Scientific Computation Project 3

Marwan Riach, CID: 01349928

March 27, 2020

Part 1

1.1)

In repair1, code has been added to make the function complete (in the B section). The code that has been provided in repair1 does provide a correct output when it is completed, however there are numerous inefficiencies:

- R0 is a copy of R and this is never used and does not need to be used
- *ak* and *bk* are never used in the code
- The set, S, is never used
- Before proceeding with the iterations, permutations should be pre-computed
- Iterating through *mlist* and *nlist* can be avoided by vectorising
- No stopping conditions are used for when successive A and B are very close in value numerically despite dA and dB being calculated

My solution of repair2 addresses all of the above inefficiencies. Specifically, for the random permutations, I cycle over three different seeds, so as to preserve variety when updating individual elements, but avoid computing *niter* sets of random permutations which is time-wise fairly costly. Furthermore, all superfluous calculations are removed and everything is vectorised where possible. In addition to this, stopping conditions are added so that if successive values of A and B are sufficiently close, the whole algorithm is terminated. Running repair1 and repair2 for $p = 5$ results in running times of 68 seconds and 5 seconds respectively, corroborating the superiority of repair2.

Below is the visualisation of the heights of the waves, using a rank of 5 at $\lambda = 0$:

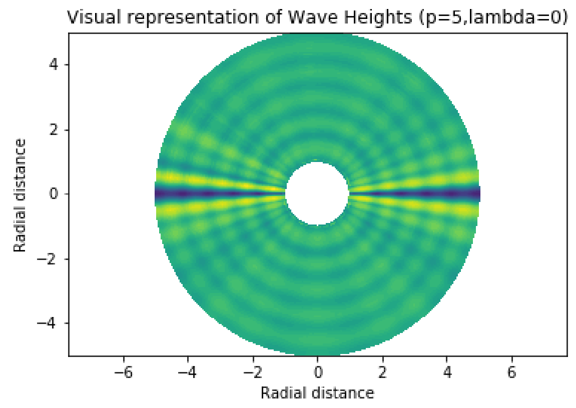

Figure 1: Image representation of repaired dataset

In the case of $\lambda = 0$, we are finding A and B such that we minimise the modified Frobenius norm (as discussed in lecture 14). Below is a plot of the natural logarithm of the singular values of our original data1 matrix:

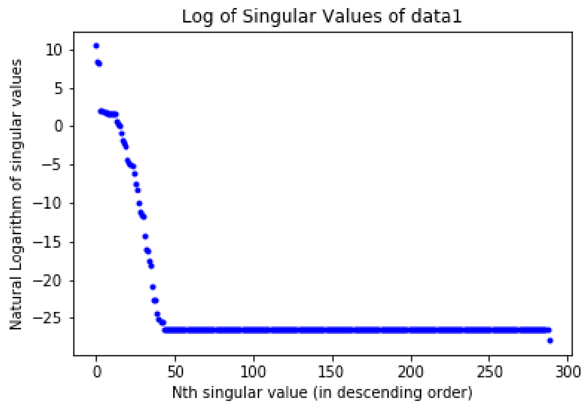

Figure 2: Natural logarithm of singular values of data1

What this plot tells us is that the vast majority of information in the matrix is retained when using a rank of 5. Therefore, reducing the rank from 300 to 5 to fill in the missing data provides an accurate approximation. It is also worth noting, that the rank of our data1 matrix is 47, and so using a p value greater than 47 would be completely superfluous. The same principle applies for non-zero values of λ; the extent to which data is retained changes very little, except that increasing the value of λ leads to smaller values of A and B, and consequently, lower values for the heights.

1.2.i)

The equation by which the data in data2 is governed is the wave equation, which when written in polar form with c equal to 1 is as follows:

$$\frac{\partial^2 h}{\partial t^2} \; = \; (c=1)^2 \left(\frac{\partial^2 h}{\partial r^2} + \frac{1}{r}\frac{\partial h}{\partial r} + \frac{1}{r^2}\frac{\partial^2 h}{\partial \theta^2} \right) \tag{1}$$

From multivariable calculus, we know that the general solution to this equation is of the following form, which has used the sepration of variables technique:

$$h(r,\theta,t) = \phi(r,\theta)\cos(\omega t + \delta) \tag{2}$$

For the purposes of the rest of the write up, I will represent the t component as follows:

$$T(t) = \cos(\omega t + \delta)$$

Plugging the second equation into the first yields the following result:

$$-\left(\frac{\partial^2 \phi}{\partial r^2} + \frac{1}{r}\frac{\partial \phi}{\partial r} + \frac{1}{r^2}\frac{\partial^2 \phi}{\partial \theta^2} \right) \; = \; \omega^2 \phi \tag{3}$$

Using the separation of variables method a second time, this time on ϕ:

$$\phi(r,\theta) = R(r)\Theta(\theta)$$

Plugging this into the third equation yields:

$$\frac{r^2 R''}{R} + \frac{rR'}{R} + \omega^2 r^2 = \frac{\Theta''}{\Theta} = k$$

where k is the separation constant.
The boundary conditions of the system are such that:

$$\Theta(0) = \Theta(2\pi), \Theta'(0) = \Theta'(2\pi)$$

And from multivariable calculus, it follows that:

$$\Theta = A_m \cos(m\theta) + B_m \sin(m\theta)$$

$$k = m^2 (m \in N)$$

Now considering the R component of the equation:

$$r^2 \frac{\partial^2 R}{\partial r^2} + r\frac{\partial R}{\partial r} + R(\omega^2 r^2 - m^2) = 0$$

Applying the change of variable $x = r\omega$ to the equation directly above results in Bessel's equation.

