The foundations of superposition and its use in quantum walks on complex networks

Peter Gregory Lewis
Imperial College, London, Department of Physics

Submitted for the degree of Doctor of Philosophy in Physics
Except where explicitly referenced, the following work is my own.

The work in chapter 2 was published as [LJBR12], and is used under a Creative Commons Attribution 3.0 Licence. In chapters 3 and 4 simulations were performed using Python, NumPy and SciPy, and plots produced with Matplotlib.

The copyright of this thesis rests with the author and is made available under a Creative Commons Attribution Non-Commercial No Derivatives licence. Researchers are free to copy, distribute or transmit the thesis on the condition that they attribute it, that they do not use it for commercial purposes and that they do not alter, transform or build upon it. For any reuse or distribution, researchers must make clear to others the licence terms of this work.
In classical physics, it is possible in principle to predict with certainty the outcome of any measurement that we might perform on a system. Where we are not able to predict outcome with certainty, we can attribute the uncertainty to incompleteness in our knowledge about the history of the system (its pre-measurement state), or the behaviour of the apparatus we use to measure it. Quantum theory, on the other hand, describes systems where what we can find out about the pre-measurement state is significantly restricted: we are not generally able to predict with certainty the outcome of an arbitrary measurement, even with full knowledge of that system’s history. Superposition states are at the heart of this phenomenon, and do not have an analogue in classical physics. In this thesis we examine superposition states from two points of view.

Firstly, following long-running arguments about whether quantum theory can be considered complete, we examine the possibility of an underlying ‘ontological’ model of quantum theory that explains the quantum measurement statistics. We derive the first such model that has the property that a single state in the underlying model is compatible with distinct quantum states, and recovers the measurement statistics for systems of dimension greater than two.

Secondly, we examine the dynamics of continuous time quantum walks in complex networks. Discovering community structure in these networks is a useful task to be able to perform, and there are several algorithms using the classical dynamics of random walks to return that structure. We introduce a new ‘centrality measure’ based on the observable dynamics of a quantum walker, and provide an algorithm using our new measure for community detection.
I have been exceptionally lucky to have had the chance to have spent four years surrounded by people who are enthusiastic about quantum physics. In particular, I would like to thank my supervisors Terry Rudolph and Jon Barrett for their support, encouragement and tolerance of the wide range of projects that I’ve wanted to pursue. They have included foundations of quantum theory, fine art (including residencies at Yinka Sinobare’s Guest Projects and Flat Time House), fixing local democracy, learning to code and continuous time quantum walks. Some of these interests I have written about here. Almost everything I have learned about quantum theory I have learned from them, and they have been among the foremost available teachers I could imagine.

Their excellent supervision has been wonderfully complemented by the two postdocs with whom I worked most closely – David Jennings (particularly in the first part of this thesis) and Elham Ashoori (in the second part). Their expert guidance and ability to translate vague ideas into good ones has been invaluable. In addition, it has been great fun to have been part of the first cohort of the DTC – surely the best context in which to do a PhD. Finally, I’m grateful to Viv Kendon and Myungshik Kim for an enjoyable viva and their helpful and improving insights.

Outside physics, my parents Josephine and David (who sparked my interest in how stuff works) and my brothers Stephen and Andrew have been a source of continual support; I couldn’t have got to here without their help. I’ve also been lucky to have similarly supportive parents- and brothers-in-law: Maureen, Keith, Richard and James.

Most of all, I would like to thank my wife Sarah. She has been with me throughout, and without her none of this would have been possible. This thesis is dedicated to her (though for chapter 4, Oscar – never one to miss a good walk – comes along too).

Peter Lewis,
London, 1st October 2014
Contents

Summary 5

1 Introduction 9
   1.1 Operational primitives ................................. 12
   1.2 Quantum theory: a brief outline ....................... 15
   1.3 Miscellaneous concepts ............................... 23
   1.4 Plan .................................................. 27

2 Quantum states embedded in a realistic theory 29
   2.1 Motivating the epistemic view of quantum states ....... 32
   2.2 Ontological models .................................... 35
   2.3 Determinism in ontological models ..................... 43
   2.4 Quantum states as states of knowledge ................. 46
   2.5 Other $\psi$-epistemic OMs ............................ 58
   2.6 $\psi$-epistemic OMs and simulating quantum teleportation .... 62
   2.7 Summary ............................................. 65

3 Networks, graphs and communities 67
   3.1 Graph representation of networks, partitions and communities .... 70
   3.2 Dynamics in networks .................................. 75
   3.3 Using CRW to investigate community structure in networks ...... 82
   3.4 Benchmark graphs for community detection ............... 86
   3.5 Evaluating and comparing algorithms ................... 97
   3.6 Summary ............................................. 101

4 Community detection with superposition states 103
   4.1 Quantum flashiness .................................... 106
   4.2 Flashiness in simple graphs ............................ 110
   4.3 Flashiness in benchmark graphs ......................... 122
This thesis is about the central objects of quantum theory – quantum states. Along with the rest of the machinery of quantum theory, quantum states provide us with a description of nature unrivalled in its success across an exceptionally wide range of phenomena ([Ein05, You04, Ein25, BB84], and somewhat self-referentially [Fey82]).

Yet the character of quantum theory is markedly different from pre-quantum (or classical) theories that have seen success in the past. The characteristic differences have caused controversy among physicists in the way that quantum theory should be interpreted: indeed, as Feynman’s quote suggests, on the face of it quantum theory seems quite arbitrary. The features prompting the controversy are particularly apparent at the boundary between where physical scenarios require quantum theory, and where the classical description is ‘good enough’ – usually macroscopic situations. Unsurprisingly, much discussion has centred on the operation where this boundary usually occurs. This is the point of an experiment where we uncover (classical) information about systems: measurement.
One of the first points of departure from classical physics is that the outcome of measurement on a quantum system cannot generally be predicted in advance; a phenomenon seen as characteristic of superposition states. Perhaps this is not so unusual: for classical systems we occasionally lack complete knowledge of a system’s state of affairs, or the tools we use for measurement are noisy or imprecise. In these cases we also cannot predict with certainty the outcome of a measurement. But for quantum theory the outcome is generally unpredictable even in the case where we know everything about the measurement and the apparatus used to perform it, as well as precisely the way in which the system we are measuring was prepared.

The difference is that for classical systems, there is in principle a single outcome of any measurement that occurs with certainty; our inability to predict the outcome that arises due to a lack of knowledge about the experiment, be it in the way that the system was before the measurement, or the precise behaviour of the measuring device itself. For quantum systems, the unpredictability is apparently fundamental. That is, no experiment has ever demonstrated that one can predict the outcome of measurement with greater chance of success than quantum theory ‘allows’.

Nevertheless, a fundamental restriction on what can be predicted based purely on abstract mathematics is physically unsatisfactory: one may justifiably ask whether the restriction can be established as a consequence of a more physical axiom. As we shall see, one recent approach can not only explain the unpredictability of measurement outcomes, but also some of the peculiar features of quantum theory. In chapter 2 we provide the first example of a model under this approach that recovers the quantum measurement statistics in any finite dimension.

The branch of science that considers how we should reason with incomplete knowledge about classical systems is information theory. Information theory allows us to make formal statements about how noisy our measurement devices might be, how much information is encoded in a particular system and how systems relate to each other (answering questions such as ‘if I know X about Y, what do I know about Z?’ etc.). An analogous branch of science considers the same types of quantity for quantum systems. Quantum information theory has allowed us to frame many aspects of foundations of quantum
theory in a rigorous way. Indeed, one can perform experiments (e.g. [ADR82]) based on information processing tasks that tell us about which physical axioms are compatible with each other.\footnote{For example, Bell's theorem [Bel64] tells us about the relationship between free will, local causality and the existence of an underlying reality.}

The relatively new tools of quantum information theory at worst allow us to phrase the controversy described above in a precise way, and at best allow us to learn new, fundamental parts of nature’s character. But perhaps more importantly they indicate that there are deep similarities between the key objects of quantum theory – quantum states – and our knowledge of classical systems. One good way of looking at the model we present in chapter 2, particularly in light of other recent work, is as providing a landmark for what can be explained when we make some assumptions about these similarities, and disallow others.

Another area of science that has seen growing attention from physicists even more recently than quantum information is research in the structure of complex networks. Complex networks are ubiquitous: we are constantly surrounded by systems that can be accurately described as a collection of systems with local interactions. Recent dynamical approaches to investigating the mesoscopic qualities of these systems have opened up a wide range of technological possibilities, as well as being of intellectual interest. The mesoscopic characteristics of networks’ structure are an interesting case in physics in general. Typically, a physical theory either describes precisely the behaviour of a system (for systems small enough for such a description to be worked out), or systems are large enough (in terms of their number of components) that we are able to resort to a statistical limit. The regime where one is able neither to describe the behaviour exactly, nor to justify taking a statistical limit is relatively unexamined.

The overwhelming majority of work in this area, however, has involved the dynamics of classical systems; usually classical random walkers. There are many scenarios where a classical treatment of systems is not clearly justifiable, and where already experimental evidence is suggesting that quantum phenomena are in play.

In chapter 4 we provide a new means for uncovering structure in complex networks
based on a continuous time ‘quantum walk’. This process is analogous to a classical random walk, the usual classical dynamical approach, except that for quantum walks superposition states play a key role. Indeed, the dynamics of a quantum walk is deterministic, so all of the uncertainty arising in measurements of the position of the walker in the graph can be attributed to the superposition states. Importantly, aspects of the dynamics are often observable in cases where the network itself is hidden. Our extension to quantum walks takes this restriction seriously, and requires access only to the experimentally accessible aspects of the walk: i.e. measurements in the vertex basis.

Before we present our result on structure detection with quantum walks, in chapter 3 we review the graph representation of networks and community structure within those graphs, classical random walks and quantum walks. We also examine some of the dynamical approaches to discovering community structure and evaluating a particular partitioning of a graph with respect to the that structure. Finally, in order to draw together these two studies of superposition states, we provide a summary in chapter 5.

1.1 Operational primitives

Throughout this thesis we shall make use of an operational approach to describing experiments, recently advocated by [Har01, Spe05, Bar07]. This means that we regard an experiment as a set of instructions and ingredients that lead to some set of outcomes: i.e. if an experimentalist follows such-and-such a set of instructions, then there is some set of possible outcomes and each will obtain with some probability. A good way to think about such an arrangement is as a set of devices with various inputs and outputs that can be composed in certain ways to form an experiment. Each device has a certain type of input or output that restricts the way in which it can be composed; indeed, there is an active research area focusing on the characteristics of general operational theories with various restrictions (for example [CDP11, CDP10, Har10]).

Quantum theory in particular provides a description of experiments that fits this framework: it is an operational theory. The instructions and ingredients that quantum theory describes boil down to three operations or procedures, each of which can be thought of
as a device, and that can be operated in concert to form an experiment.

– A preparation device is always the first component of an experiment. The preparation device has a control where an experimentalist can configure the device, influencing the way that the experiment behaves. From an informational perspective, this constitutes an operation where information is added to the experiment. The configuration is the state of the operational theory, so we say that the preparation device prepares a state from the theory. In order to initiate a ‘run’ of the experiment, we have some notion of activating the preparation device: causing it to prepare a system.

In common with classical information theory, in quantum theory we are usually agnostic about the particular realisation of the operational state. Although, strictly speaking, it forms a component of what we know about the preparation, our experience indicates that a wide range of types of system have similar behaviours in experiments: the operational states are fungible. So we regard the operational state of the system as containing all of the relevant information about the system’s preparation.

– A transformation device takes as input a prepared state and a setting (again, an instance of information into the experiment), and outputs another state. Note that preparation devices and transformation devices can in principle be permanently coupled. Since the combined device takes as input a setting and outputs a system, it is indistinguishable from a preparation device set to the the appropriately transformed state. In chapter 2 this indistinguishability allows us to mention transformations only in passing.

In chapter 4 – where we discuss quantum walks – the transformation represents the structure over which the walk evolves, and includes a time setting (i.e. the system evolves under the transformation over a period of time). Our main purpose in that chapter will be to determine the structural qualities of the transformation device when a significant restriction is placed on the measurements that we are allowed to perform. Note that thinking of the preparation and transformation devices together is analogous to using the Schrödinger picture in the quantum formalism.
A measurement device takes as input a state and a setting, and outputs one of a spectrum of readable outcomes: i.e. it is the operation where information is extracted from the experiment. We can think of the setting as a control for which aspect of the state – usually called the observable – we would like the outcome to inform us about. So measurement devices are our means of learning about systems that have been prepared.

In the same way as a transformation can be combined with a preparation device, a transformation device can also be combined with a measurement device, equivalent to a measurement device with a different setting. In this case, where there is a time setting, the analogous treatment is the Heisenberg picture in quantum theory.

In addition, any sharp measurement (see below) can be effected using a transformation device with an appropriate setting and a measurement device that measures in a single, known basis. So the information that we put in to a measurement device is strictly speaking unnecessary if we allow arbitrary transformations.

We provide a schematic for the operational framework in figure 1.1.

Particularly useful in the first part of this thesis, the operational approach allows us to investigate subtle aspects of quantum theory without having our analysis coloured by interpretational baggage. Further, we can explore the operational phenomena in a context that admits more general theories. This gives us hints about the precise structure of the physical constraints that give rise to observed quantum phenomena. Our aim in chapter 2 is to map out the structure of physical assumptions that can give rise to the measurement statistics we observe in quantum theory.

In the later chapters, taking the operational approach means that our results are not contingent on a particular interpretation of quantum theory.

We are now in a position to stitch the mathematical formalism of quantum theory to the operational framework. Importantly, we can do this without making any commitment about the nature of quantum states: they are, so far, simply descriptions of operations. This does not rule out the possibility of quantum states having a deeper, more fundamental role, but does give us the freedom to explore that role with some rigour.
CHAPTER 1. INTRODUCTION

1.2 Quantum theory: a brief outline

We start the groundwork by linking the elements of the operational approach to the mathematical objects that form quantum theory. Throughout this thesis our investigation will be limited to the finite dimensional, non-relativistic case which simplifies things somewhat. We assume a basic familiarity with linear algebra (for a primer in the vein of quantum information, check out [NC04]).

Preparations

A preparation device is associated with a particular dimension of system – for example in the case of the polarisation of a photon, the dimension \( d = 2 \). The dimension tells us the maximum number of states of the system that can be mutually distinguished with certainty in a single measurement. That is, for a single use of a measurement device,
the maximum number of preparation device settings that can be unambiguously distin-
guished is \(d\). A system of dimension \(d = 2\) can be used as a qubit, the most basic unit of
quantum information and analogous to a bit (or cbit) in classical information theory.

A special subset of the possible settings for the preparation device is the set of so-called
‘pure’ quantum states. For each such configuration of the device there is a corresponding
quantum state \(|\psi\rangle\) described by a ray in the complex Hilbert space with dimension \(d\):
\(|\psi\rangle \in \mathbb{C}P^{d-1}\). Pure state preparations are those that behave most predictably: that is, for
any pure state there exists at least one measurement that can be performed where we can
predict the outcome with certainty, given knowledge of the preparation.

Many different types of system (e.g. photon polarisation, electron spin, arbitrary two-
level systems) come with the same dimension, and the collection of possible settings for
the preparation device is identical. The same is true of the measurements that we intro-
duce below. Although the experimental apparatus may be different for measurements on
different types of system, the representation in the formalism is the same. This motivates
us to gather all preparation devices preparing states with the same dimension into an
equivalence class, allowing us to remain agnostic about the exact realisation of an exper-
iment and focus on the experiment’s general characteristics.

Pure quantum states can be expressed as a ‘ket’ in Dirac notation: e.g.

\[ |\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle + \ldots + \alpha_{d-1}|d-1\rangle. \tag{1.1} \]

The \(\alpha_i\) are generally complex, have norms in the range \(|\alpha_i| \in [0, 1]\), and the sum of their
squares is \(\sum_i |\alpha_i|^2 = 1\). The integer \(|i\rangle\) correspond to a particular orthonormal basis of
the Hilbert space. This same state can be expressed as a trace-1 positive operator on the
d-dimensional Hilbert space \(\mathcal{H}^d\) as \(\rho_\psi = |\psi\rangle\langle\psi|\). This means we can safely ignore a global
phase so \(|\psi\rangle \sim e^{i\theta}|\psi\rangle\). However, the relative phase is important \(e^{i\theta}(\alpha_0|0\rangle + \alpha_1|1\rangle) \neq \alpha_0|0\rangle + e^{i\theta}\alpha_1|1\rangle\). This relative phase can lead to interference phenomena between differ-
ent ‘components’ of the state in the formalism. These interference phenomena are par-
ticularly important in the second part of this thesis, giving rise to observable differences
between the classical and quantum dynamics.
Where two states can be distinguished with certainty, they are represented by orthogonal vectors in the Hilbert space: we write the inner product of two such states as $\langle \psi | \psi^\perp \rangle = 0$. Where $|\psi\rangle \in \mathcal{H}^2$ there is exactly one orthogonal state $|\psi^\perp\rangle$, though in general there is a continuum of orthogonal states occupying a $d - 1$-dimensional subspace orthogonal to $|\psi\rangle$.

Where two preparation devices prepare states that are non-orthogonal, they cannot be distinguished with certainty by a ‘single-shot measurement’ (i.e. single use of a measuring device). Having said that, pure states are the only quantum states having the property that there is in principle a set of measurements that we can perform where we are able to predict the outcome that will obtain with certainty. This is not the case for mixed states – the other type of quantum state. We return to the issue of distinguishability below.

Aside from the pure states, the alternative possible settings for a preparation device in $d$ dimensions are referred to as ‘mixed’ states. Whereas a pure state is in a sense the fullest description that we can provide for a quantum system, a mixed state that for a single system contains an element of classical uncertainty.

For example, a mixed quantum state can arise in cases where we lack comprehensive information about the procedure that was used to prepare the system. This lack of information can be under our control, such as a scenario where one of several possible pure states is prepared according to the value of a (classical) random variable, such as a coin toss. Or the lack of information could be external to our control, for example in the case where a preparation device does not behave entirely predictably (i.e. noise). In either case we represent the state as a classical mixture of pure state density matrices:

$$\rho = b_0 |\psi_0\rangle \langle \psi_0 | + b_1 |\psi_1\rangle \langle \psi_1 | + b_2 |\psi_2\rangle \langle \psi_2 | + \ldots$$

In this case, and in contrast to the $\alpha$’s above, the coefficients are real, positive and sum to 1; they meet the requirements of a classical probability distribution. Note also that the $|\psi_i\rangle \langle \psi_i |$ need not be orthogonal, and that there can generally be arbitrarily many of them.

Both pure and mixed states can be represented as density operators [NC04]. For qubit states, an important representation of the space of density operators is the Bloch sphere. Any density operator corresponding to a quantum state can be decomposed into four
components
\[ \rho = \frac{1}{2} [x_0 \mathbb{1} + x_1 X + x_2 Y + x_3 Z] \] (1.2)

where the \( x_i \) are real and the \( X, Y, Z \) are the traceless Pauli matrices. Since any normalised quantum state has trace \( \text{Tr} \rho = 1, x_0 = 1 \). In order that the density operator remains positive the other \( x_i \) take values in the range \([-1, 1]\). The real vector \( \hat{x}_\rho = (x_1, x_2, x_3) \) is called the \textit{Bloch vector} for the qubit density matrix, and since \( |\hat{x}_\rho| \leq 1 \) (the positivity requirement generalised) the set of Bloch vectors corresponding to every qubit state forms a sphere. The mixed states of the qubit occupy the interior of the sphere, with \( |\hat{x}_\text{mix}| < 1 \). This visualisation of the qubit state space will be useful in chapter 2.

One feature unique to \textit{classical} systems, on the other hand, is that there is exactly one way to decompose a mixed classical state into pure states [Bar07]. This is because the state space for classical systems is a simplex. In quantum theory, a mixed state can be prepared using appropriate concentrations of any set of states whose convex hull includes the mixed state. Further, this principle carries over to the geometrical representation of the Bloch sphere: a mixed state Bloch vector can be prepared by a convex combination of any set of Bloch vectors whose convex hull includes the state we wish to prepare.

This statement is a direct consequence of the existence of superposition states, and so possibly gives some immediate motivation to study the interplay between mixing – as described,\(^2\) a purely classical process – and superposition. In chapter 4 this difference is precisely what separates classical random walks from the continuous quantum walks that we study.

In addition, we are able to combine different preparations so that we can perform joint transformations and measurements on quantum systems. That is, we can use two preparation devices that feed into the rest of an experiment, and indeed we can also generally split the parts of a system in order to feed each into different measurement devices. In these cases, the state spaces and states are combined with a tensor product. Our use of the tensor product structure is basic, but for completeness we direct the curious reader to

\(^2\)Mixed states can also occur when we examine a subsystem of some larger system: for example one qubit of an entangled pair, whose ‘reduced state’ is given by ‘tracing out’ the second qubit using a partial trace operation. The corresponding reduced state has the same observable properties as if that same state had been prepared directly and in isolation.
[NC04] for a full exposition. It is worth noting that the use of a tensor product to combine quantum states, rather than the Cartesian product that would be used for classical states, is involved in many of the peculiar phenomena associated with quantum theory.

Transformations

A transformation device is associated with a map from quantum states to other quantum states in a particular dimension. In order to maintain the positivity of states, and consequently the positivity of the outcome probabilities we introduce below, a valid transformation must itself be a positive operator. A special kind of transformation is that that takes all pure states to other pure states. Such a transformation takes the form of a unitary operator on $\mathcal{H}_d$, and is operationally characterised as a process in which the system is closed (has no interaction with any external system, such as a source of randomness). In the case of unitary evolution over time $t$, we have $\rho(t) = U(t)\rho(0)U^\dagger(t)$, where $\rho(0)$ is the prepared state.

If the transformation involves the non-trivial interaction of the system of interest with an external system then the transformation on the system of interest is generally not unitary. In order to recover unitary dynamics, we must 'zoom out' sufficiently far to include enough of the external system that the dynamics of the combination are unitary:

$$T(\rho \otimes \sigma) = U(\rho \otimes \sigma)U^\dagger$$  \hspace{1cm} (1.3)

where we have combined quantum states in the usual way, using a tensor product. In order to recover the dynamics of the target system itself, we use the 'partial trace' over the external system initially in state $\sigma$. This gives

$$T(\rho) = \text{Tr}_\sigma(U\rho \otimes \sigma U^\dagger)$$
$$= \sum_i M_i \rho M_i^\dagger$$  \hspace{1cm} (1.4)

where the $M_i$ are the Kraus operators associated with the unitary $U$ from equation 1.3. We mention evolution of a quantum system whose evolution is influenced by an external system only in passing in what follows, and so refer the interested reader to [NC04] for
a more detailed description. It is important in what follows, however, to appreciate that there is a notion of combining quantum systems and of describing the dynamics in terms of its effect on different components of a system.

One particular transformation is central to our discussion in the work on networks in chapter 4. We use a closed, continuous time process in the network where the shape of the network is encoded in an Hamiltonian operator. The Hamiltonian is the generator for evolution of the system, and as such we can write the general evolution for a particular network as the unitary

$$U(t) = e^{-iHt}.$$ (1.5)

We use this transformation to give the quantum state corresponding to a process which has been evolving in the network for time $t$. Our aim in chapter 4 will be to extract the mesoscopic structure of $H$ from a restricted set of allowed measurements – i.e. measurements of the walker in the vertex basis of the graph.

**Measurement**

Finally, we complete the triptych by describing measurement. In spite of the differences we have already seen, measurement of quantum systems represents perhaps the most apparent departure from the *modus operandi* of classical physics. In classical physics, a measurement is modelled as a standard interaction between two systems: the measuring device and the measured system. In quantum theory, measurement interactions are represented in a manner completely different from the other kinds of transformation that we referred to above.

Specifically, each observable $A$ of a system – that is, aspect of the system that can be measured – is represented by a set of $n$ measurement outcomes $A_0, \ldots, A_{n-1}$. The probability for the $k$th outcome to obtain is given by the Born rule:

$$\Pr(k|A, \rho_\psi) = \text{Tr}(A_k\rho_\psi) \in [0, 1].$$ (1.6)

Immediately this leads to one of the most troubling properties of quantum theory: that al-
though as we have seen closed interactions are deterministic (i.e. unitary), measurements have more than one possible outcome for the same state. This is in contradiction to the description we gave above for joint systems in general, and our (reasonable) assumption that we should be able to describe the interaction of a measuring device and measured system in this way. The departure from the standard description of unitary dynamics hints that there is more to the quantum state than quantum theory as described so far specifies. This is a theme that we will pick up in chapter 2 when we describe how the probabilities of quantum measurement could arise in a deterministic setting.

In contrast, for classical systems there is exactly one ‘full’ measurement that can be performed on a system of a given dimension. For example, an observation of a rolled die can have one of six possible outcomes, corresponding to the six faces of the die. Any other measurement (e.g. odd/even, prime/not prime) is simply a coarse graining of the full measurement. This principle extends to higher dimensions, and indeed the continuum. Although typical apparatus that we use to gain information from classical systems tend to coarse-grain the full description (i.e. grouping several pure states into a single outcome), we are in principle always able to discover the full classical state from a single measurement. It is not possible to discover the full quantum state of a system from a single measurement.

Broadly analogously to the ‘full’ and coarse-grained classical measurements, there are two types of quantum measurement.

The first is often referred to as a *sharp, projective* or *projective value measure (PVM)* measurement. PVMs (and, in general, rank-1 observable operators) are the most useful kind of measurement in the case where we are tasked with distinguishing a pair of states with certainty. This is because there is a set of states in the subspace orthogonal to the rank-1 projector for which that outcome can *never* occur.

Each outcome of a complete PVM is associated with a projector onto a state from a mutually orthogonal set. So for a complete PVM $\Pi$ on a system with a state in $\mathcal{H}^d$, there are $d$ orthogonal measurement outcomes $\Pi_0, \ldots, \Pi_{d-1}$, with each $\Pi_k$ having the form $\Pi_k = |k\rangle\langle k|$; i.e. projecting onto an associated state $|k\rangle$. The full set of associated states that a PVM projects onto forms a mutually orthogonal set of states that spans the Hilbert space.
space associated with the measurement. Further, \( \sum_k \Pi_k = 1 \) and the \( \Pi_k \) are all positive operators. These two conditions, when taken with the set of possible states described above, give us that for each measurement exactly one outcome occurs for any particular run of the experiment, and that all of the outcome probabilities are positive. Finally, \( \Pi_k^2 = |k\rangle\langle k| k\rangle\langle k| = \Pi_k \).

An important and characteristic property of PVM measurements is that they are repeatable. If we perform a PVM in such a way as the system is not destroyed by the measurement, then the post-measurement state of the system will be the state projected onto by the outcome that occurred. If we repeat the measurement, then the probability to observe the same outcome again is equal to 1; the state after the first measurement is orthogonal to all other projectors comprising the measurement.

The second type of measurement is called a positive operator value measure (POVM), which can be implemented in two different ways. Firstly, a POVM can arise in the case where the measurement device adds noise: either by performing the intended PVM measurement then sending the result through a noisy classical channel, or by performing a PVM other than that we had intended. Secondly, a POVM can be effectively carried out on a part of a joint system when we perform a PVM on the full system. That is, a reduced state is to a joint state as a POVM element is to a PVM element on a larger system.

Each operator making up a complete POVM is conventionally called an effect, and for a POVM measurement \( E \) we write \( E = \{E_k\} \), the set of effects. Just as with PVMs, the effects corresponding to each outcome are positive, and the sum over all effects making up a POVM measurement is equal to 1. It is not, however, the case that \( E_k^2 = E_k \); the effect operators are not trace-1. This, combined with the summing to identity condition, implies that POVMs often require more than \( d \) outcomes. Nevertheless, they are unable to reveal more information about the pre-measurement state of a quantum system than PVMs: this bound is due to Holevo [Hol73].

In order to calculate the probability distribution over the outcomes of a particular measurement given a particular input state, we simply combine the appropriate measurement operator with the state in the way prescribed by equation 1.6.
1.3 Miscellaneous concepts

Entanglement. For the joint states that we described in passing above, sometimes the separate parts of the system are distributed in space; indeed, they could be in different labs. In a couple of places in the periphery of our main results we will come across this kind of scenario. Typically we will consider bipartite states where a system is partly held by one party (Alice) and partly held by another (Bob). In such a situation, both parties have access to their own measuring devices and can perform measurements on a part of the joint state. Further, each obtains results from the measurement according to a distribution over the outcomes of the observable that they chose determined by the reduced state, as described above.

Since by assumption Alice cannot perform measurements on Bob’s part of the joint system (and vice versa), the situation is effectively a restricted treatment of the situation where a single party is able to perform any measurement on the whole system. A large area of research considers what possible information processing tasks Alice and Bob can achieve with access to an entangled resource, including the practical tasks that are realised by quantum key distribution algorithms (such as [BB84]). Note that this kind of distributed situation in classical physics can be fully described by a Cartesian product of the two components of the system.

The description in quantum theory is somewhat more complicated. Specifically, the joint state is a component of a tensor product of the individual state spaces:

$$\mathcal{H}_{AB} = \mathcal{H}_A \otimes \mathcal{H}_B.$$ (1.7)

In order to recover a description of the ‘reduced state’ of one of the party’s components, we employ the partial trace over the joint system:

$$\rho_A = \text{Tr}_B(\rho_{AB}).$$ (1.8)

Importantly, it is generally not the case that the joint state $\rho_{AB}$ is given by the tensor
product of the reduced states:

\[
\rho_{AB} \neq \text{Tr}_A(\rho_{AB}) \otimes \text{Tr}_B(\rho_{AB}).
\] (1.9)

A consequence of this is that there is information contained in the joint state that is not contained in the individual parts. This extra information is extractable in the correlations that exist between the outcomes obtained by Alice and Bob. The ‘strength’ of these correlations is the key element of Bell’s theorem, perhaps the most famous result in foundations research. Indeed, one can characterise a 2-qubit state in its entirety by considering the set of steerable states for one of the qubits, and the reduced state of the other [JPJR14].

Of course bipartite classical systems can be correlated with each other. In [Bel64] (Bell) and subsequently [CHSH69] and [Cir80] the authors find a bound for the extent to which the outcomes of measurements on classical systems, assuming local causality, can be correlated. Local causality is the property of a physical theory that the full description of a system is determined only by events in the past light-cone of that system at the time of measurement. That is, if we know every event in a particular slice of the past light-cone of the system, assuming local causality means that we can give a full description using only that information. In quantum theory, the apparent influence of events outside this history is non-trivial: the bound on correlations is broken by some entangled, multipartite quantum states.

Breaking this bound demonstrates that no locally-causal ‘hidden variable model’\(^3\) can recover the quantum measurement statistics. Of course it is worth noting that the same is true for quantum states: that quantum states are fundamentally ‘non-local’ objects. The study of entanglement and other similar qualities of quantum systems is a large field that, for the purposes of this thesis, we can appreciate from afar. For a more detailed account of some features of entangled systems, the reader could consult [HHHH09]. But most importantly for motivating our work, Bell’s work and the work that follows provided the first example of a formal and experimentally testable theorem in foundations.

\textit{State update.} Sometimes a system that has been measured is available for further interro-

---

\(^3\)This term contains some more assumptions, which we discuss in detail in the next chapter.
CHAPTER 1. INTRODUCTION

gation by another measuring device – essentially as a preparation in a new experiment. For the distributed systems that we were discussing immediately above, when Alice performs a measurement Bob’s system is still available for measurement. In these situations, a pertinent question is what quantum state we should use to describe the system that emerges from the measurement. We say that the system undergoes ‘update’ following a measurement: a post-measurement state with a new set of values for each observable. The update rules are provided for us by quantum theory, as describe in [NC04].

