university of groningen

faculty of mathematics
and natural sciences

Time-Frequency Analysis of the Kuramoto Model

Bachelor Project Mathematics

June 2016

Student: B.J. Liefting

First supervisor: Prof. H. Waalkens

Second supervisor: Prof. A. van der Schaft

Abstract

The phenomenon of synchronisation is studied by means of the Kuramoto model. This model describes a large population of coupled oscillators with natural frequencies taken from a narrow distribution. It is assumed that the coupling between the oscillators is mean-field and purely sinusoidal. We follow Kuramoto's analysis to obtain a formula for the critical coupling. Then the properties of the Kuramoto model are studied with the aid of Poincaré maps. We then conclude with a time-frequency analysis of the order parameter. With the aid of this time-frequency analysis we were able to detect partial synchronisation.

Contents

1 Introduction

One of the interesting phenomena of nature is that of collective synchronisation. The mathematics behind this will be discussed here. We will consider a large collection of oscillators that are coupled to each other. Thus the oscillators interact.

We introduce the concept of synchronisation and the model that Kuramoto used to describe it. We are interested in the behaviour of the oscillators for different distributions of their natural frequencies. How strong should the oscillators be coupled to each other in order for them to synchronise? We will introduce a measure of synchronisation. This measure will be analysed using time-frequency analysis. The insights gained from this particular analysis will be discussed. We will see that we cannot only detect global synchronisation but also partial synchronisation using this method. Thus even though not all oscillators are part of the synchronised pack, we can detect those oscillators that have synchronised. Even the cases where multiple packs synchronise separately to different frequencies can be detected by time-frequency analysis.

We will start with an introduction to synchronisation in Section 2. We will make some assumptions on the oscillators and their interaction in order to make the problem tractable. These assumptions will be discussed in Sections 3 and 4. After having introduced the Kuramoto model, we will introduce the order parameter in Section 5. In Section 6 we follow the analysis done by Yoshiki Kuramoto. We then proceed with an analysis of the model by introducing a Hamiltonian system in which the Kuramoto model is contained. The Hamiltonian system is then studied with the aid of Poincaré maps in Section 7. Finally we introduce time-frequency analysis in Section 8 in order to study the Kuramoto model for a larger number of oscillators.

2 Synchronisation

As mentioned before, we will consider a large collection of coupled oscillators and study the behaviour of the system. Such a large collection of oscillators might spontaneously lock into a common frequency. In other words, without force put upon them, the oscillators will take on the same frequency despite having different natural frequencies.

Well known examples of collective synchronisation in biology are those of fireflies flashing and crickets chirping in sync. Collective synchronisation also appears within organism. For example in insulin-secreting cells in the pancreas and in pacemaker cells in the heart. Furthermore there are cells in the brain and spinal cord that synchronise in order to control the rhythm of breathing and running. [1]

The interaction between fireflies, crickets or neurons happens through pulses. The insects or cells respond to sudden impulses of their neighbours. This behaviour is difficult to model mathematically. We would like to model continuous behaviour rather than the discontinuous pulses. Thus we consider only the rhythm of an individual or neuron. We might think of this as oscillators with a certain natural frequency. When their periods (or frequencies) coincide, they are said to be in sync.

We will now formally define what we mean by synchronisation. Given a system of N oscillators with phases ϕ_i for $i = 1, \dots, N$. These oscillators are said to synchronise if $\dot{\phi}_i - \dot{\phi}_j \to 0$ as $t \to \infty$ for every $i, j = 1, \dots, N$. Thus when they tend to travel with the same speed as $t \to \infty$. They are said to be exactly synchronised when moreover $\phi_i - \phi_j \to 0$ as $t \to \infty$ for every $i, j = 1, \dots, N$. Thus next to having the same speed, the oscillators will follow the exact same path as $t \to \infty$.

3 Winfree's research on synchronisation

Winfree started his research on large populations of limit-cycle oscillators in 1966. [1] He used computer simulations, mathematical analysis and experiments with electrically coupled neontube oscillators. In his mathematical analysis Winfree considered a large population of interacting limit-cycle oscillators and applied some simplifications in order to analyse the behaviour. Firstly he assumed the coupling between the oscillators was weak and secondly he assumed that the oscillators were nearly identical.

A third simplification made by Winfree was that the natural frequencies are taken from some narrow probability function. Furthermore the oscillators are assumed to be coupled to the collective rhythm of the population. That is, rather than being coupled to each of the other oscillators it is coupled to some average of the frequencies.

4 Introducing the Kuramoto model

Inspired by the works of Winfree, Kuramoto decided to study the phenomenon as well. [3] He expanded on the ideas of Winfree and derived a model that describes the long term dynamics of any system of nearly identical weakly coupled limit-cycle oscillators. In spite of the assumptions made in order to simplify the model, the oscillators described by the Kuramoto model will still synchronise under certain conditions. Therefore the model can and has been used to study collective synchronisation.

The Kuramoto model describes a system of N oscillators that are coupled by means of their phase differences. A general form for the rate of change of the i'th oscillator is given by

$$\dot{\phi}_i = \omega_i + \sum_{j=1}^{N} \Gamma_{i,j}(\phi_j - \phi_i), \tag{1}$$

for $i = 1, \ldots, N$. Here ω_i is the intrinsic frequency of the respective oscillator and $\Gamma_{i,j}$ denotes the interaction function. Kuramoto considered purely sinusoidal coupling between all oscillators. Thus he obtained

$$\dot{\phi}_i = \omega_i + \sum_{j=1}^{N} K_{i,j} \sin(\phi_j - \phi_i), \qquad i = 1, \ldots, N, \tag{2}$$

where $K_{j,l}$ denote the coupling constants. For small coupling constants the model approximately describes a system of independent oscillators moving at frequencies ω_j. Because the model is closely related to this solvable system, it is a useful tool in our study. As we will find out, the coupling constants should be sufficiently great in order for synchronisation to occur.

Different types of coupling may be considered. Such as next-neighbour coupling, mean-field coupling and long-range coupling [6]. We will consider mean-field coupling, each oscillator is coupled to some average frequency. The coupling constants $K_{i,j}$ are identically equal to $\frac{K}{N}$ for all $i, j = 1, \ldots, N$. The model then becomes

$$\dot{\phi}_i = \omega_i + \frac{K}{N} \sum_{j=1}^{N} \sin(\phi_j - \phi_i), \qquad i = 1, \ldots, N. \tag{3}$$

Each oscillator is coupled to the collective rhythm, as was also assumed by Winfree. This is not evident from the equations for the Kuramoto model as introduced above (2). To get a better idea of the model and the dependence between the oscillators we therefore introduce the *order parameter*.

5 Order parameter

Let ϕ_i describe the angle of a point running around the unit circle. The model then describes a collection of N points running around this unit circle. We can define a collective rhythm for this collection of points as

$$re^{i\psi} = \frac{1}{N} \sum_{j=1}^{N} e^{i\phi_j}. \tag{4}$$

Here ψ is the average phase and r measures the phase coherence of the oscillators. The order parameter will have a value located in the unit circle. If the points are spread around the circle, r will be close to zero. If the points are located closer to each other, r will be closer to 1. So values of $re^{i\psi}$ close to zero indicate that absence of synchronisation whereas values close to the unit circle indicate (partial) synchronisation.

We can write equations (3) in terms of the order parameter. First we rewrite (4) by multi-

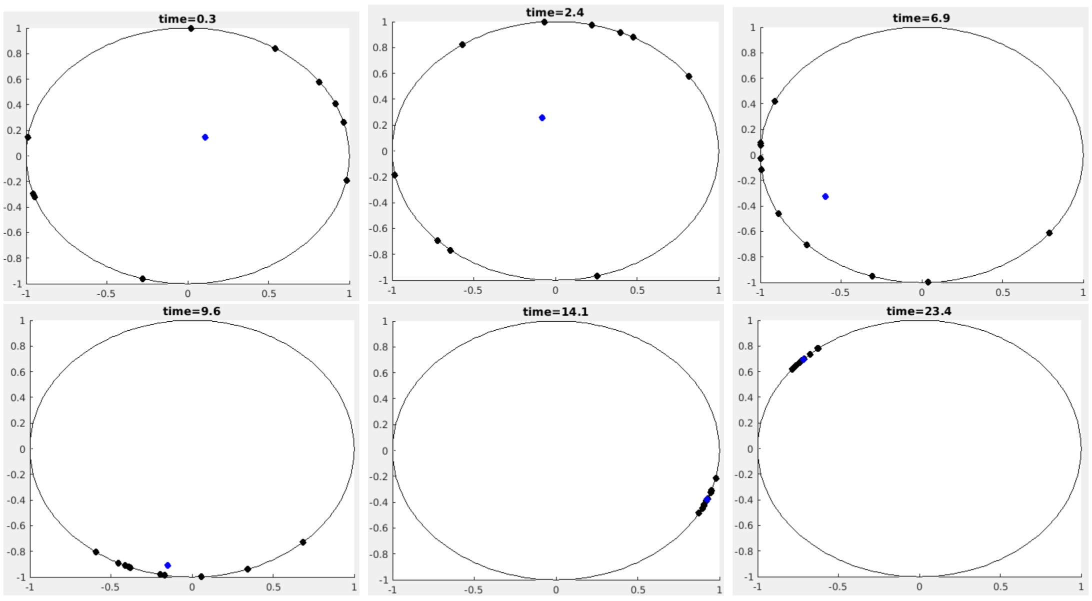


Figure 1: Several frames capturing the paths of 10 oscillators (black) and the corresponding order parameter (blue) are shown. After the 5th frame the oscillators remain together. The natural frequencies $\omega_1, \ldots, \omega_{10}$ are taken from a normal distribution with mean $\mu = 0.3$ and standard deviation $\sigma = 0.05$. The coupling is $K = 0.04$. Animations for different coupling strengths are available [9]

plying both sides by $e^{-i\phi_i}$ to obtain

$$re^{i(\psi-\phi_i)} = \frac{1}{N}\sum_{j=1}^{N} e^{i(\phi_j-\phi_i)}.$$

We then equate imaginary parts:

$$r\sin(\psi-\phi_i) = \frac{1}{N}\sum_{j=1}^{N} \sin(\phi_j-\phi_i). \tag{5}$$

By this result equation (3) can be rewritten to

$$\dot{\phi}_i = \omega_i + Kr\sin(\psi-\phi_i), \qquad i = 1, \ldots, N. \tag{6}$$

Modifying the model using this order parameter leads us to the conclusion that each oscillator is coupled to the others only through the mean-field quantities r and ψ. As the population becomes more coherent, r increases and therefore the effective coupling term Kr increases. This leads to more and more oscillators becoming part of the synchronised group of oscillators. This behaviour was first discovered by Winfree and specifically stands out in the Kuramoto model.

