesting because the dispersion relation for the Yanai mode (Kaufman *et al.*) is

$$k = \frac{\omega}{C} - \frac{\beta}{\omega} \tag{2.14}$$

where $\beta$ is the comes from the Coriolis parameter in the $\beta$-plane approximation

$$f(y) = \beta y \tag{2.15}$$

From these two dispersion relations comes the claim that resonance between Kelvin and Yanai modes is possible. The two dispersion curves intersect at a well-defined frequency and wavenumber; there is resonance (see figure 2.1) (Kaufman *et al.*).

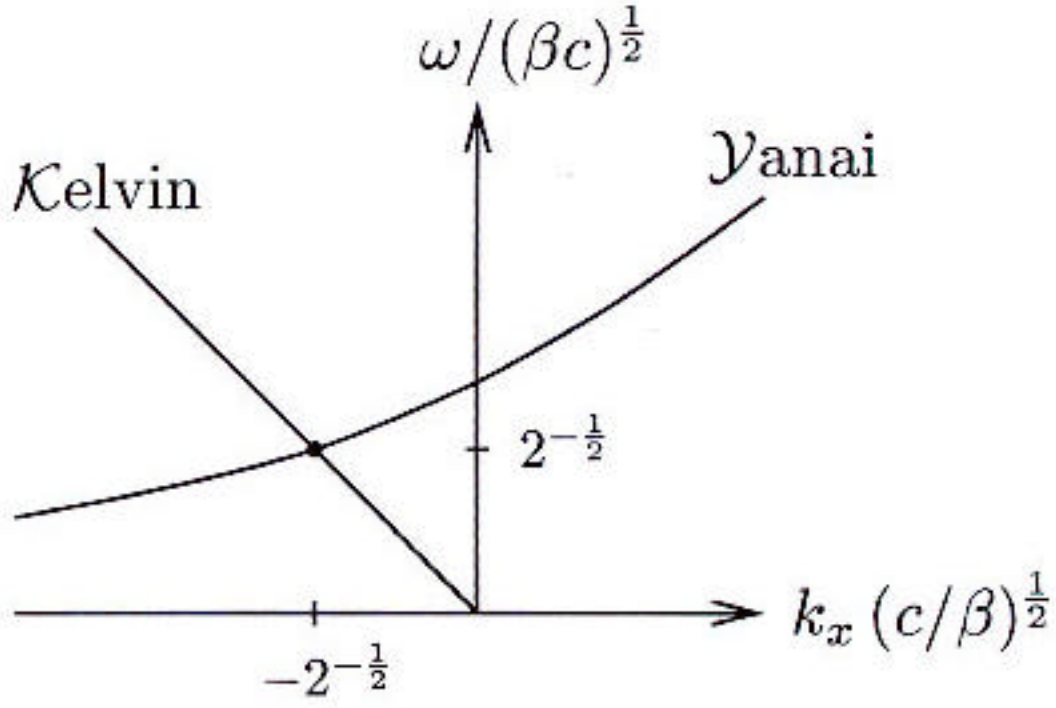


Figure 2.1: The intersection of the Kelvin and Yanai dispersion relations in dimensionless form. The curves cross at $\omega_R/(\beta c)^{\frac{1}{2}} = 2^{-\frac{1}{2}}, k_R(c/\beta)^{\frac{1}{2}} = -2^{-\frac{1}{2}}$ (Kaufman *et al.*).

Looking back at figure 1.1 it is seen that the two modes are travelling in opposite directions. This is because the group velocity of the two modes propagate in opposite directions. Returning to figure 2.1, it may be seen that Kelvin and Yanai dispersion relations have opposite slope (group velocity) but intersect at a well defined wave number and frequency with phase speed in the same direction.

Finally there are a couple of numbers that will be useful in later discussions:

The Rossby Deformation Radius is

$$R = \frac{C_0}{2\Omega} \tag{2.16}$$

and is the distance over which the gravitational tendency to flatten the fluid surface is balanced by the Coriolis acceleration to deform the surface (Pedlosky).

- The Rossby number is a dimensionless number which is the ratio of inertial force to geostrophic force and is used to determine if a motion is large scale

$$\varepsilon = \frac{U}{2\Omega L} \tag{2.17}$$

with U equal to the horizontal velocity scale and L equal to the horizontal length scale.

**3. The Model:**

For the purposes of this discussion, we are only going to develop a model Yanai wave, not a Kelvin wave. This is because the Yanai mode is dispersive and therefore harder to detect.

The equation used to build the model wave field is

$$\Psi = \sum_j a_j \cos(k_j x - \omega_j t + \phi_j). \tag{3.1}$$

The amplitude of the j-th mode is

$$a_j = e^{-(j-j_d)^2/s^2} \tag{3.2}$$

and varies between zero and one with $j_d$ as the index of the dominant mode (amplitude one). The wavenumber of the j-th mode is

$$k_j \equiv \frac{-2\pi j}{L} \tag{3.3}$$

with L as the characteristic length scale, the Rossby radius of the resonant mode, 365km. The frequency of the Yanai mode of the j-th term is derived from the dispersion relation to be

$$\omega_j = \frac{1}{2} k_j C_0 + \frac{1}{2}\sqrt{k_j^2 C_0^2 + 4\beta C_0}\,. \tag{3.4}$$

$\phi_j$ is a random phase between 0 and $2\pi$. It is of note that (3.1) will be expanded in odd values of j about $j_d$.

The model Yanai wave field is constructed using code developed in C++ (see appendix 2) to create two dimensional (space and time), double-precision arrays. The arrays are created by sampling the wave field at

$$x_i = iL/100.0 \tag{3.5}$$

$$t_k = \frac{2\pi k}{100.0\omega_{j_d}} \tag{3.6}$$

This process creates an array with indices i and k which are related to the position and time by the above scale factors. Figure 3.1 is an example of a model Yanai mode.

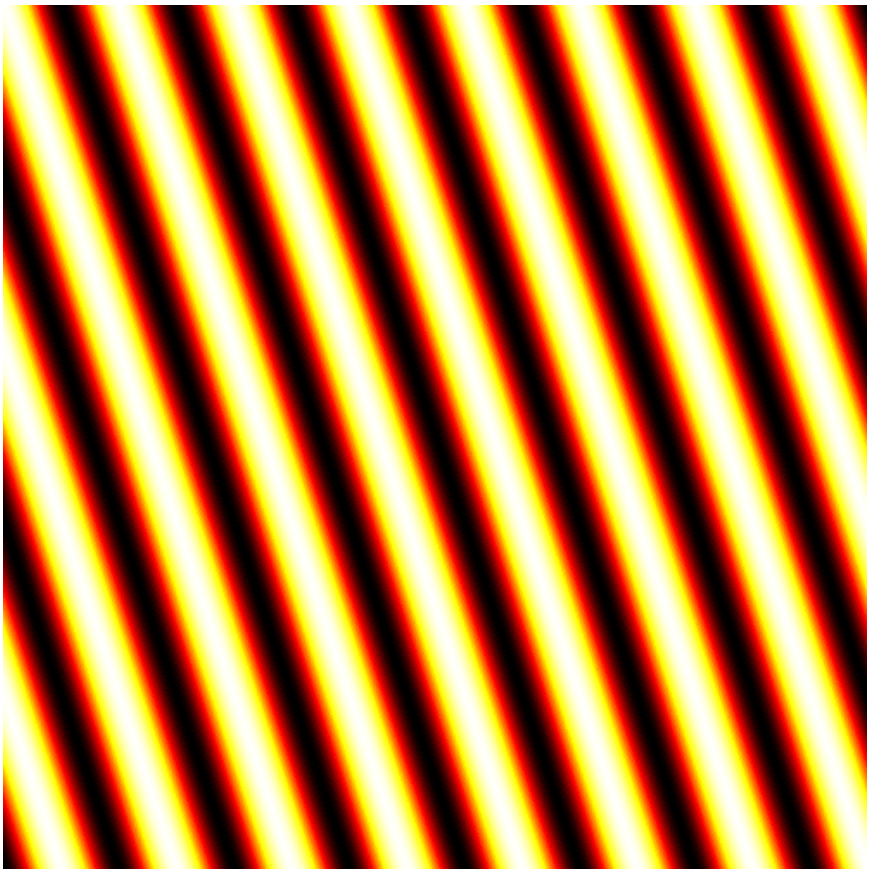

Figure 3.1: A model Yanai wave field in the second quadrant (negative space, positive time)

The end in mind is to create a model Yanai wave signal that can be manipulated to test the limits of the space-time correlation method of data analysis. This model allows us to test the four essential limits on this data analysis technique: the size of the data sample, the maximum allowed bandwidth, the minimum signal to noise ratio, and the sparsity in space and time of the data samples allowed to detect a propagating signal.

**4. Limits:**

The ability to detect a wave will be tested by using a single j-term expansion of the wave at j=15 (an arbitrary value). The test for the size of the data set will be discussed further in the section on statistical methods. However, it may be seen that the size of the array [i,k] may be varied. The maximum allowed bandwidth will be tested by varying the $s^2$ term in the power spectrum and then including all significant j-terms in the expansion of (3.1). Finally, the minimum signal to noise ratio will be tested by creating gaussian noise with a variable RMS (see appendix 3) and then calculating the RMS of the wave field. This done, the two may be superimposed and used as a new wave field to be tested using the data analysis techniques described in the next section.

**5. Statistics:**

The question at hand is one of interpretation and analysis by statistical methods. The technique used to discern waves propagating in a data field is space-time correlation. The basic idea being that if a signal is propagating at some velocity

$$v = \frac{\Delta x}{\Delta t} \tag{5.1}$$

significant correlations should be found between data sets sampled from points in space and time that are separated by space and time lags whose ratio returns the propagation velocity plus or minus some bin width. In other words, space-time correlations will be shown to reveal a signal propagating at velocity v within some confidence level (bin width).

