Learning random geometric graphs, and their applications

A Thesis Submitted for the Degree of Doctor of Philosophy

By

Chuqiao Zhang

Department of Mathematics,
Brunel University London

2025

To the memory of my grandfather, Mr. Qiu, whose wisdom continues to inspire me, and to all those who have loved and supported me throughout this journey.

Declaration

I hereby declare that this thesis entitled "Graphical Models and Networks: Learning of Random Geometric Graphs and Further Applications" and the work presented in it are my own and have been generated by me as the result of my own original research. This work has not been submitted, either in whole or in part, for any other degree or qualification at this or any other institution.

Where I have drawn upon the work, ideas, or writings of others, these have been fully acknowledged by means of reference to the relevant sources. All work presented herein conforms to the university's regulations on plagiarism and academic integrity.

Chuqiao Zhang
July 2025

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Dr. Dalia Chakrabarty, for her invaluable guidance, encouragement, and unwavering support throughout my Ph.D research.

I thank all the staff and colleagues at Brunel University London, for creating an inspiring and supportive environment. Special thanks to doctoral researchers Gargi and Kane for every meetup and discussion.

Finally, I am deeply grateful to my family, my friends and my cats, for their love, patience, and constant encouragement during this journey.

Abstract

In the Bayesian world, referring to a variable as "random" means that this variable – a scalar, tensor, a network, etc. – takes a value with a probability, such that it attains a values that live in the support of this probabilistic distribution. lies in a given interval, with uncertainties. We want to make inference on the unknown parameters of a model, using the modelled probability distribution of each such unknown, given the data. In my thesis, I discuss the motivation for undertaking inference via sampling unknowns given the data, and introduce a learning of realisations of a random graph variable. This random graph is a Random Geometric Graph (RGG) that is drawn in a probabilistic metric space. Further learning of this graph variable has also been explored in the thesis, to identify the optimal cut-off value that is imposed on the posterior probability of any edge given the (multivariate) dataset at hand. Such an optimal cut-off then corresponds to graph realisations that produces the most robust - to changes in the cut-off values - graph. Theoretical illustrations of the learning of this graph and the applications of such graph learning are presented using real-world data, towards: (1) scoring of severity of a disease, by computing the distance between the posterior of the learnt random graph variables, given the time series data on the physiological parameters of two patients as they suffer from the disease; (2) a method for the learning of the individualised recovery trajectory of patients who are enrolled on a physical rehabilitation programme, with the aim of regaining their lost mobility; (3) a new way of recognising critical residues of an example protein, using static and dynamic nodal degree distributions of random graphs learnt using molecular dynamical simulations of the protein. Based on my PhD research, future work is discussed.

Contents

Chapter 1

Background

Graphs and networks are foundational tools for representing and analysing complex systems composed of interconnected elements. Whether modelling the internet; transportation systems; biological interactions; social relationships; etc. - networks offer a powerful way of understanding how structure influences function. In each of the cases enumerated above, each node in a network represents an individual unit, such as a computer; city; gene; one single person; etc., while edges capture interactions or relationships. Thus, graphs and networks provide a language that helps describe and investigate diverse phenomena, using a common mathematical framework. Moreover, many important behaviours in complex systems - such as the spread of diseases or the flow of information - are not properties of individual components, but emerge from the pattern of connections amongst them. Understanding this whole structure, and how it influences dynamics, is central to the science of networks.

This reveals one more strength of graphs and networks: clear and better visualisation of inter-variable relationships, which offers intuition on how to inspect the structure and behaviour of complex systems. By representing entities as nodes, and their interactions as edges, network diagrams allow us to see patterns that would be difficult to detect from raw data alone. This visual clarity is valuable in most fields, such as social network analysis; biology; computer science; etc, where understanding how parts relate to one another is often more important than understanding the parts themselves.

In practice, however, the precise structure of a network is often unknown - too large to fully observe, or constantly changing. Relationships are rarely formed by design; instead, they arise from local interactions; environmental constraints; or probabilistic behaviour. This is where randomness becomes essentially relevant. Rather than relying on a fixed or deterministic structure, researchers often model graphs as random structures, in which the collection of nodes and edges is generated according to probabilistic rules. By embedding these aspects into models of random graphs, researchers can construct more flexible, robust representations of real-world systems and better understand the interplay between chance and structure in complex networks. For instance, by treating a social network as a random graph, one can ask how likely it is to be connected, how information might propagate, or how

vulnerable it is to node failures. Random structures also allow for the exploration of universality in networks - certain global properties emerge not from the specific details of each system, but from shared statistical features. Many networks exhibit small-world characteristics or scale-free degree distributions (Barabási and Albert, 1999; Barabási et al., 2000; Kumar et al., 2000), regardless of their underlying mechanisms. Modelling networks randomly helps reveal these broad patterns and provides a testbed for comparing theoretical predictions with empirical observations.

Random graph models also help identify phase transitions. For example, in the classic Erdös-Rényi model (Erdos and Rényi, 1959), as the probability of edge formation increases, the network undergoes a critical transition: below a certain threshold, most nodes are isolated or in small clusters, but above it, a giant connected component rapidly emerges (Amaral et al., 2000). This behaviour mirrors phase transitions in physical systems, such as the shift from liquid to gas, and has profound implications for understanding real-world networks. In epidemiology, such a transition might represent the tipping point at which a disease outbreak becomes an epidemic (Mata, 2021), while in communication networks, it could mark the point at which reliable connectivity is achieved (Cohen et al., 2000). Random graphs thus offer a valuable framework for studying the emergence of global properties from local randomness, and for anticipating critical thresholds in complex systems.

1.1 Random Geometric Graphs: existing studies

Random Geometric Graphs (RGGs) are an important class of spatially constrained random graphs that have been widely studied in probability theory, combinatorics, and network science. It provides a realistic model for networks where connectivity is limited by geometric distance. An RGG is an undirected graph formed by randomly sampling nodes from an identified distribution, (such as a uniform distribution) in a metric space, where an edge is placed between two nodes if and only if the distance between them in this host space, is less than a pre-defined threshold, excluding self-loops. Here this pre-chosen threshold plays a crucial role in determining network connectivity and clustering.

In the Erdös-Rényi model (Erdos and Rényi, 1959), every graph defined on a fixed vertex set, with a predetermined number of edges, is equally probable. Gilbert (1961) established fundamental connectivity properties, showing that a critical connection radius exists: the Gilbert model, also known as the Erdös-Rényi–Gilbert model, each possible edge has been assigned an independent probability of existence. Subsequent research by Penrose (1999, 2003) provided a rigorous probabilistic learning of RGGs, exploring their degree distributions, clustering properties, and connectivity thresholds. Dall and Christensen (2002) further investigated the critical connectivity of RGGs by examining the size of the largest cluster and derived an analytical expression for the clustering coefficient of RGGs. The study of higher-order connectivity properties extends RGG theory beyond basic connectivity. Penrose (1999) introduced the concept of "k-connectivity", which ensures network robustness by requiring each node to have at least k independent paths to others.

Another variation of RGGs is the k-nearest-neighbour (k-NN) graph, where each node is connected to its k closest neighbours instead of using a fixed radius (Balister et al., 2005). Devroye and Fraiman (2014) extended RGGs to inhomogeneous random graphs, where the node density varies across space, and the probability of an edge existing between two vertices varies according to a specified kernel function. These models play a crucial role in probabilistic graph theory, as a rigorous framework for defining properties that hold for graphs in large-scale settings. RGGs have found numerous applications in wireless networks (Mao and Anderson, 2012; Gupta and Kumar, 1998; Kumar and Gupta, 2000; Muthukrishnan et al., 2005), biological interaction networks (Doncheva et al., 2012; Haspel and Jagodzinski, 2017; Kantelis et al., 2022), and social network modelling (Newman et al., 2002), and have been extensively analysed to improve real-world network designs.

However, in many real-world applications, connections between nodes are not strictly deterministic - this limitation is then addressed by Soft Random Geometric Graphs (Soft RGGs) (Penrose, 2016; Giles et al., 2016), where edge formation follows a probability function. The concept of probabilistic connections in geometric graphs was first introduced by Waxman (1988), which became a foundational model for SRGGs. SRGGs are an extension of traditional RGGs, where the connectivity rule is probabilistic. This makes them useful in modelling real-world networks with uncertain connectivity. Penrose (2003, 2016) provided a theoretical framework for analysing connectivity in both RGGs and SRGGs, which was a landmark in geometric graphs.

1.2 Inter-graph distance

When talking about the distance between two graphs in my research, we are actually referring to the distance between two probability distributions - to be specific, the posterior probabilities of two random graph variables, learnt given two datasets. Several statistical distance measures play significant roles in probability theory, information theory, and machine learning. One of the earliest and most influential measures of statistical distance is the Kullback-Leibler (KL) divergence introduced by Kullback and Leibler (Kullback and Leibler, 1951). KL divergence quantifies the divergence between distribution Q and distribution P, and it is asymmetric. A symmetric divergence, Jensen-Shannon (JS) divergence (Lin, 1991; Gibbs and Su, 2002), which has widely applied in bioinformatics and genome comparison. Wasserstein distance (Kantorovich, 1939; Vaserstein, 1969; Villani, 2008) provides a meaningful distance metric even when distributions have non-overlapping support. Beyond divergence measures, probability metrics provide another way to quantify distributional differences, such as Total variation (TV) distance (Csiszár, 1967), which has been widely used in hypothesis testing and Bayesian inference. The Hellinger metric (Hellinger, 1909; Bhattacharyya, 1943; Nikulin, 2001) is a robust distance measure between densities, and is popular in the statistics literature given its ease of computation. The Hellinger distance is particularly effective in cases where distributions have overlapping support and is frequently used in pattern recognition and Bayesian

updating.

To compute the inter-graph distance/divergence between the posteriors of the graph variable learnt given two distinct datasets, I have applied the Hellinger distance and the KL divergence.

1.3 Gist of my thesis

In my research, I have learnt realisations of a random the graph variable, given multivariate datasets, where this random graph-valued variable is a Random Geometric Graph (RGG) drawn in a probabilistic metric space. I implement this graph learning by learning each edge of such a sought graph, given the inter-column correlation of the given dataset, and apply this in multiple real-world settings.

In particular, I implement the computation of a statistical distance/divergence between the posterior probability of a random graph given the correlation of one given dataset, and that given a second dataset. Such a computed distance informs on the strength of the effect of the generative mechanism that gives rise to one dataset, distinguished from the second set. Such mechanisms comprise experimental conditions under which the two datasets are generated, or even the time. Again, my doctoral work is marked by applying this inter-graph distance to the solution of multiple real-world problems as indicated below.

A random graph drawn in a probabilistic metric space also allows a data-driven thresholding that defines the geometric aspect of the sought RGG. In this thesis I demonstrate the details of our statistically driven, context-specific threshold selection technique and apply it to several datasets, in the latter chapters. By rooting threshold selection in the data rather than in user preference, the resulting graphs are interpretable, reproducible, and computationally tractable across a wide range of applications.

1.3.1 In Chapter 1

Earlier in this chapter, I introduced graphs and networks as comprising a mathematical framework for modelling complex systems, emphasising probabilistic approaches. I discussed the basic concept of a random graph, including random geometric graphs, to realistically represent the correlation structure of a multivariate dataset, alongside Bayesian techniques for learning graph structures from data. Details of RGGs will be discussed in Chapter 2, and the learning of such graphs will be undertaken within a Bayesian framework. In fact my entire doctoral work is performed within the Bayesian approach.

I also undertake the Bayesian inference, following Bayesian modelling, and devote the rest of this chapter to the discussion of such sampling-based inference methods, as relevant to my graph learning endeavours. Sampling-based Bayesian inference methods - such as MCMC and Rejection sampling - are applied to sample from the posterior distribution of the correlation between two variables given the data at hand; and to sample from the edge variable connecting two nodes in the graph,

given the correlation between the random variables that are attached respectively to these two nodes. The discussion of uncertainty quantification on any learnt variable is further supported through Highest Posterior Density regions, which are incorporated in the graph learning that I undertake in my work. Thus, the graphs are learnt such that any edge is learnt with identifiable uncertainty. These inferential techniques establish the fundamentals of the inference that I undertake towards learning the RGGs in various real-world settings, as discussed in the subsequent chapters. In these chapters, I discuss further investigation of developing methodology and applications of such probabilistic graph learning methods to real-world datasets. I illustrate these sampling techniques by applying them to small real-world, as well as synthetic datasets.

1.3.2 In Chapter 2

I discuss the details of learning the random (geometric) graph - drawn in probabilistic metric spaces - in Chapter 2, along with studies of thresholding edges with a cut-off probability, that underlines the "geometric-ness" of the random *geometric* graph that I seek, given the relevant dataset. A graph is learnt by learning the edges of this graph variable, within a Bayesian approach, that permits the uncertainty of any edge variable to exist in the graph, given the correlation structure of the data. In this chapter, I also discuss the details of computing the statistical distance/divergence between the posterior probabilities of RGGs learnt given two distinct - and disparately-long datasets. As stated above, such an inter-graph distance/divergence is then useful in providing a scalar value that embodies the difference between the capacities in the mechanism that generates the two relevant datasets. Thus, Chapter 2 can be summarised as the chapter in which the basal technique for graph learning, and for the usage of the learnt graphs, towards solutions to real-world problems, various examples of which I will address in the remaining chapters.

As I stated above, in this chapter I choose the cut-off probability on the edge posterior probability, allowing for acceptance of this edge in the pursued RGGs. To clarify, I recall that by definition, an RGG places an (undirected) edge between two nodes whenever the distance between them is below a certain threshold. In an RGG drawn in a probabilistic-metric space, each potential edge is represented by a probability distribution over positive support, and we include that edge in the graph only if its associated probability exceeds a cut-off parameter τ. Choosing τ "meaningfully" is crucial - if τ is set too high, very few edges will be accepted and the resulting graph will be overly sparse, potentially missing important structural relationships in the data from the visualised graph. Conversely, if τ is too low, almost every pair of nodes will become connected, yielding a dense graph that obscures meaningful patterns. In my thesis, I derive a method for selecting this threshold τ, so that the learnt graph faithfully represents the correlation structure of the data. In the future, I will seek other possible approaches to explore and reinforce our threshold choice.

1.3.3 In Chapter 3

Out of the several real-world applications that this graph-learning paradigm offers a powerful lens on, Chapter 3 demonstrates its utility on understanding and usage of clinical datasets comprising observations on patient phenotypic parameters, during their period of sufferance from a disease called Veno Occlusive Diseases (or VOD) that afflicts recipients of Bone Marrow Transplant (or BMT). Patient sof certain blood cancers and other haematological ailments (such as Thalassemia) sometimes opt to undergo a BMT, as a cure for their ailment, but the very BMT often induces VOD in the patient. I am involved in a project in which the aim is the prediction of a score that informs on the severity with which VOD will develop in a patient after they would have undergone their BMT, but we make this prediction at the pre-transplant stage. This prediction is made possible by learning the relationship between the vector of pre-transplant variables and this score, which in itself is possible only if we have the adequate training dataset to use. The VOD severity score of any patient who contributes to this training data then has to be learnt. I learn this a part of my doctoral work, by learning the realisation of an RGG variable, given the time series data on phenotypic parameters, recorded for each patient in the training cohort, i.e. for each patient who has already had their BMT. Then during a time interval that includes the time point of the BMT for such a patient, their parameters recorded, though some patients die early, resulting in disparately-long time series for the different patients. I learn the RGG given such a time series data for each patient, and compute the inter-graph distance between the RGGs learnt for any patient, and that of an arbitrarily-chosen "reference patient". Such a distance then offers the score that we seek, to populate the aforesaid training data.

1.3.4 In Chapter 4

In Chapter 4 again, I work on computing the distance between learnt RGG variables, but in this application, the graph-pairs that I learn the distance and divergence between, are learnt given (time series) data generated by the same patient, at distinct instances of them undertaking a given exercise. Such an exercise is undertaken at multiple, sequential instances, by a patient who is enrolled in a physical rehabilitation programme to regain their mobility, that is impeded in them, owing to a critical illness or injury. Repeating the exercises helps the patient recover their lost mobility. The time series data that is generated on doing any exercise at any instance, comprises values of the locations of 20 joints in the skeletal frame of the patient, that change with time as the patient is undertaking their exercise. Indeed, the correlation between locations of any two joints in this frame, reflects the level of recovery that is current in the patient. Therefore, the inter-graph distance/divergence, informs on patient recovery. Thus, I forward a score of such recovery, between one instance of the patient doing an exercise, and the next instance, as the distance/divergence between the posterior probability of the RGG learnt given the time series data generated by the patient at the second of these two instances, compared to that generated at the first instance. Interestingly, the time series data on the joints is recorded at

every instance of a patient doing an exercise, on a virtual platform.

1.3.5 In Chapter 5

Chapter 5 discusses a contribution to protein design. Three distinct methods for the automated identification of critical residues are introduced, provided by the method that permits the learning of RGGs in probabilistic metric spaces, where such RGGs are objectively and meaningfully thresholded. Such RGGs are learnt given a large multivariate dataset obtained from Molecular Dynamical simulations, performed by a collaborator. My work in this project is the identification of critical residues of this simulated protein, using random graphs that I learn, and meaningfully threshold. Identifying critical residues in proteins is essential since these residues often play key roles in maintaining structural integrity and in enabling catalytic activity; facilitating molecular recognition; and regulating function. Understanding which residues are critical allows, scientists to interpret the functional impact of genetic mutations, many of which are linked to diseases. In practical applications, this knowledge underpins drug development; protein engineering; and synthetic biology.

I learn the binary edge variable between any two nodes of this RGG, given the correlation between the state variable of any two residues computed using the simulated data that is the high-dimensional time series data on the state of each of the (156) residues of this protein. I develop a parameter (called β parameter) that is the degree of each node of the RGG that is learnt such that, to any node of this RGG, is attached the (categorical) random variable that informs on the state of a residue of the considered protein. In addition to this parameter, I also compute the η parameter that is the measure of dispersion of the temporally-evolving degree of any residue, across time points that mark the temporal locations of intervals, that I partition the full range of time included in the simulation, such that, a subset of the full simulated dataset is produced at each such interval. In addition to this dynamic parameter, I also define a parameter (δ parameter) for any residue, that is the difference between the posterior probability of the RGG learnt with data on all residues, and the posterior of the RGG learnt with the data on all but the considered residue. Ranking these parameters, allows us to identify the critical residues - residues with: the highest β values; the highest η values; and the lowest δ values indicate criticality.

1.3.6 In Chapter 6

So in summary, in Chapter 6, I state that my thesis utilises newly developed theory for learning a random geometric graphs of a multivariate datasets, the inter-variable correlation structures of which are known or can be computed. While this graph-learning approach offers a powerful new lens for examining dataset structure and - as I present in the various chapters - finds many practical applications, and also addresses the incompleteness in the current theory. The graph learning method - and the inter-graph distance computation - will be applied to various real-world applications from Healthcare and Biology, that I have expounded upon in the earlier

chapters, with the basic sample-based Bayesian techniques discussed in the very first chapter in the thesis. I will wrap up the chapter with a discussion of future work.

As stated above, all learning is performed within a Bayesian framework, using sample-based Bayesian inference, that includes rejection Sampling and MCMC. I discuss these techniques now, and include some experiments on small real data, to illustrate such inference.

1.4 Motivation, objectives and contributions

The motivation behind undertaking of my doctoral work is to employ random graphs towards the solution of distinct real-world problems that are marked by multivariate data. It is challenging to model the interaction amongst pairs of components of the generative mechanism that produces such data, and such interactions can be tracked within the high-dimensional setting, using random graphs. I am particularly interested in learning graphs of such data as random structures, so that we can attribute a probability distribution to the sought graph variable. Said probabilistic signature has the advantage of allowing for the computation of statistical distance/divergence between the graph variables given pairs of datasets generated under distinct experimental conditions, where said distance/divergence then serves as a proxy for the strength of the conditions that cause one of the datasets to be different from the other. I am also motivated to learn such random graphs as geometric, in order to avail of greater flexibility in my search for the strength of interaction between the distinct random variables that contribute to the data at hand. Ultimately, my aim is to apply such graph learning techniques to address different real-world problems.

Objectives of my doctoral study are enumerated as follows:

1. Learn random geometric graph variables of a given multivariate dataset, using sampling-based Bayesian techniques. Compute statistical distance/divergence between the posterior probabilities of RGG variables learnt given respective datasets. I will use the Hellinger metric and Kullback-Leibler measure as the distance and divergence measures respectively, in my work.

2. Develop a meaningful thresholding on the RGGs that I learnt, towards clear interpretability.

3. Learn the RGGs of time series data on physiological parameters of each patient in a cohort, as they progress through their sufferance of an oncological disease.

4. Learn recovery trajectories of each patient in a cohort of patients who are suffering from mobility loss, and are enrolled in a physical rehabilitation programme to regain their mobility. I will learn these trajectories by computing the distance/divergence between the RGGs of each patient learnt using the time series of performance indicators that the patient realises during their undertaking of an exercise on two consecutive instances.

5. Use the RGG learning techniques to identify critical residues of a simulated protein.

The methodological contributions of my work are enumerated below.

- In my doctoral work, I have implemented a recently-advanced method for learning random geometric graph variables drawn in probabilistic metric space, given multivariate datasets that comprise observations of a small to a high number of random variables. I extended this method to include datasets that comprise observations of categorical variables.
- I have also included learning of the correlation matrix using MCMC, within my doctoral work, in addition to the learning of the graph variable.
- I have developed a new interpretable method for thresholding in the geometric graphs that I have learnt. This thresholding technique exploits the random nature of the graph variable I learn, by identifying stationary points of the rate of change of the (logarithm of the) graph posterior probability, with the change in the thresholding. Thus, the best threshold in the given data is organically identified.

I have applied graph learning, and inter-graph distance computation in the following real-world applications.

- In my doctoral work, I have learnt the graphs of each of a cohort of patients given the respective patient's time series data on multiple physiological parameters that were recorded from a few days before to 18 days after the patient underwent a Bone Marrow Transplant. My learnt graphs were used within my collaborating team, to compute the distance between the posterior probability of the graph of each patient in this cohort, and an arbitrarily chosen "reference patient", as the relative score of severity of a disease that developed due to the Transplant, in the patient.
- I have employed my graph learning and inter-graph distance and divergence computation to learn the recovery trajectory of each patient in a cohort of patients who were suffering from mobility impediment, and were enrolled in a physical rehabilitation programme to help them regain their lost mobility.
- I have used the graph learning technique and my thresholding identification, to identify critical residue of a simulated protein, by developing three distinct parameters of criticality.

1.5 Sampling-based Bayesian inference I use in the thesis

When we cannot perform closed-form inversion of the cumulative distribution function - corresponding to the (posterior) probability density of the variable that we

want to learn given the data - we need to undertake Bayesian inference by sampling from said posterior. If the posterior density is univariate and fully specified, we can undertake Rejection Sampling, but otherwise, we will need to undertake Markov Chain Monte Carlo (MCMC).

1.5.1 Rejection Sampling

Before we get into implementing MCMC algorithm, we introduce another method for sampling a probability distribution, namely, rejection sampling (Forsythe, 1972; Devroye, 1986). We need to invert $f(x)$ to know the value of x to generate a sample from $f(x)$, while this $f(x)$ might not be invertible in a closed-form way. In this case, an invertible distribution, $g(x)$ has been invoked, to help sample from $f(x)$, such that $X = x^*$ is a sample that is easily generated from $g(x)$, and then $X = x^*$ is accepted, (or not), according to a prescribed acceptance criterion, such that $X = x^*$ is a sample from $f(x)$, (or not).

Definition 1.5.1. *If we sample $X = x^*$ from a function $g(x)$, for a real $M \in (0, \infty)$ such that*

$$Cg(x) \geq f(x), \forall x;$$

For $u \sim U[0, 1]$, and we accept the sampled x^* if

$$\frac{f(x^*)}{Cg(x^*)} \geq u,$$

then the probability distribution of this sampled variable x^*, is $f_X(x)$

1.5.2 MCMC

Markov Chain Monte Carlo (MCMC) methods were developed to address the challenge of sampling from complex probability distributions, particularly in scenarios where direct computation of expectations is infeasible, which has been widely applied and developed in fields such as statistics, econometrics, physics and computing science. MCMC provides an essential tool for Bayesian inference. In this section, we outline the motivation for using MCMC in the context of our research, along with the applications in real-world data.

Motivation and theoretical foundation

One of the main challenges in Bayesian statistics is computing the posterior distribution. Recall the very basic Bayes rule:

$$P(A \mid B)P(B) = P(B \mid A)P(A)$$

We replace A by an event - that we refer to as "*model*" - that some model parameters $X_1, ..., X_p$ take values of $x_1, ..., x_p$, and replace B by the event - referred to as "*data*" - that some observed variable Y takes value of y_i at the i-th instance, and there are N such instances of making observations.

$$P(model \mid data)P(data) = P(data \mid model)P(model)$$

$$\Rightarrow P(model \mid data) = \frac{P(data \mid model)P(model)}{P(data)}$$

In this equation, $P(model \mid data)$ is exactly what we want, namely the **posterior** probability that $X_1, \ldots, X_p$ take the value $x_1, \ldots, x_p$ respectively, given data $\{y_1, \ldots, y_N\}$. $P(data \mid model)$ is easy to conceptualise and interpret since the probability of that which is sought - i.e. the model - is provided. Such a posterior probability of the sought model is given, which is essentially fixed, namely the data. We compute the sought posterior using that which is easy to compute (i.e. the **likelihood**), and the **prior** (Gelman et al., 2013). The apparent shortcoming of Bayesian statistics is that it only gives the probability of unknowns, instead of the values of the unknown, which is ultimately desired in the real world. One ready way to learn the values of unknowns is that, by maximising the posterior probability density function given data, also known as MAP, namely, we are asking "which values of X are most probable, given the data?" We observe the values of X with the highest probability. However there are still some problems with optimisation.

- For multimodal posterior probability density function, it is difficult to perform Bayesian inference objectively, i.e. which peak do we choose, and why?
- Marginal posterior probability distribution is likely to be difficult to compute. Suppose there are p unknowns, $X_1, ...X_p$, such that the joint posterior pdf is

  $$\pi(X_1, ..., X_p \mid data).$$

  Then maximising this joint posterior pdf does not tell us where X_i attains its highest posterior probability, $\forall i = 1, ...p$. i.e. we want to know the marginal posterior probability distribution, for all X_i, given the data $i = 1, ..., p$. There is nothing in the optimisation idea that offers a marginal posterior of X_i, given data $\forall i = 1, ...p$.

To overcome these problems of optimisation, we prefer Bayesian Inference that comprises sampling from the joint posterior, allowing for the marginal posterior of X_i given the data $\forall i = 1, \ldots, p$. Markov Chain Monte Carlo (MCMC) provides a sampling-based approach that bypasses the difficult direct computation of the marginal posterior of X_i via integrating $X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_p$ out of the joint posterior, by generating samples from the posterior distribution with an easy and straightforward way, allowing for efficient approximation of the marginal distribution of X_i, $\forall i = 1, \ldots, p$.

Thus, MCMC offers a method for generating samples from complex probability distributions when direct sampling is infeasible. MCMC is particularly crucial in Bayesian inference, where evaluating high-dimensional integrals is computationally expensive, ruling out optimisation. MCMC operates by constructing a Markov chain whose stationary distribution corresponds to the desired target distribution. By

simulating this chain over multiple iterations, one can obtain a representative sample set that can be used to approximate expectations, compute marginal distributions, and perform model selection, etc.

The development of MCMC methods traces back to the Metropolis algorithm, introduced by Metropolis et al. (1953), which was initially applied in statistical physics to simulate thermodynamic systems. The algorithm was later generalised by Hastings (1970), leading to the Metropolis-Hastings (MH) algorithm, one of the most widely used methods for probabilistic inference. Hastings extended the Metropolis algorithm by introducing a more flexible proposal distribution, which allowed the algorithm to be applied to a wider range of statistical models and non-symmetric proposal distributions. Chib and Greenberg (1995) provided an intuitive step-by-step explanation of the MH algorithm, breaking it down into proposal distributions, acceptance criteria, and Markov chain properties. They also highlighted the differences and connections between the MH algorithm and Gibbs sampling, demonstrated that Gibbs sampling is a special case of MH, where the acceptance probability is always 1, making it computationally efficient when full conditionals are available. Geman and Geman (1984) introduced Gibbs sampling as a special case of MCMC that enables efficient sampling from high-dimensional distributions by sequentially updating individual parameters from their conditional distributions. Gelfand and Smith (1990), introduced Gibbs Sampling to Bayesian Statistics, as a general statistical tool for estimating posterior distributions. Casella and George (1992) provides one of the most accessible and widely cited explanations of Gibbs sampling - a clear, statistical explanation and mathematical justification of Gibbs sampling, and describes how Gibbs sampling constructs a Markov chain by iteratively sampling from full conditional distributions, making it accessible to statisticians unfamiliar with MCMC.

Over time, MCMC has evolved to include advanced variants. Robert and Casella (2004) in their book, provided a detailed description of Monte Carlo methods and in-depth Coverage of MCMC algorithms, not only MH algorithm, but also algorithms developed in the past few decades, such as Hamiltonian Monte Carlo (HMC) (Neal, 1993), Reversible Jump MCMC (RJMCMC) (Green, 1995), and Sequential Monte Carlo (SMC) (Del Moral, 2004), each designed to improve sampling efficiency and convergence in complex models. Additionally, not only in the field of Bayesian statistics (Chib and Greenberg, 1995; Gelman et al., 1995; Bolstad, 2010), MCMC methods have found extensive applications in the field, such as spatial statistics (Banerjee et al., 2014), machine learning (Andrieu et al., 2003), and financial areas such as portfolio optimisation and financial forecasting(Asmussen and Glynn, 2007; Rubinstein and Kroese, 2007).

The principle of MCMC algorithm is, for an arbitrary starting value $x^{(0)}$, we can generate a chain $(X^{(t)})$ using a transition kernel with a stationary distribution f, such that the distribution of $(X^{(t)})$ converges to a random variable from f. We could consider MH algorithm with target density f and the proposal density q, to produce a Markov chain $(X^{(t)})$:

Algorithm 1: Metropolis-Hastings Algorithm (Robert and Casella, 2004)

Given $x^{(t)}$

Generate $Y_t \sim q(y|x_{(t)})$ Take

$$X^{(t+1)} = \begin{cases} Y_t & \textbf{with probability } \rho(x^{(t)}, Y_t) \\ x^{(t)} & \textbf{with probability } 1 - \rho(x^{(t)}, Y_t) \end{cases}$$

where $\rho(x, y) = min\{\frac{f(y)}{f(x)}\frac{q(x|y)}{q(y|x)}, 1\}$.

It can be seen that the MH algorithm depends only on the ratios

$$\frac{f(y_t)}{f(x^{(t)})} \text{ and } \frac{q(x^{(t)} \mid y_t)}{q(y_t \mid x^{(t)})}.$$

We implement MCMC on the real-world problem, namely, the learning of the probability of failure of the O-rings of the Challenger Space Shuttle, which has been widely used in statistical analysis, Bayesian inference, and logistic regression modelling. We applied the following 4 types of MH and different proposal density q in each case will be learned:

- Random Walk Metropolis-Hastings
- Vanilla MCMC
- Independent Sampler
- Gibbs Sampling

1.5.3 Metropolis-Hastings algorithm implementation: on real (O-ring) data

On January 28, 1986, a routine launch was anticipated for the Challenger space shuttle, the shuttle broke apart after seventy-three seconds, killing all seven crew members on board. An investigation into the cause of the disaster focused on the critical seal, namely, O-ring. It is believed that damage to these O-rings due to the low temperature during the launch led to this disaster (Dalal et al., 1989). Table 1.1 below summarises observational data on O-rings for 23 shuttle flights. In the table, we have the temperature at flight time (measured in Fahrenheit units) and the failure of O-rings (1 stands for failure, 0 for success).

Mission	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
Temperatrue	53	57	58	63	66	67	67	67	68	69	70	70	70	70	72	73	75	75	76	76	78	79	81
Failure	1	1	1	1	0	0	0	0	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0

Table 1.1: O-ring Data

To learn the data, we consider the failure status of the O-ring to be a variable X, and the temperature to be a variable T. Here $X = 1$ is "Fail", i.e. O-ring stiffened, or $X = 0$, stands for "Not fail". Therefore we have $X \sim Bernoulli(p)$, where p is

the probability that $X = 1$, namely the O-ring fails. The Bernoulli probability mass function is

$$Pr(X = x) = p^x(1-p)^{1-x}, x = 0, 1$$

Hence, it is reasonable to regress this Bernoulli response X by using logistic regression:

$$Pr(X = 1) = \frac{1}{1 + exp(-(\beta_0 + \beta_1 T))},$$

Remark 1.5.1. *We consider the log odds, which is a linear predictor,*

$$log(\frac{Pr(X=1)}{1-Pr(X=1)}) = \beta_0 + \beta_1 T,$$

$$\begin{aligned}
\Rightarrow \frac{1 - Pr(X=1) + Pr(X=1)}{1-Pr(X=1)} &= 1 + e^{\beta_0+\beta_1 T} \\
\Rightarrow 1 - Pr(X=1) &= \frac{1}{1+e^{\beta_0+\beta_1 T}} \\
\Rightarrow Pr(X=1) &= 1 - \frac{1}{e^{\beta_0+\beta_1 T}} \\
&= \frac{e^{\beta_0+\beta_1 T}}{1+e^{\beta_0+\beta_1 T}} \\
&= \frac{1}{1+e^{-\beta_0-\beta_1 T}} \qquad (1.5.1)
\end{aligned}$$

$$\textit{i.e. } Pr(X=1) = \frac{1}{1+e^{-\beta_0-\beta_1 T}} \textit{ is our regression equation.}$$

Definition 1.5.2. *We then consider the likelihood of N independent and identically distributed (iid) samples: $X_1, X_2, ...X_N$, $\mathcal{L} = \prod_{i=1}^N$ pmf of Bernoulli(p), i.e.*

$$\mathcal{L} = p(t_i)^{x_i}(1-p(t_i))^{1-x_i},$$

where

$$p(t_i) \equiv Pr(X = 1 \mid T = t_i) = \frac{1}{1+e^{-\beta_0-\beta_1 t_i}}.$$

And therefore we have

$$\mathcal{L} = \prod_{i=1}^{N} (\frac{1}{1+e^{-\beta_0-\beta_1 t_i}})^{x_i} (\frac{1}{1+e^{\beta_0+\beta_1 t_i}})^{1-x_i},$$

$$\underbrace{log\mathcal{L}}_{\equiv \ell} = \sum_{i=1}^{N} [-x_i \ log(1+e^{-\beta_0-\beta_1 t_i})] + \sum_{i=1}^{N} [-(1-x_i) \ log(1+e^{\beta_0+\beta_1 t_i})].$$

Hence $log\mathcal{L} \equiv \ell$ is the log likelihood, namely, log(pdf of $\{t_i; x_i\}_{i=1}^N$, given β_0, β_1).

With likelihood at hand, we now start to apply sampling on the posterior, i.e. we apply MCMC sampling on the posterior.

Step I. Compute posterior

We need to compute the posterior first.

Remark 1.5.2. *Recall the Bayes rule,*

$$P(model \mid data) = \frac{P(data \mid model)P(model)}{P(data)},$$

$$\Rightarrow \pi(unknowns \mid data) = \frac{likelihood \times prior}{pdf(data)}$$

such that log posterior can be written as follows:

$$\begin{aligned} log\ posterior = log\ likelihood + log\ (prior\ of\ \beta_0) \\ + log\ (prior\ of\ \beta_1) - log\ (pdf(data)), \end{aligned}$$

here log posterior is the function of β_0, β_1,, i.e. $log(\pi(\beta_0, \beta_1 \mid data))$, and we have

$$\pi(\beta_0, \beta_1 \mid data) = \frac{likelihood \times prior}{pdf(data)},$$

the log posterior probability function, given the data.

Note that since $\log(pdf(data))$ is NOT a function of model parameters β_0, β_1, it can be considered to be a constant with respect to β_0, β_1. Recall in MH algorithm, we have the acceptance ratio to be defined as the ratio between posteriors, i.e. in one iteration, for the proposed value β_i^*, and the current value β_i^c , the acceptance ratio is shown as follows:

$$Acceptance\ ratio\ =\ min\{\frac{\pi(\beta_0^*, \beta_1^* \mid data)}{\pi(\beta_0^c, \beta_1^c \mid data)} \times \ldots, 1\}$$

the constant $pdf(data)$ in the posterior, both in the nominator and denominator, will be cancelled in the calculation, which again reinforces one of the advantages MCMC technique - no need for computing the pdf of data.

Step II. Decide prior $\pi_0(\beta_0), \pi_0(\beta_1)$

Now we need to consider the prior of model parameters.

Definition 1.5.3. *set-up There is a simple setup for the prior:*

$$\pi_0(\beta_0) \sim Uniform[a, b]$$

$$\pi_0(\beta_1) \sim Uniform[c, d]$$

Another example of a prior widely chosen is the Normal prior, namely the prior of β_0, β_1

$$\pi_0(\beta_0) = \frac{1}{\sqrt{2\pi\sigma_0^2}} \exp[\frac{-(-\beta_0 - \beta_{0(sp)})^2}{2\sigma_0^2}]$$

$$\pi_0(\beta_1) = \frac{1}{\sqrt{2\pi\sigma_1^2}} \exp[\frac{-(-\beta_1 - \beta_{1(sp)})^2}{2\sigma_1^2}]$$

Here our Normal priors are centred on some β_0, β_1 values, and variances σ_0^2, σ_1^2 are chosen by us.

There are a lot of types of priors and various studies in terms of prior chosen (Bernardo and Smith, 1994; Gelman, 2006; Gelman et al., 2008; Robert and Casella, 2004; Banerjee et al., 2015). We will use simple priors such as a uniform prior and Normal prior as we defined above, in this initial setup of our implementation on O-ring data. After setting up the posterior and the priors, we then begin to do the sampling by MCMC.

Step III. Sampling by MCMC

We consider the Random Walk Metropolis-Hastings first. To apply Random Walk MH to our O-ring data, we illustrate the detailed steps as follows:

1. Choose a seed value of β_1 and a seed value of β_0.

   Recall:

   $$Pr(X = 1 \mid T) = \frac{1}{1 + \exp(-\beta_0 - \beta_1 T)} = \frac{1}{1 + e^{-\beta_0} e^{-\beta_1 T}}$$

   Since we don't want the term $e^{\beta_1 T}$ to be too large or too small, we then suggest that the β_1 be the order of $\frac{1}{T}$ - the given O-ring data set the temperature is of the order of 50 °F, therefore we suggest β_1 to be the order of $\frac{1}{50}$ (or $\frac{1}{10}$ maybe).

2. Propose the 1st iteration, $\beta_1^{(*1)}$, $\beta_0^{(*1)}$. That is,

   $$\beta_1^{(*1)} \sim Normal(\beta_1^{(0)}, \sigma_a^2)$$
   $$\beta_0^{(*1)} \sim Normal(\beta_0^{(0)}, \sigma_b^2),$$

   here σ_a^2, σ_b^2 are the variances we choose.

   We need to decide if we will accept $(\beta_0^{(*1)}, \beta_1^{(*1)})$ or not, depend on the acceptance ratio α:

   $$\alpha = log(\pi(\beta_1^{(*1)}, \beta_0^{(*1)} \mid data)) - log(\pi(\beta_1^{(0)}, \beta_0^{(0)} \mid data)) \geq log(u),$$

   here uniform random number $u \sim U[0, 1]$.

   If the inequality holds, then we have

   $$\beta_0^{(1)} = \beta_0^{(*1)};$$

   if not, β_0 remains the same value, i.e.

   $$\beta_0^{(1)} = \beta_0^{(0)}$$

   And we apply the same process for generating $\beta_1^{(*1)}$.

3. Repeat 2 in the next iteration, and keep doing it!

Results of MCMC implementation

In this section we show the results of the application on O-ring data. We generated our own Python code to implement MCMC, with four types of MH algorithms applied.

Random Walk To start with, we apply the uniform prior, which normally shows little information about the parameters, namely:

$$\pi_0(\beta_0) \sim U[a,b], \pi_0(\beta_1) \sim U[c,d]$$

for some real a, b, c and d.

Consider the log posterior, recall that

$$\begin{aligned} log\ posterior = log\ likelihood + log\ (prior\ of\ \beta_0) \\ + log\ (prior\ of\ \beta_1) - log\ (pdf\ of\ data), \end{aligned}$$

the log of uniform priors β_0 and β_1, also the log of pdf of data, can be scaled in the calculation - in this case, the log posterior will simply be the log likelihood of β_0 and β_1. We generate 500,000 iterations with the seed value $\beta_{0(seed)} = 1, \beta_{1(seed)} = -\frac{1}{50}$ and the Normal proposals with pre-chosen variances, namely

$$\beta_0^{(*)} \sim Normal(\beta_0^{(c)}, \sigma_0^2), \beta_1^{(*)} \sim Normal(\beta_1^{(c)}, \sigma_1^2).$$

The traces of MCMC and distributions of β_0 and β_1 are shown below.

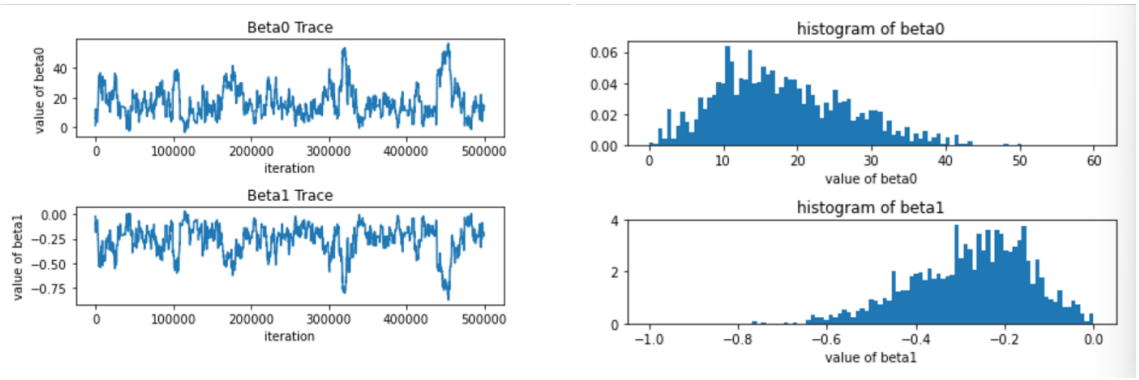


Figure 1.1: Using MCMC with a uniform prior along with a Normal proposal to generate β_0 and β_1. The top plots indicate the values and histogram of β_0 with proposal $\beta_0^{(*)} \sim N(\beta^{(c)}, 4^2)$, and the bottom plots show the values and histogram of β_1 with proposal $\beta_1^{(*)} \sim N(\beta^{(c)}, 4^2)$

For the case in Figure 1.1, we choose jump scale $\sigma_0^2 = \sigma_1^2$ for β_0 and β_1. While in the real world, in case there are many parameters that need to be considered, it will be a more reasonable choice to set up the individual jump scale, as a factor multiple to the parameter. Figure 1.2 shows the results of MCMC with updated choice on

variance σ, the MCMC traces improve the convergence and the histograms present a nice bell-shape.

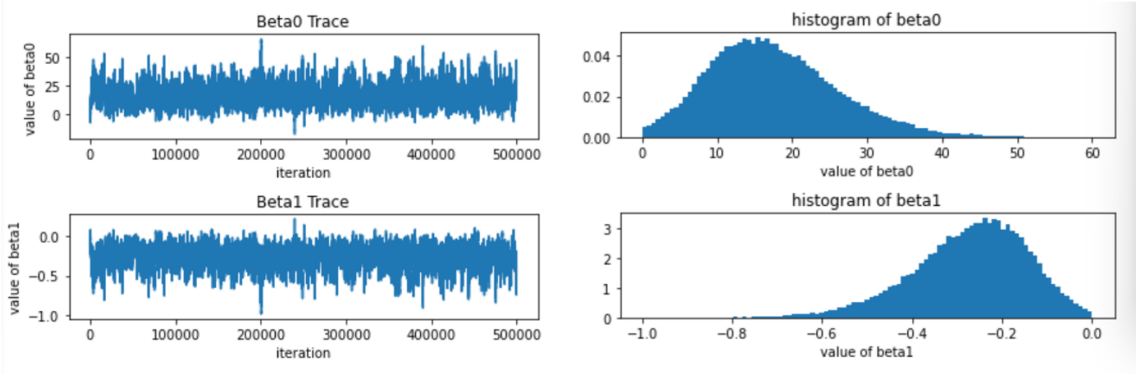


Figure 1.2: Using MCMC with a uniform prior along with a Normal proposal to generate β_0 and β_1. The top plots indicate the values and histogram of β_0 with proposal $\beta_0^{(*)} \sim N(\beta_0^{(c)}, (2\times 1)^2)$ and the bottom plots show the values and histogram of β_1 with proposal $\beta_1^{(*)} \sim N(\beta_1^{(c)}, (2 \times \frac{1}{50})^2)$

Apart from the uniform prior, we can use the Normal prior:

$$\pi_0(\beta_0) \sim Normal(\mu_0, {s_0}^2), \pi_0(\beta_1) \sim Normal(\mu_1, {s_1}^2)$$

The traces of 500,000 iterations are shown in Figure 1.3, for the Normal prior $\mu_0 = \mu_1 = 0, s_0 = s_1 = 2$.