Kuramoto used this order parameter to study the behaviour of the model. We will have a look at his analysis in the next section. In Section 8 we will analyse the order parameter in a different manner. Namely by using time-frequency analysis.

6 Kuramoto's analysis

In this section, we will follow Kuramoto's analysis of the model as described by Strogatz [3]. We have rewritten (2) in terms of the mean-field quantities in the following way:

$$\dot{\phi}_i = \omega_i + Kr\sin(\psi - \phi_i), \qquad \text{for } i = 1, \dots, N. \tag{7}$$

We will now analyse the model following Kuramoto's procedures. In his analysis, Kuramoto sought for particular solutions, namely those in which $r(t)$ is constant and $\psi(t)$ rotates uniformly at some frequency Ω. By then moving into a rotating frame with this frequency Ω we can set $\psi = 0$ to obtain

$$\dot{\phi}_i = \omega_i - Kr\sin(\phi_i), \qquad \text{for } i = 1, \dots, N. \tag{8}$$

This equation has two different types of solutions. One corresponding to the oscillators that are in the synchronised pack, the other corresponds to the oscillators that are not.

Oscillators for which their natural frequency satisfies $|\omega_i| \leq Kr$ have solutions approaching a stable fixed point. This fixed point satisfies $\dot{\phi}_i = 0$ and can therefore be implicitly described by

$$\omega_i = Kr\sin(\phi_i), \qquad \text{where } |\phi_i| \leq \frac{1}{2}\pi. \tag{9}$$

These are the oscillators that are part of the synchronised pack. With respect to the original frame, these oscillators are locked to the frequency Ω. For coupling constants great enough (relative to the natural frequencies), each oscillator will tend to a fixed point. Hence they will all synchronise when we take the limit $K \to \infty$.

However, this is not necessarily the case. Some of the oscillators might have natural frequencies such that $|\omega_i| > Kr$. These will not lock to the frequency Ω. Instead, they will run around the circle in an incoherent manner. As they interact with the other oscillators, they will speed up at some of the phases and slow down at others.

Recall that we are considering solutions such that the order parameter (4) is constant. To ensure that the order parameter is constant even though not all oscillators lock their phase, Kuramoto required the drifting oscillators to form a stationary distribution. He required that oscillators pile up at slow places and thin out at fast places. Suppose the distribution of (infinitely many) oscillators around the circle is given by $\rho(\phi, \omega)$. Then we should have that it is inversely proportional to the speed at ϕ. Hence,

$$\rho(\phi, \omega) = \frac{C}{|\dot{\phi}|} = \frac{C}{|\omega - Kr\sin(\phi)|}. \tag{10}$$

The normalisation constant C is determined to be

$$C = \frac{1}{2\pi}\sqrt{\omega^2 - (Kr)^2}$$

by setting $\int_{-\pi}^{\pi} \rho(\phi, \omega) d\phi = 1$ for each ω.

Since we are in the rotating frame such that $\psi = 0$, we can rewrite the order parameter to $re^{i\psi} = r$. The order parameter should describe the collective rhythm of the oscillators. As we take the limit $N \to \infty$, we denote the order parameter by

$$r = \langle e^{i\phi} \rangle_{\text{lock}} + \langle e^{i\phi} \rangle_{\text{drift}}, \tag{11}$$

where $\langle e^{i\phi} \rangle_{\text{lock}}$ and $\langle e^{i\phi} \rangle_{\text{drift}}$ denote the averages of the oscillators locked to the phase $\psi = 0$ and the drifting oscillators respectively. For a locked state ϕ_{lock}, we have that $\dot{\phi}_{\text{lock}} = 0$ and hence

$$\sin(\phi_{\text{lock}}) = \frac{\omega}{Kr}. \tag{12}$$

We assumed that the distribution of the oscillators satisfies $g(\omega) = g(-\omega)$ in the limit of infinitely many oscillators. Together with (12) this implies that the number of oscillators at ϕ_{lock} is equal to the number of oscillators at $-\phi_{\text{lock}}$ and therefore $\sin(\phi_{\text{lock}}) = 0$. Thus the average of the locked phases is given by

$$\langle e^{i\phi} \rangle_{\text{lock}} = \langle \cos\phi \rangle_{\text{lock}} \tag{13}$$

The drifting oscillators have natural frequencies satisfying $|\omega| > Kr$. Their contribution to the average of the population is given by

$$\begin{aligned}
\langle e^{i\phi} \rangle_{\text{drift}} &= \int_{-\pi}^{\pi} \int_{|\omega|>Kr} e^{i\phi} \rho(\phi, \omega) g(\omega) d\omega d\phi \\
&= \int_{-\pi}^{\pi} \int_{Kr}^{\infty} \Big(e^{i\phi} \rho(\phi, \omega) g(\omega) + e^{i(\phi+\pi)} \rho(\phi + \pi, -\omega) g(-\omega) \Big) d\omega d\phi \\
&= \int_{-\pi}^{\pi} \int_{Kr}^{\infty} \big(e^{i\phi} \rho(\phi, \omega) g(\omega) - e^{i\phi} \rho(\phi, \omega) g(\omega) \big) d\omega d\phi \\
&= 0.
\end{aligned}$$

Here we have used the symmetry $\rho(\phi+\pi,-\omega)=\rho(\phi,\omega)$ implied by (10). Combining this result with (13) we obtain

$$\begin{aligned} r &= \langle\cos\phi\rangle_{\text{lock}} \\ &= \int_{|\omega|\leq Kr}\cos(\phi(\omega))g(\omega)d\omega. \end{aligned}$$

For the locked oscillators, we can use the expression for ω given by (7) to change variables to ϕ.

$$\begin{aligned} r = &\int_{|\phi|\leq\frac{1}{2}\pi}\cos(\phi)g(Kr\sin(\phi))Kr\cos(\phi)d\phi \\ = Kr&\int_{|\phi|\leq\frac{1}{2}\pi}\cos^2(\phi)g(Kr\sin(\phi))d\phi \end{aligned} \tag{14}$$

First of all, this equation has the trivial solution with $r=0$. The distribution of the oscillators is given by $\rho(\phi,\omega)=\frac{C}{|\omega|}=\frac{1}{2\pi}$ for all ϕ and ω. The corresponding state is completely incoherent, it exhibits no kind of synchrony.

The non-trivial solutions of equation (14) satisfies

$$1=K\int_{|\phi|\leq\frac{1}{2}\pi}\cos^2(\phi)g(Kr\sin(\phi))d\phi. \tag{15}$$

From this condition Kuramoto derived an exact formula for the critical coupling K_c. That is the value for the coupling K below which no synchronisation occurs and above which (partial) synchronisation occurs. We obtain this value by letting $r\to0^+$. Then equation (15) becomes

$$\begin{aligned} 1 &= K\int_{|\phi|\leq\frac{1}{2}\pi}\cos^2(\phi)g(0)d\phi \\ &= K\int_{|\phi|\leq\frac{1}{2}\pi}\frac{1}{2}(\cos(2\phi)+1)g(0)d\phi \\ &= K\left[(\frac{1}{3}\sin(2\phi)+\frac{1}{2}\phi)g(0)\right]_{\phi=-\frac{\pi}{2}}^{\phi=\frac{\pi}{2}} \\ &= K\frac{\pi}{2}g(0). \end{aligned}$$

Thus we obtain the critical coupling

$$K_c=\frac{2}{\pi g(0)}. \tag{16}$$

If we consider for example a normal distribution

$$g(\omega, \sigma, \mu) = \frac{1}{\sigma 2\pi} \exp\left(-\frac{(x-\mu)^2}{2\sigma}\right), \tag{17}$$

we obtain the critical coupling constant

$$K_c = 4\sigma \exp\left(\frac{\mu^2}{2\sigma}\right). \tag{18}$$

Later on we will pick the natural frequencies from a normal distribution and compute the critical coupling using equation (18).

7 Hamiltonian system

We will now consider a Hamiltonian function that is closely related to the Kuramoto model. We will follow the approach of Witthaut and Timme [2]. In fact, the Hamiltonian function introduced by Witthaut and Timme will generate the Kuramoto model on an invariant torus. Therefore we study the properties of the Kuramoto model by studying the dynamics of this specific Hamiltonian system.

We will consider the Hamiltonian system for $N = 3$ oscillators. We will see that for this number of oscillators we can obtain a 2-dimensional Poincaré section. For more than three oscillators, this will not be possible any more. To study the Kuramoto model for more oscillators we will introduce a different method.

The Hamiltonian system we consider describes the dynamics of the Kuramoto model on a family of invariant tori. The Hamiltonian function is

$$\begin{aligned}\hat{H}(q_1, p_1, \ldots, q_N, p_N) =& \sum_{l=1}^{N} \frac{\hat{\omega}}{2}(q_l^2 + p_l^2) + \frac{L}{4}(q_l^2 + p_l^2)^2 \\ &+ \frac{1}{4} \sum_{l,m=1}^{N} \hat{K}_{l,m}(q_l p_m - q_m p_l)(q_m^2 + p_m^2 - p_l^2 - p_l^2).\end{aligned} \tag{19}$$

We rewrite (19) in terms of the action-angle variables

$$I_l = (q_l^2 + p_l^2)/2, \tag{20}$$

$$\phi_l = \arctan(q_l/p_l) \tag{21}$$

for $l = 1, \dots, N$. The Hamiltanian is then transformed to

$$H(I_1, \phi_1, \dots, I_N, \phi_N) = \sum_{l=1}^{N} \hat{\omega}_l I_l + L I_l^2 - \sum_{l,m=1}^{N} \hat{K}_{l,m} \sqrt{I_m I_l} (I_m - I_l) \sin(\phi_m - \phi_l). \tag{22}$$

The equations of motion are

$$\dot{I}_j = -\frac{\partial H}{\partial \phi_j} = -2 \sum_{m=1} N \hat{K}_{m,j} \sqrt{I_m I_j} (I_m - I_j) \cos(\phi_m - \phi_j), \tag{23}$$

$$\dot{\phi}_j = \frac{\partial H}{\partial I_j} = \omega_j + L I_j + \sum_{m=1}^{N} \hat{K}_{m,j} \left[2\sqrt{I_j I_m} \sin(\phi_m - \phi_j) - \sqrt{I_m / I_j} (I_m - I_j) \sin(\phi_m - \phi_j) \right]. \tag{24}$$

If all actions are equal at some state, e.g. $I_j = I$ for $j = 1, \dots, N$, then from (23) it follows that $\dot{I}_j = 0$ for $j = 1, \dots, N$. In the case where all actions are equal, (24) can be rewritten to

$$\dot{\phi}_j = \hat{\omega}_j + LI + \sum_{m=1}^{N} \hat{K}_{m,j} 2I \sin(\phi_m - \phi_j), \qquad \text{for } j = 1, \dots, N. \tag{25}$$

Recall that the dynamics of the Kuramoto model with mean-field coupling were described by

$$\dot{\phi}_j = \omega_j + \sum_{m=1}^{N} \frac{K}{N} \sin(\phi_m - \phi_j), \qquad \text{for } j = 1, \dots, N. \tag{26}$$

Thus the Kuramoto dynamics are described on this tori if we take the coupling matrix with $\frac{K}{N} = 2I\hat{K}_{m,j}$ and shift the frequencies such that $\omega_j = \hat{\omega}_j + LI$. Therefore we can study the dynamics of the Kuramoto model by considering the Hamiltonian system given by equations (23) and (24).