The correlation coefficient relating two data samples x and y

$$\begin{aligned} x &= \{x_i\} = (x_1, x_2, x_3, \ldots x_n) \\ y &= \{y_i\} = (y_1, y_2, y_3, \ldots y_n) \end{aligned} \tag{5.2}$$

is defined as

$$R = \frac{n\left(\sum_{i=0}^{n-1} x_i y_i\right) - \left(\sum_{i=0}^{n-1} x_i\right)\left(\sum_{i=0}^{n-1} y_i\right)}{\sqrt{n\left(\sum_{i=0}^{n-1} x_i^{\,2}\right) - \left(\sum_{i=0}^{n-1} x_i\right)^2}\sqrt{n\left(\sum_{i=0}^{n-1} y_i^{\,2}\right) - \left(\sum_{i=0}^{n-1} y_i\right)^2}} \tag{5.3}$$

where n is the number of statistically independent points in each set. The first extension from this is to create an auto-correlation function. Assume that $x = x(t)$ then the autocorrelation function of that data set in time would be defined as

$$R(\Delta t) = \frac{n\left(\sum x(t)x(t+\Delta t)\right) - \left(\sum x(t)\right)\left(\sum x(t+\Delta t)\right)}{\sqrt{n\left(\sum x(t)^2\right) - \left(\sum x(t)\right)^2}\sqrt{n\left(\sum x(t+\Delta t)^2\right) - \left(\sum x(t+\Delta t)\right)^2}} \tag{5.4}$$

This autocorrelation function is a function of the time lag between the first sample from the data set x and the second sample. Just as in the calculation of the correlation coefficient R, the size of the two samples must be the same (the summations must have the same limits). The autocorrelation function can also be calculated as a function of space-lag. This function is interesting because it allows the observer to determine what statistically independent samples are in space and time. Two statistically independent points in space-time are separated by at least the significant decorrelation time and space. Although autocorrelation functions in space or time independently will not be used in the analysis of the wave model, they are essential in analyzing data sets from satellite altimetry (Sciremamano).

From this point the derivation of a space-time correlation function is somewhat elementary. Instead of only having x as a function of t, now $x = x(z,t)$ where z is a spatial coordinate. Then the space-time correlation function is defined as

$$R(\Delta z, \Delta t) = \frac{n\left(\sum x(z,t)x(z+\Delta z, t+\Delta t)\right) - \left(\sum x(z,t)\right)\left(\sum x(z+\Delta z, t+\Delta t)\right)}{\sqrt{n\left(\sum x(z,t)^2\right) - \left(\sum x(z,t)\right)^2}\sqrt{n\left(\sum x(z+\Delta z, t+\Delta t)^2\right) - \left(\sum x(z+\Delta z, t+\Delta t)\right)^2}} \tag{5.5}$$

and is a function of the space-lag and time-lag separating the two samples of the data set x. For the purpose of detecting a propagating signal, one would expect to see a peak in the correlation function when

$$\frac{\Delta z}{\Delta t} \approx C_0 \tag{5.6}$$

Once again turning to the physical problem at hand, x is actually a discrete data set, not a continuous function. With this in mind, $x(z,t) \to x_{l,m}$ where l is the space index and m is the time index used in (3.6) and (3.7) to sample the model wave. With the data set now in matrix form, $R(\Delta z, \Delta t) \to R_{i,k}$ and is defined as

$$R_{i,k} = \frac{n\left(\sum_{l,m} x_{l,m} x_{l+i,m+k}\right) - \left(\sum_{l,m} x_{l,m}\right)\left(\sum_{l,m} x_{l+i,m+k}\right)}{\sqrt{n\left(\sum_{l,m} x_{l,m}{}^2\right) - \left(\sum_{l,m} x_{l,m}\right)^2}\sqrt{n\left(\sum_{l,m} x_{l+i,m+k}{}^2\right) - \left(\sum_{l,m} x_{l+i,m+k}\right)^2}} \tag{5.7}$$

where i and k are offsets to the space and time indices respectively. This correlation function creates a two-dimensional array of correlation coefficients with the offsets as indices. This allows the observer to see patterns that will be directly related to space and time lag and therefore be able to detect a propagating signal with velocity

$$C_0 - \varepsilon \le \frac{\Delta z}{\Delta t} \le C_0 + \varepsilon \tag{5.8}$$

where $2\varepsilon$ is a bin width of wave speeds.

Having established the correlation functions and a general concept of the method of detection of signals, we turn to a discussion of the methods used in discovering a significant correlation indicative of a propagating signal. For the purposes of this paper, visual analysis of correlation functions will be sufficient to recognize evidence of propagating signals. However, before actual data handling can take place, further exploration of numerical methods to derive wave speed is needed.

## 6. Results:

Visual analysis of the wave and correlation fields yields conceptual results, although it will take a great deal more analysis to achieve numerical results. There are six essential results that will be estimated in this section, the effects of: bandwidth, noise, bandwidth and noise, sparse sampling, sparse sampling of a broadband, noisy field, and coarse sampling on the ability of an observer to detect a propagating signal.

Broadening the bandwidth did not appear to have a significant effect on the ability to observe a propagating signal (see Appendix 2 for C++ code used to construct broadband wave fields). Figures 6.1 and 6.2 illustrate the effects of broadening the bandwidth. Although the figures are not showing the widest bandwidth, the beat frequency of the 25 modes can be seen to have positive slope where the wave itself has negative slope. This difference in sign is to be expected from figure 2.1 where it is seen that the phase speed is negative, but the slope of the dispersion relation is positive. This implies that the beat frequency seen in 6.1 and 6.2 is somehow linked to the phase velocity of the wave detected.

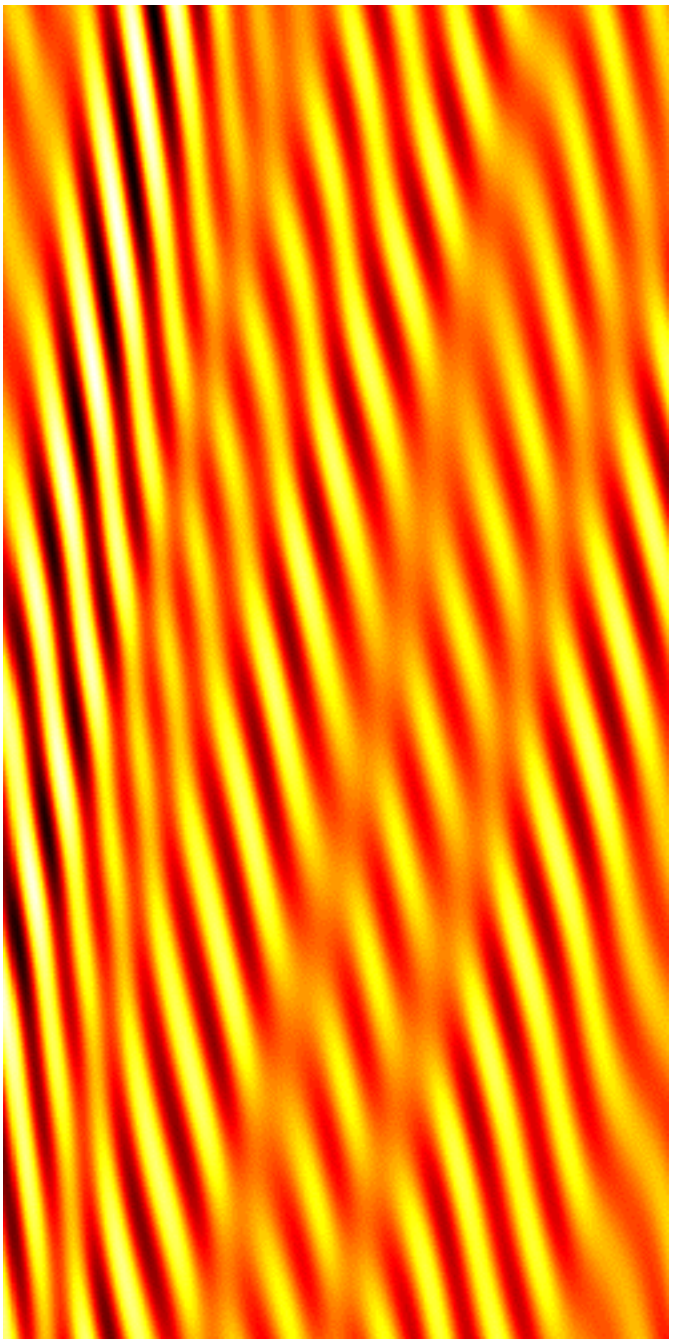

Figure 6.1: Yanai wave field expanded in 25 terms about j=25 with $s^2 = 100.0$ in the power spectrum

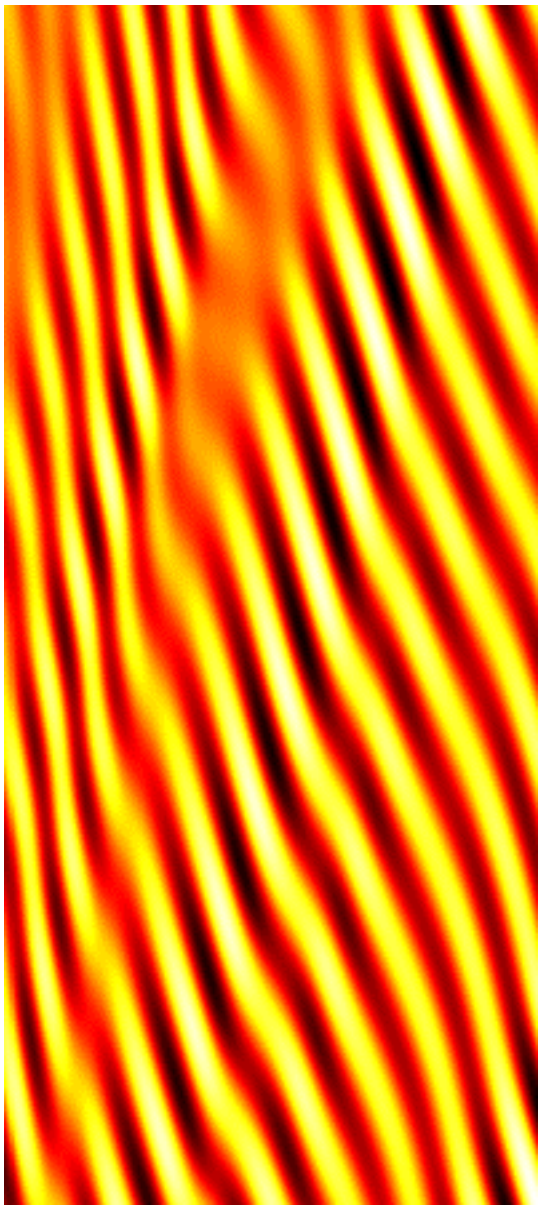

Figure 6.2: Correlation field of a Yanai wave expanded in 25 terms about j=25 with $s^2 = 100.0$ in the power spectrum

The second limiting factor to be added is noise. For the purposes of this model, Gaussian noise was generated and superposed on the wave field (see Appendix 3). To maintain a controlled experiment, the wave used is a single mode Yanai wave expanded about j=15. The RMS of the wave field used (figure 6.3) is 0.707. The maximum signal to noise ratio of a wave still visibly detectable in the correlation function was found to be 0.035 where the noise has an RMS value of 20.0. This result can be seen in figures 6.4 and 6.5. With the RMS of the noise at 20.0, the wave is essentially indistinguishable in figure 6.4. However, the signal reappears in the correlation function in figure 6.5. The conclusion to be drawn from this is that using correlation fields to detect waves allows for a significant amount of noise in the signal.