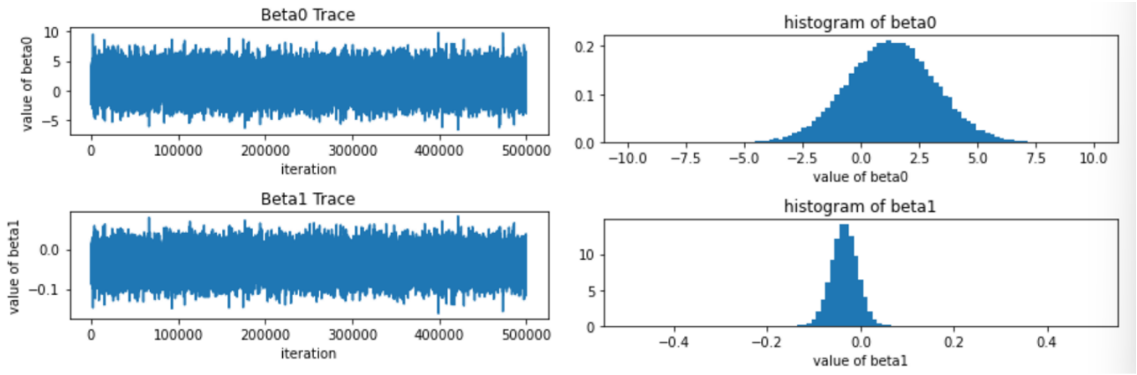


Figure 1.3: Using MCMC with Normal prior along with a Normal proposal to generate β_0 and β_1. The top plots indicate the values and histogram of β_0 with proposal $\beta_0^{(*)} \sim N(\beta_0^{(c)}, (2\times 1)^2)$ and the bottom plots show the values and histogram of β_1 with proposal $\beta_1^{(*)} \sim N(\beta_1^{(c)}, (2 \times \frac{1}{50})^2)$

Immediately we see the problems from the β_0 trace - the β_0 trace shows a convergence between 0 and 5, which does not converge to, about 15, from our previous study. The reason is that when we are proposing from the Normal density, the support of the Normal density is the whole of the real line - we are suggesting that

our β_0 can take both positive values as well as negative values, similarly β_1 can take negative values as well as positive values. Also, the scale of the excursion is determined by the jump scale. In this case a very large jump scale, $\sigma = 2 \times \beta_{(seed)}$ for both β_0 and β_1, has been applied to the calculation -the rule of thumb is that the jump scale should be at least two orders of magnitude smaller than the typical value of the variable. Our β_0 has a value of the order of 10, so the jump scale should roughly have a value of about 0.1.

We then applied a smaller jump scale with $\sigma = 0.2 \times \beta_{(seed)}$ for both β_0 and β_1, results shown in Figure 1.4. Currently we have a prior that is centred at 0 and has a standard deviation of 2 - we have a prior that is too broad. Since our prior is centred at 0, we are forcing the algorithm to basically move only around 0. When our prior is wider, the prior will be less informative, and it will be less strong. The weaker prior will be easier to overcome. So even though our prior says that our value is centred at zero, the details in the calculation will overcome the prior. After we adjust the proposal σ to the seed values times a factor of 0.2, we can find out that the traces converge better than the previous attempt in Figure 1.3.

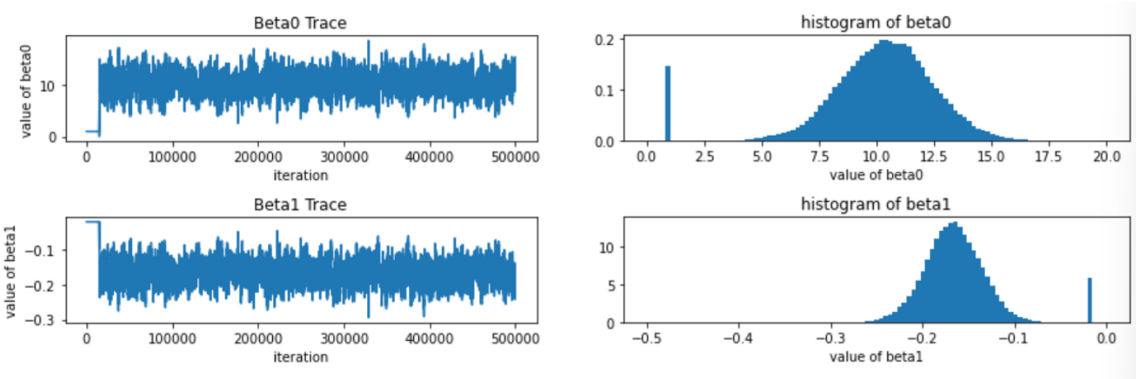


Figure 1.4: Using MCMC with Normal prior along with a Normal proposal to generate β_0 and β_1. The top plots indicate the values and histogram of β_0 with proposal $\beta_0^{(*)} \sim N(\beta_0^{(c)}, (0.2 \times 1)^2)$ and the bottom plots show the values and histogram of β_1 with proposal $\beta_1^{(*)} \sim N(\beta_1^{(c)}, (0.2 \times \frac{1}{50})^2)$

From the trace of β_0 in Figure 1.4, we can find out that our β_0 still has a tendency to be positive, namely, converges to positive - we have more sample values that are positive than negative. Suppose we did not have any idea as to what the value of β_0 is, then this kind of trace that we have run, even using this kind of wide and misplaced prior, we can still see that the next run will aim to achieve a positive β_0 - we can always analyse and take information from our previous traces. Bayes rule allows us to sequentially update our results. And now, we have a bit more information about the unknowns, after we had these previous attempts.

Vanilla MCMC After the random walk implementation in the previous section, we now have more information about our parameters, therefore we then use a truncated normal proposal instead, only taking positive values of β_0 and negative values

for β_1.

Definition 1.5.4. *When the proposal distribution is not Normal, the MCMC is not a random walk anymore - in this case, it is called a Vanilla MCMC - the standard MCMC algorithms, more advanced MCMC methods such as Hamiltonian Monte Carlo, Slice Sampling, are built on top of Vanilla MCMC to improve efficiency and convergence.*

Notes that the acceptance ratio will not be simply the ratio between the posterior of proposed values and the posterior of current values, since in the Random Walk we have a Normal distribution as our proposal density q, which is symmetric. In this case, the error functions of the truncated Normal proposal q will not be cancelled (Wilkinson, 2012).

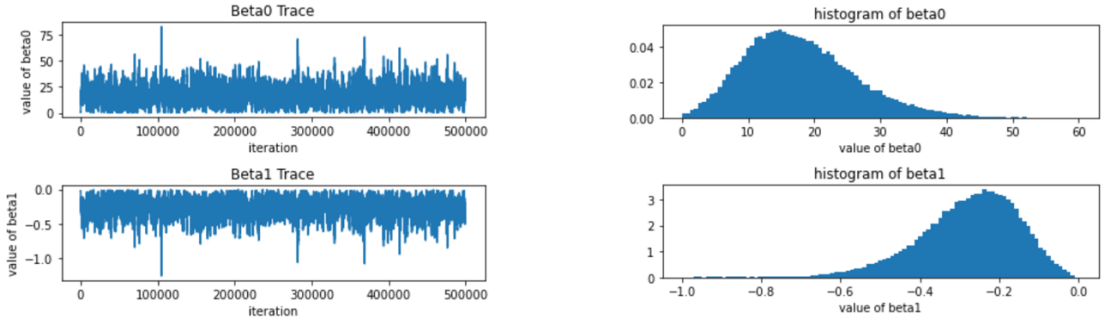


Figure 1.5: Using MCMC with Normal prior along with Truncated Normal proposal to generate β_0 and β_1. The top plots indicate the values and histogram of β_0 with proposal $\beta_0^{(*)} \sim TN(\beta_0^{(c)}, (2 \times 1)^2)$, and the bottom plots show the values and histogram of β_1 with proposal $\beta_1^{(*)} \sim TN(\beta_1^{(c)}, (2 \times \frac{1}{50})^2)$.

We slightly tweak the jump scales of β_0 and we have:

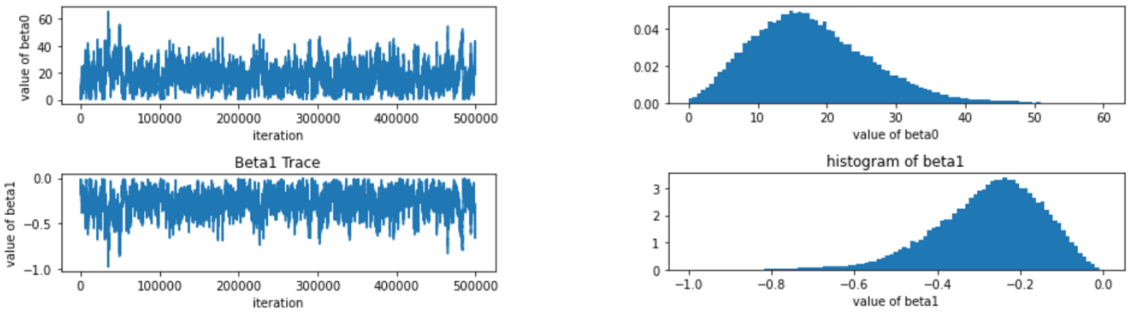


Figure 1.6: Using MCMC with Normal prior along with Truncated Normal proposal to generate β_0 and β_1. The top left and top right plots indicate the values and histogram of β_0 with proposal $\beta_0^{(*)} \sim TN(\beta_0^{(c)}, (1 \times 1)^2)$, and the bottom left and bottom right plots show the values and histogram of β_1 with proposal $\beta_1^{(*)} \sim TN(\beta_1^{(c)}, (2 \times \frac{1}{50})^2)$.

In this section, we can see the traces fit the typical truncated Normal property - we only sample the positive β_0, similarly we have only negative β_1. The β_0 still can be found to converge to about 15, and β_1 converges to -0.2 roughly. In general, since we keep updating our information about parameters from the traces, the truncated Normal proposal shows better convergence in the traces and the nice bell-shape are presented in the histograms of β_0 and β_1.

Independent samplers

Definition 1.5.5. *Independent sampler works similarly to vanilla MCMC, while the mean of the proposal density is no longer the value of the unknown in the current iteration, instead will be a fixed mean we have chosen.*

Compared with the Normal proposal, in the calculation of independent sampler, the proposal density computed at the proposed value in the acceptance ratio, has nothing to do with the current value in the iteration - immediately it can be seen that one disadvantage of an independent sampler is the poor proposal will not easily drive us to a good result, since we do not know how good our chosen proposal mean will be, unless we know the approximate value of the unknown, which is often difficult in the real world. The quality of the routine depends on the choice of the proposal distribution, and its coverage of the target distribution.

Figure 1.7 shows the on example of an independent sampler we implemented. The traces of unknowns we gain vary a lot when we are learning different fixed proposal mean values $\beta_0 = \mu_0^{(*)}, \beta_1 = \mu_1^{(*)}$ and proposal standard deviations of β_0, β_1, to be $\sigma_0^{(*)}$ and $\sigma_1^{(*)}$ respectively. The seed values $\beta_{0(seed)} = 15, \beta_{1(seed)} = -\frac{1}{50}$ have applied.

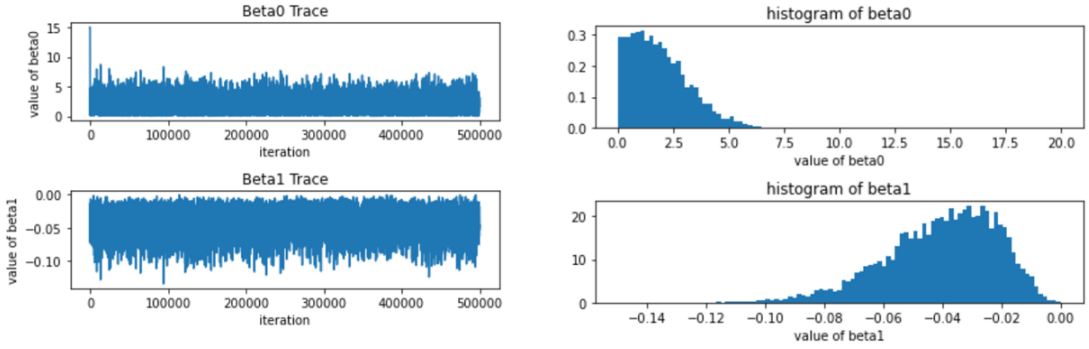


Figure 1.7: Using MCMC for the Normal prior $\mu_0 = 10, \mu_1 = 0, \sigma_0 = \sigma_1 = 2^2$, with fixed proposed truncated Normal mean and standard deviation values $\mu_0^{(*)} = \mu_1^{(*)} = 0$ and $\sigma_0^{(*)} = 15$, $\sigma_1^{(*)} = \frac{1}{50}$. The top plots indicate the values and histogram of $\beta_0^{(*)}$ and the bottom plots show the values and histogram $\beta_1^{(*)}$

The traces are not as good compared to the implementation on Random Walk and Vanilla MCMC previously. In general, we can see that independent samplers show worse performance in our data if we set our fixed proposal mean to be too

far away from the real value of the parameters. Also, the wider jump scales might be the reason why we have difficulties converging the values as we are using RW or Vanilla MCMC - normally, we do not prefer using independent samplers to learn the unknowns.

Gibbs sampling Gibbs sampling gives the idea that when we have more than one random variable, we are trying to learn. Consider a vector-valued y given the data, which is in p dimensions. We can update each of the components of y given the data at the same time, like our previous implementation, or we can update the first component of y only and not update everything else. Then we have another block of this t iteration, in this second block we update the second element of y, namely, we are updating y_2. We write down the joint of y_1, y_2, y_3 and so on till y_p given the data - for this joint posterior, we will compute the y_2 at the proposed under the current and the y_1 that we will input will be the current y_1 that we have **already updated** in the previous block of this same t iteration.

Algorithm 2: Algorithm: Gibbs Sampling (Robert and Casella, 2004)

Given $(y_1^{(t)}, y_2^{(t)}, ..., y_p^{(t)})$, we simulate:

1. $Y_1^{(t+1)} \sim g_1(y_1|y_2^{(t)}, ..., y_p^{(t)})$,
2. $Y_2^{(t+1)} \sim g_2(y_2|y_1^{(t+1)}, y_3^{(t)}, ..., y_p^{(t)})$,

   $\vdots$
3. $Y_p^{(t+1)} \sim g_p(y_p|y_1^{(t+1)}, y_2^{(t+1)}, ..., y_{p-1}^{(t+1)})$

And the acceptance ratio in Gibbs has been proven to always equal 1 (Robert and Casella, 2004).

Consider MH, the way we learn the unknowns θ with p dimensions, is that we have only one updating block inside the whole step, so we are in the t-th iteration, we have only one block and we update θ_1, θ_2, θ_3 ,..., θ_p all of them in that same block - when we are updating all of them at the same go, we are not really using the correlation that is intrinsic between θ_1 and θ_2 when we are updating θ_2. Now suppose we want to bring in information on how to update θ_2 by what the updated value of θ_1 is, we want to use the correlation between θ_1 and θ_2 in a more effective way - we want to update θ_2 by using the correlation between θ_2 and θ_1, we then update θ_2 only after we have finished choosing the current value of θ_1. One example is that there is a strong correlation between the parameters we want to learn. Consider a function we do not know how to learn, we can take the domain of the function and partition the domain into little bits, then we have the first partition, the second partition, the third partition and so on. The value of the function over the first partition is θ_1, the value of the function of the second partition is θ_2, and so on till θ_p - now this θ_1 is very strongly correlated with θ_2, since they are values of the same function, θ_1 and θ_2, and further to θ_p they are all very strongly inter-correlated - in such a case, we will want to apply correlation to affect the updating.

For the comparison between a Gibbs sampling method and other cases in MH

algorithm, Gibbs sampling derives its conditional distributions from the true distribution f, whereas a MH, at best, is based on an approximation of this distribution f. In particular, it can be seen that Gibbs sampling methods are more straightforward than MH methods, since they cannot have a "bad" choice of the instrumental distribution and hence they could avoid useless simulations (rejections). On the other hand, we can see the drawback of Gibbs - in reality, we do not often know the marginal distribution of our parameter, so that we can make Gibbs derive its conditional distributions from the marginal. In this case, Gibbs sampling is applicable if we find out that the conditional distribution of the parameters is known and easier to sample from, when the joint distribution is not known or is difficult to sample from directly.

Metropolis within Gibbs Ghirmai (2015) suggested that although the Gibbs sampler requires the availability of full conditional probability density functions of all the parameters of interest, the full conditional pdfs of all the parameters of interest might not all be available. In such cases, the MH method can be incorporated within a Gibbs sampler to draw samples from the parameters.

Definition 1.5.6. *Consider we have unknowns that have more than one dimension to learn, $\theta = (\theta_1, \theta_2, ..., \theta_p)$, we will update θ_1 by proposing $\theta_1 \sim$ Proposal density 1, given $\theta_2, ..., \theta_p$, i.e. acceptance ratio will be*

$$\frac{posterior(\theta_1^*, \theta_2, ...\theta_p|data)}{posterior(\theta_1, \theta_2, ...\theta_p|data)} / \frac{proposal\ density\ 1(\theta_1^*|\theta_2, ..., \theta_p)}{proposal\ density\ 1(\theta_1|\theta_2, ...\theta_p)}$$

Let $\theta_1^{current}$ is θ_1^c. Then in next block, update θ_2 by proposing $\theta_2 \sim$ Proposal density 2, given $\theta_1^*, ..., \theta_p$, i.e. acceptance ratio will be

$$\frac{posterior(\theta_1^c, \theta_2^*, \theta_3, ...\theta_p|data)}{posterior(\theta_1^c, \theta_2, ...\theta_p|data)} / \frac{proposal\ density\ 2(\theta_2^*|\theta_1^c, \theta_3, ..., \theta_p)}{proposal\ density\ 2(\theta_2|\theta_1^c, \theta_3, ...\theta_p)}$$

In our O-ring case, we can implement MCMC by using Metropolis within Gibbs - we update β_0 first, then update β_1 within an iteration. With a Normal prior,

$$\pi_0(\beta_0) \sim N(0, 2^2), \pi_0(\beta_1) \sim N(0, 2^2),$$

we adjust our implementation, with better seed values and priors of β_0 and β_1, since we know that β_0 is centred about 15 from our previous study, we set $\beta_0^{(seed)} = 15$ and similarly $\beta_1^{(seed)} = -\frac{1}{50}$. We also consider more reasonable jump scales for our iteration, i.e.

$$\sigma_0 = 0.2 * \beta_0^{(seed)}, \sigma_1 = 0.2 * \beta_1^{(seed)}$$

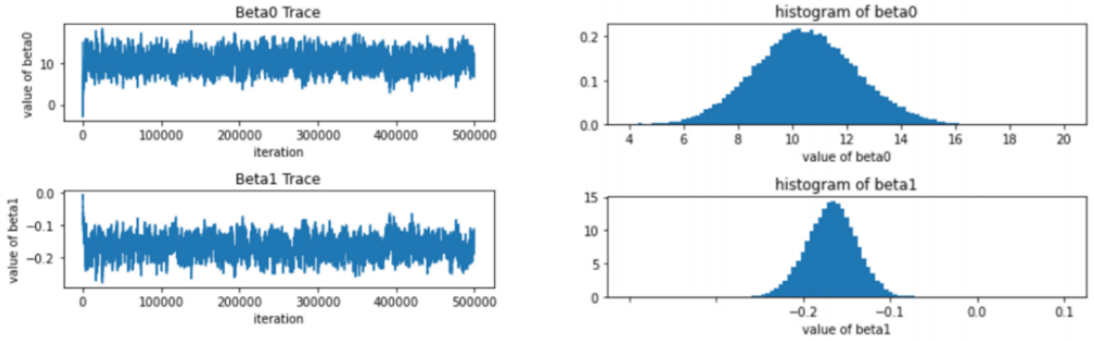


Figure 1.8: MH with Gibbs with a Normal proposal. The top plots indicate the values and histogram of $\beta_0^{(*)}$ with proposal jump scale $\sigma_0 = 0.2 \times \beta_0^{(seed)}$, and the bottom plots show the values and histogram $\beta_1^{(*)}$ with proposal jump scale $\sigma_1 = 0.2 \times \beta_1^{(seed)}$

From Figure 1.8 above, we can see both β_0 and β_1 traces show a good convergence to their real value. For a truncated Normal proposal, Metropolis within Gibbs also works well, shown in Figure 1.9, as follows:

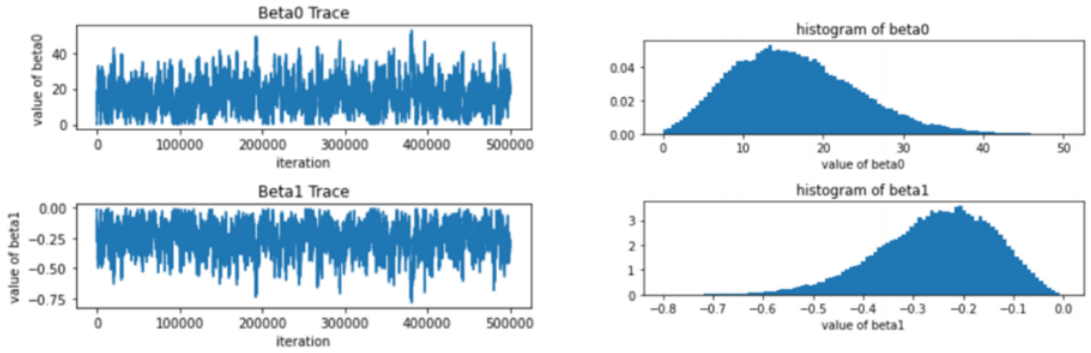


Figure 1.9: MH with Gibbs with Truncated Normal proposal. The top plots indicate the values and histogram of $\beta_0^{(*)}$, and the bottom plots are values and histogram of $\beta_1^{(*)}$

Indeed, there is not much point in implementing O-ring data by Gibbs since we only have two variables to learn. Consider in the case if there are 100 unknowns and we make that partition of the 30 and 70 variables in each partition, we believe that (from some information) 30 variables share some kind of commonality in their correlation and the 70 of them also share a commonality in their correlation, but the 30 variables and the 70 variables are disparate in their correlations - we are actually using the correlation structure to our advantage when making the updating. Then we will get better results with Metropolis within Gibbs than with MH alone.

1.5.4 Discussion

In a real-world situation, we will not usually know the value of unknowns - we might choose the priors mistakenly s.t. that parameters become far away from what they really should be. In this O-ring case, our β_0 and β_1, after learning a series of different MCMC approaches, at the end we find that the β_0 is centred about 15, which means that our Python code implementation is quite resilient to strange priors that we are inputting by hand - a strange prior means it might cause MCMC very difficult to converge. We then tweak the jump scale a little bit, by looking at the current traces we have, then we improve the traces and get convergence:

- An increasing trend in the trace - slightly increase the jump scale.
- A decreasing trend in the trace - slightly decrease the jump scale.
- No showing an increase or a decrease, but not good - slightly decrease the jump scale or make a better prior, i.e. change the prior mean, which is not recommended. We would like to tweak the jump scale and see if we are getting slightly better or slightly worse. For example, making a jump scale smaller and running over a longer period of iterations usually gives better results than a bigger jump scale and shorter period of running time.
- Large variety in trace - reduce the jump scale.

We always want to look at the traces and make a tweak for a better trace, then sequentially run the chain. Again, this indeed shows the beauty of the Bayes rule - allows us to sequentially update our results, and have better learning of the unknowns as we constantly improve.

1.5.5 Highest posterior density regions applied to O-ring Data

In the previous session, we had a discussion on the motivation for performing Bayesian inference, one of the reasons is that we have the joint posterior probability of all the unknowns, from the Bayes rule. In our O-ring case, we have the joint of β_0 and β_1. What we want is the marginal posterior probability density function of unknowns given the data, so that we can examine the probability for unknowns to lie between some values - we now actually have the marginal posterior pdf of each unknown given the data when we are using MCMC, namely, the traces we plotted. MCMC gives the histograms of the unknowns given the data, and the histogram looks like a bell-shaped function; sometimes it might turn out to be multi-modal. We can then find out the highest probability density credible region to encapsulate the most likely values of unknowns given the data, as we discuss now.

The ideal summary of θ is an interval (or region) with a certain probability of containing θ. Note that a classical (frequentist) confidence interval does **NOT** exactly have this interpretation: a confidence interval states the probability that a range of values contains the true value if the experiment were repeated many times.

Definition 1.5.7. *A credible interval (or in general, a credible set) is the Bayesian analogue of a confidence interval. A* $100(1-\alpha)\%$ *credible set* $\mathcal{C}$ *is a subset of* Θ *such that*

$$\int_{\mathcal{C}} \pi(\theta \mid X) d\theta = 1 - \alpha$$

and if the parameter space Θ *is discrete, a sum replaces the integral.*

We can similarly find the equal-tailed credible interval - the equal-tail credible interval approach is ideal when the posterior distribution is symmetric. When $\pi(\theta|x)$ is skewed, a better approach is to create our interval of θ-values having the highest posterior density.

Definition 1.5.8. *A* $100(1-\alpha)\%$ *Highest Posterior Density (HPD) region for* θ *is a subset* $\mathcal{C} \in \Theta$ *defined by*

$$\mathcal{C} = \{\theta : \pi(\theta|x) \geq k\}$$

where k *is the largest number such that*

$$\int_{\theta:\pi(\theta|x)\geq k} \pi(\theta|x) d\theta = 1 - \alpha$$

The value k can be thought of as a horizontal line placed over the posterior density whose intersection(s) with the posterior define regions with probability $1-\alpha$. In other words, the highest density region is a region $\mathcal{C}$ that contains $100(1-\alpha)\%$ of the posterior distribution and has the property that for all $\theta_1 \in \mathcal{C}$ and $\theta_2 \notin \mathcal{C}$,

$$\pi(\theta_1|x) \geq \pi(\theta_2|x).$$

The HPD region is defined so that inside the region, the posterior density is everywhere higher than outside the region (Kruschke, 2014). It indicates which points of a distribution are most credible, and which cover most of the distribution - the HPD region is the region with $100(1-\alpha)\%$ probability of containing θ with the shortest total length. Note that the HPD region is not invariant under reparametrisation (Bernardo, 2005). For unimodal symmetric distributions, the HPD and credible interval are the same. Credible intervals are generally easier to compute (particularly if we are doing so by placing $\alpha/2$ in each tail) and are invariant to transformations of the parameters. HPDs are more complex, and there is a potential lack of invariance under parameter transformation. However, they are more meaningful for multinomial distributions.

We applied the HPD concept to our MCMC implementation, since we already have the marginal posterior pdf of β_0 and the marginal posterior pdf of β_1, given the data. The table containing all HPD regions computed based on all traces in this chapter has been included in the Reference Table in the Appendix A.

In Appendix C I have included a short discussion of Gaussian Process based learning of random functions. This becomes relevant in my consideration of future

work, following my learning of random graphs. For example when the learning of the relationship between pre-transplant variables and a severity score variable becomes relevant - as in the project, the early stage of which is my work, presented in Chapter 3 - the sought relationship can be modelled as a random function that is in turn considered modelled with a Gaussian Process.

1.6 Summary

In this chapter, I have spoken about the fundamental concepts of random graphs - such as random geometric and soft random geometric graphs - that I will be using throughout this thesis. I also spoke about the need for an impactful threshold that we can impose on such graphs, to bring out the geometric nature of these random graphs. Additionally, I have spoken about the idea on computing statistical distances and divergences between posterior probabilities of realisations of such a graph-valued variable, given two distinct and (generally) disparately-long multivariate datasets.

All such graph learning requires me to

- learn the correlation of the presented multivariate dataset, unless I compute this correlation as estimated using the data, and
- learn the edge variable between two nodes of the graph, given the correlation between the two random variables that are attached to each of the nodes.

This learning is undertaken using Bayes rule at its core, but it then requires me to learn the sought value of the unknown, given the data, using the posterior probability of thus variable that has resulted from using Bayes rule. In other words, I need to undertake Bayesian inference of the value of the variable, using its posterior, given the data. I undertake such inference using sample-based techniques, such as Rejection sampling (from univariate, closed-form posterior) or MCMC (when the available density is a joint posterior density of multiple variables, or when the full posterior is not known). I dedicate the latter part of this thesis to discussions of these techniques, including illustration on a small real dataset.

Chapter 2

Theoretical Background to the Learning of Random Geometric Graphs

2.1 Introduction

As introduced briefly in the previous chapter, my Ph.D project involves the learning of the realisations of the random graph variable, given a multivariate data set, to then compute a statistical distance/divergence between the posterior of a pair of realisations of two such graphs, learnt given the respective data set. This inter-graph distance then tells us about the strength of the effect of the experimental conditions (and/or time) - that have brought about the difference between the values of the two random graph variables.

This random graph variable carries information about the correlation between each of the pairs of those random variables, observations of each of which populate the respective column of the available multivariate dataset. Each of the nodes of the sought random graph is associated with one of these observed random variables. The inter-nodal edge is driven by the correlation between the observables that are attached to the two considered nodes. These graph edges are learnt within a Bayesian approach, which allows the learning of the edge - and thereby the whole graph - with uncertainty, given the correlation structure of the data. Thus, the posterior probability of this graph-valued random variable can be computed, conditional on the correlation matrix, any element of which is the absolute correlation between a pair of the observed variables. Consequently, we can learn the inter-graph distance as the distance between the posterior probabilities of a pair of random graphs realised given two given datasets.

In this chapter, I discuss details of the method of graph learning, and as I will explain below, the edge posterior is in fact closed-form, given the correlation between the pair of observables that sit at the two nodes that straddle this edge. We are then able to perform the sampling of the edge variable from its known closed-form posterior, using rejection sampling. I perform this rejection sampling multiple

times, and the relative frequency of the sampled values of an edge variable then approximates the probability with which this edge exists, given the correlation of the given dataset. If this probability of existence of this edge exceeds a pre-chosen cut-off probability, this edge exists in the graph of this dataset, otherwise not. Thus the graph is constructed.

The methodology of the graph learning that I report in this chapter, is existing and under review; I refer to this below. While such method development is not my own doctoral work, its implementation is. Indeed, my doctoral work is an endeavour in making multiple new applications of a new method for learning random geometric graphs, and computing inter-graph distances, where relevant. One methodological contribution of my doctoral work is an attempt at the identification of the optimal cut-off probability, at which any edge of the sought graph exists. In this chapter, I discuss such methodological details, and will discuss three distinct applications that I have undertaken - using such methodology - in the following three chapters, using the sampling-based Bayesian inference that I discussed in the previous chapter.

2.2 Constructing an RGG given a dataset

As I introduced in the previous chapter, in a Random Geometric Graph (RGG) (Giles et al., 2016; Penrose, 2016; Chakrabarty et al., 2023), edges exist between pairs of points generated from a Poisson Point process, if the inter-point distance is less than a cut-off distance. A Soft Random Geometric Graph (SRGG) (Penrose, 2016; Dettmann, 2018) generalises this model by introducing uncertainty on the edges, such that (s.t.) the probability for the edge to exist between a pair of points is given by a function (called the connection function) of the edge length.

Following Chakrabarty (2023); Chakrabarty et al. (2023); Zhang et al. (2024), I learn the random graph in a probabilistic metric space, which means that to any pair of points in this space - and thereby, any two nodes of the graph - we can assign a probability distribution defined over a non-negative support. This is just like we can assign a non-negative distance to any two points in any metric space. That means that any edge that joins two nodes, is then a cumulative distribution function (cdf) over positive support. As I am going to learn a graph variable in such a probabilistic metric space, any such edge length - which is the distance between the relevant nodal pair - is a cdf. Also, since the graph that I seek is an RGG, if this inter-nodal distance falls short of a cut-off probability, it will be assigned a value of 1, i.e. will exist. Otherwise this edge variable will not exist, i.e. will be assigned the value 0. But the inter-nodal distance is the complement to the inter-nodal affinity. So if the inter-nodal affinity exceeds the cut-off, then the edge exists, otherwise not. In the random graph variable that we learn, the inter-nodal affinity is given by the edge posterior that is conditional on the correlation between the random variables that are attached to the two considered nodes. I will learn each edge variable by sampling from this closed-form edge posterior, to laern the full graph variable.

2.2.1 Formal exposition

Consider a multivariate dataset, $\mathbf{D}$ built with n observations of each of p observables, i.e. $\mathbf{D} = [x_{ij}]_{i=1;j=1}^{n;p}$. With this data, we will construct an RGG in a probabilistic metric space, as follows.

- The sought random graph is defined on the vertex set $\boldsymbol{V} = \{1, 2, ...p\}$, with the random variable X_i attached to the i-th node in the graph, $\forall i \in \boldsymbol{V}$, s.t.
- G_{ij} exists between the i-th and j-th nodes (i.e. $G_{ij} = 1$) if the inter-nodal distance $d(x_i, x_j)$ falls below a cut-off τ.
- Since the inter-nodal distance for graphs drawn in probabilistic metric spaces is a cdf, this cut-off is a probability, i.e. $\tau \in [0, 1]$.
- The probability of a random (edge) variable G_{ij}, conditional on correlation between X_i and X_j, is given by the affinity between the i-th and j-th nodes, where the inter-nodal affinity is complementary to the inter-nodal distance $d(x_i, x_j)$.
- Here $X_i \in \mathcal{X}$, $\forall i \in \boldsymbol{V}$.

2.2.2 Disparity between two nodes of the graph

Intuition: The distance $d(X_i, X_j)$ between points X_i and X_j, is a cdf, since we draw the graph in a probabilistic metric space. Then the complementary inter-nodal affinity function is also a cdf. Here the affinity between the i-th and j-th nodes is given by the posterior probability of the edge G_{ij} conditional on the correlation $|\rho_{ij}|$ between X_i and X_j, (with all other random variables $\{X_k\}_{k\in\boldsymbol{v};k\neq i,k\neq j}$ held fixed). This edge posterior then informs on the probability of the parameter that measures the similarity - or lack of disparity - between the edge variable G_{ij}, and the absolute correlation $|\rho_{ij}|$. Then the edge posterior of G_{ij} given ρ_{ij} is equivalently the probability of the disparity variable $S_{ij} := G_{ij} - |\rho_{ij}|$, between X_i and X_j, s.t. when disparity is highest, the edge posterior probability is minimal, and when disparity is lowest (=0), the edge posterior is maximum. (Note above, that in the strictest sense, all relevant random variables other than X_i and X_j are held fixed when considering the correlation between X_i and X_j. Therefore, ρ_{ij} is the partial correlation. Difficulties with the computation of the partial correlation can lead us to work with the correlation $corr(X_i, X_j)$ instead. I will discuss partial correlations later, and proceed with the notation that $\rho_{ij} = corr(X_i, X_j)$.)

Formalisation: Disparity S_{ij} is the absolute difference between:

- connectivity between i-th and j-th nodes, where such connectivity can attain values of either 1 or 0,
- and the absolute correlation $|\rho_{ij}|$ between X_i and X_j, where the matrix of correlations between observable pairs in this dataset is $\boldsymbol{\Sigma}^{(C)} = [\rho_{ij}]$.

We define disparity $S_{ij} := |G_{ij} - |\rho_{ij}||$.

Definition 2.2.1. *We construct a Random Geometric Graph (RGG) variable upon the vertex set $i \in \boldsymbol{V} = \{1, 2, \ldots, p\}$, where this RGG is drawn in the probabilistic metric space or PMS $\{\mathcal{X}, d(\cdot, \cdot)\}$, given the dataset $\mathbf{D}$.*

To the i-th node of the RGG, we attach the random variable $X_i \in \mathcal{X}$ $\forall i \in \boldsymbol{V} = \{1, 2, \ldots, p\}$.

The distance function $d(x_i, x_j)$ between the two points $X_i \in \mathcal{X}$ and $X_j \in \mathcal{X}$, is a cdf with positive support. For us, this distance function of the PMS is the cdf $F_{S_{ij}}(s)$ of disparity variable $S_{ij} := |(G_{ij} - |\rho_{ij}|)| \in [0, 1]$, $\forall X_i, X_j \in \mathcal{X}$. Thus,

1. $\forall i, j \in \boldsymbol{V}, d(x_i, x_j) = 0 \equiv F_{S_{ij}}(s) = 0 \iff i = j$;
2. $d(x_i, x_j) = d(x_j, x_i)$, *i.e.* $F_{S_{ij}}(\cdot) = F_{S_{j,i}}(\cdot)$;
3. $d(x_i, x_j) + d(x_j, x_k) \leq d(x_i, x_k)$, *i.e.* $F_{S_{ij}}(\cdot) + F_{S_{j,k}}(\cdot) \leq F_{S_{i,k}}(\cdot)$.

Thus, in an RGG that is drawn in a probabilistic metric space (Menger, 1942; Schweizer and Sklar, 2011), the distance between any pairs of nodes of the graph variable is a probability distribution that is defined over a non-negative support.

2.2.3 Learning/estimation of the correlation matrix

We learn the edge variable G_{ij} of this RGG, conditional on the absolute correlation variable R_{ij}, s.t. we define the edge posterior probability $m(G_{ij} = g_{ij} | R_{ij} = |\rho_{ij}|)$, which will be discussed right after this section. We discuss the details of the correlation structure of the dataset $\mathbf{D}$ now.

Definition 2.2.2. *Let the correlation matrix of this data $\mathbf{D}$ be $\boldsymbol{\Sigma}^{(C)} = [\rho_{ij}]$, where $R_{ij} = |\rho_{ij}|$ is the absolute correlation between the random variables X_i and X_j, for $i, j = 1, \ldots, p$.*

We can learn each element of the correlation matrix $\boldsymbol{\Sigma}^{(C)}$, by applying sampling methods, such as Markov Chain Monte Carlo (MCMC), which I discussed in the previous chapter. If we can identify the posterior probability density of the correlation matrix $\boldsymbol{\Sigma}^{(C)}$ given the data $\mathbf{D}$ at hand, we could learn this correlation matrix by sampling from its posterior density. The simpler alternative is to estimate the correlation between the observables X_i and X_j in the given dataset, i.e. using the sample of n number of values of each of observables X_i and X_j, as included within the dataset, $\forall i < j; i, j \in \boldsymbol{V}$. This plug-in estimate of $corr(X_i, X_j)$ is of course only a sample estimate, and therefore, it gets closer to the true correlation, as the sample size increases, i.e. $n \to \infty$. For small n, it could be wrong to consider the correlation between these two observables to be given by such estimates.

Definition 2.2.3. *The unbiased estimate of the correlation between X_i and X_j, given the sample $\{x_{i,1}, \ldots, x_{i,n}\}$ and $\{x_{j,1}, \ldots, x_{j,n}\}$ of values of these two observables respectively, is denoted $\hat{\rho}_{ij}$, with*

$$\hat{\rho}_{ij} = \frac{1}{n-1} \sum_{k=1}^{n} \frac{(x_{i,k} - \bar{x}_i)}{s_i} \frac{(x_{j,k} - \bar{x}_j)}{s_j}.$$

Here the sample mean $\bar{x}_i = \sum_{k=1}^{n} x_{i,k}$ *and* $s_i = \sqrt{\frac{1}{n-1} \sum_{k=1}^{n} (x_{i,k} - \bar{x}_i)^2}$ *is the unbiased estimate of the standard deviation. Similarly* $\bar{x}_j$ *and* s_j *are defined,* $\forall i, j = 1, \ldots, p$.

Then the inter-column correlation matrix of the dataset $\mathbf{D}$, $\mathbf{\Sigma}^{(C)}$ *can be estimated as* $\hat{\mathbf{\Sigma}}^{(C)} = [\hat{\rho}_{ij}]$

2.2.4 Correlation and partial correlation

The correlation matrix is a fundamental tool in multivariate statistical analysis, offering several advantages that contribute to its widespread use in research (Hotelling, 1936; Anderson, 2003; Johnson and Wichern, 2019; Bozhilova et al., 2020). One of its primary strengths lies in its ease of computation, requiring only standardised pairwise covariances, which are straightforward to estimate even in large and complex datasets. As we discussed in the previous section, we learn RGG conditional on the correlation structure of the dataset. Another choice is to learn an edge variable, given the partial correlation matrix of the data, instead of the correlation matrix. This is because the correlation can be confounded by shared influences of variables other than the pair of variables that we are seeking the correlation between. On the other hand, partial correlation quantifies the relationship between two variables while considering the influence of the other variables fixed. This offers a refined view of the direct association between the considered pair of variables and has been explored in the context of linear regression and multivariate analysis. Kendall and Stuart (1979) provides foundational derivations of partial correlation in the context of linear models, and in recent decades, partial correlation has been widely applied in graphical models (Lauritzen, 1996; Whittaker, 2009). This has led to its integration into Gaussian graphical models, particularly in the construction of sparse networks using the inverse of the covariance matrix (also known as the precision matrix). These applications are prominent in high-dimensional data analysis, in fields such as genomics (de la Fuente et al., 2004; Schäfer and Strimmer, 2005) and neuroimaging (Marrelec et al., 2006).

The partial correlation ψ_{ij} between variables X_i and X_j is defined as follows:

Definition 2.2.4. *The partial correlation* ψ_{ij} *between observables* X_i *and* X_j *is computed using the estimated correlation matrix* $\mathbf{\Sigma}^{(C)}$. *Given the precision matrix* $\mathbf{\Theta} = [\theta_{ij}] \coloneqq (\mathbf{\Sigma}^{(C)})^{-1}$, *the partial correlation matrix* $\mathbf{\Psi} = [\psi_{ij}]$ *is defined as follows,*

$$\psi_{ij} = -\frac{\theta_{ij}}{\sqrt{\theta_{i,i}\theta_{j,j}}}, i \neq j, \text{ and we have } \psi_{i,i} = 1 \text{ for } i = j. \tag{2.2.1}$$

here $i < j$, $i, j \in \boldsymbol{V}$.

The partial correlation matrix $\mathbf{\Psi} = [\psi_{ij}]$ of the given data $\mathbf{D}$, is a random variable. While partial correlations have the theoretical advantages in isolating

direct effects between variables, they require inversion of the covariance matrix, which makes it infeasible - or rather computationally intensive, in high-dimensional settings. We appreciate both the partial correlation matrix $\mathbf{\Psi}$, and the correlation matrix $\mathbf{\Sigma}^{(C)}$, when we learn the graph given the correlation structure of the data, conditioned on which, the graph variable is learnt. Therefore in my thesis, I would define the posterior of the graph variable conditioned on $\mathbf{\Psi}$,or $\mathbf{\Sigma}^{(C)}$ (in the high-dimensional case, see Chapter 5). In fact, in the case I learn the RGG conditioned on the partial correlation matrix $\mathbf{\Psi} = [\psi_{ij}]$, where the partial correlation variable between X_i and X_j is Ψ_{ij} that takes the value ψ_{ij}, in all discussions regarding the graph learning, we only need to substitute R_{ij} with Ψ_{ij}; ρ_{ij} with ψ_{ij}; and $\mathbf{\Psi}$ with $\mathbf{\Sigma}^{(C)}$.

2.2.5 Edge posterior given correlation & inter-nodal distance

To define the graph posterior give $\mathbf{\Sigma}^{(C)}$, we would invoke Bayes rule, and *first* input into it, the likelihood of the model parameters in the observations, i.e. the probability density of the observable $\mathbf{\Sigma}^{(C)}$ given the model parameters. Here, said model parameters are the edge variables that build the graph. Thereafter, we will input into Bayes rule, the prior probability of each edge variable. So we begin by understanding what the density of $\mathbf{\Sigma}^{(C)}$ is, given the relevant edge variable.

Recall that our RGG variable is drawn in a PMS as we defined in Definition 2.2.1 and therefore the posterior of the edge variable G_{ij} is the cdf $F_{S_{ij}}(s) \equiv Pr(S_{ij} \leq s)$. Here G_{ij} is a binary random variable, taking the value $g_{ij} = 1$ if the edge between the i-th and j-th nodes; otherwise $g_{ij} = 0$. Recall that in the definition of RGG, the edge exists between the i-th and j-th nodes, if and only if the inter-nodal distance between these nodes falls below a cut-off. We then need to consider: what is the probability for $G_{ij} = 1$? Chakrabarty (2023) addressed this question by learning the edge posterior, conditional on the (partial) correlation between pairs of nodes.

Definition 2.2.5. *For the RGG variable constructed according to the Definition 2.2.1, the edge posterior is denoted* $m(G_{ij} = g_{ij}|R_{ij} = |\rho_{ij}|)$.

Taking cue from Chakrabarty (2023), I start by identifying the *pdf* of the partial correlation R_{ij} between X_i and X_j, given the edge between the i-th and j-th nodes, $\forall i < j,\ i, j \in \boldsymbol{V}$. To learn this pdf, Chakrabarty (2023) has based on the limited information about the existence of edge G_{ij} and the correlation ρ_{ij} between X_i and X_j.

- The probability of $G_{ij} = g_{ij} = 1$, given a high value of correlation ρ_{ij}, is high.
- The probability of $G_{ij} = g_{ij} = 1$, given a low correlation ρ_{ij}, is low.
- The probability of $G_{ij} = g_{ij} = 0$, given high correlation ρ_{ij}, is low.
- The probability of $G_{ij} = g_{ij} = 0$, given low correlation ρ_{ij}, is high.