One of the solutions to Bessel's equation is $H_{m,1}(r\omega)$, which is the Hankel function of the first kind. The output of this function is complex-valued, and so in the context of the problem, only the real part of this function is what we wish for.

Since m covers the natural numbers, the full solution of R is:

$$R(r) = \sum_{m=0}^{\infty} H_{m,1}(r\omega)$$

with the real value being taken as the solution.

In my coded solution, I take the sum of the first 10 values of my Hankel function since the sum does converge as you approach m to infinity for most r. As a result of this, the output that is produced in the coded solution is not perfectly accurate, though it does provide the heights of the waves at times between 0 and $\frac{\pi}{4}$ and θ between 0 and 2π.

In order to do this, I extract the values at $r = 1$ and then proceed as follows, exploiting the format of the separation of variables:

$$f = h(r = 1, \theta, t) = R(1)\Theta(\theta)T(t)$$

$$h(r = r_0, \theta, t) = R(r_0)\Theta(\theta)T(t) = \frac{fR(r_0)}{R(1)}$$

The above shows how the heights where r is equal to $r0$ are extracted. All that is left to do to complete the extraction is to work out the value of omega.

Looking at equation (2), we can see that the time component of the equation is sinusoidal. Therefore, the value of omega can be determined by seeing when the function repeats itself at fixed r and θ. Below is a plot of the heights over time at a chosen radius and argument:

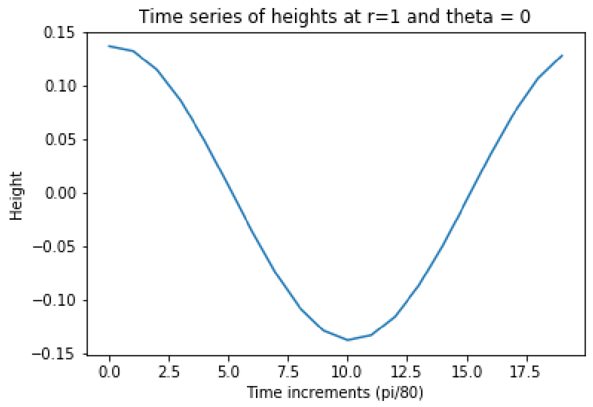

Figure 3: Time series of heights between t=0 and pi/4

As can be seen here, the time series repeats itself at the end of the graph. This means that the value of ω is simply equal to $\frac{2\pi}{\frac{20\pi}{80}} = 8$.

1.2.ii)

Below is a plot of how the waves' heights oscillate over time at the boundary of the island and the sea. Looking at the graph, it appears that the sea is calmer at the argument of $\frac{\pi}{4}$ as compared with the other arguments. In addition to this, unlike in data2, there does not appear to be any sense of periodicity which tells that the waves do not oscillate sinusoidally in time; there is also a similar volatility among the arguments of $\frac{3\pi}{4}$ and $\frac{5\pi}{4}$. Furthermore, the means of the heights over time at the different arguments are all very nearly zero showing one commonality across all arguments.

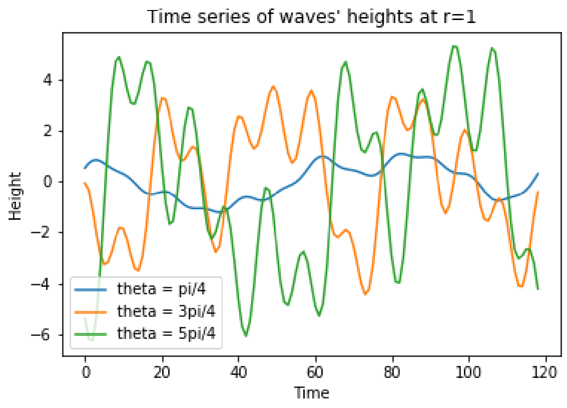

Figure 4: Behaviour of waves over time at different theta values, r=1

The behaviour of the heights of the waves at the three different arguments appears to be much more similar at a further distance of $r = 5$ away from the centre of the circular island as can be seen from Figure 5. Another point worth mentioning is that the range of heights at this distance is 2 units lower than at $r = 1$, which would perhaps indicate that as you approach an infinite distance from the island, the height of the waves tends to zero. Additionally, at this distance, the time series appears much less chaotic and more periodic across all given values of θ.