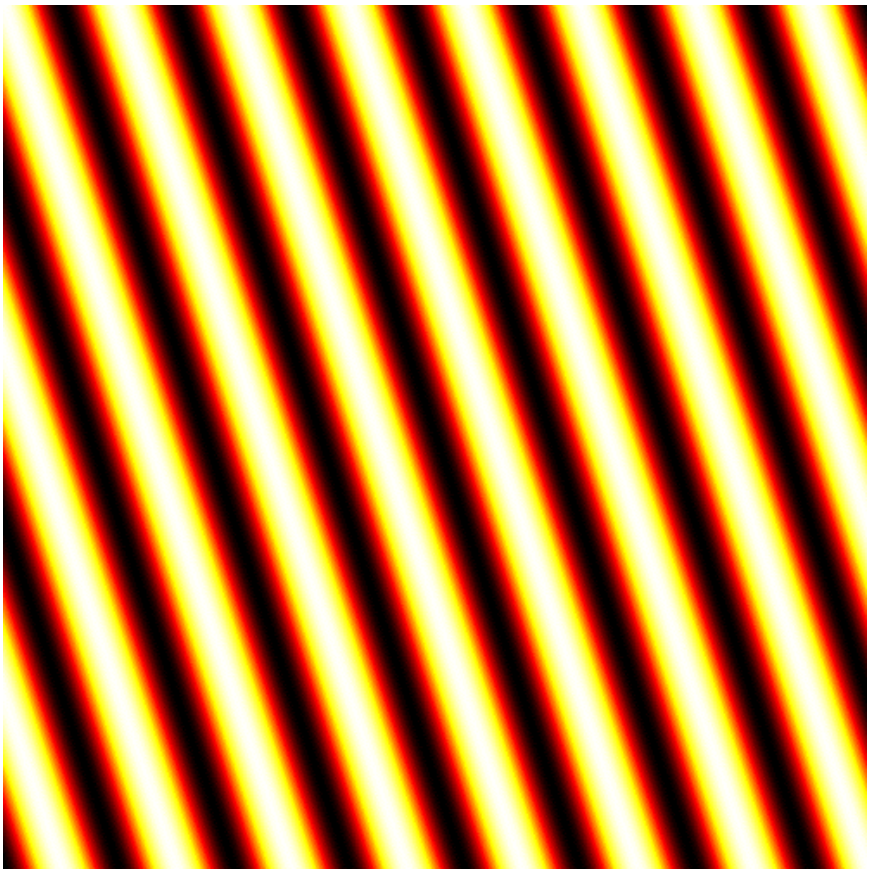

Figure 6.3:  A single mode Yanai wave at j=15

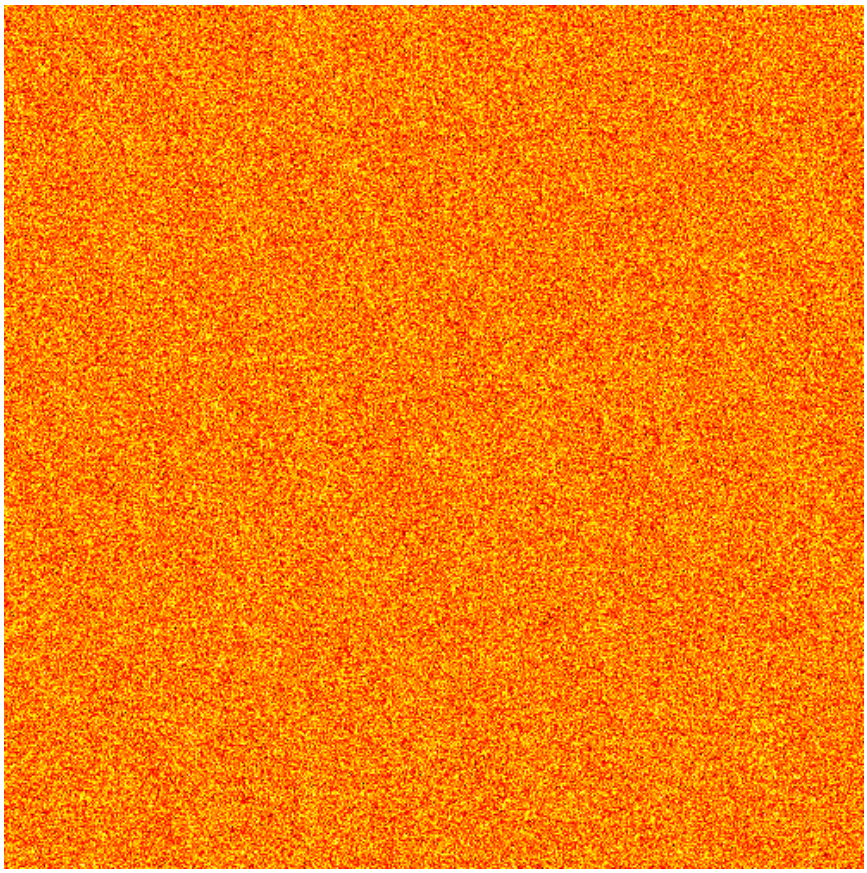

Figure 6.4: A single mode Yanai wave with RMS 0.707 at j=15 with Gaussian noise of RMS=20.0 superposed (signal to noise ratio is 0.035).

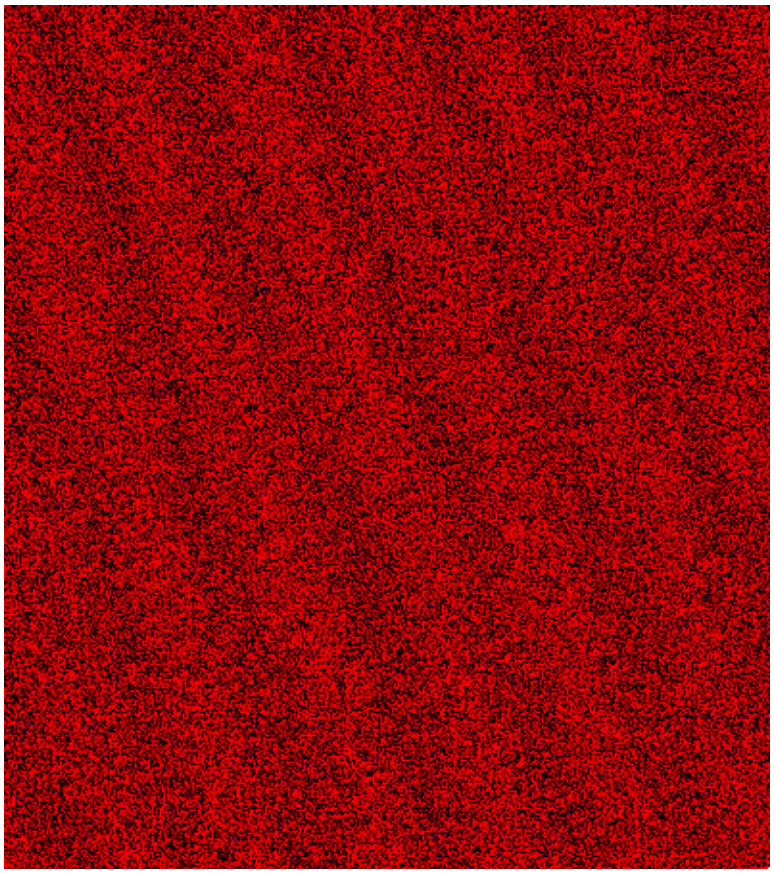

Figure 6.5: Correlation field of a single mode Yanai wave with RMS 0.707 at j=15 with Gaussian noise of RMS=20.0 superposed (signal to noise ratio is 0.035).

The obvious next step is to take the previous two alteration to the data and combine them to test the limits of observation in a noisy broadband wave field. For this result a 100 mode Yanai wave expanded about j=100 with $s^2 = 1000.0$ in the power spectrum (figure 6.6) will be used with Gaussian noise of RMS equal to 10.0 (figure 6.7). Although the signal to noise ratio is higher, the ability to detect the wave is more challenging. This is because the wave pattern to be found in the noise is more complex. Once again, though, the correlation field (figure 6.8) brings out the signal for visual analysis.

