TU Delft

Faculty of Applied Sciences, BSc Program Applied Mathematics & Applied Physics

Bachelor Thesis

Noise and Synchronization in Kuramoto-type Networks

By:
Jostijn Dessing

Supervisors:
Dr. J.L.A. Dubbeldam,
Dr. A.J.L. Adam

August 2023

TU Delft

Abstract

The Kuramoto model (KM) is a well known mathematical model of coupled oscillators that is frequently used to study synchronization phenomena. In this bachelor thesis we investigate the effects of noise on synchronization in Kuramoto-type networks.

In the first part we follow the methods of Maggi and Paoluzzi [1], but include detailed in between steps, to obtain an analytical expression for the critical coupling strength, k_c, of the KM in the thermodynamic limit under the influence of time-correlated noise (i.e non-white noise). The coupling strength, k, is a parameter in the KM that essentially determines to what extent oscillators influence each other. When $k > k_c$ we start to see synchronization. Our numerical simulations agree with the results found in [1], in that the analytical expression for k_c holds up for low values of correlation time, but quickly breaks down as correlation time increases.

In the second part of this thesis we consider a Kuramoto-type adaptive dynamical network that is also investigated in Fialkowski et al. [2]. The dynamical phenomena that are observed in [2] are also present in our simulations. We explain these dynamical phenomena with the help of a variety of plots. Subsequently, the Kuramoto-type adaptive dynamical network is expanded to include white noise terms in the coupling dynamics. We find, through the use of simulation, that under the same conditions as in [2], synchronization is observed for significantly lower values of coupling strength. This result is explained qualitatively. Simulations also show, that for specific values of noise strength, the hysteric behaviour observed in [2] is not present.

Other conclusions, like the degree to which the noise can reduce the coupling strength required for full synchronization, or beyond what value of noise strength full synchronization can no longer occur, are unable to be drawn. Additional simulation work and a further analytical work is recommended for an ensuing study.

Contents

1 Introduction

Synchronization is a very general concept that plays an important role in everyday life. Often times synchronization is something we strive for; whether it be to improve efficiency, like in assembly lines or scheduling for multi-core processors, or to make an experience more appealing, like in music or in dance.

Some less well known but equally important examples of synchronization can be found in the natural sciences. Synchronization is known to play a major role in neural networks [3] and links have been found between abnormal synchronous behaviour and schizophrenia [4]. Another example is found in chemical oscillators, which are known to exhibit phase locking when their natural frequencies are close to each other [5].

While chemical oscillators and neural networks are inherently different, both have been modelled by the Kuramoto model (KM). The KM is a mathematical model that can describe networks of coupled oscillators [6]. More specifically, it is an example of a static dynamical network; the word 'static' refers to the network topology and the word 'dynamic' refers to the node dynamics (e.g. the phase of an oscillator).

In the past decades the KM has been studied extensively and expanded upon to include, among other things, the effects of noise [1, 7, 8] and temporally changing networks [2]. In this thesis we focus on these adaptations and see how they impact synchronization phenomena.

We examine these two adaptations because of how significant they are in the capability of a model to accurately represent a real system; noise is ubiquitous in almost any physical system, and most real-world networks change over time. Moreover, the Kuramoto-type adaptive dynamical network we will explore also displays interesting dynamical phenomena, other than synchronization, that will briefly be looked into.

This thesis is split up into two main parts. The first part, consisting of Sections 2 and 3, considers the KM on a static fully connected network in the thermodynamic limit. The second part, consisting of Sections 4, 5 and 6, considers the KM on a comparatively small adaptive dynamical network. The effects of noise will be investigated in both parts.

In Section 2 the KM will be explained in more detail and part of Kuramoto's original analysis will be presented. In Section 3 we introduce and derive the main results from [1], which considers the effects of non-white noise on the KM. In Section 4 a very brief overview of adaptive dynamical networks is given. Subsequently, in Section 5, some of the main results from [2], which looks at a Kuramoto-type adaptive dynamical network, are shown and explained through the use of numerical simulation. In Section 6 we expand upon the system investigated in [2] by seeing how dynamical phenomena change under the influence of noise. Finally, the results are discussed in Section 7.

The required background knowledge on topics that are not taught as part of the double bachelor program applied maths and physics at the TU Delft is included in the appendices.

2 Kuramoto Model

The original KM consists of N globally coupled oscillators that are governed by the following set of equations [6, 9]

$$\dot{\phi}_i = \omega_i + \frac{k}{N}\sum_{j=1}^{N}\sin(\phi_j - \phi_i). \tag{1}$$

Here ϕ_i is the phase of the i^{th} oscillator, k is the coupling parameter, and ω_i is the natural frequency of the i^{th} oscillator. Note that in this set of equations only the ϕ_i are time-dependent. The natural frequencies are independent identically distributed random variables, often distributed according to the Gaussian or Lorentzian distributions. For $k > k_c$, where k_c is the critical coupling strength, one starts to observe synchronization despite differing natural frequencies. That is, a (significant) fraction of the oscillators will be phase-locked and will be close to the mean phase. The degree of synchronization is determined by the complex order parameter, [9]

$$re^{i\phi_0} = \frac{1}{N}\sum_{j=1}^{N} e^{i\phi_j}, \tag{2}$$

where $r = r(t)$ is a measure of the degree of synchronization and ϕ_0 is the mean phase. One can readily see that $r = 1$ when all oscillators are in phase. If there is no macroscopic synchronization at all, i.e. the phases are spread perfectly evenly around the unit circle, then $r = 0$. If the phases are random variables, which is the case since the ω_i are random, and they are uniformly distributed around the unit circle (again implying no macroscopic synchronization) then $\langle r\rangle$ is of size $\mathcal{O}(1/\sqrt{N})$. Proofs for both these claims are shown in the appendix.

Since the phases are random variables, by the law of large numbers we have that

$$\frac{1}{N}\sum_{j=1}^{N} e^{i\phi_j} \to \langle e^{i\phi}\rangle,$$

as $N \to \infty$. Thus, in the thermodynamic limit the complex order parameter becomes

$$re^{i\phi_0} = \int_0^{2\pi} n(\phi, t)e^{i\phi}d\phi, \tag{3}$$

where $n(\phi, t)$ is the probability distribution[1] of phases at time t. From Eq. 3 it is clear that for uniformly distributed phases in the thermodynamic limit $r = 0$. This also agrees with our earlier result that $\langle r\rangle$ is of size $\mathcal{O}(1/\sqrt{N})$ in the absence of macroscopic synchronization.

In working with the Kuramoto model it is often easier to consider the natural frequencies to be set. In that case one no longer uses the general probability distribution of the phases, $n(\phi, t)$, but the conditional probability distribution of the phases given the natural frequencies, $n(\phi, t \mid \omega)$. In this case the order parameter may be written in terms of conditional expectations:

$$\begin{aligned} re^{i\phi_0} &= \big\langle\langle e^{i\phi} \mid \omega\rangle\big\rangle \\ &= \int_{-\infty}^{\infty}\int_0^{2\pi} n(\phi, t \mid \omega)e^{i\phi}g(\omega)d\phi d\omega \end{aligned} \tag{4}$$

Here the notation $\langle A \mid B\rangle$ is used for the conditional expectation of A given B and $g(\omega)$ is the distribution of natural frequencies. The second set of angled brackets is there because ω is also random variable.

To make the analysis easier Kuramoto rewrote Eq. 1 using the order parameter. If we multiply Eq. 2 by $e^{-i\phi_i}$ and consider its imaginary part we notice that it's identical to the last term in Eq. 1, except for the coupling parameter k. Thus

$$\dot{\phi}_i = \omega_i + kr\sin(\phi_0 - \phi_i). \tag{5}$$

[1] In this thesis the probability distribution is what is referred to as the probability density in mathematical literature.

At this point it becomes clear that each oscillator is attracted to the mean phase of the system. This phenomenon is known as mean-field coupling.

The basis of Kuramoto's analysis was to find the steady solutions for which r would be constant. This analysis only considered the thermodynamic limit, i.e. $N \to \infty$. Kuramoto's original article can be found here [6] and further details can be found his book [9]. In our derivation for the critical coupling strength, k_c, we use Kuramoto's original analysis along with some of the techniques found in [5].

Upon finding a probability distribution assuming constant r, one can use a self-consistency argument and plug the probability distribution back into Eq. 4 to get a critical value for k, beyond which synchronization will start to occur. To obtain the probability distribution we split the oscillators into two groups; the phase-locked oscillators and the drifting oscillators. When $|\omega_i| \leq kr$ then at some time $\omega_i = kr\sin(\phi_i - \phi_0)$ meaning $\dot{\phi_i} = 0$ and so oscillator i will become locked. If on the other hand $|\omega_i| > kr$ then this cannot occur and oscillator i will drift.

Suppose $|\omega_i| \leq kr$. Then the synchronization condition is $\phi_i - \phi_0 - \sin^{-1}(\omega_i/kr) = 0$. Moreover, stable equilibrium is only reached when $\phi_i - \phi_0 \in (-\pi/2, \pi/2)$.[2] Thus for the locked oscillators we have

$$n(\phi \mid \omega) = \delta\big(\phi - \phi_0 - \sin^{-1}(\omega/kr)\big)H\big(\cos(\phi - \phi_0)\big), \qquad |\omega| \leq kr, \tag{6}$$

where H denotes the Heaviside step function.

Obtaining the distribution function for the locked oscillators requires a bit more work. The key step involves obtaining a partial differential equation (PDE) containing the probability distribution from Eq. 5. Here this is done using a continuity equation, but later on we make use of the Fokker-Planck equation.[3] Denote the dynamic phase velocity of an oscillator at phase ϕ by $v = \omega - kr\sin(\phi - \phi_0)$. For a slice $d\phi$ one has

$$\begin{aligned}
&\text{change \# oscillators in } (\phi, \phi + d\phi) = \text{in} - \text{out},\\
&\frac{\partial}{\partial t}\big(n(\phi \mid \omega)d\phi\big) = (nv)|_{\phi} - (nv)|_{\phi + d\phi},\\
&\frac{\partial n}{\partial t} + \frac{\partial}{\partial \phi}(nv) = 0.
\end{aligned}$$

Thus for stationary solutions (with $|\omega| > kr$) we can see that $nv = C$, i.e.

$$n(\phi \mid \omega) = \frac{C}{|\omega - kr\sin(\phi - \phi_0)|}, \qquad |\omega| > kr, \tag{7}$$

where C is to be determined by the normalization condition, and the absolute value signs appear because one cannot have negative probability.

By using the self-consistency argument and plugging Eqs. 6 and 7 into Eq. 4 we get

$$\begin{aligned}
r = &\int_{|\omega| \leq kr} \int_{-\pi/2 + \phi_0}^{\pi/2 + \phi_0} e^{i(\phi - \phi_0)}\delta\big(\phi - \phi_0 - \sin^{-1}(\omega/kr)\big)g(\omega)d\phi d\omega \quad +\\
&\int_{|\omega| > kr} \int_{0}^{2\pi} e^{i(\phi - \phi_0)} \frac{Cg(\omega)}{|\omega - kr\sin(\phi - \phi_0)|} d\phi d\omega.
\end{aligned} \tag{8}$$

The second term is zero (see appendix). By considering the real part and making the substitution $\phi \to \phi - \phi_0$ one obtains

[2] Suppose ϕ_i is locked, i.e. $\dot{\phi_i} = 0$. If $\phi_i - \phi_0 \in \left(\frac{-\pi}{2}, \frac{\pi}{2}\right)$ then $\dot{\phi_i^{\epsilon}} = \omega_i - kr\sin(\phi_i - \phi_0 + \epsilon) < 0$ meaning ϕ_i is stably locked. If on the other hand $\phi_i - \phi_0 \in \left(\frac{\pi}{2}, \frac{3\pi}{2}\right)$ then $\dot{\phi_i^{\epsilon}} = \omega_i - kr\sin(\phi_i - \phi_0 + \epsilon) > 0$ meaning ϕ_i is not stably locked.

[3] The Fokker-Planck equation is a PDE describing how the probability distribution of a stochastic variable changes with time. See appendix for more details.

$$
\begin{aligned}
r &= \int_{|\omega|\leq kr} \int_{-\pi/2}^{\pi/2} \cos(\phi)\delta\big(\phi - \sin^{-1}(\omega/kr)\big)g(\omega)d\phi d\omega \\
&= \int_{|\omega|\leq kr} \cos\big(\sin^{-1}(\omega/kr)\big)g(\omega)d\omega \\
&= \int_{-\pi/2}^{\pi/2} \cos(\phi)g\big(kr\sin(\phi)\big)kr\cos(\phi)d\phi \\
&= kr\int_{-\pi/2}^{\pi/2} \cos^2(\phi)g\big(kr\sin(\phi)\big)d\phi. \qquad (9)
\end{aligned}
$$

In the third step the substitution $\omega = kr\sin(\phi)$ is made. Clearly $r = 0$ is a solutions of Eq. 9. Suppose that $r > 0$. Then we may write Eq. 9 as

$$
1 = k\int_{-\pi/2}^{\pi/2} \cos^2(\phi)g\big(kr\sin(\phi)\big)d\phi.
$$

Now a critical value for k is obtained by letting r go to 0 from above (i.e. taking the limit $r \to 0^+$). This then gives

$$
k_c = \frac{2}{\pi g(0)}. \qquad (10)
$$

A typical plot of the order parameter against coupling strength in the thermodynamic limit ($N = 10^4$) is shown in Fig. 1. Here the KM is simulated using the forward Euler method with $\Delta t = 0.001$. For each value of k the system is simulated for $5 \cdot 10^4$ time steps and the corresponding value for r is calculated using Eq. 2 and by averaging over the last $2.5 \cdot 10^4$ time steps. The coupling strength, k, is increased from 0.5 to 4.5 in 41 steps.

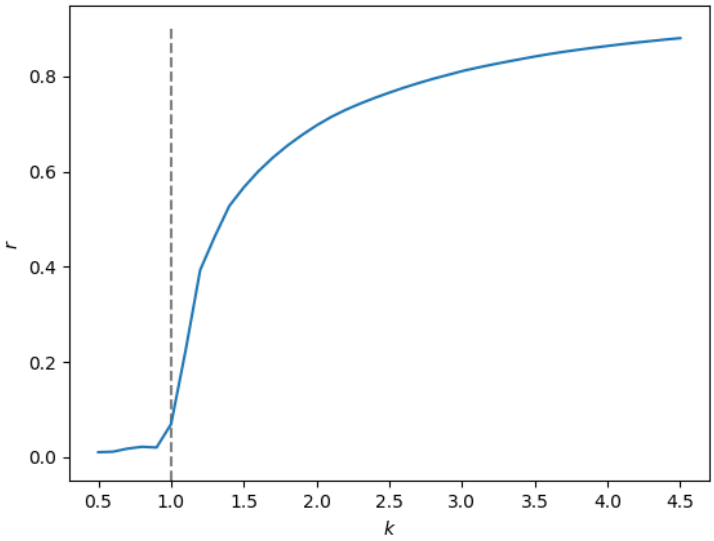


Figure 1: Plot of r against k for regular KM given by Eq. 1. The Lorentz distribution with $\lambda = 0.5$ was chosen for the natural frequencies, and $N = 10^4$. The dashed line indicates the theoretical value for critical coupling strength, k_c, as predicted by Eq. 10.