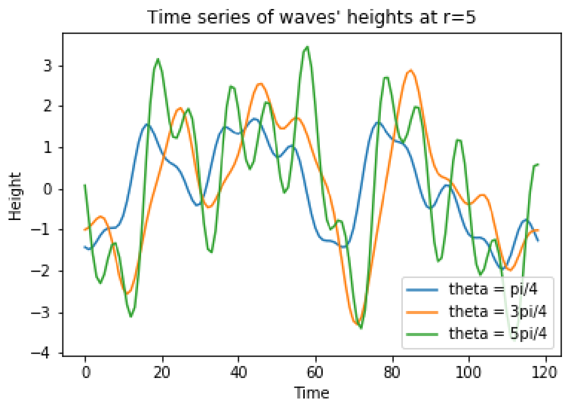

Figure 5: Behaviour of waves over time at different theta values, r=5

What can be seen from Figure 6 is that across all radial values and all arguments that we are observing, the mean of the heights of the waves across the given time period is of the order of 10^{-15} which to all intents and purposes is 0; this is another commonality across all three arguments.

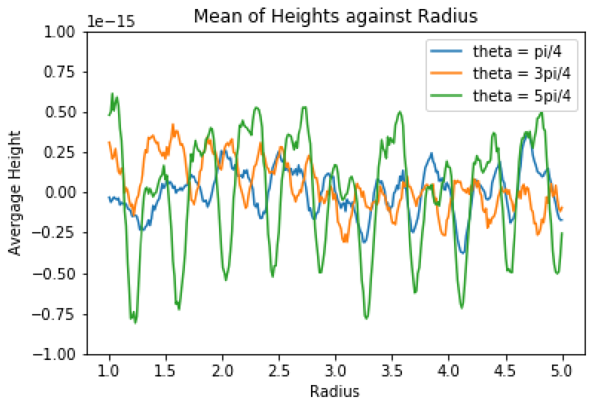

Figure 6: Mean of heights across a fixed time period over different radii

In Figure 7, the arguments of $\frac{3\pi}{4}$ and $\frac{5\pi}{4}$ both appear to show that the oscillatory behaviour of the waves becomes less tempestuous as you get further away from the island. What is interesting to note is that at $\frac{\pi}{4}$, the variance gradually increases to the same as that of the other two arguments; this agrees with the findings in Figure 5, which showed that the behaviour at $\theta = \frac{\pi}{4}$ begins to mimic the behaviour at the other two arguments, as you draw further away from the island.

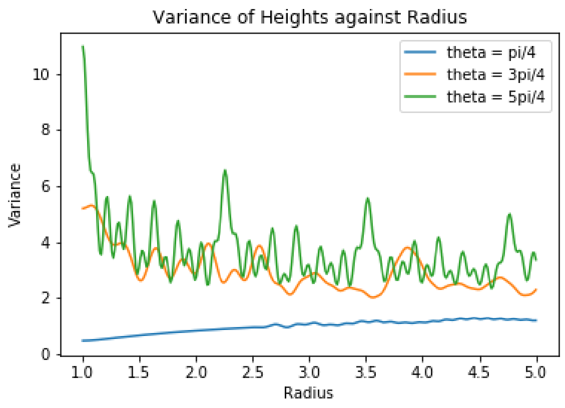

Figure 7: Variance of heights across a fixed time period over different radii

Another tool that can be used to analyse the time series of the height dynamics at different radial values is the power spectrum. The power spectrum of a time series describes the distribution of power into frequency components composing that signal. According to $Fourier$ analysis, any physical signal can be decomposed into a number of discrete frequencies, or a spectrum of frequencies over a continuous range; where there is greater power associated with a particular frequency, tells us the extent to which the particular frequency is prevalent in the time signal.

Figures 8 and 9 produce a plot of the power spectral densities at $r = 1$ and $r = 5$ respectively. The method used to produce these plots was $Welch's$ method. $Welch's$ method was used here due to the fact that the time series we are dealing with are aperiodic; due to the aperiodicity, we are unable to use discrete Fourier transforms and so $Welch's$ method is used in lieu of this. We are told that the time increment of the dataset is $dt = 1$, so there is no scaling applied to the sample frequencies along the x axis of figures 8 and 9.

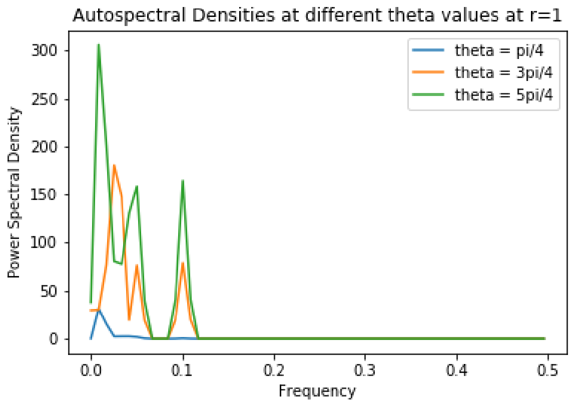

Figure 8: Spectral densities, r=1

Figures 8 and 9 agree with what has already been identified in figure 4 and 5: that there is more "power" in the waves at $\theta = \frac{3\pi}{4}$ and $\frac{5\pi}{4}$ at $r = 1$ and that at $r = 5$, the behaviour of all three time series is much more similar than at $r = 1$.