Principally, when a system undergoes a PVM, the consequent system ‘collapses’ to the state associated with the occurring outcome’s projector. That is, if we have a system prepared in state $|\psi\rangle$ on which we perform measurement $\Pi = \{\Pi_k\}$ obtaining the outcome corresponding to $|k\rangle\langle k|$, then the resultant post-measurement state is $|k\rangle$. Where we perform such a measurement but do not find out the outcome, the resultant state is a classical mixture – with coefficients given by the Born rule – over the eigenstates of the measurement operators.

Where the measurement is performed on part of a bipartite system, we observe a possibly spooky (the question is addressed, though not explicitly answered, in [Pus13]) phenomenon sometimes called steering, whereby the component of the state that is not subject to measurement undergoes update. A comprehensive discussion of state update can be found in [NC04]. The state update rules place some restrictions on the behaviour of the models that we use to recover quantum measurement statistics in chapter 2.

*Distances in quantum state space and distinguishability.* We finish this section by discussing the relationships between different quantum states in a particular dimension. We have seen that where a set of states is mutually orthogonal they can be unambiguously distinguished from each other in a single run of an experiment. That is, we can design a measurement such that the outcome of the measurement reveals unambiguously which of the set was the prepared state.

Where a set of states is not mutually orthogonal, we are unable to distinguish them with certainty. As an example, consider a pair of distinct, non-orthogonal quantum states $|\psi\rangle$ and $|\phi\rangle$. Such a pair of states spans a qubit subspace, so it is sufficient to restrict ourselves to dimension $d = 2$ even though the full states may occupy a higher dimension. If we
have a preparation device that we know is configured with either the setting $|\psi\rangle$ or the setting $|\phi\rangle$, what is the best strategy for establishing what the setting is?

There are a number of possibilities for answering this question, depending on whether we wish for the outcomes to be ($|\psi\rangle$, ‘don’t know’) or ($|\phi\rangle$, ‘don’t know’) or ($|\psi\rangle$, $|\phi\rangle$, ‘don’t know’), and whether we are interested in minimising the frequency of ‘don’t know’, or maximising the frequency of a definite answer. Taking the first of these sets of outcomes, the best strategy involves projecting onto the orthogonal states $|\phi\rangle$ and $|\phi\rangle^\perp$. The important figure – determining the proportion of runs for which we expect to be able to output $|\psi\rangle$ – is given by the probability to observe the $|\phi\rangle^\perp$ outcome given the preparation device was set to $|\psi\rangle$:

$$\Pr(\phi^\perp|\{\Pi_\phi, \Pi_{\phi^\perp}\}, \psi) = |\langle \phi^\perp | \psi \rangle|^2.$$  \hspace{1cm} (1.10)

This equation can alternatively be written in terms of the Bloch vectors for the two states [NC04]:

$$\Pr(\phi^\perp|\{\Pi_\phi, \Pi_{\phi^\perp}\}, \psi) = \text{Tr}(\rho_\psi \Pi_{\phi^\perp}) = \frac{1 + \hat{\phi}^\perp : \hat{\psi}}{2}. \hspace{1cm} (1.11)$$

Following the geometrical interpretation of this equation, we find that there is a relationship between the rate at which we can succeed at this kind of game and the geometrical representation of the states in question in the Bloch sphere. Namely, where two states are ‘close together’ in the Hilbert space (that is, they have large mutual overlap), they are hard to distinguish and our success rate will be low. On the other hand, where the states are further apart (for example nearly orthogonal), the states are easier to distinguish and our success rate increases. These two cases are related in the qubit case by the angular distance between the Bloch vectors:

$$|\langle \phi^\perp | \psi \rangle|^2 = \frac{1 + \hat{\phi}^\perp : \hat{\psi}}{2} = \cos^2 \theta(\hat{\psi}, \hat{\phi}^\perp) \hspace{1cm} (1.12)$$

where $\theta(\cdot, \cdot)$ is the angular distance between the two states. It turns out that in general the success rate is related to the distance under the Fubini-Study metric on the quantum state space. For pure qubit states this gives us this angular distance $\theta = D_{\text{FS}}$ between the
corresponding Bloch vectors. In general, for systems of arbitrary dimension:

$$\cos D_{FS}(|\psi\rangle, |\phi\rangle) = \sqrt{\langle \psi | \phi \rangle \langle \phi | \psi \rangle} = |\langle \psi | \phi \rangle|.$$  \hspace{1cm} (1.13)

In chapter 2, we use the triangle inequality associated with this metric in order to bound the rate at which non-orthogonal states can be distinguished from each other even when we do not use the ‘optimal’ measurement. This allows us to construct a set of states in our model for which we can guarantee that a particular outcome of any measurement has greater than 0 probability to obtain.

### 1.4 Plan

We use the above groundwork to explore superposition from two points of view.

Firstly, we consider how the probabilities associated with measuring superposition states could arise. Further, is it possible that the explanation for the probabilistic nature of quantum theory could go some way to making its more interpretationally problematic aspects more natural? We review the ontological models formalism in order to show a few ways in which quantum theory – taken as an operational theory – can be embedded in an objectively realistic theory. Each of these embeddings has certain properties which help us untangle some of the more counter-intuitive aspects of the theory, such as (lack of) determinism in measurement. Having motivated its importance, we introduce the first example of a so-called ‘\(\psi\)-epistemic’ ontological model that recovers the quantum measurement statistics in any dimension.

Secondly, we put superposition states to work. It is well known that classical dynamical processes can be used to discover structure in graph representations of complex networks. In chapter 3 we provide a review of this area of research. Then, in chapter 4, we will examine the characteristics of continuous time quantum walks on these networks, and introduce a method that reveals community structure within the networks using a new centrality measure that we call ‘flashiness’.
In both parts we take a practical, operational approach to quantum theory in which we focus on the measurement statistics that can be established through relatively simple (in terms of the formalism) laboratory procedures. In the first section, this prevents us from straying into heuristic or aesthetic aspects of the interpretation of quantum states. In the second section, it helps us to maintain a pragmatic approach to what can be a fuzzy and heuristic field.

Finally, in chapter 5, we draw together the threads from these two disparate fields.
Quantum states embedded in a realistic theory

The work contained in this chapter was published as [LJBR12], where it was selected as Editor’s Choice.

Superposition states are strongly associated with the fact that we are – in general – unable to predict with certainty the outcome of measurements on quantum systems. Since the mathematical formalism of quantum theory does not provide us with physical motivation for why this unpredictability occurs, a number of candidate explanations have been proposed by various (groups of) physicists over the years. Like any interpretation of a physical theory, these fall into two broad camps.

Firstly, realist approaches assume the existence of an underlying reality that in some sense physics is ‘about’. That is, the mathematical formalism is, directly or indirectly, our way of representing the happenings at the underlying level. A good example where this kind of approach seems natural is in Newtonian mechanics where we can understand billiard balls having certain positions and momenta, and these real properties being represented
by mathematical objects in the formalism. These properties, we assume, would be instantiated by the balls even if they were the only objects in the universe. Where we have incomplete knowledge of a classical state of affairs, we use Liouville mechanics to describe the dynamics of our states of knowledge, represented as probability distributions over the classical state space. To use the terminology of Spekkens [Spe14], Liouville mechanics is the statistical theory for the underlying theory of classical mechanics. We return to Liouville mechanics below.

The second approach does not assume the existence of such a reality, and regards physics as tool with which we can describe the phenomena that we observe. In some sense this is the less risky approach, and for those who wish to avoid any kind of philosophical discussion about what physics represents it provides a refuge. As a disadvantage, much of the wonder of quantum theory simply disappears: it is not clear why we should be mesmerised by non-local causation, for example, if there are no objectively real objects taking part in causal relations. The operations (including measurement) are treated as primitives, and the theory is successful if it provides the correct description of those operations and nothing more.

We might argue that while the realist position is the most satisfactory in the sense that it represents a natural evolution of the classical way of doing physics, the anti-realist approach is the one most compatible with our prejudices about how the world should be. That is, if the strange qualities of physical systems described by quantum theory are really true – and few people seriously doubt that they are – then the best thing is if the strangeness exists only in the minds of physicists. The anti-realist approach takes a purely operational attitude towards the machinery of quantum theory. The role of quantum states in this setting is as a convenient summary that contains all of an agent’s degrees of belief about the behaviour of measuring devices that are composed with the relevant preparation. The view has recently been strongly advocated by [CFS02] and branded ‘Quantum Bayesianism’, recalling the Bayesian approach to probability in classical physics.

In this chapter we explore whether or not the strange properties of quantum theory can exist in the minds of physicists while at the same time a ‘traditional’ but possibly undis-
coverable dynamics takes place at the real, underlying level. That is, can we take an anti-realist approach to quantum states in particular, but a realist approach to physics in general?

Assuming the existence of an underlying reality, there are three possible options for the role of quantum states. These were set out explicitly by [HS10], and we follow their terminology:

– Quantum states are a complete description of the underlying reality – that is, there is no more informational content to physical systems than is contained in their quantum state. Theories of this form are called ψ-complete.

– Quantum states are an incomplete description of the underlying reality, and must be supplemented with extra information (i.e. extra ‘hidden’ variables) in order to provide a complete description of the underlying reality. Theories of this form are called ψ-supplemented. Perhaps the most famous example is Bohmian mechanics [Boh52].

– Finally, quantum states are an incomplete description of the underlying reality, and in fact we could justifiably use several distinct quantum states to refer to the same state of reality. Theories of this form are called ψ-epistemic.

It is the last of these three options that brings the advantages of the anti-realist position to realist approaches. Before presenting some more formal arguments in favour of taking a ψ-epistemic view of quantum states, it is worth noting that this position has a long history of illustrious advocates [EPR35, Pop67, Bal70, Pei79, Jay80, GMHG+99, CFS02, Spe07]. Indeed, it has been argued that the third of the options outlined above is the one that Einstein explicitly advocated [Ein, How06]:

But then for the same [real] state of [the system] there are two (in general arbitrarily many) equally justified Ψ [quantum states], which contradicts the hypothesis of a one-to-one or complete description of the real states.
This incompleteness can be understood as an example of the approach where the quantum state represents an agent’s information about the state of reality, but where the quantum state is not a part of reality itself: i.e. the $\psi$-epistemic approach.

Below we shall demonstrate by example that a so-called ‘$\psi$-epistemic ontological model’ can explain the probabilities given by quantum theory (realist toy models for operational theories, or ontological models, are described in detail below). Ours is the first such explanation for systems of dimension $d > 2$. We will focus on ‘prepare–measure’ experiments: experiments containing a preparation device set according to a quantum state, and a measurement device set to perform a particular measurement (associated with a set of measurement operators). First, however, we provide motivation for what is on the face of it a convoluted way to explain the measurement statistics.

### 2.1 Motivating the epistemic view of quantum states

Taking an anti-realist view of quantum states allows us to render some of the peculiar features of quantum theory as natural. Most clearly, the discontinuity associated with the instantaneous update of the quantum state following measurement can be explained as the update of what we know about the system in question. If, on the other hand, the quantum state represents an objective property of the system, then update remains mysterious: the underlying state discontinuously jumps in the same way as the quantum state.

It is natural to ask whether advantages such as this – where the peculiar features can be naturally explained – are compatible with a realist theory in the same vein as we usually think of Newtonian mechanics. Quantum states have many qualitative aspects in common with classical probability distributions, which are usually associated with states of knowledge about the underlying state of a system. This suggests, and indeed it is the case, that many of the features that are usually cited as characteristic of quantum theory can be qualitatively recovered in a realist theory if we regard the operational state as a state of knowledge. In this section we follow some of these arguments as we review some of the reasons that the epistemic view is appealing. Following this motivation, we
provide the first example of an ontological model that recovers all of the quantum measurement statistics for prepare–measure experiments, and in which the quantum state is not a part of the underlying reality. Our model is conceptually inspired by the material in this section.

Classical probability distributions over the state space of a particle represent our knowledge about the outcome of unperformed measurements\(^1\) of a particle’s position in that space. Immediately, it can be seen that two agents are able rationally to assign different but overlapping probability distributions over this state space. For example, one agent could hold a belief that a rolled die landed with an even number showing, while another could assign a probability distribution with support over only the prime possibilities. Both of these beliefs are compatible, because of the possibility that the outcome of a measurement of the die (i.e. when we look to see which face lies up) is ‘2’.

Suppose we have a die that always lands with an even number showing, and another that always lands with a prime number showing. They are otherwise indistinguishable, with all possibilities satisfying these restrictions equally likely to occur for their respective die. If we pick one of these dice at random, roll it and obtain the ‘2’ outcome, we are not able to state with certainty which of the dice we used: our two agents could both rationally stick to their original positions. A similar situation, as we saw when considering distinguishability, also arises with quantum states: non-orthogonal quantum states cannot be unambiguously distinguished in single-shot measurements. The key feature here is that the probability distributions overlap on the state space of the die. For 1/3 of the experiments where we roll either die, we will not be able to rule out either of the agents’ positions.

The theory about how probability distributions such as these evolve under classical dynamics is called *Louville mechanics*. Like quantum theory, Liouville mechanics contains a no-cloning theorem [DPP02]. The key idea is that the distance between probability distributions under the transformations that are allowed in Liouville mechanics is preserved. The authors show that the conservation of informational distance\(^2\) between different

---

\(^1\)We follow Bayesian probability theory, but do not rely on any of its properties that are not equally present in a frequentist approach.

\(^2\)The Kullback-Leibler distance between probability distributions is conserved under Liouvillian dynam-
starting states of the copying process (i.e. with different distributions to be cloned) is incompatible with any process where the marginal distributions associated with a source and target system behave as if the source were being cloned. That is, any process where the probability distributions over the cloning machine, source and target spaces evolve such that $\mu_C^{S|T} \rightarrow \mu_C^{S|S}$ is ruled out by the conservation of the informational distance. Thus a realist theory can explain naturally the existence of a no-cloning theorem, often cited as one of quantum theory’s characteristically peculiar features.

Another feature of quantum theory is that the amount of information required to encode a particular quantum state is much larger than the amount of information that can be extracted from it. For example, specifying exactly the state of a (pure) qubit requires two continuous parameters (the zenith and azimuthal angles of the Bloch vector), yet any projective measurement on the qubit yields at most one bit of information out. This bound on the amount of information that can be extracted by any measurement is due to Holevo [Hol73]. The situation is analogous to the state of a weighted coin. Whereas our knowledge of the state of the coin (which includes the extent to which it is weighted) requires giving a value in the range $[0, 1]$ according to the probability we assign to ‘heads’, for example, any particular observation of the underlying state of the coin yields one bit: heads or tails.

Hardy has termed this extra information that is required to specify the quantum state precisely, relative to the information that can be extracted through measurement, ‘excess baggage’. Further, along with Galvão, Hardy demonstrated a particular information processing task that requires a qubit to have access to a continuum of states that act as a memory, even though only a single bit needs to be extracted in order to solve the problem [GaH03]. In a similar vein, Montina has demonstrated that any realist theory that captures the Markovian dynamics that are ubiquitous in quantum mechanics must also encode the same exponential (in the dimension of the system) amount of information as the quantum state [Mon11]. A circumstantial explanation for the apparent state space expanding at the rate we see in quantum theory – i.e. exponentially in the dimension of the system – is that the effect is similar to the rate at which the space of classical probability
distributions expands as a function of system size. These two results however show that the rate of expansion of the state space in quantum theory must also be reflected in the underlying state space.

These properties and others are qualitatively recovered by Spekkens in a simple ‘toy’ theory [Spe07]. This seminal work has since been extended in several different directions, but in every case its explanatory power has been attributed to a ‘knowledge balance principle’: a restriction on the operational states of the theory based on what is knowable about the underlying reality. These include (amongst others) interference, multiplicity of convex decomposition of mixed states, impossibility of discriminating nonorthogonal states, no universal state inverter, no cloning and no broadcasting (a recent list can be found in [Spe14]).

The range and accuracy with which phenomena strongly associated with quantum theory are recovered by these non-quantum theories lent strong support to the $\psi$-epistemic position. More importantly, the simple restriction on what is knowable about the underlying reality appeared to explain the phenomena without appealing to some exotic physical axiom. While this work, and other similar investigations (for example [Har99, BRS11]) qualitatively recover quantum characteristics, it was not known whether the exact measurement statistics of experiments on quantum systems could be recovered in a realist theory where quantum states are associated only with agents’ knowledge about the underlying reality.

A $\psi$-epistemic ontological model – the type suggested by these arguments – can be distinguished from one that is $\psi$-ontic in the ontological models (OM) formalism. Our result in this chapter makes use of the OM formalism, which we review right now.

### 2.2 Ontological models

Ontological models provide a formalism in which we can clearly distinguish the operational *description* of an experimental process and an *explanation* in terms of underlying

---

3 A $\psi$-ontic ontological model is one that is not $\psi$-epistemic.
properties of systems that accounts for why particular events occur. In the OM formal-
ism, we remain agnostic about the particular qualities of the operational theory, and ac-
cept only that it provides a correct description of what is observed when we perform
an experiment – i.e. that the description contains elements corresponding to observable
events, operations or devices. We can then examine the relationship between elements
of the ontological model and elements of the operational theory in order to sharpen our
understanding of how the operational theory – in our case quantum theory – relates to
elements of reality.

In writing down an OM we make two assumptions:

– *Existence of an underlying reality*. We assume that there are objectively existent objects
  with real properties. This is in contrast to the view taking the operational descrip-
tion as fundamental.

– *Underlying states cause measurement outcomes*. We are able to interact with the objects
  in such a way that we can extract information, and there is a causal relationship
  between the real properties and the information that we extract.

These two assumptions sit naturally together, and it is worth noting that they are of-
ten implicitly made. This is because interesting arguments about the nature of quantum
states usually rely on additional assumptions. Indeed, the extra assumptions often form
the basis of the most famous theorems in foundations of quantum theory: for example
Bell’s theorem, which follows from investigating consequences of the assumption that
causal relationships exist only for locally interacting systems.

Several different types of ontological model can be distinguished [HS10], characterised
by the relationship between the ‘ontic states’ (or elements of reality) and the operational
states (in our case, quantum states). OMs can be categorised along a few different axes, of
which two are of particular interest to us. The first is related to the number of ontic states
that can be prepared according to each quantum state:

– \( \psi \)-complete: For each ontic state, there is exactly one operational state that could have
  prepared it. Further, for each operational state, there is exactly one ontic state that
could be prepared: the operational states and ontic states are in one-one correspondence, and the operational state provides a complete description of the underlying reality.

– \(\psi\)-supplemented: For each operational state, there is more than one (different) ontic state that could be prepared. In order for the ontic states that are possibly prepared by a single operational state to be different, they are supplemented by extra information. Bohmian mechanics is a good example of a \(\psi\)-supplemented theory, with the particle position being the supplementary information contained in the ontic state.

These two form a mutually exclusive and exhaustive categorisation of OM. The other categorisation – and the one that particularly interests us here – is related to the mapping from ontic states to quantum states:

– \(\psi\)-ontic: For each ontic state, there is exactly one operational state that could prepare it. Practically, this means that if one were able to find exactly the ontic state of a system (this may be ruled out by the model), then one could be sure precisely which operational state resulted in its preparation. The relation between ontic states and operational states is many-one (\(\psi\)-ontic, \(\psi\)-supplemented) or one-one (\(\psi\)-ontic, \(\psi\)-complete).

– \(\psi\)-epistemic: There exists at least one ontic state that is possibly prepared by more than one distinct operational state. That is, if one were able to find out exactly the ontic state of a system then in some cases one could not be sure which of a range of operational states were used in the preparation. If the ontological model recovers the quantum measurement statistics, then the model is necessarily \(\psi\)-supplemented, since for every pair of quantum states there exists a measurement where the outcome probabilities are different. The relation between ontic states and operational states is many-many.

This distinction is also mutually exclusive and exhaustive, and is represented schematically in figure 2.1. The relationship between the \(\psi\)-complete/\(\psi\)-supplemented and \(\psi\)-ontic/\(\psi\)-epistemic is shown in table 2.1.
ψ-ontic | ψ-epistemic
---|---
ψ-complete | one-one
ψ-supplemented | one-many

Table 2.1: Relationship between operational (i.e. quantum) states and ontic states for various categories of ontological model. Note that since any distinct pair of operational states has distinct measurement statistics, they cannot be associated with a single ontic state, forbidding many-one relationships in any ontological model that recovers the operational theory.

While the qualitative arguments for the ψ-epistemic view are compelling, especially in the range of features usually considered characteristically quantum, we lack a ψ-epistemic ontological model that recovers the quantum measurement statistics. Here we demonstrate by example that such a model is technically feasible.

**Formal statement of an ontological model**

An ontological model specifies

- A measurable space Λ containing ontic states λ, which can be thought of as the states of the underlying reality in the model. The λ have all of the qualities that we might expect of objectively real properties. For example, when a preparation device prepares a system, the system has a particular state λ independently of subsequent operations performed on it, and any operation that is subsequently performed on the system can in principle be described as manipulating this state (or revealing information about it).

- A mapping from each state of the operational theory – i.e. quantum state ψ – to a probability measure μψ with associated probability density function μψ(λ) over the ontic state space. In particular,

\[
\int_{\Lambda} \mu_\psi(\lambda) d\lambda = 1 \quad \forall \mu_\psi \\
0 \leq \mu_\psi(\lambda) \leq 1 \quad \forall \mu_\psi, \lambda.
\]  

The distribution is called the ‘epistemic state’ associated with the preparation ψ, i.e. the distribution μψ describes what is known about the ontic state following
the preparation \( \rho_\psi \). Note that the epistemic state is a standard, classical probability distribution over a measure space, encoding what is known about the underlying ontic states – unlike the quasi-probability distributions that are occasionally mentioned in other areas of quantum theory (e.g. Wigner representation of quantum states [GK05]). We say that an epistemic state is compatible with an ontic state \( \lambda \) iff \( \lambda \in \text{Supp} \mu_\psi \); that is, \( \lambda \) could have been prepared by a preparation device set to prepare \( \psi \).

- For each observable \( A \), a set of response functions, each corresponding to an outcome of the measurement for that observable. Each of these response functions \( \xi_{A,k}(\lambda) \in [0, 1] \) is such that \( \xi_{A,k}(\lambda) \) is the probability to observe outcome \( k \) of observable \( A \) given ontic state \( \lambda \). In particular, since every measurement should have an outcome for all ontic states

\[
\sum_k \xi_{A,k}(\lambda) = 1 \quad \forall \lambda. \tag{2.2}
\]

An ontological model is deterministic if \( \xi_{A,k}(\lambda) \in \{0, 1\} \), or equivalently \( \xi_{A,k}(\lambda) = (\xi_{A,k}(\lambda))^2 \). For a deterministic OM, we are able to predict with certainty the outcome of any measurement so long as we know the ontic state of the pre-measurement system.

If the response function for a particular outcome of a measurement (i.e. measurement operator) cannot be specified without the context of the complete measurement of which it is a part then the model is said to be contextual. Formally, in a contextual model, we have at least one pair of response functions corresponding to measurement operators \( A_k, B_j \) from two different measurements such that \( \xi_{A,k}(\lambda) \neq \xi_{B,j}(\lambda) \) even when the operators in the quantum formalism are equal: \( A_k = B_j \). Kochen and Specker [KS67] demonstrated that quantum theory (and consequently the OM due to [BB95] that we review presently) are the only non-contextual approaches to recovering the quantum measurement statistics in \( d > 2 \).

For \( d = 2 \), we review their explicit counter-example below.
Finally, the requirement that the ontological model recovers the quantum measurement statistics is expressed as

$$\Pr(k|A, \psi) = \int_\Lambda \xi_{A,k}(\lambda) \mu_\psi(\lambda) d\lambda = \text{Tr}(A_k \rho_\psi).$$

(2.3)

Two observations are worth noting immediately. Firstly, measurement necessarily disturbs the ontic state. Furthermore, the disturbance must leave the ontic state in the support of the epistemic state for the eigenstate of the corresponding outcome’s operator. This requirement is imposed since when a measurement is repeated, the same outcome is observed for the second measurement as was observed for the first. But the ontic state must nevertheless be disturbed, otherwise a sequence of distinct measurements could reveal more about the ontic state than is specified by a preparation. If the ontic state remained undisturbed during measurement then in any $\psi$-supplemented model we would be able to gain more knowledge about the ontic state than is granted by any preparation in quantum theory. That is, there would be more obtainable operational states in the OM than there are quantum states in quantum theory.

Secondly, orthogonal pairs of quantum states (i.e., pairs that can be distinguished with certainty using a single measurement) cannot be associated with epistemic states that overlap: $\text{Supp}(\mu_\psi) \cap \text{Supp}(\mu_\psi^\perp) = \emptyset$. Since by assumption measurement devices have access only to the ontic state — not necessarily to the quantum state describing the preparation — overlapping epistemic states cannot be distinguished with certainty from a single run of the experiment.

The argument runs as follows: we know that there is at least one measurement for which any pair of orthogonal quantum states can be unambiguously discriminated in a single shot. This measurement is realised by a set of response functions over $\Lambda$ for the system of interest. For each ontic state in $\text{Supp}(\mu_\psi)$, we know that $\xi_{\Pi_\psi}(\lambda) = 1$ since the outcome corresponding to the projector onto $\psi$ ($\Pi_\psi$) occurs with certainty. Because $\sum_k \xi_{A,k}(\lambda) = 1$ for all measurements, we know that for any other response function that forms part of the measurement (all of which have support exclusively on $\text{Supp}(\mu_\psi^\perp)$ in the quantum
formalism) the response function is $\xi_{A_k}(\lambda) = 0$, $A_k \neq \Pi_\psi$, $\lambda \in \text{Supp} (\mu_\psi)$. We are able to run the same argument on any state orthogonal to $\psi$, from which it follows that the supports of the corresponding epistemic states are disjoint.

Although we do not examine them in detail, it is worth noting for completeness that a transformation device would act as a map on the ontic state space, so that following a transformation acting as $T_{QT} : \rho_\psi \rightarrow \rho_\varphi$ we have in general terms $T_{OM} : \text{Supp} (\mu_\psi) \rightarrow \text{Supp} (\mu_\varphi)$. Significantly, disturbance due to measurement may be of this kind.

Pertinent to the question of whether the quantum state provides a complete description of the underlying state of affairs, there are two different classes of embedding of quantum theory in an OM. Following [HS10], we denote the two types $\psi$-ontic and $\psi$-epistemic. For a $\psi$-ontic OM, knowing precisely the ontic state of a system tells us precisely the setting of the preparation device – i.e. the quantum state that prepared it. For a $\psi$-epistemic OM, knowing precisely the ontic state does not necessarily tell us precisely the quantum state. This second class describes the kind of incompleteness that [HS10] argue Einstein meant.

Within the OM formalism, this difference is characterised by the existence of epistemic states for distinct quantum states that overlap:

$$\int \lambda \mu_\psi(\lambda)\mu_\varphi(\lambda)d\lambda > 0. \quad (2.4)$$

It is important to note that the inequality is strict.

Suppose that we know that some preparation prepared an ontic state $\lambda \in \text{Supp} (\mu_\psi) \cap \text{Supp} (\mu_\varphi)$ (or in general the intersection over arbitrarily many epistemic states). We are now in a position where we cannot be certain whether the ontic state was prepared according to $\rho_\psi$ or $\rho_\varphi$: the information about the setting of the preparation device is not contained in the ontic state. Such an ambiguous situation can only arise in a $\psi$-epistemic OM. Further, the quantum state is not a component of the underlying (real) states, and is rendered simply a means for us to label preparation procedures. This makes the fact that we are unable to distinguish $\psi$ and $\varphi$ with certainty in this case natural, because $\lambda$ is compatible with both preparations. A schematic of a $\psi$-epistemic OM is shown as figure 2.1.
Figure 2.1: Schematic of two ontic state spaces in different ontological models. The supports of epistemic states associated with four quantum states are shown. In (a) each ontic state is in the support of the epistemic state for at most one $\psi$; the model is $\psi$-ontic. In (b) those ontic states in the highlighted ‘epistemic region’ $E$ do not uniquely identify a quantum state, and could result from either of the associated preparation procedures: the model is $\psi$-epistemic. [Reproduced from [LJBR12] under licence.]

Prior to [LJBR12], the only known $\psi$-epistemic OM that recovers the quantum measurement statistics was due to Kochen and Specker [KS67]. We review the model after presenting our work, as it has some appealing properties lacking from our model. The Kochen-Specker model has never been extended, however, to systems of dimension $d > 2$ [Rud06].

An example. The OM formalism includes the standard Hilbert space description through the trivial assignment of $\Lambda$ as the complex projective space $CP^{d-1}$ (the boundary of the quantum state space), and $\mu_\phi(\lambda)$ being a delta-function distribution centred at $\lambda_\phi = |\psi\rangle\langle\psi|$ [BB95]. It is for this reason the term ‘ontic’ is used – as opposed to ‘hidden’, for example. Response functions in this model are non-deterministic: $\xi_{A,k}(\lambda) = \text{Tr}(\lambda A_k)$.

It is immediate that this model recovers the quantum measurement statistics, since the response function directly uses the Born rule, and the only contribution to the integral in equation 2.3 comes from the $\delta$-distribution. It is worth noting that by the Kochen-Specker theorem [KS67], this is the only OM that is non-contextual. For any other OM, the response function for a particular outcome generally has to depend on the co-measured
outcomes. Secondly, since the model includes the quantum state directly, it is not locally causal. That is, the measurement statistics for part of a bipartite system necessarily include a description of the full system, since the ontic state in this case is the quantum state of the preparation (which was bipartite). Further, when a measurement is effected on part of the system (for example containing measurement operators of the form $A^{(A)} \otimes 1^{(B)}$), the update is of the whole ontic state: including the ‘component’ on Bob’s side.

### 2.3 Determinism in ontological models

As a warm up to this chapter’s main result, and to introduce the OM on which our work is based, we examine a possible quality of measurement: whether or not measurement can be deterministic at the same time as giving the quantum measurement statistics. Equivalently, is it possible to account for the quantum measurement statistics in a realistic theory where for each state of reality the outcome of any measurement can be predicted with certainty? The answer that we review here is qualitatively similar to our work demonstrating a $\psi$-epistemic model: Bell provides an explicit OM that possesses the quality whose possibility we wish to decide [Bel66].

Bell’s model was originally formed as a counterexample to ‘von-Neumann’s silly assumption’ that no deterministic hidden variable model could reproduce the quantum measurement statistics. The model is capable of describing systems in any dimension $d$. That is, any quantum state in any dimension has a corresponding epistemic state in the model.

The ontic state space for a quantum system in $\mathcal{H}^d$ is $\Lambda_d = C^{d-1} \times [0,1] = \Lambda_q \times \Lambda_x$ so that ontic states are pairs of the form $\lambda = (|\psi\rangle, x)$. Note that here we use a Cartesian product. The model is $\psi$-supplemented, since full specification of the ontic state requires more than the quantum state: we also require a value for $x$.

When a state $|\psi\rangle$ is prepared, the ontic state that the preparation device prepares is sampled from a distribution sharply peaked at $|\psi\rangle$ in $\Lambda_q$ and uniform over $\Lambda_x$. Since the epistemic states for distinct pure quantum states do not overlap, the model is $\psi$-ontic. In particular, if we have an ontic state $\lambda = (|\psi\rangle, x)$, then it is unambiguous that it was
prepared by a preparation device set to $\psi$. For mixed states comprised of a convex combination of pure quantum states $\rho_a = \sum_i a_i |i\rangle \langle i|$, the corresponding epistemic state is a convex combination of the components’ epistemic states:

$$\rho_a = \sum_i a_i |i\rangle \langle i| \rightarrow \mu_{\rho_a} = \sum_i a_i \mu_i.$$  \hfill (2.5)

As a consequence of this, we cannot be sure given a particular ontic state that the preparation device was not configured with a mixed quantum state as the setting. Note however that this particular ambiguity is not sufficient to render the model $\psi$-epistemic: the overlap of epistemic states in the ontic state space must exist for pure states.