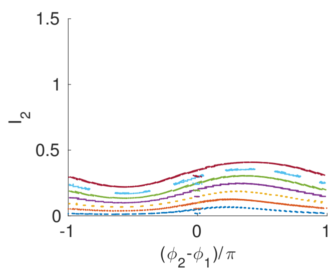
We will consider this system for $N = 3$ which is 6-dimensional. This can be reduced to a 3-dimensional system by applying two constants of motion and a phase shift. One of the constants of motion is of course the Hamiltonian function H. The second constant of motion is introduced by Witthaut and Timme [2] and given by

$$N = 2 \sum_{j=1}^{N} I_j. \tag{27}$$

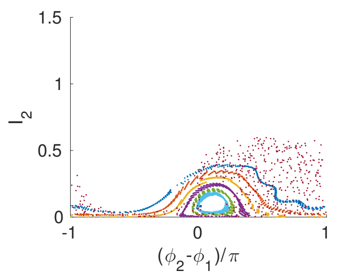
Furthermore, since the dynamics depend only on the phase differences, the dynamics will be invariant under a global phase shift.

We will make a change of actions by multiplying the actions by unimodular matrix M and subsequently make a change of phases by multiplying with its inverse transpose M^{-T}.

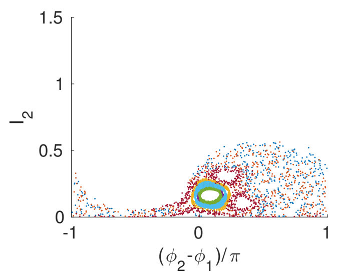
Figure 2: Poincaré surface of sections for $N = 3$, $L = 0$, $\omega_1 = -2$, $\omega_2 = -1$, $\omega_3 = 3$ and Energy$= 3$



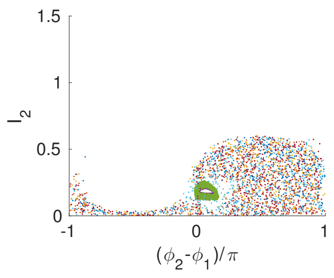
(a) $\frac{K}{N} = 1/3$



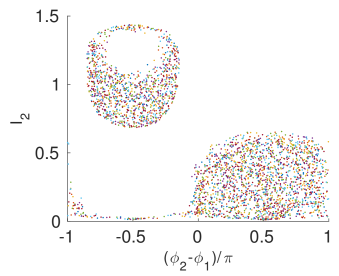
(b) $\frac{K}{N} = 1$



(c) $\frac{K}{N} = 5/3$



(d) $\frac{K}{N} = 2.25$



(e) $\frac{K}{N} = 5$

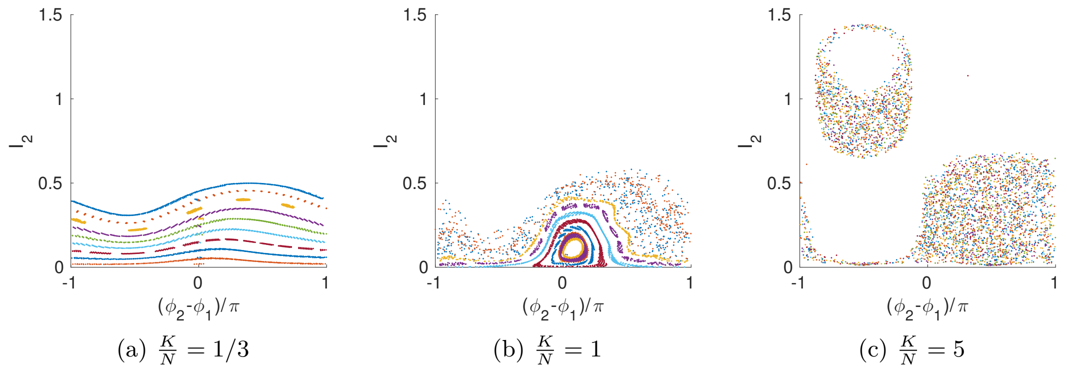
$$\begin{pmatrix} J_1 \\ J_2 \\ J_3 \end{pmatrix} = M \begin{pmatrix} I_1 \\ I_2 \\ I_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} I_1 \\ I_2 \\ I_3 \end{pmatrix} = \begin{pmatrix} I_1 \\ I_2 \\ I_1 + I_2 + I_3 \end{pmatrix}$$

$$\begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix} = M^{-T} \begin{pmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{pmatrix} = \begin{pmatrix} \phi_1 - \phi_3 \\ \phi_2 - \phi_3 \\ \phi_3 \end{pmatrix}$$

For $N = 3$, the Hamiltonian in terms of ϕ_i's and I_i's is given by

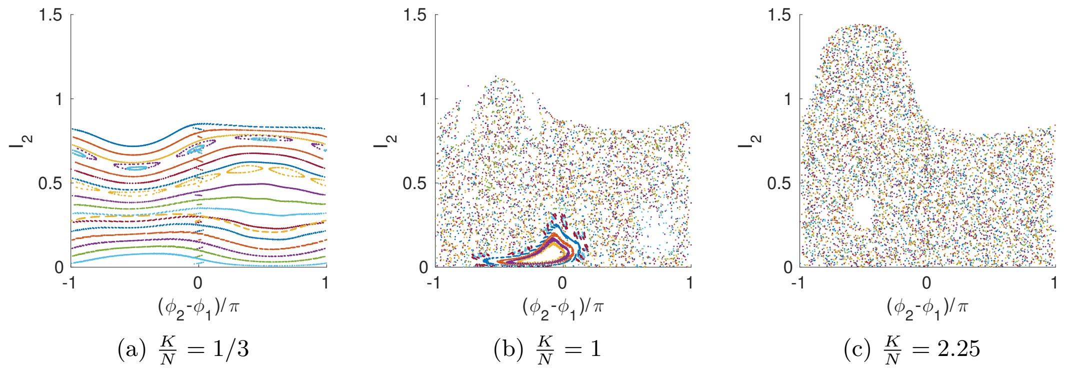
$$\begin{aligned} H(I_1, \phi_1, \ldots, I_3, \phi_3) = {} & \omega_1 I_1 + \omega_2 I_2 + \omega_3 I_3 + L(I_1^2 + I_2^2 + I_3^2) \\ & - 2\frac{K}{N}\sqrt{I_2 I_1}(I_2 - I_1)\sin(\phi_2 - \phi_1) \\ & - 2\frac{K}{N}\sqrt{I_3 I_1}(I_3 - I_1)\sin(\phi_3 - \phi_1) \\ & - 2\frac{K}{N}\sqrt{I_3 I_2}(I_3 - I_2)\sin(\phi_3 - \phi_2) \end{aligned}$$

Figure 3: Poincaré surface of sections for $N = 3$, $L = 0$, $\omega_1 = -2$, $\omega_2 = -1$, $\omega_3 = 3$ and Energy$= 2.5$



(a) $\frac{K}{N} = 1/3$ (b) $\frac{K}{N} = 1$ (c) $\frac{K}{N} = 5$

Figure 4: Poincaré surface of sections for $N = 3$, $L = 0$, $\omega_1 = -2$, $\omega_2 = -1$, $\omega_3 = 3$ and Energy$= 1$



(a) $\frac{K}{N} = 1/3$ (b) $\frac{K}{N} = 1$ (c) $\frac{K}{N} = 2.25$

Expressing ϕ_i's and I_i's in terms of new variables p_i's and J_i's and making use of (27) we obtain

$$\begin{aligned} H(J_1, p_1, J_2, p_2) = {} & \omega_1 J_1 + \omega_2 J_2 + \omega_3(3/2 - J_1 - J_2) + L(J_1^2 + J_2^2 + (3/2 - J_1 - J_2)^2) \\ & - 2\frac{K}{N}\sqrt{J_2 J_1}(J_2 - J_1)\sin(p_2 - p_1) \\ & - 2\frac{K}{N}\sqrt{(3/2 - J_1 - J_2)J_1}(3/2 - 2J_1 - J_2)\sin(-p_1) \\ & - 2\frac{K}{N}\sqrt{(3/2 - J_1 - J_2)J_2}(3/2 - J_1 - 2J_2)\sin(-p_2). \end{aligned} \tag{28}$$

Observe that we are now left with only four dimensions. Thus we can now produce Poincaré surface of sections. We will consider three oscillators with natural frequencies $\omega_1 = -2$, $\omega_2 = -1$ and $\omega_3 = 3$.

Figures (2), (3) and (4) were produced with the C++-program included in the appendix. The equations just obtained are implemented and as the surface of section we take $\phi_3 - \phi_1 = 0$. We then plot I_2 versus $\phi_2 - \phi_1$.

Initial conditions were selected from a grid such that the energy is constant. The reason for this is that we would like to study the dynamical changes when we vary the coupling constant K. In each of the 3 figures we increase the coupling $\frac{K}{N}$ from left to right.

We observe that for small coupling constants (e.g. $\frac{K}{N} = 1/3$) the system exhibits regular behaviour. However, as we increase the coupling K, the Poincaré sections indicate the emerge of chaos. The initial conditions did not change and neither did the total energy. Nonetheless the Poincaré sections indicate chaos for higher coupling (e.g. $\frac{K}{N} = 5$).