3 Noise in the Original Kuramoto Model

3.1 White Noise

The presence of noise is nearly unavoidable in any physical system. It may not always play an important role, but there are many situations in which it does. Hence it is of significant interest to see how noise impacts the Kuramoto model.

In his 1988 paper [7], Sakaguchi looked at the classic KM under the influence of external fields. He looked at models of the form

$$\dot{\phi}_i = \omega_i + \frac{k}{N}\sum_{j=1}^{N}\sin(\phi_j - \phi_i) + f_i,$$

where $f_i = f_i(t)$ is an external force. In the latter half of his paper he took $f_i = \sqrt{D}\xi_i$ with ξ_i a set of uncorrelated white noises:

$$\langle \xi_i(t)\rangle = 0, \qquad \langle \xi_i(t)\xi_j(t')\rangle = 2\delta_{ij}\delta(t'-t). \tag{11}$$

In this case the critical coupling strength can be written as

$$\begin{aligned} k_c &= 2\left[\int_{-\infty}^{\infty}\frac{D}{D^2+\omega^2}g(\omega)d\omega\right]^{-1}, \\ &= 2\left[\int_{-\infty}^{\infty}\frac{1}{\omega^2+1}g(D\omega)d\omega\right]^{-1}, \end{aligned} \tag{12}$$

which clearly reduces to Eq. 10 in the case $D = 0$. We omit the derivation because the derivation in the subsequent section, which we include in full detail, is quite similar.

3.2 Coloured Noise

Though Eq. 12 is a nice result, it may only ever *approximate* real systems, for any physical noise source will have nonzero correlation time. Therefore we consider

$$\dot{\phi}_i = \omega_i + \frac{k}{N}\sum_{j=1}^{N}\sin(\phi_j - \phi_i) + \eta_i, \tag{13}$$

with the η_i defined by the stochastic differential equation (SDE)

$$\dot{\eta_i} = \frac{-\eta_i}{\tau} + \frac{\sqrt{D}}{\tau}\xi_i, \tag{14}$$

where the ξ_i are white noise sources defined in the same way as before (Eq. 11). Now the noise term, η_i, has correlation function [10, 11]

$$\langle \eta_i(t)\eta_j(t')\rangle = \frac{D}{\tau}e^{-|t'-t|/\tau}.$$

So the correlation timescale is τ. In this section we will get an analytical expression for the critical value of k in the limit $N \to \infty$. We closely follow the derivation in [1] (which itself closely follows [7]) but include detailed in-between steps. The derivation consists of 3 main steps:

1. Find the Fokker-Planck equation (FPE) for the probability distribution of ϕ, $n(\phi \mid \omega)$.
2. Solve the FPE.
3. Substitute the found probability distribution in Eq. 4 to get a critical value for k.

At this point you may ask: what is the Fokker-Planck equation? For that we refer you to Appendix B.

To begin we rewrite Eq. 13 in the same way as before using the order parameter

$$\dot{\psi}_i = F_i(\psi_i) + \eta_i, \tag{15}$$

where $\psi_i \equiv \phi_i - \phi_0$ is the 'distance' of phase i to the mean phase, and $F_i(\psi) \equiv \omega_i - kr\sin(\psi)$ is the mean-field force. Since all oscillators and noise terms are described by identical equations (except for differing natural frequencies) we could at this point write down the 2-dimensional FPE for the probability distribution $n(\psi, \eta, t \mid \omega)$. But solving this higher dimensional FPE is quite difficult so we take a few more steps in order to reduce the problem to solving a one-variable FPE. This is also where the current derivation differs most from the one done by Sakaguchi, for he could write down the FPE immediately without any further steps.

Differentiating Eq. 15 with respect to time and using Eq. 14 along with $\eta_i = \dot{\psi}_i - F_i(\psi_i)$ gives

$$\ddot{\psi}_i + \Big[\frac{1}{\tau} - F_i'(\psi_i)\Big]\dot{\psi}_i - \frac{1}{\tau}F_i(\psi_i) = \frac{\sqrt{D}}{\tau}\xi_i.$$

All terms except for the first contain a factor $\frac{1}{\tau}$. Thus it makes sense to assume that for small values of τ the first term will contribute little. So in the small-τ regime we get

$$\dot{\psi}_i = \Big[1 - \tau F_i'(\psi_i)\Big]^{-1} F_i(\psi_i) + \Big[1 - \tau F_i'(\psi_i)\Big]^{-1}\sqrt{D}\xi_i. \tag{16}$$

From Eq. 16 we can obtain the FPE for one variable, giving us a PDE for $n(\psi, t \mid \omega)$. Note that the index i will be dropped because all oscillators satisfy identical equations except for their unique frequencies. The frequency information is preserved because the probability distribution *given* ω will be found.

Now the FPE will be written down using the notation from Appendix B. Let

$$\begin{aligned} D^{(1)}(\psi) &= \Big[1 - \tau F'(\psi)\Big]^{-1} F(\psi) + \sqrt{D}\Big[1 - \tau F'(\psi_i)\Big]^{-1}\frac{d}{d\psi}\Big\{\sqrt{D}\Big[1 - \tau F'(\psi_i)\Big]^{-1}\Big\}, \\ &= \frac{Dkr\tau\sin(\psi) + \big[\omega - kr\sin(\psi)\big]\big[1 + kr\tau\cos(\psi)\big]^2}{\big[1 + kr\tau\cos(\psi)\big]^3}, \end{aligned}$$

and

$$\begin{aligned} D^{(2)}(\psi) &= \Big\{\sqrt{D}\Big[1 - \tau F'(\psi_i)\Big]^{-1}\Big\}^2, \\ &= \frac{D}{\big[1 + kr\tau\cos(\psi)\big]^2}. \end{aligned}$$

Then the FPE for the system described by Eq. 16 reads

$$\frac{\partial}{\partial t}n(\psi, t \mid \omega) = \Big[-\frac{\partial}{\partial\psi}D^{(1)}(\psi) + \frac{\partial^2}{\partial\psi^2}D^{(2)}(\psi)\Big]n(\psi, t \mid \omega).$$

This equation is still not easy to solve but since we will look for solutions with constant r (like in Section 2), we should assume that $n(\psi, t \mid \omega)$ is constant in time. This means the method shown in Appendix B.1 can be used.

Define

$$\begin{aligned} A(\psi, \omega) &\equiv \frac{kr\big[kr\tau\cos(2\psi) + 4\tau\omega\sin(\psi) + 4\cos(\psi)\big] + 4\psi\omega}{4D}, \\ B(\psi) &\equiv 1 + kr\tau\cos(\psi). \end{aligned}$$

Then

$$\frac{D^{(1)}(\psi)}{D^{(2)}(\psi)} = \frac{Dkr\tau\sin(\psi) + \big[\omega - kr\sin(\psi)\big]\big[1 + kr\tau\cos(\psi)\big]^2}{D\big[1 + kr\tau\cos(\psi)\big]},$$
$$= \frac{kr\tau\sin(\psi)}{1 + kr\tau\cos(\psi)} + \frac{kr\big[-2kr\tau\sin(2\psi) + 4\tau\omega\cos(\psi) - 4\sin(\psi)\big] + 4\omega}{4D},$$
$$= \frac{-B'(\psi)}{B(\psi)} + \frac{\partial}{\partial\psi}A(\psi,\omega).$$

Now we can define

$$\Phi(\psi,\omega) \equiv \ln D^{(2)}(\psi) - \int_0^\psi \frac{D^{(1)}(\psi')}{D^{(2)}(\psi')}dx',$$
$$= \ln(D) - 2\ln|B(\psi)| + \int_0^\psi \frac{\partial}{\partial\psi'}\Big[\ln|B(\psi')|\Big]d\psi' - \int_0^\psi \frac{\partial}{\partial\psi'}\Big[A(\psi',\omega)\Big]d\psi',$$
$$= \ln(D) - \ln|B(\psi)| - A(\psi,\omega).$$

Note that the two constant terms that result from the two integrals are not written down. This can be done because those constants get absorbed into N' and S' in the next step. The stationary probability distribution, $n_s(\psi \mid \omega)$, becomes

$$n_s(\psi \mid \omega) = N'e^{-\Phi(\psi,\omega)} - S'e^{-\Phi(\psi,\omega)}\int_0^\psi \frac{\big[B(\psi')\big]^2}{D}e^{\Phi(\psi',\omega)}d\psi',$$
$$= N|B(\psi)|e^{A(\psi,\omega)} - S|B(\psi)|e^{A(\psi,\omega)}\int_0^\psi |B(\psi')|e^{-A(\psi',\omega)}d\psi'. \tag{17}$$

In order to find N in terms of S a periodic boundary condition is imposed; $n(0 \mid \omega) = n(2\pi \mid \omega)$. This leads to

$$N|B(0)|e^{A(0,\omega)} = N|B(2\pi)|e^{A(2\pi,\omega)} - S|B(2\pi)|e^{A(2\pi,\omega)}\int_0^{2\pi}|B(\psi')|e^{-A(\psi',\omega)}d\psi',$$
$$\big[e^{A(0,\omega)} - e^{A(2\pi,\omega)}\big]N = -e^{A(2\pi,\omega)}S\int_0^{2\pi}|B(\psi')|e^{-A(\psi',\omega)}d\psi',$$
$$\big[e^{A(2\pi,\omega)-A(2\pi,\omega)} - e^{A(2\pi,\omega)-A(0,\omega)}\big]N = S\int_0^{2\pi}|B(\psi')|e^{-A(\psi',\omega)}d\psi',$$
$$N = \frac{S}{1 - e^{-2\pi\omega/D}}\int_0^{2\pi}|B(\psi')|e^{-A(\psi',\omega)}d\psi'.$$

Finally, substituting this back into Eq. 17 results in

$$n_s(\psi \mid \omega) = S|A(\psi)|e^{B(\psi,\omega)}\left[\frac{e^{2\pi\omega/D}}{e^{2\pi\omega/D} - 1}\int_0^{2\pi}|B(\psi')|e^{-A(\psi',\omega)}d\psi' - \int_0^\psi |B(\psi')|e^{-A(\psi',\omega)}d\psi'\right],$$
$$= S|A(\psi)|e^{B(\psi,\omega)}\left\{\frac{e^{2\pi\omega/D}}{e^{2\pi\omega/D} - 1}\left[\int_0^\psi |B(\psi')|e^{-A(\psi',\omega)}d\psi' + \int_\psi^{2\pi}|B(\psi')|e^{-A(\psi',\omega)}d\psi'\right]\right.$$
$$\left. - \left[\frac{e^{2\pi\omega/D}}{e^{2\pi\omega/D} - 1}\int_0^\psi |B(\psi')|e^{-A(\psi',\omega)}d\psi' - \frac{1}{e^{2\pi\omega/D} - 1}\int_0^\psi |B(\psi')|e^{-A(\psi',\omega)}d\psi'\right]\right\},$$
$$= S|A(\psi)|e^{B(\psi,\omega)}\left[\frac{-e^{2\pi\omega/D}}{1 - e^{2\pi\omega/D}}\int_\psi^{2\pi}|B(\psi')|e^{-A(\psi',\omega)}d\psi' - \frac{1}{1 - e^{2\pi\omega/D}}\int_0^\psi |B(\psi')|e^{-A(\psi',\omega)}d\psi'\right],$$
$$= S\frac{|A(\psi)|e^{B(\psi,\omega)}}{1 - e^{2\pi\omega/D}}\left[\int_\psi^0 |B(\psi')|e^{-A(\psi',\omega)}d\psi' + e^{2\pi\omega/D}\int_{2\pi}^\psi |B(\psi')|e^{-A(\psi',\omega)}d\psi'\right],$$

where S is determined by the normalization condition. Now, like in Eq. 8, we substitute this expression for $n_s(\psi \mid \omega)$ in Eq. 4 (and use $\psi = \phi - \phi_0$) to get

$$r = \int_{\infty}^{\infty} \int_{0}^{2\pi} n_s(\psi, t \mid \omega) e^{i\psi} g(\omega) d\psi d\omega. \tag{18}$$

It is probably impossible to get an explicit expression for r using Eq. 18. Therefore we take a series expansion of the RHS of Eq. 18 in r. Although this is technically not difficult, it is quite cumbersome due to the number of terms involved. Hence a Mathematica script is used (see Appendix D.1). This results in

$$r - kr \int_{-\infty}^{\infty} g(\omega) Q(\omega) d\omega + k^3 r^3 \int_{-\infty}^{\infty} g(\omega) P(\omega) d\omega = 0, \tag{19}$$

where $\mathcal{O}(r^4)$ terms are ignored and

$$\begin{aligned} Q(\omega) &= \frac{1}{2}\left(\frac{D}{D^2+\omega^2} + \tau\right), \\ P(\omega) &= \frac{2D^3 - 4D\omega^2 + \tau(6D^5 + 6D^3\omega^2) + \tau(4D^4 + 5D^2\omega^2 + \omega^4)}{8(D^2+\omega^2)^2(4D^2+\omega^2)}. \end{aligned}$$

From Eq. 19 it can be seen that two additional real roots appear (other than $r = 0$) when

$$-4\left[1 - k\int_{-\infty}^{\infty} g(\omega) Q(\omega) d\omega\right]\left[k^3 \int_{-\infty}^{\infty} g(\omega) P(\omega) d\omega\right] > 0.$$

Thus a critical value for k is obtained by

$$\begin{aligned} k_c &= \frac{1}{\int_{-\infty}^{\infty} g(\omega) Q(\omega) d\omega} \\ &= 2\left[\tau + \int_{-\infty}^{\infty} \frac{D}{D^2+\omega^2} g(\omega) d\omega\right]^{-1}, \end{aligned} \tag{20}$$

which clearly reduces to Eq. 12 in the case $\tau = 0$. This yields the expected result that an increase in correlation time yields a decrease in critical coupling strength.