- The probability of R_{ij} given G_{ij} decreases, as the distance between G_{ij} and $|\rho_{ij}|$ increases, i.e. $|g_{ij} - |\rho_{ij}||$ increases.

Definition 2.2.6. *Given the limited information about the unknowns as enumerated above, we suggest that the probability density of R_{ij} conditional on the edge variable G_{ij} and variance variable ν_{ij}, computed at $R_{ij} = |\rho_{ij}|$ is $\mathcal{N}(g_{ij}, \nu_{ij})$. Thus, $f_{R_{ij}}(\rho_{ij}) \propto \exp(-(|\rho_{ij}| - g_{ij})^2/2\nu_{ij})$, given $G_{ij} = g_{ij}$. Here, the variance ν_{ij} of the density of R_{ij} is modelled to attain the value ν.*

Then, according to Bayes' Rule, using a $Bernoulli(0.5)$ prior on the binary variable G_{ij}, and the Uniform $Uniform(0, 1]$ prior on the variance parameter ν_{ij}, the joint posterior probability of the edge variable G_{ij} and the variance parameter σ_{ij}, conditional on the correlation $R_{ij} = |\rho_{ij}|$, is

$$\pi(G_{ij}, \nu_{ij}||\rho_{ij}|) = k\frac{1}{\sqrt{2\pi\nu_{ij}}}\exp[-\frac{(G_{ij} - |\rho_{ij}|)^2}{2\nu_{ij}}] \tag{2.2.2}$$

Then marginalising the variance parameter out of this joint, by imposing $\nu_{ij} \in (0, 1]$, one model for the marginal posterior of the edge variable G_{ij}, conditional on $R_{ij} = |\rho_{ij}|$ can be computed. We refer to this edge marginal posterior as $m(G_{ij}|R_{ij} = |\rho_{ij}|)$.

Definition 2.2.7. *The edge posterior probability of G_{ij} conditional on the correlation R_{ij} is denoted $m(G_{ij}||\rho_{ij}|)$. A closed-form expression for this conditional edge probability is computed by marginalising the variance parameter $_{ij}$ from it, using a Uniform[0,1] prior on ν_{ij}. The resulting edge marginal is identified as the following:*

$$m(G_{ij} = g_{ij}|R_{ij} = |\rho_{ij}|) = K\left[\sqrt{\frac{2}{\pi}}\exp\left(\frac{-(s_{ij})^2}{2}\right) - (s_{ij})\text{erfc}\left(\frac{|s_{ij}|}{\sqrt{2}}\right)\right], \tag{2.2.3}$$

where the disparity variable $S_{ij} = s_{ij} \equiv |g_{ij} - |\rho_{ij}||$, with $S_{ij} := |G_{ij} - R_{ij}|$ and the complementary error function is erfc$(\cdot) = 1 -$ erf$(\cdot)$. K is closed-form, $K > 0$.

Now, the distance between the i-th and j-th nodes is $d_{ij} \equiv d(x_i, x_j)$, where

$$d(x_i, x_j) = K'\left[\sqrt{\frac{2}{\pi}}\exp(-(s_{ij})^2/2) + (s_{ij})\,\text{erf}(|s_{ij}|/2)\right], \tag{2.2.4}$$

where $S_{ij} = s_{ij}$ is the disparity $|(G_{ij} - |\rho_{ij}|)|$. So $d_{ij} \in [0, 1]$ is a cdf, as long as the global scale K' is $[\text{erf}(1/\sqrt{2}) + \sqrt{2/e\pi} - \sqrt{2/\pi}]^{-1}$. (As we can see, for $i = j$, $d_{ij} = 0$; $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$).

Definition 2.2.8. *We adjust the zero-point of $m(G_{ij} = g_{ij}||\rho_{ij}|)$, to define $a_{ij}(s) := m(G_{ij} = g_{ij}||\rho_{ij}|) - K\sqrt{2/\pi}$, where $K = (\sqrt{2/(\pi\exp(1))} + \text{erf}(1/\sqrt{2}) - 1)^{-1}$. Then $a_{ij}(0) = 0$ and $a_{ij}(1) = 1$. Also, $a_{ij}(s)$ is an increasing function of $s \in [0, 1]$. Here $a_{ij}(s)$ is the affinity function between the i-th and j-th nodes, that takes the value of the disparity variable $S_{ij} = |G_{ij} - R_{ij}|$ as its input.*

Thus, the inter-nodal affinity function is given by the edge marginal posterior, conditional on the correlation between the random variables that are attached to these nodes.

This distance function is related to this affinity function in the following way.

- As d_{ij} increases, $m(G_{ij} = g_{ij}||\rho_{ij}|)$ decreases, i.e. $a_{ij}(\cdot)$ increases. Similarly, d_{ij} decreases $\implies$ $m(G_{ij} = g_{ij}||\rho_{ij}|)$, and therefore $a_{ij}(\cdot)$ increases.
- So when d_{ij} falls short of the cut-off probability τ, we set $G_{ij} = g_{ij} = 1$. This coincides with the following: when $m(G_{ij} = g_{ij}||\rho_{ij}|)$ exceeds τ, i.e. $m(G_{ij} = g_{ij}||\rho_{ij}|) > \tau \implies g_{ij} = 1$; else $g_{ij} = 0$.
- This is equivalent to saying that the edge G_{ij} exists (i.e. $g_{ij} = 1$), if the affinity a_{ij} exceeds a cut-off, which in turn is equivalent to the distance d_{ij} falling short of a cut-off probability. Again, $g_{ij} = 0$ if affinity falls short of a cut-off, i.e. distance exceeds a cut-off.

Definition 2.2.9. *The distance function $d(x_i, x_j)$ between the two points X_i and X_j, $i, j \in \mathbf{V}$ is the cdf $F_{S_{ij}}(s)$, is:*

$$d(x_i, x_j) = F_{S_{ij}}(s_{ij}) = K' \left[\sqrt{\frac{2}{\pi}} \exp(-s_{ij}^2/2) + s_{ij} \operatorname{erf}(s_{ij}/2) \right], \tag{2.2.5}$$

Here the disparity $S_{ij} = |(G_{ij} - |\rho_{ij}|)|$ and the constant K' is positive, $[\operatorname{erf}(1/\sqrt{2}) + \sqrt{2/e\pi} - \sqrt{2/\pi}]^{-1}$.

2.2.6 Random graph variable

Since the graph is built with independent edge variables, and there are no self-loops in this graph, we then have the closed-form posterior probability of the graph variable, given the correlation structure of the data $\mathbf{D}$, as presented next.

Definition 2.2.10. *The posterior of the RGG variable $\mathcal{G}_{\boldsymbol{V},m}(\boldsymbol{\Sigma}^{(C)}, \tau)$ that is*

- *defined on the vertex set $\boldsymbol{V}$;*
- *is learnt using the edge posterior $m(\cdot|\cdot)$ (Definition 2.2.7) s.t. the edge exists if the edge posterior exceeds the cut-off τ, given the learnt/estimated/known correlation matrix $\boldsymbol{\Sigma}^{(C)} = [\rho_{ij}]$ of data $\mathbf{D}$,*

$$\pi(\mathcal{G}_{\boldsymbol{V},m}(\boldsymbol{\Sigma}^{(C)}, \tau)|\{\rho_{ij}\}_{i,j\in\boldsymbol{V}}) = \prod_{i,j\in\boldsymbol{V}} m(G_{ij} = g_{ij}||\rho_{ij}|).$$

Here the RGG is constructed so that it has no self-loops, and the edges are included independently of each other.

Since the edge posterior of G_{ij} is known $\forall i < j; i, j \in \boldsymbol{V}$, we can simply perform rejection sampling (Definition 1.5.1), which is a basic technique used to generate samples from a target distribution by using a proposal distribution. We learn the edge variable G_{ij} by sampling from the marginal $m(G_{ij}|R_{ij} = |\rho_{ij}|)$ directly, i.e. the target distribution in the rejection sampling algorithm refers to this edge posterior, given the correlation between X_i and X_j. I undertake rejection sampling of g_{ij} from the closed-form edge posterior probability $m(G_{ij}||\rho_{ij}|)$, of G_{ij}. I generate N samples of G_{ij} as $\{g_{ij}^{(1)}, \ldots, g_{ij}^{(N)}\}$. Here, $g_{ij}^{(n)}$ can be 1 or 0, $\forall n = 1, \ldots, N$.

Remark 2.2.1. *If we sample $X = x^*$ from a function $g(x)$, for $M \in (0, \infty)$ s.t.*

$$Mg(x) \geq f(x), \forall x;$$

with $u \sim U[0, 1]$, we accept the sampled x^* if

$$\frac{f(x^*)}{Mg(x^*)} \geq u,$$

then $x^* \sim f_X(x)$.

Definition 2.2.11. *We undertake rejection sampling of the edge G_{ij} to generate N sampled values of this edge variable. The relative frequency of the edge variable G_{ij} in this sample $\{g_{ij}^{(1)}, g_{ij}^{(2)}, \ldots, g_{ij}^{(N)}\}$ is*

$$\omega_{ij} = \sum_{n=1}^{N} g_{ij}^{(n)}/N.$$

In the final graph that I will learn for the data $\mathbf{D}$, over all N trials of rejection sampling, the edge between the i-th and j-th node will exist, (i.e. $g_{ij} = 1$) if and only if $\omega_{ij} > \tau$, $\forall i < j$; $i, j \in \boldsymbol{V}$.

2.3 Inter-graph distance

In previous chapter, our discussion about the distance between two graphs is referring as a distance between their corresponding posterior probability distributions learned from different datasets, which enables graph comparison using tools from probability theory and information theory. In particular, measures such as Hellinger distance and Kullback–Leibler (KL) divergence are used to quantify differences between these posterior distributions. These metrics provide a principled way to compute inter-graph distance in a probabilistic framework.

Definition 2.3.1. *The square of Hellinger distance between the probability density $p_1(x)$ and another density $p_2(x)$ (for $x \in \mathcal{X}$) is defined as :*

$$D_H^2(p_1, p_2) = \int_{\mathcal{X}} (\sqrt{p_1(x)} - \sqrt{p_2(x)})^2 dx.$$

Therefore, for the graph variables learnt for data $\mathbf{D}_1$ and $\mathbf{D}_2$, given that the graph posterior probability is computed at each of the N number of samples generated using sampling methods (which will introduced later in this chapter), we compute the Hellinger distance between these two posterior.

Definition 2.3.2. *Given the graph posterior probability of graph variables $\mathcal{G}_1, \mathcal{G}_2$ learnt for data $\mathbf{D}_1$ and $\mathbf{D}_2$ respectively, the square of discretised Hellinger distance between these posterior probabilities is*

$$D_H^2(\mathcal{G}_1, \mathcal{G}_2) = \frac{\sum_{r=1}^{N} \left(\sqrt{\pi(\mathcal{G}_1|\mathbf{D}_1)^{(r)}} - \sqrt{\pi(\mathcal{G}_2|\mathbf{D}_2)^{(r)}} \right)^2}{N}, \tag{2.3.1}$$

where $\pi(\mathcal{G}_1|\mathbf{D}_1)^{(r)}$ is the graph posterior given the data $\mathbf{D}_1$, computed with the r-th sample of the edges of the graph, $\forall r = 1, \ldots, N$.

Definition 2.3.3. *We can compute the Kullback-Leibler divergence between the posteriors of the graphs $\mathcal{G}_1$ and $\mathcal{G}_2$, given the data $\mathbf{D}_1$ and $\mathbf{D}_2$, as:*

$$D_{KL}(\mathcal{G}_1, \mathcal{G}_2) = \sum_{r=1}^{N} \pi(\mathcal{G}_1|\mathbf{D}_1)^{(r)} log\left(\frac{\pi(\mathcal{G}_1|\mathbf{D}_1)^{(r)}}{\pi(\mathcal{G}_2|\mathbf{D}_2)^{(r)}} \right), \tag{2.3.2}$$

where the posterior is computed of the graph variable given the datasets.

Details of the implementation of Hellinger distance and KL divergence have been represented in the learning of real-world datasets in later chapters. In general, we discuss the conceptualisation of learning of graphical variable, by computing the distance and divergence between graphs, given real-world time series data. In my thesis, I conduct such inter-graph computation for each of a pair of time series datasets, and aim for further studies, such as learning the relationship - between a system variable and an observable - as a function that is generated from a Gaussian Process. An example real-world implementation of such learning has been included in the Appendix C.

2.4 Choosing cut-off τ

In the previous section, we explored the theory behind the learning of the graph variable, given a multivariate dataset, the inter-variable correlation structure of which is known. Such graph learning provides a new and robust way of exploring datasets and data structures, and this thesis discusses applications of this technique in various real-world scenarios (details in later chapters). However, the exploration of the theoretical aspect of the RGG learning is far from complete. We still need to delve deeper into the identification of the cut-off probability τ in our sought RGGs. This τ is a parameter of the RGG variable $\mathcal{G}_{\boldsymbol{V},m}(\boldsymbol{\Sigma}^{(C)}, \tau)$ that we seek, given data $\mathbf{D}$.

We recall that in an RGG an edge exists if the length of the edge, i.e. the distance between two nodes, exceeds a cut-off distance. For an RGG in a probabilistic metric space, its edge is a probability distribution over positive-definite support, s.t. the edge exists if this probability exceeds a cut-off probability τ. Then, in order to yield a graph which presents sufficient information about data structure, we need to decide this cut-off parameter τ carefully - the larger the τ we use, the fewer the edges that will be accepted in the graph, by the definition of edge acceptance criterion Definition 2.2.11.

2.4.1 Existing studies

There are various studies that exist on selecting thresholds in network learning, where such thresholding influences network topology; model performance; and interpretability (Freeman et al., 2007; Bassett et al., 2008; Drakesmith et al., 2015). Fixed thresholding is simple and computationally efficient (Freeman et al., 2007). Another trick is retaining the top k percent of the edges in terms of correlation, which is particularly useful in maintaining network sparsity (Ala et al., 2008). Perkins and Langston (2009) addressed the challenge of determining thresholds in constructing gene co-expression networks and proposed a systematic approach utilising spectral graph theory. By computing eigenvalues and eigenvectors of a transformed adjacency matrix at various threshold levels, they employ spectral clustering methods to assess the community structure within the data. This analysis facilitates the identification of thresholds that accurately reflect the underlying biological relationships, moving beyond arbitrary or conventional cut-off values. Theis et al. (2023) combine learning of normalised mutual information along with the density and clustering coefficient to compare the baseline and thresholded networks.

However, there are still limitations that impact the accuracy, interpretability, and lack of adaptability to different datasets. Fixed thresholding approaches apply a chosen cut-off to determine the presence or absence of edges in a network. Then the choice of the threshold might be subjective, and in general, it lacks a systematic motivation, leading to inconsistencies across different datasets. Moreover, some biologically or socially relevant interactions may have low correlation but may still contribute to network functionality - a fixed threshold may oversimplify the complex structures of a network. Also, spectral graph analysis requires significant processing power, making it infeasible for learning of very large networks.

Hence we are looking for a data-driven thresholding method, which leverages statistical analysis to determine meaningful optimal thresholds. Thus, the threshold selection used here is context-specific rather than arbitrary, with this criterion of the optimal τ, opening a possibility for preservation of network topology.

2.4.2 RGG-based thresholding approach comparing against COGNET-based thresholding

I treat the posterior of the RGG variable, as a function of τ, and examine the τ at which the rate of change of this function is least, and, most. Thus, the slope of

this function (i.e. of the RGG posterior as a function of τ) is examined, in order to identify the values τ_{min} of τ at which this is a minimum, and identify the values τ_{max} of τ at which this slope is maximum. Then at $\tau = \tau_{min}$, the rate of change of the (logarithm of the) posterior probability of the RGG variable is minimum, while the graph posterior is changing the most with τ, when $\tau = \tau_{max}$. Thus, the RGG realised at $\tau = \tau_{max}$ is most sensitive to changes in τ, while the graph learnt at $\tau = \tau_{min}$ is the least sensitive to changes in τ. In other words, at $\tau = \tau_{min}$, I learn the most robust - to changes in τ - RGG.

In this implementation, I seek values τ^* of τ, at which the slope of the logarithm of the posterior of the RGG variable, conditional on $\mathbf{\Sigma}^{(C)}$, treated as a function of τ, is (locally) maximised/minimised. As motivated above, such a locally maximum/minimum slope informs on the realisation of the graph variable that is maximally/minimally sensitive to changes in τ, at a given $\mathbf{\Sigma}^{(C)}$. I show more examples of this approach in a real-world dataset in the later chapters (details in Chapter 3, Chapter 4 and Chapter 5).

- Hence I attempt to identify the local maxima $\tau = \tau_{max}$ of the function that is the slope of the RGG posterior, as a function of τ. Then $\tau = \tau_{max}$ yields the graph that most sensitively reflects the correlation of the dataset.

- I also looked at the slope function at $\tau = \tau_{min}$, since these values of τ yield the highest consistency amongst the graphs learnt for a given data. Here, RGG compatibility with $\mathbf{\Sigma}^{(C)}$ changes (i.e. RGG posterior changes), only because τ has changed. I looked for the local minima in the slope of the log posterior probability of the RGG given $\mathbf{\Sigma}^{(C)}$. At this local minima in this slope, I claim to have found the 'most robust' RGG, i.e. the RGG that is most 'robust' to changes in τ.

I have looked for a chance to compare the results of thresholding in the learning of RGGs by looking for the most-sensitive or the most-robust graphs, where such graphs are identified by the highest and lowest rates of change of the RGG posterior, with change in the threshold τ. In order to address such a comparison, I looked at another thresholding technique that addresses the consistency of graphs/networks. This is the technique called COGENT (COnsistency of Gene Expression NeTworks). COGENT is an R package designed to aid the construction of gene co-expression networks, without the need for annotation or other external validation data. It can be used to choose between competing co-expression measures (e.g. Pearson vs. Kendall correlation coefficients), as well as to inform on the score cut-off choice (Bozhilova et al., 2020; Pardo-Diaz et al., 2021, 2022).

In order to further understand the connection between our learnt τ and the threshold given by COGENT, we learnt RGGs for the same dataset that was used by Bozhilova et al. (2020) to learn the most consistent network. This is the yeast dataset that is taken from the Gene Expression Atlas and it contains 200 genes expressed across 26 samples. Bozhilova et al. (2020) identified the threshold that yields the most consistent network drawn for this dataset. I learn the posterior of the RGG variable at different values of τ for this same dataset, and generate a

sample of posterior values of the RGG variable given the inter-gene correlation of the data, generated at the chosen cut-off values of $\tau \in (0, 1)$. Tracking changes in the logarithm of the posterior of the graph variable with changes in τ, was motivated to identify values of τ that correspond to maximal and minimal rates of change in the log posterior with change in τ.

Definition 2.4.1. *I define the rate of change of (logarithm of) posterior with τ, by computing - via simple differencing between log posteriors generated at successive τ values - the 'slope' of the RGG posterior γ, as follows:*

$$\gamma := \frac{\textit{log posterior at } \tau_i \textit{ - log posterior at } \tau_j}{\delta_\gamma},$$

here $\delta_\gamma = \tau_i - \tau_j . \forall i, j \in \mathbb{N}, \tau \in (0, 1)$

I approximate this slope, γ of the RGG posterior, as a function of τ. In Figure 2.1 I plot a sample of posterior values of the RGG variable, generated at the chosen cut-off values of $\tau \in (0, 1)$, along with corresponding calculated γ.

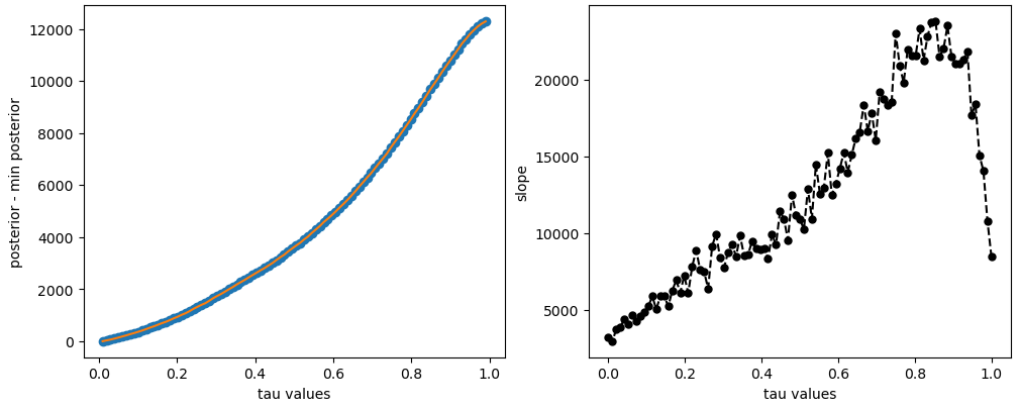


Figure 2.1: The left panel presents values of the posterior probability of RGGs learnt for the gene expression data that Bozhilova et al. (2020) applied the COGENT thresholding technique to. Each RGG is learnt at a distinct value of τ s.t. the log posterior value is plotted against τ. A linear interpolation fitted line in orange. The right panel shows the slope γ of the graph posterior, plotted against the corresponding τ value. In my work, τ_{max} is approximately 0.80.

From the inspection of this slope γ, the most-robust (to changes on τ) RGGs that are the most robust - to changes in the threshold τ - occur at local minima in the γ, which is attained at $\tau = \tau_{min} \approx 0.4$, The most-sensitive RGGs are those that are learnt around $\tau = \tau_{max} \approx 0.8$. On the other hand, COGENT suggests that of networks computed with this dataset, the networks that are the most consistent, are attained at a threshold of about 0.3 (thresholding the top 70% of calculated Pearson correlation coefficient values). This is depicted in Figure 2.2.

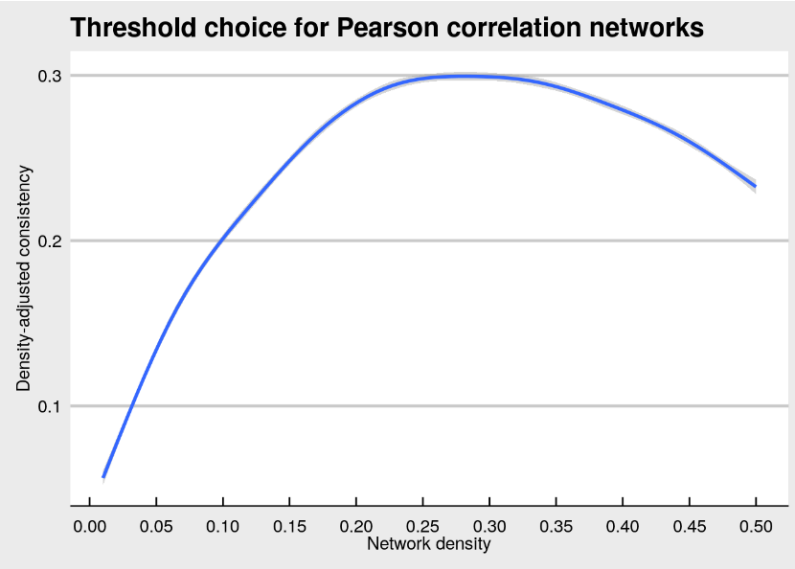


Figure 2.2: The threshold value about 0.3 yields the most consistent network, suggested by the COGENT method (Bozhilova et al., 2020), using the gene data.

I produce RGGs and the networks for the gene expression data presented by Bozhilova et al. (2020), to compare the results using COGENT and the thresholding method that I use on my learnt RGGs. The threshold at which COGENT yields the most robust network, is referred to as $\tau = \omega$. The RGGs learnt at $\tau = \tau_{max}$ and $\tau = \tau_{min}$ are identified in the set of RGGs learnt using a given data, across values of τ.

– I learn the graph at a $\tau = \tau_{min}$ that produced the most-resilient (to changes in τ) graph, and a $\tau = \tau_{max}$, the least-resilient i.e. most-sensitive (to changes in τ) graph;

– I also compare the results to learning the graph - as an RGG - at a $\tau = \omega$ that produces the most-robust network as COGENT suggest;

– and I learnt the graph as a network driven purely by the Pearson Correlation coefficients, as done within the COGENT package.

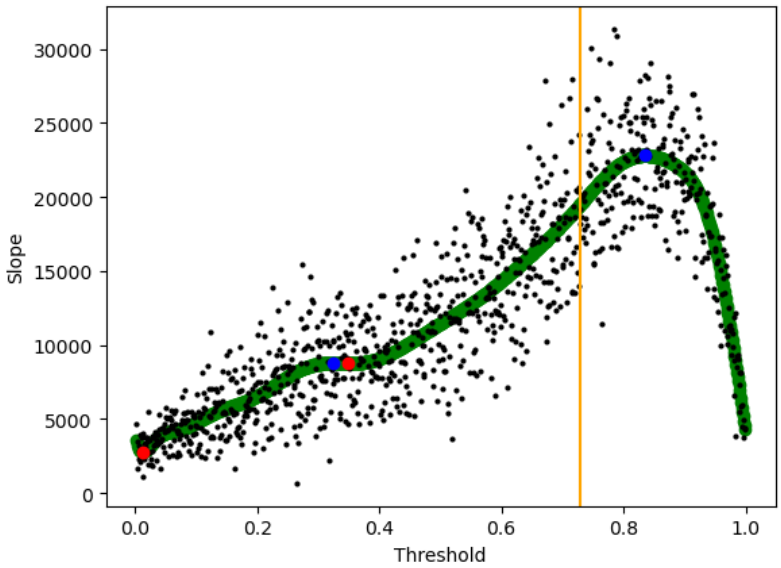


Figure 2.3: For the data used by Bozhilova et al. (2020), the computed slope γ of the log posterior of the learnt RGG is plotted against thresholds. A fit to the data is superposed in green, and the orange vertical line indicates the COGENT-suggested best threshold of $\omega = 1 - 0.27 = 0.73$ (thresholding the top 73% of calculated Pearson correlation coefficient values) that yields the most-consistent network for this dataset. The local maxima/minima of γ are labelled in blue/red dots respectively.

.

I then produced the RGGs at the thresholds of τ_{min} and τ_{max} values identified by our criterion of thresholding, as suggested in Figure 2.3, and compared the learnt RGGS to the network that was computed by Bozhilova et al. (2020) for the gene expression data, at their best threshold of $\omega = 1 - 0.27 = 0.73$.

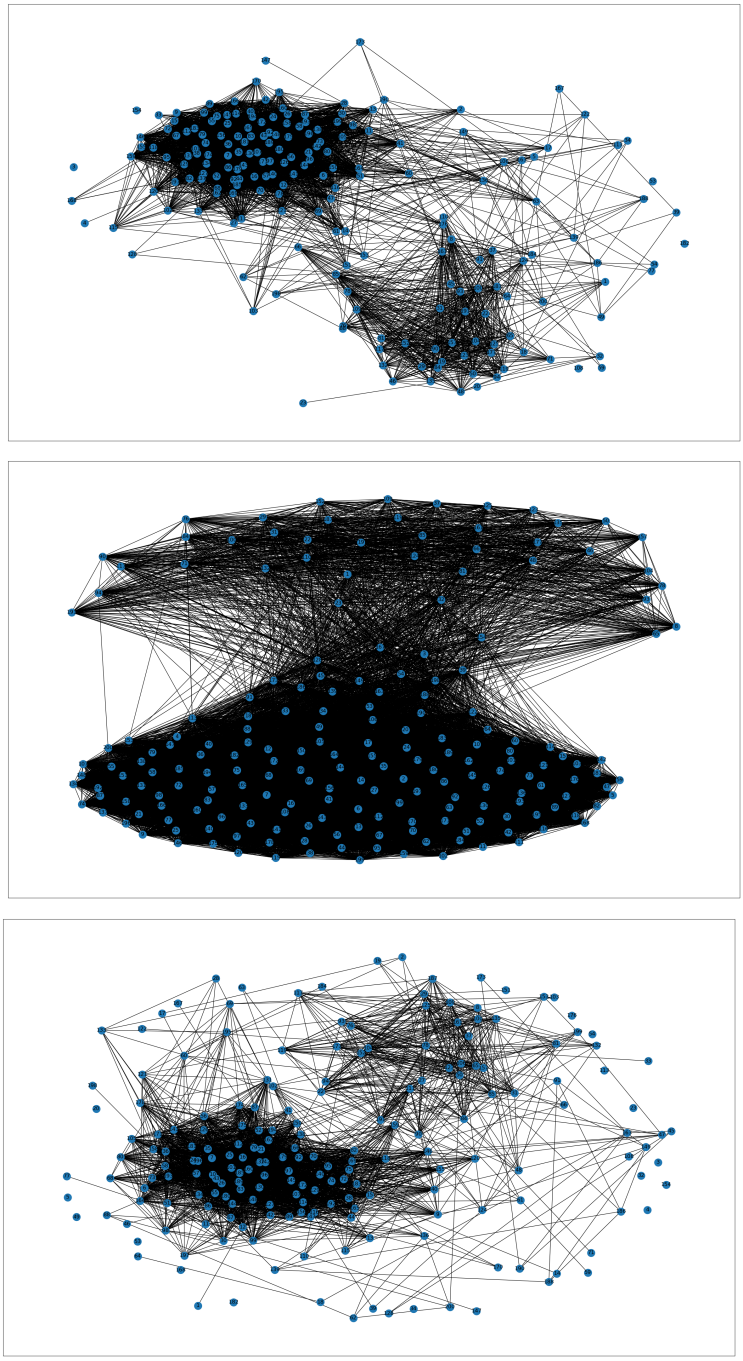


Figure 2.4: The graphs produced by the thresholds suggested in Figure 2.3. From top to bottom at τ given by the best suggestion by COGENT ($\omega = 0.73$); at $\tau = \tau_{min}$ (chosen $= 0.3368365$); and $\tau = \tau_{max}$ ($= 0.834835$)

I also compared the nodal degree distribution of the RGGs learnt given the data used by Bozhilova et al. (2020) at (i)$\tau = \tau_{max}$; (ii)$\tau = \tau_{min}$; (iii) and such degree distribution in the network computed using the COGENT-suggested best threshold ω.

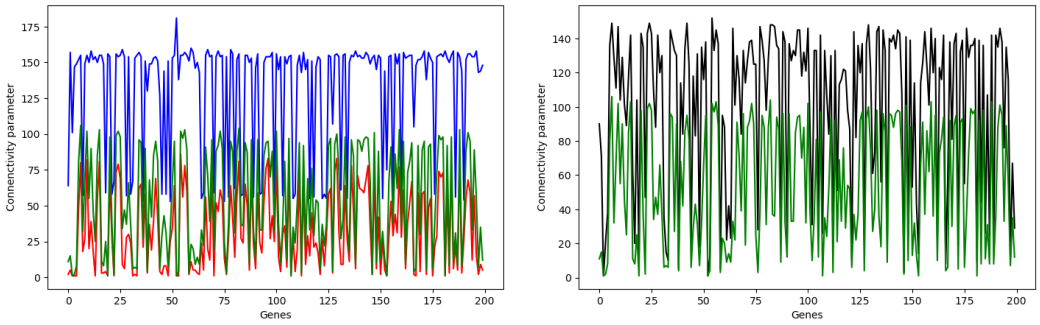


Figure 2.5: Left panel: The nodal degree is plotted against the nodal index, in RGGs learnt for the data used by Bozhilova et al. (2020), at the thresholds τ_{min}, τ_{max} and ω identified in Figure 2.3. The degree values computed at $\tau = \omega = 0.73$ are joined with green lines; at $\tau = \tau_{max} \approx 0.834835$, is in red; at $\tau = \tau_{min} \approx 0.3368365$ is in blue. Right panel: degree plotted against nodal index in RGG drawn at $\tau = \omega (= 0.73$, in black); and at the same threshold ω, when the graph is in fact a Pearson Correlation-driven network.

- From Figure 2.4, both RGGs drawn at the $\tau = \omega$ and $\tau = \tau_{max}$ show two clusters. After all, these two threshold values are closer to each other (at about 0.73 and 0.8). The RGG learnt at $\tau = \tau_{min}$ is indeed very different.

- In the right panel of Figure 2.5, we note that learning a Pearson Correlation-driven network (shown in green) leads to a lower local discrimination in values of the nodal degree across nodes - compared to when the graph is learnt as an RGG (in black).

In later chapters, I will discuss further details of the application of the thresholding technique that we have introduced.

2.5 Summary, discussion and contribution

The graph-learning framework that I have elucidated above, is firstly, easy to learn. The closed-form edge posterior that is allowed within this framework, is very useful since it allows for easy Bayesian inference - using rejection sampling from this known closed-form edge posterior - to be carried out on each edge variable individually, given the correlation between the random variables that are attached to each of the two nodes that straddle this considered edge. This is an easily interpretable way of defining an edge, and thereby the full graph that comprises said edges. At the same time, this graph-learning framework suggests embedding the sought graph

in a probabilistic metric space, such that the distance between any two nodes of this graph, is in fact a probability distribution. Then, as long as this inter-nodal distance falls short of a chosen cut-off probability, the edge exists in this Random Geometric Graph; otherwise it does not. Equivalently, the edge exists if the inter-nodal affinity exceeds this cut-off probability (where such affinity is complementary to the inter-nodal distance).

The design of an optimal cut-off probability is my own methodological contribution, and I have discussed this above, as an attempt at recognising that cut-off, which minimises the rate of change of the posterior of the RGG variable, with change in the cut-off. Thus, my method of identification of the optimal cut-off stems directly from the formulation of the sought graph as a random variable that admits a posterior probability, given the data at hand - which in our consideration is the correlation between the random variables attached to each of the considered two nodes.

Importantly, multiple applications will become possible by computing a statistical distance/divergence measure between the posterior of the RGG variable given the correlation structure of one dataset, and that given another dataset, (with a different length than the other dataset, in general). Such a holistic inter-graph distance is possible only because the graph is treated in this graph-learning framework, as a graph-valued random variable, for which we can define a posterior probability, given the correlation structure of a dataset. In lieu of such a lucid definition of the graph posterior, one may use a node-by-node distance between two graphs, or other more *ad hoc* distance definitions, such as Hamming distances (Hamming, 1950; Waggener, 1995; Jarrous and Pinkas, 2009). The inter-graph distance defined above is very easily interpretable, and easy to compute, while it is a lossless definition of the difference between two graphs defined on a small/large-sized vertex set, given multivariate datasets that can be of general types, including numerical observables, as well as values of categorical observables. This is why I refer to this distance as "holistic" above, and such a generic inter-graph distance will be invoked in the next two chapters of my thesis, to undertake difficult real-world problems.

In the previous chapter, I have qualitatively invoked existing concepts of random graphs; probabilistic metric spaces; correlation-based construction of random geometric graphs; and motivated the background for modelling uncertainty in our graph learning. In this chapter, I established the existing theoretical and methodological framework for learning RGGs of multivariate datasets, within a Bayesian setting - a novel framework has been reported, in which the graph is modelled as a random variable in a probabilistic metric space. In such a space, inter-nodal distances are defined as cumulative distribution functions (of a disparity variable), and directly relevant to the existence of any edge. Within this framework, a closed-form expression exists for the posterior probability of the edge variable, conditioned on the correlation between this pair of nodes. We propose rejection sampling to sample the edge from its posterior.

One of my methodological contributions is a data-driven threshold selection, in which the optimal cut-off parameter is determined by analysing the sensitivity/robustness of the graph posterior with respect to the threshold. This approach

addresses limitations of existing fixed thresholding techniques. In general, the chapter forms the methodological basis for the applications and of the extensions, that are developed in subsequent chapters.

Chapter 3

Empirical Illustration of RGG Learning on Oncological Data

3.1 Introduction

I have discussed the methodological framework within which we can learn graph-valued random variables, the geometric aspect of which is acknowledged in this framework. We aim to learn the edge variables of a graph by sampling from their posterior distribution, given observed data. The posterior of an edge captures the uncertainty in the presence or absence of edges, informed by the correlation structure of the dataset.

In Chapter 2, for any nodal pair, I discussed the joint posterior distribution of model parameters, namely the edge variable and the variance parameters, in Equation 2.2.2 - to learn the edge variable G_{ij}, conditional on the partial correlation Ψ_{ij} of a given dataset $\mathbf{D}$, s.t. we can define a posterior probability $m(G_{ij} = g_{ij}||\Psi_{ij}| = \psi_{ij})$. Then all that is required to learn the edge variable G_{ij}, is to sample from this closed-form edge posterior; of course, the geometric signature of the graph can be then imposed, by accepting a sampled edge as 1, only if the inter-nodal distance falls short of a chosen cut-off probability. Learning for all relevant nodal pairs, then results in the learning of the random geometric graph variable. A fallout of this graph learning is the inter-graph distance that is the distance between the posterior probabilities of the RGG variable, given two distinct multivariate datasets.

It is an application of this inter-graph distance computation, that I discuss in this chapter. I am part of a collaboration in which we employ inter-graph distances between the posterior probabilities of an RGG variable that is learnt given time series data recorded on the physiological parameters of one patient, and that given such data on similar parameters of another patient. The inter-graph distance then informs on the strength of the effect - that is the differential progression or severity of the underlying disease in one patient, compared to that in the other - that distinguishes the graph learnt for one patient, from that of another. Thus, the inter-graph distance tells us about the relative severity of the disease in one patient, compared to that in another. This is the part of the work in this collaboration that I have conducted as part of my doctoral work. I focus on this part of this project in this chapter.

3.2 Method

Remark 3.2.1. *By Bayes' Rule, using* $Bernoulli(0.5)$ *and* $Uniform(0,1]$ *to be the priors for binary variable* G_{ij} *and the variance parameter* ν_{ij}*, the joint posterior probability of the edge variable* G_{ij} *and the variance parameter* σ_{ij}*, conditional on the partial correlation* $\Psi_{ij} = \psi_{ij}$ *is*

$$\pi(G_{ij}, \nu_{ij}||\psi_{ij}|) = k\frac{1}{\sqrt{2\pi\nu_{ij}}}\exp\left[-\frac{(G_{ij} - |\psi_{ij}|)^2}{2\nu_{ij}}\right]$$

While we can sample both edge and variance parameters from this joint posterior using MCMC, in most applications, we perform rejection sampling of the edge parameter only, from the edge marginal posterior that is obtained by marginalising out ν*:*

$$m(G_{ij}|\Psi_{ij} = \psi_{ij}) = K\left[\sqrt{\frac{2}{\pi}}\exp\left(\frac{-(s_{ij})^2}{2}\right) - (s_{ij})\text{erfc}\left(\frac{|s_{ij}|}{\sqrt{2}}\right)\right], \tag{3.2.1}$$

where the disparity variable $S_{ij} = s_{ij}$*, with* $S_{ij} := |G_{ij} - |\Psi_{ij}||$ *and the complementary error function is* $erfc(\cdot) = 1 - erf(\cdot)$*. K is closed-form,* $K > 0$*. This holds* $\forall i < j; i, j \in \{1, 2, \ldots, p\}$*.*

I discuss my undertaking of sampling-based Bayesian inference (or MCMC) on both graph parameters G_{ij} and variance parameter ν_{ij} in detail, given a real-world dataset, in the Appendix B, $\forall i < j; i, j \in \{1, 2, \ldots, p\}$. In this chapter, however, I will undertake Bayesian inference from the posterior of any edge variable, after having marginalised out the variance parameter from the joint posterior of the edge and the variance. Since the resulting edge marginal is closed-form, I undertake Bayesian inference on the edge, using rejection sampling.

To perform rejection sampling of $G_{i,j}$ from the edge marginal $m(G_{ij}|\psi_{ij})$, I chose the proposal distribution p_{ij} to be Uniform, or Bernoulli, on separate instances. I performed rejection sampling to draw N samples of the edge variable G_{ij} that joins the i-th and j-th nodes of the random graph variable, where the i-th node of this RGG is associated with the i-th variable X_i in the dataset $\mathbf{D}$, for $i = 1, \ldots, p$. The edge G_{ij} is sampled from its closed-form marginal posterior $m(G_{ij}|\psi_{ij})$, such that the binary edge variable G_{ij} its value is denoted $g_{ij}^{(t)}$ in the t-th sample. I accept the proposed edge if the acceptance ratio exceeds or equals u where $U = u$, with the Uniform random variable $U \sim U[0, 1]$. The acceptance ratio of the t-th iteration is defined as $\dfrac{m(g_{ij}^{(t)}|\psi_{ij})}{Cp_{ij}}$, with the constant $C > 0$ chosen to ensure that $Cp_{ij} \geq m(G_{ij}|\psi_{ij})$, s.t. rejection sampling is valid.

When I choose to perform MCMC to learn the unknowns, the constant K in marginal posterior $m(G_{ij}|\psi_{ij})$ becomes irrelevant in the calculation, since MCMC works oblivious to global constants. Although K is known, in the implementation of rejection sampling, we can simply choose the constant C in the acceptance ratio to be large enough s.t. the denominator Cp_{ij} is greater than the numerator all the

time. The relative frequency of the edge variable in this generated N-sized sample $\{g_{ij}^{(1)}, g_{ij}^{(2)}, \ldots, g_{ij}^{(N)}\}$ is recorded, as discussed in Definition 2.2.11.

3.2.1 My implementation of RGG learning & inter-graph distance given real Oncological data

I developed Algorithm 3 as the protocol for learning the RGG of data $\mathbf{D}$. I have implemented this RGG learning technique to several real-world datasets.

Algorithm 3: Algorithm for implementation of RGG

```
/* Learning RGG of data D.                                          */
```
Dataset $\mathbf{D} = [x_{ij}]_{i=1;j=1}^{n;p}$ which comprising N_t measurements of $X = (X_1, ..., X_p)$
Estimate $p \times p$-dimensional inter-column correlation matrix $\mathbf{\Sigma}^{(C)} = [\rho_{ij}]$.
Calculate $p \times p$-dimensional partial correlation matrix $\mathbf{\Psi} = [\psi_{ij}]$.
for $i \leftarrow 1$ **to** p **increment by** 1, **do**
  for $j \leftarrow 1$ **to** p **increment by** 1, **do**
    Given i, j-th element of $\mathbf{\Psi}$, sample g_{ij} from edge marginal
    $m(G_{ij} = g_{ij}|\psi_{ij})$ given in Definition 2.2.7
    Compute posterior probability $\pi(\mathcal{G}_{\boldsymbol{V},m}(\mathbf{\Psi}, \tau)|\{\psi_{ij}\}_{i<j;j\in\boldsymbol{V}})$ of random graph variable $\mathcal{G}_{\boldsymbol{V},m}(\mathbf{\Psi}, \tau)$
    ```
    /* Sampling the edge posterior by performing rejection
       sampling                                                     */
    ```
    for $t \leftarrow 1$ **to** N **increment by** 1, **do**
      Sample from posterior probability $\mathcal{G}_{\boldsymbol{V},m}(\mathbf{\Psi}, \tau)$ using rejection sampling, to generate sample $\{g_{ij}^{(1)}, g_{ij}^{(2)}, \ldots, g_{ij}^{(N)}\}$ for edge variable $G_{ij} = g_{ij} = 1$
    Compute relative frequency $\omega_{ij} := \sum_{t=1}^{N} g_{ij}^{(t)}/N$.
    $G_{ij} = 1 \iff \omega_{ij} > \tau$.

Below, I have used the data on a cohort of patients who were at risk of developing a disease called Veno Occlusive Disease (or VOD). This disease often develops in recipients of Bone Marrow Transplants (BMTs), which is a procedure that is opted for by sufferers of certain blood cancers, and other haematological diseases. However, the very technique that such patients opt for as a cure, induces VOD in these patients, where VOD can be terminal. In the project that I am a partof, the aim was to predict a score of the severity with which VOD will develop in a patient, where this prediction is sought before the patient has had their BMT. Then it is on the basis of the patient's pre-BMT variables that this prediction needs to be made. Such a prediction is undertaken after the relationship between the vector of these pre-BMT variables and a score variable is learnt. Such learning can be undertaken only if a training dataset exists, comprising pairs of values of these the pre-BMT variable vector of a patient, and the corresponding score of the severity with which VOD developed in this patient after their BMT.

So we needed to learn this severity score of each patient in a cohort of patients who had already received their BMT, and were suffering from various levels of VOD severity as a result. Patient charts were maintained from a few days before, to a

few days after the BMT. Such charts included time series data on 11 physiological parameters, recorded during this time interval. The scores were learnt by learning the aforesaid RGG variable given the estimated correlation structure of a patient's 11-dimensional time series data - this has been my contribution to this project. Thereafter, the distance between the graphs learnt for any patient, and that learnt for an arbitrarily-chosen "reference patient" was computed, to learn the scores. This is the patient whom we assign an arbitrarily-chosen score value. We used this assigned score value of 1. So we asked, what is the score of the severity with which VOD has onset in the i-th patient, on the scale in which the reference patient has a severity score of 1, $\forall i = 1, 2, \ldots, 25$. Given the reference patient's assigned score of 1, the relative score of every other patient in the cohort can be computed. Results of the illustration on one such patient's time series data - i.e. their patient chart - will be discussed later in this chapter. *My contribution in the project includes such inter-graph distance computation, (following the RGG learning of each of the 25 retrospective patients).*

Thus, for my part of the undertaken work, the data set of each patient was considered; such a dataset included time series data on the 11 different physiological features of 25 anonymised retrospective patients, in a cohort. Such feature parameters include: each patient's blood pressure; pulse rate; temperature; body weight, etc., recorded from six days before the BMT that the patient undergoes, to 18 days after. These time series data on these multiple patient parameters are of diverse temporal coverage, since some of the patients may expire before the interval of observation is over, given the terminal nature of the diseases that the patients are suffering from.