From the Poincaré section we were able to draw conclusions about the behaviour of the Hamiltonian systems for $N = 3$. However, this method will not be very useful when we increase the number of oscillators. For the Poincaré section will then be a 3 or more dimensional manifold. Therefore we introduce another method to investigate the Kuramoto model for more than 3 oscillators.

8 Time-frequency analysis

We will introduce time-frequency analysis based on wavelets. This method works especially well for analysing rapidly varying dynamics. It works similarly to the windowed Fourier transform but has some advantages which we will discuss here. In principle the analysis of frequencies is mathematically defined for infinite time signals. The limitations of this type of analysis done on finite time signals have been described by Carmona [4].

In our time-frequency analysis we will analyse the order parameter that was derived earlier, i.e.

$$re^{i\psi} = \frac{1}{N}\sum_{j=1}^{N} e^{i\phi_j}. \tag{29}$$

8.1 Basic Fourier Transform

The basic Fourier transform decomposes a signal, a complex time signal in our case, into its frequencies. The Fourier transform expresses a signal as a sum of sinusoids. This results in a decomposition of frequencies. If a certain frequency is present in a signal, the Fourier representation will have a peak at this frequency. There is no information about the location in time or the duration of this frequency. We only know if a frequency is present in the signal or not.

Thus there is no time related information. However, in our problem we would like to analyse exactly that. We are interested in the occurrence of synchronisation in the system, therefore we need the present frequencies to be linked the time at which they appeared. Furthermore, we might be able to confirm or contradict the appearance of chaos as observed in the Poincaré sections in the previous section.

8.2 Windowed Fourier Transform

The windowed Fourier transform uses the same principles but instead of looking at the signal as a whole, it is split up into different windows. The function is multiplied by some window function that is localized in time. Each window is then analysed separately. Thus obtaining for each window information about the present frequencies. The difficulty in this analysis is choosing the window size. For if the windows are too small, the frequency might not be measured accurately. Yet if the windows are too wide, we obtain less information about the time related to the frequencies. This is known as Heisenberg's uncertainty principle.

8.3 Analysis based on Wavelets

Thus instead of decomposing the signal by using sinusoids, we will use wavelets. The main difference with sinusoids is that wavelets are localised on a finite time interval only. When analysing for a certain frequency this wavelet is stretched or shrunk and thus automatically adjusts its window. The adaptation of wavelets in the method ensures that the length of the window is adjusted according to the frequency. This way rapid variations are better captured. Whereas events that happen on short timescales, e.g. timescales less than the length of the window, would be missed by the original time-frequency method.

The wavelet we will consider in our analysis is the Morlet-Grossmann wavelet with a Gaussian window $g(t) = e^{-t^2/2\sigma}$. We use the function

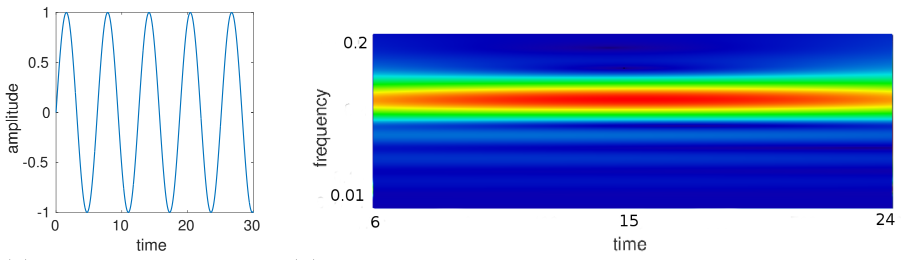
$$\psi_{a,b}(t) = a^{-1/2}\psi\left(\frac{t-b}{a}\right), \tag{30}$$

where the Morlet-Grossmann mother wavelet ψ is given by

$$\psi(t) = \frac{1}{\sigma\sqrt{2\pi}}e^{2\pi i\lambda t}e^{-t^2/2\sigma^2}. \tag{31}$$

The daughter wavelets are then generated by transforming and scaling the mother wavelet. We take $\lambda = 1$ and $\sigma = 2$. Furthermore, $a > 0$ is the scaling factor and $b \in \mathbb{R}$ determines the shift, the wavelet is centred at b. The scaling factor determines how much we stretch or compress the

Figure 5: An example to demonstrate the time-frequency analysis. We analyse the signal $\sin(t)$ for $0 < t < 60$.



(a) Graph of a time signal with increasing frequency. (b) Time-frequency plot of a time signal with increasing frequency.

mother wavelet. The actual wavelet for a signal $f(t)$ transform is given by

$$L_\psi f(a,b) = \langle f, \psi_{a,b} \rangle = a^{-1/2} \int_{-\infty}^{\infty} f(t)\bar{\psi}\left(\frac{t-b}{a}\right) dt. \tag{32}$$

To demonstrate the method we can take for example the signal $f(t) = e^{i2\pi\upsilon t}$. We then obtain the expression for the transform

$$\begin{aligned} L_\psi f(a,b) &= a^{-1/2} \int_{-\infty}^{\infty} e^{i2\pi\upsilon t}\bar{\psi}\left(\frac{t-b}{a}\right) dt & (33)\\ &= a^{1/2} \int_{-\infty}^{\infty} e^{i2\pi\upsilon as} e^{i2\pi\upsilon b}\bar{\psi}(s) ds & (34)\\ &= a^{1/2} e^{i2\pi\upsilon b}\bar{\hat{\psi}}(a\upsilon), & (35)\end{aligned}$$

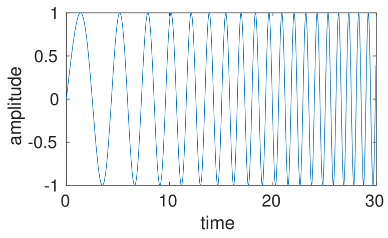
where we denoted the Fourier transform of ψ by $\hat{\psi}$. We will be interested in the modulus of this transform which is given by

$$|L_\psi f(a,b)|^2 = a^{1/2} e^{-2\pi^2\sigma^2(\upsilon a-\lambda)^2}. \tag{36}$$

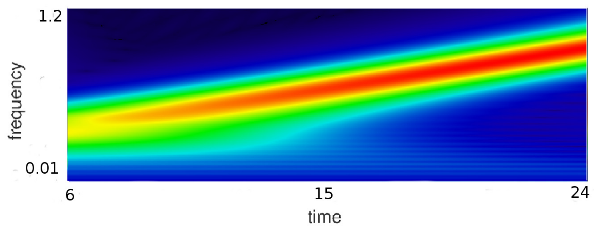
8.4 Time-frequency analysis: some examples

To demonstrate the use of the time-frequency analysis that will be used to analyse the Kuramoto model we first introduce a few examples. Firstly consider the time signal given simply by $\sin(t)$ its graph and time-frequency plot are shown in Figure 5. The highest amplitude frequencies are colour coded in red. The lowest amplitude frequencies are coloured dark blue. In this case we observe a red constant line indicating that the signal has a constant frequency. Now consider the time signal give by $\sin(t + 0.1t^2)$. Its graph is given in Figure 6a. The frequency of the signal increases in time, as can be seen in the time-frequency plot in Figure 6b. The programs used to produce the time-frequency plots can be found in Appendix B.

Figure 6: An example to demonstrate the time-frequency analysis. We analyse the signal $\sin(t + 0.1t^2)$ for $0 < t < 30$.

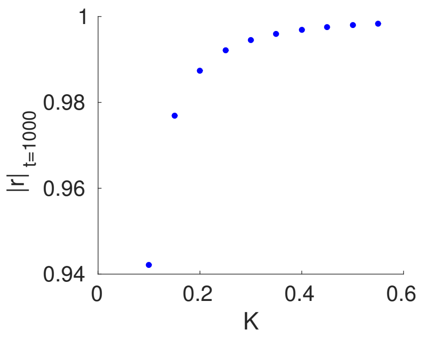


(a) Graph of a time signal with increasing frequency.



(b) Time-frequency plot of a time signal with increasing frequency.

Figure 7: Here we plotted the limiting value of $|r|$, that is $|r|_{t=1000}$ in our case, vs K. We only consider coupling values for which $|r|$ converges. The numbers of oscillators is $n = 10$, natural frequencies $\omega_1, \ldots, \omega_{10}$ taken from a normal distribution with mean $\mu = 0.3$ and standard deviation $\sigma = 0.05$.



8.5 Time-frequency analysis of the order parameter

We consider the Kuramoto model with natural frequencies picked from a normal distribution with mean $\mu = 0.3$ and standard deviation $\sigma = 0.05$. The initial phases are spread uniformly over the interval from 0 to 2π. We then estimate the critical coupling by equation (18).

$$K_c = 4\sigma \exp\left(\frac{\mu^2}{2\sigma}\right) \tag{37}$$

$$= 4 \cdot 0.05 \exp\left(\frac{0.3^2}{2 \cdot 0.05}\right) \tag{38}$$

$$\approx 0.492. \tag{39}$$

This value for the critical coupling was derived by Kuramoto in the limit of infinitely many oscillators. In the time-frequency analysis done here we increase the number of oscillators to 100. We found that for smaller values of the coupling the system already attains synchronisation. For 100 oscillators we have a critical coupling of $K \approx 0.1$. Although we cannot confirm the critical coupling $K_c \approx 0.492$ for exact synchronisation, we expect that as we increase the number of oscillators, the critical coupling K_c will be closer to the value estimated above.

The critical coupling K_c is the lowest value for which the system globally synchronises. For values lower than this K_c the model might show interesting behaviour i.e. separate groups of oscillators may synchronise. We will see that using time-frequency analysis this behaviour can be detected.

Figure 8 shows the results of our time-frequency analysis. The colour plots show the presence of frequencies in the range $0.2/2\pi < \Omega < 0.4/2\pi$. The colourcode is as follows: low amplitude frequencies are shown in dark blue and the highest amplitude frequencies are coloured red. In each of the Figures 9a, 9b and 9c, we have shown a plot of the radius r and the original complex signal of the order parameter for different coupling constants.

What appears to happen is that for small coupling the oscillators do tend to pack together but after some time they spread out again. Only to repeat this later. Increasing the coupling seems to prolong the period of time during which the oscillators pack together. Presumably for values great enough, this pattern is broken and the oscillators stay packed together.

Figure 7 shows a plot of the limiting values of $|r|$ versus the coupling constant K for 10 oscillators. For smaller values of K the system obtains normal synchronisation. That is, their speeds match up, yet the positions of the oscillators on the circle differ. As we increase the coupling strength, we notice that the limiting value of $|r|$ converges to 1. Thus as we increase the coupling constant we approach exact synchronisation in which also the positions of the oscillators match up.