To give an explicit example, first we consider the qubit case, $d = 2$. Here the ontic state space is isomorphic to $\Lambda = S^2 \times [0, 1]$, where $S^2$ is the two-dimensional surface of the Bloch sphere. This isomorphism makes the model relatively easy to visualise; so here we will use the pure state Bloch vector rather than the complex quantum state vector. When a preparation device prepares a system according to the setting $\psi \in \mathcal{H}^2$, the ontic state of the prepared system is sampled from the distribution $\mu_\psi = (\hat{\lambda}, x) = \delta(\hat{\lambda} - \hat{\psi})$, where we adopt the convention that the Bloch vector corresponding to state $\psi$ is $\hat{\psi}$. As mentioned above, this distribution is uniform over the subset $\{(\hat{\psi}, x) : 0 \leq x \leq 1\}$. We refer to the subset of the ontic state space $\hat{\psi} \times [0, 1]$ as the interval ‘above’ the state $\psi$.

For a projective measurement on the qubit, there are two response functions each corresponding to an outcome of the measurement. The response functions for a particular measurement $A = (A_0, A_1)$ are deterministic:

$$\zeta_{A,k}(\hat{\lambda}, x) = \Theta \left[ (\langle \lambda_\psi | A_k | \lambda_\psi \rangle - x)(-1)^k \right],$$  \hfill (2.6)

where $\Theta(\cdot)$ is the Heaviside step function.\(^4\) Note – as Bell did – that there are many possible choices for the response functions that would work equally well. Eq. 2.6 gives a particular example of a set of response functions that satisfies this condition.

The only requirement is that the support of each response function over the subset $L = \{ (\hat{\psi}, x) : 0 \leq x \leq 1 \}$ has measure equal to the probability occurring in the Born rule,

---

\(^4\)Defined by $\Theta(x) := 1, x > 0; 0$ otherwise.
given the preparation $\psi$:

$$\int_{\text{Supp} \xi_{A,k}} d\lambda \delta(\lambda - |\psi\rangle) = \text{Tr}(A_k \rho_{\psi});$$  \hspace{1cm} (2.7)

how this support is distributed is entirely arbitrary.

One other aspect of this OM to note is that it is contextual in dimensions $d > 2$. That is, where the subspace $\{(|\psi\rangle, x) : x \in [0,1]\}$ is divided into three or more regions corresponding to the three or more outcomes of a PVM on a higher-dimensional system, it is not possible to fix the response functions of more than one outcome in the measurement independently of the others that complete the measurement.

To see this, consider a measurement $A = \{A_0, A_1, A_2\}$ on a qutrit state. For geometrical clarity, we replace the interval $[0,1]$ with a two dimensional (in general $d - 1$ dimensional) simplex. Call the simplex space $\Delta_d$, the convex hull of the $d$ corners of the simplex. The $x$ component of the ontic state is once again sampled from the uniform distribution, but this time over the simplex rather than the segment $[0,1]$.

The response functions for a projective measurement now divide the simplex into $d$ regions; set the regions so that each response function has support over a region given by the convex hull of $d$ vertices, $d - 1$ of which are in common with $\Delta_d$, one of which is within the simplex. i.e., in the triangle case, each response function has support on the convex hull of an edge of $\Delta_3$ and a further point in $\Delta_3$. This (single) point specifies the probabilities for all 3 outcomes of the measurement above each quantum state. Examples for the possible response functions for an outcome $A_k$ are shown in figure 2.2. Note that the single point where all three response functions meet above each quantum state is fixed only once we know the probabilities for all three outcomes, given that preparation.

Now, consider the task of specifying the response function for a single outcome in any dimension above the state $\psi$. In the two-dimensional case, as we have seen, the response function has support on the interval from one end of $\Delta_2 = [0,1]$ to a point $|\langle A_k | \psi \rangle|^2$. For a three dimensional system, the response function has support along an edge of $\Delta_3$, and this time we require that the area of the set, once normalised by the total area of $\Delta_3$, is $|\langle A_k | \psi \rangle|^2$. This is satisfied by any point on a line parallel to the $k$th outcome’s corre-
\( \{ \lambda : |\lambda \rangle = |\psi \rangle \} : \)

Figure 2.2: Subset of ontic states above the point corresponding to a particular quantum state \( |\psi \rangle \). For the left figure, each of the four shaded areas corresponds to a possibility for the support of an idempotent response function \( \xi_{A,0}(\lambda) \). Note that the areas covered are all the same (\( = \text{Tr}(A_0 \rho_\psi) \)): so the probability for this outcome is the same for all of the possible response functions. The response function for this outcome in the absence of response functions for other outcomes completing the measurement is specified only up to a point on the dashed line. On the right we show a complete set of response functions for all three outcomes for a particular measurement (each represented by a different colour).

sponding edge. A similar argument follows for the other response functions for PVMs in any dimension, with the response function ambiguous up to specification of a point in \( \Delta_{d-1} \) parallel to the (generalised) edge for the response function.

Our method of visualising the ‘supplementary’ region of the Bell OM’s ontic state space restores symmetry to the outcomes of any measurement, and makes explicit the extent to which each outcome’s response function is determined just by the appropriate probability. In what follows, however, the extension is not necessary, and so for the rest of this chapter we recall the Bell model as having the original – \([0, 1]\) – supplementary region.

Next, we present a modification of the Bell model that renders it \( \psi \)-epistemic.

2.4 Quantum states as states of knowledge

In this section we present the main result of the chapter: that Bell’s model (as described above) can be modified in such a way as to render a \( \psi \)-epistemic model in any finite dimension. We begin by describing the model in the qubit case for pure preparations and projective measurements. In this case, the ontic state space is relatively easy to visualise. Then we provide a formal description of the model in higher dimensions.
Once again, since the quantum state space is isomorphic to the surface of a sphere, we are able to visualise a slice of the ontic state space $\Lambda$ as an annulus with $\hat{\lambda}$ specifying the direction, and $x$ the radial distance is depicted in Fig. 2.3(a). For the argument that we make below, we can now restrict our focus to a plane of the ontic state space that contains a single slice through the Bloch sphere. The first component is the Bloch vector corresponding to the quantum state, while the supplementary subspace extends the surface to an annulus.

Let $\hat{z}$ correspond to the north pole of the Bloch sphere, and let $\hat{\lambda} \cdot \hat{z} = \cos(\theta_{\lambda})$, where $\theta_{\lambda}$ is the polar angle of the Bloch vector $\hat{\lambda}$. We label the upper ($\theta_{\lambda} < \pi/2$) hemisphere $R_0$ and the lower ($\theta_{\lambda} > \pi/2$) hemisphere $R_1$. Given a projective measurement $\Phi = \{|\phi_0\rangle\langle \phi_0|, |\phi_1\rangle\langle \phi_1|\}$, assume that the outcomes are labelled such that $|\langle \phi_0|z\rangle|^2 \geq |\langle \phi_1|z\rangle|^2$. That is, for every measurement the first outcome projects onto a state with a Bloch vector in the northern hemisphere, with the other outcome projecting onto the diametrically opposite state.

Our first step given this prescription for labelling the outcomes of projective measurements is to examine the possible response functions that are associated with each measurement.

Using the response functions that we defined in Eq. (2.6), the crucial observation is that there is a set of ontic states that are always within the support of the outcome that we labelled 0; i.e. the outcome projecting onto a state in $R_0$. Explicitly, the ontic states in the set

$$E_0 = \{ (\hat{\lambda}, \lambda_x) : \hat{\lambda} \in R_0 \text{ and } 0 \leq \lambda_x < (1 - \sin \theta_{\lambda})/2 \} \quad (2.8)$$

all result in the outcome projecting onto $\hat{\phi}_0$ for any measurement $\Phi = \{\hat{\phi}_0, \hat{\phi}_1\}$.

Similarly, those in the set

$$E_1 = \{ (\hat{\lambda}, \lambda_x) : \hat{\lambda} \in R_1 \text{ and } (1 + \sin \theta_{\lambda})/2 < x \leq 1 \} \quad (2.9)$$

all result in the $\hat{\phi}_1$ outcome. These sets are illustrated in Fig. 2.3(b).

---

5Since the equator of the Bloch sphere is a set of measure 0, it does not make a difference to our subsequent calculations how it is accounted for.
In order to construct a $\hat{\psi}$-epistemic model, note that if an epistemic state $\mu_{\hat{\psi}}$ assigns non-zero probability to the subset $E_0$, then this much probability weight can be redistributed over $E_0$ without changing the Born rule statistics. This is because ontic states in $E_0$ behave identically to one another, as far as predictions for quantum measurement outcomes go:
no ontic state in the region $E_0$ can be distinguished from any other ontic state in this region by any measurement described by quantum theory. The same holds true for $E_1$.

This observation allows us to define a modified version of Bell’s model where the probability weight assigned by the original Bell epistemic states is ‘spread out’ over the whole of the epistemic regions $E_0$ and $E_1$, according to which the original epistemic state had support on. With this in mind, define a modified Bell model such that for $\hat{\psi} \in \mathcal{R}_0$, 

$$
\mu_{\psi}(\hat{\lambda}, x) = \delta(\hat{\lambda} - \hat{\psi}) \Theta \left( x - \frac{1}{2} (1 - \sin \theta_{\psi}) \right) + \frac{1}{2} (1 - \sin \theta_{\psi}) \mu_{E_0}(\hat{\lambda}, x)
$$

(2.10)

where $\theta_{\psi}$ is the polar angle of $\hat{\psi}$, and $\mu_{E_0}$ is essentially arbitrary but can be taken to be the uniform distribution over $E_0$. A similar expression defines $\mu_{\psi}$ for states with $\hat{\psi} \in \mathcal{R}_1$. Note that one choice for the distribution over $E_0$ is simply a delta-distribution at a single point. We use this particular choice in converting our OM to a classical protocol for simulating quantum teleportation below.

The modified Bell model still reproduces the Born rule, but now the model is $\psi$-epistemic. Any two quantum states in the same hemisphere are described by distributions that overlap, either in $E_0$ or in $E_1$. The preparation of an ontic state from either of these regions does not reveal a unique quantum state, but only reveals in which hemisphere the quantum state resides.

**Generalisation to higher dimensions**

Here we modify the Bell model to produce a $\psi$-epistemic model for arbitrary finite dimension. Our broad approach is to exploit the freedom that we have in re-arranging the response functions described above in such a way that we create a set of special ontic states analogous to $E_0$ above. These special ontic states have the property that regardless of the measurement that is performed, a particular outcome with respect to a well defined ordering of outcomes always obtains. We also use that the response functions are deterministic. That is, we find for example a set of ontic states for which $\Pr(k|A, \lambda) = \delta_{k,0} \forall A$.

The ontic state space for the $d$-dimensional modified Bell model remains $\Lambda = \mathbb{C}^{d-1} \times \ldots$
For the moment, we keep the epistemic state distribution corresponding to a quantum state $|\psi\rangle$ as $\mu_{\psi}(|\lambda\rangle, x) = \delta(|\lambda\rangle - |\psi\rangle)$.

Response functions for a projective measurement $A$ are defined such that $\xi_{A,k}$ has support of length $\langle \lambda | A_k | \lambda \rangle$ on the line segment $\{(|\lambda\rangle, x): 0 \leq x \leq 1\}$. This model is $\psi$-ontic since the delta functions do not overlap for distinct $|\psi\rangle$.

In order to construct a $\psi$-epistemic model, fix an arbitrary preferred state $|0\rangle$. For each measurement $A$, assume that the outcomes are ordered such that $\langle 0 | A_0 | 0 \rangle \geq \langle 0 | A_1 | 0 \rangle \geq \cdots \geq \langle 0 | A_{d-1} | 0 \rangle$. Fix the response functions so that

$$\xi_{A,k}(|\lambda\rangle, x) = 1 \quad \text{if} \quad \sum_{i=0}^{k-1} \langle \lambda | A_i | \lambda \rangle \leq x < \sum_{i=0}^{k} \langle \lambda | A_i | \lambda \rangle$$

(2.11)

and

$$\xi_{A,k}(|\lambda\rangle, x) = 0 \quad \text{otherwise.}$$

(2.12)

In Equation (2.11),

$$\sum_{i=0}^{k-1} \langle \lambda | A_i | \lambda \rangle$$

(2.13)

is taken to be 0 when $k = 0$. For completeness, let us specify that for $x = 1$, $\xi_{A,k}(|\lambda\rangle, x) = 1$ iff $k = d - 1$.

We are now in a position to look for the subset $\mathcal{E}_0$ of $\Lambda$ such that ontic states in $\mathcal{E}_0$ will always cause the same measurement outcome for all measurements. To this end, note that $\langle 0 | A_0 | 0 \rangle \geq 1/d$; our ordering allows us to bound from below the probability to obtain the 0th outcome for any measurement performed on the state $|0\rangle$.

Given this, we can use a triangle inequality from the Fubini-Study metric on the quantum state space to bound the probability for the 0th outcome for all states in a region around $|0\rangle$. In particular, we use the inequality defined in terms of the overlaps of the three vectors corresponding to the prepared state, measurement outcome projector and the special state $|0\rangle$. That is, if $|\langle \lambda | 0 \rangle|^2 > \frac{d-1}{d}$, then $\langle \lambda | A_0 | \lambda \rangle > 0$. Specifically, we use that

$$D(|\lambda\rangle, |0\rangle) \geq D(|A_0\rangle, |\lambda\rangle) - D(|A_0\rangle, |0\rangle)$$

(2.14)
where \( \cos D(|\psi\rangle, |\phi\rangle) = |\langle \psi | \phi \rangle| \). Substituting the bounds from above we get that
\[
|\langle \lambda | 0 \rangle| \geq \cos \left( \frac{\pi}{2} - \arccos \sqrt{\frac{1}{d}} \right) \geq \sqrt{1 - 1/d}.
\] (2.15)

Generalising this approach to arbitrary \(|\chi\rangle\), let
\[
z(|\chi\rangle) = \inf_{|\phi\rangle: |\langle 0 | \phi \rangle|^2 \geq 1/d} |\langle \phi | \chi \rangle|^2.
\] (2.16)

This is the minimum probability over all possible \(A_0\) outcomes to obtain a \(k = 0\) outcome, given preparation \(|\chi\rangle\). Explicitly,
\[
z(|\chi\rangle) = \cos^2 \left[ \arccos \sqrt{\frac{1}{d}} + \arccos |\langle 0 | \chi \rangle| \right].
\] (2.17)

Now we are in a position to write down the ‘epistemic region’ around the special state \(|0\rangle\):
\[
E_0 = \left\{ (|\lambda\rangle, x) : |\langle \lambda | 0 \rangle|^2 > \frac{d - 1}{d} \text{ and } 0 \leq x < z(|\lambda\rangle) \right\}.
\] (2.18)

Any ontic state \(\lambda \in E_0\) has the property that whatever measurement is performed, the outcome is \(A_0\). The epistemic states can therefore be modified as above to produce a \(\psi\)-epistemic model.

Informally, the idea is the same: any probability that \(\mu_\psi\) assigns to ontic states within the set \(E_0\) can be redistributed over the whole of \(E_0\) without changing Born rule statistics. More specifically, when \(|\langle \psi | 0 \rangle|^2 \leq (d - 1)/d\), let
\[
\mu_\psi(|\lambda\rangle, x) = \delta(|\lambda\rangle - |\psi\rangle),
\] (2.19)

and when \(|\langle \psi | 0 \rangle|^2 > (d - 1)/d\), let
\[
\mu_\psi(|\lambda\rangle, x) = \delta(|\lambda\rangle - |\psi\rangle) \Theta(x - z(|\psi\rangle)) + z(|\psi\rangle) \mu_{E_0}(|\lambda\rangle, x),
\] (2.20)

where, as above, \(\mu_{E_0}\) is arbitrary but could be taken to be the uniform distribution over \(E_0\). Further, we are able to repeat this procedure \(d\) times over the entire ontic state space,
creating $d$ epistemic ‘islands’ over which the epistemic states for preparations within those islands can be spread. The ordering for the measurement outcomes changes discontinuously at the boundary of these islands, but since the labelling has no operational significance this does not cause a problem.

The model that we have presented here recovers the Born rule statistics for any dimension, admits a realist theory for the underlying states yet, by virtue of the overlapping epistemic states, also has the appealing properties of $\psi$-epistemic models in general. Although the model is contrived and inelegant, we have demonstrated that this kind of model is mathematically plausible within the requirements for a well-formed ontological model.

**No-go theorems for $\psi$-epistemic models**

The ontological model presented here is particularly pertinent in light of several recently published no-go theorems demonstrating that such $\psi$-epistemic models in general cannot recover the quantum measurement statistics. The results include those by Pusey et al., Colbeck and Renner and also Hardy (respectively [PBR12, CR11a, Har12]). In each of these cases, auxiliary assumptions are added to those that we made in order to demonstrate that the quantum measurement statistics cannot be recovered by a $\psi$-epistemic model. Here we review how each of the additional assumptions used by these authors is violated by the model that we presented above.

Pusey et al. [PBR12] assume a *preparation independence principle*. Their proposed experiment involves the independent preparation of $n$ systems (labelled $0, \ldots, n-1$), then a joint measurement performed on the combined system. In quantum theory, the outcome probabilities for such an experiment are calculated using the tensor product of the prepared systems’ states:

$$
\Pr(k|A, \{\psi_0, \ldots, \psi_{n-1}\}) = \langle \psi_0| \otimes \ldots \otimes |\psi_{n-1}\rangle A_k(|\psi_0\rangle \otimes \ldots \otimes |\psi_{n-1}\rangle). \quad (2.21)
$$

Here the measurement operator $A_k$ acts on the product Hilbert space $\mathcal{H}_0 \otimes \ldots \otimes \mathcal{H}_{n-1}$.
and the $|\psi_i\rangle$ describe the operational state for each independent preparation device.\footnote{This independence is meant in the same sense as independently chosen measurement settings in a Bell test experiment.}

Pusey et al.’s assumption is characterised as the preparation stage having a particular structure in the ontological models formalism. Specifically, if two preparation devices are independent, then the ontic states of the systems that they prepare are also assumed to be independent. This is represented by a Cartesian product of ontic states in the model. Further, no extra information than that local to the ontic states of the systems to be combined can have an effect on the output given by the measuring device. Practically, this rules out global degrees of freedom from forming part of the description for preparations carried out in independent regions of space-time; similar, as pointed out by [Lei14], to Einstein separability [How85].

In [PBR12], the ontic state of the combined system is then of the form

$$\lambda_{\text{comb}} = \lambda_0 \times \ldots \times \lambda_{n-1}. \quad (2.22)$$

For our model the entire preparation, including the case where several preparation devices are used independently, is considered as a single operation. It is not generally the case that the ontic state of the combined system is of the form given in equation 2.22. This is why our model is able to recover the measurement statistics in spite of Pusey et al.’s argument. Nevertheless, we are able to provide an explicit example for a pair of states that Pusey et al.’s argument shows cannot both be compatible with a single ontic state, yet in our model have finite overlap. We use a modified version of their simplest example. Consider a pair of states

$$|a\rangle = \cos \frac{\pi}{8} |0\rangle - \sin \frac{\pi}{8} |1\rangle$$

$$|b\rangle = \cos \frac{\pi}{8} |0\rangle + \sin \frac{\pi}{8} |1\rangle. \quad (2.23)$$

One can easily check that these states are non-orthogonal. Further, in our model the corresponding epistemic states overlap in $E_0$, the epistemic region around $|0\rangle$. In order to construct the contradiction, we will also require the respective orthogonal states, $|a^\perp\rangle$ and $|b^\perp\rangle$. 

---

53
Consider then the four permutations of preparing pairs of systems in these states: $|a\rangle \otimes |a\rangle, |b\rangle \otimes |a\rangle, |a\rangle \otimes |b\rangle, |b\rangle \otimes |b\rangle$. Note that in each case the ontic state for both systems in each pair is possibly $\lambda \in \mathcal{E}_0$. In particular, using Pusey et al.’s assumption for the combined ontic state, $\lambda_{\text{comb/d}} = \lambda \times \lambda$ is compatible with all four pairs. Although our model provides no prescription for combining systems, the content of the preparation independence principle is that there is no more useful information in the combined ontic state than in the Cartesian product of the components.

Following Pusey et al.’s argument, we now consider a joint PVM on both systems projecting into the basis

$$
|\phi_0\rangle = \frac{1}{\sqrt{2}} \left( |a\rangle \otimes |a^\perp\rangle + |a^\perp\rangle \otimes |a\rangle \right)
$$

$$
|\phi_1\rangle = \frac{1}{\sqrt{2}} \left( |a\rangle \otimes |b^\perp\rangle + |a^\perp\rangle \otimes |b\rangle \right)
$$

$$
|\phi_2\rangle = \frac{1}{\sqrt{2}} \left( |b\rangle \otimes |a^\perp\rangle + |b^\perp\rangle \otimes |a\rangle \right)
$$

$$
|\phi_3\rangle = \frac{1}{\sqrt{2}} \left( |b\rangle \otimes |b^\perp\rangle + |b^\perp\rangle \otimes |b\rangle \right)
$$

For each of the four preparations, there is one outcome from this measurement that is orthogonal to the combined pair of systems. Consequently, for each corresponding response function $\zeta_{\phi,k}$ there is a preparation for which every possible ontic state lies outside the support of $\zeta_{\phi,k}$.\(^\text{7}\) This implies that no ontic state that can give the correct measurement statistics is compatible with all four of the product preparations, contradicting the model we presented above.

The crucial step is the means of combining ontic states for independently prepared systems without global degrees of freedom: Pusey et al.’s preparation independence principle. In our model, such a set up is treated as a whole. Specifically, the entire preparation stage is regarded as a single preparation of the state $|\psi_0\rangle \otimes \ldots \otimes |\psi_{n-1}\rangle$. In particular, in our model the states $|a\rangle \otimes |a\rangle, |b\rangle \otimes |a\rangle, |a\rangle \otimes |b\rangle, |b\rangle \otimes |b\rangle \in \mathcal{H}^4$ do not correspond to overlapping epistemic states.

Pusey et al.’s no-go theorem had significant impact on the landscape of ontological models that can explain the quantum measurement statistics. Particularly pertinent is that the

\(^\text{7}\)Note that the response functions for these joint measurement outcomes would be defined on the combined ontic state space $\Lambda = \Lambda^{(a)} \times \Lambda^{(b)}$. 

---

54
construction relies only on prepare–measure experiments such as those we considered above, rather than possibly more sophisticated models that also provide an account of dynamics.

The two further no-go theorems that we mention here, due to Hardy and Colbeck/Renner, both involve additional assumptions about how transformations are treated in the ontological model (as pointed out by Leifer [Lei14]). That is, when we implement some process involving the system such that its quantum state changes, how does the ontic state change? The most significant possibility to note is that the ontic state need not necessarily change deterministically, even when the quantum description of the transformation is unitary.

Where transformations are involved, we are able to specify certain restrictions immediately: for example, a unitary transformation that acts as \( U|\psi\rangle = |\psi'\rangle \) is certainly represented by a transformation in the ontic state space that has codomain \( \text{Supp} \xi_A \), with \( A = |\psi'\rangle \langle \psi'| \) (so that measurements involving the projector \( A \) give that outcome with certainty).

In the case of Hardy [Har12], the extra assumption is ontic indifference. A model obeys ontic indifference if any transformation that leaves any particular quantum state \( |\psi\rangle \) unchanged also has no effect on any ontic state \( \lambda \in \text{Supp} (\mu_\psi) \). Clearly if there is only one such ontic state (i.e. the model is \( \psi \)-complete) then this statement is trivial, but for any \( \psi \)-epistemic model there are necessarily cases where more than one ontic state is compatible with a single (pure) quantum state.

As a practical example, Hardy uses a Mach-Zehnder interferometer with a locality assumption about a particular transformation – a phase shift – that can be inserted into one of the arms. Specifically, the Hardy assumption in this case posits that when a phase shift is introduced into one arm of the interferometer, the transformation effects no change in the ontic state of any preparation with support exclusively in the (quantum) subspace of the other arm.

The argument proceeds to demonstrate that there can be no overlap between the epistemic states \( \mu_\psi \) and \( \mu_\phi \) where \( |\psi\rangle = |x_0\rangle \) and \( |\phi\rangle = \frac{1}{\sqrt{2}} (|x_0\rangle + |x_1\rangle) \). The effect of insert-
Figure 2.4: Mach-Zehnder interferometer as used in [Har12] to demonstrate ontic indifference. The phase shifting device $P$ can be inserted into the $x_1$ arm of the interferometer, implementing a transformation which Hardy argues we should regard as ‘ontically indifferent’ for the state $|x_0\rangle$.

Inserting a $\pi$ phase shift into the $x_1$ arm (implemented by device $P$) is different for these two states. Specifically

$$P(|x_0\rangle) = |x_0\rangle$$

$$P\left(\frac{1}{\sqrt{2}}(|x_0\rangle + |x_1\rangle)\right) = \frac{1}{\sqrt{2}}(|x_0\rangle + |x_1\rangle) = |\phi^\perp\rangle.$$  \hspace{1cm} (2.25)

Note that since $|\phi\rangle$ is orthogonal to $|\phi^\perp\rangle$, the epistemic states $\mu_\phi$ and $\mu_{\phi^\perp}$ necessarily have no mutual overlap. Now we are in a position to run the argument. Assume that $\text{Supp} (\mu_\phi) \cap \text{Supp} (\mu_{\phi}) = \mathcal{E} \neq \emptyset$. Further, since $\text{Supp} (\mu_\phi) \cap \text{Supp} (\mu_{\phi^\perp}) = \emptyset$, we must have that $\mathcal{E} \nsubseteq \text{Supp} (\mu_{\phi^\perp})$. By ontic indifference, the use of $P$ has no effect on any ontic state $\lambda \in \text{Supp} (\mu_\phi)$, in particular including $\lambda \in \mathcal{E}$. Since $P(|\phi\rangle) = |\phi^\perp\rangle$, $P$ certainly acts on any ontic state $\lambda \in \text{Supp} (\mu_\phi)$ in such a way that $P(\lambda) \in \text{Supp} (\mu_{\phi^\perp})$. In particular, this includes $P$ having non-trivial action on any ontic state in $\mathcal{E}$, contradicting ontic indifference. The conclusion is that ontic indifference is incompatible with the states $\psi$ and $\phi$ having any mutually compatible ontic states.

Hardy’s proceeds to extend his argument to any pair of quantum states for a qubit (by slightly complicating the interferometer example), then in full generality with any finite-dimensional system. Further, the assumption can be weakened to restricted ontic indifference, where Hardy only requires one quantum state to obey the ontic indifference assumption.
The full argument appears in [Har12], as well as in the review by [Lei14] and in a condensed form in [PPM13]. Our model provides no account for transformations of the kind that Hardy proposes, but it is nevertheless worth noting that the argument relies on a similar type of non-local causation to that involved in any OM that recovers the correct measurement statistics in a Bell-type experiment. In particular, if an ontological model has a component representing the vacuum state in the $x_1$ arm of the interferometer, then there is no a priori reason why there should not be a global degree of freedom that is not indifferent to the use of device $P$.

Finally, we consider the argument of Colbeck and Renner. Their initial work [CR11b, CR11a] goes further than ruling out $\psi$-epistemic OMs, demonstrating in fact that any OM must be $\psi$-complete; i.e. the quantum state provides a complete description of any underlying reality. More recently, they relaxed one of the assumptions in their argument in order to focus specifically on the $\psi$-epistemic question. Their new argument is synthesised by Leifer [Lei14] following an exposition presented by Renner [Ren].

The new argument utilises an experiment in which a single system is subject to a pair of measurements with ‘freely chosen’ settings, in a similar fashion to common demonstrations of Bell’s theorem. The free choice of measurement settings for the pair of measurements can more formally be expressed, as Renner does [Ren], as a statement about the independence of probability distributions over the random variables describing those settings. Indeed, Renner points out that the distinction usually made between measurement settings and ontic states is merely operationally or intuitively motivated.

However, as pointed out by Leifer [Lei14], the parameter independence assumption that is a key component of Colbeck and Renner’s free will assumption is typically broken by OMs in order to explain the strength of correlations in Bell-type experiments. Specifically, Alice’s choice of measurement can be interpreted as a part of the preparation of the system at Bob’s side (or vice versa). This is certainly one way to account for the outcome

---

8One can think of their stronger theorem – that any OM must be $\psi$-complete – as proving that one knows the exact ontic state if one knows the quantum state, and one knows the quantum state if one knows the ontic state. By relaxing the assumption that they call ‘QMb’, the implication is only in the second direction; but nevertheless enough to rule out $\psi$-epistemic OMs.

9An in-depth discussion of the relationship between parameter independence, outcome independence and Bell locality can be found in [Shi93].
statistics of Bell-type measurements in the model that we present above.

Any no-go theorem for ψ-epistemic models has an impact beyond the ψ-ontic/ψ-epistemic debate. As such, much of the discussion following this question (and in particular following the publication of Pusey et al. [PBR12]) focused on whether the auxiliary assumptions could be dropped or weakened. The key contribution of our work here is to demonstrate that these auxiliary assumptions are strictly necessary in order to prove any no-go theorems on ψ-epistemic OMs.

2.5 Other ψ-epistemic OMs

In this section we review two other ψ-epistemic OMs. The first, due to Kochen and Specker, has the property that any non-orthogonal pair of states has non-zero overlap. In fact, the distinguishability of the epistemic states given any choice of response functions is exactly the distinguishability of the corresponding quantum states using the optimal quantum measurement. This implies that the model is in a sense the ‘most’ ψ-epistemic model possible: all of the indistinguishability of quantum states is explained by the overlap of the corresponding epistemic states.

The second model that we describe is based on our modified Bell model. By shrinking the size of the epistemic islands and ‘mixing’ different OMs, [ABCL13] (following a thread on MathOverflow [AL06]) show that our model can be modified along similar lines to our approach but this time so that any non-orthogonal pair of quantum states have overlapping epistemic states.

*The Kochen-Specker qubit model* [KS67] successfully explains the statistics of PVM measurements performed on pure preparations of a qubit. Although the model provides the only other known example of a ψ-epistemic OM that is not in the spirit of our model, the primary motivation for Kochen and Specker was to demonstrate that their contextuality theorem (that no non-contextual OM could recover the quantum measurement statistics in dimension $d > 2$) did not apply to the qubit case; not to provide a ψ-epistemic OM.

The model has two particularly appealing properties. Firstly, the epistemic states asso-
Figure 2.5: (a) Bloch sphere showing a representation of the Kochen-Specker epistemic state for $|\psi\rangle$, a distribution over ontic states centred at the Bloch vector $\hat{\psi}$. The height of the distribution above the surface of the sphere corresponds to the probability density at each point. (b) quantum state $\rho_\psi$ in red, with 5000 ontic states sampled from the Kochen-Specker distribution $\mu_\psi$.

Associated with different quantum states are exactly as distinguishable as the corresponding quantum states. Secondly, there is no ‘preferred’ direction – the model retains all of the unitary symmetries of quantum states.

For the Kochen-Specker model, the ontic state space is isomorphic to the surface of the Bloch sphere. So we have that $\Lambda$ is isomorphic to $S^2$, and an ontic state $\hat{\lambda}$ corresponds to a point on the surface. However, each pure quantum state $\psi$ is associated with a distribution over the ontic state space given by

$$\mu_\psi(\lambda) = \frac{1}{\pi} \hat{\lambda} \cdot \hat{\psi} \times \Theta(\hat{\lambda} \cdot \hat{\psi}).$$

(2.26)

A picture of the epistemic state associated with a particular $\psi$, along with a collection of samples from that distribution is shown in figure 2.5.