I start the detailed discussion of this application with a short discussion on VOD.

3.3 Application on real-world data: Veno Occlusive Disease (VOD) data

3.3.1 Existing studies on VOD

Hepatic Veno Occlusive Disease (VOD), also known as Sinusoidal Obstruction Syndrome (SOS), is a severe and potentially fatal complication that may occur following hematopoietic stem cell transplantation (HSCT, refer to 'transplant' later). Despite advances in transplantation medicine, VOD remains a major cause of early transplant-related mortality, particularly in high-risk patients. Early studies often described VOD based on clinical observations (Berk et al., 1979; McDonald et al., 1984). These foundational studies laid the groundwork for more detailed investigations into the syndrome's pathogenesis and the outcomes of VOD. In recent years, the understanding of its pathogenesis and risk factors has improved. Carreras et al. (1998) and Coppell et al. (2010) conducted large cohort studies identifying busulfan-based conditioning, pre-existing liver dysfunction, and younger age as significant contributors to VOD risk. Nevertheless, challenges remain in achieving early diagnosis and timely intervention, which are crucial for improving clinical

outcomes. There are various studies on the early diagnosis of VOD. Mohty et al. (2016) proposed new severity grading for adult patients, allowing earlier diagnosis based on clinical progression rather than fixed time frames, Corbacioglu et al. (2018) emphasised age-specific diagnostic challenges and introduced adapted severity classifications. Corbacioglu et al. (2019) later expanded on this by exploring endothelial injury and inflammation as key mechanisms in disease progression. Additionally, recent studies have explored the application of machine learning to predict VOD after HSCT (Shah and Shenoy, 2023; Lee et al., 2022; Salah et al., 2019).

Since no existing model reliably maps patient physiological parameter time series to VOD severity, we (Chakrabarty et al., 2023) proposed a new approach, as described above. A score of severity of VOD is aimed for in a patient who is suffering from this disease (post their Bone Marrow Transplant). In the cohort we consider, the absolute difference between such a score of the i-th retrospective patient and that of the arbitrarily-chosen "reference patient" is modelled as proportional to the Hellinger distance between the posterior probability of the RGG learnt for the i-th patient and that of the reference patient. Since we are defining the score, we can set the proportionality constant to be 1. We recall from above, that in our work, we arbitrarily assigned a score of 1 to the reference patient. Thus, maintaining internal consistency amongst scores of all patients in this 25-sized cohort, we can learn the scores of all patients. here $i = 1, 2, \ldots, 25$.

As part of my thesis, my work is the learning of the RGG of 1 patient in this cohort. I constructed an RGG in a probabilistic metric space.

3.3.2 Data

In the anonymised retrospective patient dataset I considered, the time series data on various physiological parameters were recorded from a pre-transplant time point to a post-transplant time point, where the disease VOD often follows BMTs. A patient's chart include time series data on the following features: systolic blood pressure; pulse rate; temperature; body weight, bilirubin level; and a few other features that are markers of VOD. For a considered patient, values of 11 such parameters were recorded over 12 days. These observations form the 12×11-dimensional dataset $\mathbf{D}$, with the 11 different physiological parameter values presented in columns of this dataset $\mathbf{D}$. Thus 12 values of the i-th observable X_i are recorded in the i-th column of the patient chart, $i \in \{1, \ldots, 11\}$, and on the j-th day, X_i takes the value x_{ij} for $j \in \{1, \ldots, 12\}$. Thus we have $\mathbf{D} = \{x_{ij}\}_{i=1;j=1}^{11;12}$. In Table 3.1, we present this data.

	blood_high	blood_low	pulse_rate	respiratory	temperature	caplillary	body_weight	fluid_balance	bilirubin	transaminase	creatinine
Day 1	130	70	84	18	97.8	98	68.3	10	0.54	20	0.84
Day 2	120	60	90	20	97.6	98	65.7	-950	0.63	22	0.71
Day 3	118	66	90	20	97.8	99	63	-1640	0.62	20	0.98
Day 4	110	60	100	20	97.4	98	63	400	0.78	18	0.87
Day 5	125	70	88	22	97.4	97	64.8	-10	0.75	17	0.85
Day 6	122	64	120	20	101	97	64	-560	0.63	14.3	0.7
Day 7	120	60	98	18	98.6	99	63.9	1307	0.42	33.6	0.79
Day 8	136	70	100	19	97.6	98	65.1	6137	1.37	22.6	0.5
Day 9	130	70	120	20	98.7	98	64.7	1456	2.22	25	0.61
Day 10	130	70	94	18	97.8	98	60.1	761	1.71	22.1	0.98
Day 11	150	80	94	14	97.1	97	61.3	1262	1.04	12.2	0.66
Day 12	120	60	89	18	97.8	97	60.1	-484	1.26	8.2	0.5

Table 3.1: 11-columned time series data on physiological parameters of a patient suffering from VOD was observed on 12 days, from before transplant to after. The patient expired before the full observational period was over.

3.3.3 Computing correlation and partial correlation

I wanted to learn the RGG variable $\mathcal{G}_{\boldsymbol{V},m}(\boldsymbol{\Psi},\tau)$ defined on the vertex set $\boldsymbol{V} \coloneqq \{1,\ldots,11\}$, given the partial correlation matrix $\boldsymbol{\Psi} = [\psi_{ij}]$ (for $i,j \in \boldsymbol{V}$) of our patient data $\mathbf{D}$. I first computed the inter-column correlation matrix $\boldsymbol{\Sigma}^{(C)}$, by computing the unbiased estimate of the correlation between each pair of observables in the time series data. I then transformed each estimated correlation to the partial correlation to compute the partial correlation matrix $\boldsymbol{\Psi}$, as defined in Definition 2.2.4. I present an excerpt of the inter-column correlation matrix $\boldsymbol{\Sigma}^{(C)}$ and the partial correlation matrix $\boldsymbol{\Psi}$ of the time series data generated by the patient data, in Table 3.1.

$$\boldsymbol{\Sigma}^{(C)} = \begin{bmatrix} 1 & 0.90437591 & -0.003579755 & \ldots & -0.327600617 \\ 0.90437591 & 1 & -0.025185308 & \ldots & -0.022730316 \\ -0.003579755 & -0.025185308 & 1 & \ldots & -0.309250562 \\ -0.665616881 & -0.472481885 & 0.181907337 & \ldots & 0.212750929 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -0.327600617 & -0.022730316 & -0.309250562 & \ldots & 1 \end{bmatrix}$$

$$\boldsymbol{\Psi} = \begin{bmatrix} 1 & 0.961626853 & -0.634144217 & \ldots & -0.784329968 \\ 0.961626853 & 1 & 0.638902994 & \ldots & 0.871323903 \\ -0.634144217 & 0.638902994 & 1 & \ldots & -0.565141096 \\ -0.745984214 & 0.588022985 & -0.372874316 & \ldots & -0.344469457 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -0.784329968 & 0.871323903 & -0.565141096 & \ldots & 1 \end{bmatrix}$$

3.3.4 Sampling from the edge marginal

To learn the realisation of the RGG variable, I learn the edge variables of the graph by sampling from their posterior, given by the correlation of data $\mathbf{D}$.

I drew 500,000 samples of the edge variable G_{ij}, which represents the connection between the i-th and j-th nodes of the RGG. To each node is attached, one of the 11 physiological parameters in $\mathbf{D}$, with $i, j = 1, \ldots, 11$. I performed rejection sampling to sample the edge G_{ij}, from its closed-form marginal posterior $m(G_{ij}|\psi_{ij})$ that is defined in Definition 2.2.7. The binary edge variable G_{ij} attains a value $g_{ij}^{(r)}$ in the r-th sample. A proposal distribution p_{ij} is either a Uniform or Bernoulli distribution, and it is accepted if the acceptance ratio satisfies

$$U \leq \frac{m(g_{ij}^{(r)} \mid \psi_{ij})}{C p_{ij}},$$

where $U \sim \mathcal{U}[0,1]$ and $g(ij)$ denotes the proposal density. The constant C is selected such that $Cp_{ij} \geq m(G_{ij} \mid \psi_{ij})$ for all values of G_{ij}, ensuring the rejection sampling procedure is valid.

The MCMC application of sampling on both graph parameters G_{ij} and variance parameter ν_{ij} given this real-world dataset has been included in the Appendix B.

3.3.5 Results: realisations of RGG variable

In this section, we present the realisation of the RGG variable using the time series data $\mathbf{D}$. Figure 3.3 presents results of using different proposal densities that have been applied in our graph learning, namely, $Bernoulli(p)$, with parameter $p = 0.7, 0.8$ or 0.85 respectively. It can be seen that the learnt graphs are strongly similar, even when the $Bernoulli(p)$ proposal density is used in the rejection sampling.

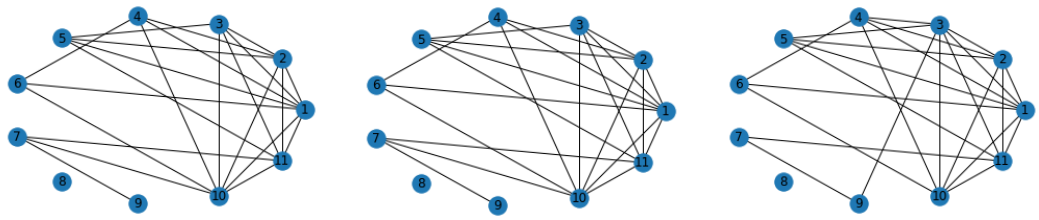


Figure 3.1: Graph realisations learnt using Bernoulli proposal densities $Bernoulli(p)$, with $p = 0.7, 0.8$ and 0.85 (from left to right). Here, the constant $C = 100$ and cutoff $\tau = 0.4$.

The resulting graphs agree with each other, as well as when I alternatively use a Uniform proposal density in the rejection sampling. Figure 3.2 below shows the graphs learnt using Uniform proposal densities, at different cutoffs, i.e. at different values of τ.

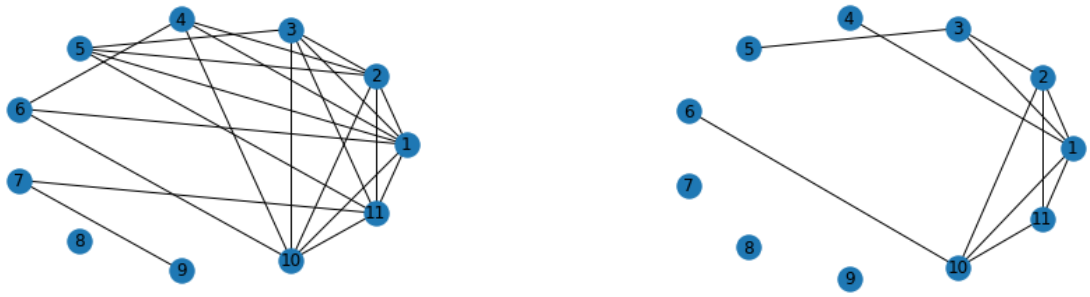


Figure 3.2: Graph realisations learnt using $Uniform\ [0, 0.8]$ proposal density, with constant $C = 100$, and cutoff $\tau = 0.4$ (left) and 0.6 (right).

It will be worth comparing the graph realisation generated by MCMC (detail included in Appendix B) and rejection sampling - the graphs learnt at the same τ are strongly similar.

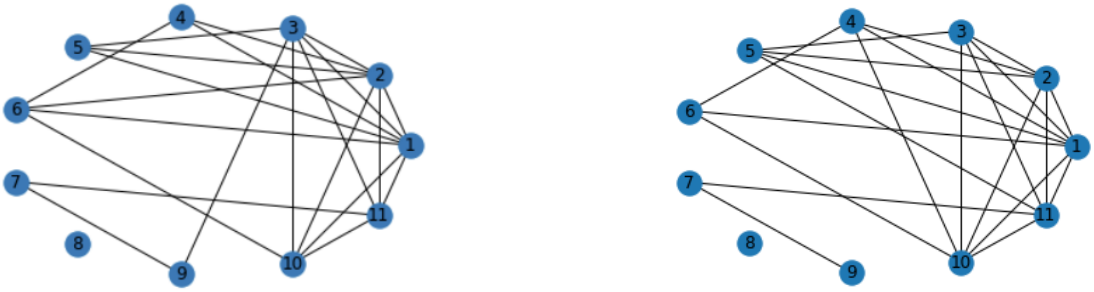


Figure 3.3: Left: graph generated by MCMC approach. Right: graph learnt using rejection sampling. Both graphs are generated at a cutoff $\tau = 0.4$.

In Figure 3.4, I include RGGs I have learnt for three different patients in this cohort.

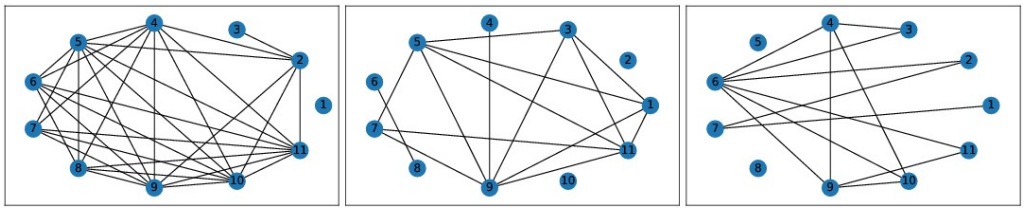


Figure 3.4: RGGs of three different patients in this cohort. All graphs are generated at a cutoff $\tau = 0.6$.

Impact of my RGG learning in VOD severity score prediction

My contribution to the project is the crucially-important step towards developing the desired capacity for predicting the VOD severity score of a new or prospective patient, who has not had their BMT at the time of the prediction. Such early and individualised prediction allows for early intervention on the part of the doctors

who will be treating this patient. They place the patient on a medication (called Defibrotide) if the predicted score, and/or clinical assessments suggest that the patient bears a high risk of developing severe VOD. The price of Defibrotide being very high, only the very at-risk patients are placed early on this medicine. Details of the project are discussed in Chakrabarty et al. (2023), and in a second paper from the project, which has recently been submitted to Scientific Reports.

My learning of the RGGs for the patients in the retrospective cohort gives rise to information about the posterior probabilities of each learnt RGG. This includes the posterior of the reference patient. Then computing the distance between the posterior of the RGG of any patient and that of the reference patient, my collaborators in the project computed the scores of each patient, which are shown in Table 1 of Chakrabarty et al. (2023). Basically, my learning of the RGGs is accompanied with a sample of posterior values of the RGG, obtained by collating the RGG posterior generated across the multiple instances of sampling that I undertake. My collaborators compute the Hellinger distance between the posterior of the RGGs of two distinct patients, using the sample of RGG posterior values generated by learning the RGG of one patient, given their time series data, and a similar sample of another patient.

The Hellinger distance $\delta_{i,ref}$ between the graph of the i-th patient, and the reference patient is set as proportional to the absolute difference between the score of the i-th patient and that of the reference patient. We know that the latter score is 1, since we have by-hand assigned the reference patient's score as 1. Here the score of a patient is in $\mathbb{R}$, since a patient can suffer from VOD, more severely than the reference patient, or could be suffering a less severe version of VOD than the reference patient. Then setting the proportionality constant as 1, (since we are defining the unit of the score), the i-th patient's score can be either $1 - \delta_{i,ref}$ or $\delta_{i,ref} + 1$. My collaborators have resolved this problem by ensuring that the choice sits consistently with the score of the the i-th patient relative to that of the j-th patient, given the computed (known) distance $\delta_{i,j}$ between the RGGs of these two patients, for any $i \neq j$, $i, j \in \{1, \ldots, 25\}$. This consistency is sought within an MCMC-based inference that is described in Chakrabarty et al. (2023). As a result, the score of any patient is learnt with uncertainties - emanating from the MCMC-based inference. Thus, the scores in Table 1 of Chakrabarty et al. (2023) are learnt for each patient - except the reference patient - within 95% Highest Probability Density credible region.

3.3.6 Results of choosing cut-off τ

For an RGG in a probabilistic metric space, any edge is a probability distribution. Then, as we introduced in the previous chapter, an edge exists between each nodal pair, if the length of the edge, i.e. the distance between two nodes, falls below a cutoff distance i.e. probability, which equivalently implies that the edge exists if the affinity (which is complementary to the distance) if the affinity exceeds this cut-off probability. Here, the affinity is given by the edge posterior probability, (givn the correlation between the random variables that are attached to the relevant nodes).

Then an edge exists if this edge posterior probability exceeds a cut-off probability τ. By the nature of RGG, fewer edges will be accepted in the graph, for a larger value of the cutoff τ.

Remark 3.3.1. *I undertake rejection sampling of G_{ij} to generate N sampled values of each edge variable of the graph $\mathcal{G}_{\boldsymbol{V},m}(\boldsymbol{\Psi}, \tau)$. If the relative frequency for N to attain the value of 1 in this sample exceeds the chosen cut-off probability τ, we retain the edge between the i-th and the j-th nodes in the final graph of data $\mathbf{D}$.*

The relative frequency of edge variable G_{ij} in the sample $\{g_{ij}^{(1)}, g_{ij}^{(2)}, \ldots, g_{ij}^{(N)}\}$ is $\omega_{ij} = \sum_{r=1}^{N} g_{ij}^{(r)}/N$. Hence in our graph, edge between the i-th and j-th nodes exists, i.e. $g_{ij} = 1$, if and only if the relative frequency $\omega_{ij} > \tau$, $\forall i, j'$; $i \neq j$; $i, j = 1, \ldots, 11$.

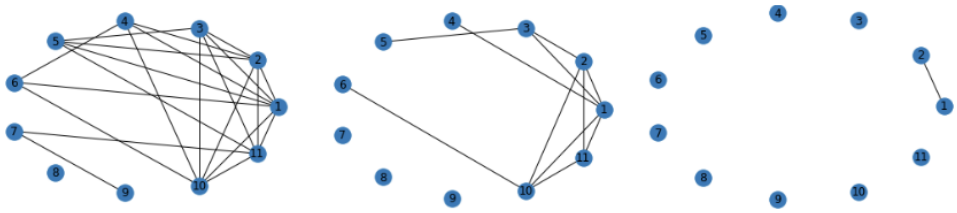


Figure 3.5: Graph realisations learnt using proposal density $Uniform$ $[0, 0.8]$, with constant $C = 100$ at cutoff $\tau = 0.4$, 0.6 and 0.9 (from left to right).

As I introduced in Chapter 2, COGENT (Bozhilova et al., 2020; Pardo-Diaz et al., 2021, 2022) addresses the consistency of graphs/networks and informs on a cut-off choice. We implement COGENT on our dataset $\mathbf{D}$ and extract results from this exercise. Figure 3.6 shows that COGENT suggests the optimal thresholds are approximately at 0.15 - i.e. thresholding the top 85% of calculated Pearson correlation coefficient values - to construct the network with the desired high consistency, given our dataset $\mathbf{D}$.

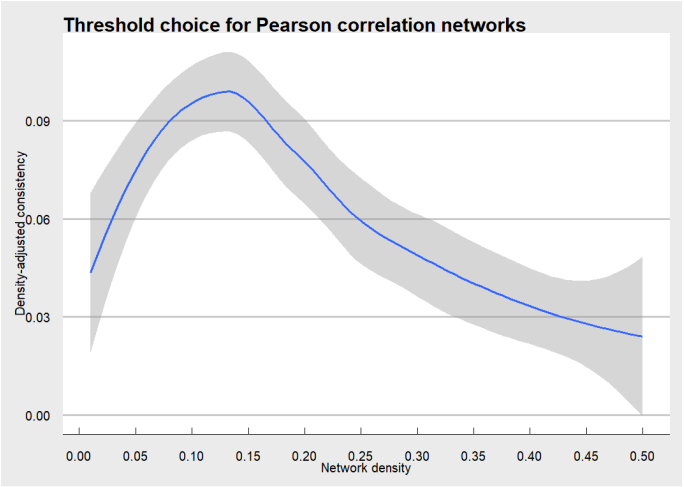


Figure 3.6: The threshold value about 0.15 yields the most consistent network, suggested by the COGENT (Bozhilova et al., 2020), using the our dataset **D**.

Similar to what we studied in Chapter 2, I produce RGGs and the networks for this dataset **D**, and compare the results obtained at the optimal τ (discussed in Section 2.4) with those obtained using COGENT. I recall that I had identified the RGG that is most robust (to changes in τ), as the value of the cut-off that minimises the rate of change of the log posterior of the RGG, with change in τ. Here, the RGG variable is learnt given a dataset. Figure 3.7 represents the plot of the logarithm of the RGG posterior probability and the corresponding slope γ, which is computed at a chosen τ, across the different τ values considered. Here, any RGG is learnt given the correlation structure of the time series data on the physiological parameters of the VOD patient - as recorded in their patient chart during a time interval that includes the time point of this patient undergoing a BMT. Said data of one patient is depicted in Table 3.1. We refer to this as the dataset **D** below.

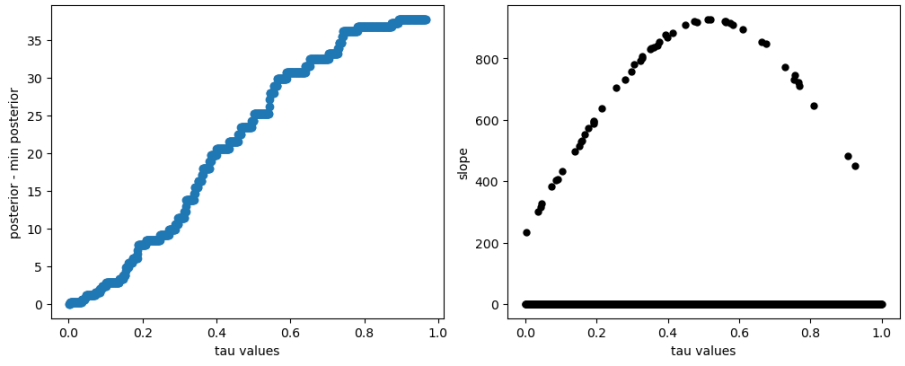


Figure 3.7: The left panel presents values of the log posterior probability of RGGs learnt for the data **D**, which are learnt at distinct values of τ s.t. the log posterior value is plotted against τ. The right panel shows the slope γ of the graph posterior, plotted against the corresponding τ value.

Figure 3.8 displays the results of comparing the optimal τ identified using the

slope γ computed for the data $\mathbf{D}$, with the COGENT-computed threshold for the same dataset. Recall that the COGENT threshold yields the most robust network; I refer to this threshold as $\tau = \omega$. My optimal τ on the other hand, focuses on the posterior of RGGs learnt at τ that are identified as the local maxima/minima in the γ profile, obtained by scanning across values of τ, using given data. Indeed, while the τ value relevant to a minima in γ identifies with the most-resilient (to changes in τ) RGG, the maxima of γ identifies the τ at which the most sensitive (to changes in the thresholding) RGG is produced - for the given dataset.

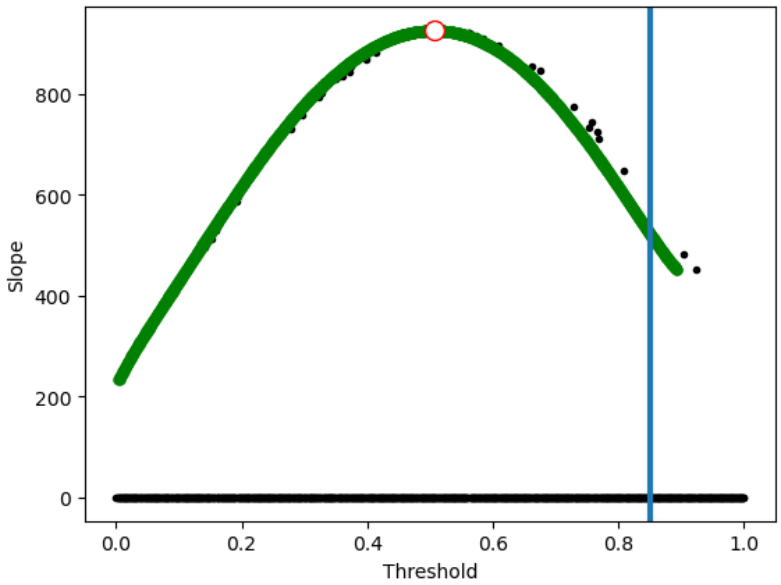


Figure 3.8: For data $\mathbf{D}$, the computed slope γ of the log posterior of the learnt RGG is plotted against threshold τ. A fit to the data is superposed in green, and the vertical line indicates the COGENT-suggested best threshold of $\omega = 1 - 0.15 = 0.85$ (thresholding the top 85% of calculated Pearson correlation coefficient values) that yields the most-consistent network for this dataset. The local maxima of γ, at $\tau_{max} \approx 0.51$, is labelled in red dots.

.

We can see that for this dataset $\mathbf{D}$, using the slope γ, can only identify the thresholding, at which the most-sensitive RGG is produced. This thresholding happens at $\tau = \tau_{max}$, i.e. the graph is most sensitive to the changes in τ, at the value τ_{max} of the thresholding. Note that for $\tau \geq 0.967$, the log posterior probabilities of RGGs become zero, i.e. no edge exists between nodes. Results of implementation of the best thresholding, on more datasets are included in later chapters.

Other thoughts: applying normalisation

I also considered normalising the log of the posterior of the RGG learnt at any τ - given the partial correlation matrix of data $\mathbf{D}$ - by the number of edges that comprise this learnt RGG.

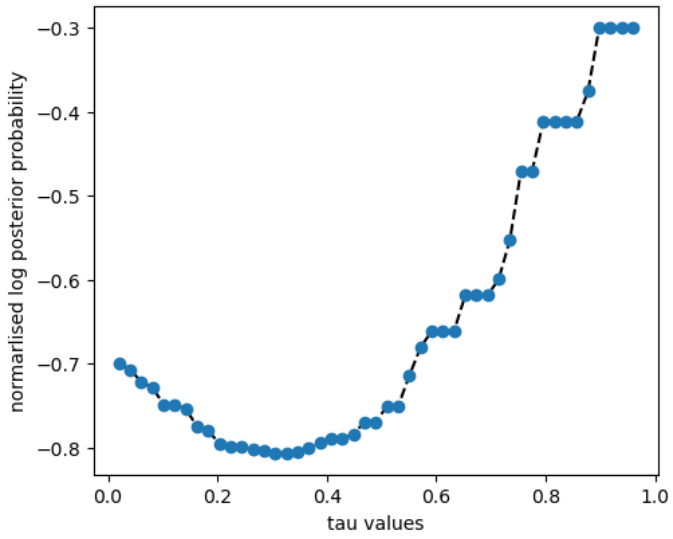


Figure 3.9: The plot displays the log posterior of the RGG variable $\mathcal{G}_{\boldsymbol{V},m}(\boldsymbol{\Psi}, \tau)$, conditional on the inter-column partial correlation matrix $\boldsymbol{\Psi}$ of time series dataset $\mathbf{D}$ (depicted in Table 3.1)), normalised by the number of accepted edges, plotted against the chosen $\tau \in (0, 1)$

However this learning on the normalised log posterior has limited interpretive value. Therefore I will not consider this learning with normalisation any further.

3.4 Discussions and Conclusions

In this chapter, I discuss my contribution to an undertaken Bayesian learning of RGG variables, given data on patients' sufferenace during their undergoing of a Bone Marrow Transplant, as manifest in 11 of their time-varying physiological parameters. Each such BMT-recipient is then rendered susceptible to VOD, which is a potentially terminal illness. My role in this work has been to contribute to the learning of the RGGs of the different patients in a cohort, such that the distance between the RGGs learnt for a patient, and an earmarked "reference patient", produces the considered patient's severity score of VOD, relative to that of the reference patient. Then on the scale in which the reference patient is assigned a score of 1, the score of the considered patient is computed. This way, scores of each patient in this cohort are computed. Such score learning then allows for populating a training dataset that comprises pairs of values of a patient's pre-BMT parameters, and their score. Using such a training set, we can then learn the relationship between the vector of the pre-BMT variables and the score variable, allowing for the prediction of the score of a test patient.

However, in this chapter, I limit discussions to my contribution to the project, which is the learning of the RGGs of each patient in the considered cohort (of 25 patients). Some of these results are included above in Figure 3.4.

It is appreciated that the computing of a reliable, relative score of any patient has

been possible only because the RGG that I learnt is a random variable, and therefore can be ascribed a probability distribution. Then a statistical distance between the posterior probability of one patient's graph, and that of another patient, can be easily computed, using the samples of the RGG realisations that are generated across the instances at which rejection sampling of each of the edge variables (of the RGG) is undertaken. It is the lossless nature of this inter-graph distance that allows us to use this distance as a reliable measure of the holistic difference between the information included in one patient's RGG, distinguished from another patient's learnt RGG. Future work will extend the proposed framework by refining the choice of threshold parameter and improving computational efficiency. Learning patient-specific RGG from clinical time-series data, constitutes a novel application of RGGs in the biomedical field. The applicability of this inter-graph distance to Healthcare is then clear via the application. I will discuss another such application in the next chapter.

One technical issue relevant to my RGG learning is the Bayesian inference on the edge variables. For this, I have primarily used rejection sampling, (as reported above), but have also tried Markov Chain Monte Carlo (MCMC), to sample from the relevant posterior distributions. The similarity of the results on graph learning demonstrates that the proposed RGG learning framework produces robust graph structures, across different proposal distributions and sampling strategies, which again validates the Bayesian approach to modelling uncertainty in graph edges.

A remaining issue involves the choice of the cut-off probability at which any RGG - drawn in a probabilistic metric space - is produced. This is my methodological contribution, which I support with empirical illustrations as well, inclusive of the illustrations on the data of different patients, as shown above. Again, it is the probabilistic treatment of the RGG that addresses the problem of recognising the cut-off probability that is optimal given the data at hand. I have addressed this by identifying the value of the cut-off probability τ, at which we can generate the most sensitive or most resilient RGG, given changes in the RGG with changes in τ. Such is made possible by identifying the τ at which the logarithm of the rate of change of the posterior of the RGG variable - with change in τ - is stationary. I have computed this when trying to identify the optimal τ for the RGG that I learnt given an available dataset, using which, another thresholding strategy (called COGENT) had computed the threshold value that gave rise to the "most resilient" network. In the following chapters, I will illustrate my identification of the optimal τ when learning RGGs in different real-world contexts.

Chapter 4

Learning Recovery Trajectories of Individual Patients Using Distance/divergence between Learnt RGGs

In real-world scenarios, we often need to compare distinct outcomes of an event that arise under different experimental conditions. For instance, when trigger variables assume one set of values, they generate a particular system response, which has been recorded as a time series. And a different set of trigger values will produce an alternative response, yielding a separate time series. In such cases, our goal is typically to compare the influence of the trigger variables across these differing conditions - this comparison is made by analysing the resulting time series. Notably, the system parameters do not need to cover the identical time intervals - the trigger variables may give rise to the responses of varying durations, leading to time series of different lengths. In this chapter, we address typical issues in real-world settings, the comparison of such multivariate time series data that are diversely-long in length with complex temporal evolution, which often display inhomogeneous correlation structures. Our ultimate goal is to construct a single scalar-valued random variable that captures the essential differences between two such multivariate time series, regardless of their differing lengths. To achieve this, we propose a framework based on graph representations of time series. For each pair of time series, we learn graphs from the data and compute distances or divergences between these graphs. As we introduced in previous chapters, each of such graphs is treated as a realisation of an RGG, and we estimate the probability distribution over such graph realisations conditioned on the observed time series data. And the distance between two graphs reflects a statistical distance between the respective probability distributions.

We illustrate our methodology using data collected from an automated e-health platform, *MIRA*. These data are recorded as patients perform mobility exercises during recovery from injury or critical illness that affects their ability to move. We undertake the learning of the graph variable given the relevant data, and discuss the implementation of the inter-graph distance. We also discuss the advantages of our

method over existing approaches used for assessing individualised patient recovery, using the data we have (these results have been published as Zhang et al. (2024)).

4.1 Introduction and existing studies

Modern medicine requires early and individual-specific diagnosis. Stroke, the second most common cause of death and the third-leading cause of adult disability (Feigin et al., 2022), often results in reduced mobility and pain. One critical aspect of rehabilitation is predicting a patients recovery trajectory to optimise therapy. Recovery quantification methods in the existing studies vary widely, with a lack of standardized definition. The International Classification of Functioning, Disability and Health (ICF) (Organization, 2001) has worldwide usage as a reference model in the assessment of functioning recovery of patients who have suffered from movement-impeding conditions, such as in the cases following traumatic injuries and stroke. Walton (2009) advanced 30 different primary operationalisations of recovery, in the context of acute Whiplash Associated Disorder (WAD) and Ritchie and Sterling (2016) summarised approaching the concept of recovery from WAD, based on self-reporting of relevant symptoms of the patient such as pain and dizziness, and psychological observations such as pain catastrophising (Casey et al., 2015), along with post-traumatic stress disorder symptoms. Kasch et al. (2001) and Ozegovic et al. (2009) considered recovery as parametrised by proxy measures such as reduction in working hours; sick leaves undertaken; and insurance claims.

In current literature, the learning of recovery trajectories using conventional methods, refers to grouping patients by recovery instead of looking at the recovery of individual patient, which lacks personalised analysis. Growth Mixture Modelling (GMM) and Latent Class Growth Analysis (LCGA) (Nagin, 1999; Muthén and Muthén, 2000; Nagin, 2005) are commonly used to classify patient recovery trajectories. Jung and Wickrama (2008) applied LCGA and GMM techniques to identify patient groups that are homogeneously recovered; Ram and Grimm (2009) summarised a four-step procedure to implement a GMM-based analysis; Panken et al. (2016) undertook GMM to describe the evolution of the intensity of lower-back pain; Lee et al. (2020) identified meaningful subgroups of patients who suffered from muscelo-skeletal trauma by using GMM and Walton et al. (2021) developed the Traumatic Injuries Distress Scale, and predicted the rate of recovery of people with muscelo-skeletal trauma. Machine learning-based assessment of individual movement correction has also been widely used in recent studies (Osgouei et al., 2020; Noureddin et al., 2019; Capecci et al., 2019; Paiement et al., 2014; Frangoudes et al., 2022). The patients' data is often collected by patients' responses to questionnaires (Walton et al., 2016; Herrera et al., 2007; Stark et al., 2013), or data obtained from clinics (Royse et al., 2010).

My contributions: Existing work falls short of the desired characteristics that I enumerate below:

1. Some data were collected from healthy individuals, rather than from patients,

and a universal benchmark is used for all patients, regardless of the variation of injury level.

2. An important shortcoming in the existing work is that there is no objective, data-driven and model-free evaluation of the state of a patient's recovery.

3. Current recovery assessments lack personalisation, relying on standardised classification models rather than the specific data recorded for each patient.

To address this, I applied the learning of RGGs for reliable assessment of the mobility recovery of patients with movement deficiencies, to construct each patient's individualised recovery trajectory, as they undergo physical rehabilitation. I use the time-series data generated by a patient from them undertaking a given exercising - as part of a rehabilitation programme - at each instance of the patient undertaking this exercise. The distance between graphs learnt using the time series data generated from undertaking an exercise at two consecutive instances, then informs on this patient's recovery between the time points at which the exercise was undertaken at these two instances. In other words, the distance then offers an objective, model-free and fully data-driven parametrisation of this particular patient's recovery between two given time points.

Using such a parametrisation of recovery, I construct the individualised recovery trajectory for this patient as they undertake this exercise. I perform similar recovery trajectory learning for each patient in a cohort, undertaking distinct exercises. Ultimately, I analyse the constructed recovery trajectories, and recommend the optimal exercise routines that are tailored to patients with varying degrees of mobility impairment.

My identification of an individual recovery trajectory is supplemented by the pre-therapy information about the patient's mobility status. In future work, we will augment each trajectory with pre-therapy mobility metrics and, through supervised learning, model the relationship between a patient's baseline status and the ensuing trajectory. This extension will enable patient-specific, pre-treatment forecasts of recovery under a given therapeutic programme, advancing the goal of precision rehabilitation.

4.2 Methodology

4.2.1 Data

Our dataset has been collected via an at-home rehabilitation enabled by virtual reality-assisted equipment, which is referred to as "telerehabilitation". Such telerehabilitation allows patients to exercise in a home environment and helps avoid travel difficulties for patients, especially for low-mobility patients. There is an increase over time in outpatient rehabilitation, as opposed to inpatient rehabilitation (Olasoji et al., 2022). The telerehabilitation equipment extracts information about movement biomarkers and collects such information using motion sensor devices, such that the time series data have been generated while a patient is undertaking

an exergame, that towards the construction of recovery trajectories - these exergames used in physical rehabilitation are designed to stimulate certain movements, which focus on improving physical activity. This is an important step in the attempt to automatically assess individualised physical functioning within a home environment. By invoking the telerehabilitation capacity, our method of learning an individualised recovery-trajectory can inform the patient and their therapists, of the progress on achieved recovery, obtained at an instance of rehabilitation. We are able to generate a reliable mobility recovery score and accordingly provide advanced recommendations regarding optimal exercise routines for patients with a given level of pre-therapy mobility deficiency, by learning such recorded movement data.

The data was secondary anonymised data, which was judged as not in requirement of ethics clearance, as stated in a letter with reference 41180-NER-Nov/2022-42403-1, signed by Prof. Simon Taylor at Brunel University London. The data was collected while a cohort of patients who were suffering mobility deficiencies due to a stroke and were undergoing physical rehabilitation - played exergames on the virtual rehabilitation platform *MIRA Rehab* (Rehab). Data consists of recordings of X, Y, Z coordinates of 20 joints of the body of a patient who is undertaking an exercise. The same is also referred to as the patient "playing an exergame". Typically, a patients movement is recorded for 3 minutes at a rate of 30Hz. In the cohort, 13 patients have played various games, with multiple repetitions for each of them and at various levels of difficulty, as presented in Figure 4.1.

	PATIENT ID													
GAME	3085	3070	3147	3071	3311	3327	3431	3462	3430	3165	3379	3011	3084	Total sessions by game
Airplane	26		26	6		20				1				79
Animals	12		8	13				21		3				57
Atlantis	24	55		10		20				2				111
Catch	76	55	53	48	35	32	26	24	22	7	4			382
Color Blocks	12		25											37
Color Clouds	15		12							1				28
Firefly	24	55		10	35	32	26	22	22	1	2			229
Flight Control										1				1
Follow	52		27	40	35	32	26	22	22	1	2			259
Forest Leaves	28		8											36
Izzy the Bee	26	58	12	38	17	12	26	24	22	7	2	3		247
Jugger	76	56	53	48	35	32	26	25	22	1	1			375
Matchday Striker	40		26	19	18					4				107
Nimble	75	57	35	48	35	31	26		22	1	1	1		332
Piano	15		24	18						1				58
Seasons			25											25
Space Ship	35	58	11	24	35	12	26	23	22	2	2	1	1	252
Star Find			26	19						1				46
Total sesions by subject	536	394	371	341	245	223	182	161	154	34	14	5	1	2661

Figure 4.1: Sessions played by each subject for each of the 18 games.

A game consists of a sequence of frames, with each frame capturing the 3D coordinates of each of 20 joints of the human skeletal structure. This translates into a sequence of time series, one for each dimension of each of the considered joints. Each game engaged the subject to execute a certain physical exercise or a combination of two or more exercises (for instance: knee flexion; knee extension; lateral flexion; shoulder abduction; frontal flexion; hip abduction; full body turn,

etc). Overall, we had 2,661 sessions recorded, which are not uniformly distributed amongst patients and games.

The e-platform also outputs a machine-learnt point gained for each exergame played, assigned by the platform itself - the in-built points are expected to function as an indicator of patient performance in executing the exergame. However as we will notice in the results, these points do not offer meaningful information.

Definition 4.2.1. *In our dataset, spatial and rate coordinates of each of the 20 joints are recorded, as the i-th patient plays the k-th game, at the j-th instance. The three spatial coordinates of the s-th joint, recorded at time t during playing an exergame, are denoted by variables $X_s(t), Y_s(t), Z_s(t) \in R, s = 1, ..., 20$. The total number of times the i-th patient plays the k-th game denote as $N_{i,k}$; $j = 0, 1, \ldots, N_{i,k}$, for $i = 1, \ldots, N_p$ and $k = 1, \ldots, N_g$.*

In general, since the i-th patient will not necessarily play all the available N_g games, we then might have some k, s.t. $N_{i,k} = 0$.

Definition 4.2.2. *The three spatial coordinates of the s-th joint, recorded at time t during playing an exergame, has denoted by variables $X_s(t), Y_s(t), Z_s(t) \in R, s = 1, ..., 20$.*

We then denote the location of the s-th joint to be

$$R_s(T) := \sqrt{(X_s(t))^2 + (Y_s(t))^2 + (Z_s(t))^2}.$$

Then we generate the time series data $\{r_1(t), r_2(t), \ldots, r_{20}(t)\}_{t=1}^{(t_{i,k,j}^{(max)})}$, i.e. we record the time series of the locations of each of the 20 joints, at time $T = t$.

Definition 4.2.3. *We generate the time series data on the location of each of the 20 monitored joints, i.e. we generate the time series data $\mathbf{D}_{i,k,j}$ that comprises values of $R_1(t), \ldots, R_{20}(t)$, for time $t \in \{1, 2, \ldots, t_{i,k,j}^{(max)}\}$ as the i-th subject plays the k-th game, in the j-th instance. So the dataset $\mathbf{D}_{i,k,j}$ has $t_{i,k,j}^{(max)}$ rows and 20 columns, i.e. $\mathbf{D}_{i,k,j}$ is a $t_{i,k,j}^{(max)} \times 20$-dimensional matrix.*

Remark 4.2.1. *The length of time by a subject takes to play a game in one instance can differ from the length of time taken by the same or a different subject to play the same or a different game in another instance.*

In Table 4.1, we present the first few lines of values of the location of joints of the body that are monitored on the e-platform MIRA, during the patient with reference 3071 played *Airplane*.

hipcentre	spine	head	...	shouldercentre
1.684013855	1.680728421	1.694426789	...	1.694013087
1.687542467	1.680152844	1.689994086	...	1.690999615
1.688034792	1.680716982	1.6897329	...	1.692213936
1.688174173	1.680867813	1.690434507	...	1.69244093
1.688240062	1.680919118	1.690826397	...	1.692456947
1.688247039	1.681416153	1.690978133	...	1.693617492
1.688279093	1.681536837	1.691291841	...	1.693789355
1.688287023	1.681549649	1.691346302	...	1.69374882
1.688258504	1.681548505	1.691332804	...	1.693702386
1.688319709	1.681569607	1.691362489	...	1.693638131
...	...	...	...	...

Table 4.1: The first few lines of the time series data on locations of the 20 joints of the body recorded at different time points, as Patient 3071 played the exergame *Airplane*.

We aim to capture the level of recovery of movement ability while i-th patient plays the k-th game in the j-th instance, compared to when they play the game at the $j-1$-th instance, $\forall j > 1;\ j \leq N_{i,k}$. In our work, we learn this recovery of movement by learning the difference between the correlation structure of the time series data, recorded at the j-th instance of playing the game, and that recorded at the previous instance of playing this game - the correlation structure given by the location of the 20 joints of the human body that are tracked during playing exergames, comprises information interplay between each pair of these 20 joints. A patient who is facing movement difficulties in a certain joint, might have to invoke the usage of other (nearby) joints to perform an action - there exists higher correlation amongst pairs of (nearby) joints in such a patient, than in the case for healthy individual with no or less severe movement difficulties, i.e. their inter-joint correlations are then lower in general. So we look at the correlation structure of the locations of the monitored joints as a marker of recovery of movement. To illustrate the correlation structure of our multivariate dataset, we learn the graphical model of the time series data on the spatial coordinates of each of 20 joints (monitored on the e-platform MIRA, on every instance when the i-th subject plays the k-th game), using the inter-column correlation matrix of our dataset. Namely, we compute the strength of the effect that is responsible for the dataset $\mathbf{D}_{i,k,j}$ to change into dataset $\mathbf{D}_{i,k,j^{/}}$, where the length of the two datasets $t^{(max)}_{i,k,j} \neq t^{(max)}_{i,k,j^{/}}$ in general, for $j \neq j^{/}$; $j, j^{/} = 1, \ldots, N_{i,k}$ - we compute this strength of the effect as a distance/divergence between the probability distribution of the graphs learnt condition on the inter-joint correlation matrix $\mathbf{\Sigma}^{(C)}_{(i,k,j)}$ of the dataset $\mathbf{D}_{i,k,j}$, and the graph learnt given the inter-joint correlation matrix of the dataset $\mathbf{D}_{i,k,j-1}$.

4.2.2 Learning/estimation of the correlation matrix

Definition 4.2.4. *The correlation matrix of this data* $\mathbf{D}_{i,k,j}$, $\mathbf{\Sigma}^{(C)}_{i,k,j} = [\rho_{s,s'}]$, *where* $\rho_{s,s'}$ *is the correlation between the random variables* R_s *and* $R_{s'}$, $s, s' = 1, \ldots, 20$. *The random variable* R_s *comprises the* s*-th column of the dataset* $\mathbf{D}_{i,k,j}$.