Up until now we have considered oscillators that all synchronise to each other. This happened as a consequence of choosing the oscillators from a narrow probability distribution. The time-frequency analysis confirms what we could also observe by looking at simple graphs of the order parameter. And what was observed from the animations of the oscillators themselves [9].

We will now consider a large group of oscillators that contain oscillators whose natural frequencies are more spread out. We do this to demonstrate the use of time frequency analysis. Instead of taking the natural frequencies from one normal distribution, we take them from two normal distributions. That is, we take the $\omega_1, \ldots \omega_{50}$ from the normal distribution with mean $\mu = 0.3$ and standard deviation $\sigma = 0.05$ and we take $\omega_{51}, \ldots \omega_{100}$ from the normal distribution with mean $\mu = 1$ and standard deviation $\sigma = 0.05$.

For coupling great enough, all oscillators will still synchronise. However, an interesting phenomenon might be observed for small coupling. Namely that of partial synchronisation. Since we have chosen the natural frequencies in a very specific way, we might expect two groups to form and thus two separate partial synchronisations to occur.

This is hardly observable from the graphs of the order parameter as shown in Figure 11a. Our expectations are however confirmed by the time-frequency plot of the signal of the order parameter shown in Figure 10a. We observe that around the values of means of the natural frequencies there appear two beams in the time-frequency plot. This confirms the expected behaviour. As we increase the coupling K from 0.2 to 0.6, we observe that the two highest amplitude frequencies are now located closer together (Figure 10b). Thus we still have that there are two synchronised packs, yet the frequencies to which they are synchronised have changed. If we then increase K to 1.5 we observe that there is only one dominant frequency (Figure 10c). Thus for sufficiently large coupling the oscillators have all synchronised.

If we chose the natural frequencies ourselves, we can of course have calculated two different order parameters in order to observe the synchronisation. These order parameters are shown in Figure 11c for the first 50 oscillators and in Figure 11b for oscillators 51 to 100. Both order parameters tend to values around 0.8 and vary slightly in time around this value. This is due to the interaction between the two packs of oscillators. The synchronisation is thus not exact and not as perfect as the global synchronisation observed earlier.

Consider another collection of oscillators. We now take 30 oscillators, 10 of which have natural frequencies chosen from the narrow normal distribution considered before with mean $\mu_1 = 0.3$ and standard deviation $\sigma_1 = 0.05$. The remaining 20 oscillators have natural frequencies picked from a broad normal distribution with mean $\mu_2 = 1.0$ and standard deviation $\sigma_2 = 0.3$. The time-frequency plots are shown in Figure 12 for varying coupling constant. We observe that the oscillators from the small pack have synchronised even for small coupling constants. Even though the other 20 oscillators are moving with widely varied frequencies. These appear as the light blue scattered pattern at higher frequencies. For sufficiently large coupling constant we again observe that all oscillators synchronise and that the order parameter has one constant frequency. The corresponding order parameters for a coupling of $K = 0.3$ are shown in Figure 13 and confirm what we can conclude from the time-frequency analysis.

The comparison with the two separate order parameters will only work if we have this type of knowledge about the oscillators. Thus if we do not have an idea about which of the oscillators might synchronise with each other, the time-frequency analysis will be of use.

9 Concluding remarks

We have seen that by varying the coupling constant the behaviour of the oscillators vary. They may synchronise completely, partially synchronise or not synchronise whatsoever. Thus the same oscillators may behave differently depending on the coupling strength. This is also the reason that for example neurons can synchronise in different rhythms and manners. The same oscillating cells are responsible for a horses walk, its gallop as well as other paces.[1] Only the coupling strength between the neurons needs to be changed.

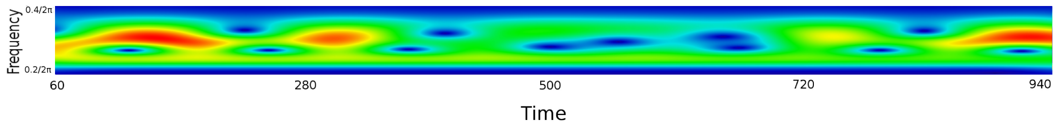
The phenomenon of synchronisation was introduced along with a manner to study it. We followed Kuramoto's analysis and obtained a value for the critical coupling when considering the model has infinitely many oscillators. Then we studied a Hamiltonian system that contains the Kuramoto model on an invariant tori. From the Poincaré sections obtained we concluded that the Hamiltonian system becomes chaotic as the coupling constant is increased.

To study the behaviour of the model for a greater number of oscillators we took another look at the order parameter. The order parameter is a useful tool by itself. Whether or not global synchronisation appears can be determined by analysing the path of the order parameter. Where synchronisation is characterised by the convergence of the order parameter to a circle around the origin. In the case of exact synchronisation the order parameter would converge to the unit circle.

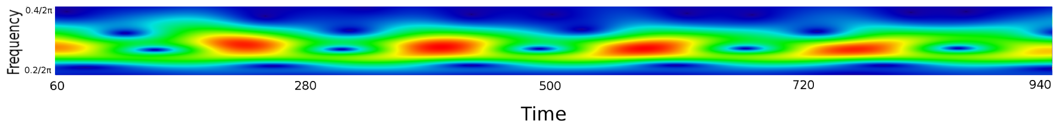
In the case where we do not have global synchronisation but rather two or more clusters of oscillators synchronising separately, this cannot be concluded from simply looking at the order parameter. Instead we used time-frequency analysis to analyse the order parameter in this case. We concluded that using this analysis it is possible to detect global synchronisation as well as the synchronisation of separate clusters of oscillators.

Figure 8: These time-frequency plots correspond to the plots of the order parameter in Figure 9. Frequencies $0.2/2\pi < \Omega < 0.4/2\pi$ vs *time*. For $N = 5$ oscillators, natural frequencies $\omega_1, \ldots, \omega_5$ taken from a normal distribution with mean $\mu = 0.3$ and standard deviation $\sigma = 0.05$.

(a) Coupling constant $K = 0$. Each oscillator moves with its own frequency. The red spots indicate high amplitude frequencies.



(b) Coupling constant $K = 0.05$. The red spots again indicate high amplitude frequencies, the locations of these spots seem more regular than without coupling. The coupling is not strong enough for synchronisation to appear.



(c) Coupling constant $K = 0.1$. The red line indicates that there is one constant high amplitude frequency present on the whole time interval. Thus all oscillators have taken on the same frequency, i.e. they have synchronised.

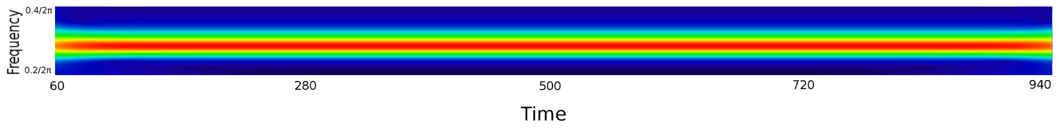
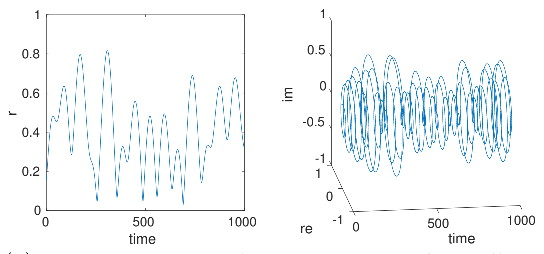
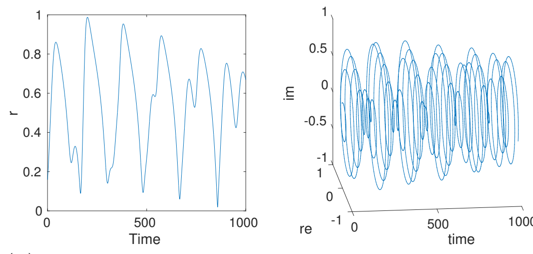


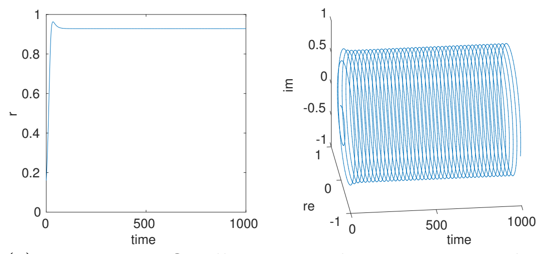
Figure 9: These order parameter plots correspond to the time-frequency plots in Figure 8. Left: plot of $|r|$ vs *time*. Middle: path of $re^{i\psi}$ Right: time-frequency plots for $200 < t < 800$ of the original complex order parameter $re^{i\psi}$. For $N = 5$ oscillators, natural frequencies $\omega_1, \ldots, \omega_5$ taken from a normal distribution with mean $\mu = 0.3$ and standard deviation $\sigma = 0.05$.



(a) $K = 0$. No synchronisation. Each oscillator moves with its own natural frequency.



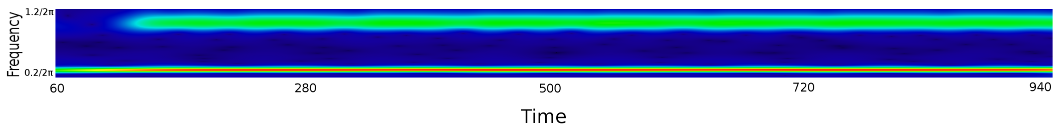
(b) $K = 0.05$. Some interaction between the oscillators. Oscillators appear to synchronise but then spread out again. This repeats itself.



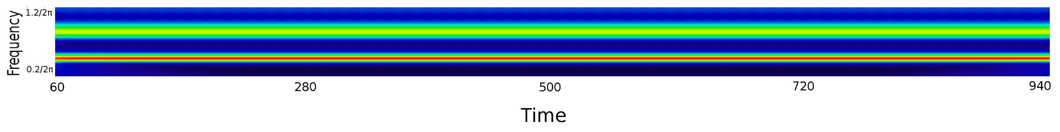
(c) $K = 0.1$. Oscillators synchronise. Note that the order $|r| \neq 1$. The oscillators do synchronise, but their phases differ slightly. As we increase the coupling K, $|r|$ tends to 1.