3.2.1 Numerical Simulation Coloured Noise

To simulate this system, Langevin Eqs. 13 and 14 can be integrated numerically. At each time step Eq. 13 is simulated using the forward Euler method and Eq. 14 is simulated using the Euler-Maruyama method (see Appendix C.1). Moreover, the identity

$$\begin{aligned} \sum_{j=1}^{N} \sin(\phi_j - \phi_i) &= \sum_{j=1}^{N} \big[\sin(\phi_j)\cos(\phi_i) - \cos(\phi_j)\sin(\phi_i)\big] \\ &= \cos(\phi_i) \sum_{j=1}^{N} \sin(\phi_j) - \sin(\phi_i) \sum_{j=1}^{N} \cos(\phi_j), \end{aligned}$$

can be used to significantly speed up computations because it means one only has to calculate these large sums twice per time step rather than N times per time step.

For all of the following simulations the parameters are: $\Delta t = 0.001$, $N = 5000$[4], 50000 time steps per k-value, initial conditions for ϕ_i: drawn from $U(0, 2\pi)$ distribution, initial conditions for η_i: drawn from $N(0, \sqrt{2\Delta t})$ distribution. The frequency distribution is the Lorentz distribution with $\lambda = 0.5$.

In Fig. 2 one can see the result of simulating Eq. 13 and 14 with $D = 1$ for different values of k and τ. It is clear that k_c decreases as τ increases, as Eq. 20 predicts. This does not however confirm its validity.

[4]This should ensure that the system resembles a system in the thermodynamic limit.

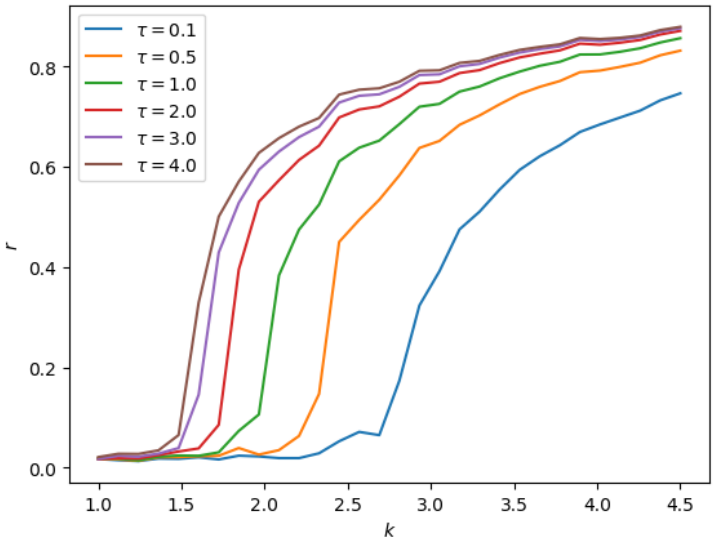
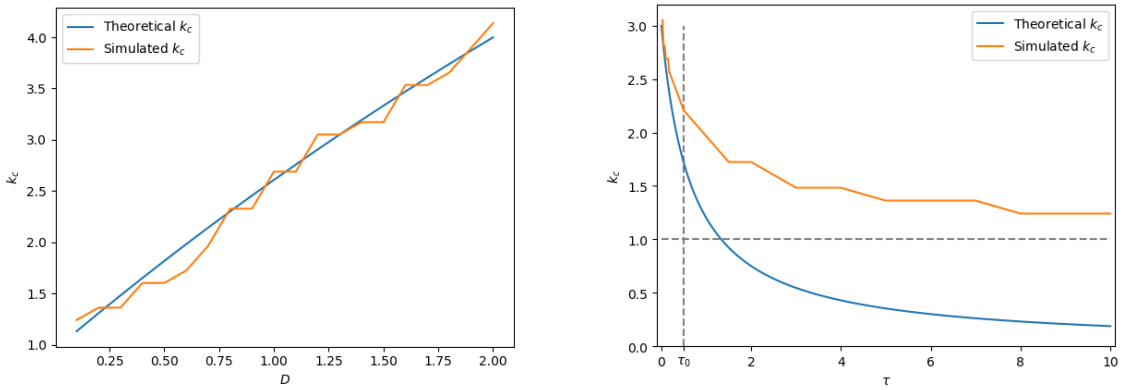


Figure 2: Plot of k against r for different values of correlation time τ. Simulation conditions: $N = 5000$, $\Delta t = 0.001$, 50 000 time steps, random initial conditions. The coupling strength, k, was increased from 1 to 4.5 in 30 steps.

To see where Eq. 20 is valid, plots of simulated k_c along with theoretical k_c are made in Fig. 3. From Fig. 3a it is clear that for a small correlation timescale ($\tau = 0.1$) the theory is accurate for a wide range of values of noise strength D. However, as can be seen in Fig. 3b, Eq. 13 quickly breaks down for higher values of τ. This is in agreement with the results found in [1]. At $\tau_0 = 0.5, D = 1$ (shown as the dotted line in Fig. 3b) the difference between the numerical en theoretical values is already 0.31.



(a) $k_c(D, 0.1)$ **(b)** $k_c(1, \tau)$

Figure 3: Plots of theoretical value of k_c along with simulated value of k_c. This is done for k_c as a function of D (with $\tau = 0.1$) and for k_c as a function of τ (with $D = 1$).

To get a total region of validity one would need to make a 2d plot of $k_{c,\text{simulated}} - k_{c,\text{theoretical}}$ as a function of D and τ. However since a single 'upsweep' of k values can take up to 30min, a simple 10 by 10 grid would take a long time already. Of course simulations could be sped up by starting them at roughly the correct values for critical coupling. Exploring how critical τ[5] varies with noise strength, D, could be done in a few days if deemed interesting.

For more numerical simulations of Eqs. 13 and 14 we refer the reader to [8].

[5]Here 'critical τ' means that if $\tau < \tau_{\text{critical}}$ then $k_{c,\text{simulated}} - k_{c,\text{theoretical}}$ is small enough.

4 Adaptive Dynamical Networks

At this point it is worthwhile to go over some network terminology. So far we have looked at (continuous time) dynamical networks; networks where the state of node i is dynamic. Most often dynamical networks with N nodes are written as

$$\dot{x_i} = f_i(x_i, t) + \sum_{j=1}^{N} \kappa_{ij} g_{ij}(x_i, x_j, t), \tag{21}$$

where f_i is now a generic function of x_i and t, and is no longer defined as in Section 3.1, and g_{ij} is a generic function of x_i, x_j and t. In our case we looked at a fully connected network with $f_i(x_i, t) = \omega_i \ (+ \eta_i)$, $g_{ij}(x_i, x_j, t) = \sin(x_j - x_i)$, and $\kappa_{ij} = k/N$. Of course more general dynamic networks that are not of the form Eq. 21 are also possible.

While systems of type (21) can be useful, the structure of most real-life networks changes with time. Generic examples of networks with changing topology include: social networks and transport networks (e.g. changing roads). For more specific examples we refer the reader to [12], which contains many relevant and recent examples (e.g. applications in neuroscience and machine learning).

Dynamic networks with temporally evolving connectivity are referred to as *adaptive dynamical networks*. Such networks are frequently written in the form

$$\begin{aligned} \dot{x_i} &= f_i(x_i, t) + \sum_{j=1}^{N} \kappa_{ij} g_{ij}(x_i, x_j, t), \\ \dot{\boldsymbol{\kappa}} &= h(\mathbf{x}, t), \end{aligned} \tag{22}$$

with h a generic function of $\mathbf{x}$ and t. It is clear from Eq. 22 that the network dynamics affect the network topology. Often times one only considers pairwise interaction, where the link strength only depends on the adjacent nodes. In this case

$$\dot{\kappa}_{ij} = h_{ij}(x_i, x_j, t),$$

with h_{ij} a generic function of x_i, x_j and t. Though the capacity to model interesting phenomena has most certainly increased, so has the complexity; the $N-$dimensional dynamic network has been replaced by the $N + N^2-$dimensional adaptive dynamical network!

For many systems the plasticity (change in topology) occurs at a much slower rate than the network dynamics. Then we write

$$\dot{\kappa}_{ij} = \epsilon h_{ij}(x_i, x_j, t)$$

for the network topology, where $\epsilon \ll 1$ is some small parameter.

Some interesting phenomena that may occur in these systems are synchronization, clustering, and multistability, all of which will be discussed in more detail in the following sections. Other dynamical phenomena that these networks may exhibit include noise-resistance [13, 14] and recurrent synchronization [15].

5 Synchronization in a Kuramoto-type Adaptive Dynamical Network

In the present section we will look at an adaptive dynamical network that is described by

$$\begin{aligned} \dot{\phi}_i &= \omega_i - \frac{\sigma}{N}\sum_{j=1}^{N}\kappa_{ij}\sin(\phi_i - \phi_j), \\ \dot{\kappa}_{ij} &= -\epsilon\big[\kappa_{ij} + \sin(\phi_i - \phi_j + \beta)\big], \end{aligned} \tag{23}$$

where we note that the coupling strength is now denoted by σ rather than k in order to avoid confusing notation with the coupling weights κ_{ij}. The ω_i's are drawn from the $U(-\hat{\omega}, \hat{\omega})$ distribution, ϵ is a small parameter to separate the timescales as explained in Section 4, and β can be used to change the adaptation rules. The focus will be on *symmetric adaptation rules*, which means that $\beta \approx -\frac{\pi}{2}$. The same system and its properties were also investigated in [2].

In this section and the next synchronization will be quantified using the synchronization index S,

$$S = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{N} s_{ij},$$

where

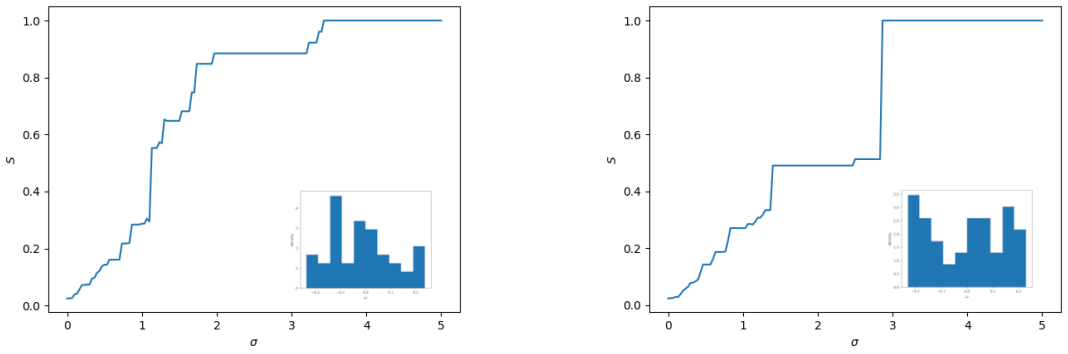
$$s_{ij} = \begin{cases} 1, & \langle\dot{\phi}_i\rangle = \langle\dot{\phi}_j\rangle, \\ 0, & \text{otherwise.} \end{cases}$$

Note that the angled brackets here indicate averaging and not expectation value. The averaging is done over a sufficiently long time frame (ideally infinite) and the averaging starts after a sufficiently large transient time (in order for the network to reach its stable state).

In this section and the subsequent section we are interested in small networks (e.g. $N = 15$, $N = 50$). Not only does this make the simulations much less time intensive, but finite-size effects will play an interesting role as well.

5.1 Synchronization and Frequency Clustering

As the coupling strength, σ, is increased, the system described by Eq. 23 transitions from fully asynchronous ($S \approx 0$) to fully synchronous ($S = 1$). This transition may be observed by simulating system (23) for a long time while slowly increasing the coupling strength (we refer to such a simulation as an upsweep). The results of two of these simulations are shown in Fig. 4.



(a) Multistep transition

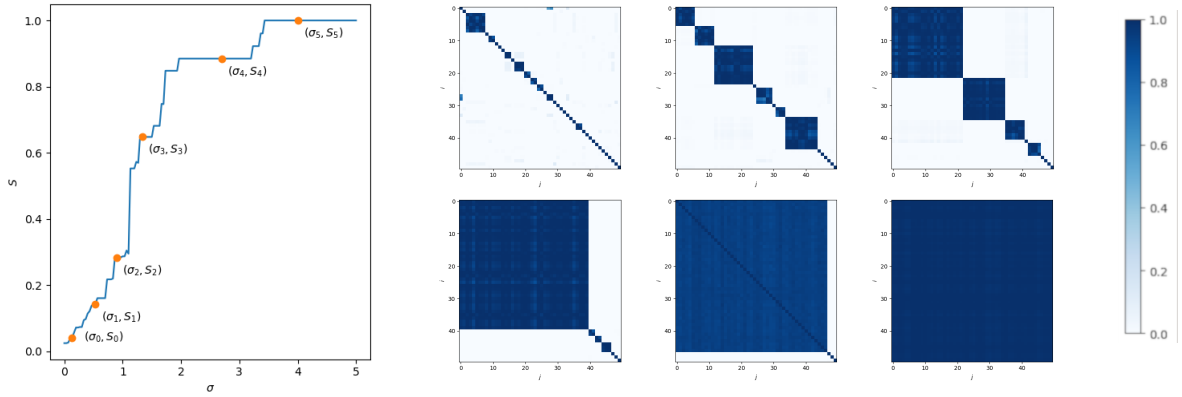
(b) Singlestep transition

Figure 4: Simulation of $N = 50$ oscillators with $\beta = -0.53\pi$ and $\hat{\omega} = 0.25$ (like in [2]). Both (a) and (b) are initiated with random initial conditions, and density plots of the natural frequency realizations are shown as insets. Each σ value is simulated for 10 000 time units, and S is calculated using the last 5000 time units. After every such simulation we increase σ by $5/150 \approx 0.03$, and use the final system values of the previous σ as initial conditions for the subsequent σ. The difference between the transitions is driven by the natural frequency distributions.

The first equation of system (23) is simulated using the dynamic time step Runge-Kutta-Fehlberg method (see Appendix C.2 for more details) and the second equation is simulated using the Euler forward method. This is justified because the coupling weights change at a much slower rate due to the ϵ. Also the results of simulations did not change significantly when the coupling weights were simulated using the Runge-Kutta-Fehlberg method. In the case of Figs. 4, 5 and 6 the system is simulated for a total of $1.51 \cdot 10^6$ time units, with $N = 50$, $\epsilon = 0.01$ and $\hat{\omega} = 0.25$. After every 10 000 time units σ is increased by approximately 0.03 and the synchronization index is calculated by averaging over the previous 5000 time units. This enables one to obtain a value for S for each value of σ. Moreover, the sorted coupling weights matrix $\boldsymbol{\kappa}$ is shown at specific points in time (i.e. values for σ) in Figs. 5b and 6b.