The number of rows of the dataset $\mathbf{D}_{i,k,j}$ and the dataset $\mathbf{D}_{i,k,j'}$ are unequal, which does not affect our Bayesian learning of the correlation matrices $\mathbf{\Sigma}^{(C)}_{i,k,j}$ and $\mathbf{\Sigma}^{(C)}_{i,k,j'}$, and therefore, inter-graph distance is robust to the difference in the number of rows in either dataset. As we introduced in Chapter 1, we can undertake MCMC-based learning to learn each element of the correlation matrix directly. However, in this case we have $(20 \times 20 - 20)/2 = 190$ distinct off-diagonal elements of the symmetric correlation matrix - it is extremely time-consuming to learn 190 (correlation) parameters using MCMC. It may also be possible to undertake parametrisation of the correlation matrix, using a covariance kernel. While considering the input variable we have in our case, namely the location of the joint at a given time point as the patient plays an exergame, the joint index has been randomly assigned. Therefore the difference between the indices of two joints might not necessary meaningful.

Instead of learning each element of the correlation matrix, we can therefore estimate each element, using the sample of values of the location variable R_s and that of $R_{s'}$, which includes across the $t^{(max)}_{i,k,j}$ rows of this dataset. We compute the unbiased estimator of the correlation $\rho_{s,s'}$ between R_s and $R_{s'}$. The partial correlation $\psi_{s,s'}$ is defined as follows:

Definition 4.2.5. *The unbiased estimator of the correlation* $\rho_{s,s'}$ *between* R_s *and* $R_{s'}$, *using the values* $r_{s,1}, \ldots, r_{s,M}$ *and* $r_{s',1}, \ldots, r_{s',M}$, *where* $M \equiv t^{(max)}_{i,k,j}$.

Then the inter-column correlation matrix $\mathbf{\Sigma}^{(C)}_{(i,k,j)}$ *of the dataset* $\mathbf{D}_{i,k,j}$, *can be estimated as* $\hat{\mathbf{\Sigma}}^{(C)}_{(i,k,j)} = [\hat{\rho}_{s,s'}]$

Definition 4.2.6. *The partial correlation* $\psi_{s,s'}$ *is computed using the estimated* $\mathbf{\Sigma}^{(C)}_{(i,k,j)}$ *- for the precision matrix* $\mathbf{\Theta} = [\theta_{s,s'}] \coloneqq (\mathbf{\Sigma}^{(C)}_{i,k,j})^{-1}$, *the partial correlation matrix* $\mathbf{\Psi} = [\psi_{s,s'}]$ *is hence*

$$\psi_{s,s'} = -\frac{\theta_{s,s'}}{\sqrt{\theta_{s,s}\theta_{s',s'}}}, s \neq s', \text{ and we have } \psi_{s,s} = 1 \text{ for } s = s'. \tag{4.2.1}$$

We present an excerpt of the inter-column correlation matrix estimated from the time series data generated by Patient 3071 playing *Airplane*, at the first instance, i.e.

$$\mathbf{\Sigma}^{(C)}_{3071,Airplane,1} =$$

$$\begin{bmatrix} 1 & 0.330917541 & -0.26122886 & ... & -0.165895188 \\ 0.330917541 & 1 & 0.68856342 & ... & 0.870462199 \\ -0.26122886 & 0.68856342 & 1 & ... & 0.874589366 \\ -0.319381464 & 0.544254133 & 0.692496065 & ... & 0.730793308 \\ -0.022265821 & 0.128763652 & 0.183350249 & ... & 0.151056752 \\ 0.16721271 & -0.094951013 & 0.02250426 & ... & -0.164855981 \\ -0.051876216 & -0.138002972 & -0.006260628 & ... & -0.100540177 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -0.165895188 & 0.870462199 & 0.874589366 & ... & 1 \end{bmatrix}$$

Then using this inter-joint-location correlation matrix, we compute the partial correlation matrix $\mathbf{\Psi}^{(C)}_{3071,Airplane,1}$, as:

$$\mathbf{\Psi}^{(C)}_{3071,Airplane,1} =$$

$$\begin{bmatrix} 1 & 0.229345539 & -0.150743042 & ... & -0.201536622 \\ 0.229345539 & 1 & -0.12561909 & ... & 0.941688165 \\ -0.150743042 & -0.12561909 & 1 & ... & 0.299565913 \\ 0.041884517 & 0.13053044 & 0.210506791 & ... & -0.023659251 \\ 0.024567646 & 0.052365098 & -0.128422074 & ... & -0.018381679 \\ 0.067497723 & 0.002226935 & 0.259867599 & ... & -0.070459859 \\ -0.096152059 & -0.023513329 & -0.111474649 & ... & 0.069071476 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -0.201536622 & 0.941688165 & 0.299565913 & ... & 1 \end{bmatrix}$$

4.2.3 Random graph variable

We learn the graphical model of the $t^{(max)}_{i,k,j} \times 20$-dimensional $\mathbf{D}_{i,k,j}$ dataset, using realisations of the random graph-valued variable that is defined on the vertex set $\boldsymbol{V} = \{R_1, \ldots, R_{20}\}$. This graph is drawn using the 20×20-dimensional partial correlation matrix $\mathbf{\Psi}^{(C)}_{i,k,j}$ that can be computed by Equation 4.2.1. This graph variable is a RGG drawn in a probabilistic metric space, s.t. any edge of this graph is a probability, and the binary edge $G_{s,s'}$ between the s-th and s'-th nodes exists, (i.e. $g_{s,s'} = 1$), if the probability of this edge variable exceeds a cut-off or threshold probability $\tau \in [0, 1]$ which we choose. Here $s, s' \in \boldsymbol{V}$ and $s \neq s'$, i.e. self-loops are excluded from the graph. The probability of the edge variable $G_{s,s'}$ is conditional on the partial correlation $\psi_{s,s'}$ between the random variables R_s and $R_{s'}$ that sit at the s-th and s'-th nodes, respectively. Thus, we write the edge probability of $G_{s,s'}$ conditional on $\psi_{s,s'}$ as $m(G_{s,s'}|\psi_{s,s'})$. As we already discussed in details in Chapter 2, the edge marginal stems from the modelling of the probability density of the magnitude of the partial correlation $\psi_{s,s'}$ between the variables R_s and $R_{s'}$, as $\mathcal{N}(g_{s,s'}, \upsilon_{s,s'})$, where $\upsilon_{s,s'}$ is the variance parameter relevant to this s, s'-th nodal pair. Weak priors on the two model parameters have been applied - Bernoulli(0.5) on $G_{s,s'}$ and Uniform on $\upsilon_{s,s'}$ - we write the joint posterior probability density of the

model parameters $G_{s,s'}$ and $\upsilon_{s,s'}$, given the correlation of the dataset. Such a joint is then proportional to $(2\pi\upsilon_{s,s'})^{-1/2}\exp(-(g_{s,s'}-\psi_{s,s'})/2\upsilon_{s,s'})$.

Remark 4.2.2. *Recall Definition 2.2.7 and Definition 2.2.10, we have the edge probability of $G_{s,s'}$ conditional on the partial correlation $\psi_{s,s'}$ between the nodes by integrating over the variance parameter and a closed-form expression for this conditional probability, $m(G_{s,s'}|\psi_{s,s'})$, as following:*

$$m(G_{s,s'}|\psi_{s,s'}) = K\left[\sqrt{\frac{2}{\pi}}\exp\left(\frac{-(S_{s,s'})^2}{2}\right) - (S_{s,s'})erfc\left(\frac{|S_{s,s'}|}{\sqrt{2}}\right)\right], \tag{4.2.2}$$

where $S_{s,s'} = |G_{s,s'} - |\psi_{s,s'}||$ as defined previously and the complementary error function $\mathrm{erfc}(\cdot) = 1 - \mathrm{erf}(\cdot)$. *$K$ is a positive global scale.*

The posterior of the RGG variable $\mathcal{G}_{i,k,j}(\boldsymbol{V},\tau)|\mathbf{D}_{i,k,j})$ that is learnt using the edge posterior $m(\cdot|\cdot)$, given the estimated inter-column correlation matrix $\boldsymbol{\Sigma}^{(C)}_{(i,k,j)}$ of data $\mathbf{D}_{i,k,j}$, is

$$\pi(\mathcal{G}_{i,k,j}(\boldsymbol{V},\tau))|\{\psi_{s,s'}\}_{s,s'\in\boldsymbol{V}}) = \prod_{s,s'\in\boldsymbol{V};k<k'} m(G_{s,s'}=g_{s,s'}|\psi_{s,s'})$$

, where the RGG is constructed without self-loops, and the edges are included independently of each other.

4.2.4 Sampling from the posterior

We draw N samples of the edge variable $G_{s,s'}$ that joins the s-th and s' nodes of the random graph variable, where the s-th node of this graph is associated with the location of the s-th of the 20 monitored joints, as a patient plays an exergame. We undertake rejection sampling of $G_{s,s'}$ from this closed-form marginal posterior probability $m(G_{s,s'}|\psi_{s,s'})$ (of edge $G_{s,s'}$), to generate N samples of $G_{s,s'}$ as $\{g^{(1)}_{s,s'},\ldots,g^{(N)}_{s,s'}\}$. Here, $g^{(n)}_{s,s'}$ can be 1 or 0, $\forall n = 1,\ldots,N$. The binary edge variable $G_{s,s'}$ attains a value $g^{(r)}_{s,s'}$ in the r-th sample, where $G_{s,s'} \sim Bernoulli(|\psi_{s,s'}|)$.

Remark 4.2.3. *Recall Definition 1.5.1, if we sample $X = x^*$ from a function $g(x)$, for a real $M \in (0,\infty)$ such that*

$$Cg(x) \geq f(x), \forall x;$$

For $u \sim U[0,1]$, and we accept the sampled x^* if

$$\frac{f(x^*)}{Cg(x^*)} \geq u,$$

then the probability distribution of this sampled variable x^*, is $f_X(x)$

We use a uniform proposal density in our implementation of rejection sampling, and we accept the proposed edge if and only if the acceptance ratio exceeds or equals $U \sim U[0,1]$, where the acceptance ratio of the r-th iteration is

$$\frac{m(G_{s,s'}^{(r-1)}|\psi_{s,s'}^{(r-1)})}{Cq(s,s')},$$

with a chosen value assigned the constant C, such that we have $Cq(s,s') \geq m(G_{s,s'}|\psi_{s,s'})$.

Definition 4.2.7. *We undertake rejection sampling of $G_{s,s'}$ to generate N sampled values of the edge variable. If the relative frequency for N to attain the value of 1 in this sample exceeds the chosen cut-off probability τ, we retain the edge between the k-th and the k'-th nodes in our final graphical model of data $\mathbf{D}$.*

The relative frequency of edge variable $G_{s,s'}$ in the sample $\{g_{s,s'}^{(1)}, g_{s,s'}^{(2)}, \ldots, g_{s,s'}^{(N)}\}$ is $\omega_{s,s'} = \sum_{n=1}^{N} g_{s,s'}^{(n)}/N$.

Hence in our graph, edge between the s-th and s'-th nodes exists, (i.e. $g_{s,s'} = 1$) if and only if $\omega_{s,s'} > \tau$, $\forall s, s';\ s < s';\ s = 1, \ldots, 20$.

In general, if this relative frequency of $G_{s,s'}$ that is generated using rejection sampling, exceeds the chosen cut-off probability τ, we retain the edge between the s-th and the s'-th nodes in our final graphical model $\mathcal{G}_{\boldsymbol{V},m}(\boldsymbol{\Psi}, \tau)$ that is learnt on the vertex set $\boldsymbol{V}$, using the edge probability $m(\cdot|\cdot)$ defined previously, for the inter-column partial correlation matrix $\boldsymbol{\Psi}$ of the dataset $\mathbf{D}_{i,k,j}$, for the chosen cut-off τ.

For example the patient with identity reference 3071, (or ID3071), playing the game *Airplane* 6 times, we use rejection sampling to draw a sample of size 50,000 of the edge variable $G_{s,s'}$, $\forall s' > s; s = 1, 2, \ldots, 20$, for each of the 6 instances. In other words, we learn each of the 6 graphical models for this subject playing this exergame on these 6 occasions.

4.2.5 Inter-graph distance formalisation

With samples of the edge variable $G_{s,s'}$ in hand, we are able to calculate the joint posterior of all edges of the relevant graph variable, given the partial correlation matrix of the time series data on the 20 joints, generated as the i-th patient plays the k-th game on this j-th instance, $\forall j = 1, \ldots, N_{i,k}^{(max)}$. We then compute the inter-graph distance/divergence between the joint posterior of all edges, i.e. posterior of the graph variable learnt using the data generated for the j-th instance, and that using the data obtained for the $(j-1)$-th instance, $\forall j = 2, \ldots, N_{i,k}^{(max)}$ - a statistical distance between the posterior probability of the graph variable conditional on the time series data $\mathbf{D}_{i,k,j}$, and that given data $\mathbf{D}_{i,k,j'}$.

As we introduced in Chapter 1, the Hellinger distance is an example of such a statistical distance.

Remark 4.2.4. *Recall Definition 2.3.1, the square of Hellinger distance between the probability density $p_1(x)$ and another density $p_2(x)$ (for $x \in \mathcal{X}$) is defined as :*

$$D_H^2(p_1, p_2) = \int_{\mathcal{X}} (\sqrt{p_1(x)} - \sqrt{p_2(x)})^2 dx.$$

Therefore, for the graph variables learnt for data $\mathbf{D}_{i,k,j}$ and $\mathbf{D}_{i,k,j'}$, given that the graph posterior probability is computed at each of the N number of samples generated using rejection sampling, we compute the Hellinger distance between these two posterior.

Definition 4.2.8. *Given the graph posterior probability of graph variables learnt for data $\mathbf{D}_{i,k,j}$ and $\mathbf{D}_{i,k,j'}$, the square of the discretised Hellinger distance between these posterior probabilities is*

$$\begin{aligned} &D_H^2(\mathcal{G}_{i,k,j}(\boldsymbol{V},\tau), \mathcal{G}_{i,k,j'}(\boldsymbol{V},\tau)) \\ &= \frac{\sum_{r=1}^{N} \left(\sqrt{\pi(\mathcal{G}_{i,k,j}(\boldsymbol{V},\tau)|\mathbf{D}_{i,k,j})^{(r)}} - \sqrt{\pi(\mathcal{G}_{i,k,j'}(\boldsymbol{V},\tau)|\mathbf{D}_{i,k,j'})^{(r)}} \right)^2}{N}, \end{aligned} \tag{4.2.3}$$

where $\pi(\mathcal{G}_{i,k,j}(\boldsymbol{V},\tau)|\mathbf{D}_{i,k,j})^{(r)}$ is the graph posterior given the data $\mathbf{D}_{i,k,j}$, computed with the r-th sample of the edges of the graph, $\forall r = 1, \ldots, N$.

We compute the Hellinger distance $D_H(\mathcal{G}_{i,k,j}(\boldsymbol{V},\tau), \mathcal{G}_{i,k,j-1}(\boldsymbol{V},\tau))$ between graph posterior given the data $\mathbf{D}_{i,k,j}$ and $\mathbf{D}_{i,k,j-1}$, for the i-th subject playing the k-th game, for all i and k relevant in our data to $j > 1$, i.e. for all subjects who play a game multiple times.

Definition 4.2.9. *We can also compute the KL divergence between the posteriors of the graphs $\mathcal{G}_{i,k,j}(\boldsymbol{V},\tau)$ and $\mathcal{G}_{i,k,j-1}(\boldsymbol{V},\tau)$, given the data $\mathbf{D}_{i,k,j}$ and $\mathbf{D}_{i,k,j-1}$, as:*

$$\begin{aligned} &D_{KL}(\mathcal{G}_{i,k,j}(\boldsymbol{V},\tau), (\mathcal{G}_{i,k,j-1}(\boldsymbol{V},\tau)) \\ &= \sum_{r=1}^{N} \pi(\mathcal{G}_{i,k,j}(\boldsymbol{V},\tau)|\mathbf{D}_{i,k,j})^{(r)} log\left(\frac{\pi(\mathcal{G}_{i,k,j}(\boldsymbol{V},\tau)|\mathbf{D}_{i,k,j})^{(r)}}{\pi(\mathcal{G}_{i,k,j-1}(\boldsymbol{V},\tau)|\mathbf{D}_{i,k,j-1})^{(r)}} \right), \end{aligned} \tag{4.2.4}$$

where the posterior is computed of the graph variable given the datasets generated by the i-th subject playing the k-th game on successive instances, i.e. at the $j-1$-th and j-th instances, for $j = 2, \ldots N_{i,k}^{(max)}$.

Since the posterior of a graph given the time series data on the 20 joint locations will comprise the joint posterior of all edges between $(20^2 - 20)/2$ nodal pairs, the posterior of the graph variable $\mathcal{G}_{i,k,j}(\boldsymbol{V},\tau)$ is a small number, which leads to the distance/divergence between the posterior of $\mathcal{G}_{i,k,j}(\boldsymbol{V},\tau)$ given data $\mathbf{D}_{i,k,j}$, and of $\mathcal{G}_{i,k,j-1}(\boldsymbol{V},\tau)$ given data $\mathbf{D}_{i,k,j-1}$, is small as well. To avoid working with such small numbers, we scale the posterior of any random graph variable, learnt using any data that we consider in our work, by 10^{15}, when we compute the Hellinger distance

between any pair of graph posteriors, given any pair of datasets. Similarly, we scale any such posterior value with 10^{25} when we compute the KL divergence. Table 4.2 shows the scaled Hellinger distance and KL divergence between the posterior of the graph variable given the partial correlation structure of the time series data generated when the patient with ID3071 successively plays of the game *Airplane*.

Successive instances	Hellinger distance	Kullbeck-Leibler Divergence
(1,2)	0.10604219	0.10995603
(2,3)	0.10589288	2.5307512e-5
(3,4)	0.0040824151	2.4243133e-6
(4,5)	0.017579886	2.3849898e-3
(5,6)	0.022596656	1.1390906e-3

Table 4.2: Hellinger distance and Kullback-Leibler divergence of patient ID3071 playing the exergame *Airplane* between posterior of the graph variable learnt on successive instances of playing the exergame.

4.2.6 Construction of recovery trajectories

We compute the inter-graph distance/divergence between adjacent instances, by the posterior probability of a graph variable learnt given time-series datasets on joint locations generated by the patient playing a given exergame at successive instances. And such an inter-graph distance computed between the j-th and the $j-1$-th instances of the i-th patient playing the k-th game indicate the movement recovery attained by this patient between these two instances of playing the exergame, for $j = 2, \ldots, N_{i,k}^{(max)}$. We will then treat this movement recovery as the difference between the scores attained by adjacent instances, as we are defining below.

Definition 4.2.10. *We assign a Mobility Recovery Score or MRS, $\mathcal{MRS}_{i,k}(j)$ attained by i-th patient when playing the k-th game in the j-th instance, and a $\mathcal{MRS}_{i,k}(j-1)$ attained in the $j-1$-th instance of playing this game.*

Then the difference between $\mathcal{MRS}_{i,k}(j)$ and $\mathcal{MRS}_{i,k}(j-1)$ is defined as proportional to the movement recovery, namely, the inter-graph distance/divergence between the random graph variables learnt given the datasets $\mathbf{D}_{i,k,j}$ and $\mathbf{D}_{i,k,j-1}$,

Definition 4.2.11. *We denote the score as $\mathcal{MRS}_{i,k}^{(Hell)}(\cdot)$ when we compute the MRS using the Hellinger distance. On the other hand, when we use the KL-divergence to give the value of the MRS, we denote the score $\mathcal{MRS}_{i,k}^{(KL)}(\cdot)$.*

We can model the difference between $\mathcal{MRS}_{i,k}^{(Hell)}(j)$ and $\mathcal{MRS}_{i,k}^{(Hell)}(j-1)$ as proportional to the Hellinger distance between the graph variables learnt given the time series data $\mathbf{D}_{i,k,j}$ and $\mathbf{D}_{i,k,j-1}$ respectively, i.e.

$$\mathcal{MRS}_{i,k}^{(Hell)}(j) - \mathcal{MRS}_{i,k}^{(Hell)}(j-1) \propto D_H(\mathcal{G}_{i,k,j}(\boldsymbol{V},\tau), \mathcal{G}_{i,k,j-1}(\boldsymbol{V},\tau)).$$

We define the value of 1 for the proportionality constant.

Also, $\mathcal{MRS}_{i,k}(j) - \mathcal{MRS}_{i,k}(j-1)$ is the difference in the MRS between the j-th and $j-1$-th instances of playing the k-th game, is the same as the rate of change of MRS at the j-th instance of playing this game.

Definition 4.2.12. *The i-th patients rate of change of mobility improvement at the j-th instance of playing the k-th game, using Hellinger distance is defined as:*

$$D_H(\mathcal{G}_{i,k,j}(\boldsymbol{V},\tau), \mathcal{G}_{i,k,j'}(\boldsymbol{V},\tau)),$$

such that their MRS at this j-th instance of playing this game is

$$\mathcal{MRS}_{i,k}^{(Hell)}(j) = D_H(\mathcal{G}_{i,k,j}(\boldsymbol{V},\tau), \mathcal{G}_{i,k,j'}(\boldsymbol{V},\tau)) + \mathcal{MRS}_{i,k}^{(Hell)}(j-1),$$

where $j = 2, \ldots, N_{i,k}^{(max)}$.

We applied the same definition via using the KL-divergence to compute the inter-graph distance, we set the MRS of the i-th patient at the j-th instance of playing the k-th game to be:

$$\mathcal{MRS}_{i,k}^{(KL)}(j) = D_{KL}(\mathcal{G}_{i,k,j}(\boldsymbol{V},\tau), \mathcal{G}_{i,k,j'}(\boldsymbol{V},\tau)) + \mathcal{MRS}_{i,k}^{(KL)}(j-1),$$

with $j = 2, \ldots, N_{i,k}^{(max)}$.

We set $\mathcal{MRS}_{i,k}(1) = 0$ originally.

We plot $\mathcal{MRS}_{i,k}^{(\cdot)}(j)$ against j, which constitutes the recovery trajectory for the i-th patient, as they play the k-th game.

Table 4.3 shows the MRSs that are computed using the distance/divergence calculated in Table 4.2, which were generated from the time series data when the patient with ID3071 successively plays of the game *Airplane*, along with the points assigned by the e-platform MIRA originally.

Instance	MRS via Hellinger	MRS via KL	Points assigned by MIRA
1	0.05302109	0.05497801	230
2	0.15906328	0.16493404	255
3	0.26495617	0.16495935	290
4	0.26903858	0.16496177	360
5	0.28661847	0.16734676	395
6	0.30921512	0.16848585	315

Table 4.3: MRS calculated using the Hellinger distance and KL Divergence displayed in Table 4.2, and points assigned by e-platform MIRA, for Patient ID3071 playing the exergame Airplane on 6 instances

We can then visualise the recovery trajectories as the variation of the computed MRS against the index of the occasion of a given patient playing a given game. We learnt MRS using either the Hellinger distance or the KL divergence, and thereby learnt the recovery trajectory, and compared our MRS against the points that are

assigned by the e-platform, across instances in which a game is played. Figure 4.2 shows examples of these recovery trajectories for patient ID3071, playing Airplane on 6 occasions.

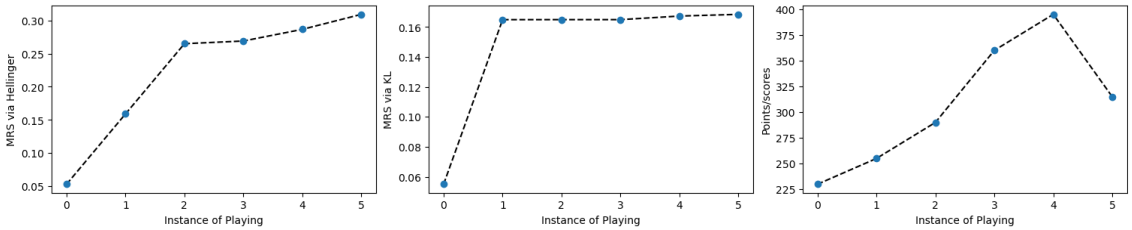


Figure 4.2: Figure displaying plots of performance of playing the game Airplane, by patient ID3071, against the index of the instance of playing this game. On the left, the plot of MRS computed using the Hellinger distance is plotted against the index of the instance of playing. In the middle panel, the MRS computed using Kullback-Leibler divergence is plotted on the 6 sessions during which this patient plays this game. On the right, points assigned by MIRA platform for playing the game on 6 instances are plotted against the instance index.

4.3 Gist of the method

As we defined previously, in order to accomplish the learning of the recovery trajectories of patients who play exergames and the data have been recorded on the e-platform *MIRA*, we learn the graphical model of the time series data generated as a patient plays an exergame. Then we compute the inter-graph distance/divergence between the posterior probabilities of the pair of graphs, generated by two successive instances of playing this exergame - we compute the marginal posterior of the edge of the graph variable, conditional on the estimated correlation structure of the dataset and undertake rejection sampling on such marginal posterior, to construct the graphical model of the relevant dataset.

- We learn the graphical model to comprise realisations of a random graph variable. We learn this graph variable, condition on the correlation structure of the time series data on locations of the 20 joints, given the i-th patient plays the k-th game, in the j-th instance.

- Then we define a statistical distance/divergence between the probability distribution of the graph variable learnt given the time series dataset generated by playing the k-th game in the j-th instance, and the graph learnt given the data generated by playing the game in the $j-1$-th instance. We do this for all instances relevant to the playing of the k-th game by the i-th patient, i.e. $\forall j > 1;\ \ j \leq N_{i,k}$.

- The distance computed for the $j, j-1$ pair is then a recovery score obtained between the playing of this game in the j-th instance and the previous instance,

which can be both positive and negative. Using this recovery score (we learnt MRS via using either the Hellinger distance or the KL divergence), we can generate a recovery trajectory for the i-th patient playing this k-th game. It is a clear visualisation of the recovery that the i-th patient has achieved, by playing the j-th instance of the k-th game - we can visualise the recovery trajectories by plotting learnt MRS against the index of the occasion of a given patient playing a given game.

4.4 Results

Previously in this chapter, we have discussed our concept of computing a parameter that carries as much relevant information as possible about the time-series datasets we have, namely, the progress of the patient recovering their movement ability, as they play an exergame. We have defined such a parameter as the distance or divergence between the graphical models that are learnt given the respective time series data. We conduct such inter-graph computation relevant to each patient in the cohort, playing each of the games, on successive instances - we will compute the inter-graph distance for each pair of graphs - learnt given the successive pairs of time series data - to construct a recovery trajectory for each patient playing an exergame.

4.4.1 Graphical Models

In Figure 4.3 we present the graphical models that are learnt given the joint location time series datasets, produced, as patient ID3071 plays *Airplane* on six different instances. Graphical models learnt given data produced by the patient ID3311 playing game *Follow*, are presented in Figure 4.4.

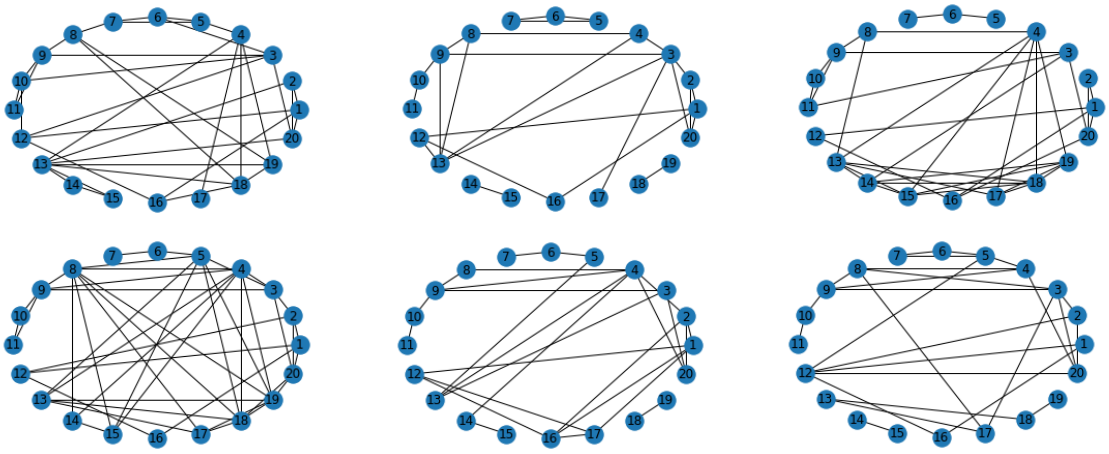


Figure 4.3: Graphical models learnt given data recorded for patient ID3071 playing game Airplane on 6 successive instances, in the order of the instances of playing.

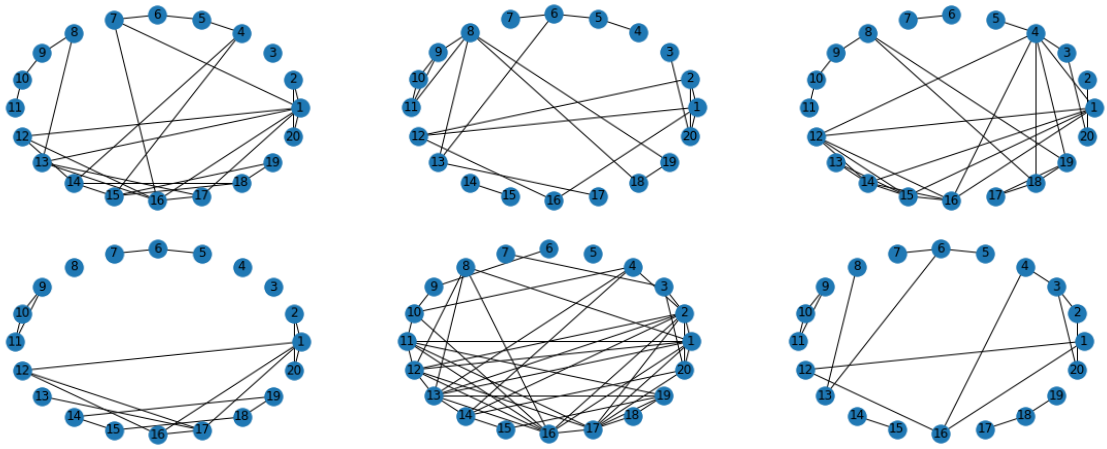


Figure 4.4: Graphical models learnt given data recorded for the patient ID3311 playing game Follow, on 35 successive instances, 6 of the learnt graphs presented here, in the order of the instance of playing.

To show the inference on the basis of which I learnt the RGGs shown above, the following Figure 4.5, I present the heatmaps of the frequency fraction of the edge variable $G_{ij} = 1, \forall i, j = 1, ..., 20$, in each iteration in Rejection sampling, i.e. edge $G_{ij} = 1$ I got a '1' from rejection sampling of a given edge ij, given data for the patient ID3071 playing game *Airplane*, as we show in Figure 4.3.

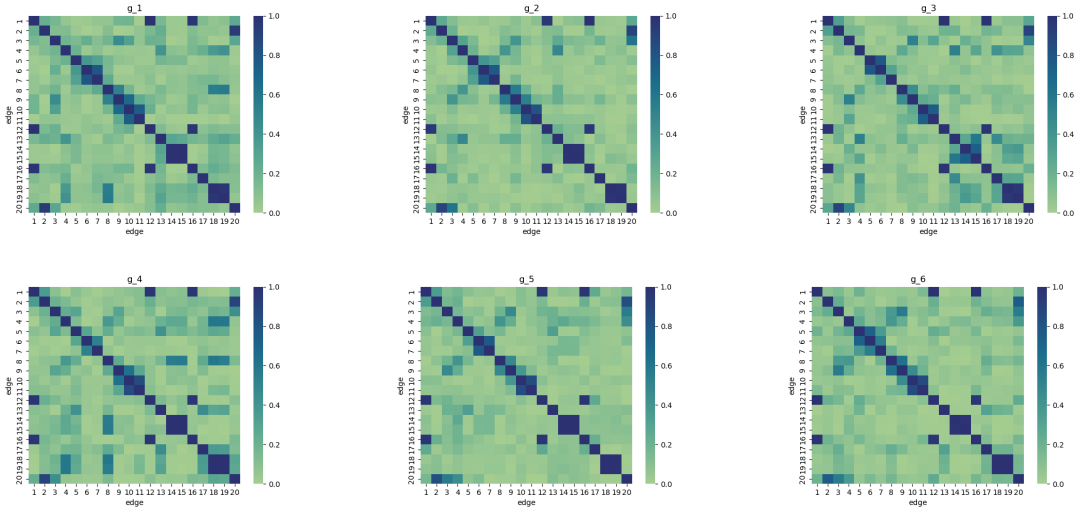


Figure 4.5: The heatmaps of the frequency of the edge appear in Rejection sampling, given data recorded for the patient ID3071 playing game Airplane, on 6 successive instances

4.4.2 Recovery Trajectories and MRS

Having built the graphs, we now construct the recovery trajectories for each patient as they play a given game. The following figures show the evolution in the learnt MRS as the patient repeatedly plays a game, across all instances of playing it. In Figure 4.6 and Figure 4.7 we displays recovery trajectories of the patient ID3085 playing game *Catch* 76 times, and the patient ID3327 playing game *IzzytheBee* 12 times from our learnt graphical models. For each patient, we show the comparison of the rate of change for MRS and the MRS, using the Hellinger distance, KL divergence, along with the scores assigned by MIRA.

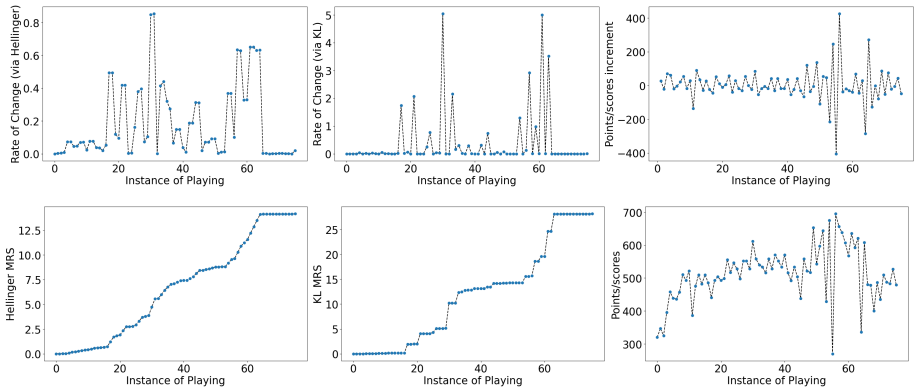


Figure 4.6: The comparison of computing scores of the patient ID3085 playing the game Catch 76 times. The first column displays plots of the rate of change of MRS (top) and MRS (bottom), across instances. Hellinger distance has been applied to compute the distance between posteriors of the graph variables using data generated at successive instances of playing the game. The second column shows the results for the inter-graph divergence computed using the Kullback-Leibler divergence. The plots in the 3rd column are the points provided by the e-platform MIRA, in successive instances.

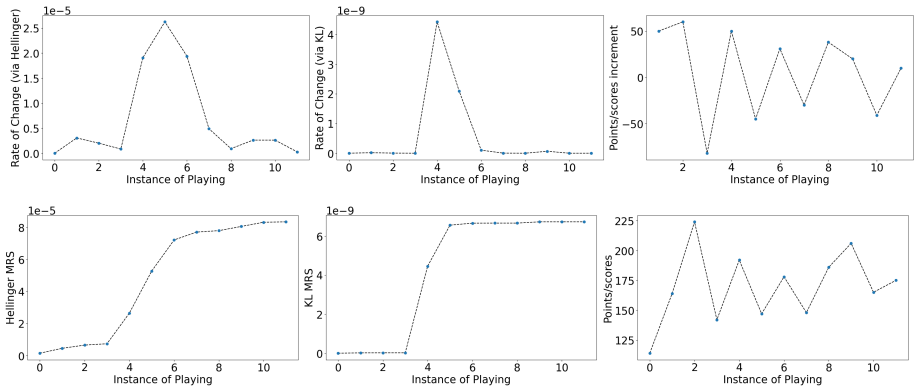


Figure 4.7: Same as for Figure 4.6, with results displayed for the patient ID3327 playing the game IzzytheBee 12 times.

From Figure 4.6 and Figure 4.7, the MRS using Hellinger distance and KL exhibit clear, robust trends and these trends remain consistent regardless of the number of game-play instances or the duration of each session. In contrast, the native point-scoring mechanism in MIRA yields highly variable performance indicators, obscuring any meaningful recovery pattern.

Furthermore, while both Hellinger- and KL-based MRS trajectories generally align, their mathematical properties produce distinct local behaviours. The Hellinger distance, as a true metric, is always non-negative and thus ensures that successive MRS values do not decrease. However, the KL divergence may take negative values when the posterior distributions of successive graph models become more similar, allowing the MRS to decrease between trials. This non-monotonicity is observed in several subjects (e.g., ID3147 in *Follow* and ID3327 in *Nimble*). Nevertheless, the overall agreement between the two measures suggests that most patients demonstrate genuine improvements in movement ability with repeated gameplay.

In Appendix D, I present all the constructed recovery trajectories for each patient in the cohort playing each exergame. Overall, the derived recovery trajectories reveal that most patients experience measurable improvements in physical function throughout the training program - an effect not readily discernible from the MIRA platform's native statistics. This conclusion is further corroborated by MRI profiles and standardised functional assessments administered at the start and conclusion of the approximately three-month intervention, demonstrating significant gains in motor performance (Firwana et al., 2022). In general, our methodology offers a clear, intuitive visualisation and interpretation of patient recovery dynamics.

4.4.3 Robustness of the graph learning

To assess the robustness of our graph-learning procedure under different proposal densities in the rejection sampler, we learn the recovery trajectories of the patient ID3071 playing the game *Airplane* on six instances, using both a Bernoulli and a Uniform proposal. For each pair of successive graphs, we drew a sample of size $500,000$ for $g_{s,s'}$, $\forall s' > s; s, s' = 1, 2, \ldots, 20$, with a $Bernoulli(|\psi_{s,s'}|)$proposal,and separately, with $Uniform[0, 1]$ proposal. We then computed the inter-graph distances and derived the corresponding recovery trajectories using both the Hellinger distance and the KL divergence. As shown in Figure 4.8, the two trajectories nearly coincide, demonstrating that our graph-learning implementation is insensitive to the choice of proposal density.

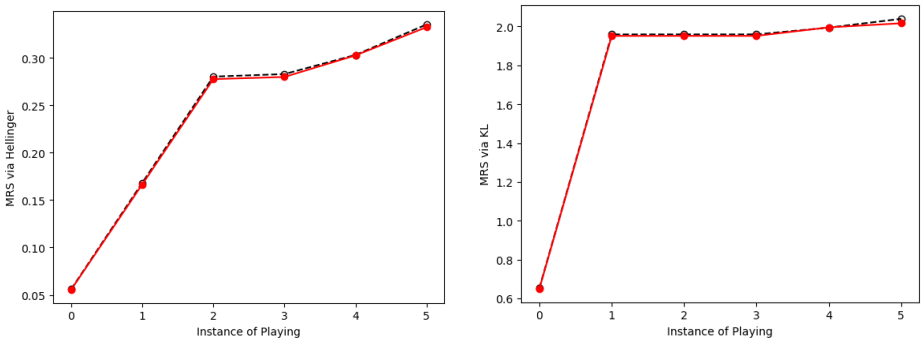


Figure 4.8: Comparison of recovery trajectories for patient ID3071 playing Game Airplane in 6 instances, learnt using Uniform proposal (in black dashed lines and unfilled circle markers) and the Bernoulli proposal (in red solid lines and red circle markers).

4.4.4 Discussion of results

We aggregate recovery trajectories from across the patient cohort to advance recommendations for the most appropriate exergame, given each patient's initial mobility impairment, when the physical rehabilitation programme is just about to begin - it is the identification of the pre-rehab mobility level. We use the initial automatic score (assigned by e-platform MIRA) of the i-th patient undertaking the k-th exergame at the first instance, $\mathcal{S}_{i,k}(1)$, as a proxy for pre-rehabilitation mobility - the lower is $\mathcal{S}_{i,k}(1)$, the higher is the mobility impediment faced by the i-th patient.

Although the complete recovery trajectory - a vector-valued approximation of a function - provides the richest depiction of patient progress, we summarise this information with a single scalar. Here, we focus on the trajectory derived from the KL-based MRS and define the recovery parameter.

Definition 4.4.1. *We invoke a scalar-valued recovery parameter $\alpha_{i,k}$ of the i-th patient playing the k-th exergame. We define*

$$\alpha_{i,k} := \frac{\mathcal{MRS}_{i,k}^{(KL)}(N_{i,k}^{(max)}) - \mathcal{MRS}_{i,k}^{(KL)}(1)}{\mathcal{MRS}_{i,k}^{(KL)}(N_{i,k}^{(max)})}.$$

Here we have the recovery parameter $\alpha_{i,k}$ is in $[0, 1]$.

We then plot $\alpha_{i,k}$ against $\mathcal{S}_{i,k}(1)$, for all those patients who played the k-th game at least four times. In Figure 4.9, we show the plots for the 16 relevant exergames, *Airplane, Animal, Catch, ColourBlocks, ColourClouds, Firefly, Follow, ForestLeaves, IzzytheBee, Jugger, MatchdayStriker, Nimble, Piano, Seasons, SpaceShip, StarFind*, which were undertaken by 10 patients.

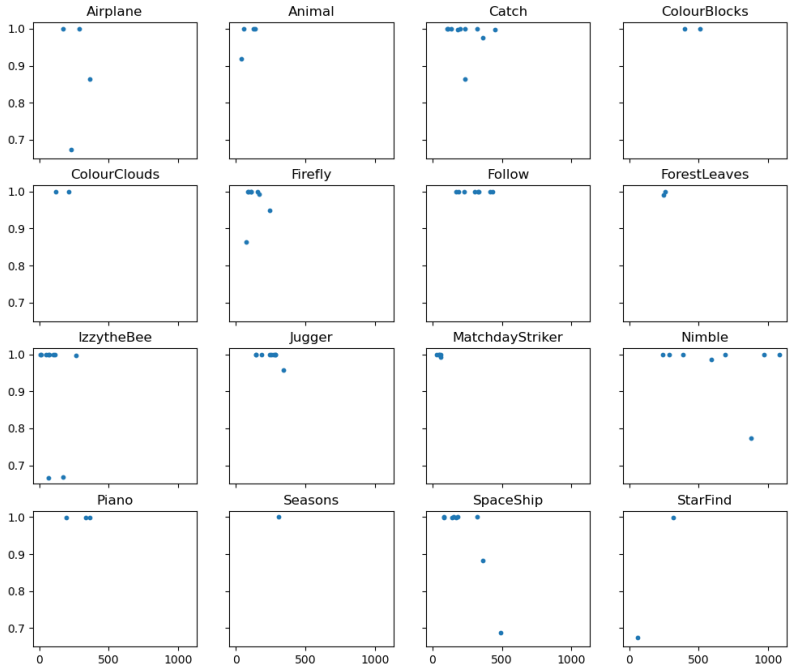


Figure 4.9: Plots of computed recovery parameter of all patients playing a particular exergame, against the points scored by a patient on the MIRA e-platform at the first instance when playing the considered exergame. The recovery parameters are plotted only for those patients who played this exergame in at least four instances.

We focus on:

1. When comparing any two exergames, the one that yields a higher recovery parameter α at lower initial automatic scores - that is, for patients with more severe mobility impairments - is deemed more efficient.

2. We can then identify and recommend the best exergame for a given level of initial impairment (as parametrised by the initial score), by using this figure.

Therefore according to points above, based on our cohort analysis, we rank exergames by their rehabilitative efficacy relative to patients baseline mobility impairments - for patients presenting with the most severe impairments (lowest initial scores), the top three games are *IzzytheBee; MatchdayStriker; Animal*; *Firefly* and *Spaceship* also perform strongly. In this dataset, *Catch; Follow; Jugger* also exhibit consistently high recovery parameters, given a wide range of severity of mobility impediments. Patients with the mildest impediments are recommended to the games *Catch; Follow; Jugger*. Notably, while *Nimble* produces uniformly high recovery across all severity levels, its lack of discrimination makes it a less informative choice for tailoring rehabilitation to a patient's initial condition.

4.5 Further study on the choice of cut-off τ in Graph Model

We recall the basic concepts of RGG, throughout our learning - for an RGG in a probabilistic metric space such that its edge is a probability distribution over positive support, the edge exists if this probability exceeds a cut-off probability τ. As we introduced in the previous chapter (see Chapter 2), our approach to learning on the cut-off probability τ from the posterior probability of the graph variable condition on the partial correlation, given by the multivariate dataset. Although in the chapter, as we parametrised the movement recovery by the difference between the posterior probability of the graph variable conditional on the time series data directly, i.e. we are including all edge probabilities in our calculation, such that our parametrisation will not rely on the choice of cut-off τ. However, it is worth applying the τ learning to the various datasets.

According to the approach that we mentioned in the previous chapter, we learnt the posterior probability of the graph variable given the correlation of the multivariate dataset. Recall the Patient ID3085 who played game Catch, we have learnt 76 graphs corresponding to 76 instances. By looking at the graph generated by the first instances, the following Figure 4.10 represents the graphs generated at different values of the cut-off τ that we have chosen.