Figure 10: As we increase the coupling, the highest amplitude frequencies move closer together. For sufficiently large coupling there is only one dominant frequency i.e. all oscillators have synchronised to this frequency. Figures for $K = 0.2$, $K = 0.6$ and $K = 1.5$. $n = 100$. Natural frequencies taken from two different normal distributions with $\mu_1 = 0.3$, $\mu_2 = 1.0$, $\sigma_1 = \sigma_2 = 0.05$. Frequency range: $0.2/2\pi < \Omega < 1.2/2\pi$

(a) Coupling constant $K = 0.2$, corresponding to Figure 11. Two rays of constant frequencies dominate the signal. These are located at the values chosen for the means of the normal distribution, namely $0.3/2\pi$ and $1.0/2\pi$. Oscillators 51 to 100 take more time to synchronise than the others as the top band starts later. This corresponds to the small radius of the order parameter in Figure 11f.



(b) Coupling constant $K = 0.6$, the two bands indicating the highest amplitude frequencies have moved closer together. Thus the two packs that have synchronised separately have now adjusted their frequencies to be closer to each other.



(c) Coupling constant $K = 1.5$, all oscillators have adapted the same frequency. Thus for sufficiently large coupling all oscillators synchronised.

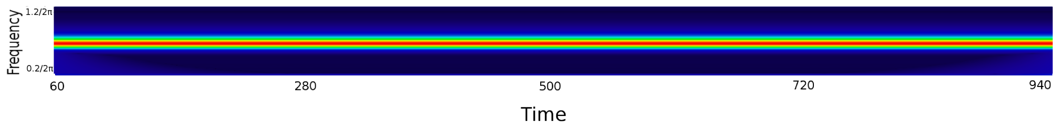
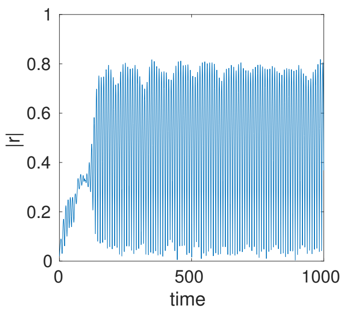
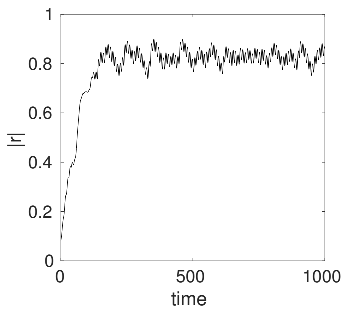


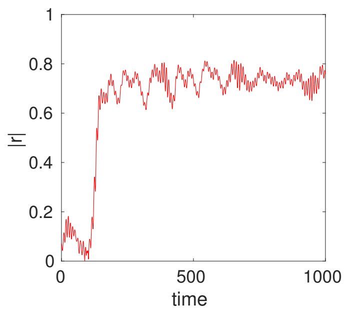
Figure 11: Three different order parameters are plotted for a system of 100 oscillators with coupling $K = 0.2$. Natural frequencies taken from two different normal distributions with $\mu_1 = 0.3$, $\mu_2 = 1.0$, $\sigma_1 = \sigma_2 = 0.05$. First the distance of the order parameter to the origin (Figures 11a, 11b and 11c). Secondly the paths of the complex order parameter are shown in Figures 11d, 11e and 11f. From the order parameter that includes all 100 oscillators no conclusions about synchronisation can be drawn (Figures 11a and 11d) except for the absence of global synchronisation. From Figures 11b and 11e we conclude that the first 50 oscillators have synchronised though the pack is subjected to disturbance of the other oscillators. The same can be concluded from Figures 11c and 11f for oscillators 51 to 100.



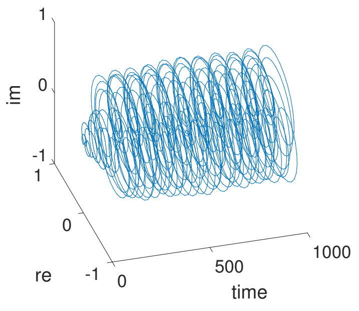
(a) $\omega_1, \ldots, \omega_{100}$



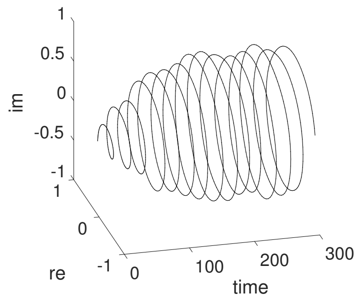
(b) $\omega_1, \ldots, \omega_{50}$



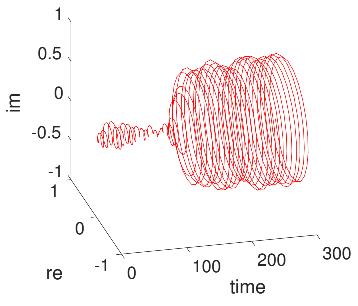
(c) $\omega_{51}, \ldots, \omega_{100}$



(d) $\omega_1, \ldots, \omega_{100}$



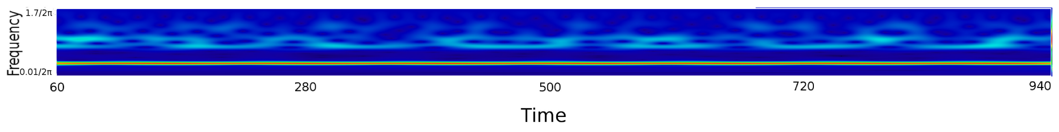
(e) $\omega_1, \ldots, \omega_{50}$



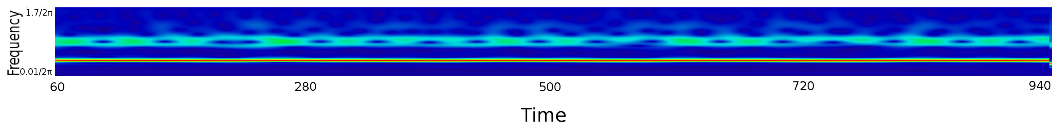
(f) $\omega_{51}, \ldots, \omega_{100}$

Figure 12: Time-frequency plots for a group of $n = 30$ oscillators. Natural frequencies for oscillators 1 to 10 taken from a narrow normal distribution with $\mu_1 = 0.3$ and $\sigma_1 = 0.05$ and natural frequencies for oscillators 11 to 30 taken from a wider normal distribution with $\mu_2 = 1.0$ and $\sigma_2 = 0.3$. Frequency range: $0.01/2\pi < \Omega < 1.7/2\pi$.

(a) Coupling constant $K = 0.3$, corresponding to Figure 13. The highest amplitude frequency is located at approximately 0.3 which is the mean chosen for the normal distribution of oscillators 1 to 10. Furthermore we note the presence of higher frequencies by the scattered light blue area which corresponds to oscillators 11 to 30.



(b) Coupling constant $K = 0.5$, the high amplitude frequency has increased slightly. As the coupling increases it is moving towards the lower amplitude frequencies.



(c) Coupling constant $K = 1.0$. There is one constant frequency present in the signal, this indicates that all oscillators have synchronised for this high coupling constant.

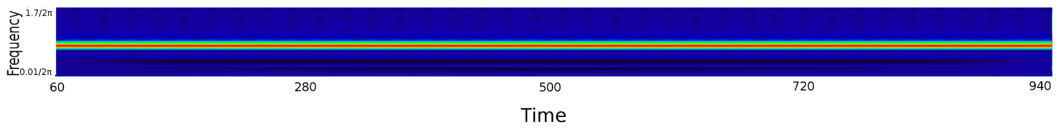
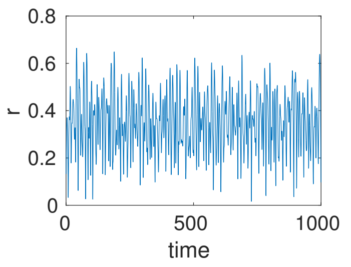
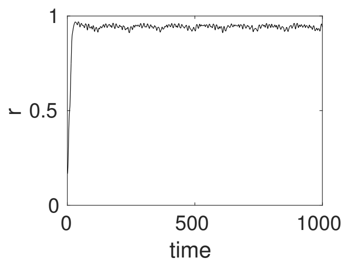


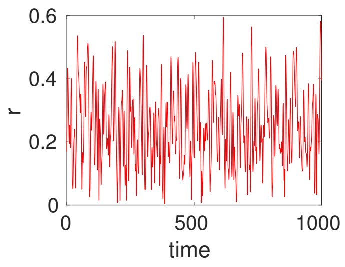
Figure 13: Plots for different order parameters as described before. $K = 0.3$. $n = 30$. Natural frequencies taken from two different normal distributions with $\mu_1 = 0.3$, $\mu_2 = 1.0$, $\sigma_1 = 0.05$, $\sigma_2 = 0.3$. The first 10 oscillators have synchronised yet the other 20 oscillators have not.



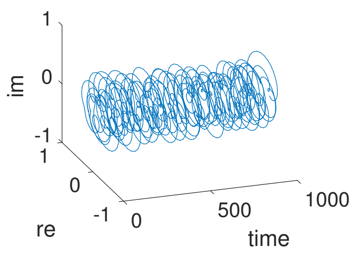
(a) $\omega_1, \ldots, \omega_{30}$



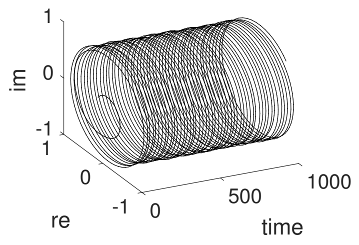
(b) $\omega_1, \ldots, \omega_{10}$



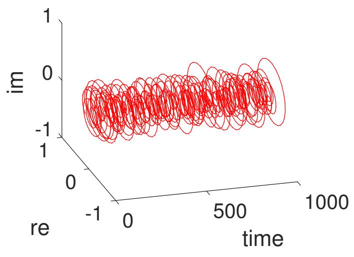
(c) $\omega_{11}, \ldots, \omega_{20}$



(d) $\omega_1, \ldots, \omega_{30}$



(e) $\omega_1, \ldots, \omega_{10}$



(f) $\omega_{11}, \ldots, \omega_{20}$

References

[1] Steven H. Strogatz and Ian Steward. *Coupled Oscillators and Biological Synchronization* Scientific American December, 1993 VOL. 269 NO. 6

[2] Dirk Witthaut, Marc Timme. *Kuramoto dynamics in Hamiltonian systems.* Physical Review E **90** 032917 (2014).

[3] Steven H. Strogatz. *From Kuramoto to Crawford: exploring the onset of synchronization in populations of coupled oscillators.* Physica D 143 (2000) I-20.

[4] René Carmona, Wen-Liang Hwang and Bruno Torrésani. *Practical Time-Frequency Analysis* Academic Press (1998)

[5] William H. Press, William T. Vetterling, Saul A. Teukolsky and Brian P. Flannery. *Numerical Recipes in C++* Cambridge University Press; 2nd edition edition.