It is clear that the two simulations in Fig. 4 exhibit quite different properties. These are the finite size effects that we alluded to earlier. Fig. 4a is a typical example of a so called *multistep transition*. In this case the synchronization index, S, grows in many smaller steps as σ increases. Sometimes large steps are possible, but there is never a single large step to full synchronization. Fig. 4b, on the other hand, is a a typical example of a *singlestep transition*. Here S grows gradually until $S \approx 0.5$ where it stays for a relatively large range of σ values. Then when σ is high enough a jump to full synchronization is made.

To explain these differences it is helpful to look at the natural frequency distribution of the network (insets of Fig. 4) and the frequency clusters that are formed (Figs. 5 and 6). These explanations can be backed up by the simulations from [2]. For multistep transitions there is a high density spot in the center of the natural frequency distribution. This leads to one big frequency cluster with a phase velocity around the central frequency, that grows as σ increases (look at Fig. 5b). Since there is one dominant cluster that 'swallows' small clusters, the synchronisation index S grows in many small steps.



(a) Multistep transition

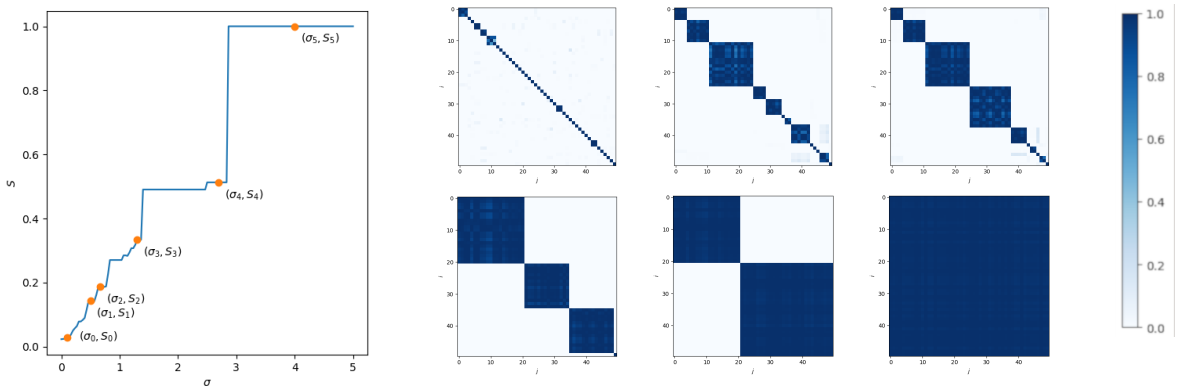
(b) Colour plots for sorted $|\boldsymbol{\kappa}|$ matrix for different values of σ.

Figure 5: Clustering for multistep transition. Clusters can be seen in (b). Each highlighted point in (a) corresponds to a plot in (b). Going from left to right and top to bottom in (b) means increasing σ. Note that the plot in (a) is identical to the one in Fig. 4a.

In the singlestep case, the natural frequency distribution has high density spots at the ends. This means there will be two dominant frequency clusters; one with phase velocity around $-\hat{\omega}$, and one with phase velocity around $\hat{\omega}$. Both clusters will grow in smaller steps. So up until the point where there are just two clusters left (see second to last plot in Fig. 6b), the singlestep transition looks like the multistep transition. When there are just two clusters left, a relatively high coupling strength, σ, is needed to synchronize the two clusters due to the relatively big difference in phase velocity of the two clusters. Therefore the two clusters will remain asynchronous over a large σ-range.

In Figs. 5b and 6b the (absolute value of the) coupling weight matrices, $\boldsymbol{\kappa}$, are plotted, yet we refer to frequency clusters. To show that the blue squares in these plots are indeed phase locked, it is sufficient to show that $\langle \phi_i \rangle \neq \langle \phi_j \rangle \implies \kappa_{ij} \approx 0$. According to [16], the coupling between different clusters is $\mathcal{O}(\epsilon)$. [6] Since $\epsilon \ll 1$ we have the required result.

[6] In [16] they actually consider a slightly different network, where each oscillator has the same natural frequency. However, it is not too hard to imagine that in the case of a narrow frequency distribution the clustered solutions will look very similar. The article on which this section is based also references [16] when stating that $\kappa_{ij} = \mathcal{O}(\epsilon)$.



(a) Singlestep transition

(b) Colour plots for sorted $|\boldsymbol{\kappa}|$ matrix for different values of σ.

Figure 6: Clustering for singlestep transition. Clusters can be seen in (b). Each highlighted point in (a) corresponds to a plot in (b). Going from left to right and top to bottom in (b) means increasing σ. Note that the plot in (a) is identical to the one in Fig. 4b.

It is also possible to justify that within frequency clusters $\kappa_{ij} \approx 1$. Since the phases in a cluster are locked, for phase i of cluster μ, we can write

$$\phi_{i,\mu} = s_\mu(t) + a_i, \tag{24}$$

where $s_\mu(t)$ is the collective phase of cluster μ, and a_i is the offset of phase i. By plugging Eq. 24 in the second equation of system (23) one obtains

$$\dot{\kappa}_{ij} = -\epsilon\big[\kappa_{ij} + \sin(a_i - a_j + \beta)\big],$$

which has general solution (use method of integrating factor)

$$\kappa_{ij} = -\sin(a_i - a_j + \beta) + C_{ij}e^{-\epsilon t},$$

where C_{ij} is a constant due to integration. For $\epsilon = 0.01$, after 5000 time units the second term becomes negligible. So for large t we get

$$\kappa_{ij} \approx -\sin(a_i - a_j + \beta). \tag{25}$$

Now an assumption needs to be made: we assume that the phase differences within a cluster are small. This is not a wild assumption to make since a cluster is like a little KM, which, for sufficiently high coupling strength, σ, implies a high value for order parameter r. This in turn implies small phase differences. Using the assumption and the fact that $\beta \approx -\frac{\pi}{2}$ means $\kappa_{ij} \approx 1$ within clusters.

This analytical result can easily be backed up numerically by calculating the synchronization index, S, by letting $s_{ij} = 1$ if $|\langle\kappa_{ij}\rangle| > 0.8$, and $s_{ij} = 0$ otherwise, and comparing it to the S calculated in the usual way. In Fig. 7 it can be seen that the absolute value of the difference is usually below 0.005.

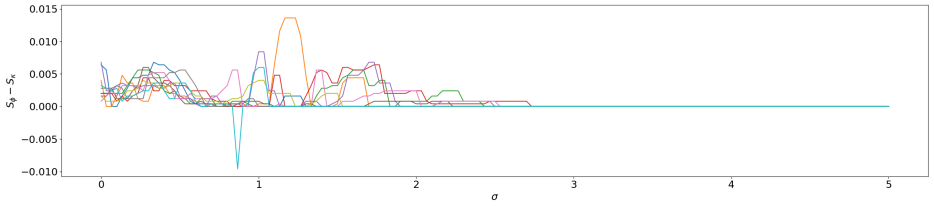


Figure 7: Difference in synchronization index using kappa calculation method and phi calculation method for 10 different upsweeps. Some smoothing is applied to eliminate peaks that exist for a single σ value.

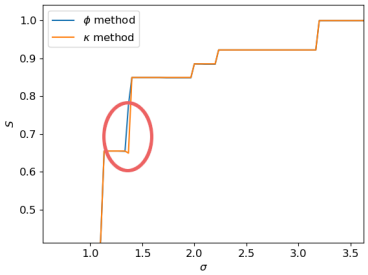


Figure 8: Example of the synchronization index calculated using the κ method lagging behind a single σ value.

The difference in synchronization index for the two calculation methods is shown for 10 different simulations in Fig. 7. For each simulation the initial conditions and natural frequency distributions are randomized. All other system properties (N, ϵ, and $\hat{\omega}$) remain constant and are the same as before.

A median filter of size 3 is applied to eliminate peaks that exist for a single σ value. This is justified because when a relatively big jump in S is made, occasionally the new coupling weights that are added to the bigger cluster lag behind to such an extent that their average just doesn't exceed the threshold value of 0.8, but will for the next value of σ. An example of such a situation is shown Fig. 8.

Finally, it is worth mentioning why we can use the absolute value of κ_{ij} in these calculation for S (and why plotting the absolute value of $\boldsymbol{\kappa}$ in Figs. 5b and 6b is justified). This is because system (23) is invariant under the transformation $\phi_i \to \phi_i + \pi$, $\kappa_{ij} \to -\kappa_{ij}$. The proof is not difficult and can be found in [2]. It is also clear that Eq. 25 still works; under this transformation $a_i - a_j \approx \pi$ meaning that the RHS becomes $-\kappa_{ij}$.

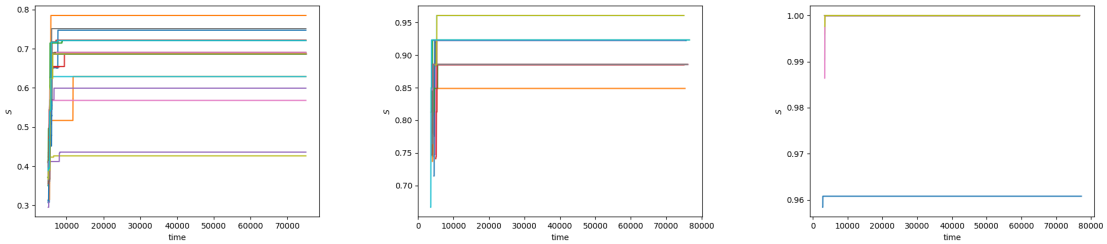
5.2 Multistability

Another important dynamical feature that system (23) displays is multistability. That is, given identical network properties (but necessarily different initial conditions) the system can reach different stable states. This phenomenon can easily be shown by simulating identical versions of system (23) for different initial conditions and plotting the synchronization index against time.

In the following simulations the Runge-Kutta-Fehlberg and Euler forward methods are used again. The system properties are: $N = 50$, $\epsilon = 0.01$ and $\hat{\omega} = 0.25$ (all like before). In the following simulations the ω-distributions are all identical and such that the system is a multistep system. Three values of coupling strength are looked at: $\sigma = 1.3$ in Fig. 9a, $\sigma = 2.0$ in Fig. 9b and $\sigma = 2.7$ in Fig. 9c. For each value of σ, 20 random initial conditions are simulated.

The plots below were made by taking the moving average of the instantaneous synchronization index. The moving average window is, in all cases, approximately[7] 10 000 time units.

Fig. 9 strongly suggests multistability.



(a) $\sigma = 1.3$ **(b)** $\sigma = 2.0$ **(c)** $\sigma = 2.7$

Figure 9: Different initial conditions but identical system properties lead to different stable values of S due to multistability. This is a multistep system.

[7] We say 'approximately' because in the moving average calculations the assumption of a constant time step is made. A varying time step means some errors may appear in the moving averages. Due to the small fluctuations in step size it is assumed that these errors are negligible. The number 10 000 is obtained because usually the step size is close to 1.

6 Synchronization in a Kuramoto-type Adaptive Dynamical Network with Noise

In this section we look at a modified version of system (23), where the coupling weights are affected by a white noise source. The new system can be written as

$$\begin{aligned} \dot{\phi}_i &= \omega_i - \frac{\sigma}{N}\sum_{j=1}^{N}\kappa_{ij}\sin(\phi_i - \phi_j), \\ \dot{\kappa}_{ij} &= -\epsilon\big[\kappa_{ij} + \sin(\phi_i - \phi_j + \beta)\big] + \sqrt{\mu}\xi_{ij}, \end{aligned} \tag{26}$$

where all system parameters have the same meaning as before and the ξ_{ij} are a set of uncorrelated white noises:

$$\langle \xi_i(t)\rangle = 0, \qquad \langle \xi_{ij}(t)\xi_{lk}(t')\rangle = \delta_{il}\delta_{jk}\delta(t'-t),$$

and μ is the noise strength. Simulations show that system (26) can achieve full synchronization for significantly lower values of σ when performing a similar upsweep as before. It appears to be the case that increasing the noise strength leads to faster full synchronization. Of course there is a limit to this relationship; if the noise strength exceeds a certain critical value the network cannot reach full synchronization anymore.

In the following simulations we will exclusively be looking at singlestep transitions. The phases are simulated using the Runge-Kutta-Fehlberg method, but a maximum step size of 0.05 is implemented. This is because for the coupling weights we use the Euler-Maruyama scheme, which in this particular case has order 1 (see Appendix C.1). According to [17], for nearly constant drift and diffusion coefficients the Euler-Maruyama scheme usually gives good results. Since ϵ is small and μ is constant it is assumed that the numerical results of the Euler-Maruyama method are reliable. Note that in this case we should take more care using the Euler method compared to the simulations in the previous section because it is no longer possible to rely on the slow dynamics of the coupling due to the noise.