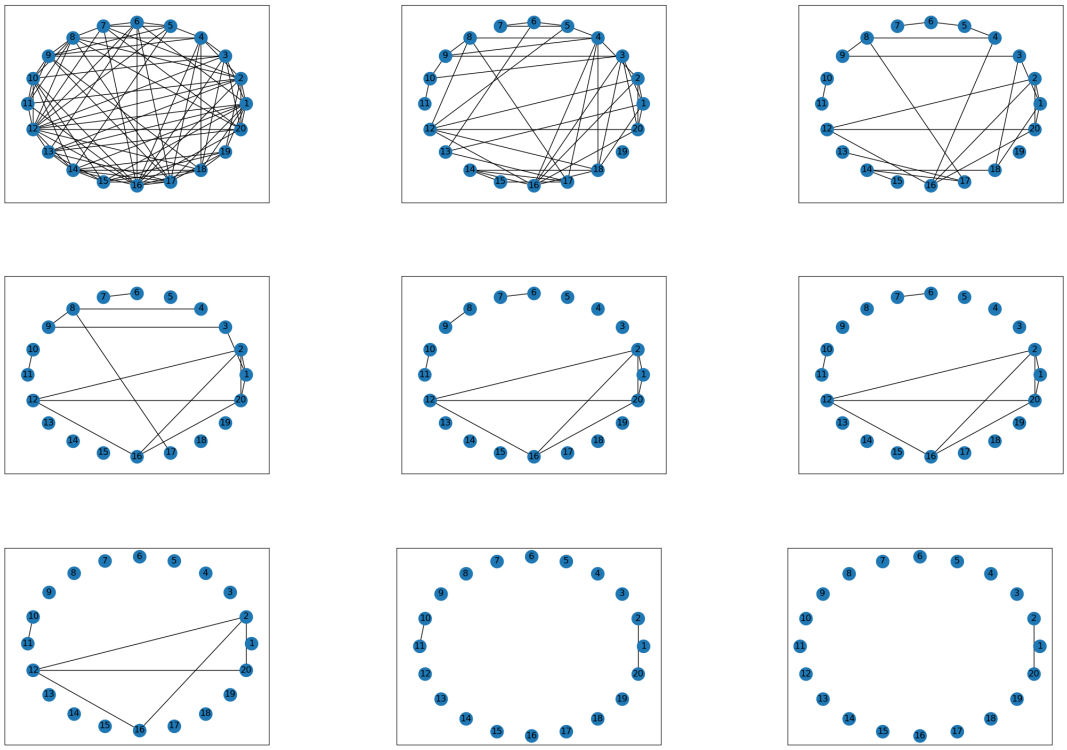


Figure 4.10: Graphical models learnt given data recorded for patient ID3085 playing game Catch at first instance, here $\tau = 0.1, 0.2, , 0.9$

Similar to previous examples shown earlier in my thesis, I compare the results using our γ thresholding method that learnt RGGs and results using COGENT. Figure 4.11 represents the plot of the log posterior probability of RGGs and the corresponding slope γ at a chosen $\tau \in (0, 1)$, given the dataset of the patientID 3085 playing game Catch at first instance.

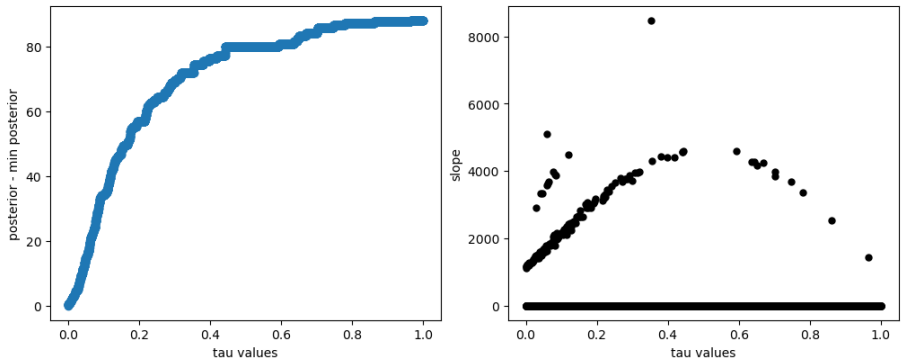


Figure 4.11: The left panel presents the log posterior probability of RGGs learnt for the patientID 3085 playing game Catch at first instance, which are learnt at distinct values of τ s.t. the log posterior value is plotted against τ. The right panel shows the slope γ of the graph posterior, plotted against the corresponding τ value.

Figure 4.12 shows that COGENT suggests the optimal thresholds are approximately at 0.05 - i.e. thresholding the top 95% of calculated Pearson correlation coefficient values - to construct the network with the desired high consistency. Notice that in this case, COGENT also provide the threshold that yields the least consistency, approximately at 0.45.

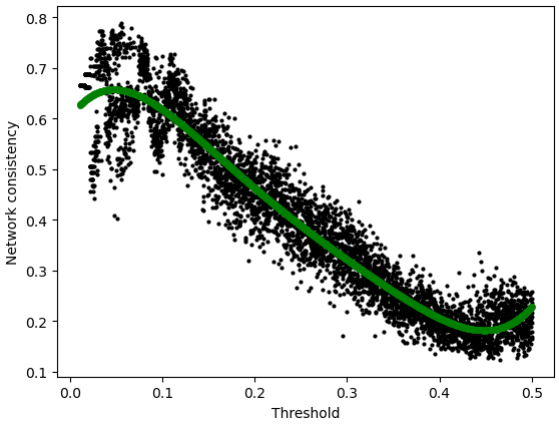


Figure 4.12: The threshold value about 0.05 yields the most consistent network (fitted values in green), suggested by the COGENT (Bozhilova et al., 2020), using the data of the patientID 3085 playing game Catch at first instance.

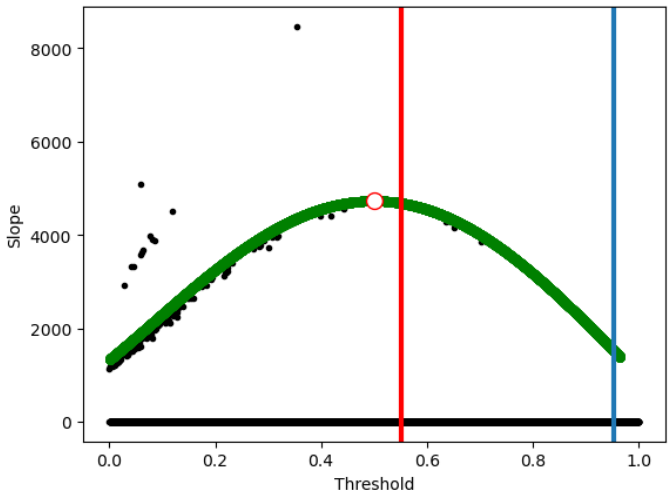


Figure 4.13: For same data, the computed slope γ of the log posterior of the learnt RGG is plotted against thresholds τ. A fit to the data is superposed in green. The blue vertical line indicates the COGENT suggested best threshold of $\omega = 1 - 0.04722222 \approx 0.95$ that yields the most-consistent network for this dataset, and red vertical line indicates the COGENT suggested the threshold of $1-0.4486036 \approx 0.55$, yields least network consistency. The local maxima of γ, at $\tau_{max} \approx 0.51$, is labelled in red dots.

.

In Figure 4.13 we again compare the results of our slope γ thresholding and COGENT thresholds chosen. We can see that for this dataset, our slope γ threshold identified the least resilient $\tau = \tau_{max}$, showing a close to threshold resulting the least network consistency, as suggested by COGENT. Indeed we have a few outliers of our γ for τ less than this τ_{max}, which might need to be treated carefully and consider further study. In general, from the trend of γ, we agree that the graph produced by the patientID 3085 playing game Catch at first instance, becomes more robust, as the τ values increase, for $\tau > \tau_{max}$.

As we introduced, we aim to use a large dataset to produce random graphs and networks to learn the resilience of which to changes in the threshold parameter, and we compare our γ approach with the existing studies on threshold choosing, such as COGENT (Bozhilova et al., 2020). In the next chapter, we will undertake this cut-off τ learning on the learnt SRGGs, which contains more than 100 nodes.

Chapter 5

Applications of Graph Learning: Graphical models for protein design

A residue is one unit of a complex molecule such as a protein. Within a protein, certain residues - commonly referred to as "critical residues" - are essential for protein function. Identifying those residues of a protein that most potently influence protein function is therefore of significant importance. This can be done by analysing the results of molecular dynamical simulations of the evolution of a considered protein structure. In this chapter, I discuss a robust and fast identification of the "critical residues" of a particular protein called *skp1*, where in this work, criticality implies the propensity to undergo interactions with other residues. In fact, the most critical residues are identified within our learning of the realisations of a random graph variable, using data from molecular dynamical simulation of the evolution of the states attained by *skp1*, over a given number of time steps. To recall from above, here the learnt graph is a Random Geometric Graph drawn in a probabilistic metric space.

5.1 Existing studies on identifying critical residues of protein

Proteins are fundamental to biological systems, playing key roles in molecular recognition, structural stability, and allosteric regulation. These functions are largely determined by the protein's three-dimensional (3D) structure, which relies on intricate residue-residue interactions. Identifying these residues is crucial in fields like computational biology, as it helps to understand disease-related mutations, protein engineering, and drug design.

Several computational strategies have been developed to identify these residues. One widely used method is Sequence Conservation Analysis, which assumes that highly conserved residues are crucial for function. Shannon Entropy (Cover and Thomas, 1991; Sander and Schneider, 1991; Shenkin et al., 1991), which is a measure of sequence variability, where conserved residues exhibit low entropy, indi-

cating high functional importance. An extension, Jensen - Shannon Divergence (JSD), quantifies conservation more effectively by incorporating weighted evolutionary distances (Capra and Singh, 2007). Meanwhile, Statistical Coupling Analysis (SCA), developed by Lockless and Ranganathan (1999), extends conservation analysis by detecting statistical dependencies between residues, revealing networks of functionally linked residues. More recently, evolutionary coupling (ECs) methods have further enhanced this approach. For example, Hopf et al. (2017) employed evolutionary couplings derived from large sequence alignments to model sequence co-variation - their EVmutation model is an unsupervised probabilistic method that predicts how mutations alter protein stability based on evolutionary data, effectively capturing residue-residue dependencies and their effects on protein fitness. Another key strategy focuses on protein stability and energetics - these methods assume that functionally important residues contribute to maintaining the protein's native state and facilitating structural transitions. Elcock (2001) employed continuum electrostatics methods to compute the energetic contributions of individual residues within a protein structure to assess how each residue influenced the overall stability, across six well-characterised structures proteins - in his calculations, his method identifies those most destabilising residues, which are known as the functionally important residues based on previous studies. Su et al. (2011) developed an Elastic Network Model (ENM), which treats proteins as spring-like networks and identifies key residues by analysing how perturbations affect free energy differences during conformational transitions.

However, there are some limitations of these approaches in the current studies. Sequence conservation-based methods do not consider protein structure and dynamics - some highly dynamic residues, which may lack functional significance, while residues critical for structural stability or allosteric regulation may not necessarily exhibit strong conservation. Thermodynamic models, which offer high-resolution functional insights, on the other hand, are computationally demanding and depend on highly accurate protein structures. Moreover, they do not directly capture network-wide residue interactions beyond local energetic contributions.

To overcome the limitations of traditional approaches, network-based modelling has emerged as a powerful alternative for analysing residue importance in proteins. These models treat proteins as Residue Interaction Networks (RINs), where amino acid residues are represented as nodes and their interactions as edges, based on structural distance, hydrogen bonding, or functional coupling (Doncheva et al., 2012; Haspel and Jagodzinski, 2017; Kantelis et al., 2022). Unlike sequence-based methods, which assume local conservation provides, RINs provide a global perspective on residue influence, capturing long-range dependencies and interaction pathways within proteins. To determine which residues are functionally important, various graph centrality measures are applied to protein networks. del Sol and O'Meara (2005) used small-world network analysis to show that residues with high connectivity tend to be functionally important; Thibert et al. (2005) evaluated several centrality measures such as Degree Centrality (measuring connectivity), Closeness Centrality (measuring how close a residue is to others), and Betweenness Centrality (identifying residues that act as key bridges in the network), and combined with

phylogenetic analysis to improve predictions and showed that network properties predict residue importance more effectively than sequence-based conservation metrics alone. More recently, Cusack et al. (2007) introduced dynamic connectivity (dk), a measure that evaluates how residues contribute to long-range information flow within the protein and it considers how residues mediate communication across the entire protein - a functionally important residues, such as active-site residues or allosteric regulators, tend to have high dk values because they act as key mediators in protein dynamics. The existing network-based methods also exhibit limitations on rigid network construction - many RINs use fixed cutoff distances, which do not fully capture flexibility in protein structures. Other methods provide useful snapshots but do not incorporate time-dependent conformational changes, resulting in the lack of dynamic adaptability.

Giles et al. (2016) proposed a Soft Random Geometric Graph (SRGG) model, which incorporates probabilistic edge weights instead of binary interactions. Building on this idea, in our work we used a new Random Geometric Graph (RGG) defined over a probabilistic metric space to parameterise the criticality of residues within a given protein, using data obtained from molecular dynamics (MD) simulations. We demonstrate this approach by identifying critical residues in simulation data for the protein *skp1*. We compute the inter-residue correlation matrix of this protein, and this correlation matrix is then used to generate a realisation of a random graph variable, where each node corresponds to the random variable that represents the state a residue attains. In the realisation of the random graph variable, edges exist with a probability, between a pair of nodes (residues, in this case), given the inter-residue correlation matrix computed from the MD simulations that record the states attained by each residue over a certain time interval. A higher edge probability indicates a stronger interaction between the associated residue pair.

5.2 Methodology

5.2.1 Data

Inside our cells, old or damaged proteins need to be removed so they don't interfere with normal function. This clean-up process is handled by a system that tags these unwanted proteins for recycling, and this process is essential for maintaining cellular health, regulating signals, and ensuring proper cell division. In this system, *E3* ligases (Berndsen and Wolberger, 2014) act like inspectors and they identify which proteins need to be removed and attach a special "tag" (called ubiquitin) to mark them for disposal. And *skp1*, is a connector within this tagging team. It links the core machinery of the E3 ligase to specific proteins that need to be tagged. Because *skp1* can work with many different partners, it plays a key role in deciding which proteins are marked for recycling. When this process goes wrong, it can lead to diseases like cancer (Liu et al., 2015) or neurodegeneration (Fishman-Jacob and Youdim, 2024).

There are wide structural studies of *skp1* in the literature. These studies re-

vealed that *skp1* can change its shape depending on whether it is working alone or connected to a partner protein (i.e. an F-box protein) (Chandra Dantu et al., 2016; Bhattacharya et al., 2022). In particular, helix 8 of *skp1* (residues 147–158), is a key part of *skp1*. When *skp1* is bound to a partner, this section forms a tight, spiral-like shape (i.e. a helix), which is stable and well-organised. But when *skp1* is on its own, helix 8 becomes more flexible and unstructured (i.e. a coil). In our study, we used molecular dynamics (MD) simulations to track this helix–coil transition and identify the critical residues driving it - the initial coordinates for our MD simulations were derived from the structure of Schulman et al. (2000), as reinterpreted by Chandra Dantu et al. (2016). The simulations recorded data on states attained by all residues of the protein *skp1*, over a total time period of about 1000 nanoseconds (ns). Outputs are recorded at the end of each time window of width=100 picoseconds, (or ps, where 1 ps $= 10^{-3}$ ns), i.e. outputs are recorded in the simulation, at $N_T \approx 1000/0.1 = 10,000$ number of time steps. In fact, $N_T = 10,006$. Thus, we have tracked the evolution of the protein over $N_T = 10,006$ number of time steps.

Secondary structure assignment was carried out using DSSP v3.0 (Define Secondary Structure of Proteins version 3.0), a program used to assign secondary structure to amino acid residues in protein structures, based on their 3D atomic coordinates (Joosten et al., 2010; Kabsch and Sander, 1983). In our data, DSSP assigned the states of each residue attained, into one of the following: ∼: Coil, E: Beta-Sheet; B: Beta-Bend; S: Bend; T: Turn; H: Alpha-helix; I: 5-Helix; G: 3-Helix. In Figure 5.2, each state that is attained by a residue of the protein in the simulation, is represented in 1 of 8 different colours. Thus, the state of a residue is a categorical variable in our work. We denote residues with the index variable K that then takes the value $k \in \{1, ..., 158\}$. Since the first and last residues in our MD simulation data, $\mathbf{D}$, do not participate in the evolution that is tracked within the simulations, we then ignored them from the dataset, resulting in 156 residues. Therefore in dataset $\mathbf{D}$, the state attained by the k-th residue at the t-th time point as $x_k^{(t)}$, where $k \in \{1, \ldots, 156\}$ and $t \in \{1, 2, \ldots, N_T\}$. States that are attained by all 156 residues at any time point, form a distinct row of the simulated dataset $\mathbf{D}$, while states attained by each residue form a column of this dataset. Therefore we have our dataset $\mathbf{D}$, with 10006 rows, and 156 columns.

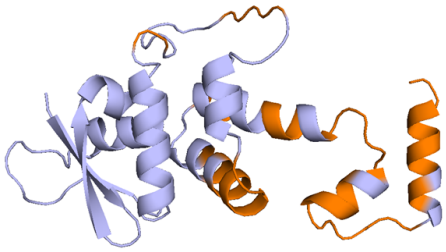


Figure 5.1: Residues of *skp1* interacting with different F-box partners are highlighted across the protein.

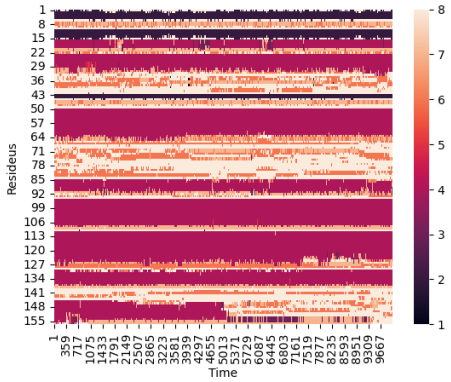


Figure 5.2: A heat map of the simulated data $\mathbf{D}$ that depicts the state attained by the k-th residue at the t-th time point, for $t = 1, \ldots, N_T = 10006$ and $k = 1, \ldots, 156$. The categorical state variable that takes values at eight levels, is denoted one of eight shades, each of which represents a geometric configuration; these configurations are denoted 'B': 1, 'E': 2, 'G': 3, 'H': 4, 'I': 5, 'S': 6, 'T': 7, '∼': 8

5.2.2 A random graph variable

To identify the most critical residues in the simulations of *skp1*, we first learn the graph of dataset $\mathbf{D}$, where this graph is learnt using values (or realisations) of a random graph variable. We learn values of this graph variable given the data $\mathbf{D}$ - this random graph variable is an RGG (Giles et al., 2016; Penrose, 2016; Chakrabarty, 2023; Chakrabarty et al., 2023) that is drawn in a probabilistic metric space (Menger, 1942; Schweizer and Sklar, 2011). As stated in Chapter 2, just as in Euclidean space we can assign a non-negative distance to any pair of points, in a probability metric space we can assign a probability distribution - defined over positive support - to any two points in this space. Thus, the distance between any two nodes of the graph variable we consider, is a probability distribution that is defined over a non-negative support.

We define X_k to be the random variable that represents the state of the k-th residue, and $k = 1, \ldots, p$, for $p = 156$. Then the variable X_k is attached to the k-th node of the graph variable. Recall the definition in Chapter 2, the posterior of the edge $G_{kk'}$ that is either 1 or 0, to suggest that k-th and k'-th nodes are connected or not, given by the absolute correlation $|\sigma_{kk'}|$ between X_k and $X_{k'}$ that are attached to the k-th and k'-th node.

Remark 5.2.1. *Recall Definition 2.2.7, we write the edge probability of $G_{kk'}$ conditional the absolute correlation $R_{kk'} = [|\sigma_{kk'}|]$ between the nodes that this edge joins - as $m(G_{kk'}|R_{kk'})$. We identify a closed-form expression for this condition probability $(m(G_{kk'}|R_{kk'}))$ as following:*

$$m(G_{kk'}|R_{kk'}) = K\left[\sqrt{\frac{2}{\pi}}\exp\left(\frac{-(S_{kk'})^2}{2}\right) - (S_{kk'})\,\mathrm{erfc}\left(\frac{|S_{kk'}|}{\sqrt{2}}\right)\right], \tag{5.2.1}$$

where $S_{kk'} = |G_{kk'} - |R_{kk'}||$ *as defined previously and the complementary error function* $\text{erfc}(\cdot) = 1 - \text{erf}(\cdot)$.

Therefore the RGG that we construct in the probabilistic metric space, has $p = 156$ nodes in the vertex set $\boldsymbol{V} = \{1, 2, \ldots, p\}$, with the random variable X_k attached to the k-th node, $\forall k \in \boldsymbol{V}$, and the edge variable between the k-th and k'-th nodes denoted $G_{kk'}$ s.t. $G_{kk'} = g_{kk'} \in \{0, 1\}$. So we state that in our graph variable, $G_{kk'} = 1$, (i.e. the edge exists between the k-th and k'-th nodes), if the posterior probability $m(G_{kk'} = g_{kk'}|R_{kk'} = |\sigma_{kk'}|)$ exceeds a cutoff probability τ. The edge variable $G_{kk'}$ of this RGG is then learnt, conditional on the absolute correlation $R_{kk'}$.

Thus, the probability of the whole graph variable, given the inter-column correlation structure of data $\mathbf{D}$ is then fully known, since the graph is built out of edge variables that connect each nodal pair in this graph, with the edges formed independently of each other, and in the absence of self-loops in this graph (by definition). This closed-form identification of the posterior probability of the graph variable, given the correlation structure of the data at hand, is useful since it permits the computation of a statistical distance (or divergence) between the posterior probability of a graph variable given one dataset, and that of the graph variable given a different dataset. Recall from Definition 2.2.10, the posterior of the RGG variable $\mathcal{G}_{\boldsymbol{V},m}(\tau, \boldsymbol{\Sigma})$ that is learnt using the edge posterior $m(\cdot|\cdot)$, given the known inter-column correlation matrix $\boldsymbol{\Sigma} = [\sigma_{kk'}]$ of data $\mathbf{D}$, is

$$\pi(\mathcal{G}_{\boldsymbol{V},m}(\tau, \boldsymbol{\Sigma})|\{\sigma_{kk'}\}_{kk' \in \boldsymbol{V}}) = \prod_{kk' \in \boldsymbol{V}; k<k'} m(G_{kk'} = g_{kk'}|\sigma_{kk'},)$$

where the RGG is constructed without self-loops, and the edges are included independently of each other. Here $m(\cdot|\cdot)$ is defined in Definition 2.2.7.

5.2.3 Inter-residue correlation matrix

To learn the RGG variable, the absolute correlation $\sigma_{kk'}$ between the state of the k-th and the k'-th residues is sought, so that the inter-column correlation matrix $\boldsymbol{\Sigma} = [\sigma_{kk'}]$ of simulated dataset $\mathbf{D}$ can be estimated. Here $\sigma_{kk'}$ is the value of the absolute correlation $|corr(X_k, X_{k'})| \equiv R_{kk'}$ between the state of the k-th and the k'-th residues. Now, in the dataset $\mathbf{D}$, we have X_k and $X_{k'}$ taking values in eight categories, i.e. X_k is a categorical variable, $\forall k \in \{1, \ldots, 156\}$. Then to learn the value $\sigma_{kk'}$ of the absolute correlation $R_{k,k'}$ between X_k and $X_{k'}$, we invoke the Cramér's V (Cramér, 1946), which is a measure of correlation between categorical variables.

To define Cramér's V between X_k and $X_{k'}$, we need to recall the chi-squared parameter, $\chi^2_{kk'}$ computed using the frequency of occurrence of X_k and $X_{k'}$.

Definition 5.2.1. *To define the correlation between the categorical variables* X_k *and* $X_{k'}$ *- that take values at 8-levels - we first construct an* 8×2*-dimensional contingency table that we denote* $\mathcal{W}_{kk'}$*. The* ℓ, q*-th cell of this table contains the frequency* $\omega_{\ell,q}$ *with which* X_q *attains value at the* ℓ*-th level in data* $\mathbf{D}$*,* $\forall q \in \{kk'\}$*;* $\ell \in \{1, \ldots, N_\ell\}$.

Then $\chi^2_{kk'}$ is defined as:

$$\chi^2_{kk'} := \sum_{q=kk'} \left[\sum_{\ell=1}^{N_\ell} \frac{(\omega_{\ell,q} - \mu_{\ell,q})^2}{\mu_{\ell.q}} \right],$$

where $\mu_{\ell.q}$ is the estimated expected frequency with which X_q attains value in the ℓ-th level in $\mathbf{D}$, i.e.

$$\mu_{\ell.q} = \frac{\textit{sum of the frequencies in the } \ell\textit{-th row} \times \textit{sum of the frequencies in the } q\textit{-th column}}{\textit{total frequency } N^{(T)}_{kk'} \textit{ for } X_k \textit{ and } X_{k'} \textit{ to attain values in all the 8 levels}},$$

given by the 8×2-dimensional contingency table $\mathcal{W}_{kk'}$. Here $k < k'; k' \in \boldsymbol{V}$.

Definition 5.2.2. *With the computed $\chi^2_{kk'}$, we construct the inter-residue correlation matrix $\boldsymbol{\Sigma} = [\sigma_{kk'}]$ as:*

$$\sigma_{kk'} = \sqrt{\frac{\chi^2_{kk'}/2N^{(T)}_{kk'}}{min(8-1, 2-1)}} = \sqrt{\chi^2_{kk'}/2N^{(T)}_{kk'}}, \tag{5.2.2}$$

since there are $N^{(T)}_{kk'}$ number of realisations of X_k and $X_{k'}$ that we consider in the computation of this correlation and the normalisation is done by (one less than) the minima of the dimensions of the data considered here - there are eight states possibly attained by the residues and there are two residue pairs we are considering here.

Thus, we can now compute $\boldsymbol{\Sigma}$ for the dataset $\mathbf{D}$ - the 156×156 inter-residue correlation matrix to enable construction of the corresponding graphical model for this $\mathbf{D}$.

5.2.4 Sampling from the edge marginal

With the marginal posterior probability $m(G_{kk'}|R_{kk'})$ defined in Definition 2.2.7, we undertake rejection sampling of $G_{kk'}$ to generate N_{tot} values of this edge variable, i.e. these sampled values of $G_{kk'}$ are used to construct a sample of size N_{tot}. Such undertaking of rejection sampling is discussed in Definition 2.2.11, Chapter 2. If the relative frequency for $G_{kk'}$ to attain the value of 1 in this sample exceeds the chosen cut-off probability τ, we retain the edge between the k-th and the k'-th nodes in our final graphical model of data $\mathbf{D}$, as we have introduced in the previous chapter. The Algorithm 4 below shows the implementation of RGG.

Algorithm 4: Algorithm for implementation of RGG

`/* Learning RGG of data` $\mathbf{D}$. `*/`
States attained by k-th residue at $t = 1, ..., N_T = 10006$, $\forall k = 1, \ldots, 156$, populate the $N_T \times 156$-dimensional dataset $\mathbf{D}$
Estimate 156×156-dimensional inter-column correlation matrix $\mathbf{\Sigma} = [\sigma_{ij}]$, using Cramér's V
for $i \leftarrow 1$ **to** 156 **increment by** 1, **do**
  for $j \leftarrow 1$ **to** 156 **increment by** 1, **do**
    Given i, j-th element of $\mathbf{\Sigma}$, sample g_{ij} from edge marginal $m(G_{ij} = g_{ij}|\sigma_{ij})$ given in Definition 2.2.6
    Compute posterior probability $\pi(\mathcal{G}_{\boldsymbol{V},m}(\tau, \mathbf{\Sigma})|\{\sigma_{ij}\}_{i<j;j\in\boldsymbol{V}})$ of random graph variable $\mathcal{G}_{\boldsymbol{V},m}(\tau, \mathbf{\Sigma})$
    for $r \leftarrow 1$ **to** N **increment by** 1, **do**
      Sample from posterior probability $\mathcal{G}_{\boldsymbol{V},m}(\tau, \mathbf{\Sigma})$ using Rejection Sampling, to generate sample $\{g_{ij}^{(1)}, g_{ij}^{(2)}, \ldots, g_{ij}^{(N)}\}$ for edge variable G_{ij}
    Compute relative frequency $f_{ij} := \sum_{n=1}^{N} g_{ij}^{(n)}/N$.
    $G_{ij} = 1 \iff f_{ij} > \tau$.

5.2.5 Parameterisation of criticality

We propose three distinct methods for the automated identification of the Q-most critical residues in the protein *skp1*, using its simulated trajectory data $\mathbf{D}$.

Before I discuss the development of the parametrisation of "criticality" of residues of the considered protein, I mention my definition of criticality. I define a residue to be more "critical" than another, in two ways.

1. Firstly, a residue A in a protein is considered more critical than another residue called B, if the influence of A on the random graph, nodes of which host states of the residues of this protein, is higher than that of B, where I define this influence in a well-defined way below. Thus, if we learn the graph of all the residues except residue A, the probability of that random graph is more different from the graph of the full protein, than is the graph of the protein that comprises all residues except B. So in this treatment, the criticality of a residue is viewed as a probabilistic influence of this residue, on all other residues of the protein.

2. Secondly, such a residue-on-residue influence is viewed as the number of edges that exist in the graph of the protein. Thus, if residue A interacts with fewer residues of the protein, than residue B, then A is considered more critical than B. Here, one question that arises is the time scale over which such interactions are tracked. This time-scale could be the time interval over which the evolution of the protein is monitored, or it could be a shorter time interval than the time taken by the protein to evolve. Thus, in the latter consideration, the dynamic criticality of residues is identified.

Parametrisation of functional instability of a residue: 'Hold-One-Out'

Our first identification of the critical residues suggests omitting each residue of the protein from our consideration, one at a time. This is equivalent to stating that we learn the posterior probability of the random graph variable given the data $\mathbf{D}_{-k}$ which is constructed by omitting the k-th column of the dataset $\mathbf{D}$, for $k = 1, \ldots, p, p = 156$. Thus, $\mathbf{D}_{-k}$ is the dataset that bears $p-1$ columns and N_T (=10,006) rows after omitting the k-th column from data $\mathbf{D}$. So we will need to estimate anew, the correlation matrix $\mathbf{\Sigma}_{-k}$ of the data $\mathbf{D}_{-k}$. Then we will learn the graph variable $\mathcal{G}_{\boldsymbol{V}_{-k},m}(\tau, \mathbf{\Sigma}_{-k})$, defined on the vertex set $\mathbf{V}_{-k} = \{X_1, \ldots, X_{k-1}, X_{k+1}, \ldots, X_p\}$ - we will learn this graph given data $\mathbf{D}_{-k}$. On the other hand, we would already have learnt the graph variable $\mathcal{G}_{\boldsymbol{V},m}(\tau, \mathbf{\Sigma})$, defined on the vertex set $\mathbf{V} = \{X_1, \ldots, X_{k-1}, X_k, X_{k+1}, \ldots, X_p\}$, given data $\mathbf{D}$. We will compute the distance between the posterior probability of $\mathcal{G}_{\boldsymbol{V}_{-k},m}(\tau, \mathbf{\Sigma}_{-k})$ given data $\mathbf{D}_{-k}$ and that of $\mathcal{G}_{\boldsymbol{V},m}(\tau, \mathbf{\Sigma})$ given data $\mathbf{D}$. We refer to this distance as δ_k. We will compute δ_k for $k = 1, \ldots, p$.

Definition 5.2.3. *We learn the RGG variable $\mathcal{G}_{\boldsymbol{V}_{-k},m}(\tau, \mathbf{\Sigma}_{-k})$ defined on the vertex set $\boldsymbol{V}_{-k} := \{1, \ldots, k-1, k+1, \ldots, p\}$, given inter-column correlations $\mathbf{\Sigma}_{-k}$ of data $\mathbf{D}_{-k}$, $\forall k \in \{1, \ldots, 156\}$, and compute δ_k as:*

$$\delta_k := \pi(\mathcal{G}_{\boldsymbol{V}_{-k},m}(\tau, \mathbf{\Sigma}_{-k}) | \{\sigma_{-k}\}_{i,j \in \boldsymbol{V}}) - \pi(\mathcal{G}_{\boldsymbol{V},m}(\tau, \mathbf{\Sigma}) | \{\sigma_{ij}\}_{i,j \in \boldsymbol{V}}),$$

$\forall k = 1, \ldots, p = 156$. *Here, $\mathbf{\Sigma}_{-k}$ is the inter-residue correlation matrix of dataset $\mathbf{D}_{-k}$, s.t. the i,j-th element of $\mathbf{\Sigma}_{-k}$ is $|corr(X_i, X_j)|$, where $i, j \in \boldsymbol{V}_{-k}$.*

Hence for $\delta_{k'} \leq 0$ implies that the "reduced" graph $\mathcal{G}_{-k}(\tau, \mathbf{\Sigma}_{-k})$ is less compatible with the correlation structure of data $\mathbf{D}_{-k}$ than the "full" graph $\mathcal{G}(\tau, \mathbf{\Sigma})$ is with the correlation of data $\mathbf{D}$. Indeed, if for $\delta_{k'} < 0$ and $\delta_{-k} < 0$, $\delta_{k'} < \delta_k$, for any $k', k = 1, \ldots, p$; $k' \neq k$, it implies that omission of values of $X_{k'}$ from the considered data renders the graph learning worse (than the benchmark case, i.e. the full graph) than when X_k values are omitted from the data. Thus, without such $X_{k'}$, the correlation structure of data $\mathbf{D}$ is more incorrectly learnt than without such an X_k. This could happen if the k'-th residue is a (comparatively) more stable part of the dataset $\mathbf{D}$ than the k-th residue, in the sense that, over time, the k'-th residue retains its patterns of interplay with the other residues of the protein.

On the other hand, $\delta_k > 0$ implies that the k-th residue is volatile - the k-th residue of the protein is ephemeral in the way it interacts with the other residues - it was correlating with another residue at one time point (during the evolution of the protein) and again breaking this connection soon thereafter. Then dropping this k-th residue from consideration of inter-residue correlation will improve the compatibility of the resulting correlation representation, i.e. the reduced graph, with the respective dataset, compared to the benchmark situation. Thus, the more volatile the k-th residue is in the protein, in the way it interacts with the other residues of the protein, the less negative is δ_k.

We define this volatility of connections of a given residue with other residues of *skp1* (as present as δ_k), to be an indicator of criticality of this protein - we interpret

$\delta_k > 0$ to suggest that the k-th residue is critical. So we sort the values $\delta_1, \ldots, \delta_p$, and the Q highest values of this inter-graph distance then informs on the Q most critical residues of this protein.

Remark 5.2.2. *If the k-th residue provides a high δ_k, then it implies high the instability of this k-th residue, which leads to the RGG variable $\mathcal{G}_{\boldsymbol{V}_{-k},m}(\tau, \boldsymbol{\Sigma}_{-k})$ with the correlation $\boldsymbol{\Sigma}_{-k}$ to be more compatible, than the RGG $\mathcal{G}_{\boldsymbol{V},m}(\tau, \boldsymbol{\Sigma})$ is with $\boldsymbol{\Sigma}$.*

Thus, δ_k parametrises the functional instability of the k-th residue.

The Algorithm for $\delta_\cdot$ parameters computation is shown below in Algorithm 5.

Algorithm 5: Algorithm for the computation of δ_k

```
/* Start generating RGG                                              */
```
States of k-th residues attained over $N_T = 1, ..., 10006$,
gives rise to $N_T \times 156$-dimensional time series (on state location) dataset
Estimate 156×156-dimensional inter-column correlation matrix $\boldsymbol{\Sigma}$,
as per Equation 5.2.2.
for $i \leftarrow 1$ **to** 156 **increment by** 1, **do**
  for $j \leftarrow 1$ **to** 156 **increment by** 1, **do**
    Given i, j-th element of $\boldsymbol{\Sigma}$, sample g_{ij} from edge marginal
    $m(G_{ij} = g_{ij}|\sigma_{ij})$ given in Definition 2.2
Compute posterior probability $\pi(\mathcal{G}_{\boldsymbol{V},m}(\tau, \boldsymbol{\Sigma})|\{\sigma_{ij}\}_{i,j\in\boldsymbol{V}})$ of the random graph variable $\mathcal{G}_{\boldsymbol{V},m}(\tau, \boldsymbol{\Sigma})$
```
/* Start calculating δ_k parameter                                   */
```
for $k \leftarrow 1$ **to** 156 **increment by** 1, **do**
  Remove k-th residue from the dataset, gives rise to
  $N_T \times 155$-dimensional dataset $\mathbf{D}_{-k}$
  Estimate 155×155-dimensional inter-column correlation matrix $\boldsymbol{\Sigma}_{-k}$
  Compute posterior probability $\pi(\mathcal{G}_{\boldsymbol{V}_{-k},m}(\tau, \boldsymbol{\Sigma}_{-k})$ of random graph variable $\mathcal{G}_{\boldsymbol{V}_{-k},m}(\tau, \boldsymbol{\Sigma}_{-k})$
```
  /* We calculate δ_k use the difference between posterior
     probability of D_{-k} and posterior probability of D           */
```
  $\delta_k := \pi(\mathcal{G}_{\boldsymbol{V}_{-k},m}(\tau, \boldsymbol{\Sigma}_{-k})) - \pi(\mathcal{G}_{\boldsymbol{V},m}(\tau, \boldsymbol{\Sigma}))$, $\forall k = 1, \ldots, p = 156$.

Unlike the other parameters we introduce later, one advantage of using this criticality identification parameter $\delta_\cdot$ is that it is free from the choice of the cut-off probability τ. Though this method is a robust way of identifying the Q most critical residues, the computation of this instability parameter δ_k is time and resource-intensive - we need to learn $p + 1$ realisations of RGGs, using the datasets $\mathbf{D}$ and $\mathbf{D}_{-1}, \ldots, \mathbf{D}_{-p}$, respectively - learning $p = 156$ reduced graphs $\mathcal{G}_{\boldsymbol{V}_{-k},m}(\tau, \boldsymbol{\Sigma}_{-k})$ is time consuming - to accomplish the identification of the Q most critical residues.

Parametrisation of temporal variation of degree distribution

In the next parameter that we introduce to identify the critical residues, we compute the standard deviation of the sample of temporally-local values of the degree of each node of the RGG that is learnt with dataset $\mathbf{D}$, where the k-th node is attached to the variable that represents the state of the k-th residue - for each of the k nodes in

the graph, we record its degree at every time point and then calculate the sample standard deviation of that temporal sequence. Our underlying rationale is that residues which play a critical, regulatory role tend to form and break contacts more frequently than non-critical residues; in other words, their connectivity fluctuates more during protein evolution. We look at this unevenness of the temporal variation parameter. Thus, the standard deviation η_k of the temporal distribution of the degree of the k-th residue, will provide a higher value than the value of $\eta_{k'}$, if the k'-th residue is less critical, $\forall k, k' \in \boldsymbol{V}$. The bigger changes in the temporal distribution of the degree of a node, the comparatively higher criticality of the residue that corresponds with the other nodes - the higher η_k, the higher criticality of the k-th residue of the protein *skp1*.

Definition 5.2.4. *We partition the full dataset* $\mathbf{D}$ *into* N_b *blocks, each comprises* N_η *time steps, s.t each of such block is* $N_\eta \times 156$*-dimensional, for* $b = 1, ..., N_T/N_\eta$*, elements of which are values of the state variable of each of the* k *residues of the protein, over* N_η *time points. Then we learn a random graph variable* $\mathcal{G}_{\boldsymbol{V}_b,m}(\tau, \boldsymbol{\Sigma}_b)$*, given the inter-residual correlation matrix* $\boldsymbol{\Sigma}_b = [\rho_{ij}]$*,* $\forall i, j \in \boldsymbol{V}$ *of the partition data* $\mathbf{D_b}$ *- we then this graph variable at a pre-chosen cut-off probability* τ *s.t.* $g_{ij} = 1$ *if the edge posterior* $m(G_{ij} = 1|\sigma_{ij}) > \tau$*.*

We denote $\eta_k^{(b)}$ *to be number of edges connected to the k-th node in the graph* $\mathcal{G}_{m,\boldsymbol{V}}(\tau, \boldsymbol{\Sigma}_b)$*, at a chosen* τ*. We then have*

$$\eta_k^{(b)} := |\{(k, i) : g_{ki} = 1, i \neq k, i \in \boldsymbol{V}\}|, \forall k \in \boldsymbol{V}.$$

The unbiased estimate of standard deviation η_k *of sample* $\{\eta_k^{(1)}, \eta_k^{(2)}, \ldots, \eta_k^{(N_b)}\}$ *is then*

$$\eta_k = \sum_{i=1; i\neq k}^{N_b} \sqrt{(\eta_k^{(i)} - \bar{\eta}_k)^2/(N_b - 1)}, \text{ where } \bar{\eta}_k = \sum_{i=1; i\neq k}^{N_b} \eta_k^{(i)}/N_b.$$

Then for $\eta_1, \ldots, \eta_p$*,* $p = 1, ..., 156$*, we sort the Q-highest values of these* $\eta_\cdot$ *parameters, which correspond to the Q-most critical residues in the protein.*

The Algorithm 6 shows the computation of $\eta_\cdot$ parameters.

Algorithm 6: Algorithm for the computation of η_k, at a chosen $\tau = 0.854$

```
/* Start generating RGG                                              */
```
Divide the states of residues attained over $N_T = 1, ..., 10006$ into N_b partitions, each with $N_\eta = N_T/N_b$ rows and 156 columns

for $b \leftarrow 1$ **to** N_b **increment by** 1, **do**

  Estimate inter-column correlation matrix $\mathbf{\Sigma}_b$ of partition $\mathbf{D_b}$

  for $i \leftarrow 1$ **to** 156 **increment by** 1, **do**

    for $j \leftarrow 1$ **to** 156 **increment by** 1, **do**

      Given i, j-th element of $\mathbf{\Sigma}_b$, sample g_{ij} from edge marginal $m(G_{ij} = g_{ij}|\sigma_{ij})$ given in Definition 2.2.7

      if $m(G_{ij} = 1|\sigma_{ij}) > 0.854$, **then**

        $g_{ij} = 1$

```
    /* We then calculate the unbiased estimate of standard
       deviation η_k of the sample {η_k^(1), η_k^(2), ..., η_k^(N_b)}  */
```

    for $k \leftarrow 1$ **to** 156 **increment by** 1, **do**

      $\eta_k^{(b)} := |\{(k, i) : g_{ki} = 1, i \neq k, i \in \boldsymbol{V}\}|, \forall k \in \boldsymbol{V}$

      We have $\eta_k = \sum_{i=1;i\neq k}^{N_b} \sqrt{(\eta_k^{(i)} - \bar{\eta}_k)^2/(N_b - 1)}$, where $\bar{\eta}_k = \sum_{i=1;i\neq k}^{N_b} \eta_k^{(i)}/N_b$.

Parametrisation of the degree of individual nodes in the full graph

We wanted to absorb the information that is borne by the instability parameter $\delta.$ that we spoke of above, and design a parameter that would help identify criticality more easily than by computing $\delta.$ across the dataset $\mathbf{D}$. Effectively, both the value $\delta.$ and $\eta.$ parameterisation inform on how relevant the k-th residue is, in the inter-residue interplay within the protein, where such interplay amongst the residues gives rise to the evolution of the protein. Therefore we included a weaker marker of criticality, $\beta.$, i.e. the degree computed over the entire time interval $[0, N_T]$, namely, β_k denotes the degree of the k-th node that exist in RGG $\mathcal{G}_{\boldsymbol{V},m}(\tau, \mathbf{\Sigma})$, here $k \in \boldsymbol{V}$. It is interesting to look at the degree of nodes in the whole network - which has been proposed widely in the structural biology study in the literature (Hamer et al., 2010; Doncheva et al., 2012).

Definition 5.2.5. $\beta_k := |\{(k, i) : g_{ki} = 1, i \neq k, i \in \boldsymbol{V}\}|, \forall k \in \boldsymbol{V}$, *where* G_{ki} *is the edge variable straddled by the* k*-th and* i*-th nodes of the RGG* $\mathcal{G}_{\boldsymbol{V},m}(\tau, \mathbf{\Sigma})$.

Thus, the higher the number of edges emanating from the k-th node of our graph throughout the evolution of the protein, the more stable is the pattern of interplay of the other residues with the k-th residue. So we compute the number of edges emanating from a given node in the full graph $\mathcal{G}_{\boldsymbol{V},m}(\tau, \mathbf{\Sigma})$ that is learnt using information recorded over the full course of evolution of the protein, and refer to this parameter as the degree parameter β_k, such that the strength of connectivity of the k-th residue is the value β_k of β, for $k = 1, \ldots, p$.

Algorithm 7 presents the protocol used to compute the $\beta.$ parameters.