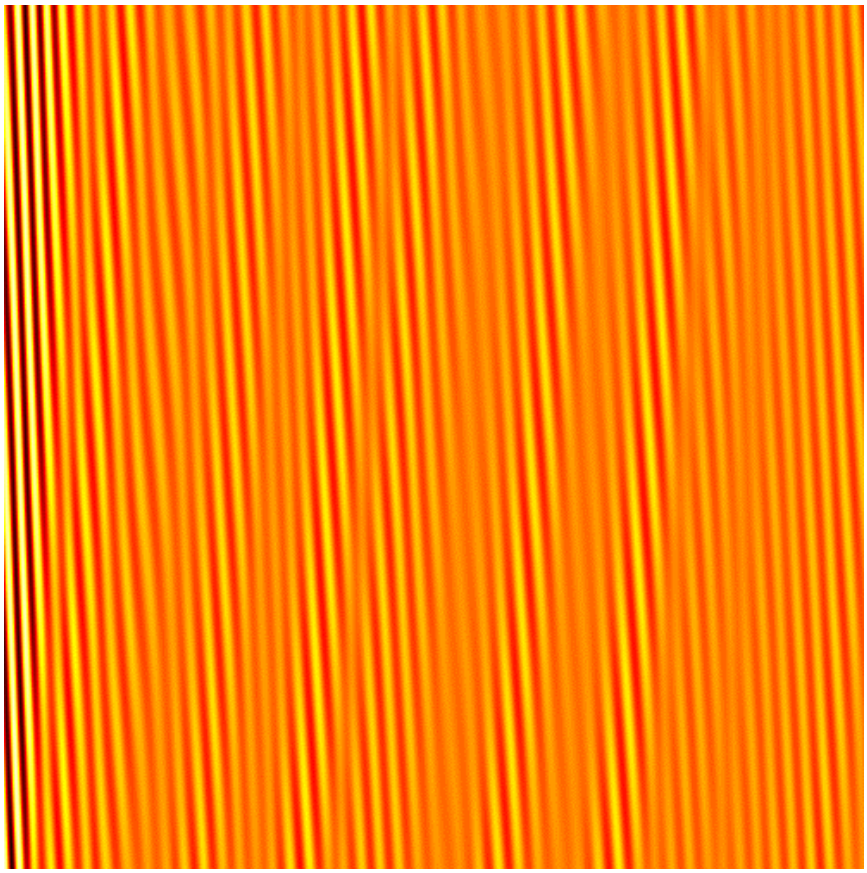

Figure 6.6: A 100 mode Yanai wave expanded about j=100 with $s^2 = 1000.0$ in the power spectrum.

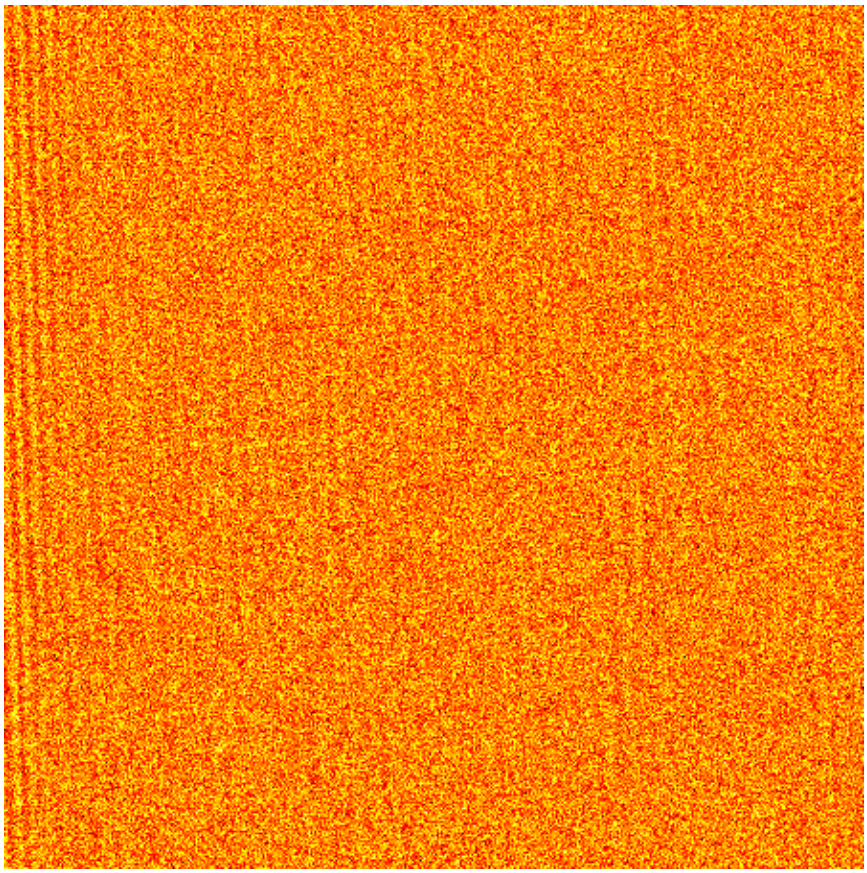

Figure 6.7: A 100 mode Yanai wave expanded about j=100 with $s^2 = 1000.0$ in the power spectrum superposed on Gaussian noise of RMS = 10.0.

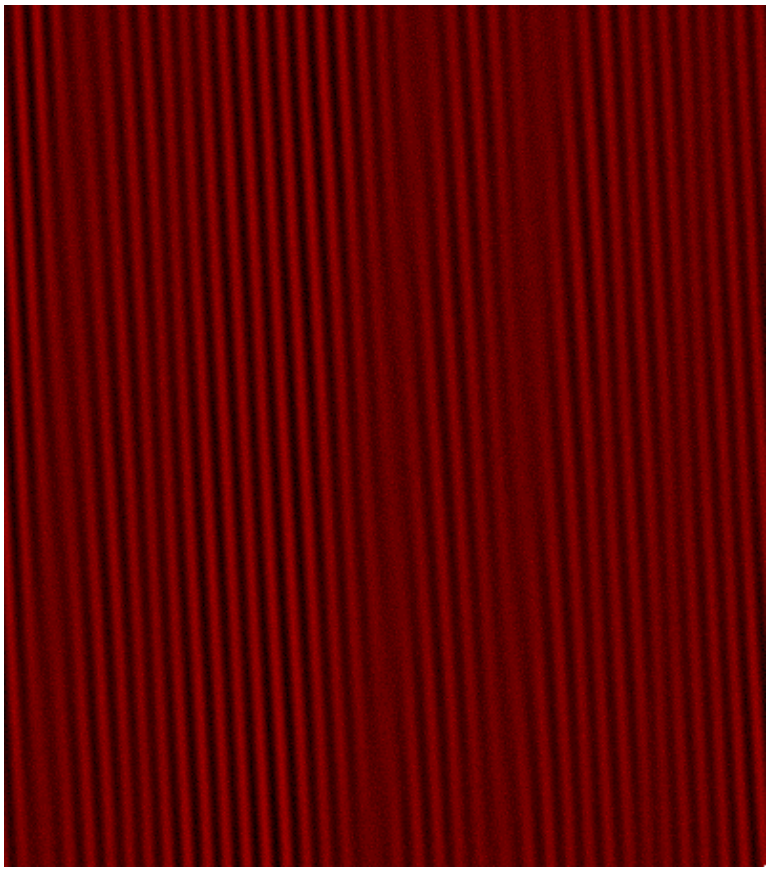

Figure 6.8: Correlation field of a 100 mode Yanai wave expanded about j=100 with $s^2 = 1000.0$ in the power spectrum superposed on Gaussian noise with an RMS=10.0.

Having estimated limits on the bandwidth and signal to noise ratio, sparse data sampling is the next result to be considered. For the purposes of this paper, sparse data sampling means that a wavefield is created and then in calculating the correlation function, only every n-th x value and m-th t value are used (with n and m being integers)(see Appendix 7). This means that with n and m equal to one a correlation field similar to those previously calculated will be constructed. The outside limit for detecting a propagating signal is found when n=10 and m=10 (figure 6.9). The interesting question in this limit is how to establish a causal link between one time series in space and the next time series in space. The answer comes from the hypothesis being tested. We are only interested in finding a wave of a given velocity (in this case -0.85 m/s). Therefore, we simply need to establish how far the wave travels in each time increment and then it becomes obvious which correlations are linked by this signal and which are not (the green line in figure 6.9). The more challenging question arises when there is discrete sampling in both time and space. This is the ultimate question to be answered in this research before satellite data will be useful in discovering a propagating signal.

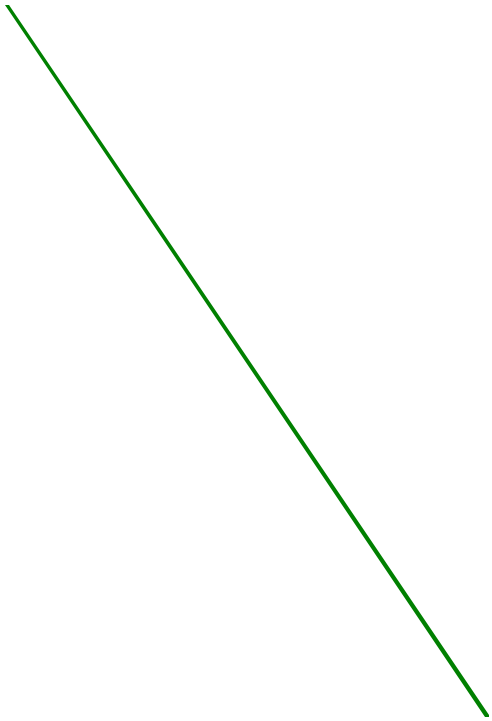

Figure 6.9: Correlation function of single mode Yanai wave at j=15 sampled on a grid of n=10 and m=10. The green line represents the wave velocity 0.85 m/s.

As previously, we are interested in combining the effects of sparse sampling with noise and bandwidth. A 25 mode Yanai wave expanded around j=25, with $s^2 = 100.0$ in the power spectrum, is superposed on Gaussian noise with a signal-to-noise ratio of 0.2877 and will be sampled on a grid of n=5 and m=5. The correlation function seen in Figure 5.10 is the result. As expected, the dominant wave velocity is harder to see in this result because of the combined effects of bandwidth, noise, and sparse sampling. However, in comparing the sparsely sampled correlation function (left) with the regularly sampled correlation function (right), the wave velocity becomes apparent. In the case of actual observation, a complete correlation function like the one seen on the right in figure 5.10 is not available. However, having done this preliminary analysis on the model wave, interpretation of the sparsely sampled wave field is now possible.