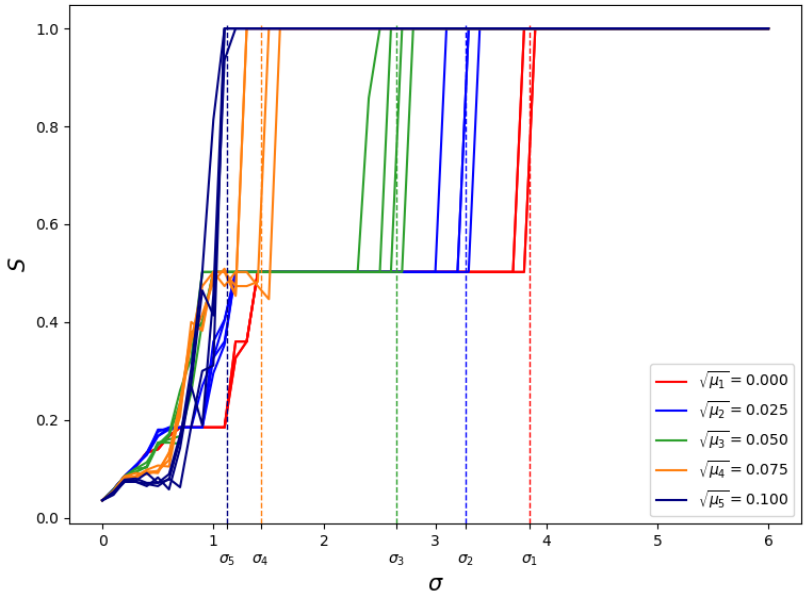
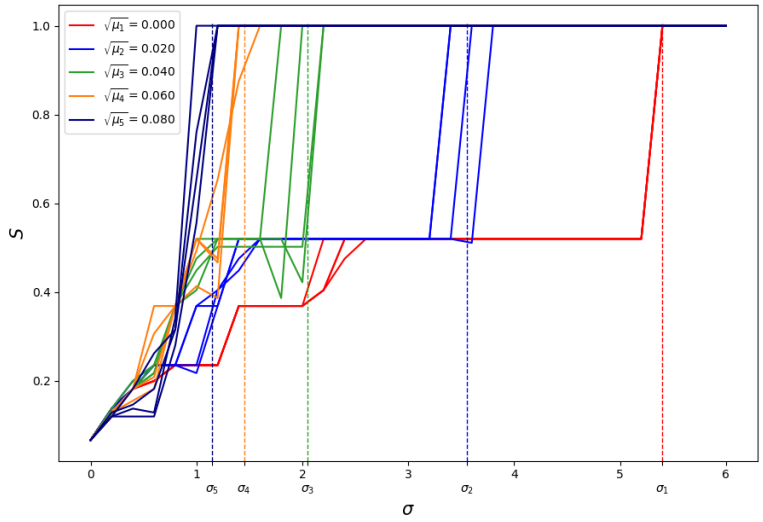
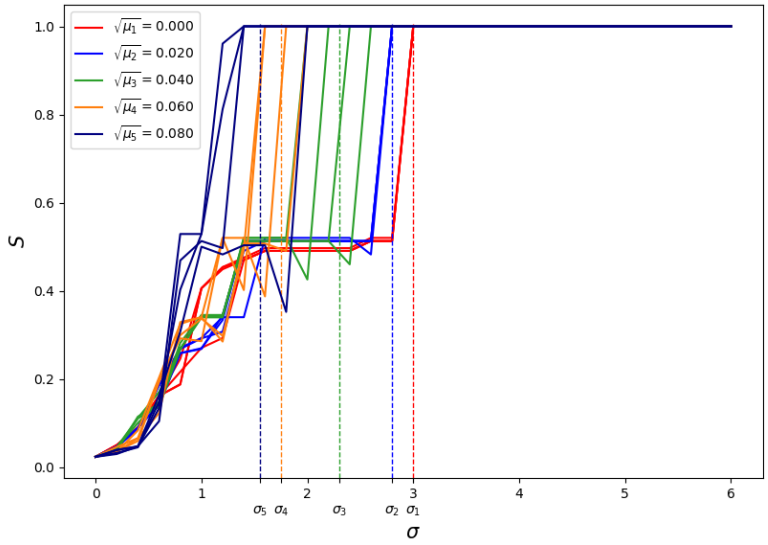


Figure 10: System size: $N = 30$, number of different initial conditions per value of noise strength: 4, number of steps from $\sigma = 0$ to $\sigma = 6$: 61, number of time units between each increase in σ: 5000. Dotted lines indicate average σ-value for which full synchronization is reached. The indices of σ and μ in this plot correspond, i.e. for μ_1 full synchronization occurs at coupling strength σ_1. We have $\mu_1 < \mu_2 < ... < \mu_5$ and $\sigma_5 < \sigma_4 < ... < \sigma_1$, so increasing noise strength leads to decreasing coupling strength for which full synchronization occurs.



(a) $N = 15$



(b) $N = 50$

Figure 11: Similar upsweep simulations as in Fig. 10 for different system sizes. Number of different initial conditions per value of noise strength: 4, number of steps from $\sigma = 0$ to $\sigma = 6$: 31, number of time units between each increase in σ: 5000. Dotted lines indicate average σ-value for which full synchronization is reached. The indices of σ and μ in this plot correspond, i.e. for μ_1 full synchronization occurs at coupling strength σ_1. In both cases we have $\mu_1 < \mu_2 < ... < \mu_5$ and $\sigma_5 < \sigma_4 < ... < \sigma_1$, so increasing noise strength leads to decreasing coupling strength for which full synchronization occurs.

In Figs. 10 and 11, system (26) is simulated for different μ values while the other system properties (ω-distribution, N, ϵ, and β) are held constant. For each value of noise strength, μ, a number of different upsweeps are performed, all with random initial conditions. In Figs. 10 and 11 the ω-distribution is such that the system exhibits a singlestep transition, $\epsilon = 0.01$, and $\beta = -0.53\pi$. Other system properties vary per plot, see captions for more details.

These plots strongly suggest that for an upsweep as described in Section 5.1, white noise in the coupling may induce full synchronization in singlestep transitions sooner than without the presence of noise. Furthermore, these plots seem to suggest that the fully synchronised state is more stable than the other states.

6.1 Qualitative Explanation

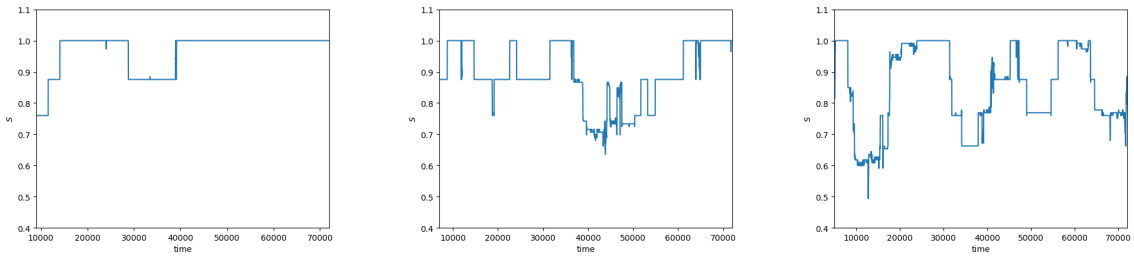
The problem of escape from stable states due to noise is pervasive in physics and other fields of science. Indeed, it seems to be what we're dealing with at present.

As explained in Section 5.2, it appears as though system (26) without noise exhibits a certain degree of multistability. In Figs. 12 and 13 we show that the additive noise term can cause jumps between stable states.[8] In these figures, system (26) is simulated in the same way as at the start of Section 6, except the coupling strength, σ, is kept constant.

Given the results from Figs. 12 and 13, it seems reasonable to assume that this is what is causing the jumps to full synchronization for lower values of σ In Figs. 10 and 11.

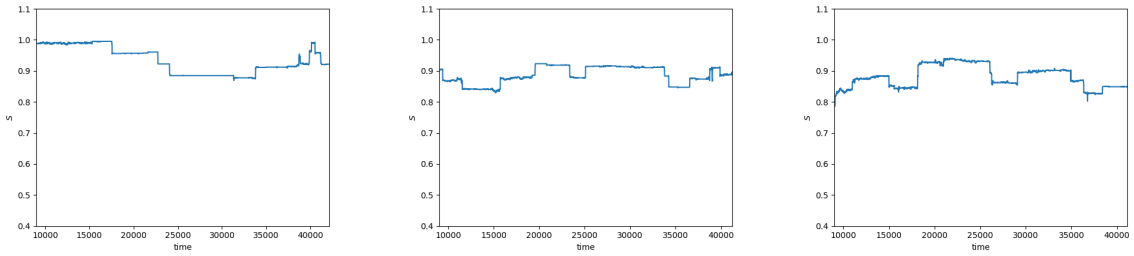
However, this does not explain why we don't see jumps from $S = 1$ back down to $S \approx 0.5$ in the upsweep plots. It does not seem plausible that it's because of the fact that σ is increasing, since this happens at a very slow rate meaning there would be enough time for a downwards jump to occur before the coupling strength is too high.[9] A more reasonable explanation would be that the fully synchronized state is more stable than the other states meaning a larger perturbation in the coupling weights is required to change the synchronization index (i.e. to desynchronize it).

In order to make any of these statements more definitive, either a more comprehensive numerical study should be done or an analytical study should be performed. However, as will be explained in the subsequent section, approaching this problem analytically is rather difficult (at least for me). Performing more simulations is certainly possible, but would require more time given that the runtime to produce Fig. 10 on my PC is in excess of 2 hours (and a lot more data is required).



(a) $\sqrt{\mu} = 0.14$ **(b)** $\sqrt{\mu} = 0.15$ **(c)** $\sqrt{\mu} = 0.16$

Figure 12: Jumping between stable states due to noise for varying values of noise strength. System size: $N = 15$, coupling strength: $\sigma = 3$.



(a) $\sqrt{\mu} = 0.27$ **(b)** $\sqrt{\mu} = 0.28$ **(c)** $\sqrt{\mu} = 0.29$

Figure 13: Jumping between stable states due to noise for varying values of noise strength. System size: $N = 50$, coupling strength: $\sigma = 3$.

[8] Although we haven't proven that system (26) is actually multistable, we will assume that it is.

[9] In Figs. 12 and 13 we do see frequent downward jumps but the noise values are much higher. These plots are only used to illustrate the fact that the noise in the coupling can induce jumps between stable states.

6.2 Hysteresis

In [2] it was shown that system (26) without noise exhibits hysteric behaviour in that a downsweep of the system (starting at high σ and decreasing its value) leads to a significantly lower value of σ for which the system can be fully synchronized. Here we show that for certain values of noise strength this hysteric behaviour can largely be eliminated.

The simulation in Fig. 14 was done identically to the one in Fig. 11a except that σ was decreased from 6 to 0. We also simulated fewer noise values and fewer initial conditions per noise value but this does not affect the dynamics of the system.

From Figs. 11a and 14 one can see that the upsweep behaviour is very similar to the downsweep behaviour if the noise strength satisfies $\sqrt{\mu} = 0.080$ and $N = 15$ (and all other system properties are as specified at the start of Section 6). This means that for these conditions no hysteresis is observed when slowly changing the coupling strength.

Finally, this downsweep behaviour seems to support the explanation that a larger perturbation in the coupling weights is required to desynchronize the system from full synchronization compared to synchronizing the system from the two cluster state.

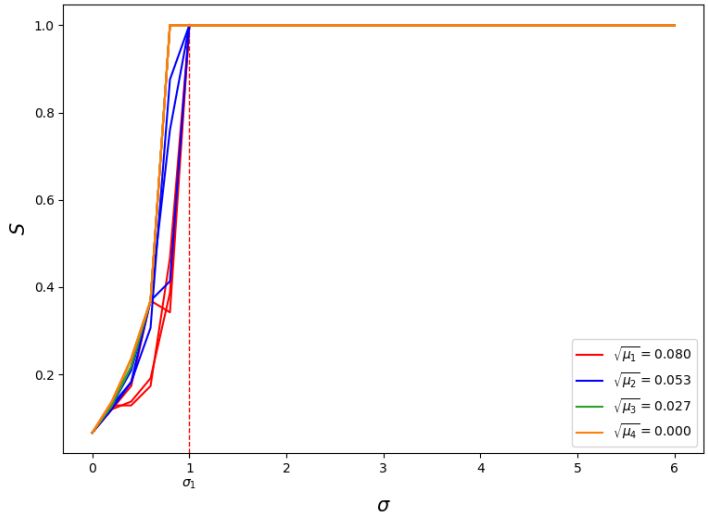


Figure 14: Downsweep instead of upsweep. System size: $N = 15$, number of different initial conditions per value of noise strength: 3, number of steps from $\sigma = 6$ to $\sigma = 0$: 31, number of time units between each decrease in σ: 5000. The dotted line indicates average σ-value (only for the highest noise value to prevent cluttering) for which the system no longer is fully synchronized.

7 Discussion

Before I go into the main points of this thesis I would like to point out a few smaller, miscellaneous discussion points.

The simulations in Section 5 could potentially have been sped up by using a different Runge-Kutta method. For example, the Dormand-Prince method seems to be superior to the Runge-Kutta-Fehlberg method [18].

On the other hand, for the simulations in Section 6 the Runge-Kutta-Fehlberg method may have been overkill. Due to the presence of noise we introduced a maximum time step of 0.05. However, the optimum adaptive time step, calculated according to the methods in Appendix C.2, was always significantly greater than 0.05. This means a lower order integration method for the phases in system (26) may have been sufficient, which could have sped up computations.

The insets in Fig. 4 don't provide the clearest example of a typical multistep frequency distribution and singlestep distribution. Ideally, one would simulate system (23) many times and take the averaged frequency distributions for all multistep transitions and all singlestep transitions. This is what is done in [2], and a clear bump is observed in the center of the distribution for multistep transitions and bumps are seen at the ends of the frequency distribution of the singlestep transitions.

Finally, it is worth mentioning that for all 6 curves in Fig. 2 a different random realization of the natural frequency distribution was taken for each increase in k. Though this means we are technically changing system properties during the simulation, since we are assuming these simulations are valid representations of the thermodynamic limit ($N \to \infty$), it shouldn't matter. A few simulations were performed for which the realization of the frequency distribution remained constant as k increased and there was no noticeable difference in the k_c value. It will be noted that curves looked smoother.

7.1 Non-Adaptive Kuramoto Model

In Section 3 an equation (Eq. 20) for the coupling strength, k_c, was derived according to the methods in [1]. This derivation essentially also showed us how Sakaguchi obtained his result for k_c in the infinite KM under the influence of white noise. On top of that, it provided a nice use case of the Fokker-Planck equation, which is an important tool in the field of statistical physics.

Eq. 20 proved to be accurate for low values of correlation time, τ, however it quickly broke down as τ increased. This was not particularly surprising given that the assumption of small τ was made in the derivation. One could argue that the use case of Eq. 20 is limited since τ has to be small, but not so small that it cannot reasonably be approximated by the white noise equation (Eq. 12).

Obtaining a more accurate equation for k_c would require solving the two variable FPE (for ψ and η), which is a difficult task.

7.2 Adaptive Kuramoto Model

In Section 5 we looked at a Kuramoto-type adaptive dynamical network (Eq. 23) that was studied in [2]. Through the use of numerical simulation it was shown that 3 interesting dynamical phenomena occur in system (23). These phenomena were in order of appearance: synchronization, frequency clustering and multistability.

Synchronization was shown to happen in steps with slowly increasing coupling strength, σ. A distinction was made between two inherently different paths to full synchronization; the singlestep transition and the multistep transition, as in [2]. The difference in these two transitions was explained by the formation of clusters. We showed that clusters in the coupling matrix, $\boldsymbol{\kappa}$, corresponded to phase locked clusters (i.e. frequency clusters). This was done analytically (with a few assumptions) and backed up numerically.

In Section 6 additive white noise was introduced in the coupling weights. Interestingly, it was shown that for a similar upsweep protocol as was used in [2] and in Section 5, this caused full synchronization to occur for lower values of coupling strength, σ. Furthermore, it was shown that the hysteric behaviour shown in [2] was not present for specific values of noise strength.

While the quicker path to synchronization due to noise was explained qualitatively, further conclusions were unable to be drawn. This was because of to the low number of simulations and the lack of an analytical study.