In addition to this, at $r = 5$, what can be deduced from figure 8 and 9 is that at $\theta = \frac{3\pi}{4}$ the time series oscillates at a faster rate from peak to trough; this is demonstrated by a higher PSD at a lower frequency as compared with at $r = 1$. This finding is supported by figure 5 and 6 where you can see that the orange time series oscillates between positive and negative values slightly more frequently at a radial distance of 5.

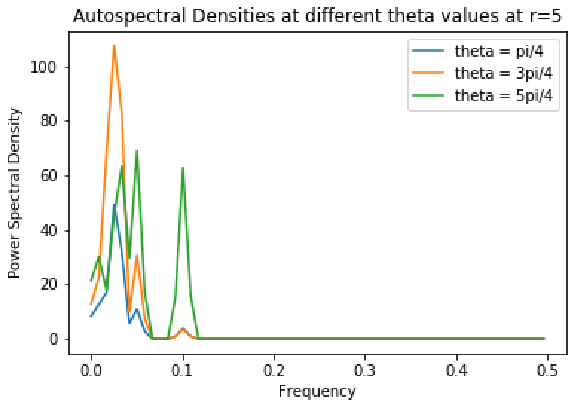

Figure 9: Spectral densities, r=5

1.3)

In order to reduce the data3 dataset, I used the method known as Principal Component Analysis. This involved me storing data at each time frame from which I could then reconstruct a dataset with the same dimensions as data3; I performed PCA on a $2D$ array and did this 119 times (the number of time frames that are recorded), storing vectors from which the dataset could be reconstructed.

At each time frame, data3 is broken down using Singular Value Decomposition and the singular values are listed in descending order; the vectors associated with the largest singular values are the ones which tell us most of the information about our dataset. It is worth noting that in data3, the distribution of singular values for each timeframe was very similar. Below is a graph of the cumulative sum of singular values and a line drawn at $x = 39$, which is what I decided to use as my cut-off point, since this stores 40 vectors for each time frame.

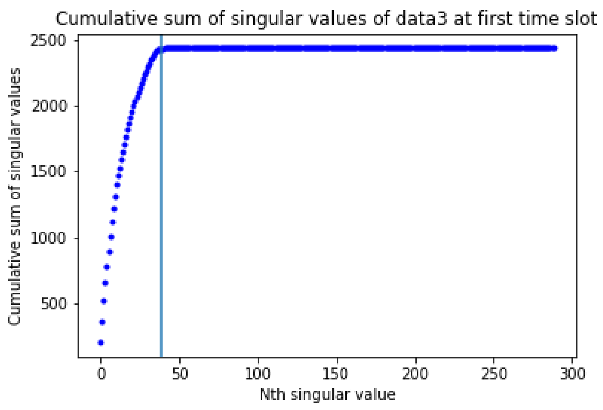

Figure 10: Singular values' cumulative sum with x=39

The reconstructed array is simply the sum of the outer product of the vectors that have been stored and outputted by the reduce function; *np.dstack* is used to concatenate the 119 $2D$ arrays and output array with the same dimensions as data3. The vectors that are stored are the first 40 U-vectors from SVD and the first 40 rows of the G matrix which is the dot product of the transpose of all Us and the dataset, data3.

The extent to which "key features" are withheld is proportional to the cumulative sum of the singular values, since the key features are represented by the vectors, U. Storing the first 40 such vectors corresponds to an information retention rate of $\frac{cumsum(S[39])}{cumsum(S[288])} = 0.99917...$ (here S is the vector of singular values in the SVD decomposition).

As can be seen from Figure 10, had the first 47 vectors been stored, no information would have been lost, since the singular values were equal to 0 from that point onwards.

The amount of memory that is saved in this process simply corresponds to the number of datapoints that is outputted in reduce as compared with the size of the original dataset3:

$$\frac{119(300 * 40 + 40 * 289)}{300 * 289 * 119} = 0.271...$$

In other words, the reduce function approximately reduces the amount of memory stored four-fold and virtually all of the information is retained. This is a significant improvement on the original problem.

Were I to extend this further, I would perform SVD in another dimension of the data3 matrix (t or r) so as to improve the extent to which memory is stored.

Part 2

2.1.i)

To complete $newdiff$ my approach was to set up a linear system of equations $Ax = b$, where A is a banded matrix, x is the vector of second derivatives we wish to compute, and b is the vector of f values.

Since A is a banded matrix, the package $scipy.linalg.solve_banded$ in Python was used as this has superior efficiency to a general matrix and even a sparse matrix when solving the system of equations for x.

In order to do this, A was converted into a matrix "ab" with dimensions $3 * N$ (see scipy documentation for examples) with each row representing the three diagonals of A with data in it.

Furthermore, the construction of my RHS b vector was completed using vectorisation to maximise efficiency and was divided by the h^2 term at the end so as to avoid superfluous computation.

The entries of A and b pertain to the compact finite difference scheme as outlined in the question.