For PVMs, the model is outcome deterministic. The indicator function $\xi_\phi$ associated with the PVM outcome projecting onto the state $|\phi\rangle \langle \phi|$ is idempotent, and has support only in the hemisphere of the ontic space centred at the same coordinates as the Bloch vector for $|\phi\rangle \langle \phi|$:

$$\xi_\phi(\gamma') = \begin{cases} 1 & 0 \leq \gamma' \leq \pi/2 \\ 0 & \text{otherwise} \end{cases}$$

(2.27)
Figure 2.6: Indicator function for the PVM effect projecting onto the state with Bloch vector shown in blue. The function takes the value 1 in the region under the blue shaded ‘cap’, and 0 otherwise. A pair of PVM effects corresponding to a complete PVM is represented by Bloch vectors $\vec{w}$ and $-\vec{w}$, so that the union of indicator functions’ support is a mutually exclusive and exhaustive cover of the entire ontic state space.

where $\gamma’$ is the angle from the point corresponding to the Bloch vector for the state $|\phi\rangle\langle\phi|$. A picture of such an indicator function is shown in figure 2.6.

An important and appealing quality of the Kochen-Specker model is the distinguishability of the epistemic states. In the introduction we reviewed the problem of distinguishing non-orthogonal quantum states; a similar problem can be examined in OMs. Suppose that we have two preparation devices with two different settings. When each device prepares a system, the system has an ontic state sampled from the epistemic state corresponding to the setting. If the epistemic states overlap, then there are ontic states in the intersection of the support that can arise from either preparation. For these ontic states, no set of response functions can distinguish which preparation was used. The consequence is that the only ontic states than do allow us to distinguish the two preparations unambiguously are the ontic states that are in the support of only one epistemic state.

This means that the most effective set of response functions for distinguishing the two preparations work by allowing us to distinguish between three regions of the ontic state space: $\text{Supp} (\mu_\psi)/\text{Supp} (\mu_\phi)$, $\text{Supp} (\mu_\psi) \cap \text{Supp} (\mu_\phi)$ and $\text{Supp} (\mu_\phi)/\text{Supp} (\mu_\psi)$. The success probability for a strategy that uses response functions such as these, and where the
epistemic states have the same profile up to a translation in the ontic state space is given by the variational distance between the epistemic states for the respective preparations:

\[ D(\mu_\varphi, \mu_\psi) = \frac{1}{2} \int_\Lambda |\mu_\varphi(\lambda) - \mu_\psi(\lambda)|d\lambda. \]  

(2.28)

Note that the existence of a pair of epistemic states \( \mu_\varphi \) and \( \mu_\psi \) in an OM such that \( D(\mu_\varphi, \mu_\psi) < 1 \) implies that the model is \( \psi \)-epistemic. Further, the epistemic states in an OM that recovers the quantum measurement statistics must be at least as distinguishable as the corresponding quantum states; this provides a lower bound on the variational distance between the epistemic states.

We now turn our attention to the second \( \psi \)-epistemic model.

Since our model was published, Aaronson et al. ([ABCL13]) examine the question of whether the model can be modified in such a way that any pair of non-orthogonal states can be associated with epistemic states having non-trivial overlap. Their answer demonstrates by explicit construction that such a model is possible. By reducing the sizes of the epistemic regions that we introduced, they are able to bound the probability for particular outcomes according to an ordering that is specific to each pair of states. By doing this, they can create an OM such that a particular pair of non-orthogonal states (and a small region around each one) now have overlapping epistemic states. Next, by ‘mixing’ the appropriate OMs for several pairs of states, they create a model for which any pair of non-orthogonal quantum states has non-trivial overlap (i.e. \( D(\mu_\varphi, \mu_\psi) < 1 \) \( \forall \psi, \phi \) where \( |\langle \phi | \psi \rangle| > 0 \)).

The mixing step involves performing a classical mixture of the epistemic states over any pair of models. This necessitates, as Aaronson et al. show, adding an extra parameter to the ontic state space that acts as a label for the original models. Take as two models \( M_1 = (\Lambda_1, \mu_1, \xi_1) \) and \( M_2 = (\Lambda_2, \mu_2, \xi_2) \); in each one a different pair of quantum states correspond to overlapping epistemic states. Then define a new model \( M_c \) with the following properties: \( \Lambda_c = (\Lambda_1 \times \{1\}) \cup (\Lambda_2 \times \{2\}) \) and \( \mu_c = a\mu_1 + (1 - a)\mu_2 \). Aaronson et al. demonstrate that a countable number of such models is sufficient in order to guarantee that every pair of non-orthogonal states has non-trivial overlap in the OM.
Like our model, on the other hand, Aaronson et al.’s model does not allow us to explain all of the uncertainty associated with distinguishing non-orthogonal quantum states in terms of the overlap of the corresponding epistemic states. Indeed, the only OM having this property is the Kochen-Specker model. Subsequently, Maroney [Mar12] and Leifer and Maroney [LM12] demonstrated that no such model exists in dimension $d > 2$.

### 2.6 $\psi$-epistemic OMs and simulating quantum teleportation

One area where $\psi$-epistemic OMs for quantum theory find practical significance is in quantifying the communication complexity of tasks such as quantum teleportation. While this is not the primary purpose of the model that we describe above, it is interesting to note that it can be translated into a classical simulation of quantum teleportation where the two parties occasionally require only one cbit of communication in order to recover the measurement statistics on a qubit exactly.

Suppose that Alice wishes to send a quantum state $\rho_\psi$ of a qubit to Bob, so that when Bob chooses a particular measurement he is able to output measurement statistics as if he had had $\rho_\psi$ in his possession. Further, suppose that the only communication allowed between Alice and Bob is over a channel, and that we can measure the extent to which the channel is used. A simple protocol for achieving their task is for Alice to send a description of the state over the channel to Bob, for Bob to prepare a system in that state and then effect his choice of measurement on that system. This simple protocol requires a channel with a capacity large enough to contain the entire description: if we allow Alice to send arbitrary states the channel capacity will have to be infinite.

Somewhat surprisingly, using quantum systems Alice and Bob can achieve this task with a communication channel of just two (classical) bits and a shared resource of two qubits $|\Psi\rangle \in \mathcal{H}^4$ – one each – prepared as a maximally entangled pair. There turns out to be a close relationship between $\psi$-epistemic OMs as we have described here and classical protocols that simulate the quantum teleportation protocol.

What is counter-intuitive about quantum teleportation? On the face of it, the apparent information communicated between parties is infinite, yet the protocol requires only two
Quantum teleportation is a protocol in which Alice can ‘send’ a qubit state to Bob using a 2-bit channel and a pair of maximally entangled qubits. The process involves three steps. Firstly, Alice performs a joint (2-qubit) measurement on the qubit state that she wishes to send and her qubit from the shared resource. Next, she sends the outcome of the measurement to Bob, which requires a channel with a capacity of two bits (there are four possible outcomes to Alice’s measurement on the two qubits in her possession). Finally, Bob performs one of four transformations on his qubit from the shared resource according to the message that he received from Alice.

Bob will find that whatever measurement he effects on the qubit in his possession, the distribution over the outcomes is as if he had performed the measurement on the state that Alice wished to send. In this sense, Alice has successfully teleported her qubit to Bob.

As described by Montina, any $\psi$-epistemic ontological model can be translated into a classical simulation protocol for quantum teleportation [Mon12]. This gives an indication of the communication complexity of the quantum teleportation protocol; i.e. the amount of classical resource in terms of channel capacity that quantum teleportation can replace. Further, Toner and Bacon demonstrated the quantum process for qubits can be simulated classically by two parties with a shared source of (classical) randomness and a channel capacity still of only two cbits [TB03].

The model that we introduced above can provide an account for a classical simulation of quantum teleportation, albeit one in which occasionally a classical channel of unbounded capacity is required. On the other hand, the state to be teleported can be in arbitrary dimension (in particular $d > 2$).

In place of the shared entangled pair, Alice and Bob now have a source of shared (classical) randomness. The classical channel remains the same, and the figure of merit is the necessary capacity of the channel in order to simulate the teleportation protocol. The ontic state space of our model is split into its two components. The interval $\Lambda_x = [0,1]$ spans the shared randomness of the classical simulation. That is, before any communica-
tion both Bob and Alice have access to \( x \in [0, 1] \) with the promise that their values are equal. In order to establish some symmetry in the model, we could also impose that the shared randomness is used to establish the ordering of measurement outcomes (i.e. the centres of the epistemic islands). For simplicity we do not take this approach.

When Alice wishes to simulate teleporting a state \( |\psi\rangle \) to Bob, she performs the following actions

1. Check whether \( \lambda = (|\psi\rangle, x) \) is contained within any of the epistemic regions \( E_i \) (for the qubit, \( i \in \{0, 1\} \)).

2. If \( \lambda \in E_i \), send \( E_i \). Note that this can be carried out using a channel capacity of \( \lceil \log_2 d \rceil \) bits, for a \( d \)-dimensional system.

3. If \( \lambda \notin E_i \), send the full description of \( |\psi\rangle \). Note that this requires a channel of unbounded capacity.

If Bob receives a label corresponding to an epistemic region, he picks an ontic state from that epistemic region, orders the outcome of his chosen measurement appropriately and outputs the outcome according to the ontological model. If Bob receives the full description of the state, he outputs the outcome according to the ontological model using the ontic state \( \lambda = (|\psi\rangle, x) \) that was sent.

In order for an OM to translate to a classical simulation of entanglement requiring only finite channel capacity, it must have the property that the entire quantum state is never encoded in the ontic state of the system. This is true for the Kochen-Specker model: there, every ontic state can be prepared by more than one (indeed, a continuum) of the possible qubit preparations. That the communication complexity for the teleportation simulation in our model is unbounded is an indicator that the model contains ontic states that uniquely identify the preparation from which they arose. Further, the work of [LM12, Mar12] demonstrates that in order for Alice to send a state in \( d > 2 \), the classical channel must always have unbounded capacity. Nevertheless, we include the corresponding teleportation simulation protocol for our model as an interesting aside.
2.7 Summary

The OM that we present here is clearly very contrived, and is not meant to be a serious proposal for a post-quantum realistic theory. Rather, the primary motivation is to understand the formal limitations of reproducing quantum theory from a deeper or more fundamental theory.

Our result demonstrates that a $\psi$-epistemic OM is not ruled out by the basic assumptions required for a well-formed OM. This means that no-go theorems such as [PBR12] necessarily require non-trivial assumptions in order to rule out a $\psi$-epistemic theory. While many of these assumptions are eminently reasonable (indeed, without the ‘preparation independence principle’ used by Pusey et al. it is hard to justify the experimental method in general as being useful to science), it is nevertheless required. Elucidating the structure of the assumptions and the types of model that they rule out points the way towards finding precisely which aspects of quantum theory give rise to the peculiar phenomena associated with it.

On the other hand, almost all of the advantages of the $\psi$-epistemic OMs are shared by the other type of epistemic view that we mentioned in the introduction to this chapter. For that reason, it is always possible for a physicist who finds the idea of real quantum states too unappealing to take the anti-realist approach, and to remain in good company.

Finally, and perhaps representing the best way forward, it is possible that we should not seek to reproduce the predictions of quantum theory from an underlying ontology that has the same structure as classical physics. This position is reinforced by the observation that even a $\psi$-epistemic OM of the form that we have described must be contextual (by Kochen and Specker’s theorem) and non-locally causal (by Bell’s theorem). Our work should therefore be viewed with these qualities very much in mind, but also as a clue for characteristics of possible schemes that will follow.
Networks, graphs and communities

In this and the next chapter, we provide a review of some of the key concepts and approaches in finding communities in the network representations of real systems, and then introduce a new centrality measure based on quantum walks that can be used in an algorithm for community detection.

Networks can be found in many physical and non-physical situations, and involve sets of actors and interactions or relationships between them. For example, the network could represent a set of people and the friendships that exist between them, or it could represent points in a town and river bridges between them, or it could represent a set of atoms in a molecule and the local interactions between those atoms. Sometimes these networks are labelled complex networks, on the grounds that they are usually neither regular nor entirely disordered. The number and diversity of applications of research investigating complex networks is large and growing – and inevitably includes examples where quantum dynamics are in play.

On closer inspection, it quickly becomes obvious that the relationships and interactions in most realistic networks contain non-trivial structure. For example, webpages tend to link
to webpages about similar topics, children tend to be friends with others in their class or school etc. This structure can be summarised as there being inhomogeneities in the density of relationships amongst the actors, and can be observed across a wide range of realisations of networks. Elements such as these are said to be ‘structural’ if its scale lies in between the microscopic level of individual actors or relationships and the macroscopic scale of the entire network.

Just as structure can be observed in a wide range of network types, there is a similarly large range of motivations for discovering network structure. Understanding the structure of networks is an important part of understanding the systems in which they are found, but often the structure of the system is not directly observable. As well as the obvious commercial advantages of being able to establish structure in a network (e.g. ranking worldwide web pages according to the rate at which they would be viewed by a random web surfer [PBMW99]), the ability to make simplifications of systems that respect underlying structure allows us to simulate larger systems with relatively little resource.

A common approach to improving our understanding of systems that admit a network description is by examining the behaviour of dynamical processes within them. The dynamical approach makes sense since our experimental access to the systems – or at least their component parts – is often mediated by dynamical processes. As well as that, characteristic features of the dynamics, such as the time it takes for a signal to spread from one actor to another, are influenced by the structural properties, suggesting that studying the dynamics is a sensible route to discovering the structure; this is obvious to anybody who has tried to access a web page hosted in a different continent. Indeed, the motivations for gaining a deeper understand of network-like systems is not new, and there is a large and varied history of approaches to revealing network structure, including using dynamical methods (see [For10] and references therein).

In addition, as the resolution of sensors increases and our ability to measure systems is overtaking our ability to simulate them exactly, the need for faithful simplifications becomes more pressing. With very few exceptions, the dynamical methods to date have only used the dynamics of classical systems, such as packet flow in the world wide web or charge flow in electronic networks. But one of the arenas in which sensor technology has
given us a detailed network-type view of systems is quantum biology [ECR+07, LCF07, CGSC+09, RMS12, LCC+13, PLLC13].

For example, in [CGSC+09] the authors are able to infer the locations of excitons as they transfer through a network of pigment and protein in the ‘antenna’ component of a light harvesting complex (LHC). The LHC is the component of the photosynthetic system that gathers energy from the sun or other source of radiation (in the form of excitons created by incident photons) towards the reaction centres where the energy is stored in various chemicals. There are a number of different types of LHC whose precise details are becoming better understood, but for our purposes it is sufficient to note that an individual antenna typically contains ~ 250 – 450 molecules acting as sites over which energy is transferred; a picture of such a network can be found in [CGSC+09]. Despite the increase in sensor resolution, researchers interested in simulating the dynamics in these large LHC networks have simplified the task using a mesoscopic representation created by coarse-graining the system manually, as pointed out in [FMJ+14].

Clearly for these types of problem it cannot be assumed that classical approaches are meaningful, motivating the question: is it possible to unfold community structure using only quantum dynamics?

Just as classical dynamics is influenced by the characteristics of a network, the dynamical properties of these biological processes – and importantly the efficiency of the transport – is related to the structural properties of the complexes in question [RMS12]: where the structural properties are altered the efficiency of the process is observed to decrease. This provides a hint that quantum dynamics are directly influenced by the structural properties, and could be suitable for community detection.

In the next chapter we provide a positive answer to the question of using quantum dynamics to reveal community structure. The aim in this chapter is to lay the essential groundwork and background for our result – not to provide a comprehensive overview of this large and diverse field, for which the reader could consult [For10].

Firstly, we describe the graph representation of networks and introduce the heuristic notions of partitions and community detection. Next, we describe two dynamical processes
that are common in network systems: classical random walks and quantum walks.

With a view to using quantum dynamics to discover network structure, we review some algorithms related to our result that represent the state of the art in using classical dynamics for community detection, as well as the closely-related task of quantifying the quality of a network partitioning with respect to community structure.

In the next chapter, we check that our algorithm returns sensible results using benchmark graphs. So to complete the groundwork in this chapter, we discuss the three types of benchmark graphs that we will use. Benchmark graphs are examples where community structure is added ‘by hand’, with a view to that particular structure being recovered by a successful algorithm – i.e. there is a ‘correct’ answer that a good algorithm should return. In particular, we introduce a new form of benchmark graph based on ‘glued trees’ [CFG02] that we will use in the following chapter as a warm-up to the main result, for which we use the more realistic benchmark graphs of [LFR08]. This particular class of benchmark graph is the standard within the field of (classical) community detection due to a number of qualities that closely reflect properties of real networks, as we note below.

Finally, we draw all the threads together in a summary, ready to present our new work.

### 3.1 Graph representation of networks, partitions and communities

Fortunately the formal representation of networks as graphs is charmingly simple – here we briefly describe the graph representation in order to fix notation.

In each of the systems mentioned above we can represent the actors and relationships by two sets of objects: vertices and edges. This pair of sets is called a graph, and provides an abstraction of the network in which we can explore structure. In a social network, for example, each vertex represents a person and each edge represents a friendship between two people.
The complete network is associated with a graph $G$:

$$G = (V, E)$$

(3.1)

with each vertex label $v_i \in V$, the set of vertices, and $E \subset V^2$ the set of pertinent relationships as ordered pairs. When two vertices $v_i$ and $v_j$ are connected by an edge (i.e. $(i, j) \in E$) we say that $v_i$ is a neighbour of $v_j$.

In principle we can represent the strength of relationships between vertices as a relative weight $w((i, j))$ attached to the edge, expressed as a real number. Furthermore the weight need not be symmetrical. In a social network example, we might have that Alice has a weight 8 friendship with Bob, but Bob is less keen on Alice and has a weight 0 friendship with her – perhaps reflecting the number of interactions that each initiates in a usual week. In this case we would say that the friendship between Alice and Bob is directed.

We mainly consider undirected graphs, and except where explicitly mentioned set the weight of each edge to 1. (Note that usually relative weights are important, so the exact choice here is not significant, but the uniformity is.)

The number of vertices $N$ is the size of the graph, and the number of edges is $M$. All of the information about the edges and vertices can be collected into an $N \times N$ adjacency matrix $A_G$ whose entries describe the edges between pairs of vertices:

$$(A_G)_{i,j} = \begin{cases} 
1 & \text{iff } (i, j) \in E \\
0 & \text{otherwise.} 
\end{cases}$$

(3.2)

(Since it is usually unambiguous which graph we are currently examining, we often drop the label $G$. ) Each vertex has associated with it a degree, which is the number of neighbours of that vertex. So for vertex $v_i$ in an unweighted graph, the degree $d(v_i) = \sum_j A_{ij}$. Note that $\sum_i d(v_i) = 2M$, since each edge is counted twice.

The edges incident and consequent to a vertex can be split into several classes. The number of incident edges is the in-degree, whereas the number of consequent edges is the out-degree: these two are equal for undirected graphs. Where a vertex is put into a cluster of other vertices – a concept we expand on below – the number of neighbours within...
the cluster is the intra-cluster degree, and the number of neighbours outside the cluster is the inter-cluster degree.

**Partitions and community structure in a graph**

The vertices of a graph can be grouped into subgraphs which form a cover of the original graph. That is, every vertex is allocated to at least one of the subgraphs. These subgraphs are called *partitions*, and a set of partitions that covers the original graph is called a *partitioning*. We restrict our attention to those partitionings when the partitions are disjoint: i.e. every vertex belongs to exactly one partition. Such partitions are sometimes called ‘clusters’, and we will use the two terms interchangeably in what follows. When we examine quantum walks below, the quantum walker will traverse an $N$-dimensional space spanned by the vertices: so the quantum analogue of the partitioning is a division of the quantum state space into sets of (vertex) basis states. Since the partitions do not overlap, these sets are mutually orthogonal.

Each of the $K$ partitions in a partitioning has a label $k \in \{0, \ldots, K-1\}$, so that a particular partitioning is given by a set of (vertex, partition label) tuples $C = \{(v_i, k) : v_i \in V\}$, and a particular partition $C_j = \{(v_i, j)\} \,(C \text{ is for ‘cluster’}).$

The number of possible partitionings of a graph with $N$ vertices grows quickly with $N$, being equal to the $N$-th Bell number, generated by [And98, Wei14]

$$B_N = \sum_{i=0}^{N-1} B_i \binom{N-1}{i}; \quad B_0 = 1. \tag{3.3}$$

An important set of partitionings are those that form *hierarchies*. Several partitionings can form a hierarchy if each partition of a lower level partitioning is fully contained within a partition of a higher level partitioning. That is, as we proceed up the hierarchy each partition on the next level up contains a set of partitions of the level that we just left. If we know that a graph contains a hierarchical structure and the range of possible sizes of communities at each level in the hierarchy, then this information can significantly reduce the size of the solution space for community detection problems.

In the context of the graph representation, a community is a partition with particular struc-
tural significance within the network. Although, as we shall see, there is little consensus around precisely what a community is, several qualitative conventions have emerged due to the relatively clear intuition and common-sense appreciation of communities in general. Indeed, the best approach may in any case depend on the network in question or the task that is being attempted.

Two features of this consensus are that vertices within the same community should behave similarly to each other with respect to the rest of the graph: i.e. they can justifiably be grouped together in a coarse-graining of the network structure. Below, we refer to this as reflecting a symmetry in the way that different vertices from the same community are related to the rest of the network. Secondly, the density of edges within a community should be high relative to the average density of edges in the graph. These two intuitions form the basis for vertex similarity measures and modularity of partitions – two concepts that we introduce below, and whose spirit our work in chapter 4 follows.

Just as with partitions in general, a community structure can be ‘layered’ into a hierarchy of communities. One practical outcome of hierarchy in the community structure of a graph is that we are then able to simplify the graph’s representation at a resolution – i.e. level of the hierarchy – of our choice. A common approach is then to ‘coarse grain’ the graph so that communities become vertices of a new graph, with edges between the new vertices just in case there were edges between vertices of those communities in the original graph (e.g. in the case of modularity optimisation [BGLL08]). A schematic for this approach is shown in figure 3.1.

Ideally, we are then able to use a resolution parameter so that the graph can be simplified to an extent that it is digestible for whatever purpose we have in mind. This may take the form of a halting condition for an agglomerative algorithm, such that the algorithm stops after some number of steps. An example of the hierarchy at various stages of a generic agglomerative algorithm, along with a halting condition that makes the algorithm return a partitioning containing five partitions is shown in figure 3.1; the dashed line corresponds to the point where a halting condition could plausibly be reached.

Even when we fix a particular intuition as a ‘target’ for detection algorithms to aim at, detecting communities in networks is a hard problem. The ‘brute force’ approach of com-
Figure 3.1: Example dendrogram for a generic agglomerative algorithm that clusters togeth
er pairs of partitions as the algorithm progresses. At each step a halting condition is
checked, which may reveal that the dashed line represents the optimum stopping point
(note that at the bottom of the diagram all 14 vertices are in their own partitions, whereas
after 100 steps there is only the trivial partition). The partitioning (with each group cor-
responding to a cluster) is: \([ [1,2,3,4], [5],[6], [7,8,9,10,11,12], [13,14] ]\).

paring every possible partitioning against the target quickly fails as the graph size in-
creases, making such an approach tractable only for graphs containing a handful (per-
haps low 10s) of vertices.

A community detection algorithm is useful if it has two properties: it returns sensible
communities, and it returns its results quickly. The second of these properties is straight-
forward to compare between algorithms: we are usually able to express the resources
(both in time and memory) required by a particular algorithm in terms of graph size and
number of edges. The first property is the challenging one to achieve. We can formalise
the intuition that gives us a target as a \textit{quality measure}, which assigns a score to parti-
 tionings based on the graph. This allows us to compare algorithms quantitatively with
respect to that intuition, but nevertheless we face the problem of picking an appropriate
quality measure to begin with.

All of the dynamical algorithms that we describe can be categorised into one of two fam-
ilies: divisive and agglomerative algorithms. Although many algorithms from both families
have a particular halting condition that determines when the algorithm should return its
working partitioning, leaving this halting condition out typically leads to an hierarchy
of partitionings as discussed above. It is in this hierarchy that the difference between an agglomerative and a divisive algorithm becomes clear. In the divisive case, the first partitioning that could be returned is that where the entire graph is contained in the same partition. As the algorithm progresses, the graph is split into more and more partitions, possibly until every vertex is in its own partition. The purpose of the halting condition is to ensure that the algorithm returns at the optimal point – usually with respect to some quality measure (see below) – in the hierarchy.

An agglomerative algorithm works in the opposite order. At the first pass, vertices are assumed each to belong to a separate partition. As the algorithm progresses, vertices are grouped together into larger and larger partitions until the entire graph occupies a single partition. Once again, a quality measure is usually used in order to halt the algorithm at an appropriate point and return the best partition.

With respect to figure 3.1, an agglomerative algorithm works from the bottom upwards, whereas a divisive algorithm works from the top down.

Utilising the hypothesis that when structure affects dynamics, dynamics ought to reveal structure, a group of algorithms has emerged that use dynamical processes in the graph representation. After a brief review of classical and quantum dynamics in networks, the rest of this chapter is about those approaches, both in terms of discovery algorithms and also in terms of quality measures.

### 3.2 Dynamics in networks

Often a network acts as the setting for dynamical processes; that is, there is some notion of a system traversing the network according to the local features described by its edges.

An important type of classical process that the graph can support is a random walk. In this section we describe two continuous-time processes: the classical random walk (CRW), and one of its quantum cousins, the continuous time quantum walk (QW). A QW is a unitary process in a quantum state space spanned by the vertices of the graph.

The other cousin – the discrete time quantum walk – requires a larger state space than
the continuous time quantum walk. In addition to the graph itself, each vertex $v_i$ has associated with it a ‘coin space’ of dimension $d(v_i)$ so that, in the larger space, the walk can proceed unitarily – that is, without intervention in the form of measurement between each step.

The reduced state of the walker in the space spanned by the graph is usually not a pure state with support on a single vertex, since it is generally entangled with the (unobserved) coin states. The mixedness of the reduced state in the vertex subspace, as well as coherent effects, lead to the ‘randomness’ of the discrete time quantum walk. We do not consider discrete time quantum walks in detail, but where there are relevant features we shall pick them out in passing.

**Classical random walks on graphs**

In a CRW, a walker traverses the graph with a trajectory that forms a series of vertex labels $w(t)$, one label for each time. A continuous time CRW can be obtained as the limit of short time steps for a discrete time CRW. In a discrete time CRW, the rule governing the dynamics is that at each timestep the walker traverses an edge from its present vertex to one of its present vertex’s neighbours chosen uniformly at random: e.g. if $w(t) = v_i$ then $w(t + 1) \in \{v_j : (i, j) \in E\}$. Since the future trajectory of the walker depends only on its current state, we say that the walk is Markovian, and in general we are not able to predict with certainty the position of a walker in the future.\(^1\)

Nevertheless, given information about a walker’s position at time $t$ in the form of a distribution $p(t)$ over $V$, we can calculate the probability to observe the walker at each vertex at the next time step as

$$p(t + 1) = AD^{-1}p(t)$$

where $D_{ij} = \delta_{ij}(d(v_i))$, the diagonal matrix of vertex degrees, and $AD^{-1}$ is conventionally called the transition matrix for each step.

In order to progress from the discrete time CRW associated with this transition matrix

\(^1\)The only exceptions are pathological cases of graphs with no edges, or the graph where the out-degree of every vertex is $=1$. 

to the continuous time case, we reduce the step length while at the same time adding a ‘laziness’ parameter that gives the (eventually instantaneous) probability for the walker to step. We can set the number of ‘hops’ per unit time to 1, and consider a new step length $c$, so that

$$p(t + c) = (cAD^{-1} + (1 - c)\mathbb{1})p(t) = (c(A - D)D^{-1} + \mathbb{1})p(t).$$ (3.5)

In the limit of the step length $c$ going to 0, and for $n = T/c$ steps we have

$$p(t + T) = \lim_{n \to \infty} \left( \frac{T}{n}(A - D)D^{-1} + \mathbb{1} \right)^n p(t) = e^{T(A - D)D^{-1}} p(t)$$ (3.6)

which gives the continuous time CRW. The matrix $L = (A - D)D^{-1}$ is called the Laplacian of the graph. As well as being the generator for the CRW, its properties are at the heart of several non-dynamical detection algorithms that we mention in passing below.

A CRW on a connected graph evolves in the limit of long time towards a stationary distribution $\pi$ where the population at each vertex $\pi(v_i)$ depends only on the degree $d(v_i)$. To see this, note that since rows of $A - D$ sum to 0, the vector $1 = (1, \ldots, 1)^T$ is an eigenvector of this matrix with 0 eigenvalue. Inserting the identity operator as $\mathbb{1} = D^{-1}D$ into the eigenvalue equation we get

$$(A - D)1 = ((A - D)D^{-1})(D1) = LD1$$ (3.7)

Normalising this eigenvector gives the stationary distribution of the walk:

$$\pi_C = \frac{D1}{\sum_i d(v_i)} = \frac{d}{2M}.$$ (3.8)

Notice that the population at each vertex in the stationary distribution depends only on the degree of the vertex, and the total number of edges in the graph: a microscopic and macroscopic quantity. The structural elements that we hope to uncover have a scale in
between these two extremes, and so we must examine the trajectory of the dynamics – not just the initial and final distributions – in order to have any chance of uncovering intermediate features of the graph. We consider one such approach in the next section.

**Digression: non-dynamical algorithms for community detection.** Our main focus will be on algorithms that make use of dynamical processes on a graph. It is worth mentioning, however, that a significant proportion of approaches to community detection do not make use of dynamics. Dynamical approaches have emerged only relatively recently, and so non-dynamical approaches are well established.

One particular class of non-dynamical algorithms that makes use of the Laplacian matrix utilises the spectral properties of the Laplacian – i.e. its eigenvalues and eigenvectors. These can reveal structure directly from the graph description (for a comprehensive review, see [For10]). As an example, the number of eigenvectors of the Laplacian with zero eigenvalues gives the number of connected components of the graph.\(^2\) The support of those eigenvectors gives the sets of vertices comprising the components. The reason for this is similar to the stationary distribution argument: if a walk is initialised wholly within a single connected component of a larger, disconnected graph, then the walk will evolve to a stationary distribution over the initial component only, with support on the rest of the graph remaining = 0.

Furthermore, representing all of the eigenvectors as points in an \(N\)-dimensional Euclidean space, the eigenvectors can be clustered using common algorithms, such as a \(k\)-means approach (see [DH73] and [Mac67] for \(k\)-means clustering). The union of supports of the corresponding eigenvectors gives a partitioning that reflects community structure, with a resolution according to the number of means picked in the \(k\)-means algorithm.

To summarise, the CRW can be written as a discrete process that, in the limit of small time-steps, gives a continuous time CRW. The evolution of the walk is generated by the Laplacian of the graph, and the long-term (stationary) population at each vertex depends only on the local degree and the total number of edges in the graph.

\(^2\)i.e. where any vertex can be reached by a walker travelling from any other vertex.
Quantum walks on graphs

As we mentioned in the introduction to this chapter, it has recently become clear that network representations are appropriate for systems that are described by quantum theory – for example transfer of excitations through large molecules in LHC II as in [PLLC13]. There is no direct quantization of a CRW, though there are two quite separate quantum walks that are analogous to the discrete and continuous CRW, introduced by [AAKV01] and [FG98] respectively.

In our work, we are interested in closed processes where all dynamics of the quantum system is given directly by the structure of the graph. As we mentioned in section 1.2, a closed quantum process in the Schrödinger picture is associated with a continuous unitary transformation of the quantum state. This unitary transformation translates the quantum state along a surface in the quantum state space – for example, unitary transformations of qubit states correspond to the Bloch vector of the state exploring a shell of constant radius in the Bloch ball. Further, the transformation is deterministic in the state space, which is why we refer to quantum walks rather than quantum random walks.

The state space of our continuous QW is spanned by $N$ orthogonal vertex states: we assume that we are always able to measure the quantum walker in the vertex basis, projecting the walker into a state with support only on that vertex. This is where the ‘randomness’ comes in. Although the evolution is deterministic, the state of the walker in a typical graph is almost always in a superposition spanning a large number of the graph’s vertices. As such, the outcome of measurement in the vertex basis cannot generally be predicted with certainty. The observable ‘signal’ that can be built up through repeatedly running the quantum walk on the graph is therefore a good approximation of the probability to observe the walker at each vertex with time, akin to the probability to observe the classical walker at each vertex.