[6] Juan. A. Acebron, L. L. Bonilla, Conrad. J. Peréz Vincente, Felix Ritort, Renato Spigler. *The Kuramoto model: a simple paradigm for synchronization phenomena.* Reviews of Modern Physics **77**: 137–185 (2005).

[7] C. Chandre, S. Wiggens, T. Uzer. *Time-frequency analysis of chaotic systems* Elsevier, Physica D 181 (2003) 171-196.

[8] Edward Ott, Thomas M. Antonsen. *Low dimensional behavior of large systems of globally coupled oscillators.* Chaos **18**, 037113 (2008).

[9] Animations of the order parameter for different coupling strengths. `https://drive.google.com/folderview?id=0B61hDr2y5OLHZmNMZnNsbnQ0bG8&usp=sharing`

[10] MATLAB R2015a

A Poincare surfaces

The following C++ program was used to the Poincare surface of sections of the Hamiltonian system in Section 7.

```
#include <iostream>
#include <iomanip>
#include <cmath>
#include "nr.h"
using namespace std;

  double w1=-2;
  double w2=-1;
  double w3=3;
  double L=0;


  double runtime=500;
  double K=1;
  double p1start = 0;

//K=2.25 Energy=3
//double J1start=0.02; double J2start=0.35; double p2start=0;
//double J1start=0.06; double J2start=0.3; double p2start=0;
//double J1start=0.1; double J2start=0.25; double p2start=0;
//double J1start=0.14; double J2start=0.2; double p2start=0;
//double J1start=0.18; double J2start=0.15; double p2start=0;
//double J1start=0.22; double J2start=0.1; double p2start=0;
double J1start=0.26; double J2start=0.05; double p2start=0;


// Driver for routine odeint
DP dxsav;  // defining declarations
int kmax,kount;
Vec_DP *xp_p;
Mat_DP *yp_p;

int nrhs;   // counts function evaluations

void derivs(const DP t, Vec_I_DP &yvector, Vec_O_DP &dydx)
{ double dH_dJ1,dH_dJ2,dH_dp1,dH_dp2;

        nrhs++;

  double J1 = yvector[0];
  double J2 = yvector[1];
  double p1 = yvector[2];
  double p2 = yvector[3];

dH_dJ1 = w1 - w3 + L*(4*J1 + 2*J2 - 3) - 2*K*sin(p1 - p2)*sqrt(J1*J2) - 4*K*sin(p1
    )*sqrt(-J1*(J1 + J2 - 1.5)) - 2*K*sin(p2)*sqrt(-J2*(J1 + J2 - 1.5)) + (K*sin(
    p1)*(2*J1 + J2 - 1.5)*(2*J1 + J2 - 1.5))/sqrt(-J1*(J1 + J2 - 1.5)) + (J2*K*sin
    (p2)*(J1 + 2*J2 - 1.5))/sqrt(-J2*(J1 + J2 - 1.5)) - (J2*K*sin(p1 - p2)*(J1 -
    J2))/sqrt(J1*J2);

dH_dJ2 = w2 - w3 + L*(2*J1 + 4*J2 - 3) + 2*K*sin(p1 - p2)*sqrt(J1*J2) - 2*K*sin(p1
    )*sqrt(-J1*(J1 + J2 - 1.5)) - 4*K*sin(p2)*sqrt(-J2*(J1 + J2 - 1.5)) + (K*sin(
    p2)*(J1 + 2*J2 - 1.5)*(J1 + 2*J2 - 1.5))/sqrt(-J2*(J1 + J2 - 1.5)) + (J1*K*sin
    (p1)*(2*J1 + J2 - 1.5))/sqrt(-J1*(J1 + J2 - 1.5)) - (J1*K*sin(p1 - p2)*(J1 -
    J2))/sqrt(J1*J2);

dH_dp1 = - 2*K*cos(p1 - p2)*(J1 - J2)*sqrt(J1*J2) - 2*K*cos(p1)*sqrt(-J1*(J1 + J2
    - 1.5))*(2*J1 + J2 - 1.5);

dH_dp2 = 2*K*cos(p1 - p2)*(J1 - J2)*sqrt(J1*J2) - 2*K*cos(p2)*sqrt(-J2*(J1 + J2 -
    1.5))*(J1 + 2*J2 - 1.5);

        dydx[0]= -dH_dp1;
  dydx[1]= -dH_dp2;
  dydx[2]= dH_dJ1;
  dydx[3]= dH_dJ2;
}


DP Hamiltonian(Vec_I_DP &yvector)
{ DP Hamiltonianis;
  double J1 = yvector[0];
  double J2 = yvector[1];
  double p1 = yvector[2];
  double p2 = yvector[3];

 Hamiltonianis = J1*w1 + J2*w2 + L*((J1 + J2 - 1.5)*(J1 + J2 - 1.5) + J1*J1 + J2*J2
    ) - w3*(J1 + J2 - 1.5) - 2*K*sin(p1 - p2)*(J1 - J2)*sqrt(J1*J2) - 2*K*sin(p1)*
    sqrt(-J1*(J1 + J2 - 1.5))*(2*J1 + J2 - 1.5) - 2*K*sin(p2)*sqrt(-J2*(J1 + J2 -
    1.5))*(J1 + 2*J2 - 1.5);
  return Hamiltonianis;
}


DP mod_2_PI(DP x)
{ while (x<=-M_PI) {
    x += 2.0*M_PI;
  }while (x>M_PI) {
    x -= 2.0*M_PI;
  }return x;
}
```

```
int main(void)
{ const int N=4, KMAX=100;
        int nbad,nok;
        DP eps=1.0e-10,h1=1E-10,hmin=0.0,x1=0.0,x2=0.0,dx = 0.03;
   DP x1max = runtime;
        Vec_DP ystart(N);
   Vec_DP ystart_old(N);

         ystart[0] = J1start;
         ystart[1] = J2start;
         ystart[2] = p1start;
         ystart[3] = p2start;

  nrhs = 0;
        dxsav=(x2-x1)/2.0;
        kmax=KMAX;
        xp_p=new Vec_DP(KMAX);
        yp_p=new Mat_DP(N,KMAX);


  ofstream outfile_t("trajectory.dat");
  ofstream outfile_E("energy.dat");
  ofstream outfile_s("SOS.dat");
  ofstream outfile_k("values.dat");


  outfile_t << x2 << " " << ystart[0] << " " << ystart[1] << " " << ystart[2] << "
       " << ystart[3]  << endl;
  outfile_E << x2 << " " << setprecision(20) << Hamiltonian(ystart) << endl;

  for(x1=0;x1<x1max;x1+=dx)
  { cout << "time: " << x1 <<" energy: " << Hamiltonian(ystart)<< endl;

    ystart[2] = mod_2_PI(ystart[2]);
    ystart[3] = mod_2_PI(ystart[3]);

    x2 = x1 + dx;

   DP p1old = mod_2_PI(ystart[2]);

    ystart_old[0] = ystart[0];
    ystart_old[1] = ystart[1];
    ystart_old[2] = mod_2_PI(ystart[2]);
    ystart_old[3] = mod_2_PI(ystart[3]);

    outfile_k <<x1 << " " <<ystart[0] <<" "<< ystart[1] << " "<<ystart[2] << " "<<
        ystart[3] <<" " << Hamiltonian(ystart)<< endl;

   NR::odeint(ystart,x1,x2,eps,h1,hmin,nok,nbad,derivs,NR::rkqs);
```

```

    DP p1new=mod_2_PI(ystart[2]);

     ystart[2] = mod_2_PI(ystart[2]);
     ystart[3] = mod_2_PI(ystart[3]);

     if((p1old<0.4*M_PI)&&(p1old>0) && (p1new>-0.4*M_PI)&&(p1new<0)){
     outfile_s << 0.5*(ystart_old[1]+ystart[1])  << " " << mod_2_PI(0.5*(ystart_old
         [3]+ystart[3]) - 0.5*(ystart_old[2]+ystart[2]))<< endl;
     }

         }


  outfile_t.close();
  outfile_E.close();
  outfile_s.close();
  outfile_k.close();


  delete yp_p;
  delete xp_p;
return 0;
}
```

B Time-frequency analysis

To produce the time-frequency plots of Section 8, we used three different programs. Firstly we compute the path of the order parameter, then apply the time-frequency analysis producing a data file to convert this with a third program to an image. The three programs are listed in this order below.

B.1 Computation of the trajectory of the order parameter

```
/* Computes the trajectory of the order parameter of the Kuramoto model.
written in c++ */

#include <iostream>
#include <iomanip>
#include <cmath>
#include "nr.h"

#include <random>
#include <complex>


using namespace std;

  const int n=10;

  double w_array[n];

  double runtime=120;
  double K=0;


double phistart_array[n];



// Driver for routine odeint
DP dxsav;  // defining declarations
int kmax,kount;
Vec_DP *xp_p;
Mat_DP *yp_p;

int nrhs;   // counts function evaluations

void derivs(const DP t, Vec_I_DP &yvector, Vec_O_DP &dydx)
{
  int i=0;
  int k=0;