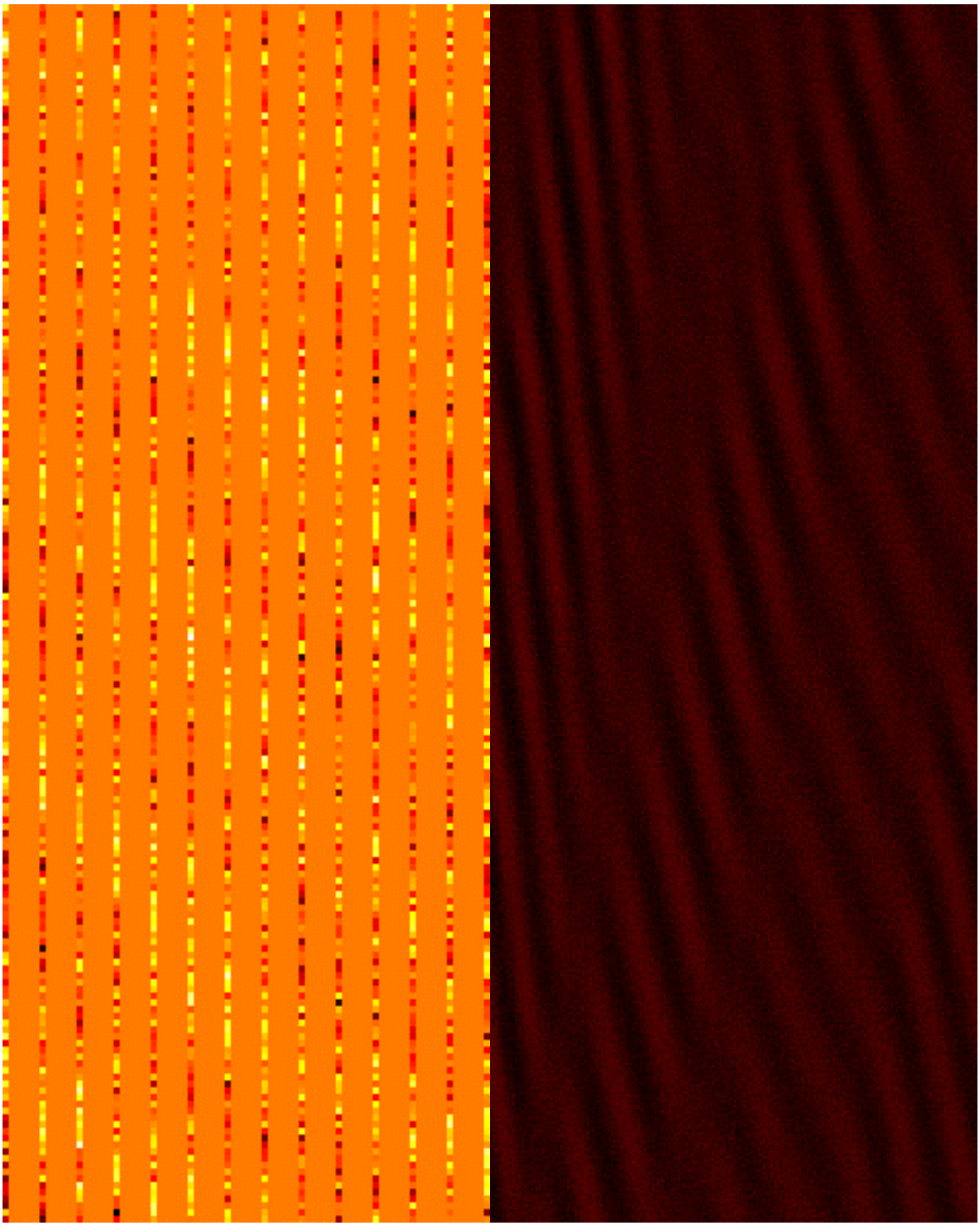

Figure 6.10: The correlation function of a 25 mode Yanai wave expanded around j=25 with $s^2 = 100.0$ in the power spectrum superposed on Gaussian noise with a signal-to-noise ratio of 0.287717 sampled on a grid of n=5 and m=5 (left) as compared to the same correlation function sampled on a grid of n=1 and m=1 (right).

The final result to be discussed is the effect of coarse sampling (see Appendix 6). The TOPEX/POSEIDON satellite samples the ocean on a discrete time and space grid.

The hypothesis presented in Kaufman *et al.* is that there is resonance in the Gulf of Guinea. Geographically, the Gulf of Guinea lies between 339.4515 and 12.0499 degrees longitude. This defines the area of interest for this discussion. Over the course of 29.7468 days, the TOPEX/POSEIDON satellite completes three cycles, recording 72 points in space-time along the equator in the area of interest (see figure 5.11).

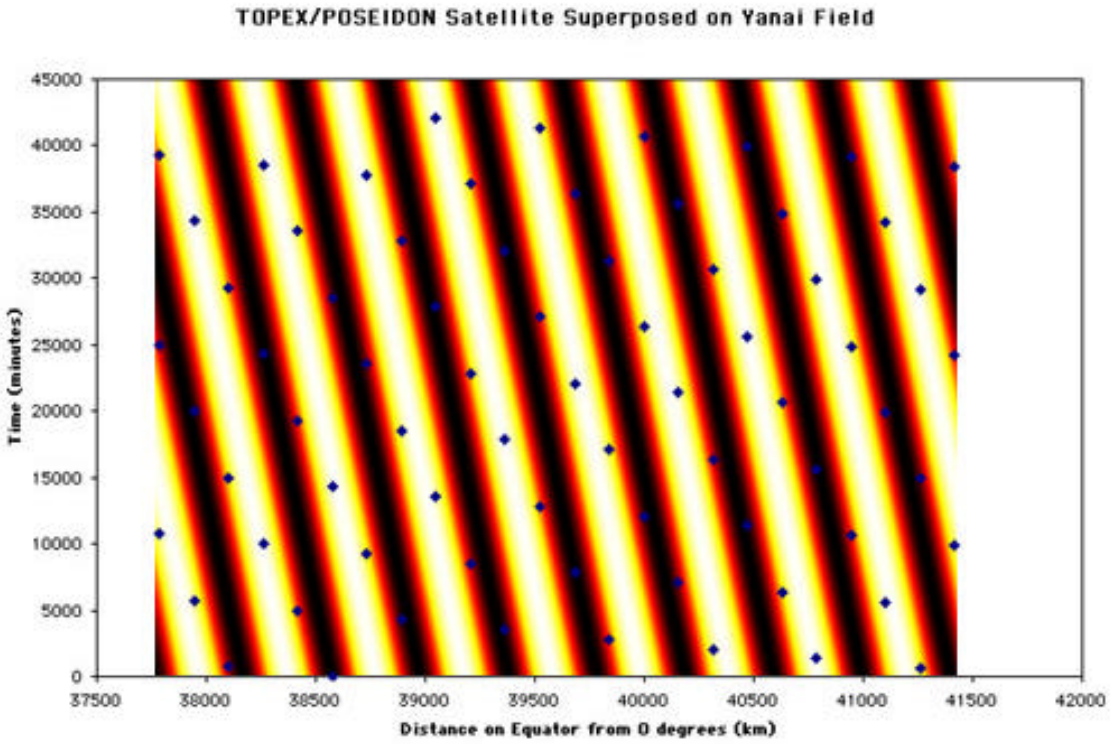


Figure 6.11: Superposition of TOPEX/POSEIDON satellite sampling in the geographical region of interest on a single mode Yanai wave at j=15.

The blue points seen in the figure represent a total distance of 3628.7974 kilometers observed over a total of 40642.96 minutes. This means that the satellite (as seen in the figure) has the opportunity to observe 9.9419 modes in space and 1.2271 modes in time because the resonant length scale of the Yanai wave is 365 kilometers and the resonant period is 33120 minutes. With the analysis of the model wave in mind, it may be seen that the regular grid of space and time lags of the TOPEX/POSEIDON data, will facilitate observation of a propagating signal in the Gulf of Guinea. This conclusion is based on the fact that if all the space-time lags between sets of two points are considered there are on the order of 40 pairs of data points collected by TOPEX/POSEIDON along the equator that are separated in space-time by 0.85 m/s $\pm 0.1 m/s$.

## 7. Conclusion:

Space-time correlation functions are an essential aspect of data analysis when the samples are not continuous in space and time. Even from this simple visual analysis of the model Yanai wave it may be seen that a broadband, low signal-to-noise ratio, sparsely sampled data set can be used to observe a propagating signal. The ability to sift through the raw data and retrieve a significant result lies in the statistical methods described in

*Sciremammano* and this paper. Further research needs to be pursued to develop numerical results. However, the evidence of the power of these statistical methods is apparent even in a cursory visual analysis of the presented results.

Appendix 1 – Derivation of the eigenvalue relation:

Having established that the horizontal velocity field is independent of z, the momentum equation (2.2) broken down into components becomes

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial v}{\partial y} - fv = -g\frac{\partial h}{\partial x} \tag{A.1a}$$

$$\frac{\partial v}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial v}{\partial y} + fu = -g\frac{\partial h}{\partial y} \tag{A.1b}$$

where $f = 2\Omega$. Still utilizing the z-independence of u and v, (2.6) can be integrated and solved with a rigid bottom boundary so that the equation for mass conservation in this approximation becomes

$$\frac{\partial H}{\partial t} + \frac{\partial}{\partial x}(uH) + \frac{\partial}{\partial y}(vH) = 0 \tag{A.2}$$

where $H = h - h_B$ with $h_B$ as the height of the rigid bottom from some reference depth. Now let the thickness of the fluid layer in the absence of motion be $H_0(x,y)$. Then with motion included as a small perturbation about this thickness

$$H(x,y,t) = H_0(x,y) + \boldsymbol{h}(x,y,t) \tag{A.3}$$

is the thickness of the fluid layer as it evolves in time.