Typical signals from each of the types of walker can easily be distinguished by direct inspection. Except in the case of directed graphs,$^3$ CRW probabilities vary slowly, and

---

$^3$For a directed graph, we can obtain cyclic elements with population that flows around the cycle periodically. This creates fluctuations in the populations at vertices contained in the cyclic element.
converge to the stationary distribution, whereas QW probabilities fluctuate with a wide spectrum of frequencies, without converging.

As with other closed quantum systems, the evolution of the quantum walker’s state is governed by the Schrödinger equation

\[ i \frac{\partial}{\partial t} |\psi \rangle = H_t |\psi \rangle. \] (3.9)

We can then write the state of the walker at time \( t \) as \( |\psi(t)\rangle = U(t) |\psi(0)\rangle = \exp(-iHt) |\psi(0)\rangle \).

Just as in the classical case, the structure of the graph and its local interactions are reflected by the walk’s generator – the Hamiltonian. In order for the evolution of the quantum walker to be unitary, the Hamiltonian must be Hermitian, which is not generally true of the Laplacian generator \((A - D)D^{-1}\) that we use for CRW. An important exception (that we make use of in the next chapter) is where the graph is regular (all vertices have the same degree). We choose \( H = D^{-1/2}(A - D)D^{-1/2} \) for all of the example graphs that we use regardless of whether they are regular graphs or not, and also consider only un-directed graphs so that the Hamiltonian remains Hermitian.

Our choice of Hamiltonian (and indeed the particular form of the Laplacian in the classical case) has a particular physical interpretation [FJB+13]. The diagonal elements of the Hamiltonian give the energies associated with each of the vertices of the graph: our choice of Hamiltonian implies that the energies are all equal. Conversely, in the classical case, we chose to have a uniform hopping rate \( c \) for the walker. Thus the net flow of walkers out of every vertex relative to population (and regardless of degree) is constant. The two generators – the Hamiltonian with uniform vertex energy and the Laplacian with a constant hopping rate (as used in equation 3.6) – can be related as

\[ D^{-1/2}LD^{1/2} = D^{-1/2}[(A - D)D^{-1}]D^{1/2} = H. \] (3.10)

Whereas in the classical case the second ‘\( D'\)'-term of the Laplacian is related to the hopping rate of the walk, in the quantum case it expresses a shift in energy over all of the vertices. As such, the term is more physically significant in the classical case, and can just-
tifiably be dropped for the quantum walker. Indeed, one plausible option for the Hamiltonian would simply be the adjacency matrix $A$.

Since unitary dynamics maintains the inner product of pairs of quantum states, the dynamics of a continuous time QW are entirely dependent on the initial state of the walker. That is

$$U(t)|\psi\rangle \neq U(t)|\phi\rangle \quad \forall U, t, |\phi\rangle \neq |\psi\rangle.$$ (3.11)

Specifically, since the inner product between the walker’s state and the vertex basis states determines the probabilities, any two initial states differing by more than a global phase generally result in different observed signals.

As well as the strong dependence on initial state, QWs do not converge to a steady state as CRWs do (eq. 3.8). This means that there is no obvious notion of the walk having ‘finished’. A common compromise for the lack of steady state for the quantum walker is to take the long-time average signal of the walker at each vertex (as used in [FJB+13], for example):

$$\pi_Q(v_i) = \lim_{T \to \infty} \int_0^T dt \langle v_i | U(t) \rho(0) U^\dagger(t) | v_i \rangle.$$ (3.12)

This is operationally equivalent to performing the vertex basis measurement at a time sampled uniformly at random from a time range that is long relative to the characteristic time scale of the walker. This method is used in (for example) the closeness function of [FMJ+14] and the quantumness of [FJB+13], and has the appealing feature that the length of time that we allow the walk to evolve does not need to be measured precisely in a practical realisation of their algorithm.

As for the symmetry broken by dependence on initial state, in some cases we are able to re-establish the symmetry by averaging a quantity derived from the walk over all starting configurations of the walk. On the other hand, the lack of symmetry can in other cases be an advantage: in an agglomerative algorithm, we are often interested first in gathering together vertices in a particular part of the network, shifting our focus to the rest of the network as necessary.

---

4A couple of viable options here might be the recurrence time of the walk, which is generally very long, or the time for the average recorded signal from each vertex to be within $\epsilon$ of the average signal as $T \to \infty$. 
3.3 Using CRW to investigate community structure in networks

The output of a successful community discovery algorithm is an assignment of every vertex to one or more (in case we allow overlaps) of the communities; thus community detection is closely related to graph partitioning. However, whereas the input to a partitioning algorithm generally specifies the number of partitions to return, community detection is a more loosely specified problem in which the number of partitions to return is not fixed.

There is a smörgåsbord of algorithms designed to return partitions that reflect community structure in graphs. Many of them aim to maximise particular community quality measures (see below) – of which there is a similar range corresponding to the various motivations that we might have for wishing to find the community structure in the first place.

Of the many practical reasons why we might wish to discover community-like structure, several are related to the dynamical properties of a graph. Perhaps we are tasked with administering the load placed on some service that is distributed over computing instances in different data centres. Or we are designing a system for robust transport of some resource between pairs of locations (vertices), and would like to know which roads (edges) are most important. In these and similar cases, the dynamics is explicitly the subject of our interest, so it makes sense that we should aim to view the structure through the lens of dynamics.

In both these dynamical cases, we would like to be able to narrow the solution space of the problem by finding equivalences or similarities between the ways in which different subsets of the graph behave, especially with respect to the other vertices in the graph. This leads us to concepts such as vertex similarity and centrality that we expand on below. The measure that we introduce in the next chapter is an example of a centrality measure, and we use an algorithm that groups vertices with a similar value of the centrality measure in order to uncover community structure.

Vertex similarity. One general maxim for community detection is to regard communities
as collections of vertices which are in some sense similar to each other (see [New05] for a brief review). The simplest example is perhaps the degree (and particularly the in-degree) of vertices, broadly reflecting the popularity of a particular actor in the network.\footnote{The in-degree in this context is the number of actors that follow the actor of interest; so information flows from the actor of interest to the following actors.} Taking an out-degree in the context of an information-flow network, on the other hand, might indicate an actor who acts as a news wire: gathering information from a wide variety of actors and disseminating to relatively few.

Another clear example is that where there is an embedding of the vertices of a graph in space. Typically in such a situation the edges of the graph will adhere to some dynamical principle; for example, the edges might behave like springs and the layout of vertices will relax to such a position that the energy stored in the springs is minimised. If this is possible, then the distance between vertices forms a measure of the \textit{dis}-similarity between vertices. That is, vertices that are far apart are allocated to different clusters. Even in cases where there is not a given embedding of a network in space, we can induce such an embedding using a graph layout algorithm that involves the edges: such as the spring layout. (See [FR91] for a specific example.)

\textit{Centrality measures}. Centrality measures give a quantitative indication of the importance of an element – vertex, or less commonly edge – of a graph. We can in principle also assign centralities to structural elements, for example by coarse-graining a graph in the case of an hierarchical clustering. There are several different types of centrality measure; all of them give an indication of when different elements of the graph play a similar role in the graph. Examples related to dynamics are mentioned in [New05]. In general, the approach is to find the extent to which elements play an important role in the dynamics. Examples of this might be the number of shortest paths between pairs of vertices that traverse an edge, averaged over all pairs of vertices [Fre79]. A natural development of this approach considers the maximum flow of information from one vertex to another, which includes along paths that are not optimal. On the other hand, in some sense the process still needs to ‘know’ the most direct path [FBW91].

\textit{Random walk betweenness centrality}. The work of Girvan and Newman [NG04, GN02] is
cited as having marked ‘a new era in the field of community detection and opened this
topic to physicists’ [For10]. In their paper they present several approaches, one of which
involves a particular betweenness centrality measure based on a CRW process on the
graph (one of several related measures that they and following authors have defined).
Such an approach makes sense because often a reasonable assumption is that the walker
itself does not have information about which is the best path between two vertices. Such
a situation could be imagined in an anonymised internet, where parties wish to hide not
only the content of their messages but also who is communicating.

Girvan and Newman’s approach proceeds as follows. Consider a CRW where the walker
is initialised at a source vertex \( v_i \) and evolves until reaching a particular sink vertex \( v_j \).
The betweenness of an edge is given by the net number of times a walker traverses that
dge during such a process, averaged over all pairs of vertices. Implicitly, it is assumed
that the graph is connected such that the process will always complete. Importantly, as
Newman points out, the random walk approach gives weight to important features of
graphs that are not accounted for in previous edge betweenness strategies (see examples
in [New05]).

Having averaged over all pairs of source and sink vertices, one obtains an assignment
of edge betweenness for every edge in the graph. Those edges that played a prominent
part in more of the processes will have a higher edge betweenness than those that did
not. This gives an indication of which edges in the graph are most central to dynamics
over the graph. The algorithm for community detection on an input graph \( G_0 \) proceeds
as follows [NG04]:

1. Calculate the edge betweenness for the graph \( G_i \).

2. Remove the edge with highest edge betweenness \(((i, j), \text{say})\) giving a new graph
   \[ G_{i+1} = (V_{i+1}, E_{i+1}) \text{ with edges } E_{i+1} = E_i / (i, j). \]

3. Repeat from step 1. until a stopping criterion is satisfied (e.g. there are \( K \) connected
   subgraphs remaining).

4. Return the partition \( C = \{(v_i, j)\} \) with \( j \) the label of the connected subgraphs, \( j \in
   \{0, \ldots, K - 1\}. \)
Figure 3.2: Simple example graph for edge betweenness measurement, and result over all pairs of source/sink. The labellings correspond to the labels that we use in the text. We expect in this case that the edge $(2, 3)$ will be identified as having the highest CRW edge betweenness when summed over all pairs of source and sink vertices – and indeed this is what we see.

An example is easiest to follow using a discrete time CRW. We use the example shown in figure 3.2a.

The result of the summed CRW edge betweenness summed over these source-sink permutations is shown in figure 3.2b. We normalise the results by dividing by the maximum edge betweenness over all edges. Note that in this case the algorithm would remove the edge $(2, 3)$ in the first iteration, but then (as can be seen from the symmetries of the graph), there would be a six-way tie in the edge betweenness of the remaining edges. Therefore the dendrogram returned for the Girvan-Newman algorithm would contain the partitionings $[0, 1, 2, 3, 4, 5, 6] \rightarrow [[0, 1, 2], [3, 4, 5]] \rightarrow [[0], [1], [2], [3], [4], [5]]$, where ‘$\rightarrow$’ represents each iteration, and vertices in the same partition are grouped in the square brackets.

A quirk of random walk edge betweenness – not uncommon among community detection algorithms – is that there will occasionally be a tie in the edge betweenness. Often in this and other similar algorithms the strategy in such a case is to pick one of the edges at random to remove, or make some arbitrary choice. Worth keeping in mind is that in some cases this can affect the outcome of the algorithm, or at best make it non-deterministic,
yet without the bias introduced by some arbitrary choice.

The Girvan-Newman algorithm is divisive; it belongs to the family of algorithms in which the entire graph is divided into communities, rather than the family where vertices are added to clusters until every vertex is classified. This means that the series of graphs $G_i$ describes a hierarchy of connected graph components. The ‘lifetime’ of each of these components (i.e. the number of $G_i$ in which a particular component is connected with respect to itself, yet separate from the rest of the graph) gives an indication of the significance of that component in the hierarchy. A component that features in more iterations is more structurally key.

Although the algorithm that we present in chapter 4 is agglomerative, it is similar in spirit to the CRW edge betweenness algorithm presented here. That is, we shall see that a centrality measure expressing the extent to which an element of the graph is ‘involved’ in a quantum walk can also be used to discover communities in graphs. Further, like the Girvan-Newman algorithm, the algorithm that we present is iterative: we use a centrality measure in order to group together vertices that play a similar role in the graph.

In order to test the efficacy of the centrality measure we introduce in the next chapter, we will explore its behaviour in a number of benchmark cases. Now that we have reviewed the classical algorithms that are spiritually similar to our quantum approach, we turn our attention to these benchmark graphs.

### 3.4 Benchmark graphs for community detection

A common approach to measuring how effective a particular algorithm is at discovering community structure is to test the algorithm in a case where the community structure is known. For example, one could study the structure of links between political blogs, which tend to be explicit about the parties that they support. Under the assumption that such blogs relatively rarely link to material that their readers would not be sympathetic to, a good community detection algorithm should be able to group together blogs supporting the same party relatively easily. But such real-world examples are not necessarily
easy to come by, so ideally we would like artificial sources of graphs with community structure on which we can test our algorithms.

*Benchmark graphs* fulfil this function. A benchmark graph is generated from a series of parameters specifying quantities such as the number of communities, their sizes and vertex degrees, possibly including a random seed such that each set of parameters can be associated with an ensemble of graphs. We call a set of parameters for a particular benchmark graph a parametrisation. As well as the graph, the benchmark includes a particular partitioning as ‘the’ community structure. The aim is then to return that partitioning using an algorithm that takes as input only the adjacency matrix of the graph.

We discuss three types of benchmark graph. The first two that we consider are highly artificial, and not designed necessarily to reflect the characteristics of real networks in all their generality, but contain very clear structures. Failing to identify these structures is a good sign that the algorithm is unlikely to be widely applicable. The third graph that we consider is the class of ‘LFR’ benchmark graphs [LFR08]. These graphs are designed more closely to reflect features of real networks.

**Artificial graphs**

Artificial graphs like those we introduce in this section are well-established as a category of benchmark graphs. One of the most popular classes of artificial benchmark graph was introduced by [GN02]. The community structure is ‘planted’ into a graph of 128 vertices: the graph is divided into four clusters of 32 vertices each. The average degree of the vertices in the graph is fixed at $d(v) = 16$, with edges preferentially added between pairs of vertices in the same cluster. The extent of this preference is set by a parameter $\bar{d}_{\text{out}}$, the expected number of edges from each vertex to vertices in other clusters.

When $\bar{d}_{\text{out}} < 8$ the four clusters are supposed to represent well-defined communities. Girvan and Newman’s construction is typical of artificial benchmark graphs, where community structure is relatively clear: failure to recover the planted structure is a good indication that a particular algorithm is not generally useful. On the other hand, even when

---

6 This choice of network size reflects the size of graph for which most algorithms could complete in a reasonable time when the benchmark was introduced.
the average intra-community degree is higher than the inter-community degree, a few vertices could end up more connected to vertices in a community not their own. This makes it non-obvious which community such vertices should be assigned to: the community with which they share most edges, or the seeded community.

The two examples that we introduce are even more artificial, but are designed to showcase two features of the centrality measure that we introduce in the next chapter. In both cases we consider a cycle of graph fragments, or subsets of vertices from the whole graph.

*Cycles of glued trees.* We introduce a particular class of benchmark graphs inspired by a case where it is known that there is a difference in performance of CRW and QW for solving an information processing problem [CFG02]. Childs et al. consider graphs consisting of two binary trees, each of the same height, and with the leaves of the two trees joined by edges. The construction is called a ‘glued tree’ (GT), with the edges joining the leaves representing the glue. An example is shown in figure 3.3.

![Glued Tree Example](image)

**Figure 3.3:** An example of GT₃, a glued tree of height 3.

Childs et al. demonstrated a separation between the speed with which a CRW and a QW traverse a GT that is exponential in the GT’s height. Namely, suppose that the walker is initialised with support only on one of the roots of the GT. Task the walker with traversing both trees and reaching the opposite root. The argument proceeds by noticing that the
GT graph can be simplified to a walk on a line with weighted edges. In the classical case, the weight of these edges grows exponentially as the walker ‘climbs’ the tree. In the quantum case, the edge weight remains constant, reducing the problem to the quantum walk on a line (initialised from an edge vertex). Further, by configuring the ‘glue’ edges in a particular way, Childs et al. demonstrate that a classical walker cannot traverse the GT efficiently even with a definite strategy (rather than strictly random walk) dependent on the local and past information available to the walker.

This separation in the performance of CRW and QW will become important in the next chapter, since our centrality measure based on QW can exploit this property to reveal the structure of GT cycles with similar efficiency.

The quantum walk on a line is important as one of the relatively few cases where we have an exact relation between the continuous- and discrete-time quantum walks [Str06]. In particular, one common feature is that the expected distance along the line travelled by the quantum walker in time $T$ is $O(T)$, whereas for the CRW the distance is $O(\sqrt{T})$. This difference, combined with the exponentially decreasing edge weight as the CRW climbs ‘down’ the tree (i.e. from the leaves towards the root), give the separation in performance.

Here we describe how to construct GTs, then combine them in such a way as to create a graph with a clear, albeit artificial community structure, and use the same simplification to make simulating walks on large cycles of high GTs tractable.

Consider a pair of binary trees $T_A$ and $T_B$ of height $h$; i.e. there are $h$ edges in the most direct route from the leaves to the root. Both of the trees have $2^h$ leaves. We now add ‘glue’ to the trees, joining the leaves. Label the leaf vertices of the two trees $0, 1, \ldots, 2^{h+1} - 1$, with even labels for the leaves of $T_A$ and odd labels for leaves of $T_B$. Taking the list of leaf labels, shuffle the even labels amongst themselves, and the odd labels amongst themselves (for example one might obtain $[2, 5, 8, 1, 6, 7, \ldots]$). This list fixes a particular cycle alternately visiting leaves of $T_A$ and $T_B$. Now add $2^{h+1} - 1$ edges between the leaves of the trees joining the $i$th element of the list to the $(i + 1 \mod 2^{h+1} - 1)$th element. A glued tree $\text{GT}_h$ is a graph of this form. We show a picture of a $\text{GT}_3$ in figure 3.3.

If we take several GT fragments, we can add edges between roots of the fragments taken
pairwise in such a way that we obtain a cycle of GT fragments. Such a cycle is shown in
figure 3.4, along with a simplification of the same graph that we describe below.

The size of glued tree graphs (and indeed individual glue tree fragments) grows quickly
with the height of the tree. In order to simulate dynamics on graphs containing high
trees without incurring an exponential computational overhead, we follow [CFG02] in
grouping vertices at a constant height from their closest root together. In doing this, we
take advantage of the fact that there are no edges between vertices in each of these groups.

The simplification proceeds as follows. For each tree, group the vertices by the number
of edges from the vertex to the nearest root. So the root vertex forms one group, its two
neighbours another, their four un-grouped neighbours a third etc. For a full $\text{GT}_h$ there are
then $2(h + 1)$ groups, and we label them from 0 to $2h + 1$. Notice that for group $k < h$,
there are $2k$ edges shared with vertices of group $k + 1$, and $k$ edges shared with vertices of group $k - 1$. Thus in the classical case, we can simplify the graph to a walk on a line of length $2(h + 1)$ with edge weights:

$$w_{ij} = 2^{-|h+j+1|},$$

the number of edges between the $i$th and $j$th groups. The vertex degrees in the simplified graph are simply the number of vertices in each group: $d_i = 2^{-|h-i|+h}$.

In the case of the quantum walk, the ‘weighting’ is different, giving a new Hamiltonian for the grouped edges $H_{grpd}$. Consider the instantaneous transfer between groups 1 and 2, which contain 2 and 4 vertices respectively:

$$\langle \text{grp 1}|H_{grpd}|\text{grp 2}\rangle = \frac{1}{\sqrt{2}} \frac{1}{\sqrt{4}} (\langle 0 | + \langle 1 |) H (\langle 2 | + \langle 3 | + \langle 4 | + \langle 5 |))$$

$$= \frac{1}{2\sqrt{2}} (\gamma + \gamma + \gamma + \gamma)$$

$$= \sqrt{2} \gamma$$

where $\gamma$ is the energy associated with each site and $|\text{grp k}\rangle$ a basis state in the grouped basis:

$$|\text{grp k}\rangle = \frac{1}{2(-|h-k|+h)^{1/2}} \sum_{i \in \text{grp k}} |i\rangle.$$

The only exception to this uniform weighting is for the edges that link each GT to its neighbour in the GT cycle. For these, the weight remains $= \gamma$.

Finally, we note that cycles of GT graphs contain vertices all of which have degree $= 3$. This means that we can simply use the Laplacian of the graph as the Hamiltonian for a quantum walker, since the uniform degree guarantees that the Laplacian is Hermitian (and indeed symmetrical).

To summarise, for both the CRW and QW we are able to simplify the graph to a cycle of vertices with weighted edges. In the quantum case, the weighting is uniform except on the edges linking one glued tree to the next. For the classical case, the weight of the edge increases in correspondence with the height of the grouped vertices that it links.
We will return to these GT cycle graphs in the next chapter, as we examine the properties of CRW and QW that traverse them.

‘Blob’ graphs. The second class of benchmark graphs that we consider also consists of a cycle of fragments. Each of these fragments consists of a fully connected set of vertices that we call a ‘blob’. Between pairs of adjacent fragments, we add an edge. Call the vertices at each end of this edge the ‘out vertices’. Each blob now comprises a collection of $N_c$ vertices, each of degree $N_c$ except for the out vertices which have degree $N_c + 1$. In order to restore the symmetry of each vertex in a blob having the same degree, we remove the edge within the blob between the out vertices.

An example of a blob graph consisting of four blobs of sizes 5, 10, 8 and 10 is shown in figure 3.5a. Note that the graph is completely specified by the sizes of the blobs and the order in which they are linked; we can swap the vertex labels within a blob without changing the structure of the graph that we wish to recover.

In the classical case, we are able to simplify the graph in the same way as with the glued trees: by grouping all vertices within a blob that do not have inter-blob edges together (see figure 3.5c).

In the quantum case, we again consider the elements of the Hamiltonian between groups of vertices. This time, each blob has three groups: two corresponding to those vertices that have inter-community edges (call them 0 and $n - 1$), and one group for all of the other vertices in the blob ($\{1, \ldots, n-2\}$):

$$|\text{grp}_{\{1,\ldots,n-2\}}\rangle = \frac{1}{\sqrt{n-2}} \sum_{i=1}^{n-2} |i\rangle. \tag{3.16}$$

We add a self-loop to the ‘other vertices’ group representing the propensity of the state to remain in the subspace spanned by those vertices. The elements of the Hamiltonian are then given in the same way as the GT case, and are labelled on the simplified graph in figure 3.5b.

The purpose of the blob graph is to demonstrate the behaviour of the centrality measure that we introduce in the next chapter where the graph has significant symmetry. We should expect (and indeed we shall see) that any reasonable centrality measure for every
(a) A ‘blob’ graph \(-B_{10} - B_8 - B_{10} - B_5-\) consisting of four fragments, of size 10, 8, 10 and 5. Note that there is exactly one edge between each blob, and that each blob has the edge connecting the ‘out vertices’ removed. Vertices in the same blob also share the same colour.

(b) QW simplification

(c) CRW simplification

Figure 3.5: A full blob graph (top), and a simplified blob of size \(n\) (bottom two). Note in the simplifications the self-loop on the central vertex, representing all of the vertices of the blob with only intra-community edges. The other two vertices represent those with inter-community edges, and the edges at the ends of the diagram are shared with the previous and next blob in the cycle.

A vertex without inter-community edges is equal: the vertices ‘see’ the rest of the graph in exactly the same way. On the other hand, the graph remains highly artificial and reflects few features of real graphs. Due to the simplification, we are able to simulate large blob graphs with relatively little resource.
**LFR benchmark graphs**

Finally, we consider the class of benchmark graphs introduced by [LFR08] – the ‘LFR’ benchmark. The purpose of LFR benchmark graphs is to mimic more closely several of the features observed in real networks that do not appear in the benchmarks considered above. Namely, the distribution of vertex degrees and community sizes in LFR benchmark graphs follow a power law distribution\(^7\). As an example, we show several LFR benchmark graphs in figure 3.6; the middle graph is the one that we use as a prototypical LFR example in chapter 4.

Real networks tend to have this kind of heterogeneity in their vertex degrees, which as [LFR08] point out explain a number of significant features of real networks. These features include resilience to failure of edges or vertices [AJB00] and the relative ease with which epidemics can spread [PSV01]. These features are not present in the artificial graphs that we described in the beginning of this section. As well as being a more faithful representation of real networks, LFR graphs are considered a challenge for most community detection algorithms [LFR08].

Construction of an LFR graph follows the specification of several target\(^8\) parameters and also a random seed. This means that two LFR graphs with the same input parameters \textit{a priori} but different random seeds will neither be identical, nor generally have the same parametrisation \textit{a posteriori}. The algorithm attempts to return a graph with the input parameters, though the discrete nature of the edges means this is not always possible – particularly for edges. Therefore the \textit{a posteriori} description may be different from the input parameters.

The input parameters for an LFR graph are:

- **The number of vertices**, which we shall fix at \(N = 400\).

- **The average vertex degree**, used to normalise the distribution of vertex degrees, and for our example graphs fixed at \(\bar{d} = 6\).

---

\(^7\)Also sometimes (more descriptively) called ‘long-tailed’ or ‘fat-tailed’ distributions.

\(^8\)It is not always possible for an LFR graph to satisfy all desired parameters simultaneously.
Figure 3.6: Example ‘LFR’ benchmark graphs of 400 vertices typical of those we use as tests for our algorithm in the next chapter. The parameters are: (number of vertices: 400; average degree: 6; maximum degree: 15; exponent for degree distribution: 5; exponent for community size distribution: 3). The mixing parameter is the only altered parameter between the three examples. The layout of the graph is according to the Graphviz NEATO algorithm, a type of energy minimizing spring layout [Nor04] and colours represent the four seeded communities. The middle graph (mixing: 0.128) is the one we use as an example in chapter 4.
- **Maximum vertex degree**, also used in the normalisation of the degree distribution.

- **Exponent for the degree distribution**, specifies the shape of the distribution from which the vertex degrees are sampled.

- **Minimum community size**, used to normalise the distribution of community sizes, and for our example graphs fixed at $N^{(\text{min})}_c = 80$.

- **Maximum community size**, also used in the normalisation of the community size distribution, and for our example graphs fixed at $N^{(\text{max})}_c = 120$.

- **Exponent for community size distribution**, specifies the shape of the distribution from which the community sizes are sampled.

- **Mixing parameter**, the proportion of edges which are inter-community (i.e., connect vertices that are members of different communities).

The construction of the LFR graph follows an iterative process described in [LFR08], but importantly converges quickly to give a graph whose parameters closely match the desired input parameters. So we are quickly able to generate a large number of graphs for which the community structure is known. Though there are several degrees of freedom that can be manipulated in generating LFR benchmark graphs, for the purpose of demonstrating that our algorithm presented in the next chapter returns appropriate communities, we are most interested in the performance with respect to the mixing parameter $\mu$.

It is worth noting that the seeded community structure is meaningful for $0 < \mu < 1/2$. Namely, the graph must be connected (an LFR graph with $\mu = 0$ is certainly disconnected), and the communities must have a higher density of intra-community edges than there are edges in the graph taken as a whole (note that the intra-community edge density is $1 - \mu$). Therefore in the next chapter we compare the results obtained by our algorithm over a range of mixing parameters.
3.5 Evaluating and comparing algorithms

A quality measure assigns a number to a particular partitioning of a graph designed to reflect how well that partitioning reflects the community structure of the graph. If the quality measure is higher, the partitioning better represents the underlying community structure than a partitioning with a lower quality measure.

A basic quality measure is based on the cut size: the number of edges that join vertices in different partitions, sometimes normalised as the proportion of all edges – and called coverage [For10]. If the partitioning reflects the community structure of the underlying network, then the cut size is generally low, relative to other possible partitionings. One of the earliest uses found for the spectral properties of the graph Laplacian, for example, was in estimating the minimum cut size for a bi-partitioning of the graph. Taking the eigenvector of the Laplacian with the lowest non-zero eigenvalue, partitioning the graph according to the signs of the components gives a partitioning with a low cut size [dA07].

Here we mention two prominent quality measures. Modularity is important because it simply reflects our initial common-sense approach of communities containing a relatively high density of edges. Stability of a partitioning is important because it provides a means to understanding both modularity and other quality measures as limits of a CRW process on a graph. Finally, in order to compare the performance of a partitioning algorithm with respect to a benchmark with seeded community structure, we also mention the normalised mutual information between partitionings: a common measure of algorithm performance.

Modularity

The modularity of a partitioning [NG04] is a measure of the relative number of edges within partitions with respect to some null model graph with the same number of vertices. The null model’s defining property is that we should not expect it to contain any
community structure. The skeletal form of all modularity measures is:

\[ Q = \frac{1}{2M} \sum_{ij} (A_{ij} - P_{ij}) \delta(C(v_i), C(v_j)). \]

Here, \( P_{ij} \) represents the expected number of edges between vertices \( v_i \) and \( v_j \) in the null model. The \( \delta(C(v_i), C(v_j)) = 1 \) if \( v_i \) and \( v_j \) are in the same partition. So since contributions come only from pairs of vertices partitioned into the same cluster, modularity measures the extra density of edges within the cluster, relative to what we might expected.

For the null model, there are a few natural choices having in common the feature of a random seed; the options are summarised in [For10]. Ideally we would like the community structure of the original graph to be the only feature that is dropped in the null model, so a good choice maintains the degree distribution of all the vertices in the graph.

A configuration model [Luc92] can be used to ‘re-wire’ the graph in such a way that the degree of each vertex is kept constant. First, break every edge in the middle so that each vertex has \( d(v) \) ‘stubs’ or half-edges connected to it. Next, label each vertex \( i \in \{0, 1, \ldots, N - 1\} \) and write each label \( d(v_i) \) times into a list of length \( 2M \). Shuffle the list, and add edges between each adjacent pair of vertices in the shuffled list.\(^9\) This approach gives an essentially random graph, except that the distribution of degrees remains identical.

**Stability across timescales**

Modularity is the most popular quality measure for partitions [For10]. Recently, however, Delvenne et al. [DYB10] have introduced a new measure that unifies several quality measures, including various modularities. The stability of a partitioning of a graph across time scales uses the dynamics of a CRW on the graph to give us a measure for how well the partitioning reflects a coarse-grained description of the walker’s dynamics.

An intuition about how the dynamics of a CRW would be affected by community structure in a graph is that communities form ‘dynamical basins’ in which walkers tend to

\(^9\)For a large graph the probability to obtain self-loops is small, though for the benchmark (LFR) graphs we use we reject any shuffling that results in self-loops.
get stuck. That is, the walker’s next step is more likely to be within the community than out of it, meaning that the entire trajectory can be divided into relatively long ‘chunks’ where with high probability the walker remains in the same community. A consequence of this is that if we group the signals from the vertices by the partition that they are in, then for a ‘good’ partitioning this grouped signal should be well-correlated with itself over relatively long times.

This intuition is formalised by Delvenne et al., where the time scale for which walkers remain in a community forms a natural resolution parameter for the community structure, giving a measure of the quality of a particular partitioning relative to others for a particular time scale. Given a particular set of partitions, the stability provides an indication of which partitioning is most successful at trapping the walker in its partitions at that time scale.

Delvenne et al.’s method utilises the autocorrelation of the vectorised signal from the vertices of a graph, grouped by a particular partitioning. The autocorrelation gives an indication of the self-similarity of signal at different times, calculated as the convolution of the signal with a time-shifted copy of itself. For the signal from the vertices, for example, the autocorrelation is given by:

$$R_{pp}(\tau) = \int_{-\infty}^{\infty} p(t+\tau)p(t)\,dt$$

(3.18)

where $p(t)$ is the distribution over the graph at time $t$. In order to calculate the stability, the method sums this quantity over vertices that are in the same partition, so that one obtains an autocorrelation matrix with elements between partitions, rather than between individual vertices. Taking the trace of this matrix gives the sum of the autocorrelations for the CRW to remain in the same partition – this is the quantity that Delvenne et al. call the stability of the partitioning. If a particular partitioning is stable for relatively long times, then it reflects community structure in the graph.