```

```
          nrhs++;

for(i=0;i<n;i++){
dydx[i] = w_array[i];
  for(k=0;k<n;k++){
  dydx[i]+=K/n*sin(yvector[k]-yvector[i]);

  }
}

}

DP mod_2_PI(DP x)
{ while (x<=0) {
    x += 2.0*M_PI;
  }while (x>2*M_PI) {
    x -= 2.0*M_PI;
  }return x;
}


int main(void)
{ const int N=n, KMAX=100;
        int nbad,nok;
        DP eps=1.0e-7; //original: 1.0e-10
  DP h1=1E-10,hmin=0.0,x1=0.0,x2=0.0,dx = 0.03;
  DP x1max = runtime;
        Vec_DP ystart(N);
  Vec_DP ystart_old(N);
  int i=0;

ofstream outfile_w("omega.dat");
ofstream outfile_p("phases.dat");

  std::default_random_engine generator;
  std::normal_distribution<double> distribution(2.5,1.0);  // normal distribution
      (mean, standarddev)

  for(i=0;i<n;i++){

w_array[i]=distribution(generator);
outfile_w << w_array[i]<<" "<<endl;

}
  std::default_random_engine generator2;
  std::uniform_real_distribution<double> distribution2(0,2*M_PI); // uniform
      distribution [a,b]

for(i=0;i<n;i++){
```

```
ystart[i]=distribution2(generator2);
//ystart[i]=0;

outfile_p << ystart[i]<< " "<<endl;
        }
  outfile_w.close();
  outfile_p.close();


  nrhs = 0;
        dxsav=(x2-x1)/2.0;
        kmax=KMAX;
        xp_p=new Vec_DP(KMAX);
        yp_p=new Mat_DP(N,KMAX);



  ofstream outfile_s("trajectory.dat");
  ofstream outfile_o("order.dat");


  for(x1=0;x1<x1max;x1+=dx)
  { cout << "time: " << x1 << endl;


  NR::odeint(ystart,x1,x2,eps,h1,hmin,nok,nbad,derivs,NR::rkqs);

  for(i=0;i<n;i++){
  ystart[i]=mod_2_PI(ystart[i]);
  }

  outfile_s << x1;

  complex<double> im=-1;
  im = sqrt(im);
for(i=0;i<n;i++){
  outfile_s<< " "<< ystart[i];
}
  outfile_s << endl;


  complex<double> order=exp(im*ystart[0]);
  for(i=1;i<n;i++){
  order += exp(im*ystart[i]);
  }

  double realorder=real(order)/n;
  double imagorder=imag(order)/n;

  outfile_o <<x1<<" "<< realorder <<" "<< imagorder<<" "<<endl;

        }

  outfile_s.close();
  outfile_o.close();
int length=runtime/dx+1;

cout << "length order.dat="<< length << endl;

  delete yp_p;
  delete xp_p;
return 0;
}
```

B.2 Time-frequency analysis

```
/*
   calculates the wavelet transform of a single trajectory of the 3D system.
   written in c++
*/

#include <math.h>
#include <stdio.h>

#define epsilon  0.04


/* parameters for the morlet wavelet */
#define sigma 2.0
#define lambda 1.0

#define ntmax 4000

FILE *out,*out1,*out2,*out3,*in,*out_main,*out_freq;

double morlet_re(double);

double morlet_im(double);

double wave_trans(double, int, double *,double *, double *);

double maxis(double,double);

double minis(double,double);

double theta(double,double,double);


int main(void)
```

```
{

  double eps=1.E-5;  // original: 1.E-14
  double h1=1.E-1,hmin=0,t1,t2,tmin = 0.,tmax = 440000.; /* 2200 */
  int nvar = 6,nok,nbad;
  int nt;

  double omega;

  /* for the wavelet transform*/

  double omega_x0,omega_y0,omega_z0;

  double wave_trans_x,wave_trans_y,wave_trans_z,wave_trans_0,wave_trans_sub,
      wave_trans_top;

  double omegamin = 0.01,omegamax = 1.0,domega; //omegamin = 0.01,omegamax = 1.0,
      domega;

  int nomega,nomegamax = 200;

  int nb,nbmin,nbmax;


   double x_array[ntmax];
   double px_array[ntmax];
   double t_array[ntmax];

   double omega_x0_array[ntmax];


  nbmin = 200;    // margins left & right
  nbmax = ntmax - nbmin;

  domega = (omegamax-omegamin)/nomegamax;

  printf("%f %f %f\n",omegamin,omegamax,domega);

  double xstart[nvar];

  printf("epsilon: %f\n",epsilon);

  /* reads trajectory from order.dat produced by orderpM.cpp */
  FILE*in_order = fopen("order.dat", "r");

  for(int k=0; k<ntmax; k++){
    fscanf(in_order, "%lf %lf %lf", &t_array[k], &x_array[k], &px_array[k] );

  printf("%lf %lf %lf \n", t_array[k], x_array[k], px_array[k]);
  }

  /* output for dat_2_ppm.C */
int l201=nbmin+1;
int l601=ntmax-2*nbmin+1;

    out1 = fopen("out.dat","w");
    fprintf(out1, "%d %d \n", l201, l601);

     for(nb=nbmin;nb<=nbmax;nb++)
     {
              printf("nb=%d\n",nb);

      for(nomega=0;nomega<=nomegamax;nomega++)
      {

   fprintf(out1,"%d %d ",nb,nomega);


   omega_x0 = omegamin +  nomega*(omegamax-omegamin)/nomegamax;


         wave_trans_0      = wave_trans(omega_x0,nb,x_array,px_array,t_array);
   fprintf(out1," %f\n",wave_trans_0);
      }
      }

    fclose(out1);
}


double morlet_re(double t)
{
   return 1./sigma/sqrt(2.*M_PI)*cos((2*M_PI*lambda*t))*exp(-t*t/(2.*sigma*sigma))
       ;
}

double morlet_im(double t)
{
   return 1./sigma/sqrt(2.*M_PI)*sin((2*M_PI*lambda*t))*exp(-t*t/(2.*sigma*sigma))
       ;
}

double maxis(double x,double y)
{
  if(x>=y)
    return x;
  else
    return y;
```

```
}

double minis(double x,double y)
{
  if(x<=y)
    return x;
  else
    return y;
}

double wave_trans(double omega, int nb, double *array_signal, double *
    array_signal_H , double *array_t)
{
   double dt,a,b,tmin,tmax,mean = 0.,trans,trans_re = 0.,trans_im = 0.;
   double *array_si;
   int nt;

    a = (lambda + sqrt( lambda*lambda + 1./(2.*M_PI*M_PI*sigma*sigma) ))/(2.*omega
        );

    b = array_t[nb];

    tmin = b - 3.*sqrt(2)*sigma*a;
    tmax = b + 3.*sqrt(2)*sigma*a;


      /* determine wavelet transform */

      dt = array_t[2] - array_t[1];

      for(nt=(int) maxis(0., tmin/dt -2.) ; nt<= (int) minis(ntmax,tmax/dt + 2.) ;
           nt++)
      {
            trans_re += (
                          array_signal[nt]    * morlet_re((array_t[nt]-b)/a) +
                   array_signal_H[nt] * morlet_im((array_t[nt]-b)/a)
      )/sqrt(a) *dt ;

      trans_im += (
                          array_signal_H[nt] * morlet_re((array_t[nt]-b)/a) -
       array_signal[nt]    * morlet_im((array_t[nt]-b)/a)

                         )/sqrt(a)* dt;

      }

      trans = sqrt(trans_re*trans_re + trans_im*trans_im);

     return trans;
```

```
}


double theta(double q,double p,double h)
{
  double phi,thetais;

  phi = acos(q/pow(4.*h,0.25));

    thetais = M_PI/2.;

  if(q>=0 && p>=0)
    return 2*M_PI-thetais;
  else if(p>0 && q<=0)
     return 2*M_PI-(M_PI-thetais);
  else if(p<0 && q<=0)
     return 2*M_PI-(thetais+M_PI);
  else
     return 2*M_PI-(2.*M_PI-thetais);

}
```

B.3 Converting data to image

```
//written in C
#include <iostream>
#include <iomanip>
#include <cmath>
#include <fstream>

using namespace std;

void hsv2rgb (double h, double s, double v, double *r, double *g, double *b)

{

  double q,p,f,t;

  int i;

   if (s==0){
      *r = *g = *b = v;
   }

  else{
     h = h / 60;
     i = (int)(h);
     f = h - i;
     p = v * (1 - s);
     q = v * (1 - (s * f));
```

```
      t = v * (1 - (s * (1 - f)));

      switch(i){

       case 0:  *r = v; *g = t; *b = p;break;
         case 1:  *r = q; *g = v; *b = p;break;
         case 2:  *r = p; *g = v; *b = t;break;
         case 3:  *r = p; *g = q; *b = v;break;
         case 4:  *r = t; *g = p; *b = v;break;
         case 5:  *r = v; *g = p; *b = q;break;
      }

   }

}

int main(void)

{
   double max_fxy=-1.E-10,min_fxy=1.E10;
   int Nx,Ny;
   double x,y,fxy;

   // find maximum and minimum values
   ifstream infile( "out.dat" );
   if( infile )
    {
       // # grid points in x and y direction
       infile >> Nx >> Ny;
       for( int ny=1 ; ny<=Ny ; ny++ )
          for( int nx=1 ; nx<=Nx ; nx++ )
           {
             infile >> x >> y >> fxy;
             max_fxy =  fxy > max_fxy ? fxy : max_fxy;

             min_fxy =  fxy < min_fxy ? fxy : min_fxy;
           }
    }
   else
    {
      cout << "file could not be opened";
      return (-1);
    }
   infile.close();
   cout << "Nx : " << Nx << " Ny : " << Ny << "\n";
   cout << "Min : " << min_fxy << " Max : " << max_fxy << "\n";
   ifstream infile_2( "out.dat" );
   ofstream outfile( "out.ppm" );
   outfile << "P3\n\n" << Nx << " " << Ny << "\n\n" << "255\n\n" ;
```

```

    if( infile_2 )
     {
        // # grid points in x and y direction
        infile_2 >> Nx >> Ny;
        double r,g,b;
        for( int ny=1 ; ny<=Ny ; ny++ )
           for( int nx=1 ; nx<=Nx ; nx++ )
            {
              infile_2 >> x >> y >> fxy;
              double r,g,b;
              hsv2rgb( (max_fxy-fxy)/(max_fxy-min_fxy)*250,1.0,1.0,&r,&g,&b);
              if(x*x+y*y>1)
              {
                  hsv2rgb( (max_fxy-fxy)/(max_fxy-min_fxy)*250,1.0,1.0,&r,&g,&b);

                  outfile << (int)( pow(fabs((fxy-min_fxy)/(max_fxy-min_fxy)),0.1)*
                        255*r) << " ";

                  outfile << (int)( pow(fabs((fxy-min_fxy)/(max_fxy-min_fxy)),0.1)*
                       255*g) << " ";

                  outfile << (int)( pow(fabs((fxy-min_fxy)/(max_fxy-min_fxy)),0.1)*
                       255*b) << "\n";
              }
              else
              {
                  hsv2rgb( (max_fxy-fxy)/(max_fxy-min_fxy)*250,0.4,1.0,&r,&g,&b);

                  outfile << (int)( pow(fabs((fxy-min_fxy)/(max_fxy-min_fxy)),0.1)*
                        255*r) << " ";

                  outfile << (int)( pow(fabs((fxy-min_fxy)/(max_fxy-min_fxy)),0.1)*
                       255*g) << " ";

                  outfile << (int)( pow(fabs((fxy-min_fxy)/(max_fxy-min_fxy)),0.1)*
                       255*b) << "\n";
              }
            }
     }
    else
     {
       cout << "file could not be opened";
       return (-1);
     }
    infile_2.close();
    outfile.close();
    return 0;
}
```