The linearized forms of (A.1) and (A.2) disregard any quadratic terms in $u, v, \boldsymbol{h}$ and become

$$\frac{\partial u}{\partial t} - fv = -g\frac{\partial \boldsymbol{h}}{\partial x} \tag{A.4a}$$

$$\frac{\partial v}{\partial t} + fu = -g\frac{\partial \boldsymbol{h}}{\partial y} \tag{A.4b}$$

$$\frac{\partial \boldsymbol{h}}{\partial t} + \frac{\partial}{\partial x}(uH_0) + \frac{\partial}{\partial y}(vH_0) = 0 \tag{A.4c}$$

which can be manipulated to obtain an equation in one variable

$$\frac{\partial}{\partial t}\left[\left(\frac{\partial^2}{\partial t^2} + f^2\right)\boldsymbol{h} - \nabla\cdot\left(C_0^2\nabla\boldsymbol{h}\right)\right] - gfJ(H_0,\boldsymbol{h}) = 0 \tag{A.5}$$

where J is the Jacobian of two functions

$$J(A,B) \equiv \frac{\partial A}{\partial x}\frac{\partial B}{\partial y} - \frac{\partial A}{\partial y}\frac{\partial B}{\partial x} \tag{A.6}$$

and $C_0^2 = gH_0$. This equation can then be used to derive two differential equations to solve for the velocity field

$$\left(\frac{\partial^2}{\partial t^2} + f^2\right)u = -g\left(\frac{\partial^2 \boldsymbol{h}}{\partial x \partial t} + f\frac{\partial \boldsymbol{h}}{\partial y}\right) \tag{A.7a}$$

$$\left(\frac{\partial^2}{\partial t^2} + f^2\right)v = -g\left(\frac{\partial^2 \boldsymbol{h}}{\partial y \partial t} - f\frac{\partial \boldsymbol{h}}{\partial x}\right) \tag{A.7b}$$

We now explore the more particular case of wave motion in a bounded channel to derive the Kelvin mode.
Imposing this boundary condition requires that the velocity in the y-direction disappear at

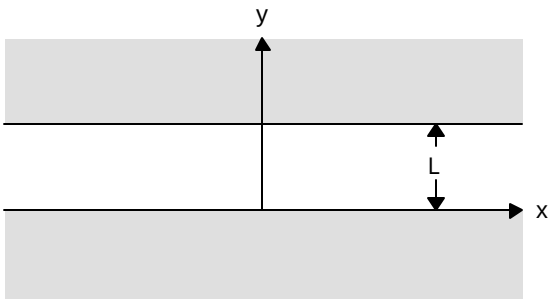


Figure 2.1 - Infinite Channel of Width L

the rigid walls. This implies (in view of (A.7)) that

$$\frac{\partial^2 \eta}{\partial y \partial t} - f\frac{\partial \eta}{\partial x} = 0; y = 0, L \qquad \text{(A.8)}$$

Therefore, substituting in wave solutions that are periodic in x and t of the form

$$\eta = \mathrm{Re}\,\bar{\eta}(y)e^{i(kx-\omega t)} \qquad \text{(A.9)}$$

we obtain an eigenvalue problem for the complex amplitude that varies in the y-direction across the channel, $\bar{\eta}(y)$.

$$\frac{d^2\bar{\eta}}{dy^2} + \left(\frac{\omega^2 - f^2}{C_0^2} - k^2\right)\bar{\eta} = 0 \qquad \text{(A.10a)}$$

$$\frac{d\bar{\eta}}{dy} + f\frac{k}{\omega}\bar{\eta} = 0; y = 0, L \qquad \text{(A.10b)}$$

Solving these yields the eigenvalue relation

$$(\omega^2 - f^2)(\omega^2 - C_0^2 k^2)\sin\left[\left(\frac{\omega^2 - f^2}{C_0^2} - k^2\right)L\right] = 0 \qquad \text{(A.11)}$$

Appendix 2 – Wave field construction (C++):

```
//BROADBAND WAVE MATRIX GENERATOR
#include<iostream.h>
#include<fstream.h>
#include<math.h>

int main()
{
double Pi=3.141592654;
double TwoPi=2.0*Pi;
double RP[1000];
double beta, L, c;
double t[500]={0.0};
double x[500]={0.0};
double wave_array[501][1001]={0.0};
double out_array[501][1001]={0.0};
double power_array[1000]={0.0};
```

```
double omega[1000]={0.0};
double k[1000]={0.0};
double sigsqr=0.0;
int numberofterms=1;

extern float ran1(long *);
long seed=-1;
long *seedpoint=&seed;
ran1(seedpoint);

fstream outfile;
char outfilename[200];

cout << "Please Enter the Output File Name:" << endl;
cin >> outfilename;

cout << "Please Enter an ODD Number of Terms for the Expansion (up to
999):" << endl;
cin >> numberofterms;
while(numberofterms%2==0)
  {
    cout << "That was not an ODD number of terms!" << endl;
    cout << "Please Enter an ODD Number of Terms for the Expansion:" <<
endl;
    cin >> numberofterms;
  }
cout << "Please enter sigma squared for the power term:"<< endl;
cin >> sigsqr;


for (int rancounter=0;rancounter != numberofterms+2;rancounter++)
  {
    RP[rancounter]=TwoPi*ran1(seedpoint);

  }


double omegadominant;
int jdominant=0;
cout << "Please Enter a value for J-dominant:" << endl;
cin >> jdominant;

c=0.85;
beta=2.3E-11;
L=3.65E5;

omegadominant=0.5*(-TwoPi*jdominant/L)*c+0.5*sqrt(pow((-
TwoPi*jdominant/L),2.0)*pow(c,2.0)+4*beta*c);

cout << "jdominant = " << jdominant << endl;
cout << "omegadominant = " << omegadominant << endl;
cout << " c = " << c << endl;
cout << " beta = " << beta << endl;
cout << " L = " << L << endl;
```

```
for(int s=jdominant-((numberofterms-1)/2);s!=jdominant+((numberofterms-
1)/2)+1;s++)
  {
          k[s]=(-TwoPi*s)/L;

omega[s]=((0.5*k[s]*c)+(0.5*sqrt((pow(k[s],2.0)*pow(c,2.0))+(4*beta*c)))
);
          //omega[s]=k[s];
          power_array[jdominant-((numberofterms-1)/2)-1]=0.0;
          power_array[s]=exp(-(pow((s-jdominant),2))/(sigsqr));
          cout << "the j= " << s << " k = " << k[s] << endl;
          cout << "the j= " << s << " omega = " << omega[s] << endl;
          cout << "the j= " << s << " power spectrum coef = " <<
power_array[s] << endl;
  }

int j,i,m,q,r;
for(j=jdominant-((numberofterms-1)/2);j!=jdominant+((numberofterms-
1)/2)+1;j++)
  {
    for(i=0;i!=500;i++)
      {
      for(m=0;m!=1000;m++)
        {
          x[i]=L*i*1.0/1000.0;
          //    t[m]=L*m*1.0/10.0;
          t[m]=(TwoPi/omegadominant)*(m*1.0/200.0);

wave_array[i][m]=wave_array[i][m]+(power_array[j]*cos((k[j]*x[i])-
(omega[j]*t[m])+RP[j]));
          if(j==jdominant+((numberofterms-1)/2))
            out_array[i][m]=wave_array[i][m];
        }
      }
  }

outfile.open(outfilename, ios::out);
for(q=0;q!=500;q++)
  {
    for(r=0;r!=1000;r++)
      {
      outfile << out_array[q][r] << "\t" ;
      }
    outfile << endl;
  }

return 0;
}
```

Appendix 3 – Gaussian noise and Wave field constructor (C++):

```
//BROADBAND and NOISE WAVE MATRIX GENERATOR
#include<iostream.h>
#include<fstream.h>
#include<math.h>
```

```
int main()
{
double Pi=3.141592654;
double TwoPi=2.0*Pi;
double RP[1000];
double beta, L, c;
double t[500]={0.0};
double x[500]={0.0};
double wave_array[501][1001]={0.0};
double out_array[501][1001]={0.0};
double power_array[1000]={0.0};
double omega[1000]={0.0};
double k[1000]={0.0};
double sigsqr=0.0;
int numberofterms=1;

extern float ran1(long *);
long seed=-1;
long *seedpoint=&seed;
ran1(seedpoint);

fstream outfile;
char outfilename[200];

cout << "Please Enter the Output File Name:" << endl;
cin >> outfilename;

cout << "Please Enter an ODD Number of Terms for the Expansion (up to
999):" << endl;
cin >> numberofterms;
while(numberofterms%2==0)
  {
    cout << "That was not an ODD number of terms!" << endl;
    cout << "Please Enter an ODD Number of Terms for the Expansion:" <<
endl;
    cin >> numberofterms;
  }
cout << "Please enter sigma squared for the power term:"<< endl;
cin >> sigsqr;


for (int rancounter=0;rancounter != numberofterms+2;rancounter++)
  {
    RP[rancounter]=TwoPi*ran1(seedpoint);

  }


double omegadominant;
int jdominant=0;
cout << "Please Enter a value for J-dominant:" << endl;
cin >> jdominant;

c=0.85;
beta=2.3E-11;
L=3.65E5;
```

```
omegadominant=0.5*(-TwoPi*jdominant/L)*c+0.5*sqrt(pow((-
TwoPi*jdominant/L),2.0)*pow(c,2.0)+4*beta*c);

cout << "jdominant = " << jdominant << endl;
cout << "omegadominant = " << omegadominant << endl;
cout << " c = " << c << endl;
cout << " beta = " << beta << endl;
cout << " L = " << L << endl;


for(int s=jdominant-((numberofterms-1)/2);s!=jdominant+((numberofterms-
1)/2)+1;s++)
  {
          k[s]=(-TwoPi*s)/L;

omega[s]=((0.5*k[s]*c)+(0.5*sqrt((pow(k[s],2.0)*pow(c,2.0))+(4*beta*c)))
);
          //omega[s]=k[s];
          power_array[jdominant-((numberofterms-1)/2)-1]=0.0;
          power_array[s]=exp(-(pow((s-jdominant),2))/(sigsqr));
          cout << "the j= " << s << " k = " << k[s] << endl;
          cout << "the j= " << s << " omega = " << omega[s] << endl;
          cout << "the j= " << s << " power spectrum coef = " <<
power_array[s] << endl;
  }

double WaveSum=0.0;
int j,i,m,q,r;

for(j=jdominant-((numberofterms-1)/2);j!=jdominant+((numberofterms-
1)/2)+1;j++)
  {
    for(i=0;i!=500;i++)
      {
      for(m=0;m!=1000;m++)
        {
          x[i]=L*i*1.0/1000.0;
          //    t[m]=L*m*1.0/10.0;
          t[m]=(TwoPi/omegadominant)*(m*1.0/200.0);

wave_array[i][m]=wave_array[i][m]+(power_array[j]*cos((k[j]*x[i])-
(omega[j]*t[m])+RP[j]));
          if(j==jdominant+((numberofterms-1)/2))
            {
            out_array[i][m]=wave_array[i][m];
            WaveSum+=(wave_array[i][m]*wave_array[i][m]);
            }
        }
      }

  }
cout << "Wave RMS is " << sqrt(WaveSum/500000)<< endl;
double sigma,a,b;
cout << "Enter the RMS of the noise:" <<endl;
cin >> sigma;
```

```
outfile.open(outfilename, ios::out);
for(q=0;q!=500;q++)
  {
    for(r=0;r!=1000;r++)
      {
      a = ran1(seedpoint);
      b = ran1(seedpoint);
      outfile << out_array[q][r]+ sigma*sqrt(-
2.0*log(a))*cos(2.0*M_PI*b) << "\t" ;

      }
    outfile << endl;
  }

return 0;
}
```