In order to gain more insight into system (26) an analytical study was attempted, but proved to be difficult, even in the extremely simplified case of just 2 oscillators. If we assume that there are 2 oscillators and that the coupling weights are symmetric (i.e. $\kappa = \kappa_{12} = \kappa_{21}$), we may write system (26) as

$$\begin{aligned} \dot{\psi} &= \Delta\omega - \sigma\kappa\sin(\psi), \\ \dot{\kappa} &= -\epsilon\big[\kappa + \sin(\psi + \beta)\big] + \sqrt{\mu}\xi, \end{aligned} \tag{27}$$

where $\psi \equiv \phi_1 - \phi_2$ and $\Delta\omega \equiv \omega_1 - \omega_2$. This is a set of coupled Langevin equations so we can easily write down the FPE for this system. The resultant FPE appeared difficult to solve so a different approach was needed. We present two methods that were looked into.

In [19] an adaptive dynamical network similar to the one described by Eq. 27 is considered. In that case the coupling weights also change at a slower timescale and an additive white noise term is also present. The key difference is that the noise term there is in the equation for the dynamic fast-changing variable. This allows them to write a one variable FPE for the probability distribution given fixed values for the coupling and then make the assumption that this probability density converges to a stationary distribution on a faster timescale than the changes in the coupling weights. This key step is something we cannot do here.

In [20] a method is presented to find the stationary probability distribution of a FPE through the use of an ansatz that is valid in the low-noise limit. Though we are interested in low values of noise strength, this method still appears to require numerical integration. Perhaps a deeper look into this approach is required.

All in all, to get some sort of analytical solution to this problem I would need more time and maybe a little bit more background knowledge.

Appendix

A Proofs

A.1 Order Parameter Proofs

A.1.1 Phases Evenly Spaced Around Unit Circle

If N points are evenly spread around the unit circle then $\phi_j = 2\pi j/N + \theta$, where $0 \leq \theta < 2\pi/N$ is some arbitrary phase offset. Thus the order parameter becomes

$$re^{i\phi_0} = \frac{1}{N}\sum_{j=1}^{N} e^{i(2\pi j/N+\theta)} = \frac{e^{i\theta}}{N}\sum_{j=0}^{N-1}\left(e^{2\pi i/N}\right)^j = \frac{e^{i\theta}}{N}\frac{1-(e^{2\pi i/N})^N}{1-e^{2\pi i/N}} = \frac{e^{i\theta}}{N}\frac{1-e^{2\pi i}}{1-e^{2\pi i/N}} = 0.$$

Therefore $r = 0$.

A.1.2 Phases Uniformly Distributed on Unit Circle

In this case the ϕ_i are random variables meaning we should find $\langle r\rangle$. However, as it turns out, this is quite a difficult problem. If we ignore the factor $1/N$ then we have to find the expectation of the magnitude of the sum of N complex number on the unit circle. This is essentially the same as finding the expected distance of a random walk in $\mathbb{R}^2$, where at each step one goes a distance of 1 unit in a random direction. For general N explicit expressions for this expectation do not exist. The interested reader may refer to [21] for explicit expressions up to $N = 3$.

In order to get an upper bound we use Jensen's inequality[10]:

$$\begin{aligned}
\langle r\rangle^2 &\leq \langle r^2\rangle \\
&= \int_0^{2\pi}\cdots\int_0^{2\pi}\left|\frac{1}{N}\sum_{j=1}^{N}e^{i\phi_j}\right|^2\frac{1}{(2\pi)^N}d\phi_1...d\phi_N \\
&= \frac{1}{(2\pi)^N}\frac{1}{N^2}\int_0^{2\pi}\cdots\int_0^{2\pi}\sum_{j=1}^{N}e^{i\phi_j}\sum_{j=1}^{N}e^{-i\phi_j}d\phi_1...d\phi_N \\
&= \frac{1}{(2\pi)^N}\frac{1}{N^2}\int_0^{2\pi}\cdots\int_0^{2\pi}\left(N+\sum_{j=1}^{N}\sum_{\substack{k=1\\k\neq j}}^{N}e^{i(\phi_j-\phi_k)}\right)d\phi_1...d\phi_N \\
&= \frac{1}{(2\pi)^N}\frac{1}{N}\int_0^{2\pi}\cdots\int_0^{2\pi}d\phi_1...d\phi_N \\
&= \frac{1}{N}.
\end{aligned}$$

The double summation in the 4$^{\text{th}}$ line disappears because $\int_0^{2\pi}e^x dx = 0$. By taking the square root of both sides one obtains

$$\langle r\rangle \leq \frac{1}{\sqrt{N}},$$

i.e. $\langle r\rangle$ is of size $\mathcal{O}(1/\sqrt{N})$. Of course we have not shown how good this upper bound actually is, though numerical simulations suggest it is not too bad (see Fig. 15).

[10] Jensen's inequality states that for any convex function $f : I \to I$ (where I is an open interval) and random variable X whose set of possible outcomes is contained within I we have $f(\langle X\rangle) \leq \langle f(X)\rangle$. For a proof of the non-general case that f is twice differentiable see [22, p.81].

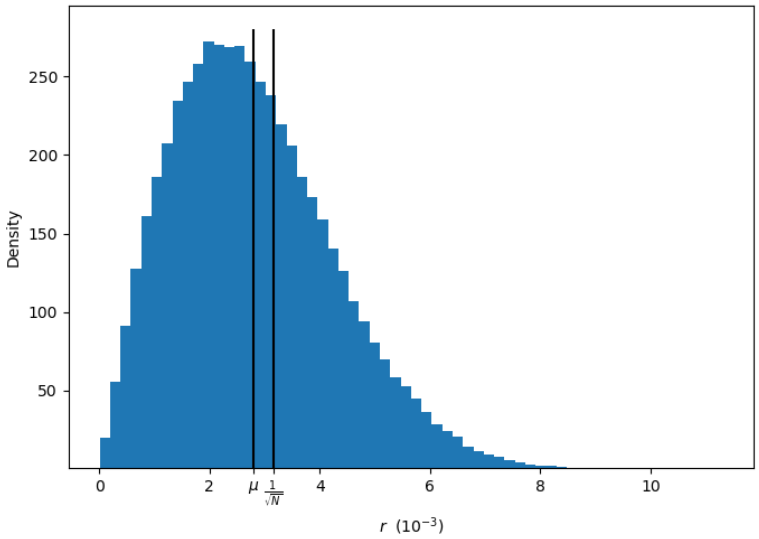


Figure 15: Density plot of 10^5 different instances of the order parameter r. Each instance of r is calculated using Eq. 2 with $N = 10^5$ and each phase is chosen from the $U_{[0,2\pi)}$ distribution. The mean μ (which is practically the same as the expectation value due to the law of large numbers) is only slightly smaller than $1/\sqrt{N}$. Moreover, the order parameter seems to be χ^2-distributed for uniformly distributed phases and large N.

A.1.3 Contribution Drifting Phases to Order Parameter is Zero

Since

$$n(\phi \mid \omega) = \frac{C}{|\omega - kr\sin(\phi - \phi_0)|}$$

we have $n(\phi \mid \omega) = n(\phi + \pi \mid -\omega)$. Furthermore, the frequency distribution is considered to be symmetric, i.e. $g(\omega) = g(-\omega)$. Now

$$\begin{aligned}
\int_{|\omega|>kr}\int_0^{2\pi} e^{i(\phi-\phi_0)} n(\phi \mid \omega) g(\omega) d\phi d\omega &= \int_{|\omega|>kr}\int_0^{2\pi} e^{i(\phi-\phi_0)} n(\phi+\pi \mid -\omega) g(-\omega) d\phi d\omega \\
&= \int_{|\omega|>kr}\int_0^{2\pi} e^{i(\phi-\phi_0)} n(\phi+\pi \mid \omega) g(\omega) d\phi d\omega \\
&= \int_{|\omega|>kr}\int_\pi^{3\pi} e^{-i\pi} e^{i(\phi-\phi_0)} n(\phi \mid \omega) g(\omega) d\phi d\omega \\
&= -\int_{|\omega|>kr}\int_0^{2\pi} e^{i(\phi-\phi_0)} n(\phi \mid \omega) g(\omega) d\phi d\omega.
\end{aligned}$$

B The Fokker-Planck Equation

In this section a quick overview of the Fokker-Planck equation (FPE) is given. One method of solution, for a very particular case, is also provided. Derivations and much (much) more detail can be found in [23, 24].

Consider a system that can be described by $\mathbf{x} = (x_1, ..., x_N)$. Suppose that these N variables satisfy the following set of stochastic equations of motion

$$\dot{x_i} = a_i(\mathbf{x}, t) + \sum_{j=1}^{N} b_{ij}(\mathbf{x}, t)\xi_j, \tag{28}$$

where ξ_j are a set of uncorrelated white noise processes with

$$\langle \xi_i(t) \rangle = 0, \qquad \langle \xi_i(t)\xi_j(t') \rangle = 2\delta_{ij}\delta(t' - t).$$

Eqs. 28 are known as Langevin equations: equations of motion that contain random forces. We refer to $a_i(\mathbf{x}, t)$ as the drift term and the $b_{ij}(\mathbf{x}, t)\xi_j$ as diffusive terms. Due to the diffusive terms, $x_1, ..., x_N$ are random variables. Thus it makes sense to talk about $\mathbf{x}$ in terms of its probability distribution function $p(\mathbf{x}, t)$. A PDE that describes the time-evolution of the probability distribution is given by

$$\frac{\partial}{\partial t}p(\mathbf{x}, t) = \left[-\sum_{i=1}^{N} \frac{\partial}{\partial x_i} D_i^{(1)}(\mathbf{x}, t) + \sum_{i=1}^{N}\sum_{j=1}^{N} \frac{\partial^2}{\partial x_i \partial x_j} D_{ij}^{(2)}(\mathbf{x}, t) \right] p(\mathbf{x}, t) \tag{29}$$

where

$$D_i^{(1)}(\mathbf{x}, t) = a_i(\mathbf{x}, t) + \sum_{j=1}^{N}\sum_{k=1}^{N} b_{kj}(\mathbf{x}, t)\frac{\partial}{\partial x_k} b_{ij}(\mathbf{x}, t),$$

$$D_{ij}^{(2)}(\mathbf{x}, t) = \sum_{k=1}^{N} b_{ik}(\mathbf{x}, t) b_{jk}(\mathbf{x}, t),$$

are the drift and diffusion coefficients respectively. Eq. 29 is known as the Fokker-Planck equation (FPE) and it provides an equivalent description of the system given by Eqs. 28.

For a one-dimensional system the FPE reduces to

$$\frac{\partial}{\partial t}p(x, t) = \left[-\frac{\partial}{\partial x} D^{(1)}(x, t) + \frac{\partial^2}{\partial x^2} D^{(2)}(x, t) \right] p(x, t) \tag{30}$$

where

$$D^{(1)}(x, t) = a(x, t) + b(x, t)\frac{\partial}{\partial x} b(x, t), \qquad D^{(2)}(x, t) = b^2(x, t).$$

Note that in some texts the FPE is given with a factor $\frac{1}{2}$ in front of the second summation term. This is also possible and it means one will get a different expression for $D^{(1)}(x)$. Here we decide to follow the convention of (probably) the most extensive book on the topic: The Fokker-Planck Equation by H. Risken ([24]).

B.1 Fokker-Planck Equation for One Variable: Stationary Solution

In this section we show how to obtain a solution for one-variable FPE with time-independent drift and diffusion coefficients. The FPE for one variable (Eq. 30) may be rewritten as

$$\frac{\partial}{\partial t}p(x, t) = -\frac{\partial}{\partial x} S(x, t), \tag{31}$$

where

$$S(x,t) = \left[D^{(1)}(x) - \frac{\partial}{\partial x} D^{(2)}(x)\right] p(x,t)$$

is known as the probability current. Now we introduce

$$\Phi(x) = \ln D^{(2)}(x) - \int_0^x \frac{D^{(1)}(x')}{D^{(2)}(x')} dx'.$$

Then the probability current can be written as

$$S(x,t) = -D^{(2)}(x) e^{-\Phi(x)} \frac{\partial}{\partial x} \left[e^{\Phi(x)} p(x,t)\right]. \tag{32}$$

If we're looking for a stationary solution (i.e. $p(x,t) = p_s(x)$) then it is clear from Eq. 31 that the probability current should be constant: $S(x,t) = S$. Thus Eq. 32 becomes

$$\frac{\partial}{\partial x} \left[e^{\Phi(x)} p_s(x)\right] = -S \frac{e^{\Phi(x)}}{D^{(2)}(x)}.$$

Integrating and rearranging gives

$$p_s(x) = N e^{-\Phi(x)} - S e^{-\Phi(x)} \int_0^x \frac{e^{\Phi(x')}}{D^{(2)}(x')} dx'.$$

The two constants, N and S, are to be determined by the normalization condition and the boundary conditions.

C Numerical Methods

C.1 Stochastic Differential Equations

A stochastic differential equation (SDE) is an equation of the form

$$\dot{x} = a(x,t) + b(x,t)\xi_t, \tag{33}$$

where ξ_t are uncorrelated Gaussian random variables, i.e.

$$\langle \xi_i(t) \rangle = 0, \qquad \langle \xi_i(t)\xi_j(t') \rangle = q\delta_{ij}\delta(t'-t).$$

and $a(x,t)$ and $b(x,t)$ are again referred to as the drift and diffusion coefficients resepctively. The formal way to interpret Eq. 33 is as follows

$$x(t) = \int_{t_0}^{t} a(x,t')dt' + \int_{t_0}^{t} b(x,t')dW(t'), \tag{34}$$

where $W(t)$ is a Wiener process and the latter integral is an Ito stochastic integral.[11] For more details on these concept we refer the reader to [17]. One important property of the *standard Wiener process* that will be given is that $W(t) - W(s)$ is normally distributed with zero mean.