2.1.ii)

To assess the accuracy of the compact scheme and centred scheme I carried wavenumber analysis on the wave function in a manner akin to lecture 16, except the second derivative was considered instead of the first. What could be seen is that for small values of kh, both schemes are appropriate, however for larger kh, the compact scheme is more appropriate. Both of these schemes scale linearly as you increase N, however both schemes will never be as accurate as the $Fourier$ method since this is the most accurate method for periodic functions. The drawback, of course, with using the $Fourier$ method is that the asymptotic complexity is $O(N \log_2 N)$.

Given that the compact scheme is a tridiagonal linear system and the centred scheme is a simpler finite difference system, the operational cost of the tridiagonal system outweighs the cost of the centred scheme with circa $7N$ operations as compared with N operations for the centred second order centred scheme. Therefore, for small kh, the centred scheme should be used as it is faster and the accuracy is negligibly different to the compact scheme. For larger kh, the compact scheme should be used since the result is considerably more accurate. To make a critical and numerical comparison between the accuracy of the two schemes over values of kh between 0 and π, I used the following approach:

The function considered was $f = e^{ikx}$ which has second derivative $f'' = -k^2 e^{ikx}$.

The second derivative using the second order centred scheme is:

$$f''_j = \frac{f_{j+1} - 2f_j + f_{j-1}}{h^2}$$

Replacing the above equation with our wave function yields the following result:

$$-k^2 e^{ikx} = \frac{e^{ik(x+h)} - 2e^{ikx} + e^{ik(x-h)}}{h^2}$$

$$\Rightarrow -k^2 h^2 e^{ikx} = 2e^{ikx}(\cos(kh) - 1)$$

$$\Rightarrow (kh)^2 = 2(1 - \cos(kh))$$

And so the 2nd order centre finite difference scheme is accurate when $(kh)^2$ is close to the above equality.

Now to consider the second derivative using the compact finite difference scheme which is below:

$$\alpha f''_{i-1} + f''_i + \alpha f''_{i+1} = \frac{1}{h^2}\left[\frac{c}{9}\left(f_{i+3} - 2f_i + f_{i-3}\right) + \frac{b}{4}\left(f_{i+2} - 2f_i + f_{i-2}\right) + a\left(f_{i+1} - 2f_i + f_{i-1}\right)\right]$$

Considering the left hand side of the equation and simplifying and grouping the exponential terms as was done in the above scheme...

$$\alpha f''_{i-1} + f''_i + \alpha f''_{i+1} = -k^2(2\alpha\cos(kh) + 1)e^{ikx}$$

And the right hand side...

$$\frac{1}{h^2}\left[\frac{c}{9}\left(f_{i+3} - 2f_i + f_{i-3}\right) + \frac{b}{4}\left(f_{i+2} - 2f_i + f_{i-2}\right) + a\left(f_{i+1} - 2f_i + f_{i-1}\right)\right]$$

$$= \frac{2e^{ikx}}{h^2}\left[\frac{c}{9}\left(\cos(3kh) - 1\right) + \frac{b}{4}\left(\cos(2kh) - 1\right) + a\left(\cos(kh) - 1\right)\right]$$

Equating the right hand side and the left hand side gives us the conditions under which the compact finite difference scheme is an accurate representation of the second derivative of the function. (I.E. when the RHS and LHS are approximately the same)

The RHS is approximately equal to the LHS when $(kh)^2$ satisfies approximate equality with the RHS of the equation below...

$$(kh)^2 = -2\frac{\left[\frac{c}{9}\left(\cos(3kh) - 1\right) + \frac{b}{4}\left(\cos(2kh) - 1\right) + a\left(\cos(kh) - 1\right)\right]}{2\alpha\cos(kh) + 1}$$

Below is a plot of the two schemes as compared with $(kh)^2$ over the range between 0 and π. It is worth noting here that the α value used is $\frac{9}{38}$ as provided in the assignment:

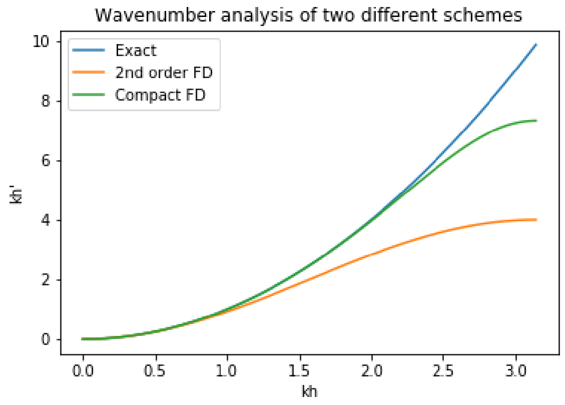

Figure 11: Wavenumber analysis of the finite different schemes

The figure above shows immediately that for small kh, the second order FD scheme behaves very similarly to the true derivative, and so given its computation cost, it should be used for small kh.

Specifically, the error is 1% when kh is approximately 0.345. Using the method as in lecture 16, having a 1% error at 0.345 implies that $\frac{\lambda}{h} \approx \frac{2\pi}{0.345} \approx 18.2$. In other words, if at each wavelength, we have approximately 18 points, then the error of the second derivative that is created is one percent.