Given a particular set of partitionings, the maximum over the partitionings of the stability at a particular time gives an assignment of partitionings to time scales over which they are most ‘stable’. Thus the stability can be used as a target to maximise for community
detection algorithms in the same way as modularity.

The appealing feature of the stability measure is that over a range of time scales, the measure captures a range of quality measures that have been used in other contexts, as well as providing a natural resolution parameter in the time over which a partitioning is stable. Significantly for our work, this recently popular measure is intrinsically linked to the dynamics of a classical process on the graph.

**Normalised mutual information**

For benchmark graphs, we are interested in how closely the partitioning returned by a community detection algorithm is related to the seeded partitioning. Or, to put the question in a slightly more convoluted way, what is the information gap between the output of the algorithm and the correct partitioning?

This gap is a quantity commonly of interest in information theory, where we are interested in the information gap between the signal incident on a noisy channel and the signal obtained after the channel has been applied. Specifically, how much extra information do we need in order to reconstruct the input signal given the output signal.

In order to translate the channel problem to the partitioning situation, consider two random variables $X$ and $Y$ associated with values $\{x_i\}$ and $\{y_i\}$, the community assignments for vertex $v_i$ from two partitionings. One is the ‘correct’ partitioning given by the benchmark graph, the other is the partitioning returned by the algorithm.

First, define a ‘confusion matrix’ whose rows correspond to the correct communities and whose columns correspond to the returned communities. The entries of the matrix $N_{ij}$ correspond to the number of vertices in the correct community $C_i^{\text{seed}}$ that also appear in the returned community $C_j^{\text{find}}$:

$$N_{ij} = |C_i^{\text{seed}} \cap C_j^{\text{find}}|. \quad (3.19)$$

We write $\sum_i N_{ij} = N_i$. Then the normalised mutual information for two partitionings $A$
and $B$ is given by \cite{Mac02, For10, DDGDA05}

$$I(A, B) = \frac{-2 \sum_{i=0}^{c_A-1} \sum_{j=0}^{c_B-1} N_{ij} \log(N_{ij}/N_i/N_j)}{\sum_{i=0}^{c_A-1} N_i \log(N_i/N) + \sum_{j=0}^{c_B-1} N_j \log(N_j/N)}$$  \hspace{1cm} (3.20)

with $c_A$ ($c_B$) the number of partitions in partitioning $A$ ($B$).

The normalised mutual information can be thought of as the amount of information contained in the structure of the graph that is correctly extracted by an algorithm, and is commonly used in recent evaluations of detection algorithms. \cite{DDGDA05} gives examples of the normalised mutual information for a large variety of algorithms with respect to benchmark graphs introduced by \cite{GN02}.

### 3.6 Summary

In this chapter we have described and motivated the discovery of community structure in graph representations of networks. We also provided a brief overview of how CRWs can be used both to discover and measure the quality of partitionings with respect to that structure.

CRWs have shown success in partitioning graphs into communities that score well with respect to common quality measures, and often also succeed at recovering ‘seeded’ communities efficiently. Naturally, we ask whether a similar but quantum dynamical process that respects the local interactions of a complex network could also be used to discover community structure.

This question has at least two motivations. Firstly, there exist networks where the only dynamics that can practicably be implemented are quantum; for example the transport of an exciton in a large molecule. Secondly, we know that there are artificial graphs (GTs to name one example) where there is a separation in the performance of QW from their CRW counterparts: we wonder whether this separation can help us achieve as practical a task as community detection with similar efficiency. Thirdly, understanding the differences in the ways that QW and CRW ‘see’ graphs could be helpful in designing or translating algorithms based on the dynamics of particular graphs.
Having said this, there are fundamental differences between the ways that CRW and QW evolve: most strikingly, that CRWs converge to a steady state, whereas for the QW the unperturbed walk proceeds unitarily. This prompts us to ask whether some quantity derived from the population signal – and that can be measured for each vertex of the graph – might be more appropriate for community detection.

We also introduced three types of benchmark graphs that we will use in the next chapter as case-studies for our new centrality measure. The first two have the quality that there is a separation in the performance of CRW and QW in traversing the graph (GTs), and that there are significant symmetries within the graph fragments (blob graphs). The third type is considered a challenging type of graph for community detection algorithms to return the seeded partitioning. We will demonstrate that our centrality measure can be used as a component of a simple algorithm to return the seeded partitionings of LFR graphs with good fidelity: given the output of the algorithm, we shall compare it to the ‘correct’ structure seeded in the benchmark.

In order to make this comparison, we use the normalised mutual information measure, which allows us to compare not only the assignment returned by our algorithm to the seeded structure, but also our algorithm to classical alternatives.
4

Community detection with superposition states

In this chapter we explore whether superposition states arising from a continuous time quantum walk (QW) can be used to detect community structure in graphs, in a similar way to the classically mixed states associated with the classical random walk’s (CRW) Markov chain. The aim is to show that a process that is purely quantum is able to return seeded communities from benchmark graphs. By ‘purely’ quantum, we mean a closed process in which all of the dynamics are unitary. The distinction is important, because we would like to be sure that the structure that we deduce is not an artefact of classical phenomena introduced by any interaction causing decoherence of the walker. We would also ideally like for our approach to be able to exploit novel properties of quantum walks that we mentioned in the previous chapter, such as improvements in the rate at which a QW traverses some types of graph.

Given the range of successes and applications that we have seen in the previous chapter for using the properties of classical dynamics in graphs, it is natural to ask whether
quantum dynamics can also be of use. Since the relationship between the QW and CRW is not at all simple, we cannot assume that classical methods can be directly translated. For example, it is not the case that a quantum walker takes a particular path through a graph (even one of which we are ignorant). This means that quantities such as the CRW edge betweenness that we encountered above cannot be directly used with a QW. Nevertheless, in this chapter we are inspired by the classical approaches in the introduction of our new QW-based centrality measure. We also follow the template of CRW-based community detection algorithms in writing down an algorithm that uses our new centrality measure, albeit that both the centrality measure and the algorithm are new.

Now that we have in mind from the previous chapter some of the tools and concepts that have previously been used in classical methods for analysing the structure of networks, we are in a position to introduce our results. We define a new centrality measure called ‘flashiness’, which is a function of the time-varying probability distribution over the graph associated with a continuous-time quantum walk. This measure can be incorporated into an agglomerative community discovery algorithm that we describe towards the end of this chapter.

In the same way as CRW-based betweenness measures, flashiness reflects the extent of involvement of a vertex in the QW. Importantly, flashiness is a directly observable quantity: its name is inspired by the appearance of a graph where we use photons to realise the walk, and these photons are scattered out of the graph towards detectors on measurement. As will become apparent, vertices in a graph undergoing such a process will appear to flash. This obviates one of the most common restrictions on experimenting with quantum systems: that we do not generally have access to the entire quantum state, but rather just a projection of the state into a particular basis. We also note that we are able in principle practically to simulate any of the walks that we mention in this chapter using a device of the form suggested by Reck et al. [RZBB94] – or perhaps more practically, with the use of a programmable quantum computer.

Having defined flashiness, we examine its properties in small graphs with simple structure so that we can point out its main behaviours. Then, in common with previous community detection approaches, we examine flashiness of a QW on particular artificial
graphs that exhibit community structure put in by hand. The first family of template graphs that we consider are the cycles of glued trees (GTs) introduced in section 3.4. We find that we are able easily to reveal aspects of the structure – the number of constituent GTs – using the QW and the flashiness of the vertices. Due to our choice of graph and following the work of [CFG02], this structure becomes apparent exponentially more quickly than could possibly be the case for a CRW that uses the ‘trapping’ heuristic mentioned above (in, e.g., CRW edge betweenness or any method derived from stability).

Next, we consider blob graphs, where the symmetrical structure emphasises that flashiness can be interpreted as a similarity measure between pairs of vertices. We find that the flashiness for elements of the blob graphs that have no inter-community edges is equal, and different from the flashiness of other vertices. This indicates two points: firstly that, in a similar way as with the GTs, we are able to deduce the number of blob fragments in the blob graph, and secondly that the flashinesses of vertices that have an identical relationship with the rest of the graph are equal.

This kind of symmetry is expressed to a lesser extent in communities in general. That is, for vertices that are in the same community the interaction with the rest of the graph is mediated in the same way. Whereas for the blob graphs vertices within the same blob have exactly the same relationship with vertices in the rest of the graph, for more general graphs – such as the LFR graphs – vertices in the same community have similar relationships to the rest of the graph. We would like for this similarity to be expressed in our centrality measure.

This idea leads us to the main result, where we write down an agglomerative algorithm based on combining vertices with similar values of flashiness. We run the algorithm on examples of LFR graphs, observing that the flashiness of various subsets of vertices separates them from the rest of the graph. These subsets are closely related to the LFR communities in which the QW was initialised, allowing us to iterate the process and amplify the effect. Using classical post-processing of the flashiness we are able to re-discover the seeded communities in LFR graphs.

Finally, we describe a few potential avenues in which flashiness could be used more generally, and summarise the results.
4.1 Quantum flashiness

Suppose that we have experimental access only to the probability distribution of a quantum walker on a graph, and not to the full quantum state. This restriction often makes sense, since many networks where we observe quantum dynamics – for example the LHC II complexes mentioned above – are arranged in space, and our detectors have a particular spatial resolution. This implies that we are able to read a signal corresponding to the probability distribution over vertices (the ‘population signal’), without access to the entire state.

Our motivating question is: is it possible to use the time-varying, vertex-basis signal from a quantum walk to find community structure within the graph?

Suppose that our QW is realised by a photon whose state traverses the graph according to the Hamiltonian

\[ H = D^{-1/2}(A - D)D^{-1/2}. \]

We configure the implementation of the network in such a way that we can switch on an interaction that scatters the photon out of the graph and towards an array of detectors. The resolution of the detectors is such that we are able to detect which vertex the photon was scattered from, effecting a measurement in the vertex basis. By repeating the walk many times and varying the time at which we switch on the interaction, we are able to build up the distribution of the walker in the vertex basis over time. Note, however, that this does not allow us to discover the complete quantum state of the walker at each time. As a simple example, the state

\[ \rho_1 = \sum_i p_i |i\rangle \langle i| \] (4.1)

is indistinguishable in this kind of setup from

\[ \rho_2 = \left( \sum_i \sqrt{p_i} |i\rangle \right) \left( \sum_i \sqrt{p_i} \langle i| \right). \] (4.2)

This restriction represents an important departure from the established classical algorithms, and takes us to the heart of the difference between superposition and classical mixture in a practical setting. Whereas for a classical walker knowing the probability

\[ ^1A \text{ is the adjacency matrix for the graph, } D = \text{diag } (\tilde{d}) \text{ is the diagonal matrix of vertex degrees.} \]
distribution over the graph is equivalent to knowing the state (and, somewhat trivially, knowing the state is equivalent to knowing the distribution), in the quantum case the relation is only one-way. If we know the quantum state then we know the observable distribution, but if we know the distribution there are in general arbitrarily many quantum states that can account for that distribution.

A consequence of this is that whereas in classical cases – for example the edge betweenness of [NG04] – it makes sense to think about the walk in terms of a large population of non-interacting classical walkers, for a quantum walk a single walker cannot be said to travel a particular route through the graph (of which we are ignorant). This means that if we disturb the quantum walker at some point between the beginning of the walk and when we perform the final measurement, we do not generally get the same distribution as our final outcome. This property of a single-walker quantum walk separates schemes of the present sort from other classical processes.

To see this, one could consider the (unbiased) quantum walk on a simple line of vertices connected by edges to their immediate neighbours. We do not go into the example in detail (one could consult [NV00]; the relationship between discrete and continuous time quantum walks is discussed in [Kon05]), but the key observation is that after time $T$, the distance from the starting point for which we have a significant chance of finding the quantum walker is $O(T)$, and for the classical walker is $O(\sqrt{T})$. A somewhat sloppy description of the situation would be that the quantum walker traverses the line in ‘both directions at the same time’, and cannot be said to take a particular path.

For this reason, along with the restriction that we usually only have access to the signal from each vertex, and not the quantum state itself, a practical approach to using a quantum system in order to learn about the characteristics of the graph relies only on the signal from each vertex.

Since measuring the position of the walker – a necessary step in tracking a particular route through the graph – is a non-unitary process, breaking the coherence of the walker, we are also unable to ‘track’ the walker without influencing its state (and consequent evolution). Searching for a direct analogue of a quantity such as edge betweenness for a QW is therefore fruitless. Nevertheless, we would like flashiness to reflect the extent to
which an element of the graph is involved in the quantum dynamics.

The flashiness of a vertex is a positive real number which characterises the range and rate at which the population signal associated with the quantum walker varies for that vertex. In common with centrality measures that we mentioned in the previous chapter – such as betweenness centrality – flashiness reflects the extent to which an element of the graph is involved in dynamics.

Although flashiness is intrinsically associated with vertices, one feature that we have observed is that vertices connected by an edge tend to have similar values of flashiness. This allows us to justify relating flashiness to an edge using a quantity derived from the flashiness of the vertices it connects – for example the average flashiness.

The probability to observe the walker at a particular vertex $v_j$ at time $t$ given a walk initialised at vertex $v_i$ is given by

$$\Pr(v_j, t|v_i) = |\langle v_j|e^{-iHt}|v_i\rangle|^2.$$  \hspace{1cm} (4.3)

So for each vertex of the graph, repeated runs of an experiment yield a time-varying signal. For a typical large graph with a realistic distribution of edges, the signal from each vertex is highly disordered and rapidly variable: containing components over a wide spectrum of frequencies and amplitudes.\(^2\)

This prompts us to look for a quantity that captures the presence or otherwise of the walker at a particular vertex using the rapidly varying signal. More specifically, we would ideally like to be able to measure the extent to which each vertex in the graph is ‘involved’ in the evolution of the walk – the hallmark of a centrality measure. Intuitively, the state vector of the walker can traverse subspaces of the total state space according to the local properties of the graph. But because the state space is richer – the space of superpositions is in some sense larger than classical mixtures – the projection into a basis that we can observe directly varies significantly, even when the state vector is ‘nearby’ the subspace of interest.

\(^2\)For small graphs with definite structure – for example rings of vertices – the signal from each vertex can directly reveal the structure. See [SF06] for example. In larger graphs, for example the LFR graphs introduced above, the number of ‘rings’ – or circular routes – involving a particular vertex is very large, accounting for the wide range of frequencies contained in the signal (one for the commute time around each ring).
FLASHINESS CAPTURES THIS RAPID VARIATION USING ONLY THE OBSERVED SIGNAL.

**Definition 4.1.** The flashiness of a vertex $v_i$ is the sum over time of the absolute difference in population signal between adjacent samples in the limit of continuous sampling, given a walk that is initialised in state $\rho$:

$$D(v_i, T|\rho) := \int_0^T |\Pr(v_i, t|\rho) - \Pr(v_i, t - dt|\rho)|dt$$

(4.4)

It is worth noting in passing that flashiness is well-defined for any process that gives a population signal over the graph, though as we shall see in some cases it is not informative.

We can optionally normalise the flashiness by multiplying by $\frac{1}{T}$. In a practical implementation there is a sampling frequency threshold above which there is no significant difference in the flashiness. Certainly we find that, for the algorithmic uses we describe below, a computer simulation of the walk where we sample $U(t) = \exp(-iHt)$ periodically can easily be well above this threshold.

Our flashiness function sees alternative use in signal processing as a method for detecting noise in images, where it is typically referred to as the ‘total variation’ (TV) of a signal [ROF92, All08]. In the context of noise detection and removal in images, the canonical approach (first introduced by Rudin et al.) is to compare the observed signal (noisy image) to *a priori* assumptions about the source. In particular, several types of noise cause uniform regions of the source signal (for example blocks of colour or other spatial regularities) to show variation. Total variation is a member of a class of regularity measures designed to detect this kind of noise.

When combined with an appropriate cost function, the TV can be used to create a filter that has a particular strength in preserving ‘sharp edges’ in the underlying signal (for example discontinuities in lightness) [BBFAC04]. In that work, the authors use a the TV approach within variety of more sophisticated approaches. Our flashiness approach is therefore naturally seen as a particular case of a broader class of approaches using noise detection measures. That is, in order to establish the extent to which a quantum walker has been ‘in the vicinity’ of a particular vertex, we apply a noise detection method to the
signal from that vertex.

The main alternative quantity that has been used to discover structure in graphs is the average occupancy of a vertex over a long timespan (equation 3.12). As the graphs grow large, interactions between pairs of vertices that are far apart mean that the recurrence time for the population signal at a particular vertex grows very long. Since a meaningful measure of the average occupancy requires that we have a sample of the signal long enough to contain complete recurrences, average occupancy approaches become increasingly problematic for larger graphs. Indeed, interesting features of the flashiness are apparent even at short times relative to the size of the graph, as we shall see in the case of the LFR graphs below.

Before turning our attention to the three more complicated benchmark graphs that we introduced in the previous chapter, we outline our computational methods and examine the behaviour of flashiness for four very simple graphs. Then, in section 4.3, we turn our attention to QWs on the GTs, blob graphs and LFR graphs. Finally we describe how flashiness can be used in an algorithm for community detection.

## 4.2 Flashiness in simple graphs

In order to gain an intuition for how flashiness reflects the structure of a graph, and to solidify our definitions above, we consider flashiness in four graphs. The first two graphs are shown in figure 4.1. For these two, our aim is to show that flashiness can reveal symmetry of a graph with respect to the starting vertex.

The second two examples are the simple ring of vertices (we investigate various sizes of ring), and the related walk on a line. These two examples demonstrate the behaviour of flashiness when the walk takes some time to percolate through the graph.

### The qubit graph

The qubit graph – so-called because the possible states of the walker form a qubit state space – is the simplest possible non-trivial graph. An Hamiltonian for a walk on this
CHAPTER 4. COMMUNITY DETECTION WITH SUPERPOSITION STATES

Figure 4.1: First two simple graphs we use to investigate flashiness.

graph is given by $H = \sigma_x - 1$ (with $\sigma_x$ the Pauli $x$ operator [NC04]). In the Bloch sphere picture, this corresponds to a rotation of the Bloch vector for the walker’s state around the $\sigma_x$ axis as the walk progresses. The projection into the vertex-basis (corresponding to the computational basis of the qubit) therefore varies sinusoidally. We choose to initialise the walker in a state with support entirely on one of the vertices, so the amplitude of the signal is equal to 1. (The initial state is also pure, and since the dynamics are unitary remains pure throughout the walk.)

Since flashiness measures the absolute difference in the signal over time, we observe a monotonically increasing flashiness for the two vertices with time. In addition, since for this graph there are only two vertices that essentially ‘trade’ probability with each other, we observe that the flashiness is equal for the two vertices as we increase the time over which we calculate the flashiness. The graph of flashiness for the two vertices is shown in figure 4.2.

The conclusion is that the symmetry of the qubit graph is reflected in the flashiness. The sinusoidally varying probability to observe the quantum walker at each of the vertices can be observed in the rate at which the flashiness varies, though the flashiness increases approximately linearly for $T \gg 0$ (the limit where the $\frac{1}{T}$ normalisation makes sense). Indeed, we are able to write down an analytic expression for the flashiness in this case up to a scaling factor in time:

$$D(v_0,t) = D(v_1,t) = at - \sin at. \quad (4.5)$$

It is easy to see in this simple example that the average population of each vertex can
Figure 4.2: Flashiness for the qubit graph shown in figure 4.1a. Note that since there are only two vertices, the flashiness is not affected by the starting vertex (the figure contains two coincident lines).

Figure 4.3: Comparison of simple population signals (left plot) with low/high amplitude/frequency with their respective flashiness (right plot). Note that the averages of each of the four signals are equal, whereas there is a clear separation in their flashiness. We use a much longer time scale in the right plot to emphasise how the flashiness diverges linearly with time.

quickly be used to establish a close approximation of the flashiness after a few cycles. Therefore we must motivate the extra utility of flashiness with respect to the average population: we do this with four explicit example signals, each with the same average but different flashiness (illustrated in figure 4.3).

Two vertices whose average population (of the type mentioned in eq. 3.12) is equal can still have divergent flashiness. To see this, consider the flashiness for a pair of vertices where the dominant frequencies of the signal are different, but the amplitude is the same. The increase of flashiness for each period of the signal is equal to twice the amplitude, but the vertex with the higher frequency picks up a correspondingly higher contribution to the
The other alternative – where the frequencies are the same, but the amplitudes are different – can be seen even in the qubit state space. If we initialise the walker in a state with some support in the $\sigma_x$ axis of the Bloch sphere (i.e. $\text{Tr}(\sigma_x \rho) \neq 1/2$), then the projection into the vertex basis still varies sinusoidally with the same mean, but a smaller amplitude. This reduced amplitude also reduces the flashiness. The flashiness is compared to some simple population signals (typical of these types of simple graph) in figure 4.3.

**The simplest non-symmetrical graph**

Next, we turn our attention to the simplest connected graph that exhibits a significant asymmetry; that is, where we are able to see by inspection that the relationship of each vertex to the rest of the graph is not uniform. The graph consists of a three-vertex clique\(^3\) plus a single vertex attached by an edge to one of the clique vertices. The aim is to demonstrate that the differences in flashiness for the four vertices in the graph reflect the symmetry of the graph with respect to the starting vertex. Note that in the case of a three-vertex graph, the edges always have the same ‘context’, and can essentially be swapped without altering their properties significantly (e.g. edge betweenness centrality would be the same).

In the present case, the context of the graph for each of the four vertices gives us (by inspection) a few reasons to expect differing dynamics depending on the starting vertex. Firstly, there are three distinct degrees among the vertices. Secondly, for any pair of non-neighbouring vertices there are multiple possible ‘routes’ from one vertex to the other. Thirdly, this represents the simplest graph where fewer than half of the possible edges are realised in the graph.

We consider the four initial states corresponding to pure preparations with exclusive support on each of the four vertices respectively, and calculate the flashiness for each starting configuration. The results are shown in figure 4.4.

We observe the following phenomena (vertex numbering as in figures 4.1b and 4.4):

\(^3\)A clique is a fully connected component of a graph.
Figure 4.4: Flashiness for four walks beginning at each of the four vertices for the graph shown in figure 4.1b. Note that the flashiness scale is different for each plot, but that the time of the walk is kept the same. Each line corresponds to the flashiness of one of the four vertices in the graph.

- Starting at vertex 0: we are able to distinguish 0 from 1, 1 from (2 and 3), (2 and 3) from 0, but not 2 and 3 from each other.

- Starting at vertex 1: we are able to distinguish 1 from (0 and 2 and 3), but not (0 and 2 and 3) from each other.

- Starting at vertex 2: we are able to distinguish all four vertices.

- Starting at vertex 3: similar to starting at $v_2$.

Taken together, these allow us to conclude a few things about the graph. Most obviously, we can tell that unless we break the symmetry by hand (i.e. in choosing to start the walk at one of vertices 2 or 3), the flashiness of vertices 2 and 3 is always equal. This reflects what we are able to observe directly: that the two vertices that we could swap without
CHAPTER 4. COMMUNITY DETECTION WITH SUPERPOSITION STATES

Figure 4.5: Population signal and flashiness from the graph of 4.1b, with an initial state \( |\psi(t = 0)\rangle = (\exp(i\pi/100)|0\rangle + |1\rangle + |2\rangle + |3\rangle)/2 \). The flashiness and population signal from vertices 2 and 3 are exactly equal.

changing the graph have the same value for our flashiness centrality measure.

More generally, it is evident from the walks presented that the separation in flashiness between pairs of vertices becomes larger as the walk progresses. On the other hand, the time over which we have to measure the flashiness in order to get an appreciable separation varies depending on the initial conditions. For example, vertices 0 and 1 take relatively longer to have distinct flashiness if we start the walk at vertex 0 than they do if we start the walk at vertex 2 or 3.

One other interesting case is where we begin the walk in a superposition with support over more than one of the vertices in the graph. Taking a uniform superposition with no relative phase, the walker remains ‘stationary’; that is, each vertex has a constant probability (equal to 1/4) to observe the walker at that vertex. Similarly, if we add a small phase to one component of the walk (e.g. initialising the walk in a state such as \( |\psi(t = 0)\rangle = (\exp(i\pi/100)|0\rangle + |1\rangle + |2\rangle + |3\rangle)/2 \), the average signal from each vertex is still close to 1/4 – to around one part in 1000. On the other hand, since the amplitude and frequency spectrum for each vertex varies, we are able to observe a significant separation in the flashiness for each of the vertices. The signal from this walk, together with the flashiness, are shown in figure 4.5. From the figure, we are able to observe that adding a small phase to the \( |0\rangle \) component has a similar effect in terms of flashiness to initialising the walk with support only on the \( |0\rangle \) state. We are still able to distinguish between vertices 0, 1 and (2 and 3).
Finally, for this simple graph we note that the same distinctions between the vertices can be made even if we do not start observing the flashiness at the beginning of the walk. One could tell that a walk on this four vertex graph had been initialised at vertex 0 from the fact that there are three distinguishable flashinesses from the four vertices. This would be possible even if we did not have access to signal from the beginning of the walk. The point is significant, since it means that there are cases where once the walk has been active for a significant period of time, we are able to distinguish the initial conditions simply by examining the present dynamics.

For these two examples, it is not clear that the walk takes any appreciable time to reach all of the vertices in the graph. Consequently the behaviour of flashiness near the beginning of the walk is not so clear, and the flashiness grows approximately lineally even at relatively short times. We now turn our attention to two larger graphs where proto-walk phenomena become apparent.

**Cycle graphs**

Our last simple example in introducing flashiness is a pair of related graphs: lines and cycles of vertices. These particular forms are well-studied in the quantum walks literature (a few examples: [NV00, Kon05, Str06, SF06]), with particular emphasis on the time take for the walk to traverse a certain distance through the graph. e.g., as we mentioned in the previous chapter, the QW on a line is an example of polynomial separation in the traversal speed between QWs and CRWs. Similarly, in [SF06] the authors investigate the recurrence phenomena that can be observed in cycle graphs.

Since the QW in lines and cycles takes some appreciable time to traverse the graph, we are able to begin to explore flashiness before the walk has fully ‘mixed’ over the graph, in contrast to our previous examples.

Essentially, the QW on these two types of graphs proceeds as a wave that propagates in both directions around the cycle/along the line. The speed of propagation in both cases is such that the wavefront travels distance $O(T)$ in time $T$. This gives an observable difference in the flashiness for the vertices, and provides a case where the flashiness is not
as obviously approximately linear in time as in the previous two examples, particularly for short times.

Lines of vertices and cycles of vertices of equal size are related: indeed, their adjacency matrices differ by just a single edge. Dynamically, as the walk reaches the end of a line of vertices, it is reflected back towards the direction whence it came. Similarly, the walk on a cycle propagates around the circle effectively travelling in both directions at the same time. This phenomenon is made clear in the figures of [SF06]4. However, as the graphs get bigger the wavepacket corresponding to the walker grows larger, making direct inspection of the probability distribution less revealing.

For flashiness, we shall see that as the wavepacket becomes more spread out the effect is that the initial increase in flashiness is smaller, but that as the packet fully passes the vertex of interest, the flashiness grows to a comparable level.

We examine smaller rings and lines (similar to those examined in [SF06]) as well as larger rings and lines, where the spreading of the wavepacket becomes more pronounced.

For the ring cases, the starting vertex is essentially moot due to the symmetry of the graph. Significant differences in the flashiness do arise, however, between when the graph contains an even and an odd number of vertices. To see this, we plot the flashiness for every vertex on cycle graphs consisting of 20 and 21 vertices in 4.6 and 4.7. (Note that we have ‘zoomed in’ to the initial period of the walk in the probability graphs, where the ring nature of graph is clearly reflected in the observed probabilities, and ‘zoomed out’ to the longer time regime in order to get a clearer picture of the separation in flashiness. The probability graphs are similar to those presented in [SF06].)

The differences that we observe for these two simple examples extend to other ring graphs, and allow us to distinguish odd from even sized rings because of the symmetries (with respect to the starting vertex) reflected in the flashiness of the rest of the graph. For example, in the case of a ring that contains an odd number of vertices, the flashiness for the two vertices that are nearly diametrically opposite the starting vertex have lower

---

4Where, incidentally, the authors introduce an experiment realising the walk with a cycle of connected quantum dots, the walker being realised by an electron in the cycle. Measurement is effected by the affect of the electron on a chemical potential near the QD, allowing the experimenters to tune the interaction.
flashiness than every other vertex in the graph. For an even-sized ring, on the other hand, the opposite vertex (and indeed its neighbours) have higher flashiness than the bulk of the vertices. This effect is apparent in rings of varying size, allowing one to distinguish even- from odd-sized rings simply by examining the relative flashiness of the vertices. We discuss the effect in more detail below.
Figure 4.6: Probabilities (top) and flashiness (bottom) for a 20-vertex ring graph. In the flashiness graph, the top two lines correspond to the starting vertex (dark blue) and the diametrically opposite vertex (light blue). The next two distinguishable lines correspond to the two vertices neighbouring the initial vertex (yellowy-green) and the two vertices neighbouring the diametrically opposite vertex (pink) – the flashinesses are superimposed. The walk starts at the green star, and flashiness for other vertices shown in grey.
Figure 4.7: Probabilities (top) and flashiness (bottom) for a 21-vertex ring graph. In the flashiness graph, the top line corresponds to the starting vertex (dark blue). The next line (yellowy-green) corresponds to the flashiness of the starting vertex’s two neighbours, which have equal flashiness. The bottom line (pink) corresponds to the two vertices that are nearly diametrically opposite the starting vertex, which also have equal flashiness. The walk starts at the green star, and flashiness for other vertices shown in grey.
Figure 4.8: Flashiness for a short time near the beginning of a walk on a 100-vertex line. The walker is initialised at the end of the line (the flashiness for this vertex is the top (blue) line of the graph), each subsequent line of the graph corresponding to the next vertex along the line.

**Line graphs**

For the line cases, we must be slightly more circumspect with respect to the symmetries of the graph: we can choose the initial state to be either at one of the ends, or at the midpoint (if there is one), or at any of the other vertices in the graph. Whereas in the cycle case the wavepacket corresponding to the quantum walker was free to propagate around the ring in both directions (albeit spreading as it went), in the line case the packet is reflected from each of the ends of the line. This means that the time it takes for a component of the wavepacket to return to the initial vertex depends on how far the initial vertex is from the end of the line, a parameter that does not have an equivalent in the cycle case.

Starting from the end vertices of the line, the results are qualitatively similar in both the even and odd graph size cases. The walker propagates along the line, and the flashiness for each vertex begins to grow as the wavepacket passes. This effect can be seen in figure 4.8. Moreover, since the wavepacket is more spread out the flashiness grows at a slower rate for the vertices more distant from the starting vertex.

For an odd-sized line graph, we are able to start the walk in the central vertex of the graph. In this case, we observe that the flashiness grows for vertices that are equidistant from the starting vertex at the same rate, respecting the left-right symmetry of the graph.

The two lessons that we take forward from studying the line and cycle graphs are firstly
that the graph asymmetries are reflected to a large extent in the flashiness, and secondly that the propagation time across the graph is reflected in the flashiness near the beginning of the walk (i.e. when the walk has not fully traversed the graph, but interference effects have had long enough to become evident). This is particularly true for QWs that are relatively short in comparison to the graph size (e.g. times $\leq O(N)$ for an $N$ vertex graph), since, unless we initialise the walk at the central vertex, the wavefunction of the walker does not have enough time to propagate across the entire graph, and so the flashiness cannot be $> 0$.

This area may be a sensible avenue for further investigation, especially with respect to finding communities on ‘branches’ of networks (where, locally, the mesoscopic topology – taking communities as vertices – is that of a line).