Algorithm 7: Algorithm for the computation of β_k, at a chosen $\tau = 0.854$

```
/* Begin by learning the RGG of data D.                              */
States of the k-th residue attained at N_T = 1, ..., 10006
 gives N_T × 156-dimensional time series (on state location) dataset D.
Estimate 156 × 156-dimensional inter-column correlation matrix Σ
for j ← 1 to 156 increment by 1, do
    for i ← 1 to j − 1 increment by 1, do
        Given i, j-th element of Σ, sample g_ij from edge marginal
        m(G_ij = g_ij|σ_ij) given in Definition 2.2.7
        if m(G_ij = 1|σ_ij) > 0.854, then
            g_ij = 1
/* Start calculating β_k parameter                                   */
for k ← 1 to 156 increment by 1, do
    β_c := |{(k, i) : g_ki = 1, i ≠ k, i ∈ V}|, ∀k ∈ V
```

Remark 5.2.3. *For a volatile residue k, the degree parameter β_k identified from graph $\mathcal{G}_{\boldsymbol{V},m}(\tau, \boldsymbol{\Sigma})$ is the lowest. During the course of unfolding of the protein, the rapidly-changing interaction between a volatile residue and the other residues, leads to the nodes associated with volatile residues being consistently identified as least connected over the whole time interval $[0, N_T]$.*

Detailed results of our implementation of these three parameters towards the identification of the Q-most critical residues of *skp1* using the trajectory data $\mathbf{D}$, is presented in Section 5.3.

5.2.6 Choosing the cut-off τ

As we mentioned in the previous chapters, recall the basic concepts of RGG in a probabilistic metric space such that its edge is a probability distribution over positive-definite support, the edge exists if this probability exceeds a cutoff probability τ. Again, the choice of the cut-off τ affects the degree of the k-th node in the RGG learnt with the data of any of the N_η partitions of data $\mathbf{D}$, and affects the degree of the k-th node in the full RGG that is learnt given the inter-column correlation matrix of dataset $\mathbf{D}$. To check the effect of τ on the $\eta_\cdot$ and $\beta_\cdot$ parameters, we consider our calculation of $\eta_\cdot$ and $\beta_\cdot$ parameters at a "meaningful" τ - we apply a τ that is not chosen arbitrarily, but instead produces the "most robust (or "sensitivity") RGG" typically.

Remark 5.2.4. *On the vertex set $\boldsymbol{V}$, using marginal posterior $m(\cdot|\cdot)$ defined in Definition 2.2.7; for the chosen cut-off τ, we have for the kk'-th edge, given $R_{kk'} = |\sigma_{kk'}|$, $g_{kk'} = 1 \iff m(G_{kk'} = 1|\sigma_{kk'}) > \tau$.*

Remark 5.2.5. *We undertake rejection sampling of $G_{kk'}$ to generate N_{tot} sampled values of edge variable. If the relative frequency for N_{tot} to attain the value of 1 in this sample exceeds the chosen cut-off probability τ, we retain the edge between the k-th and the k'-th nodes in our final graphical model of data $\mathbf{D}$.*

The relative frequency of edge variable $G_{kk'}$ in the sample $\{g_{kk'}^{(1)}, g_{kk'}^{(2)}, \dots, g_{kk'}^{(N_{tot})}\}$ is $\omega_{kk'} = \sum_{n=1}^{N_{tot}} g_{kk'}^{(n)} / N_{tot}$.

Hence in our graph, edge between the k-th and k'-th nodes exists, (i.e. $g_{kk'} = 1$) if and only if $\omega_{kk'} > \tau$, $\forall kk'$; $k < k'$; $k = 1, \dots, 156$.

We again compare our determination of thresholds with the existing technique, COGENT (Bozhilova et al., 2020; Pardo-Diaz et al., 2021, 2022), as we introduced in the previous chapter - COGENT identifies the most consistent network, while the changes in the chosen threshold. Our pursuit of the most robust RGG, which is defined on a given vertex set with given inter-residue correlation, is similar in spirit to the most consistent network - we identify the values of τ that yield the most 'robust', or rather give the most consistency of the graphs, in our learning of $\mathbf{D}$ - we computed the τ at which the slope of the (logarithm of the) posterior of our learnt graph variable which is conditional on $\mathbf{\Sigma}$ and treated as a function of τ, is (locally) minimised - such a local minima slope informs on the realisation of the graph variable that is minimally sensitive to change in τ, for a given $\mathbf{\Sigma}$ - we identify values of $\tau = \tau_{min}$ that yield the most 'robust/consistency' of the graphs, for given data $\mathbf{D}$. On the other hand, for the τ that the (logarithm of the) posterior of the RGG variable manifests a maxima, $\tau = \tau_{max}$, we denote this τ_{max} as 'most sensitive'.

Remark 5.2.6. *The RGG $\mathcal{G}_{\boldsymbol{V},m}(\tau, \mathbf{\Sigma})$ at a given $\boldsymbol{V}$ and $\mathbf{\Sigma}$, the posterior of which (as in Definition 2.2.10) changes the least with changes in τ, is referred to as the most robust RGG. The τ at which the most robust RGG is attained is denoted τ_{min}.*

On the other hand, the τ at which the posterior of the RGG variable changes the most, and the corresponding RGG realisation is then the one that is most sensitive to changes in τ. This τ is denoted τ_{max}.

We present results of learning graphs given our dataset $\mathbf{D}$, using varying values of τ, and compare these results to those produced by networks of the same dataset $\mathbf{D}$, produced at optimal thresholds identified by the method COGENT (Bozhilova et al., 2020). Recall the slope γ we defined in Definition 2.4.1, in Figure 5.3 we plot the slope γ log posterior at distinct τ values, vs $\tau \in (0, 1)$, and we present results of the choice of the value of τ given the MD simulation data $\mathbf{D}$ of *skp1*, and the COGENT-suggested threshold ω, (Bozhilova et al., 2020), generated by the same data $\mathbf{D}$. Additionally, in Figure 5.4, we display the most robust RGG that is learnt given data $\mathbf{D}$ at $\tau = \tau_{min}$, most sensitive (to changes in τ) RGG of data at $\tau = \tau_{max}$, along with the network that is learnt using this data in the method forwarded by Bozhilova et al. (2020), at the value of ω that produces the most consistent network.

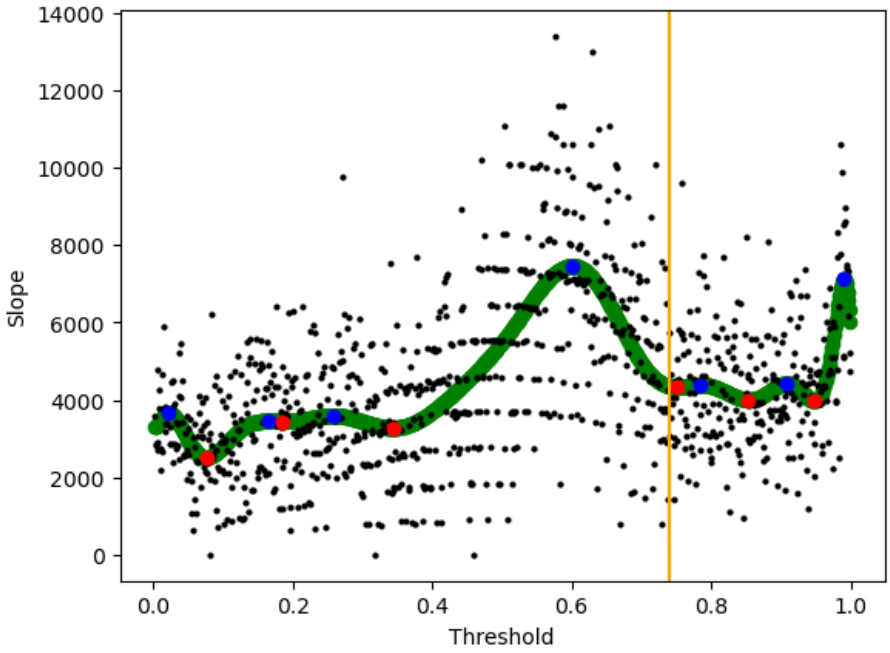


Figure 5.3: Figure displaying computed (approximate) values of the slope of the log posterior of the RGG variable with respect to τ - learnt using the simulated trajectory data $\mathbf{D}$ of the protein *skp1* - at chosen values of this τ variable. The values are shown in black filled circles. A fit is superposed in green, and the orange vertical line indicates the "best" threshold value of $\omega \approx 0.740$ that is suggested by the COGENT method (Bozhilova et al., 2020), implemented to learn the most consistent network in the data $\mathbf{D}$. The local maximums/minimums of the slope are labelled in blue/red dots respectively.

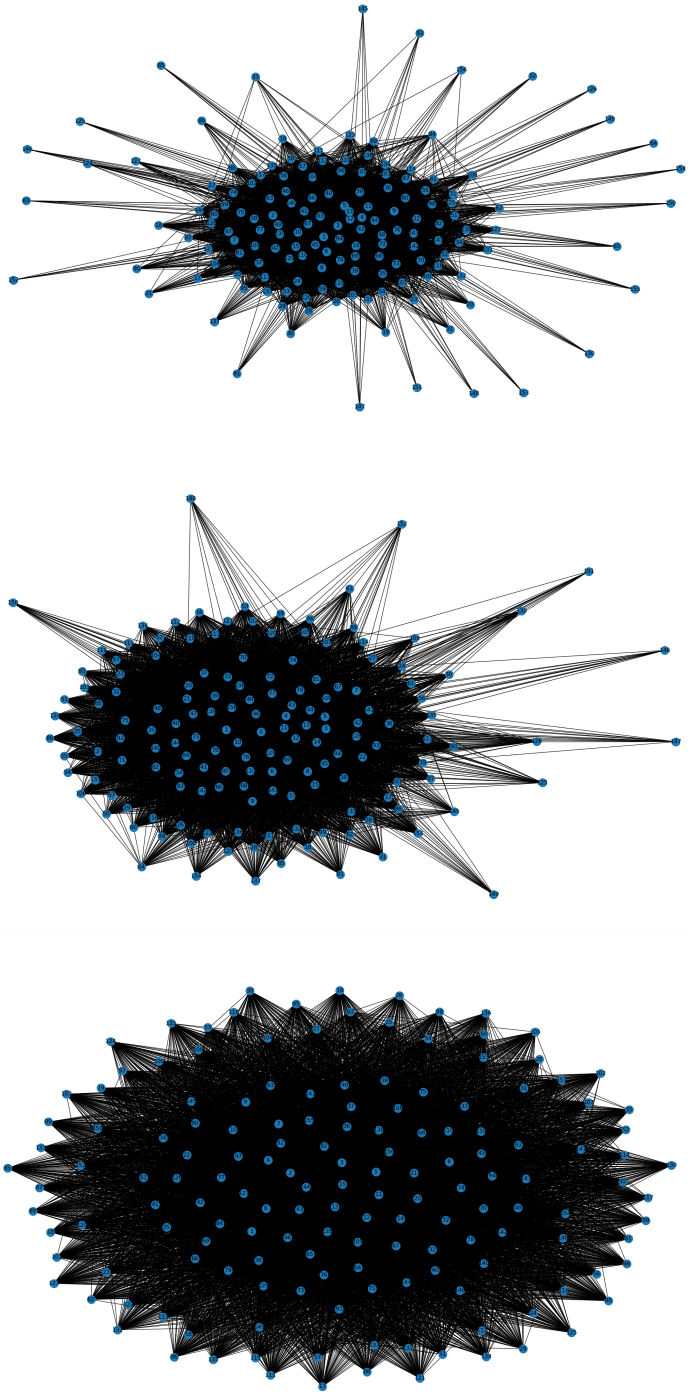


Figure 5.4: Figure displaying graphs produced by the thresholds suggest by Figure 5.3. From upper panel to bottom: network of data $\mathbf{D}$ constructed as suggested by Bozhilova et al. (2020), at the COGENT-suggested threshold of $\omega \approx 0.740$; most robust RGG of data $\mathbf{D}$ realised at $\tau = \tau_{min}$ ≈0.854; and most sensitive (to changes in τ) RGG of data $\mathbf{D}$, at $\tau = \tau_{max}$ of about 0.601.

In our search for the most critical residues using the $\eta_.$ parameters, we use the most robust RGG realisations that are attained at $\tau = \tau_{min}$. η_k computed using the most robust RGGs realised at $\tau = \tau_{min}$, is forwarded as a parametrisation of criticality of the k-th residue, $\forall k \in \boldsymbol{V}_{-k}$. Similarly, β_k is sought at $\tau = \tau_{min}$.

In Figure 5.5 we plot the β_k parameter for data $\mathbf{D}$, i.e. the degree of all, i.e. $p = 156$ nodes, for the realisation of the RGG variable, learnt at the COGENT-suggested best threshold ω, $\tau = \tau_{max}$ and $\tau = \tau_{min}$, respectively. We also present β_k computed for all nodes for data $\mathbf{D}$, in a Pearson Correlation-driven network, i.e. learnt as per Bozhilova et al. (2020), at the COGENT-suggested best threshold $\omega \approx 0.740$.

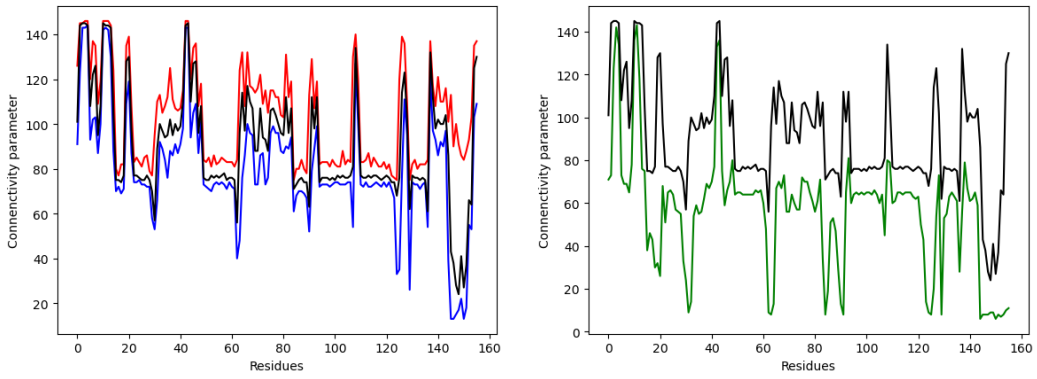


Figure 5.5: Left panel: Degree parameter β_k plotted against nodal index $k \in \{1, \ldots, 156\}$, computed using realisations of the RGG variable, at the thresholds suggested in Figure 5.4. β_k is plotted in black in a realisation of the RGG variable learnt given data $\mathbf{D}$ at the COGENT-suggested optimal threshold of $\omega \approx 0.740$). The β_k parameter against k for the RGG learnt at $\tau = \tau_{max}$ ($\approx$0.601) is in red line, while that for the RGG learnt at $\tau = \tau_{min}$ ($\approx$0.854) is in blue. Right panel: Here the green line indicates the β across the residue index values, for a Pearson Correlation-driven network that is constructed at the optimal threshold of $\omega \approx 0.740$, suggested by the method COGENT, within the COGENT package (Bozhilova et al., 2020). This is compared to the β profile - in black - obtained for the RGG learnt at $\omega \approx 0.740$, as in the left panel.

Thus Figure 5.5 indicates an overall consistency in the β parameters computed across values of τ, as we scan across $\tau \in [0, 1]$. However, there are local variations in computed values of β, noted when we move from τ values of about 0.854 to 0.740 to 0.601. An overall similarity in the computed degree distribution is also noted, irrespective of whether we learn the graph of the given data $\mathbf{D}$ as an RGG, or as a Pearson Correlation-driven network. Full details of the results are presented in the following section.

5.3 Results on identification of critical residues of *skp1*

We calculate the inter-column correlation matrix $\mathbf{\Sigma}$, of the dataset $\mathbf{D}$, and learn the realisation of the RGG variable. These are presented in Figure 5.6.

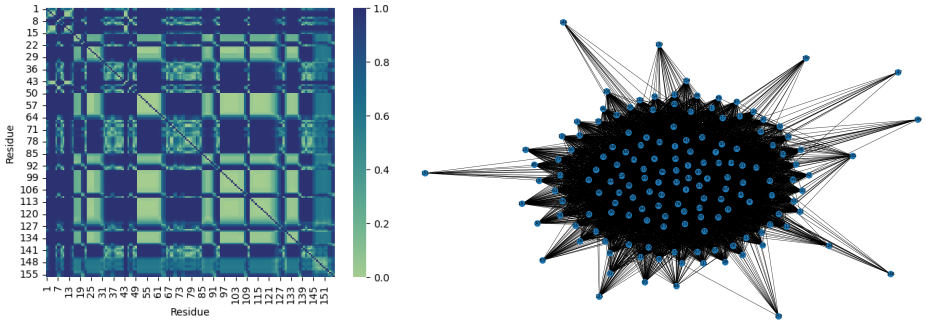


Figure 5.6: The heatmap generated by the correlation matrix $\mathbf{\Sigma}$ is shown on the left. On the right, we present a learnt realisation of the RGG variable, given this correlation matrix $\mathbf{\Sigma}$ of data $\mathbf{D}$, using a cut-off probability $\tau \approx 0.854$.

5.3.1 Criticality using δ

Our identification of the 10-most critical residues of the simulated protein *skp1* is presented in a tabulated form in Table 5.1. Here, the criterion used for this identification of criticality uses the computed $\delta_\cdot$ parameters by Definition 5.2.3. Thus, these 10-most critical residues are the 10-most functionally unstable residues, in the simulated data $\mathbf{D}$.

Residues index k	δ_k
150	0.004899353
147	0.004866306
151	0.004730643
152	0.004716151
149	0.004547655
146	0.004526048
148	0.004477291
125	0.004273505
126	0.004235072
130	0.004158892

Table 5.1: The 10-most critical residues as identified by computing the $\delta_\cdot$ parameters in data $\mathbf{D}$. The index of the residue is $k \in \{1, 2, \ldots, 156\}$.

5.3.2 Criticality using η

We computed η_k for $k = 1, \ldots, p = 156$, and plotted the degree distribution against block index in Figure 5.7. The corresponding heat map also be presented.

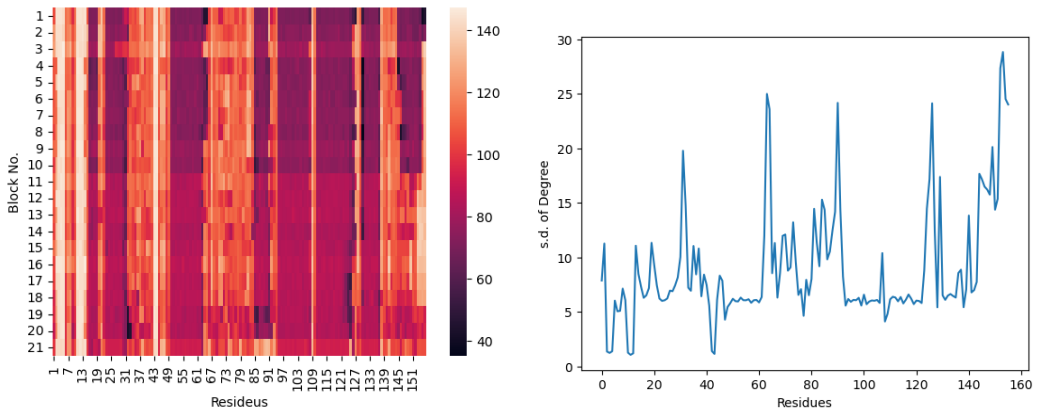


Figure 5.7: Left panel: a heat map representation of the degree distribution of the k-th node, in each of the N_b RGGs that are learnt at $\tau = \tau_{min} \approx 0.854$, given each row-wise partitions of the data $\mathbf{D}$, for all $k \in \{1, 2, \ldots, 156\}$ and across all the $N_b = 21$ number of such partitions. Right panel: the corresponding η_k is plotted against the residue index k, for $k = 1, \ldots, 156$, using the RGG learnt for data $\mathbf{D}$.

Table 5.2 shows the 10 most critical residues identified by computing the η_c parameter at $\tau \approx 0.854$, i.e. the 10 residues that show maximal temporal instability in their interaction with other residues, during the evolution of the protein, as captured by the simulations of the protein *skp1*.

Residue index k	η_k
154	28.83552546
153	27.35157845
64	24.99451187
155	24.52080425
91	24.17322219
127	24.1343556
156	24.02757297
65	23.5788829
150	20.12489349
32	19.77882236

Table 5.2: 10 residues with the highest value of the $\eta_.$ parameter at $\tau \approx 0.854$, in the data $\mathbf{D}$. Here $k = 1, ..., p = 156$.

5.3.3 Criticality using β

Table 5.3 shows the 10 most critical residues identified by the 10 smallest values of the degree distribution β_k at $\tau \approx 0.854$, over the evolution time of the protein, recorded by $\mathbf{D}$.

Residue index k	β_k
146	13
147	13
151	13
148	15
149	17
152	18
150	22
130	26
125	33
126	35

Table 5.3: Residues with the 10 smallest $\beta.$ values at $\tau \approx 0.854$, given data $\mathbf{D}$.; Here $k = 1, ..., p = 156$.

5.4 Comparison of identified criticality, with experimental results

It can be seen that the results of the three parameters show some levels of concurrence with each other, i.e. the residues that have the largest δ_k, are the residues that have the least β_k. There is indeed unevenness of the η_k parameter and the critical residues we identified, using the highest unevenness of the η_k parameter over time, tally moderately well with the residues that have been identified as critical in our other parameter, as well as the experimental results identify in the laboratory from our biologists collaborators. We tabulate these results below.

Residues identified by δ	Residues identified by η	Residues identified by β	Exprimental result
150	154	146	156
147	153	147	155
151	64	151	154
152	155	148	153
149	91	149	151
146	127	152	150
148	156	150	149
125	65	130	148
126	150	125	147
130	32	126	133
63	145	145	121
64	130	63	120
153	126	64	81
145	146	91	80
154	147	31	78
32	148	154	77
31	149	108	76
137	152	137	75
91	85	153	74
85	33	30	73

Figure 5.8: A comparison of the residues identified as critical, using the three parametrisations $\delta.$, $\eta.$, and $\beta.$ we introduce above, and those found critical via laboratory-based experiments. The shaded residues indicated themselves to be identified by all measures, given data $\mathbf{D}$. 20 of the most critical residues are named.

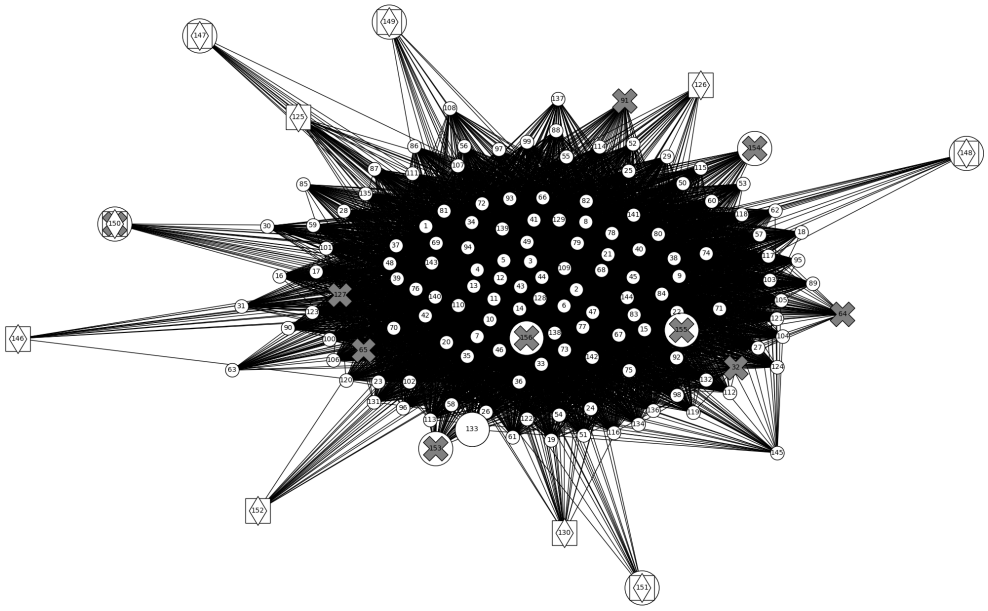


Figure 5.9: Critical residues shown in Figure 5.8, depicetd by $\delta.$ (as squares); $\eta.$ (grey crosses); $\beta.$ (diamonds); and experimental results (large white circles), marked on an RGG learnt of data $\mathbf{D}$.

Among the three criticality measures introduced above, the $\delta.$ parameter aligns most closely with experimental findings. Unlike approaches that rely on explicit physical or structural models, $\delta.$ directly captures each residue's contribution based

on the learnt random graph variable solely, given the evolution of the protein. We believe that $\delta_\cdot$ is robust across proteins of different sizes of graphs and networks, providing a consistent means of identifying functionally critical residues. We also notice that by design, $\eta_\cdot$ measures the temporal variability in the connectivity of residue, and it can accommodate cases where nodes cluster into multiple, isolated communities within the full graph. In such scenarios, a critical residue which is expected to initiate or terminate contacts more abruptly than its neighbours, will still yield a relatively higher standard deviation in its degree over time. Also, we appreciate that $\eta_\cdot$ is sensitive to two implementation details: (1) the length of each temporal block used to compute degree fluctuations, given the timescale of inter-residue interaction evolution in the MD simulations, and (2) the choice of cut-off τ when constructing the residue-interaction graph from dataset $\mathbf{D}$. An arbitrary partition τ may therefore impair $\eta_\cdot$'s performance. Similarly, the $\beta_\cdot$ parameter is also sensitive to the cut-off threshold used to draw edges in the RGG, given the data - which again illustrates the importance of the choice of cut-off in the learning of RGG. In general, our application of RGG learning contributes to protein design, by providing a new robust way of identifying critical residues. All three parameters ($\delta_\cdot$, $\eta_\cdot$, and $\beta_\cdot$) identify residues beyond those already examined in published mutagenesis studies, suggesting new candidates for experimental validation. Future work might involve the mutagenesis experiments at these predicted residues.

Chapter 6

Summary, Contribution and Further Work

6.1 Summary of the thesis

In my thesis, I discussed the research that I have undertaken on a new learning of random graph variables, and the computation of distances/divergences between the posterior probabilities of a pair of learnt graphs. I have discussed applications of these techniques to various real-world problems, within a Bayesian framework. I learn the graph of a multivariate data set by collating realisations of a random graph variable, learnt given a real-world data, to then compute the aforesaid inter-graph statistical distance/divergence between graphs learnt given two such data sets that are generated within distinct experimental conditions. Such inter-graph distance/divergence then informs on the strength of the effect of the difference in the two sets of experimental conditions.

The graph of a given multivariate dataset that I learnt, followed the computation of the correlation matrix of the data set - the correlation matrix here is such that any of its elements is the correlation between a pair of variables, observed values of which populate two columns of the given dataset. To make an inference on an edge in the graph, I employed Bayesian sampling techniques - including MCMC and rejection sampling - to sample from the posterior probability of the edge variable, conditional on the correlation between the random variables that are attached to the nodes that straddle the considered edge. Thus, it implies that I know the edge posterior conditional on said correlation (or partial correlation in some of my applications). Chapter 1 provides an overview of prior work on the kind of random graphs that I study (RGGs drawn in probabilistic metric spaces); statistical distance measures; and relevant studies on MCMC and Gaussian Processes that formed the foundational and warming-up phase of my doctoral research. Then in Chapter 2, I presented the details of my graph learning methods, along with the study of threshold selection on the cut-off probability used in the acceptance (or not) of an edge, in the learnt RGG.

My subsequent chapters were dedicated to distinct real-world applications of the considered techniques. In Chapter 3, I present the research that develops a

continuous-valued score that parametrises the severity with which a disease (called VOD) develops in individual patients, following their Bone Marrow Transplantation. To achieve this score, I learn the RGG of each patient's chart, i.e. the time series record of observations on multiple physiological parameters, over a time interval around the patient's transplant. These physiological parameters include blood pressure, temperature, pulse rate, etc. Then the Hellinger distance between the learnt RGG and a reference graph (learnt from one particular patient who has been chosen to be a "reference patient") yields a data-driven patient-specific score, on a scale in which the "reference patient" has been assigned an arbitrarily-chosen severity score, (of 1 in our work).

In Chapter 4, I applied the graph learning technique on clinical data, i.e. time series recordings of locations of 20 joints of the skeleton of each patient, while they engaged in executing an exercise, (referred to as "playing an exergame"). By computing the distance between graphs generated from successive exergame play sessions, we derive the Mobility Recovery Score relevant to each session. Thus, this application highlights a method for the learning of recovery trajectories that enables early, patient-specific recommendations for optimal rehabilitation pathways, before physical therapy begins. While the prediction of a recovery trajectory is awaited, even with the current results, we provided strong recommendations for selecting the most appropriate exergames, based on categorised levels of impairment. Here, severity is proxied by the patient's initial performance score, automatically recorded by an e-platform during gameplay. In broader applications, the initial mobility state may also be informed by other clinical assessments. We propose our method as a valuable contribution to the ongoing pursuit of early, individualised diagnosis and treatment planning. The protocol is not only automated and efficient - making it suitable for small or larger patient cohorts, but also fast in predicting optimal exergames. Most importantly, the recovery trajectories learned through this framework are both robust and explainable.

Chapter 5 discusses a contribution to protein design. Three distinct methods for the automated identification of critical residues have been introduced, with the method that permits the learning of RGGs in probabilistic metric spaces, where such RGGs are objectively and meaningfully thresholded. Such RGGs are learnt from molecular dynamical simulation data. Our work is indeed foundational in terms of real-world scenarios - identifying critical residues in proteins is essential since these residues often play key roles in maintaining structural integrity, enabling catalytic activity; facilitating molecular recognition; and regulating function. Understanding which residues are critical allows scientists to interpret the functional impact of genetic mutations, many of which are linked to diseases. In practical applications, this knowledge underpins drug development; protein engineering; and synthetic biology.

So in summary, my thesis utilises a newly developed theory for learning a random graph of a multivariate dataset, the inter-variable correlations of which are known or can be computed. While our graph-learning approach offers a powerful new lens for examining dataset structure and - as we presented in the previous chapters - finds many practical applications, we also explored the incompleteness in the current theory. In particular, I have focused on how we can choose the cut-off probability τ

on the edge posterior probability, allowing for acceptance of this edge in the pursued RGGs. Let me clarify. Recall that by definition, an RGG places an (undirected) edge between two nodes whenever the distance between them is below a certain threshold. In an RGG drawn in a probabilistic-metric space, each potential edge is represented by a probability distribution over positive support, and we include that edge in the graph only if its associated probability exceeds a cut-off parameter τ. Choosing τ "meaningfully" is crucial - if τ is set too high, very few edges will be accepted and the resulting graph will be overly sparse, potentially missing important structural relationships in the data from the visualised graph. Conversely, if τ is too low, almost every pair of nodes will become connected, yielding a dense graph that obscures meaningful patterns. In my thesis, I derive a method for selecting this threshold τ, so that the learnt graph faithfully represents the correlation structure of the data. In the future, I will seek other possible approaches to explore and reinforce our threshold choice.

The contributions of my doctoral work are included in Section 1.4, Chapter 1.

6.2 Publications arising from this thesis

My following publications have resulted in part, from the research presented in this thesis:

1. Chakrabarty et al. (2023) "Constructing a Training Set Using Distance Between Learned Graphical Models of Time Series Data on Patient Physiology, to Predict Disease Scores," *PLOS ONE* (2023). DOI: 10.1371/journal.pone.0292404

— This paper developed a new method for constructing training sets using probabilistic graph models to predict disease susceptibility in noisy, multi-parameter data spaces, in Chapters 3 of the thesis.

2. Zhang et al. (2024) "Individualised Recovery Trajectories of Patients with Impeded Mobility, Using Distance Between Probability Distributions of Learned Graphs," *Artificial Intelligence in Medicine* (2024). DOI: 10.1016/j.artmed.2024.103005

— This paper developed personalised recovery predictions using Bayesian graphical models from rehabilitation data, introducing a Movement Recovery Score (MRS) based on statistical distances, in Chapters 4 of the thesis.

3. Zhang et al. (2025) "Identifying critical residues of a protein using meaningfully-thresholded Random Geometric Graphs" *Journal of Computational and Graphical Statistics* (under review)

— This paper focus on identifying critical residues - specific amino acids essential for protein, by proposing three different parametrisations based on learning from the random graphs produced by the molecular simulations of the protein, and provides a novel way to characterise residue criticality and structural dynamics, which is crucial for understanding protein function, disease-related mutation analysis and drug design. This part of the work is in Chapter 5 of this thesis.

6.3 Future work

To further demonstrate the utility of my graph learning technique, I will consider exploring several other real-world opportunities of the same, and for the usage of computing inter-graph distance/divergence as a fully data-driven, scalar-valued single index of a multivariate effect, and the evolution/variation of the same. The relationship between such an effect and a vector representing the experimental conditions can then be learnt - via probabilistic supervised learning - to possibly enable/facilitate prediction and forecasting.

One application that I am keen on is the establishment of the connection between the time series of the invasive test results (such as Ultra SonoGraph (USG) features and some other post-transplant observations) in recipients of Bone Marrow Transplants, and their score of the severity with which such a patient develops a disease called VOD, post-transplant. Such a score is learnt using graphs of the time series data that comprises observations of physiological parameters recorded during a pre-transplant to a post-transplant time interval. After computing an inter-graph divergence between graphs learnt for two patients, given data on their respective invasive tests, I will seek the ranking of such a divergence measure between any patient in the considered cohort and the pre-chosen "reference patient". I will then compare such a ranking with that of the VOD severity score. This will inform on the validity of the easily-recovered, non-invasive VOD score, as a viable substitute for invasive tests.

For the application in Chapter 4, in future work, we aim to extend our graph learning approach to predict the complete recovery trajectory of a new patient who is about to start their physical rehabilitation programme, if the patient were to exercise using a given exercise routine. Achieving this requires a training dataset suitable for supervised learning, allowing us to model the relationship between a patient's pre-rehabilitation mobility impairment and their subsequent recovery trajectory. This trajectory is defined by the patient's performance over a sequence of k-exergames, as shown in Figure 4.9. We can model our graph observable of the data as a sample from a Gaussian Process, as we introduced in Appendix C. Using supervised learning, we can predict a patient's recovery path based on their initial level of mobility impairment at the start of therapy. This capability offers clinicians a robust, early-stage, and patient-specific assessment of the anticipated rehabilitation process.

I am also very keen on undertaking the prediction and forecasting of states that the residues in the simulated protein will attain. I plan on undertaking forecasting within brief time intervals post a temporal mark within the simulated dataset, and proceeding in a stepwise fashion thereafter. For the prediction exercise, I look forward to undertaking this at times between two identified temporal points within the dataset. I look forward to the possibility of a successful prediction that can compensate for the lack of temporal resolution in the undertaken simulations. My learnt inter-residue correlation matrix - which is interesting to compute given that states attained by any residue are categorical by nature - will be the crucially important tool in this exercise.

Appendix A

Reference Table of MCMC Implementation

Algorithm	Prior	Proposal	Notes
Random Walk	Uniform	$N(\beta_0^{(current)}, 4^2)$ $N(\beta_1^{(current)}, 4^2)$	95% HPD of β_0 : (1.5523533, 35.2633635) 95% HPD of β_1 : (-0.53081846, -0.03483466)
		$\sigma_0 = (2\beta_0^{(0)})^2$ $\sigma_1 = (2\beta_1^{(0)})^2$	95% HPD of β_0 : (1.3165861, 35.66710102) 95% HPD of β_1 : (-0.53717194, -0.03592072)
	$\pi_0(\beta_0) = N(0, 4^2)$ $\pi_0(\beta_1) = N(0, 4^2)$	$\sigma_0 = (2\beta_0^{(0)})^2$ $\sigma_1 = (2\beta_1^{(0)})^2$	95% HPD of β_0 : (-2.40103167, 5.0145067) 95% HPD of β_1 : (-0.08898875, 0.02036492)
	$\pi_0(\beta_0) = N(0, 4^2)$ $\pi_0(\beta_1) = N(0, 4^2)$	$\sigma_0 = (0.2\beta_0^{(0)})^2$ $\sigma_1 = (0.2\beta_1^{(0)})^2$	95% HPD of β_0 : (5.82341996, 15.2169578) 95% HPD of β_1 : (-0.23725697, -0.09635899)

Algorithm	Prior	Proposal	Notes

Vanilla MCMC	$\pi_0(\beta_0) = N(0, 4^2)$ $\pi_0(\beta_1) = N(0, 4^2)$	Truncated Normal $\sigma_0 = (2\beta_0^{(0)})^2$ $\sigma_1 = (2\beta_1^{(0)})^2$	95% HPD of β_0 : (2.40161169, 35.46921622) 95% HPD of β_1 : $(-0.53239835, -0.0486979)$
	$\pi_0(\beta_0) = N(0, 4^2)$ $\pi_0(\beta_1) = N(0, 4^2)$	Truncated Normal $\sigma_0 = (1\beta_0^{(0)})^2$ $\sigma_1 = (2\beta_1^{(0)})^2$	95% HPD of β_0 : (2.26894305, 34.74050198) 95% HPD of β_1 : $(-0.52220068, -0.04774905)$
	$\pi_0(\beta_0) = N(0, 4^2)$ $\pi_0(\beta_1) = N(0, 4^2)$	Truncated Normal $\sigma_0 = (0.1\beta_0^{(0)})^2$ $\sigma_1 = (0.2\beta_1^{(0)})^2$	95% HPD of β_0 : (2.83580678, 38.8319938) 95% HPD of β_1 : $(-0.58180209, -0.05575383)$

Algorithm	Prior	Proposal	Notes
Independent Samplers	$\pi_0(\beta_0)$ $= N(10, 2^2)$ $\pi_0(\beta_1) = N(0, 2^2)$	Normal $\sigma_0 = (\beta_0^{(0)})^2$, $\sigma_1 = (\beta_1^{(0)})^2$ with fixed $\mu_0 = 0$, $\mu_1 = 0$	95% HPD of β_0 : (3.14200287, 6.5259754) 95% HPD of β_1 : $(-0.08788488, -0.05989835)$
	$\pi_0(\beta_0)$ $= N(15, 2^2)$ $\pi_0(\beta_1) = N(0, 2^2)$	Normal $\sigma_0 = (0.01\beta_0^{(0)})^2$, $\sigma_1 = (0.2\beta_1^{(0)})^2$ with fixed $\mu_0 = 15$,$\mu_1 = 0$	95% HPD of β_0 : (14.62630413, 15.0306031) 95% HPD of β_1 : $(-0.01390083, -0.00347737)$
	$\pi_0(\beta_0)$ $= N(10, 2^2)$ $\pi_0(\beta_1) = N(0, 2^2)$	Truncated Normal $\sigma_0 = (1\beta_0^{(0)})^2$ $\sigma_1 = (2\beta_1^{(0)})^2$ with fixed $\mu_0 = 15, \mu_1 = 0$	95% HPD of β_0 : (2.2444736, 4.3730422) 95% HPD of β_1 : $(-0.0773282, -0.00759171)$
	$\pi_0(\beta_0)$ $= N(15, 2^2)$ $\pi_0(\beta_1) = N(0, 2^2)$	Truncated Normal $\sigma_0 = (0.1\beta_0^{(0)})^2$, $\sigma_1 = (0.01\beta_1^{(0)})^2$ with fixed $\mu_0 = 15, \mu_1 = 0$	95% HPD of β_0 : (14.60755017, 15.02576763) 95% HPD of β_1 : $(-4.6752698, -1.5494135)$

Algorithm	Prior	Proposal	Notes

Gibbs Samplers	$\pi_0(\beta_0) = N(0, 2^2)$ $\pi_0(\beta_1) = N(0, 2^2)$	$N(\beta_0^{(current)}, \sigma_0^2)$ $N(\beta_1^{(current)}, \sigma_1^2)$ with $\sigma_0 = (2\beta_0^{(0)})^2$ $\sigma_1 = (2\beta_1^{(0)})^2$	95% HPD of β_0 : $(-2.05632582,\ 4.87334981)$ 95% HPD of β_1 : $(-0.08748003,\ 0.01458658)$
	$\pi_0(\beta_0)$ $= N(10, 2^2)$ $\pi_0(\beta_1) = N(0, 2^2)$	$N(\beta_0^{(current)}, \sigma_0^2)$ $N(\beta_1^{(current)}, \sigma_1^2)$ with $\sigma_0 = (0.2\beta_0^{(0)})^2$ $\sigma_1 = (0.2\beta_1^{(0)})^2$	95% HPD of β_0 : $(6.78419035,\ 14.19235037)$ 95% HPD of β_1 : $(-0.22140035,\ -0.10904357)$
	$\pi_0(\beta_0) = N(0, 2^2)$ $\pi_0(\beta_1) = N(0, 2^2)$	Truncated Normal $\sigma_0 = (\beta_0^{(0)})^2$ $\sigma_1 = (\beta_1^{(0)})^2$	95% HPD of β_0 : $(2.05607863,\ 31.92206533)$ 95% HPD of β_1 : $(-0.4812998,\ -0.04486532)$

Appendix B

Learning Graph Parameters Using MCMC

To learn the unknown edge parameters G_{ij} and variance parameter ν_{ij}, we undertake Bayesian inference and sample the unknowns by MCMC, which has been introduced in Chapter 1. Chakrabarty (2023) applied the Metropolis-with-2-block-update algorithm, namely, update the inter-column correlation matrix of the dataset in the first part of any iteration of this inference strategy, and thereafter, in the second block, the edges are updated, conditional on the updated correlation matrix. It can be seen that the updated edge variable does not affect the correlation updating in the first block of the next iteration - so in each iteration of an MCMC chain, firstly update the value of inter-column correlation matrix $\mathbf{\Sigma}^{(C)}$ given by the data $\mathbf{D}$, and then in the second block of the same iteration, update the RGG edge variable, given the updated partial correlation matrix $\mathbf{\Psi} = [\psi_{ij}]$, as we defined in Chapter 2 (we transformed $\mathbf{\Sigma}^{(C)}$ to the partial correlation matrix $\mathbf{\Psi}$ at every iteration of the MCMC chain, once $\mathbf{\Sigma}^{(C)}$ is updated).

Recall the posterior of the $p \times p$-dimensional inter-column correlation matrix $\mathbf{\Sigma}^{(C)}$ of the given dataset $\mathbf{D}$ that has been defined, to generate values of this random inter-column correlation matrix using the Bayesian inference. The updated $\mathbf{\Sigma}^{(C)}$ is then fed into the learning of graph parameters, G_{ij} and variance parameter ν_{ij}, $\forall i, j = 1, ..., p, i \neq j$, that has been defined, along with their joint posterior probability density. We also need to consider the prior and proposal distributions of the unknowns in the implementation of the inference strategy: for the 1st block of the t-th iteration, it is clearly reasonable that correlations are proposed from a Truncated normal density that is left truncated at -1 and right truncated at +1, for the 2nd block of this t-th iteration, the RGG is updated, given the updated partial correlation matrix $\mathbf{\Psi}$ in 1st block. And the proposal density of the standard deviation parameter ν_{ij} is Normal with a mean that is equal to the updated value in the previous iteration, $\nu_{ij}^{(t-1)}$. The proposal distribution of G_{ij} is $Bernoulli(\mid \psi_{ij}^{(t)} \mid)$. Also weak priors have been applied, $Bernoulli(0.5)$ for the prior of G_{ij}, and $Uniform(0, 1]$ for the prior of ν_{ij} and we then have the joint posterior probability density of the unknowns G_{ij} and ν_{ij}, given the learnt $|\psi_{ij}|$, $\forall i, j = 1, ..., p, i \neq j$, as we defined in the Equation 2.2.2.

Remark B.0.1. *Considering the partial correlation Ψ_{ij} between r.v.s X_i, X_j - via the learning of inter-column correlation matrix $\mathbf{\Sigma}^{(C)}$, the joint posterior probability of the binary edge parameters G_{ij}, and variance parameter ν_{ij}, $\forall i, j = 1, ..., p, i \neq j$, given by dataset $\mathbf{D}$ is defined as follows:*

$$\pi(G_{ij}, v_{ij}|\psi_{ij}) = k\frac{1}{\sqrt{2\pi\nu_{ij}}}\exp[-\frac{(G_{ij} - |\psi_{ij}|)^2}{2\nu_{ij}}],$$

here k is a positive global constant, which can be omitted during MCMC.

Notice that although the proposed density of $\nu_{ij} > 0$ has been chosen to be truncated normal, to simplify the calculation in MCMC, we can replace the variance parameter with the square of the standard deviation parameter ν_{ij} - thus, the difference $(G_{ij} - |\psi_{ij}|)^2$ is normalised by ν_{ij}^2, which allow us then choose the proposed density of this standard deviation parameter ν_{ij} to be normal instead.

With the set-up of prior and proposal along with the joint posterior at hand, we can now process the MCMC learning of unknowns. Notes that from the previous study by Chakrabarty (2023), the variance parameter has been detected with a nature of instability - we will still make an attempt at learning this ν_{ij}, $\forall i, j = 1, ..., p, i \neq j$.