For the compact FD scheme, the 1% error occurs when kh is approximately 2.02, significantly higher than the previous scheme. This implies that $\frac{\lambda}{h} \approx \frac{2\pi}{2.02} \approx 3.09$. In other words, if at each wavelength, we have approximately 3 points, then the error of the second derivative that is created is one percent.

The conclusion that is drawn here is that if we have data providing us with 18 or more points per wavelength, then the second order finite difference scheme is preferable. If, however, the data with which we are provided contains fewer than 18 points at each wavelength, then the compact finite difference scheme is superior given its higher accuracy.

2.2)

In the context of the question, f and g are competitive dynamics which would suggest that the dynamics of one is inextricably linked to the dynamics of the other. Figures 12 to 15 appear to illustrate this; the shape and outline of the figures pertaining to f match the shape and outline of the figures pertaining to g. The key difference, of course, is the colour as this is what is used to represent the concentrations of the two respective species.

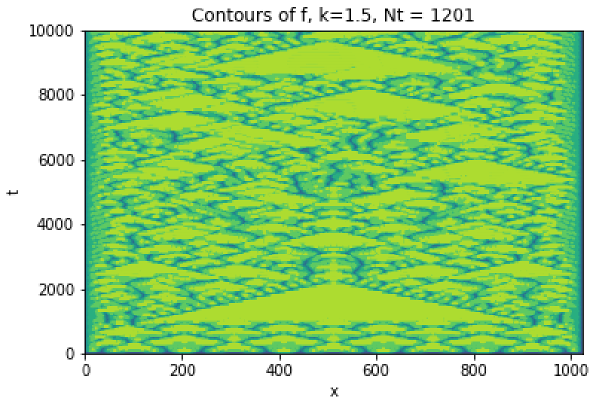

Figure 12: Dynamics of f

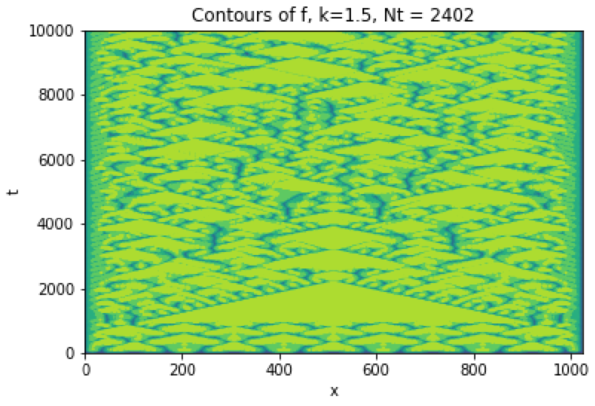

Figure 13: Dynamics of f

What can be seen from all Figures 12-17 is that the dynamics appear to be symmetric about the line $x = 512$ up to around $t = 4000$. This would suggest that initially, the dynamics of the systems lack chaos up to this time; were I to explore further, I would have carried out a bifurcation diagram plot to see the time at which the dynamics begin to exhibit chaotic behaviour.

Furthermore, doubling the value of Nt, as I have done in Figure 13 reveals more information than in Figure 12. This can be explained by the fact that the system used to compute the solutions uses a second order finite difference scheme, and so for smaller Nt, the derivatives at each stage are more accurate. (See 2.1 for the theory behind this).

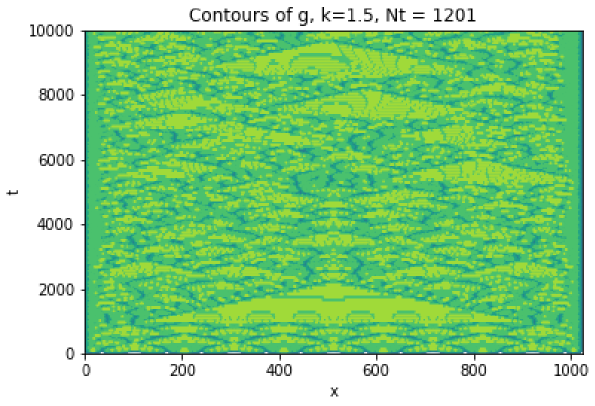

Figure 14: Dynamics of g

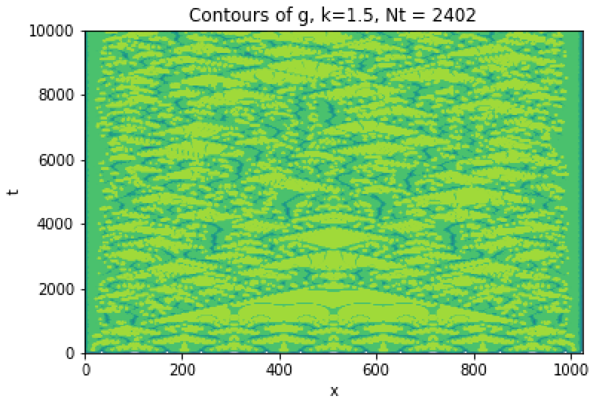

Figure 15: Dynamics of g

Over the three values of κ that I have produced plots for, there is a long time period for values below 1.7 where the system is in a steady state (as visualised by the constant colour gradient in the large triangle towards the bottom of the diagram). This is not the case at $\kappa = 2$.