Moreover, the behaviour of the flashiness for short walks gives us the first clue that the structure of the graph may be more sharply reflected at a particular time scale with respect to the walk, reminiscent of the stability of partitioning from the previous chapter. Note, however, that for exceptionally short walks the flashiness from a quantum walk is very close to the flashiness derived from signals given by a CRW. This is because interference effects that give rise to the observable qualitative differences between CRW and QW have not had an opportunity to show themselves at such short times.

For this rest of this chapter, we focus on cycles of graph fragments that are more complex. Restricting ourselves to these types of graphs helps to prevent the edge-effect phenomena that we have observed in the line graph cases from becoming conflated with the signs of community presence that we can otherwise observe in the flashiness. We will be particularly interested in using the features that we have observed in the smaller graphs to discover properties of the larger ones.

### 4.3 Flashiness in benchmark graphs

Now that we have demonstrated some of the essential properties of flashiness in simple graphs, we turn our attention to the more complicated benchmark graphs that we reviewed in the previous chapter. We introduce a couple of information processing prob-
lems for the glued tree (GT) and blob graphs, and then examine the use of flashiness as a centrality measure useful in uncovering community structure. We demonstrate that even in larger benchmark graphs the symmetry properties of the graph can be revealed by the flashiness – in the LFR benchmark case, the symmetries that are recovered relate to the community structure of the graph. For the GT graphs and blob graphs, this also allows us to identify the ‘root’ vertices of the graphs as having high flashiness, relative to the rest of the graph.

For the blob graphs, vertices that play a similar role to the root vertices in GTs – those vertices that are joined to an inter-community edge – also have a relatively high flashiness. More significantly, however, the vertices that are joined only to other vertices in the same blob have exactly equal flashiness within a blob. This emphasises that flashiness gives an indication of the context of a vertex within the graph: if two vertices have their interactions with the rest of the graph mediated in a similar way, then their flashinesses will also be similar.

This property can be seen qualitatively in LFR graphs. For these less ordered graphs, the community structure nevertheless reflects groupings of vertices whose context in the rest of the graph is in some sense similar. Our observations from a few specific examples indicate that vertices in the same community in an LFR graph tend to have similar flashinesses.

Using this property, in the next section we present an explicit algorithm for community detection using quantum walks, and test it on a large sample of LFR graphs. We compare the algorithm against the seeded community structure using the normalised mutual information measure, and note that for LFR benchmark graphs we are able to classify a similar number of the vertices correctly as state-of-the-art classical methods, even for relatively high mixing parameters.

**Flashiness in glued tree graphs**

First up, we examine flashiness in the GT graphs that were presented in the previous chapter. As we noted above, GT graphs are one of the first contexts in which it was
demonstrated that there is an exponential separation in the performance of classical random walks and quantum walks for performing a particular information processing task. Namely, a QW prepared initially at a root of a single GT reaches the opposite root exponentially more quickly than a CRW, and indeed any classical process that has a memory and only local access to the network (i.e. has access to the degree of the vertices that it has visited). A similar phenomenon was subsequently found in hypercube graphs [MR02], and other contexts where there is an exponentially large number of possible paths between pairs of vertices.

This separation, combined with our flashiness centrality measure, allows us to phrase another information processing task where there is an exponential separation in the time taken to return: counting the number of GTs in a cycle of GTs.

Though the GT graphs are more complicated than the graphs we considered in the previous section, it is as well to begin simply by examining the probability to observe the QW at each vertex of a particular GT graph. This gives us a sense for the behaviour of the QW in the graph. We begin with an example simulation of a quantum walk on a cycle of glued trees \( -\text{GT}_4 - \text{GT}_2 - \text{GT}_3 - \text{GT}_2 - \) (the \(-\) represent edges between the \(\text{GT}_h\) fragments of height \(h\)). The probability density for observing a QW at each of the 120 vertices of this graph is shown in figure 4.9.

The first thing to note is that the probability to observe the walker at each vertex varies over about 10 orders of magnitude. Secondly, the probability to be at each vertex in the same ‘group’ (according to the simplification we are able to make, described in the previous chapter) is equal for all vertices in that group. Given the simplification argument [CFG02] that we reviewed in the previous chapter, this is unsurprising, but it as well to notice that the argument is supported by the data.

On the face of it, the raw probability information is not particularly useful. Taking the probability distribution over the graph at a particular time is unlikely to reveal much of the graph’s structure, especially if the time-resolution of the detector is not sufficiently fine-grained, or we do not have access to every vertex of the graph. Distinguishing the vertices based on their one-time probabilities would be a practically hard task since it may well involve measuring differences in probability \(O(10^{-10})\). For example, in our
Figure 4.9: Probability to observe a quantum walker at each of the 120 vertices in the graph $-\Delta T_1 - \Delta T_2 - \Delta T_3 - \Delta T_2$ with time. Note the log scale in the probability. Each column of the plot corresponds to an individual vertex. The dotted lines show which vertices are represented in which columns of the plot (the width of a column in the plot corresponding to a single vertex can be seen for vertex 0, on the far left). The walk is initialised at the left root of the largest $\Delta T$, and the arrows represent the periodic boundary conditions. Note that some of the probabilities are so small that they could not practically be measured.

set up where photons are scattered out of the graph at a constant rate per vertex, the probability for a photon to be scattered from a particular vertex in a short window is exceptionally sensitive to the time of the measurement (the signal varies quickly), and requires a photon counter that works over a wide range of incidence rates (the signal varies over a large range).

If we plot the flashiness for each vertex of the graph after a walk long enough for the
Figure 4.10: Flashiness for vertices in the graph $-\text{GT}_4 - \text{GT}_2 - \text{GT}_3 - \text{GT}_2 -$ over the same walk as depicted in figure 4.9. The root vertices are 0, 59, 60, 73, 74, 103, 104 and 119. Here we simulated the walk to time $t = 50$.

Walker to have traversed the entire cycle a few times, the picture becomes much more clear: see figure 4.10. Here, we see sharp peaks corresponding to those vertices that lie at the roots of the GTs. Vertices in the same group again can be seen to have similar values for flashiness after the same time.

This effect can be replicated for GTs of varying height, and when there are many GTs in the cycle. In figure 4.11 we show the flashiness of the simplified GT cycle graph. Here, we divide the flashiness for each of the ‘grouped’ vertices by the number of vertices in the group. This is reasonable, since we see that the probability to observe the walker at each vertex in the group is equal (given our starting conditions), and that the probability associated with the single grouped vertex is the sum over the group of the individual group probabilities. Therefore the grouped flashiness varies as the sum of the individual vertex flashinesses.

These observations prompt us to construct a particular problem that can be solved using flashiness on cycles of GTs: how many GTs are there in a particular cycle?

Problem. Suppose that we are provided a cycle of GTs graph, possibly in the same kind
of black-box set-up as [CFG02]. There is a quantum walker that is initially localised at any root-vertex on the graph. We have access to the probabilities to observe the walker at each vertex in the graph with time – so we are able to calculate the flashiness for each of the vertices. We have some collection of many vertices, the relationships between which we do not know. Our task is to return the number of GTs that the vertices are grouped into using the QW.

Classical algorithms related to the edge betweenness and the stability of partitionings rely on the CRW being able to traverse a significant portion of the graph (necessarily on a scale larger than an individual community) in order to return meaningful results. As demonstrated by [CFG02], in our set-up the traversal time for a single GT is exponential in the height of the tree for the classical walker. For the quantum walker, on the other hand, the traversal time is linear in the height of the trees, so we can expect that if we initialise the quantum walker at the root vertex, it will traverse the graph in a time linear in the sum of the GTs’ heights. This is reflected in the fact that when we group the vertices by column, the resultant grouped Hamiltonian is very similar to the unbiased walk on a line.

As we have seen from figure 4.10, the flashiness for this kind of walk is peaked at those vertices that form the roots of the GTs. Indeed the flashiness of vertices in the same ‘group’ (under the simplification mentioned in the previous chapter) are equal. Taking the flashiness of all vertices, we are therefore able to distinguish the root vertices from others (and indeed we are often able to identify vertices belonging to the same group). Knowing the number of root vertices is equivalent to knowing the number of GTs in the cycle, solving the problem.

Finally, we examined the case where the glued trees are so large that we could not plausibly simulate them in the full vertex basis using a classical computer. Due to the number of vertices in a GT fragment growing exponentially in the height of the fragment, we come across this barrier relatively quickly. Following the work of [CFG02], we simulated the walk on a cycle of GTs of heights 10, 12, 8, 12 and 10 – a graph where the full representation would contain just under 10500 vertices. Since the probabilities for the individual vertices in the same group vary equally, in order to recover the flashiness for individual
vertices we divide the total flashiness for each column by the number of vertices in that
column. This gives us the average flashiness over all vertices in the group.

The results are shown in figure 4.11. As we observed with the simple graphs, the flashi-
ness grows as the walk propagates throughout the graph. We know from the work of
Childs et al. that the QW propagates across each GT in a time linear in the GT’s height.
Consequently, the walk propagates around the ring in a time $O(N/2)$ for a cycle of GTs
graph containing $N$ groups.

**Flashiness in blob graphs**

A ‘blob graph’ consists of groups of fully-connected vertices linked in a cycle (similar to
the GT graphs). A single parameter characterises each of the blobs: the number of vertices
that it contains. Then, to specify the cycle completely, we give the order in which the blobs
are connected. So, in a similar way as we had with the GTs, we can specify a blob graph
as (for example) $-B_4 - B_{10} - B_3 - B_6 - \ldots$ etc..

As we saw in the previous chapter, the blob graph can be simplified to a collection of three-vertex collections with appropriately weighted edges between them. For the classical walk, it is evident by inspection of the weights of the edges in this simplified fragment that the instantaneous probability for a walker to leave the blob is small because there are many more intra-community edges than inter-community edges. Given a classical walker in a particular blob vertex of size $h$, the probability for the walker to leave the vertex in the next instant is proportional to

$$\Pr(\text{leave}) \propto \frac{2}{h} \left( 1 - \frac{1}{h} \right), \quad (4.6)$$

the product of probability to be at an ‘out’ vertex,$^5$ and to choose the inter-community edge in the next step. (The constant of proportionality is determined by the laziness of the classical walk – essentially the hopping rate.)

For the QW the elements of the grouped Hamiltonian are symmetrical in each direction. To see this, consider the superposition state over the in vertices of a particular blob of size $h$ (analogously to the GT case, we call the collection of vertices a group):

$$|\text{grp}_h\rangle = \frac{1}{\sqrt{h-2}} \sum_{i=1}^{h-2} |i\rangle. \quad (4.7)$$

The sum runs from 1 to $h-2$ since we choose to label the out vertices as 0 and $h$. Then the modified Hamiltonian elements are given by:

$$\langle \text{grp}_h | H_{\text{grpd}} | \text{grp}_h \rangle = \frac{1}{h-2} \left( \sum_{i=1}^{h-1} \langle i | \right) H \left( \sum_{j=1}^{h-1} | j \rangle \right) \quad (4.8)$$

$$= h - 2$$

$$\langle 0 | H_{\text{grpd}} | \text{grp}_h \rangle = \frac{1}{\sqrt{h-2}} (0 | H \left( \sum_{j=1}^{h-1} | j \rangle \right)$$

$$= \frac{1}{\sqrt{h-2}} h - 2 = \sqrt{h-2} \quad (4.9)$$

$^5$Recall that an out vertex is one with a neighbour in a different blob, as opposed to an ‘in’ vertex that has neighbours only within the same blob.
with similar elements for the other out vertex \( \langle h | H_{\text{grp}} | h \rangle \) elements). Since we removed the edge between the two exit vertices that are within every blob, \( \langle 0 | H | h \rangle = \langle 0 | H_{\text{grp}} | h \rangle = 0 \).

The symmetries of the graph that allow us to make this simplification are expressed in the population signals that we can measure for a simple example of the graph. The results of a simulation giving the population signals are shown in figure 4.12. (Note that here we simulated the walk in the full vertex basis, without the simplification.)

As we found in the GT case, direct inspection of the signals at a particular time is not especially useful because of the range and rate at which the signal varies. If instead we plot the flashiness each vertex of the walk over the total time period shown in figure 4.12, we obtain results shown in figure 4.13. Similar results are obtained from a range of different graph and blob sizes.

We can immediately make a few observations:

- The flashiness of the out vertices is higher than the neighbouring in vertices – i.e. the vertices in the same blob.

- The flashiness for any pair of in vertices in the same group is equal. This reflects the fact that the vertices have the same context with respect to the rest of the graph given the initial state.

- For small blobs, the flashiness of the in vertices is relatively high.

Using these observations, we are able to solve a similar problem to that we posed for the GT graphs: how many blobs are contained in a particular cycle. But more importantly, in this class of graphs we have clear evidence that the context of a vertex with respect to the rest of the graph is captured by the (equality of) flashiness. This feature is the one that we use in the next section as we extend our results to the more realistic cases of LFR graphs, where the communities can be thought of as less strongly symmetric versions of the blobs described here.
Quantum transport and symmetry in our benchmark graphs

Before we turn our attention to the LFR graphs that form the most realistic test of our approach, it is worth noting (as pointed out in examining this thesis by Viv Kendon) that several of the graphs we have considered so far have special properties. Namely they support high fidelity or perfect state transfer – a special kind of quantum transport between certain pairs of some graphs’ vertices. A survey of perfect state transfer in a variety of graphs can be found in [KT11].
Figure 4.13: Flashiness for the vertices of a blob graph after time $t = 100$ containing seven blobs of size 4, 10, 5, 8, 12, 9 and 4. The walk is initialised at vertex $v_0$. The peaks in the flashiness correspond to the out vertices of each blob – i.e. those with inter-blob edges.

As we noted in the previous chapter, dynamical approaches to community detection in general rely on the way that the walker (quantum or otherwise) behaves in the presence of mesoscopic features of the graph. In light of this, it is important to note that in some cases particular fragments of a graph support transport of the walker from one vertex to another, either perfectly or with high fidelity. For the purposes of the algorithm that we describe below, this gives somewhat spurious results, with some vertices having anomalously high flashiness. This high flashiness is accounted for by the full (or close to full) support of the walker on a particular vertex, which in turn gives a peak in the population signal, counted by the flashiness function.

Perfect state transfer occurs between vertices $a$ and $b$ in a graph with associated continuous time quantum walk Hamiltonian $H$ if there exists a time $t$ such that

$$F_H(a, b) = \max_{t > 0} |\langle a | e^{-iHt} | b \rangle| = 1. \quad (4.11)$$

Similarly, the transport occurs with high fidelity if this quantity is close to 1. Phenomenologically, this case corresponds to that where one can expect a packet of energy or charge
(for example) to traverse from one point in a network to another with near certainty.

The transport of the walker’s state from one part of a graph to another is an important technology from the point of view of communicating states between registers in a quantum computer. Early work in this area was carried out by Bose [Bos03]. There, the aim was to find a set of local interactions between spins in order to create a short-distance communication channel. Crucially, in Bose’s scheme, no control is necessary over the intermediate parts of the channel. Indeed, the phenomenon also occurs naturally with high fidelity for energy transport in photosynthetic contexts [CLMS09, TPdVH08].

Thus in analysing this work we should be aware of those cases where we might expect artefacts of high-fidelity quantum transport to be apparent. In our work, several of the constructs that we have used support either perfect or high-fidelity state transport. In some cases, this is evident from the behaviour of the flashiness. Here we mention each of those contexts in turn, and comment on how the perfect state transfer phenomenon is expressed in our flashiness function.

The two vertex, one edge graph that we examined first clearly demonstrates perfect state transfer in both directions between the two vertices. This follows immediately from the observation that the time-varying probability to observe the quantum walker at each vertex covers the range \([0, 1]\) – which is only possible for a pure-state walker that periodically has support only on one vertex, or the other. Indeed, since the dynamics involve periodic perfect transport in each direction between the two vertices, the flashiness for both vertices at all times is equal. This observation is important in the context of the cyclical graphs that we construct later, however, since these are constructed with a single edge joining the more complicated fragments (glued trees, blobs).

Cycles support high fidelity state transport under certain circumstances. For example, in the case of the 20-vertex cycle (figure 4.6), we have

\[
F_H(v_0, v_{10}) \approx \max_{t \in [0,200]} |\langle v_0|e^{-iHt}|v_{10}\rangle| = 0.89. \tag{4.12}
\]

(The time range \([0,200]\) is long with respect to the characteristic time of the cycle.) For the 21-vertex graph, the largest value of \(F_H(0,b)\) over the same time range is only \(\approx\)
Although this fidelity decreases as the cycle gets larger, we observe that for even-sized cycles there is relatively high fidelity transport between the starting vertex and the diametrically opposite vertex. This can be seen in the gradients for the flashiness (vs time) for the corresponding opposite and nearly opposite vertices in figures 4.6 and 4.7 respectively.

Glued trees also support high-fidelity transport; in this case between the root vertices of the fragments. In the original work of Childs et al. [CFG02], the comparison was made between the transport properties of the GT graphs and the hitting time for a classical random walker: the hitting time is the closest classical analogue for high fidelity quantum transport.

Finally, the blob graphs that we used above also support perfect transfer between the entry and exit vertices of the fragments. This is a result due to Bose et al. in the context of examining the transport properties of complete graphs [BCMS09]. In that work, the authors demonstrate that although complete graphs (i.e. graphs with an edge between any pair of vertices) do not support perfect state transfer, simply removing an edge between a pair of vertices \(a, b\) allows the graph to support perfect state transfer between those two vertices.

High fidelity transport in the example graphs that we have used so far is expressed in the flashiness. For those pairs of vertices where the intermediate portion of the graph supports high fidelity transport (e.g. entry/exit of a single blob, roots of GTs, initial and opposite vertices of even-sized cycles) we observe similarly high flashiness for those pairs. In figure 4.6 for example, one can see in the flashiness graph the steeper portions of the plot at times where there are local maxima in the fidelity (eq. 4.11).

This same effect explains as well the high flashiness for the root vertices in the GT graphs, and also the entry/exit vertices in the Blob graphs. Since not only the fragments but also the links (single edges) that make the fragments into a cycle support high-fidelity transport it is not surprising that the flashiness for those vertices involved is ‘lifted’ relative to other vertices in the graph. Had the walk been initialised at vertices that were not involved in high fidelity transport pairs, this effect would not have been as pronounced.
CHAPTER 4. COMMUNITY DETECTION WITH SUPERPOSITION STATES

For our final and main category of benchmark graphs – the LFR graphs – any pairs of vertices for which high fidelity transport does exist occur by coincidence: we do not expect these phenomena to be widely present in LFR graphs. Local effects of high-fidelity transport, especially out of the community in which the walker is initialised, can have observable effects. Indeed, in those iterations of our algorithm where the ‘success’ criterion is not met for assigning a particular vertex to a community, this is one of the phenomena that could explain the failure.

Flashiness in LFR benchmark graphs

As we mentioned above, LFR graphs are designed to mimic the characteristics of real systems, particularly with respect to the distribution of vertex degrees and the variety in the sizes of communities. Their close resemblance to instances of real networks makes them an appealing benchmark, not least because they represent a source of large graphs ‘on tap’ to act as subjects for community detection algorithms. In common with other benchmarks, the algorithm for generating LFR graphs involves allocating every vertex to exactly one community, so there is in principle a notion of the ‘correct’ clustering to which the partitioning returned by an algorithm can be compared. This is used in the normalised mutual information measure as we evaluate our results.

In addition, a mixing parameter $\mu$ allows us to fix the proportion of edges that are inter-community relative to intra-community. This means that we can vary how challenging the graph is for the algorithm: for $\mu = 0$ there are no inter-community edges, and for $\mu = 0.5$ every edge is equally likely to be between communities as within them. In this case any remaining community structure arises by chance.

For low values of $\mu$, our earlier work indicates that care is required in interpreting the results. The reason for this is based on our expected behaviour of flashiness as a reflection of the centrality of a vertex with respect to the rest of the graph. If $\mu$ is particularly low, then random fluctuations in the edges that are realised between communities can introduce non-trivial structure that drastically alters the dynamics. To see this, consider three clusters of vertices $C_A, C_B, C_C$. Where the mixing parameter is very low, relationship of cluster $C_A$ to $C_B$ may be significantly different to its relationship with cluster $C_C$. This is
Figure 4.14: Population signals from a QW on the LFR graph (shown as an example in the previous chapter: figure 3.6), with a walk initially localised at vertex 0, marked with a pink star. Each column represents one of the 400 vertices.

because the addition or removal of a single inter-community edge for low values of $\mu$ could have a significant impact on the structure of the graph in a way that particularly affects dynamical approaches. (If, for example, that edge was the only one connecting $C_A$ to $C_C$, so that the interaction between those two is now mediated exclusively by $C_B$.)

Given our groundwork, we may well expect that this asymmetry is reflected in the flashiness of vertices in $C_A$ with respect to the flashiness of vertices in $C_B$ and $C_C$. This phenomenon may be useful when studying that particular graph, but where we wish to evaluate the efficacy of an algorithm these edge-cases can give a false indication of how well an algorithm performs in general.

As with the graphs in the previous subsection, we initially choose a particular LFR graph and simply present the result of simulation of a QW, giving the population signal for each vertex. We use the example graph that appears in figure 3.6; the probability signals from each vertex are shown in figure 4.14.

As with previous cases, the plot of the raw signal is generally uninformative, and shows
even less structure when taken as a whole than the blob or GT signals. We surmise that
the reason for this is that the number and variety of pair-wise interactions between the
vertices of an LFR graph are reflected in the signal. That is, for each pair there are many
‘paths’ along which components of an interaction can propagate, and indeed for a 400-
vertex graph each vertex has a large number of these components contributing to its over-
all amplitude. The observable consequence is that the probability to observe the walker
at a particular vertex varies rapidly and – if we do not have access to the structure of the
graph – unpredictably. Further, since the mean number of edges between any pair of ver-
tices in the graph is small relative to the time that we typically allow the walk to progress
for, the walker is relatively well mixed throughout the graph in the simulations that we
perform.

When we apply our flashiness measure to the LFR probabilities, a pattern begins to
emerge that reflects the qualities we found for the simple graphs in the previous sec-
tion. Namely, vertices that are in the same seeded community of an LFR graph tend to
have distinct flashiness from vertices not in the seeded community, and similar flashiness
to each other.

When we initialise the walk at a vertex in a particular community, the flashiness of ver-
tices in that community are ‘lifted’ relative to the other vertices in the graph. This reflects
that the state vector of the walk is more active in the subspace spanned by the vertices
of the community than in the complementary space spanned by the other vertices in the
graph – an effect akin to the low inter-community transport that motivates the definition
of partition stability.

We are able to amplify this effect by adding the flashiness from walks proceeding from
different initial states. Suppose that we measure the flashiness for a particular walk start-
ing at vertex \( v_i \) and find that vertex \( v_j \) has a similar flashiness: \( D(v_i, T|v_i) \sim D(v_j, T|v_i) \)
(where we condition on the starting vertex). We can amplify the flashiness by adding the
result from the first walk to the result from a second walk, this time initialised at vertex
\( v_j \), giving \( D_2(v_j, T) = D(v_j, T|v_i) + D(v_j, T|v_j) \), where we use the subscript to denote the
number of walks with different initial conditions that we sum over. So for example the
flashiness following one iteration is \( D_1 \), then, once we add this to the flashiness following
a second iteration, we obtain $D_2$, adding to flashiness from a third iteration is $D_3$ etc..

We find that iterating this process gradually separates all of the vertices within a community from the complement in the rest of the graph (with respect to the flashiness). For this particular graph (the middle example from 3.6), we have a reasonably typical mixing parameter of $\mu = 0.128$ (recall the mixing parameter is the proportion of inter-community edges).

One can see that the flashiness reveals the community of a particular vertex according to the other vertices with similar flashiness. This is the quality that we use below as we write down an explicit algorithm using the flashiness for community detection.

While we have given this example for a particular LFR graph, the effect is consistently repeatable for different LFR graphs realised with similar parameters; we are able to ‘pick out’ vertices belonging to a particular LFR community by summing the flashinesses over iterations of starting vertices in that community. Indeed, within the same graph we are able to choose which community to pick out given an example vertex from each of the communities, simply by choosing to start the QW in the first iteration from that vertex.

We have found that the effect is observable in a range of LFR graphs with typical parameters. Since the range of parameters allows an exceptionally wide range of graphs (including disconnected graphs, graphs with a single vertex per ‘community’ etc.), we restricted our study to those graphs with parameters in a standard range – for example those used in the work where the benchmark was introduced – save for the mixing parameter which we vary. This choice is natural since the mixing parameter determines the extent to which the seeded community structure is expressed in the graph, and is recognised as a parametrisation of how ‘difficult’ a challenge a particular graph represents to an algorithm.

On the other hand, there are clearly a wide range of other graphs that could be studied with structures that perhaps represent particular types of network found in nature. In a similar way, we propose that the flashiness centrality measure could find applications in different types of graph-based problem, including, once suitably normalised, as a substitute for the autocorrelation function used in classical stability of partitions. That is, the
time over which two vertices have similar flashiness would provide an indication of the
time scale over which it is appropriate for them to be clustered together.

We leave these investigations as the subject of further work, and turn our attention now
to a proposal for an explicit community detection algorithm. Our aim is to make the
role of flashiness as clear as possible, and as such we avoid (possibly complicated) post-
processing of the signal that could be used to optimise our approach.

4.4 Algorithm using flashiness for community detection

We have seen that vertices that are contained in structural elements of particular LFR
graphs tend to have similar flashinesses that are different from those for other vertices
in the graph. In this section we present a simple algorithm that can be used to return an
explicit partitioning, based on this effect.

The algorithm is agglomerative on the scale of a community: rather than joining pairs
of vertices it joins entire communities with each iteration. The key addition is use of
a statistical technique to separate vertices into two groups: those that are in the same
structural element (i.e. community) as the starting vertex, and the rest of the vertices in
the graph. This task is essentially equivalent to finding the ‘gap’ between the groups of
vertices in figure 4.15. We use the same trick of summing the flashiness from multiple
iterations with different starting vertices in the algorithm to increase the gap.

Before introducing the technique that we use to divide the vertices into two groups (those
in the same community as the starting vertex/those in the rest of the graph), we de-
scribe explicitly the iterative process that generates the flashiness separation between ‘in
community’ and ‘not in community’ categories. Call these two sets of vertices $I$ and $N$
respectively. We call this process SEPARATE. Initially every vertex is ‘not in community’:

1. Pick an initial starting vertex.\footnote{We use the convention that the ‘starting vertex’ is the vertex that a particular walk starts from, and the
‘initial vertex’ refers only to the fact that the vertex is the first to appear explicitly in the algorithm. Thus the
‘initial starting vertex’ is the first vertex the algorithm picks from which to start a QW.} This choice dictates the community for which we
will find the separation.
2. Assign the starting vertex to $\mathcal{I}$.

3. Initiate a QW with the walker initially having support only on the starting vertex.

4. Record the flashiness for each vertex in the graph. The vertex with the highest flashiness that is in $\mathcal{N}$ becomes the starting vertex for the next iteration.

5. For each vertex added to $\mathcal{I}$, add the flashiness for this iteration to an ongoing tally $D_{|\mathcal{I}|}$.

6. Repeat from 2 until a halting condition is reached. (We discuss the halting condition below.)

The result of this loop for 80 iterations is shown in figure 4.15, with snapshots of the ongoing tally of flashiness at each 10 iterations. Further, the effect is similar as we vary the initial starting vertex, or as we investigate different graphs.

With some relatively simple (classical) post-processing to establish the separation, we are able to use this effect as the cornerstone of an algorithm designed to return the community structure. Although we present here a single explicit example, the flashiness can be used as a general centrality measure, and therefore could find use in a number of different algorithms perhaps designed with different purposes in mind.

In order to complete our description of SEPARATE, we need to find a halting condition that determines when the entire community has been found. We describe one method for deciding when a community is full, but there are doubtless other possible approaches. The hallmark of a complete community is that there are two distinct groups of flashiness with a clear gap between them. We cannot, however, simply look for the largest gap in flashiness since there are occasionally a few outlying vertices (from both $\mathcal{I}$ and ‘not in’ groups) that are relatively isolated in the gap, making them hard to classify.

Instead, we examined the variance of the flashinesses for $\mathcal{I}$ vertices (the top group of flashinesses) after some number of iterations. As we add more and more vertices to this group, we plot the variance of the $\mathcal{I}$ group. The variance increases gradually, until we begin to add vertices from the ‘wrong’ side of the gap, when we observe that it increases
Figure 4.15: Snapshots of the iterative process described in the text. For each iteration, we pick a new starting vertex for the walk and add the resulting flashiness (after $t = 100$) to an ongoing tally. The starting vertex is the flashiest vertex yet to be assigned to the community. The colours of the points indicate the seed community of the LFR graph from which the first starting vertex was picked: blue for $I$, red for $N$. 
sharply. The results for the full set of initial starting vertices are shown in figure 4.16. In almost all choices of initial starting vertex we see a sharp increase in the spread of flashinesses for the vertices in I after the same number of iterations, corresponding to the community size. We use this sharp increase as the stopping condition for SEPARATE. Adding a slight complication, we can improve the results returned by the algorithm by requiring that the rate of increase in the variance as we add more vertices exceeds a certain amount; if the requirement is not met, we pick a different initial starting vertex. This method is implemented by adding a cut-off in the rate of increase of the variance, and filters out those starting vertices for which the sharp increase is less well-defined (i.e. adding a condition under which a particular iteration can ‘fail’).

In our implementation of the algorithm we forbid communities to overlap, so that each vertex is placed in the first community for which it falls on the ‘in community’ side of the flashiness gap. Occasionally, however, a vertex is not able to be categorised because it cannot be associated with a separation large enough to meet the cut-off described above. For the purpose of what follows, at the end of the algorithm we assign the vertex to a community at random; this is clearly an aspect of the approach that could be improved with more sophisticated classical post-processing, possibly taking into account the presence of overlapping communities.

Our algorithm proceeds as follows, where we use SEPARATE as a subroutine. In the beginning no vertex is assigned to a community, and the index \( i = 0 \):

1. Choose any vertex not assigned to a community as the initial starting vertex.

2. Run SEPARATE with that choice of initial starting vertex, and allocate all vertices returned in I to community \( C_i \).

3. Increment \( i \) and repeat from 1. until all vertices are assigned to a community.

4. Return the partitioning \( \{ C_i \} \).

So far, for the sake of relative simplicity, we have examined only one example of an LFR graph with a mixing parameter of \( \mu = 0.128 \). In order to evaluate our algorithm in gen-
In general, we turn to an ensemble of graphs with mixing parameters ranging from $\mu = 0.01$ up to $\mu = 0.5$.

In common with other work evaluating the performance of algorithms, we use the normalised mutual information score to measure how closely the partitioning returned by our algorithm reflects the seeded partitioning of the graph. In particular, we are interested in measuring the performance of our algorithm with respect to the mixing parameter of the benchmark graph. The results are shown in figure 4.17.

To a large extent, the success of the algorithm depends on the method that we use to detect a sharp rise in the variance of the flashiness for the ‘lifted’ group of vertices in each iteration. This aspect of the algorithm is an obvious target for optimisation, though has no bearing on the utility of flashiness per se.

In comparison to classical community detection algorithms, our present approach returns results of comparable success. We expect that our approach to detecting the gap in flashiness – not a quantum part of the algorithm – could be improved, and that consequently the returned partitioning might more closely match the seeded structure.
Figure 4.17: Normalised mutual information between the returned partition following our algorithm (coloured points) and the seeded partitioning in the LFR graph vs. the mixing parameter $\mu$ for that graph, for 108 benchmark graphs. The colour of the marker reflects the proportion of unallocated vertices in the partitioning (blue is more successful). The grey crosses result from the stability-based classical algorithm of the ‘Louvain group’ in [BGLL08], which we include for comparison.

**Other possible uses of flashiness and extensions**

Centrality measures based on classical random walks find a multitude of uses as we discussed in the previous chapter. With respect to community detection, this suggests that flashiness may find use in algorithms and measures other than that described above.