C.1.1 Euler-Maruyama Method

Suppose that we want to simulate Eq. 33 up until a time T. Let $\{t_n : n = 1, ..., N \quad \text{and} \quad t_0 = t_1 < ... < t_n < ... < t_N = T\}$ be discretization of $[t_0, T]$. Now let y_n be the approximation of $x(t_n)$. Then a natural extension of the regular Euler forward scheme is given by

$$y_{n+1} = y_n + a(y_n, t_n)(t_{n+1} - t_n) + b(y_n, t)(W(t_{n+1}) - W(t_n)),$$

which is known as the *Euler-Maruyama method*. The only difference with the regular Euler method is that one has to determine

$$\Delta W_n \equiv W(t_{n+1}) - W(t_n)$$

at every time step. Since the ΔW_n are normally distributed with zero mean, one can just take a sample from the $N(0, \sqrt{\text{var}(\Delta W_n)})$ distribution in order to simulate ΔW_n at each time step. For the standard Wiener process $\text{var}(W(t) - W(s)) = t - s$, but since our white noise is delta correlated with a factor q, we have $\text{var}(\Delta W_n) = q(t_{n+1} - t_n)$. This can be shown as follows

$$\begin{aligned}
\langle \Delta W_n^2 \rangle &= \left\langle \left(\int_{t_n}^{t_{n+1}} dW(t) \right)^2 \right\rangle = \left\langle \left(\int_{t_n}^{t_{n+1}} \xi(t)dt \right)^2 \right\rangle \\
&= \left\langle \int_{t_n}^{t_{n+1}} \int_{t_n}^{t_{n+1}} \xi(t)\xi(s)dtds \right\rangle \\
&= \int_{t_n}^{t_{n+1}} \int_{t_n}^{t_{n+1}} \langle \xi(t)\xi(s) \rangle dtds \\
&= \int_{t_n}^{t_{n+1}} \int_{t_n}^{t_{n+1}} q\delta(t-s)dtds \\
&= \int_{t_n}^{t_{n+1}} q\mathbf{1}_{[t_n, t_{n+1}]}ds \quad = \quad q(t_{n+1} - t_n).
\end{aligned}$$

According to [17], when the noise is additive[12] and a and b are sufficiently smooth, the Euler-Maruyama scheme has order 1.

[11] The reason that one cannot use a 'normal' integral is that Eq. 34 suggests that $\xi_t = \frac{dW(t)}{dt}$, which poses a problem because a Wiener process is nowhere differentiable.

[12] Additive noise means that the diffusion coefficient is not a function of x, i.e. $b(x,t) = b(t)$. If the noise is not additive then the Euler-Maruyama scheme has order 0.5.

C.2 Runge-Kutta-Fehlberg Method

In this section we present some of the key results (for our purposes) from [25]. The goal is to find a numerical approximation for $x(t)$, where

$$\dot{x} = a(x,t).$$

The main idea of the method is to use a 5^{th} order Runge-Kutta method to estimate the local truncation error of a 4^{th} order Runge-Kutta method at step n and to use this estimate to determine the optimum time step for step $n+1$. So the 4^{th} order method is used for the integration and the 5^{th} order method is used to keep the error small.

We begin by showing that the 5^{th} order method can be used as a reasonable estimate for the local truncation error at each step. Let y_n and $\hat{y}_n$ be the 4^{th} and 5^{th} order approximations of $x(t_n)$ with $x(t_{n-1})$ as starting point respectively, and h be the time step used at time t_n (i.e. $h \equiv t_{n+1} - t_n$). Then

$$y_{n+1} = w_n + hg(t_n, x(t_n), h), \tag{35}$$

is obtained from the Taylor expansion of x about the point t_n,

$$x(t_{n+1}) = x(t_n) + hg(t_n, x(t_n), h) + \mathcal{O}(h^5).$$

And, similarly,

$$\hat{y}_{n+1} = \hat{y}_n + h\hat{g}(t_n, x(t_n), h), \tag{36}$$

is obtained from the higher order Taylor expansion (also about t_n)

$$x(t_{n+1}) = x(t_n) + h\hat{g}(t_n, x(t_n), h) + \mathcal{O}(h^6).$$

Let τ and $\hat{\tau}$ denote the local truncation errors for the 4^{th} and 5^{th} order methods respectively. By the definition of local truncation error we have

$$\tau_{n+1}(h) = \frac{1}{h}\big[x(t_{n+1}) - y_{n+1}\big],$$

and

$$\hat{\tau}_{n+1}(h) = \frac{1}{h}\big[x(t_{n+1}) - \hat{y}_{n+1}\big],$$

since the approximates y_{n+1} and $\hat{y}_{n+1}$ are made with $x(t_n)$ as starting point.[13] Using this fact and looking at the two equations above it is clear that $\tau_{n+1}(h)$ is $\mathcal{O}(h^4)$ and $\hat{\tau}_{n+1}(h)$ is $\mathcal{O}(h^5)$.

Rewriting the expression for $\tau_{n+1}(h)$ a little bit leads to

$$\begin{aligned}
\tau_{n+1} &= \frac{1}{h}\big[x(t_{n+1}) - \hat{y}_{n+1} + \hat{y}_{n+1} - y_{n+1}\big], \\
&= \hat{\tau}_{n+1} + \frac{1}{h}\big[\hat{y}_{n+1} - y_{n+1}\big], \\
&\approx \frac{1}{h}\big[\hat{y}_{n+1} - y_{n+1}\big],
\end{aligned}$$

where the last step is due to the fact that $\tau_{n+1}(h)$ is $\mathcal{O}(h^4)$ and $\hat{\tau}_{n+1}(h)$ is $\mathcal{O}(h^5)$.

Now we show how this result may be used to compute the optimum time step that ensures the local truncation error is (approximately) bounded by $\epsilon_{\max}$. Since $\tau_{n+1}(h)$ is $\mathcal{O}(h^4)$, there exists a K such that $\tau_{n+1}(h) \approx Kh^4$. Now define $h_{\text{opt}} \equiv ch$. Then

$$\tau_{n+1}(h_{\text{opt}}) = c^4 K h^4 \approx c^4 \tau_{n+1}(h) \approx \frac{c^4}{h}\big[\hat{y}_{n+1} - y_{n+1}\big].$$

Since we require $\tau_{n+1}(h_{\text{opt}}) \leq \epsilon_{\max}$

[13] Which is required by definition of truncation error.

$$\frac{c^4}{h}\left[\hat{y}_{n+1} - y_{n+1}\right] \leq \epsilon_{\max}.$$

Because we want the optimum time step (i.e. least computation time meaning largest possible time step), this inequality becomes an equality and

$$h_{\text{opt}} = \left(\frac{h\epsilon_{\max}}{\hat{y}_{n+1} - y_{n+1}}\right)^{\frac{1}{4}} h. \tag{37}$$

Implementation Let z_n denote the approximation of $x(t_n)$. Then a simple adaptive step size method can be implemented as follows:

1. Set initial conditions (including an initial value for h_{opt}).
2. While time, t_n, is less than end time complete the following steps:
   (a) Compute y_{n+1} according to Eq. 35 using h_{opt} and z_n as the previous step.
   (b) Compute $\hat{y}_{n+1}$ according to Eq. 36 using h_{opt} and z_n as the previous step.
   (c) If $\frac{1}{h_{\text{opt}}}\left[\hat{y}_{n+1} - y_{n+1}\right] \leq \epsilon_{\max}$:
      i. Set $z_{n+1} \to y_{n+1}$.
      ii. Set $t_n \to t_n + h_{\text{opt}}$.
   (d) Recompute h_{opt} using Eq. 37.

Looking at this implementation one may think isn't inefficient to compute a 4^{th} *and* 5^{th} order Runge-Kutta (RK) approximation at each time step (especially because RK methods generally require quite a few evaluations of the function $a(x,t)$). This would be right for an arbitrary combination of 4^{th} and 5^{th} order RK methods. However, in a 1969 NASA technical report [26], Erwin Fehlberg showed that RK methods exist for which the 4^{th} order calculations may be used in the 5^{th} order calculations. This means only a few extra calculations are necessary at each step.

To see how to calculate y_{n+1} and $\hat{y}_{n+1}$ using the Fehlberg method have a look at the `rk45_step` function in Appendix D.2.2 (or refer to [25, 26]).

Finally, for all simulations in this report that used the Runge-Kutta-Fehlberg method, we had: $\epsilon_{\max} = 10^{-5}$.

D Code

D.1 Mathematica Script

In[1]:=
```
ClearAll[r,x,y]
```

In[2]:=
```
A[r_,x_,y_]:=(k*r*(k*r*τ*Cos[2*x]+4*τ*y*Sin[x]+4*Cos[x])+4*x*y)/(4D)
```

In[3]:=
```
B[r_,x_]:=1+k*r*τ*Cos[x]
```

In[4]:=
```
a[r_,x_,y_]:=Series[Exp[-1*A[r,x,y]]*B[r,x],{r,0,3}]
```

In[5]:=
```
b[N_,r_,x_,y_]:=N*Exp[A[r,x,y]]*B[r,x]/(1-Exp[(2* Pi*y)/D])*(Integrate[a[r,z,y],{z,x,0}]+Exp[(2*Pi*y)/D]*
Integrate[a[r,z,y],{z,2*Pi,x}])
```

In[6]:=
```
c[N_,r_,y_]=Integrate[Series[b[N,r, x, y], {r,0,3}],{x,0,2*Pi}]
```

Out[6]=
$$\frac{D\ N\ \pi\ (2\ D^2+k^2\ r^2+2\ \ y^2)}{y\ (D^2+y^2)}$$

In[7]:=
```
normalization=Solve[c[N,r,y]==1,N]
```

Out[7]=
$$\{\{N\to\frac{y\ (D^2+y^2)}{D\ \pi\ (2\ \ D^2+k^2\ r^2+2\ y^2)}\}\}$$

In[8]:=
```
d[r_,x_,y_]=b[N,r,x,y]*Cos[x]/.normalization[[1]];
```

In[9]:=
```
e[r_,y_]=Integrate[Series[d[r,x,y],{r,0,3}],{x,0,2*Pi}];
```

In[10]:=
```
SeriesCoefficient[e[r,y],{r,0,1}]
```

Out[10]=
$$\frac{k\ (D+D^2\ \tau+y^2\ \tau)}{2\ \ (D^2+y^2)}$$

In[11]:=
```
SeriesCoefficient[e[r,y],{r,0,3}]
```

Out[11]=
$$-\frac{k^3\ (2\ D^3-4\ D\ y^2+4\ \ D^4\ \tau+5\ D^2\ y^2\ \tau+y^4\ \tau+6D^5\ \tau^2+6\ D^3\ y^2\ \ \tau^2)}{8\ (D^2+y^2)^2\ (4\ \ D^2+y^2)}$$

D.2 Python Code

In the following sections we give the main bodies of code that were used to perform the simulations from sections 2, 3, 5, and 6. However, many adaptations and versions of these codes were utilized. Examples of adaptations include (but are not limited to):

- Adding a for-loop to perform a certain simulation for different parameters,
- Changing functions to output different quantities,
- Changing functions to speed up computations under certain conditions (e.g. changing an Euler-Maruyama step to a regular Euler-forward step when no noise is simulated).

Moreover, there are quite a lot of sections of code attributed to plotting. All those bits of code haven't been included either because they're not that interesting.

D.2.1 Coloured Noise Code

This code can produce the data to produce a single curve from Fig. 2. By changing parameters the data for all the curves in this figure can be produced.

Making a few small modifications allows one to create the data for Fig. 3. This data still has to be interpreted though - most importantly a function needs to be created to determine the numerical value of k_c. I did this by creating a function that checks for what k value the change in r is beyond a certain critical value (i.e. a function that checks when the slope of $r(k)$ is high enough). A good value for the critical slope seemed to be around 0.5.

```
import numpy as np


np.random.seed(0)


#Saving stuff
FILENAME = "filename_here"
TAG = 0


#Simulation settings
K_MIN = 1
K_MAX = 5
N_KS = 21
N_TIME_STEPS = 50000


class Model():
    #Define model parameters
    D = 1
    LAMBDA = 0.5
    TAU = 3.5
    N = 5000


def g(N):
    #Natural frequency distribution function
    return Model.LAMBDA*np.random.standard_cauchy(N)


def dW(dt, N):
    #Return value of integrated white-noise - i.e. Wiener process increment
    return np.random.normal(loc=0.0, scale=(2*dt)**(0.5), size=N)


```

```
def run_simulation(k, D, tau, N_steps, omega, dt=1e-3):
    #Integrate Langevin equations (13) and (14) from main text
    #Return value of order parameter at each point in time
    N = len(omega)
    percentage_indicator = N_steps / 5
    phi = 2*np.pi*np.random.rand(N)
    eta = dW(dt, N)
    r_arr = np.zeros(N_steps)
    r_arr[0] = np.abs((1/N)*np.sum(np.exp(1j*phi)))

    for i in range(N_steps-1):
        cos_sum = np.cos(phi).sum()
        sin_sum = np.sin(phi).sum()
        phi = phi + (omega + (k/N)*(sin_sum*np.cos(phi) - cos_sum*np.sin(phi))
                                                              + eta)*dt
        eta = eta - (eta/tau)*dt + ((D**(0.5))/tau)*dW(dt, N)

        if abs(i % percentage_indicator) < 1e-10:
            print("{:.2f}% done".format((i/N_steps)*100))

        r_arr[i+1] = np.abs((1/N)*np.sum(np.exp(1j*phi)))

    return r_arr


def sweep_ks(k_min, k_max, N_ks, D, tau, N_time_steps, omega):
    #Perform integration of Langevin equations (13) and (14) for range of k's
    #Return averaged order parameter for each k value
    ks = np.linspace(k_min, k_max, N_ks)
    rs = np.zeros(len(ks))

    for i, k in enumerate(ks):
        print("k: {}".format(k))
        r_arr = run_simulation(k, D, tau, N_time_steps, omega)
        rs[i] = np.mean(r_arr[int(len(r_arr)/2):])
        print("\n{}\n\n".format(rs[i] ))

    return rs, ks


rs, ks = sweep_ks(K_MIN, K_MAX, N_KS, Model.D, Model.TAU, N_TIME_STEPS, g(Model.N))


np.save(FILENAME + "_rs_" + str(TAG), rs)
np.save(FILENAME + "_ks_" + str(TAG), ks)
```

D.2.2 Adaptive Dynamical Network Code

Most of this code is reasonably self-explanatory. But we shall make a few comments on some elements.

Firstly, in the `rk45_step` and the `run_simulation` functions we note that the `dt` variable contains two elements. This is because `rk45_step` updates the time step but we still want the coupling weights to be updated by the 'old' time step that was used to update the phases. Of course other solutions are possible as well.

If one wants to simulate system (23) it is recommended to remove the `dW` term in the `euler_maruyama_step` function to avoid the unnecessary generation of N^2 random numbers at each time step.