## Appendix 4 – Space-Time Correlation (C++):

```
//CORRELATION FIELD CONSTRUCTOR

#include <iostream.h>
#include <fstream.h>
#include <math.h>

int main()
{
  double data[501][1001]={0.0};
  double r_array[400][900]={0.0};
  double r_sum=0.0;
  double r_max=0.0;
  double r_minusmax=0.0;
  double sx, sy, sxy, sx2, sy2;
  fstream infile;
  fstream outfile;
  char outfilename[200];
  char infilename[200];
  cout << "Enter input filename:" << endl;
  cin >> infilename;
  cout << "Enter output filename:" << endl;
  cin >> outfilename;
  int N=10000;
  int i,j,n,k,s,r;
  cout << "Reading in data....." << endl;
  cout << "Please Wait" << endl;

  infile.open(infilename, ios::in);
  for(int l=0;l!=500;l++)
    {
      for(int m=0;m!=1000;m++)
      {
        infile >> data[l][m];
      }
    }
  infile.close();
```

```
  cout << "Computing Correlations...... " << endl;

  for(i=399;i!= -1;i--)
    {
      cout << "-" << endl;
      for(j=0;j!= 900;j++)
      {
        sx = 0.0;
        sy = 0.0;
        sxy = 0.0;
        sx2 = 0.0;
        sy2 = 0.0;
        for(n =499;n!= 399;n--)
          {
            for(k=0;k!= 100;k++)
            {
              sx += data[n][k];
              sy += data[n-i][k+j];
              sxy += (data[n][k]*data[n-i][k+j]);
              sx2 += (data[n][k]*data[n][k]);
              sy2 += (data[n-i][k+j]*data[n-i][k+j]);
            }
          }

           r_array[i][j] = ((N*1.0*sxy)-(sx*sy))/(sqrt((N*1.0*sx2)-
(sx*sx))*sqrt((N*1.0*sy2)-(sy*sy)));

         if((N*1.0*sx2)-(sx*sx)<=0 && (N*1.0*sy2)-(sy*sy)<=0)
           {
             cout << "exception by means of x and y" << endl;
             goto myLabel;
           }
         if((N*1.0*sx2)-(sx*sx)<=0)
           {
             cout << "exception by means of x" << endl;
             goto myLabel;
           }
         if((N*1.0*sy2)-(sy*sy)<=0)
           {
             cout << "exception by means of y" << endl;
             goto myLabel;
           }
        r_sum+=r_array[i][j];
        if(r_array[i][j]>r_max && (i!=0 && j!=0))
          r_max=r_array[i][j];
        if(r_array[i][j]<r_minusmax && (i!=0 && j!=0))
          r_minusmax=r_array[i][j];
      }
    }


cout << "Creating output file....." << endl;

myLabel:
  cout << " Writing output file...." << endl;
```

```
outfile.open(outfilename, ios::out);
    for(s = 0; s != 400 ; s++)
      {
      for(r = 0; r != 900 ; r++)
        {
          outfile << r_array[s][r] << "\t";// << endl;
        }
      outfile << endl;
      }
outfile.close();

    cout << "r_max = " << r_max << endl;
    cout << "r_sum = " << r_sum << endl;
    cout << "r_minusmax = " << r_minusmax << endl;

return 0;
}
```

Appendix 5 – Histogram Binning of Correlation Function (C++):

```
//HISTOGRAM BINNING

#include <iostream.h>
#include <fstream.h>
#include <math.h>

int main()
{
  double data[501][1001]={0.0};
  double r_array[400][900]={0.0};
  double out_array[400][900]={0.0};
  double r_sum=0.0;
  double r_max=0.0;
  double r_minusmax=0.0;
  double sx, sy, sxy, sx2, sy2;
  fstream infile;
  fstream outfile;
  char outfilename[200];
  char infilename[200];
  cout << "Enter input filename:" << endl;
  cin >> infilename;
  cout << "Enter output filename:" << endl;
  cin >> outfilename;
  int N=10000;
  int i,j,n,k,s,r;
  int counttwo,countone;

  unsigned long int counterzero=0;
  unsigned long int counterone=0;
  unsigned long int countertwo=0;
  unsigned long int counterthree=0;
  unsigned long int counterfour=0;

  cout << "Reading in data......" << endl;
  cout << "Please Wait" << endl;

  infile.open(infilename, ios::in);
```

```
  for(int l=0;l!=500;l++)
    {
      for(int m=0;m!=1000;m++)
      {
        infile >> data[l][m];
      }
    }
  infile.close();

  cout << "Computing Correlations...... " << endl;

  for(i=399;i!= -1;i--)
    {
      cout << "-" << endl;
      for(j=0;j!= 900;j++)
      {
        sx = 0.0;
        sy = 0.0;
        sxy = 0.0;
        sx2 = 0.0;
        sy2 = 0.0;
        for(n =499;n!= 399;n--)
          {
            for(k=0;k!= 100;k++)
            {
              sx += data[n][k];
              sy += data[n-i][k+j];
              sxy += (data[n][k]*data[n-i][k+j]);
              sx2 += (data[n][k]*data[n][k]);
              sy2 += (data[n-i][k+j]*data[n-i][k+j]);
            }
          }

          r_array[i][j] = ((N*1.0*sxy)-(sx*sy))/(sqrt((N*1.0*sx2)-
(sx*sx))*sqrt((N*1.0*sy2)-(sy*sy)));

         if((N*1.0*sx2)-(sx*sx)<=0 && (N*1.0*sy2)-(sy*sy)<=0)
           {
             cout << "exception by means of x and y" << endl;
             goto myLabel;
           }
         if((N*1.0*sx2)-(sx*sx)<=0)
           {
             cout << "exception by means of x" << endl;
             goto myLabel;
           }
         if((N*1.0*sy2)-(sy*sy)<=0)
           {
             cout << "exception by means of y" << endl;
             goto myLabel;
           }
        r_sum+=r_array[i][j];
        if(r_array[i][j]>r_max && (i!=0 && j!=0))
          r_max=r_array[i][j];
        if(r_array[i][j]<r_minusmax && (i!=0 && j!=0))
          r_minusmax=r_array[i][j];
      }
```

```
    }


cout << "Creating output file....." << endl;

for(counttwo = 0; counttwo != 400 ; counttwo++)
  {
    for(countone = 0; countone != 900 ; countone++)
      {
      if(r_array[counttwo][countone] <= r_max-((4.0/5.0)*r_max))
        {
          out_array[counttwo][countone]=1.0;
          counterone++;
        }
      if( r_minusmax-((1.0/5.0)*r_minusmax) >
r_array[counttwo][countone])
        {
          out_array[counttwo][countone]=9.0;
          countertwo++;
        }
      if( r_minusmax-((1.0/5.0)*r_minusmax) <
r_array[counttwo][countone] &&r_array[counttwo][countone] > r_max-
((4.0/5.0)*r_max))
        {
          out_array[counttwo][countone]=5.0;
          counterthree++;
        }
      if(counttwo==countone)
        {
          out_array[counttwo][countone]=13.0;
          counterfour++;
        }
      }
  }
myLabel:
  cout << " Writing output file...." << endl;

outfile.open(outfilename, ios::out);
    for(s = 0; s != 400 ; s++)
      {
      for(r = 0; r != 900 ; r++)
        {
          if(out_array[s][r]==0.0)
            counterzero++;
          outfile << r_array[s][r] << "\t";// << endl;
        }
      outfile << endl;
      }
outfile.close();

cout << "no. in 0 = " << counterzero << endl;
cout << "no. in 1 = " << counterone << endl;
cout << "no. in 2 = " << countertwo << endl;
cout << "no. in 3 = " << counterthree << endl;
cout << "no. in 4 = " << counterfour << endl;
cout << "r_max = " << r_max << endl;
cout << "r_sum = " << r_sum << endl;
```

```
cout << "r_minusmax = " << r_minusmax << endl;

return 0;
}
```

Appendix 6 – Coarse Sampling – TOPEX/POSEIDON equator crossings in space-time:

Appendix 7 – Sparse Sampling (C++):

| Distance (km) | Time Lag (min) |
|---|---|
| 37787.5859 | 10736.938 |
| 37945.45928 | 5733.8623 |
| 38103.12115 | 730.7866 |
| 38260.99452 | 10006.1516 |
| 38418.65639 | 5003.076 |
| 38576.52977 | 0.0003 |
| 38734.19164 | 9275.3653 |
| 38892.07615 | 4272.2896 |
| 39049.73801 | 13547.6547 |
| 39207.62252 | 8544.579 |
| 39365.28439 | 3541.5033 |
| 39523.1689 | 12816.8684 |
| 39680.83077 | 7813.7926 |
| 39838.71528 | 2810.717 |
| 39996.37715 | 12086.082 |
| 40154.26165 | 7083.0063 |
| 40311.92352 | 2079.9306 |
| 40469.80803 | 11355.2957 |
| 40627.4699 | 6352.22 |
| 40785.35441 | 1349.1443 |
| 40943.00514 | 10624.5093 |
| 41100.88965 | 5621.4336 |
| 41258.55152 | 618.358 |
| 41416.4249 | 9893.723 |
| 37787.5859 | 25015.379 |
| 37945.45928 | 20012.3033 |
| 38103.12115 | 15009.2276 |
| 38260.99452 | 24284.5926 |
| 38418.65639 | 19281.517 |
| 38576.52977 | 14278.4413 |
| 38734.19164 | 23553.8063 |
| 38892.07615 | 18550.7306 |
| 39049.73801 | 27826.0957 |
| 39207.62252 | 22823.02 |
| 39365.28439 | 17819.9443 |
| 39523.1689 | 27095.3094 |
| 39680.83077 | 22092.2336 |
| 39838.71528 | 17089.158 |
| 39996.37715 | 26364.523 |
| 40154.26165 | 21361.4473 |
| 40311.92352 | 16358.3716 |
| 40469.80803 | 25633.7367 |
| 40627.4699 | 20630.661 |
| 40785.35441 | 15627.5853 |
| 40943.00514 | 24902.9503 |
| 41100.88965 | 19899.8746 |
| 41258.55152 | 14896.799 |
| 41416.4249 | 24172.164 |
| 37787.5859 | 39293.82 |
| 37945.45928 | 34290.7443 |
| 38103.12115 | 29287.6686 |
| 38260.99452 | 38563.0336 |
| 38418.65639 | 33559.958 |
| 38576.52977 | 28556.8823 |
| 38734.19164 | 37832.2473 |
| 38892.07615 | 32829.1716 |
| 39049.73801 | 42104.5367 |
| 39207.62252 | 37101.461 |
| 39365.28439 | 32098.3853 |
| 39523.1689 | 41373.7504 |
| 39680.83077 | 36370.6746 |
| 39838.71528 | 31367.599 |
| 39996.37715 | 40642.964 |
| 40154.26165 | 35639.8883 |
| 40311.92352 | 30636.8126 |
| 40469.80803 | 39912.1777 |
| 40627.4699 | 34909.102 |
| 40785.35441 | 29906.0263 |
| 40943.00514 | 39181.3913 |
| 41100.88965 | 34178.3156 |
| 41258.55152 | 29175.24 |

```
//CORRELATION FIELD CONSTRUCTOR SPARSE SAMPLING

#include <iostream.h>
#include <fstream.h>
#include <math.h>

int main()
{
  double data[501][1001]={0.0};
  double r_array[400][900]={0.0};
  double r_sum=0.0;
  double r_max=0.0;
  double r_minusmax=0.0;
  double sx, sy, sxy, sx2, sy2;
  int samplespace=0;
  int sampletime=0;
  fstream infile;
  fstream outfile;
  char outfilename[200];
  char infilename[200];
  cout << "Enter input filename:" << endl;
  cin >> infilename;
  cout << "Enter output filename:" << endl;
  cin >> outfilename;
  cout << "Enter a time sparsity index:" << endl;
  cin >> sampletime;
  cout << "Enter a space sparsity index:" << endl;
  cin >> samplespace;


  int N=10000;
  int i,j,n,k,s,r,l,m;
  int q=0,t=0;
  cout << "Reading in data......" << endl;
  cout << "Please Wait" << endl;

  infile.open(infilename, ios::in);
  for(l=0;l!=500;l++)
    {
      for(m=0;m!=1000;m++)
      {
        infile >> data[l][m];
      }
    }
  infile.close();

  cout << "Computing Correlations...... " << endl;

  for(i=399;i > -1;i-=samplespace)
    {
      cout << "-" << endl;
      for(j=0;j < 900;j+=sampletime)
      {
        sx = 0.0;
        sy = 0.0;
        sxy = 0.0;
        sx2 = 0.0;
```

```
        sy2 = 0.0;
        for(n =499;n > 399;n-=samplespace)
          {
            for(k=0;k < 100;k+=sampletime)
            {
              sx += data[n][k];
              sy += data[n-i][k+j];
              sxy += (data[n][k]*data[n-i][k+j]);
              sx2 += (data[n][k]*data[n][k]);
              sy2 += (data[n-i][k+j]*data[n-i][k+j]);
            }
          }

           r_array[q][t] = ((N*1.0*sxy)-(sx*sy))/(sqrt((N*1.0*sx2)-
(sx*sx))*sqrt((N*1.0*sy2)-(sy*sy)));

         if((N*1.0*sx2)-(sx*sx)<=0 && (N*1.0*sy2)-(sy*sy)<=0)
           {
             cout << "exception by means of x and y" << endl;
             goto myLabel;
           }
         if((N*1.0*sx2)-(sx*sx)<=0)
           {
             cout << "exception by means of x" << endl;
             goto myLabel;
           }
         if((N*1.0*sy2)-(sy*sy)<=0)
           {
             cout << "exception by means of y" << endl;
             goto myLabel;
           }
        r_sum+=r_array[q][t];
        if(r_array[q][t]>r_max && (i!=0 && j!=0))
          r_max=r_array[q][t];
        if(r_array[q][t]<r_minusmax && (i!=0 && j!=0))
          r_minusmax=r_array[q][t];
        t++;
      }
      q++;
    }


cout << "Creating output file....." << endl;

myLabel:
  cout << " Writing output file...." << endl;

outfile.open(outfilename, ios::out);
    for(s = 0; s != q ; s++)
      {
      for(r = 0; r != t ; r++)
        {
          outfile << r_array[s][r] << "\t";// << endl;
        }
      outfile << endl;
      }
outfile.close();
```

```
    cout << "r_max = " << r_max << endl;
    cout << "r_sum = " << r_sum << endl;
    cout << "r_minusmax = " << r_minusmax << endl;

return 0;
}
```

Appendix 8 – Random Number Generator (C++):

```
#define IA 16807
#define IM 2147483647
#define AM (1.0/IM)
#define IQ 127773
#define IR 2836
#define NTAB 32
#define NDIV (1+(IM-1)/NTAB)
#define EPS 1.2e-7
#define RNMX (1.0-EPS)


float ran1(long *idum)
{
  int j;
  long k;
  static long iy=0;
  static long iv[NTAB];
  float temp;

  if (*idum <= 0 || !iy)
    {
      if (-(*idum) < 1) *idum=1;
      else *idum = -(*idum);
      for (j=NTAB+7;j>=0;j--)
      {
        k=(*idum)/IQ;
        *idum=IA*(*idum-k*IQ)-IR*k;
        if (*idum < 0) *idum += IM;
        if (j < NTAB) iv[j] = *idum;
      }
      iy=iv[0];
    }
  k=(*idum)/IQ;
  *idum=IA*(*idum-k*IQ)-IR*k;
  if (*idum < 0) *idum += IM;
  j=iy/NDIV;
  iy=iv[j];
  iv[j] = *idum;
  if ((temp=AM*iy) > RNMX) return RNMX;
  else return temp;
}
```

References:

Apel, J.R. *Principles of Ocean Physics.* Academic Press: London, 1987.

Benada, Robert J. *Merged GDR (TOPEX/POSEIDON) Generation B: User Handbook.* Physical Oceanography Distributed Active Archive Center, Jet Propulsion Laboratory: California, 1997.

Berger, Neil. "Derivation of Approximate Long Wave Equations in a Nearly Uniform Channel of Approximately Rectangular Cross Section," in *SIAM Journal of Applied Mathematics.* Vol. 31, No. 3, November 1976.

Brooks, Ronald L., Dennis W. Lockwood, and Jeffrey E. Lee. "Land Effects on TOPEX Radar Altimeter Measurements in Pacific Rim Coastal Zones," from the Laboratory for Hydrospheric Processes, Wallops Flight Facility, NASA Goddard Space Flight Center, Wallops Island, VA 23337 USA.

Cane, Mark A., and E.S. Sarachik. "Forced Baroclinic Ocean Motions," a three part series in Journal of Marine Research. Vol. 34, 35, 37.

Kamenkovich, V.M. *Fundamentals of Ocean Dynamics.* Elsevier scientific Publishing Company: New York, 1977.

Kaufman, A.N., J.J. Morehead, A.J.Brizard, and E.R.Tracy. "Mode Conversion in the Ocean," To appear in the Journal of Fluid Mechanics.

Kraus, Eric B. and Joost A. Businger. *Atmosphere-Ocean Interaction.* Oxford University Press: New York, 1994.

*Ocean Wave Measurement and Analysis.* Edited by Billy L. Edge and J. Michael Hemsley. International Symposium on Ocean Wave Measurement and Analysis with the American Society of Civil Engineers: Reston, 1998. 2 volumes.

Pedlosky, Joseph. *Geophysical Fluid Dynamics.* Springer-Verlag: New York, 1979.

*Radar Scattering from Modulated Wind Waves.* Edited by G.J. Komen and W.A. Oost. Kluwer Academic Publishers: London, 1989.

Sciremammano, Frank. "Notes and Correspondence: A Suggestion for the Presentation of Correlations and Their Significance Levels," in *Journal of Physical Oceanography.* Volume 9, November 1979.

The data from the satellite is obtained on CD-ROM from the JPL and NASA website, http://topex-www.jpl.nasa.gov/ on TOPEX/POSEIDON. Each CD has three 9.92 day cycles on it, each one consisting of 254 tracks with known equatorial passing longitudes.

Software developed by JPL (in C) is provided to decompress and label the data and a hard-copy handbook is provided to explain the labels and general organization.