Firstly, since flashiness depends only on the evolving probability distribution associated with a particular dynamical process on the graph, it is in fact well defined for classical random walks. Typically, the probability to find the walker at a particular vertex in CRW varies slowly and mainly towards the stationary value with time. This means that the flashiness is (in the case of a walk where the initial probability is set to 0 for a particular vertex) either equal or very close\(^7\) to the corresponding entry in the stationary distribution over the graph – a good example can be seen in the qubit graph above. There, it is immediate to see that the classical flashiness for both vertices is $= 1/2$, from the station-

\(^7\)Occasionally features of a graph will cause the probability to find the walker at a particular vertex to ‘overshoot’ the stationary distribution value, though it is very rare to find more than a few slow oscillations except in directed graphs.
ary distribution of the classical walk. As a result, flashiness is typically of no more use than the stationary distribution for CRWs.

A more interesting case is where we have a quantum evolution similar to that described above, but where a noisy interaction with the walker causes the walker’s state to become mixed. Typically such an interaction will cause the coherence associated with the superposition state to be reduced, for example by effecting occasional measurements on the walker in the vertex basis. If we stipulate that the outcome of these measurements remains hidden, then at each measurement that walker’s state collapses into a classical mixture. This state then continues to evolve unitarily until the next interaction. For this kind of dynamics, flashiness is also well-defined. Indeed, the characteristic time that the noise takes to bring the walker into a steady distribution (at which point the flashiness ceases to change) could give us a ‘natural’ partitioning for the graph.

We considered walks that begin in a pure state at a single vertex. This approach was necessary since without an initial condition that favours a particular vertex, we observed that the population signal does not vary. As an alternative to mixing the initial state (or indeed introducing mixing through noise during the walk), one could consider initialising the walk in a coherent superposition over the vertices of the graph, and adding relative phases between the components at each vertex. Our early experiments on small graphs with this kind of initial condition suggested that the utility of flashiness with respect to average occupancy may be further enhanced with this kind of initial condition. Since the space of such initial conditions is clearly very large, a systematic approach should be taken in establishing which phases give the best advantage. Nevertheless, the impact of a small phase shift on the flashiness is evident from figure 4.5, and suggests such small shifts as a further dimension in which one could introduce asymmetry in the initial state of the walker.

Although we were not explicitly aiming for a particularly efficient community detection algorithm (our aim was to show that a quantum community detection algorithm was obtainable in principle), it is worth exploring the resource required in executing our approach. In the examples that we provided, we ran the SEPARATE subroutine for every vertex of the graph for each iteration. This was probably not strictly necessary, since the
separation between the flashiness of the in-community vertices is usually apparent after about $|C_i|$ walks, for the $i$th community. That is, for each community we have to run the quantum walk with different initial conditions a similar number of times as there are vertices in the community, possibly with constant overhead. Assuming that the walk can be completed in linear time with the size of the graph, and given that the algorithm contains $O(N)$ iterations in order to return a partitioning for an $N$ vertex graph, the entire algorithm can complete in $O(N^2)$ time.

Flashiness is not the only closeness measure that can be applied to quantum walks. Before summing up this chapter, we briefly review the main alternative approaches to community detection that use continuous time quantum walks.

### 4.5 Another pair of approaches to using quantum walks for community detection

We briefly review two other approaches that using quantum walks in community detection, both of which use a closeness measure in order to agglomerate vertices into the same community – a similar approach to the one that we describe here. The approaches are due to Faccin et al. [FMJ+14], and the key feature of both approaches is a closeness measure between partitions that reflects the quantum dynamics.

In the first case, the closeness measure depends on the inter-community transport between pairs of partitions. This is similar to the edge betweenness used classically in [NG04], except that for the reasons we mentioned above it does not make sense to think of the quantum walker taking a particular path through the network. Instead, Faccin et al. consider simply the flow of probability from one partition to another part of the graph, regardless of the ‘path’ that it took. The key object is a doubly stochastic transfer matrix whose elements refer to the probability of transport out of a cluster in time $t$. For individual vertices, such a matrix is given by $R_{ij}(t) = |\langle i | e^{-iHt} | j \rangle|^2$; for the full transfer matrix, Faccin et al. sum over the the vertices of the partition. For a particular subset of the graph
A the transfer matrix is given by:

\[
T_A(t) = \sum_{i \in A, j \notin A} \frac{R_{ij}(t) + R_{ji}(t)}{2}.
\] (4.13)

Next, this transfer matrix is used to define a closeness measure dependent on the inter-community transport given by

\[
c_{t}^{(1)}(A, B) = \frac{T_A(t) + T_B(t) - T_{A\cup B}(t)}{|A||B|}
\] (4.14)

where the numerator gives the reduction in the transfer matrix for the whole graph that occurs when the clusters \(A\) and \(B\) are merged. An algorithm that reduces inter-community flow with each agglomeration of vertices can therefore use this measure as a relevant figure of merit.

As Faccin et al. point out, however, this measure can vary as the projection of the walker’s state into each cluster oscillates. (This oscillation is simply a coarse graining of the individual oscillation we saw earlier.) To obviate this effect, the authors take a time-averaging approach where a new transport probability is given by

\[
\hat{R}_{ij}(t) = \frac{1}{t} \int_{0}^{t} R_{ij}(\tau) d\tau.
\] (4.15)

They proceed to argue that for short \(t\) the process is essentially classical (there is no interference between different components of the walker’s state), and in the limit of long \(t^8\) the components of \(R\) are related to the spectrum of the Hamiltonian. Thus the long-time average approach is a quasi-spectral method, akin to the non-dynamical methods we mentioned in passing in the previous chapter.

The second approach of Faccin et al. also involves a closeness measure, but this time considers the fidelity measured between two states of the walker at different times in the walk. Namely, the authors consider the squared fidelity between the state in which the

\^8Where \(t\) is greater than the inverse of the smallest eigenvalue gap in the Hamiltonian.
The walker was initially prepared (they use a uniform superposition) and the evolved state:

$$c_i^{(2)}(A, B) = \frac{F_{A \cup B}(t) - F_A(t) - F_B(t)}{|A||B|}$$

(4.16)

where

$$F_A(t) = \left( \text{Tr} \left[ \sqrt{\rho(0)^{1/2}\rho(t)\rho(0)^{1/2}} \right] \right).$$

(4.17)

Faccin et al. go on to show that these two methods can reveal different community structure than is conventionally used for a few different quantum complex networks, and also investigate the effect of relative phases between components of the walker’s initial state (so-called ‘chirality’ of a quantum walk [FJB+13]).

For our part, we conjecture that flashiness can also be used as a closeness measure in a similar type of algorithm to those suggested by Faccin et al., as well as the simple algorithm we described above. For example, one could define a flashiness over a collection of vertices $A = \{i\}$ (as we did for the grouped GTs)

$$D_A(t) = \sum_{i \in A} D_i(t).$$

(4.18)

A natural next step would be to compare the flashiness for a pair of clusters, normalising by the flashiness associated with both:

$$c_i^{(3)}(A, B) = \frac{|D_A(t) - D_B(t)|}{D_A(t) + D_B(t)}.$$

(4.19)

We leave further exploration of this and other avenues that make use of our centrality measure for future work.

### 4.6 Simulations

In this brief section we outline the main computational methods that we used in order to simulate the walks described above. Several useful snippets for the functions described here can be found in Appendix A.
Since in almost all cases the graphs that we studied are very large (except for the simple graphs of section 4.2), we must use a computer simulation to provide the probability ‘signal’ from each vertex as the walk progresses. The principal task is calculating the unitary matrix for the particular graph at each time \( t \); that is, the operator taking the initial state of the walker to the state after \( t \). Although the walk is continuous, our simulation outputs a list of probability distributions at a large number of times during the walk – typically 10000 ‘steps’. It is therefore important to choose carefully both the total time for which we simulate the walk, as well as the distribution of samples over that interval. We used a uniform distribution in the results above, though often when carrying out initial investigations of characteristic times for a new graph a different distribution made better sense. In these cases, the regularity of the vertices’ degrees, the connectedness of the graph and its size are among the main concerns in deciding the sampling rate threshold below which important parts of the probability signal are missed.

For all of our simulations we used the Python programming language \([\text{Pyt}]\), along with several components of SciPy: NumPy, IPython and Matplotlib \([\text{JOP}^+]\). In particular, our calculations of the appropriate unitary for each graph and time were carried out using SciPy’s linear algebra module.

The simulations were run on Google’s Compute Engine, where the user is able to choose the specifications of the instance running the simulation in proprietary ‘Google Compute Engine Units’, where ‘2.75 GCEUs represents the minimum power of one logical core (a hardware hyper-thread) on [Google’s] Sandy Bridge or Ivy Bridge platform’ \([\text{Goo}]\). As with local simulations, development work was carried out using IPython notebooks tunnelled through SSH, and data (including graphs to simulate and results) synchronised using Dropbox. The RAM for the instance carrying out the simulation was chosen according to the size of graph being simulated in order that paging was avoided. (At the time of writing, up to 104 GB is available for GCE.)

In order to carry out simulations in a timely fashion, the interval over which we wished to simulate the walk was scattered among the logical processors of the instance being used (typically 16 GCE cores; 44 GCEU), with results gathered as the simulation function
returned. This process was easily effected using IPython.parallel.\(^9\).

Although the state of the walker evolves deterministically for a particular graph, in those cases where we wished to study the general properties of a class of distinct graphs – for example the class of LFR graphs with particular parameters – we carried out multiple simulations. For our LFR analysis our sample size was 108 graphs with a variety of mixing parameters, as described above. On average, the simulation of the walk over 10000 timesteps for a 400-vertex graph took 847 seconds.

### 4.7 Summary

In this chapter, we introduced flashiness: a centrality measure directly related to the observable probability to observe a quantum walker at each vertex.

Initially we calculated the flashiness for a collection of four simple graphs: the qubit graph, the simplest asymmetrical graph, lines and cycles. This allowed us to observe that flashiness can be used to distinguish vertices from each other where there is an asymmetry between vertices with respect to the starting vertex in the walk. This was the property that we were able to use in order to write down an explicit community detection algorithm.

Prior to writing down the explicit algorithm, we examined a couple of information processing problems related to more artificial benchmark graphs: cycles of glued trees and cycles of fully connected fragments (blob graphs). We were able to recover the number of GT fragments in a cycle of GTs exponentially more quickly than any classical algorithm that is based on the trapping heuristic that has recently been of interest in the stability of graph partitionings. Examining blob graphs gave us a good indication that flashiness can be thought of as a (dis-)similarity measure between pairs of vertices with respect to their contexts in the graph.

In the main result, we used flashiness for finding community structure in LFR graphs – a class of benchmark graphs designed more closely to reflect the features of real networks.

\(^9\text{See http://ipython.org/ipython-doc/dev/parallel/}.\)
In particular, LFR graphs are considered a particular challenge for detection algorithms. We were able to write down a relatively successful algorithm (as measured by the normalised mutual information measure, relative to a well-regarded classical approach), but using exclusively quantum dynamics and directly observable qualities of the quantum walk (measurement in the vertex basis).

Finally, we noted that as a general centrality measure, flashiness can potentially be used as a component of other algorithms or investigations into the structure of a network. We suggested a way in which flashiness could be related to the two other quantum community detection algorithms that exist in the literature. Given the wide range of possible contexts, applications and motivations for studying complex networks where quantum dynamics are in play, it is inevitable that there is a large amount of further work that can be done in this area; far more than could fit in the present work.
Final thoughts and summary of contribution

In this thesis we have investigated two different areas of research where superposition states play a significant role. That the two areas are so distinct is testament to the importance of superposition as a novel feature of quantum theory.

Firstly, we investigated what options there are for understanding the relationship between superposition states and a realist underlying physical theory. We were motivated by the wide range of work that indicates that quantum states are most naturally thought of as being states of knowledge. That is, they are mathematical objects with which we can construct an information theory and possibly nothing more. This allows us to reason about the behaviour of experiments that we carry out, but crucially remains a tool for thinking rather than an objective part of reality.

In order to take this kind of approach while maintaining that systems have independent and objectively real properties we reviewed ‘ψ-epistemic’ ontological models. Before our contribution, only one ψ-epistemic model had been constructed that accounted
accurately for the measurement statistics of experiments carried out with a single qubit.

The first significant contribution of this thesis is to demonstrate by example the possibility of a \(\psi\)-epistemic model in arbitrary dimension.

Our model is especially important in light of other recent work that rules out these \(\psi\)-epistemic models. All of these no-go theorems involve making additional assumptions to those strictly required for a well-formed ontological model. Establishing whether the quantum state is more like a state of knowledge (the \(\psi\)-epistemic view) or an element of reality (the \(\psi\)-ontic view) is a fundamental question in understanding the foundations of quantum theory. Indeed, as we saw, the answer has impact beyond interpreting the theory. Our work demonstrates that in order to rule out the realist \(\psi\)-epistemic position these extra assumptions are strictly necessary.

Explicitly, our work and the subsequent modifications demonstrate that superposition states can be associated with overlapping probability distributions in the underlying (ontic) state space. Although our model does not fully explain the fact that non-orthogonal quantum states cannot be distinguished with certainty, we put a lower bound on the extent to which this indistinguishability could be attributed to a lack of knowledge about an underlying reality. We also commented on the relationship between the model that we introduce and classical simulations of quantum teleportation, extending previous classical simulations of quantum teleportation to arbitrary dimension for the teleported system.

Next, we turned our attention to dynamical random processes on networks in the guise of random walks on graphs. Graphs are representations of complex networks that are ubiquitous not only in physical but also social sciences and other areas. Understanding structure within these networks is one of the key features that allows us to build simplified versions of the network. These simplifications are important since they often allow effective simulation of complicated processes that would not be tractable on the graph in its entirety. Yet in the simplified version of the graph we can study these processes and get reasonable approximations to their behaviour.

A key feature of the structure of complex networks is community structure. We reviewed the notion of communities in graphs, and also some of the key methods that are used for discovering communities. We also mentioned some of the methods that are used to
establish when a partitioning of a graph into communities is a good reflection of the underlying structure. Relatively recently progress has been made in the field of community detection using the dynamical process of a classical random walk. Investigating this process has led to a wide range of different algorithms and quality measures for community detection and evaluation, many of which are of practical importance.

As sensor technology has improved we have been able to study the relationships between elements of networks where a quantum description of the dynamics becomes necessary. Typically these are at the scale of molecules. As an example, an important collection of such networks is the light harvesting complexes that form a part of the photosynthesis process in nature.

Our second key contributions involved extending the classical methods for community detection to quantum dynamics. We introduced a new centrality measure based on observable signals from a graph in which a quantum dynamics was evolving called flashiness. Flashiness is a measure of the extent to which a particular vertex of a graph is involved in the quantum dynamical process on the graph, inspired by classical approaches such as random walk edge betweenness and stability of partitionings. After investigating some of the simpler properties of flashiness on a series of small, artificial graphs, we turned our attention to an explicit community detection algorithm.

We investigated the utility of flashiness for community detection in so-called LFR benchmark graphs. These LFR graphs represent a standard for community detection since they have several key properties in common with many real networks. In addition, they are typically seen as challenging set of benchmarks, relative to some of the benchmarks that were previously used in classical community detection.

We wrote down an algorithm that used flashiness in order to establish when vertices of a graph behave similarly under certain dynamics. This similarity of behaviour is one of the hallmarks of a pair of vertices belonging to the same community. Our algorithm involved some simple post-processing of the observed flashiness of the vertices of a graph that allowed us to identify sets of vertices with similar flashiness. We then agglomerated those vertices into the same community, and having repeated the algorithm returned a partitioning. Our results were comparable in the accuracy of their partitioning to the
modern classical algorithm of the Louvain group (an algorithm based on the dynamics of a classical random walk).

5.1 Future work

Both of the areas that we investigated in this thesis promise further interesting work in the future. Indeed, there are several natural and direct extensions of the work presented here.

Our work and that of others addressing the $\psi$-epistemic/$\psi$-ontic distinction has helped to map out the range of physical assumptions that are compatible with each of these two categories of ontological model. Ontological models are a natural and direct expression of the kind of reality that we intuit from classical physical theories (and indeed information theory). Further, while the assumptions that rule out $\psi$-epistemic models are intuitively compelling, it is nevertheless the case that the explanatory power of taking an epistemic view of quantum states is formidable.

As we mentioned above, there are two assumptions key to any ontological model – the existence of objectively real physical states, and causal relationships between those states and measurement outcomes. In parallel to the work mentioned here, where we adopt a simple causal structure, an industry has developed in which researchers are investigating the types of causal structure, including retrocausal models, that are compatible with quantum measurement statistics (see for example [CDPV13, OCB12, WS12]). The wider area of investigating causal structure between probabilistic events has seen substantial investigation in classical information theory (see for example [KF09]), with applications from medicine to self-driving cars. Applying these techniques to quantum states and measurement statistics is likely to reveal interesting restrictions or extensions on the simple causal relationships that we assumed. Already, a landscape of assumptions and associated restrictions analogous to those that we discussed here is beginning to emerge.

Another way to view our and related work in the ontological models framework is as reducing the set of features that are usually considered characteristically quantum. The aims of this type of investigative programme are two-fold: firstly, one could hope to
reveal which features of quantum theory might explain the increased performance of quantum computers. (Or, somewhat more pessimistically, the exponentially poor performance of their classical counterparts.) Secondly, narrowing down the set of characteristically quantum phenomena could provide hints at the key features of physical axioms for quantum theory. Clearly a physical axiomatisation of quantum theory would have great significance for the goal of interpreting the theory.

In the less immediate term, the continued development of practical technologies that exploit the exotic behaviour of quantum systems – such as quantum computation and cryptography – is likely to draw further attention to foundational questions in quantum theory. This pattern is not new, with the original motivation at the genesis of quantum theory attributable to quirky observations in (at the time) new technologies.

As we discussed in chapter 3, many physical situations admit a description in the form of complex networks. A plethora of approaches has developed over recent years for analysing these networks, and increased computing power has rendered tractable – and indeed describable – many large and interesting contexts. Fundamental to understanding these networks is a means of investigating the mesoscopic structure, which was the aim of our work in chapter 4.

Our community detection technique is a single example from a wider class of approaches using noise detection algorithms. Among similar approaches, more refined noise detection algorithms could be used to map the progress of a quantum walker through the target network while still relying only on the probability signal. Refining our approach to take advantage of the nuances and sophistication of modern signal processing could well improve the performance of our algorithm, for example by increasing the ‘separation’ (or equivalent quantity) between vertices within a particular community and those without. Indeed, it is likely that different techniques prove optimal for different types of network.

Practically, this kind of investigation could find application in networks of quantum processors. Traffic analysis is a current hot topic, with the so-called meta-data of communicating parties seen as of exceptional importance by security services. Further, understanding this kind of dynamics could help answer fundamental questions about optimisation of quantum processor design – especially true in the case of a quantum computer.
that is distributed over a relatively wide area.

The key point of our work, however, was to show that just as the randomness of classical random walks can be used to discover communities in graphs, the characteristically different type of randomness associated with the unpredictability of superposition states over the vertices of a graph could also be used for a similar purpose. In spite of these similarities, there are clearly many exciting possibilities for differences to be discovered in the future.
In this brief appendix we provide a few of the key snippets useful in recreating the simulations of chapter 4. We used Python 2.7.3 with Numpy, the scipy.linalg and IPython.parallel modules. For graphing and visualisation, we used Matplotlib and Networkx. The ‘Louvain Group’s algorithm for community detection is available as a python package via Bitbucket [Ayn].

A.1 Generating graphs

First we provide snippets to generate the benchmark graphs that we used. We constructed the graphs as adjacency matrices, which are useful for the simulations and can also be digested by packages such as Networkx.

First a function for generating glued trees. We create the glued trees in three steps: firstly creating two binary trees, then generating the ‘glue’ edges, then finally arranging these three components in an adjacency matrix.

```python
def gluedTrees(height):
```
Return the adjacency matrix for a pair of glued trees
each of specified height

```
# Create two binary trees
A = gluedTreeAdj(height)

# Calculate the number of 'glue' edges, and the vertices to glue
sizeRange, size = verticesToGlue(height)

glue = randomGlue(size)

# Insert the glue edges
A[sizeRange[0]:sizeRange[-1], sizeRange[0]:sizeRange[-1]] = glue
return A
```

The functions required are as follows:

```python
def randomGlue(size):
    """Generate a particular set of glue edges. size is
    the number of leaves to glue. """

    # Leaves to glue
    face = np.zeros((size, size))

    # Create a glue bottle with the right amount of glue
    bottle = np.eye(size, size, -1) + np.eye(size, size, 1)
    bottle[size-1, 0] = 1
    bottle[0, size-1] = 1
    randomCols = range(size)
    random.shuffle(randomCols)

    # Dispense the glue between random pairs of leaves
    glue = np.array([bottle[x] for x in randomCols])

    # Piece together in the form [[GLUE, ZEROS], [ZEROS, GLUE]]
    TOP = np.concatenate((face, glue), axis=1)
    BOTTOM = np.concatenate((glue.T, face), axis=1)
    randomGlue = np.concatenate((TOP, BOTTOM), axis=0)

    return randomGlue

def randomGlueAvg(size):
    """Create 'averaged' random glue edges -- i.e. join of
    left- and right-leaf vertices such that weights sum to 2.
    size is the number of leaves to glue. """

    # Leaves to glue
    face = np.zeros((size, size))

    # Weighted glue-edges
    glue = 2./size * np.ones((size, size))

    # Piece together in the form [[GLUE, ZEROS], [ZEROS, GLUE]]
    TOP = np.concatenate((face, glue), axis=1)
    BOTTOM = np.concatenate((glue.T, face), axis=1)
    randomGlue = np.concatenate((TOP, BOTTOM), axis=0)

    return randomGlue

def verticesToGlue(height):
```

---

160
size = 2**(height - 1)
sizerrange = range(2**(height - 1) - 1, 3 * (2**(height-1)))
return sizeRange, size

def gluedTreeAdj(height):
    """ Generate an adjacency matrix for a glued tree that is 2*leftCols wide, with toMiddleEdges edges emanating from vertices towards the middle of the element, and toEdgeEdges towards the outside. """

    numberOfVertices = sum([2**n for n in range(height)])
    # Blank adjacency matrix of the correct size
    adjacency = np.zeros((2*numberOfVertices, 2*numberOfVertices))
    # First add directed edges for one tree
    for vertex in range(numberOfVertices - 2**(height-1)):
        adjacency[vertex,2*vertex + 1] = 1
        adjacency[vertex,2*vertex + 2] = 1
    # Second make those edges un-directed
    adjacency += adjacency.T
    # Third duplicate the tree
    adjacency += np.fliplr(np.flipud(adjacency))
    return adjacency

A somewhat simpler function returns the adjacency matrix for the ‘blob’ fragments:

def blob(numberOfVertices):
    """ Create a ‘blob’ fragment with numberOfVertices vertices """

    adjacency = np.ones((numberOfVertices,numberOfVertices))
    # Remove self-loops
    np.fill_diagonal(adjacency,0)
    # Remove the entry--exit edge
    adjacency[numberOfVertices-1,0] = 0
    adjacency[0,numberOfVertices-1] = 0
    return adjacency

For both the GT and also blob cases, we finally need a function to ‘tile’ the fragments together into a cycle. This takes a list of adjacency matrices for each fragment, then adds an edge between the neighbouring elements of the list (between the entry and exit vertices for each fragment):

def tileFragments(communities):
    """communities is a list of adjacency matrices for each fragment"""
    lengths = [len(x) for x in communities]
    fullGraph = np.array([[]])
    for comm in communities:
fullGraph = linalg.block_diag(fullGraph, comm)
fullGraph = np.delete(fullGraph, 0, 0)
fullGraph[-1, 0] = 1
fullGraph[0, -1] = 1
offset = 0
for comm in communities[:-1]:
    fullGraph[len(comm) + offset, len(comm) + offset - 1] = 1
    fullGraph[len(comm) + offset - 1, len(comm) + offset] = 1
    offset += len(comm)
return fullGraph

So for a cycle of glued trees of heights 3, 4, 5, one would run:

```python
>>> fullAdjacencyMatrix = tileFragments([gluedTrees(3), gluedTrees(4),
                                       gluedTrees(5)])
```

In order to generate LFR benchmark graphs, we used the script written by Andrea Lancichinetti to accompany [LFR08]1. The script outputs parameters for the generated graph, as well as a community dictionary for the vertices as seeded – which is used to evaluate the the success of our algorithm. The following function digests the output of Lancichinetti’s program returning the necessary variables for what follows:

```python
def digestLFRgraph(size, graph="/Path/To/Graph"):  
    """Given a graph label and size, populate useful objects for analysis""
    with open(graph + "/network.dat") as tsv:
        adjacency = np.zeros((size, size))
        edgeList = []
        for line in csv.reader(tsv, dialect="excel-tab"):
            adjacency[int(line[0]) - 1, int(line[1]) - 1] = 1
            edgeList.append([int(line[0]) - 1, int(line[1]) - 1])

    with open(graph + "/community.dat") as tsv:
        communities = np.zeros((size, 1))
        for line in csv.reader(tsv, dialect="excel-tab"):
            communities[int(line[0]) - 1] = int(line[1])

    degrees = np.zeros((size, 3))
    for iterator in range(len(adjacency)):
        degrees[iterator, 0] = iterator
        community = communities[iterator]
        for edge in list(np.nonzero(adjacency[iterator])[0]):
            # i.e. for each neighbour
            if communities[edge] == community:
                degrees[iterator, 1] += 1
            else:
                degrees[iterator, 2] += 1

    boundaryVertices = np.nonzero(degrees[:, 2])[0]

    return adjacency, communities, degrees, boundaryVertices, edgeList
```

1Available at https://sites.google.com/site/santofortunato/inthepress2 (accessed 2015–12–14).
A.2 Simulation

Simulating the walk is the main resource-intensive process, requiring some time to complete the matrix exponentiation at each timestep. We used the `scipy.linalg` module’s `expm` function in order to generate the unitary matrix for the evolution of the walk between the initial state and the appropriate time. Given the unitary matrix for the evolution $U(t)$, the probability for the walker to be observed at vertex $j$ having initially had support only on vertex $i$ at time $t$ is simply $(U(t)_{ij})^2$. This means that we do not need to perform the calculation for each starting vertex: it is enough to calculate the unitary once per timestep for each graph.

def simQ(hamiltonian, times, initStatesToSim):
    """Simulate a quantum walker evolving with the hamiltonian for each timestep in (list) times, and record the inter-timestep differences for walks initialised at (list) initStatesToSim""

ticker = False  # Avoid the fencepost error

    # Initialise the returned objects
    allProbabilities = []
    differences = {}
    for starting in initStatesToSim:
        differences[starting] = np.zeros((1, len(hamiltonian)))

    numberStartingStates = len(initStatesToSim)
    totalTime = np.max(times)
    currentProgress = 0

    for time in times:
        # Keep track of progress.
        if np.floor((100 * time) / totalTime) > currentProgress:
            print currentProgress,
            currentProgress += 1

        unitary = linalg.expm(-1j * time * hamiltonian)
        thisTimeProbabilities = []
        probMatrix = np.real(np.multiply(unitary, unitary.conj()))
        for starting in initStatesToSim:
            thisTimeProbabilities.append(probMatrix[starting][:])

        if ticker:
            for vertex in initStatesToSim:
                differences[vertex] += np.abs(np.subtract(lastTimeProbabilities[vertex],
                                                       thisTimeProbabilities[vertex]))

        # Remember probs for next iteration
        lastTimeProbabilities = thisTimeProbabilities
        ticker = True

    return differences, allProbabilities
A.3 Analysis

In the previous section we wrote a function in place to simulate the quantum walk on a particular graph which returns the vertex-wise differences in probability at each time. The flashiness is simply the sum of the absolute values in the differences list. With that in place, we are able to embed the function in our algorithm for community detection:

```python
def assignToCommunities(graph, probabilities):
    
    # Run the flashiness-based community detection algorithm
    # for graph using probabilities -- a list of element-wise-squared
    # unitary matrices, one for each timestep

    graphSize = graph['size']
    A = graph['adjacencyMatrix']
    comms = graph['seededCommunities'] # For comparison
    mixing = graph['mixingParameter']

    ## Run the simulation if necessary
    # D = np.sum(A, axis = 0)
    # DinvSqRt = [1./np.sqrt(i) for i in D]
    # hamiltonian = np.dot(np.dot(np.diag(DinvSqRt), A), np.diag(DinvSqRt))
    # times = np.linspace(0, 100, 10000)
    # differences, probabilities = simQ(hamiltonian, times, graphSize)

    ## Find the communities

    commsToReturn = []
    allocated = []
    allocLength = []
    toAllocate = deque(range(graphSize))

    while len(allocated) < graphSize:
        allocLength.append(len(allocated))

        # Check whether the algorithm has stalled > 15 times...
        if allocLength.count(allocLength[-1]) > 15:
            print "Something went wrong with " + str(graph['label'])
            break # ... if so, give up (rarely happens --
            # -- graph disconnected?)

        # Get a new, unallocated starting vertex
        newStartingVertex = False
        while not newStartingVertex:
            startingVertex = toAllocate.popleft()
            newStartingVertex = not (startingVertex in allocated)

        notFound = True # Success flag
        thisCommunity = [startingVertex]

        j = 0

        # Container for variances of separated group
        topVars = []
```

164
totalDiffs = np.zeros((1,graphSize))
success = True
while notFound:
    probsToUse = ([probabilities[step][startingVertex]
           for step in range(10000)])
    flashiness = flashinessFromProbs(probabilities)
totalDiffs += flashiness[-1]

j += 1

# Sort the summed flashinesses from previous iterations
sortedDiffs = labelAndSortDiffs(totalDiffs[0])

# Flashinesses from separated vertices
topDiffs = []
for vtx in thisCommunity:
    for temp in range(len(sortedDiffs)):
        if sortedDiffs[temp][1] == vtx:
            topDiffs.append(sortedDiffs[temp][0])
topVars.append(np.var(topDiffs))

if j > 12:
    # Start checking whether we have a community
    notFound = not testIfFound(topVars)
if j > graphSize / 2:
    # Failure condition
    success = False
    break

newVertex = False
i = 1
while not newVertex:
    try:
        sanityCheck = (sortedDiffs[-i][1] not in thisCommunity) and\
        (sortedDiffs[-i][1] not in allocated)
        if sanityCheck:
            newVertex = True
            thisCommunity.append(sortedDiffs[-i][1])
            startingVertex = thisCommunity[-1]
            i += 1
    except:
        success = False
        break

if success:
    for vtx in thisCommunity:
        if vtx not in allocated:
            allocated.append(vtx)
        commsToReturn.append(thisCommunity)
else:
    toAllocate.append(startingVertex)  # re-do later

answer[graph['label']] = commsToReturn

# Finally, gather ‘true’ labels in comparable format
trueLabels = []
for vtx in range(graphSize):
    for c in comms.keys():
        if vtx in comms[c]:
            trueLabels.append(c)

165
trueLabels.append(c)
break
trueLabelsAll[graph['label']] = trueLabels

return answer, trueLabelsAll

Here, the function testIfFound is used to decide whether the separation between the ‘in’-community vertices and the ‘out’-community vertices is large enough:

def testIfFound(topVars, parameter=2.5):
    meanIncrease = np.mean([topVars[i] - topVars[i-1]
     for i in range(1,len(topVars)-1)])
    variance = np.var([topVars[i] - topVars[i-1]
     for i in range(1,len(topVars)-1)])
    isFound = topVars[-2] + meanIncrease + parameter*\
     (np.sqrt(variance)) < topVars[-1]
    return isFound

Finally, the labelAndSortDiffs function keeps track of the total flashiness and vertex labels:

def labelAndSortDiffs(diffs):
    allDiffsToSort = []
    for i in range(len(diffs)):
        allDiffsToSort.append((diffs[i],i))
    return sorted(allDiffsToSort,key=lambda x: x[0])


BIBLIOGRAPHY