In lines 142-145 the weighted average of the coupling weights and phase velocities is updated. It has to be a weighted average due to the variable time step.

```
1  import numpy as np
2  from functools import partial
3  import timeit
4
5
6  SEED = 0
7  np.random.seed(SEED)
8
9
10 class SaveParam():
11     """Saving parameters"""
12
13     FILE_LOC = "file/location/here"
14     TAG = 0
15
16
17 class PlotParam():
18     """Plot parameters"""
19
20     N_NOISE_VALS = 5
21     N_ICS_PER_NOISE_VAL = 4
22     N_STEPS_SIGMA_SWEEP = 31
23     NOISE_MIN = 0
24     NOISE_MAX = 0.08
25     SIGMA_MIN = 0
26     SIGMA_MAX = 6
27
28
29 class SimParam():
30     """Simulation parameters"""
31
32     SIMULATION_TIME = 5000
33     MAX_LOCAL_TRUNC = 1e-5
34     DT_MAX = 0.05 #can be higher when simulating system (23)
35
36
37 class Model():
38     """Model parameters"""
39
40     N = 50
41     OMEGAS = np.random.uniform(-0.25, 0.25, N)
42     EPSILON = 0.01
43     BETA = -0.53*np.pi
44
45
46 def differences_matrix(x):
47     """This returns a matrix where element ij == x_i - x_j"""
48
```

```
    return np.outer(x, np.ones(len(x))) - np.outer(np.ones(len(x)), x)


def g(phi_diffs, kappas, sigma):
    """d/dt(phi) = g(phi, kappa)"""

    kappa_phi_sum = (kappas*np.sin(phi_diffs)).sum(-1)

    return Model.OMEGAS - (sigma/Model.N)*kappa_phi_sum


def h(kappas, phi_diffs):
    """d/dt(kappa) = h(phi, kappa)"""

    return -1*Model.EPSILON*(kappas + np.sin(phi_diffs + Model.BETA))


def dW(dt, size):
    """Return value of integrated white-noise - i.e. Wiener process increment"""

    return np.random.normal(loc=0, scale=(dt**(0.5)), size=size)


def update_time_step(dt, local_trunc_estimate):
    """Update time step so that the local truncation error stays small enough."""

    dt = (((SimParam.MAX_LOCAL_TRUNC)/(2*local_trunc_estimate))**(1/4))*dt

    if dt > SimParam.DT_MAX:
        dt = SimParam.DT_MAX

    return dt


def rk45_step(x_prev, dt, f, arg):
    """Compute a single time step using the Fehlberg method. Return the updated
    value of x, the velocity at this time-step, and updated dt. Note that
    dt[0] is the time step used, and dt[1] is the updated time step - to be
    used during the next step."""

    while True:
        k_1 = dt[0]*f(arg)
        k_2 = dt[0]*f(arg + (1/4)*k_1)
        k_3 = dt[0]*f(arg + (3/32)*k_1 + (9/32*k_2))
        k_4 = dt[0]*f(arg + (1932/2197)*k_1 - (7200/2197)*k_2 + (7296/2197)*k_3)
        k_5 = dt[0]*f(arg + (439/216)*k_1 - (8)*k_2 + (3680/513)*k_3 - (845/4104)*k_4)
        k_6 = dt[0]*f(arg - (8/27)*k_1 + (2)*k_2 - (3544/2565)*k_3 + (1859/4104)*k_4 -
            (11/40)*k_5)
        dx = (25/216)*k_1 + (1408/2565)*k_3 + (2197/4104)*k_4 - (1/5)*k_5
        dz = (16/135)*k_1 + (6656/12825)*k_3 + (28561/56430)*k_4 - (9/50)*k_5 + (2/55)*k_6

        local_trunc_estimate = abs(np.max(dx - dz))/dt[0]

        dt[1] = update_time_step(dt[0], local_trunc_estimate)

        if local_trunc_estimate < SimParam.MAX_LOCAL_TRUNC:
            x = x_prev + dx
            break

        dt[0] = dt[1]

    return x, dx/dt[0], dt
```

```
110
111
112 def euler_maruyama_step(y_prev, dt, a, b):
113     """Compute a single time-step using the Euler-Maruyama method"""
114
115     return y_prev + a*dt + b*dW(dt, y_prev.shape)
116
117
118 def run_simulation(sigma, gamma, phi_0, kappa_0, time_limit=SimParam.SIMULATION_TIME):
119     """Simulate system (25) (or system (23) if gamma = 0) using the RK
120     Fehlberg method for the phis and the Euler-Maruyama method for the kappas.
121     Return the weighted average of the kappa matrix and the phase velocity.
122     Average is taken after 50% of the time_limit. Also return final values for
123     kappa matrix and phis."""
124
125     averaging_time = time_limit / 2
126     kappas_mean = np.zeros((Model.N, Model.N))
127     d_phis_mean = np.zeros(Model.N)
128     kappas = kappa_0
129     phis = phi_0
130     dt = [0.01, 0.01]
131     time = 0
132     count = 0
133
134     while time < time_limit:
135         phi_diffs = differences_matrix(phis)
136         f = partial(g, kappas=kappas, sigma=sigma)
137         phis, d_phi, dt = rk45_step(phis, dt, f, phi_diffs)
138         f = partial(h, phi_diffs=phi_diffs)
139         kappas = euler_maruyama_step(kappas, dt[0], f(kappas), gamma)
140
141         if time >= averaging_time:
142                 kappas_mean = kappas_mean + (dt[0] / (time - averaging_time +
143                                             dt[0]))*(kappas - kappas_mean)
144                 d_phis_mean = d_phis_mean + (dt[0] / (time - averaging_time +
145                                             dt[0]))*(d_phi - d_phis_mean)
146
147         if count % 5000 == 0:
148             print("    count: {} |  dt: {} |  time: {}".format(count, dt, time))
149
150         time += dt[0]
151         count += 1
152         dt[0] = dt[1]
153
154     return kappas_mean, d_phis_mean, phis, kappas
155
156
157 def sweep_sigmas(sigmas, gamma, phi_0, kappa_0, IC_number):
158     """Perform simulation of system (25) for different values of signma.
159     Use final values of previous simulation as initial conditions for
160     subsequent simulation. For each value return the sync index calculated
161     using the phi method and the sync index calculated using the kappa method."""
162
163     sync_index_kappa = np.zeros(len(sigmas))
164     sync_index_phi = np.zeros(len(sigmas))
165     phi_initial = phi_0
166     kappa_initial = kappa_0
167
168     for i, sigma in enumerate(sigmas):
169         print(" SIMULATION NUMBER: {} | SIGMA: {} | NOISE VALUE: {} | IC NUMBER: {}"
170               .format(i, sigma, gamma, IC_number))
171         kappas, phase_velocities, phi_initial, kappa_initial = \
```

```
                    run_simulation(sigma, gamma, phi_initial, kappa_initial)
        sync_index_kappa[i] = ((abs(kappas) > 0.8).sum())/(Model.N**2)
        sync_index_phi[i] = ((abs(differences_matrix(phase_velocities)) <
            1e-3).sum())/(Model.N**2)
        print("    SYNC INDEX KAPPA: {} | SYNC INDEX PHI: {}\n"
              .format(sync_index_kappa[i], sync_index_phi[i]))

    return sync_index_kappa, sync_index_phi


def sweep_sigmas_N_ICs(N_initial_conditions, sigmas, gamma):
    """Perform a sigma sweep N times. Return N instances of sync index against
    sigma (for both ways of calculating sync index)."""

    sync_indexes_kappa = np.zeros((N_initial_conditions, len(sigmas)))
    sync_indexes_phi = np.zeros((N_initial_conditions, len(sigmas)))

    for i in range(N_initial_conditions):
        print("INITIAL CONDITIONS: {}".format(i))
        kappa_0 = np.random.uniform(-1, 1, (Model.N, Model.N))
        #kappa_0 = 0.5*(kappa_0 + kappa_0.T)
        phi_0 = np.random.uniform(0, 2*np.pi, Model.N)
        sync_indexes_kappa[i], sync_indexes_phi[i] = sweep_sigmas(sigmas, gamma, phi_0,
            kappa_0, i)

    return sync_indexes_kappa, sync_indexes_phi


def main():
    """Perform sigma sweep N_ICS_PER_NOISE_VAL times for the specified number
    of noise values. Save the files when done."""

    sigmas = np.linspace(PlotParam.SIGMA_MIN, PlotParam.SIGMA_MAX,
        PlotParam.N_STEPS_SIGMA_SWEEP)
    sync_indexeses_kappa = np.zeros((PlotParam.N_NOISE_VALS,
                                     PlotParam.N_ICS_PER_NOISE_VAL,
                                         PlotParam.N_STEPS_SIGMA_SWEEP))
    sync_indexeses_phi = np.zeros((PlotParam.N_NOISE_VALS,
                                   PlotParam.N_ICS_PER_NOISE_VAL, PlotParam.N_STEPS_SIGMA_SWEEP))

    noise_vals = np.linspace(PlotParam.NOISE_MIN, PlotParam.NOISE_MAX, PlotParam.N_NOISE_VALS)

    for i, noise_val in enumerate(noise_vals):
        print("NOISE VALUE: {}\n". format(noise_val))
        sync_indexeses_kappa[i], sync_indexeses_phi[i] = \
        sweep_sigmas_N_ICs(PlotParam.N_ICS_PER_NOISE_VAL, sigmas, noise_val)

    np.save(SaveParam.FILE_LOC + "_sync_index_kappa_" + str(SaveParam.TAG) + "_seed_"
            + str(SEED), sync_indexeses_kappa)
    np.save(SaveParam.FILE_LOC + "_sync_index_phi_" + str(SaveParam.TAG) + "_seed_"
            + str(SEED), sync_indexeses_phi)
    np.save(SaveParam.FILE_LOC + "_noise_values_" + str(SaveParam.TAG) + "_seed_"
            + str(SEED), noise_vals)
    np.save(SaveParam.FILE_LOC + "_sigmas_" + str(SaveParam.TAG) + "_seed_"
            + str(SEED), sigmas)


start_time = timeit.default_timer()
main()
stop_time = timeit.default_timer()
print("Runtime: {:.2f}s".format(stop_time - start_time))
```

References

[1] C. Maggi and M. Paoluzzi. The effect of time-correlated noise on the kuramoto model studied via the unified colored noise approximation, 2019.

[2] Jan Fialkowski, Serhiy Yanchuk, Igor M. Sokolov, Eckehard Schöll, Georg A. Gottwald, and Rico Berner. Heterogeneous nucleation in finite-size adaptive dynamical networks. *Physical Review Letters*, 130(6), feb 2023.

[3] Haixia Wang, Qingyun Wang, and Qishao Lu. Bursting oscillations, bifurcation and synchronization in neuronal systems. *Chaos, Solitons Fractals*, 44(8):667–675, 2011.

[4] Peter J. Uhlhaas and Wolf Singer. Abnormal neural oscillations and synchrony in schizophrenia. *Nature Reviews Neuroscience*, 11(2):100–113, 2010.

[5] J.A. Acebrón, L.L. Bonilla, C.J. Pérez Vicente, F. Ritort, and R. Spigler. The kuramoto model: A simple paradigm for synchronization phenomena. *Rev. Mod. Phys.*, 77:137–185, Apr 2005.

[6] Y. Kuramoto. Self-entrainment of a population of coupled non-linear oscillators. In Huzihiro Araki, editor, *International Symposium on Mathematical Problems in Theoretical Physics*, pages 420–422, Berlin, Heidelberg, 1975. Springer Berlin Heidelberg.

[7] H. Sakaguchi. Cooperative Phenomena in Coupled Oscillator Systems under External Fields. *Progress of Theoretical Physics*, 79(1):39–46, 01 1988.

[8] Bidhan Chandra Bag, K. G. Petrosyan, and Chin-Kun Hu. Influence of noise on the synchronization of the stochastic kuramoto model. *Phys. Rev. E*, 76:056210, Nov 2007.

[9] Y. Kuramoto. *Chemical Oscillations, Waves, and Turbulence*. Springer, 1984.

[10] Bidhan Chandra Bag and Chin-Kun Hu. Escape through an unstable limit cycle: Resonant activation. *Phys. Rev. E*, 73:061107, Jun 2006.

[11] Peter Hanggi and Peter Jung. Colored Noise in Dynamical Systems. *Advances in Chemical Physics*, 89:239–326, January 1995.

[12] Rico Berner, Thilo Gross, Christian Kuehn, Jürgen Kurths, and Serhiy Yanchuk. Adaptive dynamical networks, 2023.

[13] Leonhard Lücken, Oleksandr V. Popovych, Peter A. Tass, and Serhiy Yanchuk. Noise-enhanced coupling between two oscillators with long-term plasticity. *Phys. Rev. E*, 93:032210, Mar 2016.

[14] Oleksandr Popovych, Serhiy Yanchuk, and Peter Tass. Self-organized noise resistance of oscillatory neural networks with spike timing-dependent plasticity. *Scientific reports*, 3:2926, 10 2013.

[15] Max Thiele, Rico Berner, Peter A. Tass, Eckehard Schöll, and Serhiy Yanchuk. Asymmetric adaptivity induces recurrent synchronization in complex networks, 2022.

[16] Rico Berner, Eckehard Schöll, and Serhiy Yanchuk. Multiclusters in networks of adaptively coupled phase oscillators. *SIAM Journal on Applied Dynamical Systems*, 18(4):2227–2266, jan 2019.

[17] Peter E. Kloeden, Eckhard Platen, and Henri Schurz. *Numerical solution of SDE through computer experiments*. Springer, 2003.

[18] Ernst Hairer. *Solving Ordinary Differential Equations I: Nonstiff Problems*. Springer, 2011.

[19] Leonhard Lücken, Oleksandr V. Popovych, Peter A. Tass, and Serhiy Yanchuk. Noise-enhanced coupling between two oscillators with long-term plasticity. *Phys. Rev. E*, 93:032210, Mar 2016.

[20] R. Graham and T. Tél. Weak-noise limit of fokker-planck models and nondifferentiable potentials for dissipative dynamical systems. *Phys. Rev. A*, 31:1109–1122, Feb 1985.

[21] J.M. Borwein, D. Nuyens, A. Straub, and J. Wan. Random walks in the plane. *Discrete Mathematics and Theoretical Computer Science*, pages 191–202, 2010.

[22] R.V. Hogg, J.W. McKean, and A.T. Craig. *Introduction to Mathematical Statistics*. Pearson, 2019.

[23] C. W. Gardiner. *Handbook of Stochastic Methods: For Physics, Chemistry and the Natural Sciences*. Springer, 2002.

[24] H. Risken. *The Fokker-Planck Equation: Methods of Solution and Applications*. Springer, 1992.

[25] Richard L. Burden, J. Douglas Faires, and Annette M. Burden. *Numerical Analysis*, chapter Error Control and the Runge-Kutta-Fehlberg Method, page 294–302. Cengage Learning, 2016.

[26] Erwin Fehlberg. *Low-order classical Runge-Kutta formulas with stepsize control and their application to some heat transfer problems*. National Aeronautics and Space Administration, 1969.