One commonality among all of the figures is that as you march forward in time, no patterns appear visible, suggesting a high degree of chaos as time progresses and an ignoring of the initial transient.

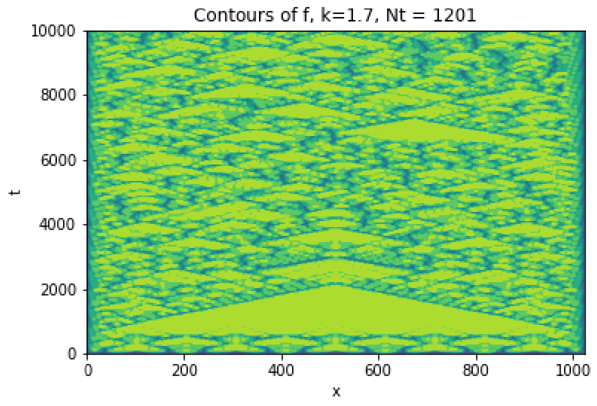

Figure 16: Dynamics of f

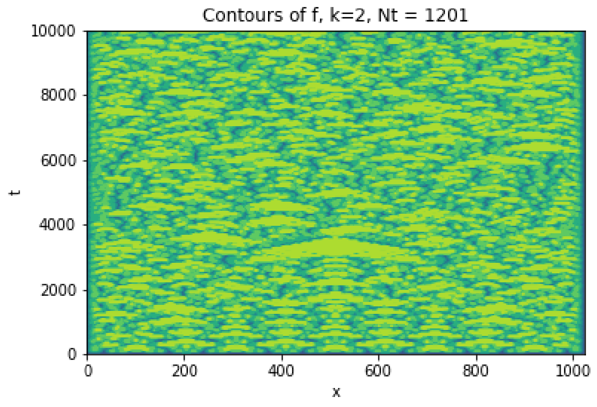

Figure 17: Dynamics of f

In order to determine the extent to which the dynamics in f, g and $data3$ correspond to chaos I am going to use the metric known as the fractal dimension.

The fractal dimension is such that if the dimension exceeds 2, the system is believed to be chaotic; and the higher the fractal dimension, the more chaos that is exhibited by the dynamics of the system.

Quantitatively, the fractal dimension is computed by considering the gradient of the logarithmic plot of $C(\epsilon)$ against ϵ. Here $C(\epsilon)$ is the correlation sum and this tells us the number of pairs of points in a dynamical system that are within an ϵ Euclidean distance of one another. Unsurprisingly, as ϵ increases, $C(\epsilon)$ increases and this makes sense as fractal dimensions, by definition, are strictly positive. $C(\epsilon)$ numerically represents the proportion of points within a given distance ϵ, and so it is fixed between 0 and 1 (see part2 code for annotation of algorithm).

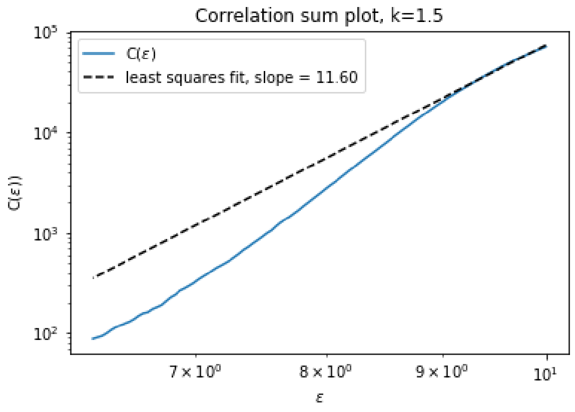

Figure 18: Correlation sum plot to determine fractal dimension

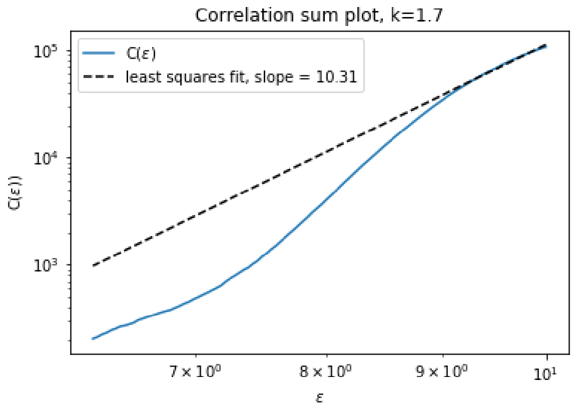

Figure 19: Correlation sum plot to determine fractal dimension

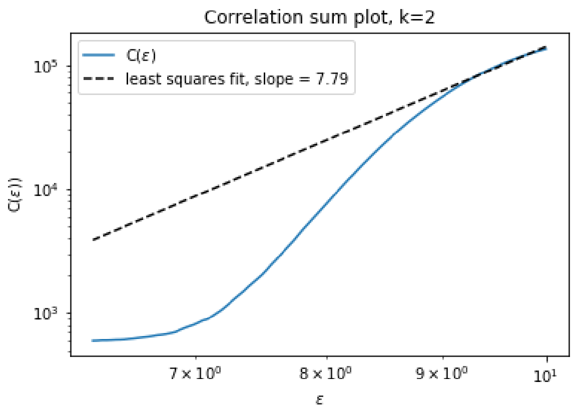

Figure 20: Correlation sum plot to determine fractal dimension

What can be seen from figures 18-20 is that as you increase κ, the fractal dimension decreases which suggests that as you approach the solution to the set of differential equations in the limit as time approaches infinity, higher κ culminates in less chaos. It is also worth noting that the calculations here are derived from f and only f, since the extent of chaos in g is governed by the extent of chaos in f and vice versa. The fractal dimension is determined by the gradient of the slope of the least squares fit. These values all exceed 2, so the system we have considered here is chaotic.

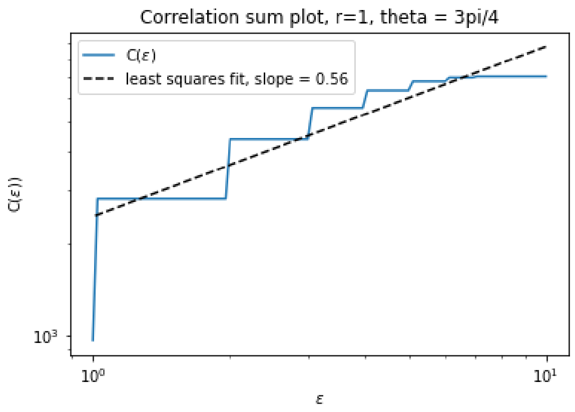

Figure 21: Correlation sum plot to determine fractal dimension, data3

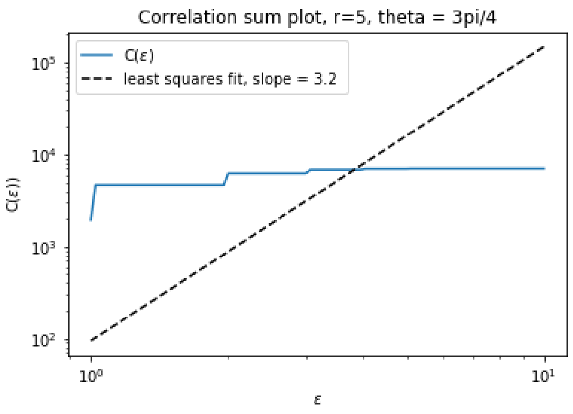

Figure 22: Correlation sum plot to determine fractal dimension, data3

For the *data*3 analysis, I simply considered the correlation sum plot at two different values of r and fixed θ. The θ value I used was $\frac{3\pi}{4}$ since 1.2.ii also gives information of the time series generated at the radii and argument I have chosen, providing more of a reference point.

At $r = 1$, the slope of the least squares fit is 0.56, which suggests that there is little chaos in the system. As a physical interpretation, this makes sense, since the amplitude of waves tends to zero as you increase your distance away from the island; a chaotic wave system would probably oscillate aggressively irrespective of its initial perturbation and distance from the island.

One caveat that is worth bearing in mind is the fractal dimension derived at $r = 5$. It can be seen immediately that the least squares fit is indeed a poor one and that the gradient of this system should be less than the gradient of the system at $r = 1$ (seems very close to zero by initial inspection). Therefore the conclusion that I draw from *data*3 at the considered argument of $\frac{3\pi}{4}$ is that the system is not chaotic.

2.3)

For finite values of T,the second half of the time series produced could be used to compute the fractal dimension of the dynamics. Of course, if the fractal dimension exceeds 2, then it can be deduced that the dynamics amount to chaos. To compute this, the correlation sum at values of epsilon is plotted logarithmically against the values of epsilon and the gradient is computed, referring to the fractal dimension. In addition to this, a lag plot of successive times could be plotted to ascertain the periodicity of the system: if the plot resembles a continuous function, then the function is periodic, else it is aperiodic.

Furthermore, over time T, the time series could be plotted up to interval values of T. The *Fourier* coefficients could then be plotted to see if at any time between 0 and T, the coefficients undergo no decay. If there were to exist a T'

between 0 and T such that the $Fourier$ coefficients underwent no decay, then the dynamics would be periodic and would therefore not represent chaos.

Over an infinite time span, to determine the long-term dynamics of the system, the Lyapunov exponent should be calculated. In order to compute this the following should be carried out:

- Pick initial arbitrary condition and march this forward in time to, say, $t = 5$, giving us a new initial condition
- Perturb our new initial condition by some $\epsilon << 1$
- Observe the Euclidean distance squared between the new initial condition and perturbed initial condition for large T
- Plot this squared distance over time on a semi-log plot
- Using $np.polyfit$ with $degree = 1$ plot the line of best fit on this semi-log plot and record the gradient
- Halve this gradient to obtain the Lyapunov exponent

What the Lyapunov exponent tells us is the maximum rate of departure for 2 initial conditions, which is in effect, a further measure of chaos in the asymptotic setting. If the Lyapunov exponent is close in value to 0, then this tells us that there is no chaos in our system. If our Lyapunov exponent is large and positive, then the dynamics over an infinite time span are chaotic.