B.1 Learning variance parameter matrix with equal elements

Let us consider the standard deviation parameter matrix where all elements are equal, i.e. we want to learn only one value of all ν_{ij} parameters, for $i, j \in \{1, ..., 11\}, i \neq j$. For a chosen seed $\nu_{(seed)} = 0.99$, With prior $Uniform(0, 1]$ for the prior of ν_{ij}, and proposal density of the standard deviation parameter ν_{ij} to be Normal - with a mean to be the updated value in the previous iteration and jump scale $\mathbf{\Sigma} = 0.001$, the trace of ν for 500k iterations of MCMC are as follows:

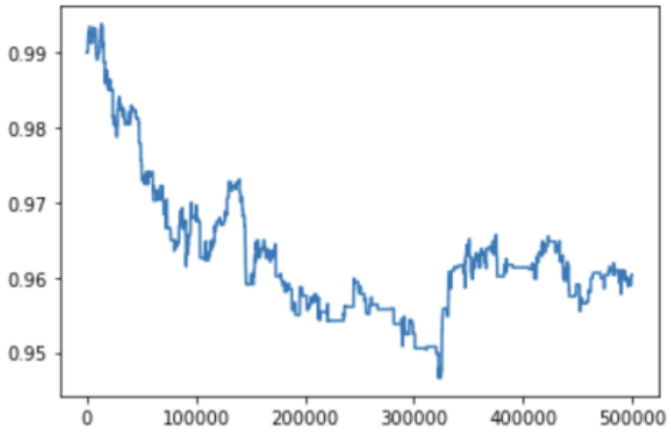


Figure B.1: ν trace with $\nu_{(seed)} = 0.99$, jump scale $\mathbf{\Sigma} = 0.001$

From the figure above, it can be seen that the trace only shows a trend of convergence roughly after 325,000 iterations. Notice that longer iterations could be generated to improve the convergence.

Simultaneously, the graphical model was learnt - the posterior of the RGG variable $\mathcal{G}_{\boldsymbol{V},m}(\boldsymbol{\Psi},\tau)$ learning by using the edge posterior $m(\cdot|\cdot)$, given the known partial correlation matrix Ψ of data $\mathbf{D}$ has been defined.

Remark B.1.1. *As we defined previously in Definition 2.2.7 and Definition 2.2.10, we obtained the marginal posterior probability of G_{ij} conditional on the partial correlation $\boldsymbol{\Psi}_{ij}$ between the nodes i and j, as following:*

$$m(G_{ij}|\boldsymbol{\Psi}_{ij}) = K\left[\sqrt{\frac{2}{\pi}}\exp\left(\frac{-(S_{ij})^2}{2}\right) - (S_{ij})\text{erfc}\left(\frac{|S_{ij}|}{\sqrt{2}}\right)\right],$$

where disparity $S_{ij} = |G_{ij} - |\boldsymbol{\Psi}_{ij}||$ as defined previously and the complementary error function $\text{erfc}(\cdot) = 1 - \text{erf}(\cdot)$.

Then given the edges to form independently of each other, the posterior probability of the RGG variable $\mathcal{G}_{\boldsymbol{V},m}(\boldsymbol{\Psi},\tau)$, conditional on the partial correlation of data $\mathbf{D}$, for all i,j pairs, of the conditional edge marginal, at a pre-chosen τ given as follows:

$$\pi(\mathcal{G}_{\boldsymbol{V},m}(\boldsymbol{\Psi},\tau)) = \prod_{ij\in\mathbf{V};i<j} m(G_{ij}|\psi_{ij})$$

The realisation of the RGG generated by these 500,000 MCMC iterations, at different τ values that were chosen as $\tau = 0.05, 0.4, 0.6, 0.8$ and 0.9, is shown below.

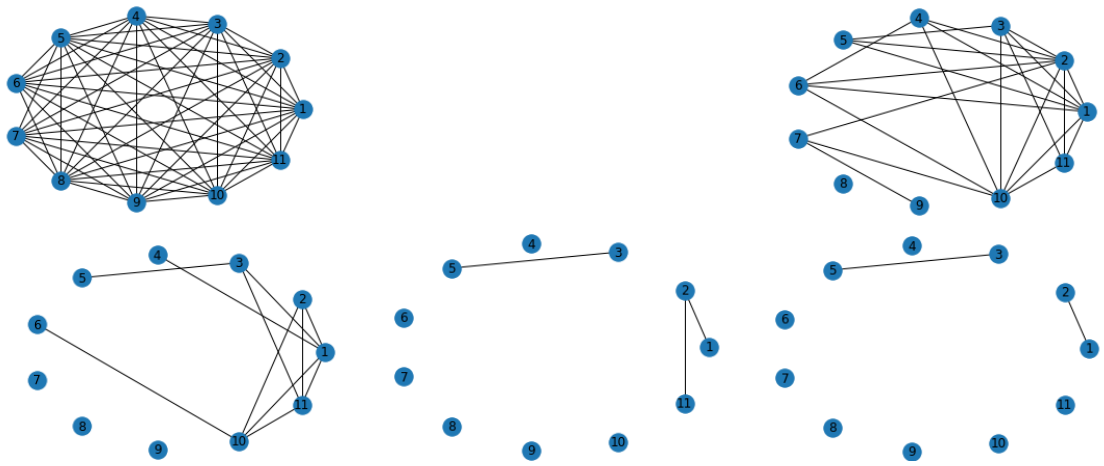


Figure B.2: RGG learnt given $\mathbf{D}$ at chosen $\tau = 0.05, 0.4, 0.6, 0.8$ and 0.9 respectively. By the nature of RGG, the larger τ we used, the fewer edges will be accepted in the graphs.

B.2 Learning variance parameter matrix with distinct elements

Considering a more realistic choice, the standard deviation parameter $\nu_{ij}, i, j \in \{1, ..., 11\}, i \neq j$ matrix, where each ij pair of ν_{ij} is assigned by distinct values.

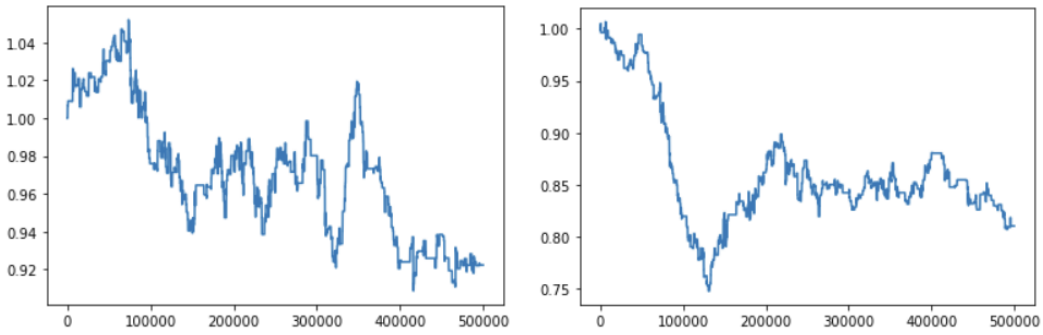


Figure B.3: Traces of some elements of learnt standard deviation matrix, $\nu_{(2,4)}$ and $\nu_{(8,10)}$ respectively, for 500k MCMC iterations with same prior and proposal as Figure B.1, and seed value $\nu_{(seed)} = [\nu_{ij}] = 1$ for all $i, j = 1, 2, ..., 11$.

We know that more iterations will help to learn a clearer trend of convergence of the unknown, we apply a post-burnin $N_b = 100,000$ - we plot histograms generated after 100,000 iterations of the same 500,000 MCMC iterations in Figure B.3. We also generated graph realisation at different chosen τ ($\tau = 0.05, 0.4, 0.6, 0.8$ and 0.9 respectively).

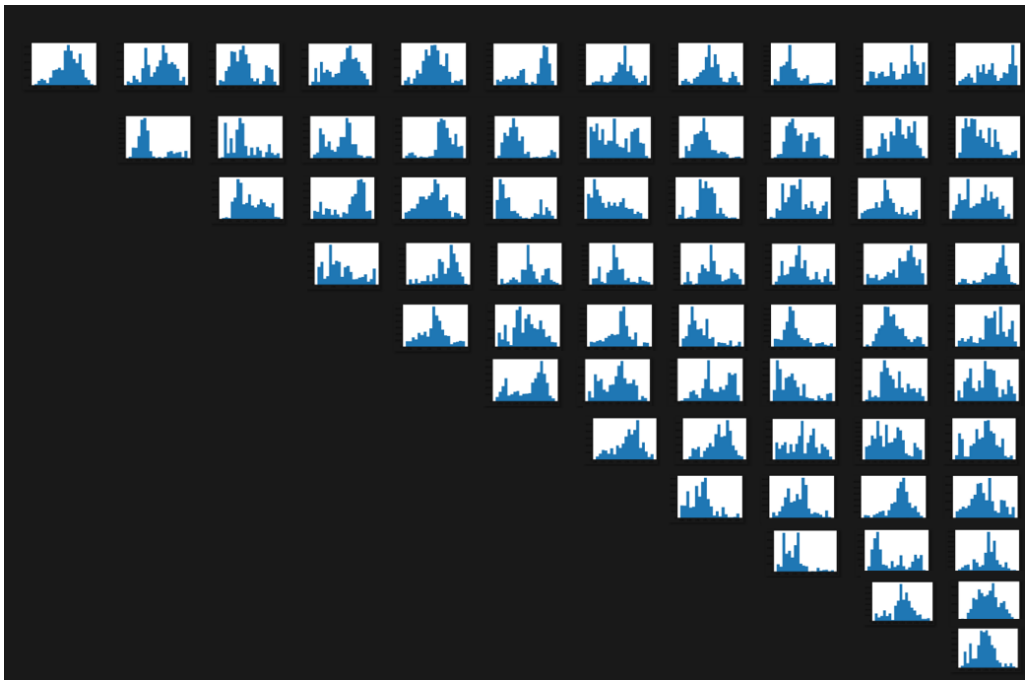


Figure B.4: Corresponding histograms for each element of $[\nu_{ij}], i, j = 1, 2, ..., 11$, according to the 500k MCMC iterations in Figure B.3, with post-burnin $N_b = 100k$.

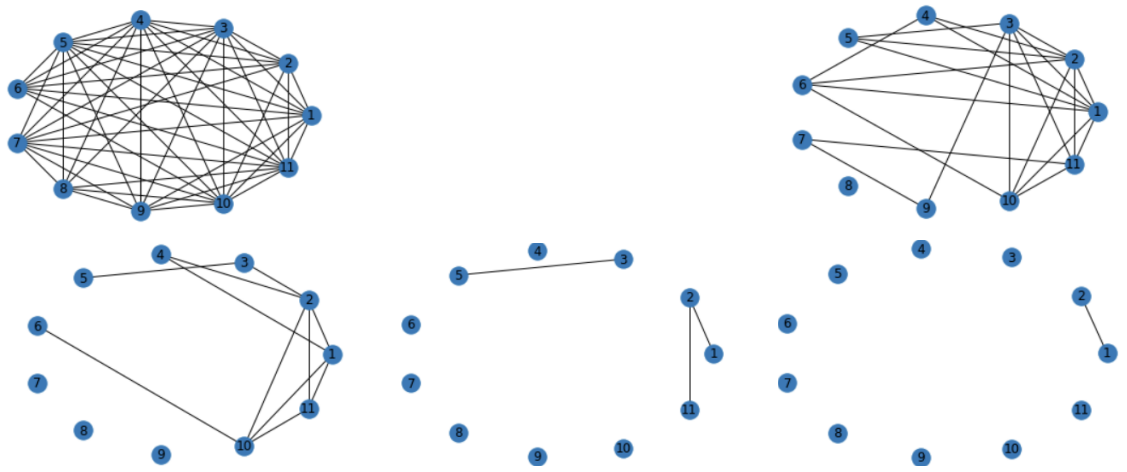


Figure B.5: Graph generated at $\tau = 0.05, 0.4, 0.6, 0.8$ and 0.9 respectively

From the MCMC traces and histogram of the ν_{ij}, $\forall i, j = 1, ..., p, i \neq j$, we attempted, we can improve the better convergence of each element ν_{ij} trend by sequentially update the learning on the unknown - we tweak some parameters individually and generate an updated MCMC accordingly, such as choose strong prior, or instead of assigning the same proposed density of each elements in ν_{ij}, i.e. $\nu_{ij}^{(t)} \sim \mathcal{N}(\nu_{ij}^{(t-1)}, 0.001)$ for all i, j at t-th iteration in our original setting, we can tweak $\mathbf{\Sigma}_{ij}$ to be a coefficient times to the mean of ν_{ij} generated from their distributions obtain by the pervious MCMC. While due to the nature of instability, as we obtained the marginal posterior probability of G_{ij} conditional on the partial correlation Ψ_{ij} between the nodes i and j, by integrating over the variance parameter, $m(G_{ij}|\Psi_{ij})$, we can learn the edge variable G_{ij},$\forall i, j = 1, ..., p, i \neq j$, directly, using MCMC technique.

B.3 Learning Graph Edge Parameter G_{ij} directly

The following figure is the graph results of 500,000 MCMC iterations with a $Bernoulli(0.5)$ for the proposal distribution $Bernoulli(\mid \psi_{ij}^{(t)} \mid)$ applied on the edge variables G_{ij},$\forall i, j = 1, ..., p = 11, i \neq j$.

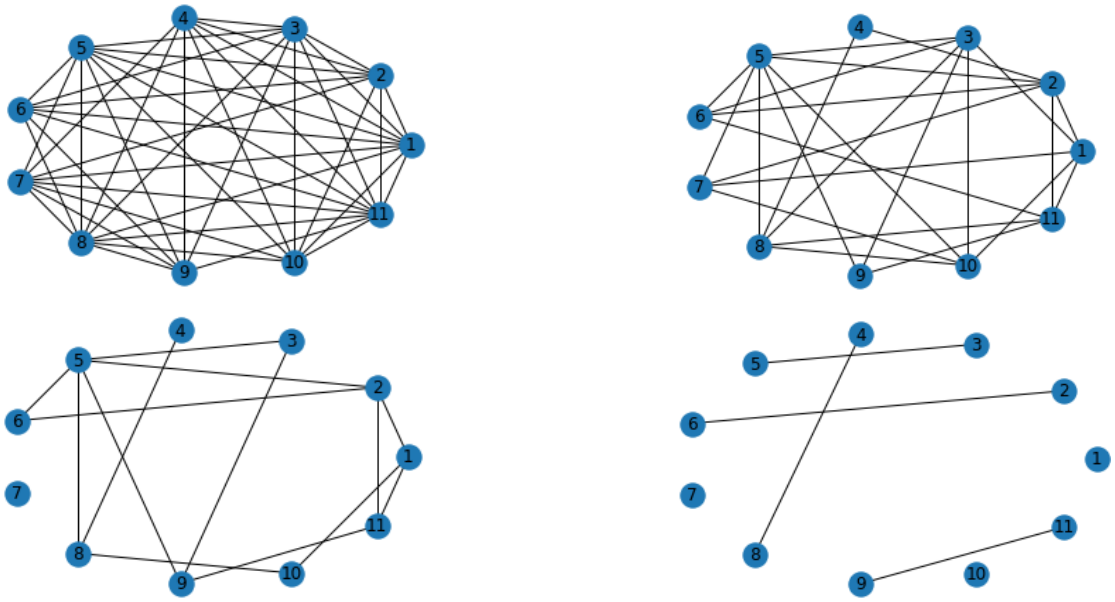


Figure B.6: Graph generated by learning edge parameter G_{ij} only, $\forall i, j = 1, ..., p, i \neq j$, from left to right and top to bottom, we have graphs realisation of $\mathcal{G}_{\boldsymbol{V},m}(\boldsymbol{\Psi}, \tau)$ at different cutoff $\tau = 0.05, 0.4, 0.6$ and 0.9 respectively

It would be worth comparing the graph realisation of $\mathcal{G}_{\boldsymbol{V},m}(\boldsymbol{\Psi}, \tau)$, via learning the variance parameter together, with our approach of learning the edge variable directly. We also compare graphs generated by one-value variance learning vs those generated by the variance matrix with distinct elements.

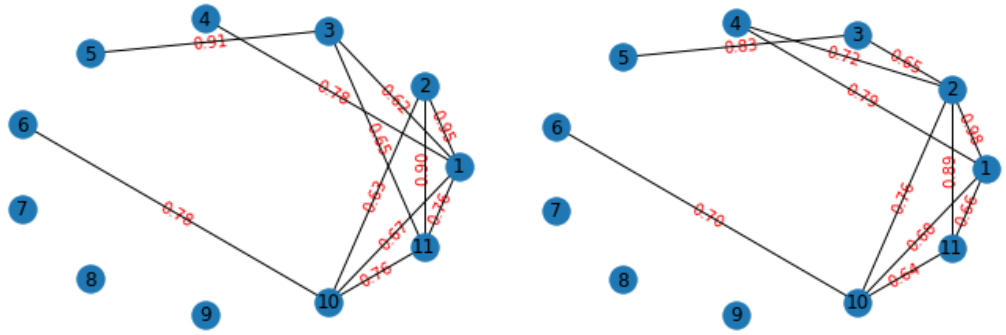


Figure B.7: Left panel: graph realisation shown in Figure B.2, for one value variance learning case, with labelling probabilities, at $\tau = 0.6$.; Right panel: graph realisation shown in the Figure B.5 with labelling probabilities, for the variance parameter matrix which contains distinct values of elements, at cutoff $\tau = 0.6$. Post-burin $N_b = 100,000$ has been applied in both cases.

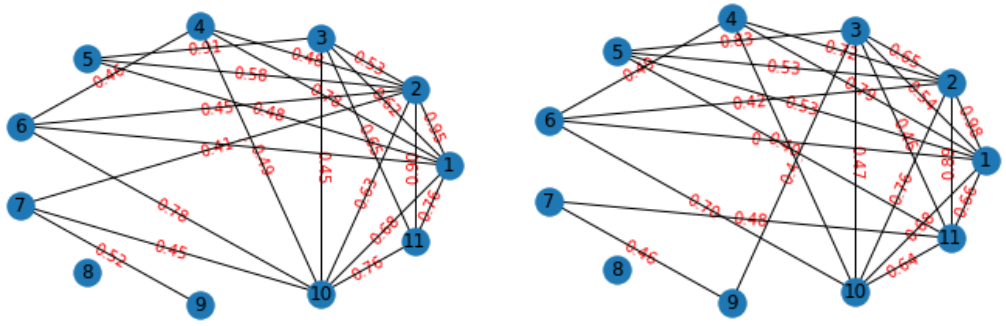


Figure B.8: Left panel: graph realisation shown in Figure B.2, for one value variance learning case, with labelling probabilities, at $\tau = 0.4$.; Right panel: graph realisation shown in the Figure B.5 with labelling probabilities, for the variance parameter matrix which contains distinct values of elements, at cutoff $\tau = 0.4$. Post-burin $N_b = 100,000$ has been applied in both cases.

We compare the graphs generated using different cutoff τ in both cases, namely, learning one value ν variance and learning ν_{ij} variance matrix. It can be seen that in general, when learning both graph parameters G_{ij} and ν_{ij} at the same time, the graphs learnt at a given τ are robust to the choice of the way to update the standard deviation parameter. While the graphs learnt when only the edge parameter G_{ij}, by sampling from the corresponding edge marginal posterior, are distinct from the graphs learnt when edges and standard deviation parameters are sought simultaneously, as shown in Figure B.9. One possible reason is that there are biases that we are bringing into the implementation when the standard deviations have not converged well enough.

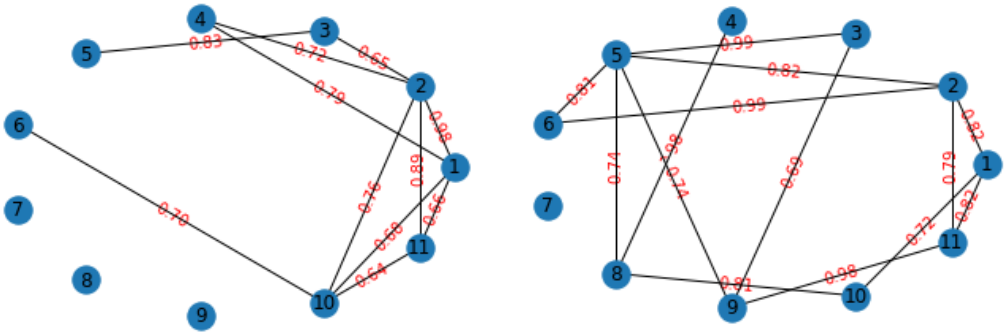


Figure B.9: Left panel: graph realisation shown in Figure B.5, with labelling probabilities, for the variance parameter matrix which contains distinct values of elements, at $\tau = 0.6$; Right panel: graph realisation shown in the Figure B.6 via learning edge parameter G_{ij} directly, $\forall i, j = 1, ..., p, i \neq j$, at cutoff $\tau = 0.6$.

B.4 Further Study on the Prior

In the approaches we discussed previously, we have learnt the standard deviation parameter matrix ν_{ij} and graph edges matrix G_{ij}, $\forall i, j = 1, ..., p = 11, i \neq j$, by using the Metropolis-Hastings-with-2-blocks-updated technique. A uniform prior was chosen for learning the variance parameter ν. Recall that using a uniform prior, we have generated a chain for 500,000 iterations with post-burnin $N_b = 100000$, the multi-panel histogram and graph results are shown as Figure B.4 and Figure B.5.

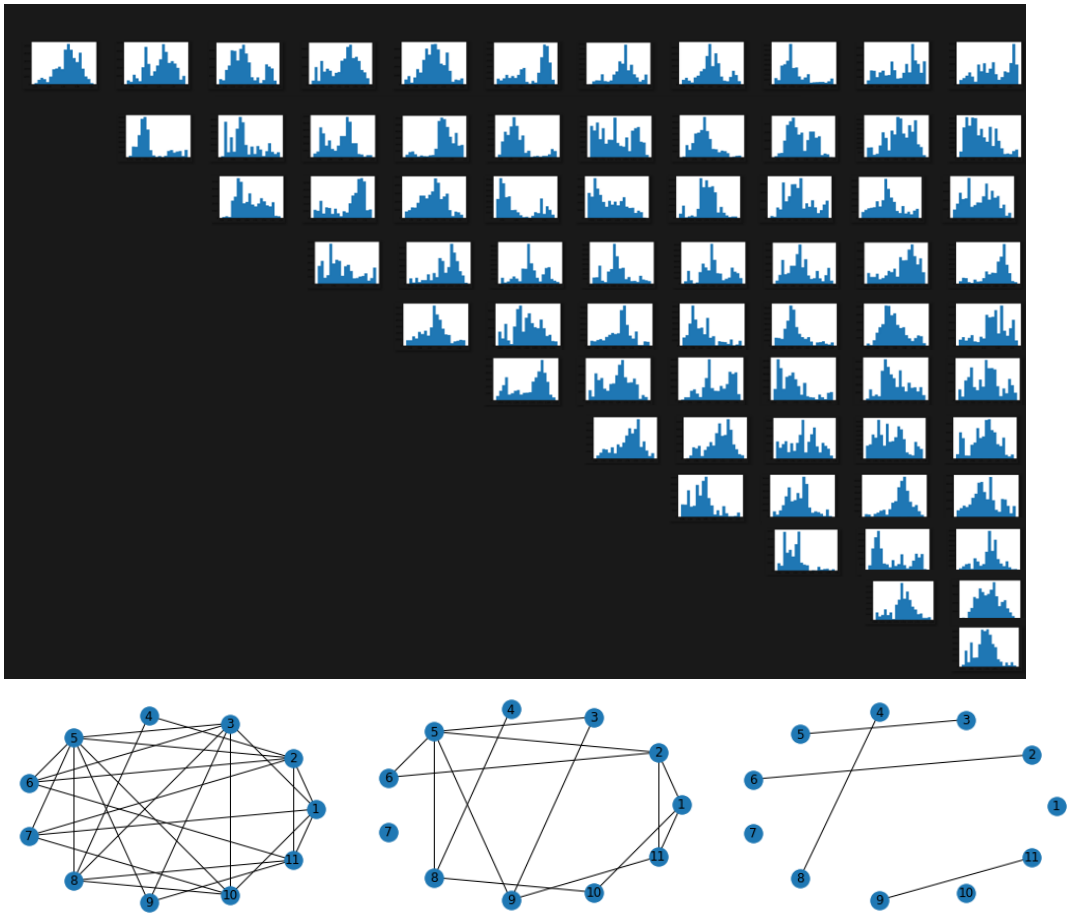


Figure B.10: The multi-panel histogram Figure B.4 and graph realisations generated at different τ in Figure B.5.

From this multi-panel histogram we can see some elements of ν_{ij},$\forall i, j = 1, ..., p, i \neq j$, have unique peaks shown in the histogram. While some histograms of elements, do represent a shape of multi-modally distributed. It would be reasonable to apply different types of priors, apart from the Uniform prior, to learn the variance unknown - we are allowed to update the prior when we have more information about the unknowns, in Bayesian inference.

B.4.1 Gaussian Prior

Firstly we apply Normal prior to learn the standard deviation parameters ν_{ij}, apart from the Uniform prior - a Normal prior with distinct parameters has been applied to each i, j pairs of standard deviation parameters ν_{ij}, $\forall i, j = 1, ..., p, i \neq j$ - namely, based on the pilot MCMC we implemented previously (shown in Figure B.3) in which we used Uniform prior on all standard deviation parameters, we have yield the distribution of standard deviation parameter ν_{ij}, and now we choose the (largest) mode of these distributions of ν_{ij} in the pilot chain, to be the mean of the Normal prior in our updated MCMC chain. And the prior standard deviation for each element of ν_{ij} has been chosen to be 0.05 times this chosen mean. In other word, we apply Normal prior $\mathcal{N}(\mu_{ij}, 0.05 \times \mu_{ij})$

For the updated MCMC with 50,000 iterations, using the Normal prior we defined above, the trace of ν_{13} in this new chain is shown below in Figure B.11; the chain converges more quickly. We also plot the distributions of ν_{ij}, $\forall i, j = 1, ..., p, i \neq j$ in Figure B.12.

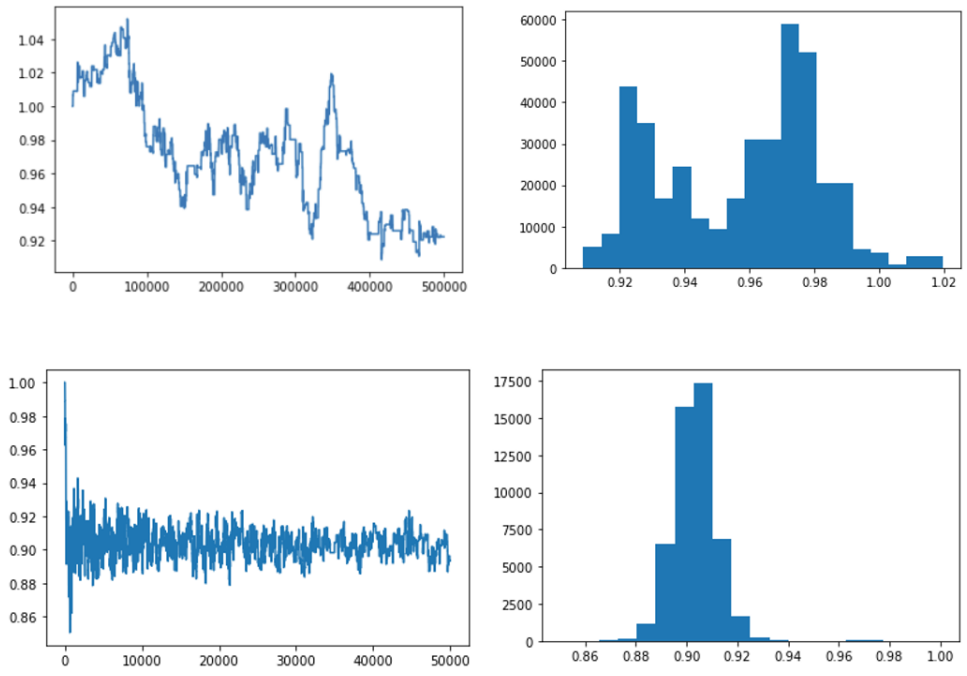


Figure B.11: The traces and histograms for ν_{24}, for uniform prior run (upper) and normal prior run(bottom). The trace for the normal prior chain converges better with fewer iterations.

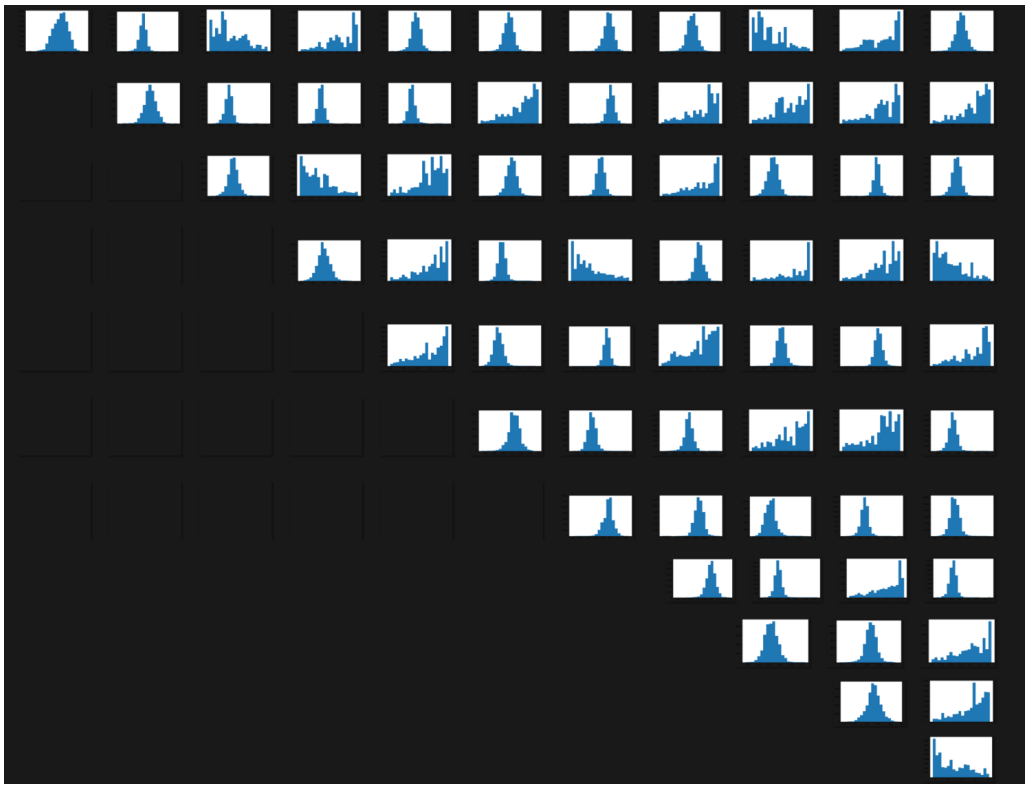


Figure B.12: The multipanel histogram according to the 50k-iteration MCMC chain using normal prior, with jump scale = 0.005, post-burnin $N_b = 100$

It is clear that compared to the multi-panel histogram in Figure B.4 in which we applied a uniform prior to the standard variance parameter ν_{ij}, in the updated Normal prior approach, most of the histograms for each ij pairs ot ν_{ij} have relatively well-converged to a bell shape. While there are still some of the ν elements that seem truncated from one side -generally, normal prior works better than uniform prior, in terms of learning the variance parameter ν, since more prior information has now been included in our MCMC.

Again we also generated graph realisation of $\mathcal{G}_{\boldsymbol{V},m}(\boldsymbol{\Psi}, \tau)$ at different chosen τ ($\tau = 0.4, 0.6$ and 0.9 respectively).

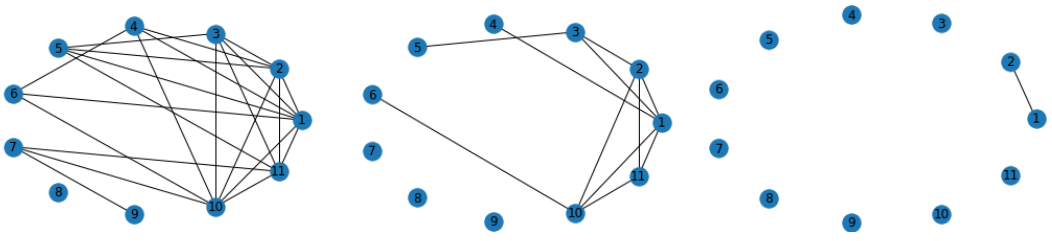


Figure B.13: Graphs generated by learning $\mathcal{G}_{\boldsymbol{V},m}(\boldsymbol{\Psi}, \tau)$ at cut-off $\tau = 0.4, 0.6$ and 0.9 respectively, from left to right.

B.4.2 Wishart Prior

We keep making our prior of individual ν_{ij} to be stronger, apart from the Normal distribution, a Wishart Distribution has been chosen, which is often used to estimate the covariance matrices in multivariate statistics. Note that in Bayesian statistics, the Wishart distribution is the conjugate prior to the inverse covariance matrix of a multivariate-normal random vector.

Definition B.4.1. *Let* $\mathbf{X}$ *be a* $p \times p$ *symmetric matrix of random variables that is positive semi-definite. Let* $\mathbf{V}$ *be a fixed symmetric positive definite matrix of size* $p \times p$*. Then, if* $n \geq p$*,* $\mathbf{X}$ *has a Wishart distribution with* n *degrees of freedom if it has the probability density function:*

$$f_{\mathbf{X}}(\mathbf{X}) = \frac{1}{2^{np/2}|\mathbf{V}|^{n/2}\Gamma_p(\frac{n}{2})}|\mathbf{X}|^{(n-p-1)/2}e^{-\frac{1}{2}tr(\mathbf{V}^{-1}\mathbf{X})}$$

For the Wishart prior we design for the variance parameter ν, we again use previous normal prior chain to update our prior choice - choosing the biggest mode which indicates in each histogram, to be our Wishart prior mean for each individual ν_{ij}, $\forall i, j = 1, ..., p, i \neq j$. And we use the coverage of that mode, times coefficients such as 0.05, to define each individual element of the fixed symmetric $\mathbf{V}$ matrix in the Wishart pdf. While with Wishart prior has been applied through, there are some elements of ν_{ij}, such as the trace of ν_{34} in Figure B.14 shown below, the trace does perform some levels of convergence at the beginning, while after 20,000 iterations the trace is getting stuck. This might be due to our Wishart prior design for this ν_{34} being so peaky that the chain is getting stuck at a value that is presumably close to the true value of the ν_{34}.

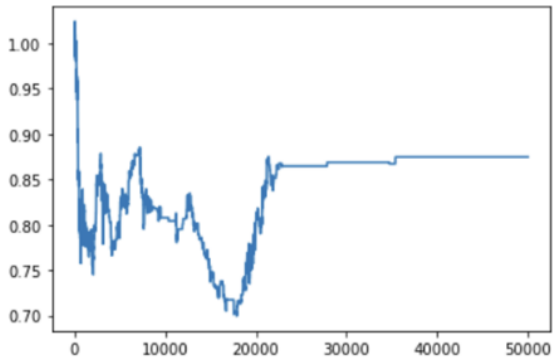


Figure B.14: Example of the trace with Wishart prior applied, ν_{23} in this case.

In light of this result with our first implementation of the Wishart prior, we decided to make our prior less peaky. The following multi-panel trace and histogram for a 500,000-iteration MCMC chain, with a chosen Wishart prior with the same mean, but the symmetric matrix $\mathbf{V}$ fixed to 3 times that used in the previous chain in Figure B.14.

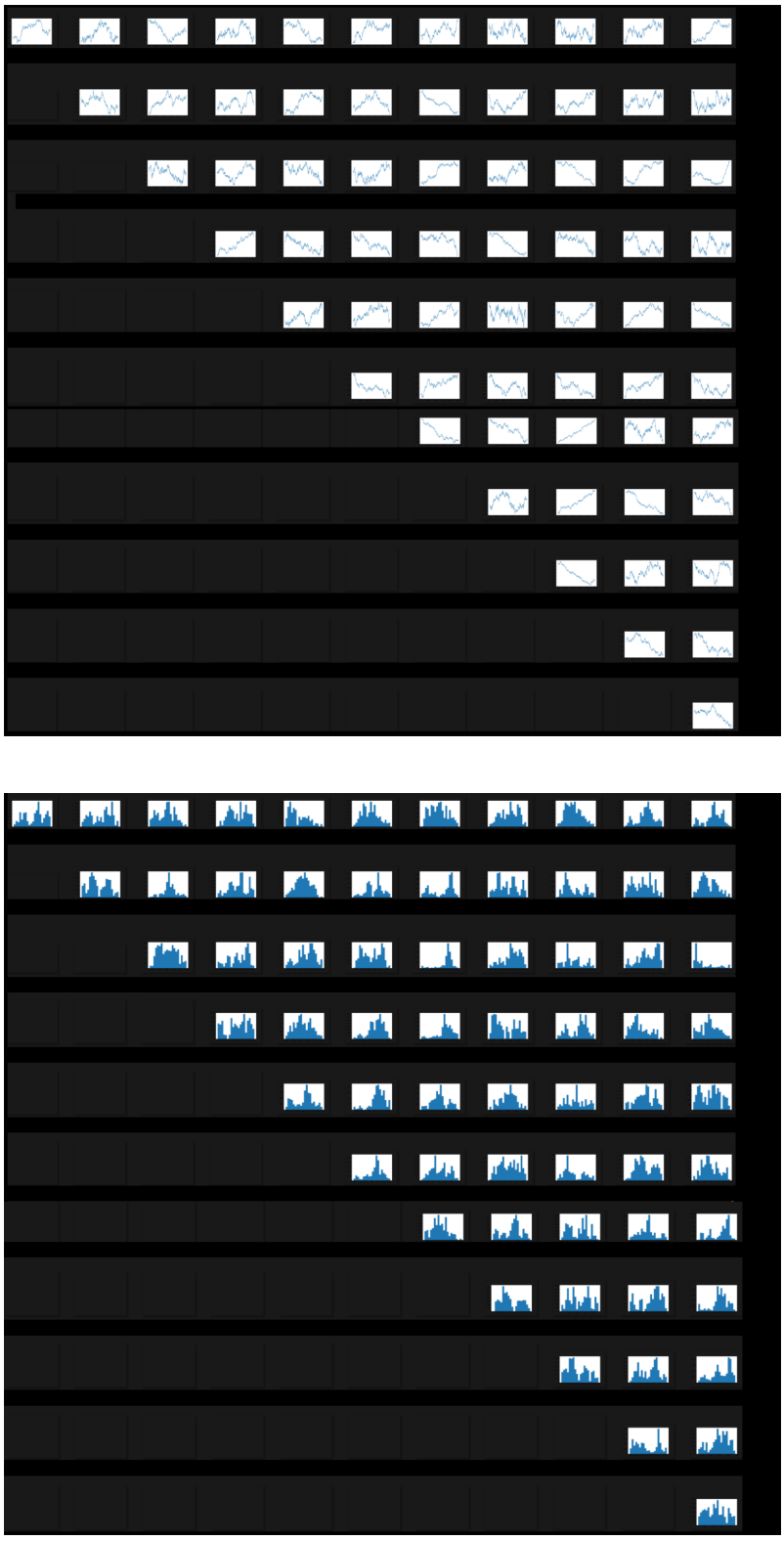


Figure B.15: The traces and distribution of ν_{ij}, $\forall i, j = 1, ..., p, i \neq j$ for the 500k-iteration MCMC chain, with jump scale = 0.0005 and post burn-in $N_b = 200000$.

And we also generate the graph results for this run, at different cutoffs.

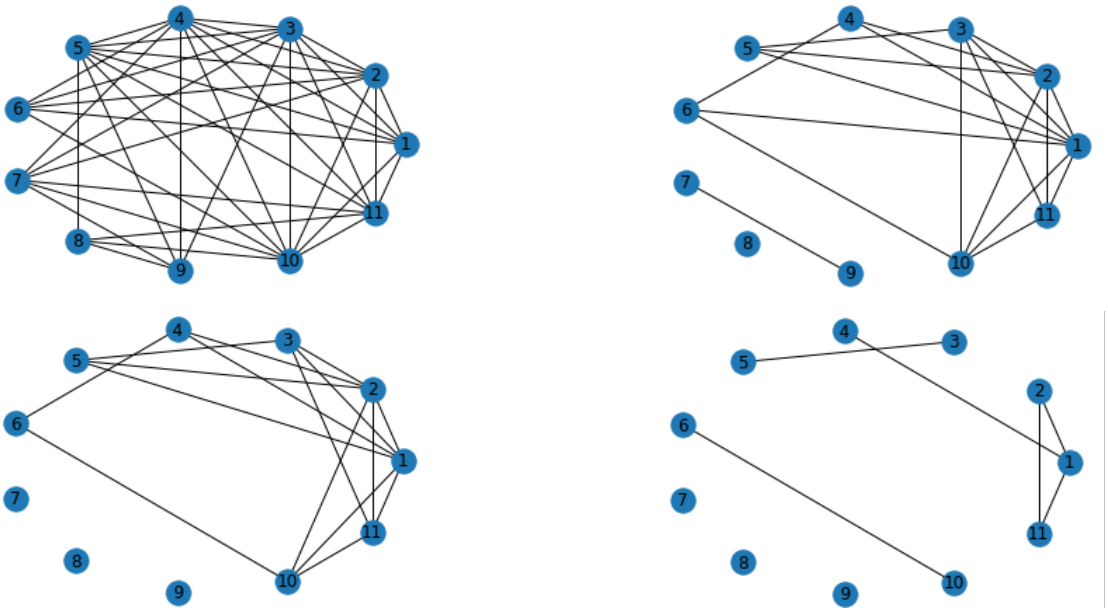


Figure B.16: From left to right and top to bottom, we have graphs realisation of $\mathcal{G}_{\boldsymbol{V},m}(\boldsymbol{\Psi},\tau)$ at different cutoff $\tau = 0.05, 0.4, 0.6$ and 0.9 respectively, generated by updated MCMC chain.

From Figure B.15 we can see that there are some elements that did not show a good convergence - either an increase or a decrease trend, and the poor convergence affects our histogram. The histograms are generally much worse than when the Gaussian prior we have in Figure B.12.

As we keep updating our Gaussian prior results, by using updated means for the Gaussian priors of the standard deviation parameters and updated new Gaussian prior variance to be such as Wishart variance as we have used in the previous run. We set up MCMC chains sequentially and keep tweaking the parameters. Although we can still see quite a few trend-displaying traces in some particular run, the graph realisations, generated by our MCMC approach with updated Gaussian prior chain, are as follows:

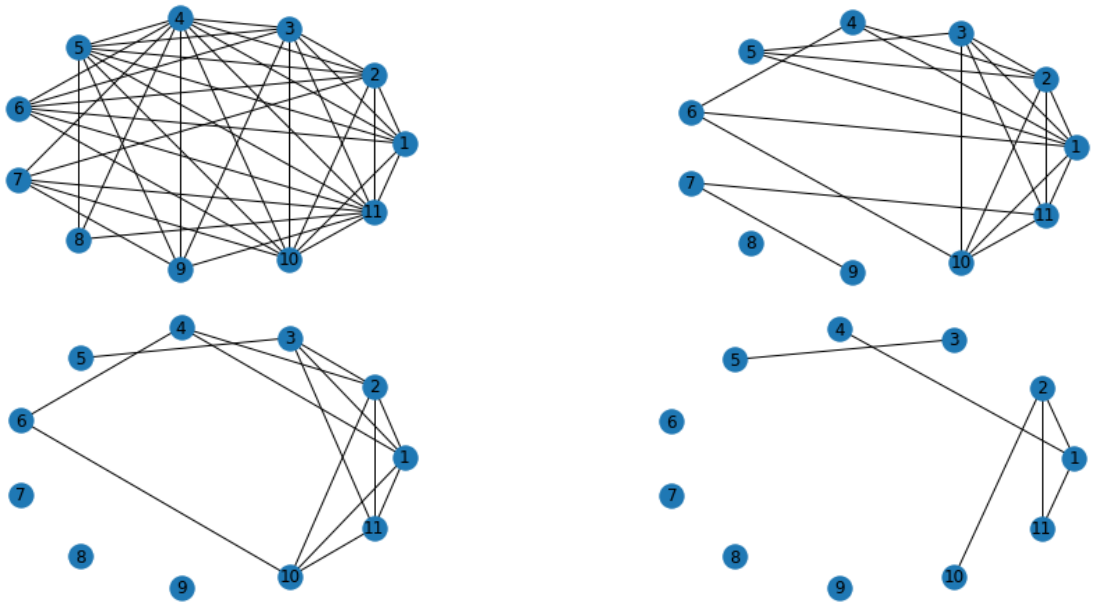


Figure B.17: From left to right and top to bottom, we have graphs realisation of $\mathcal{G}_{\boldsymbol{V},m}(\boldsymbol{\Psi},\tau)$, at different cutoffs $\tau = 0.05, 0.4, 0.6$ and 0.9 respectively, generated by one of the updated MCMC with a Gaussian prior.

Comparing the graph results in Figure B.13, Figure B.16 and Figure B.17, in general, the graph realisations represent the similarities at given τ values. It is encouraging that our graph results indicate an agreement with each other, even for the different prior choices.

Appendix C

Gaussian Process: Learning a random function

Early in my doctoral research, I have undertaken learning of a function that has been treated as a sample function drawn from a Gaussian Process (GP), where said function represents the relation between two random variables. I will now discuss such GP-based supervised learning, and present results of such learning given a real dataset.

C.1 Background

When we want to learn the relationship between a pair of variables, we are trying to learn the functional form of this relationship. Here the said variables are the input X and output variable, Y, s.t. their relationship is depicted as $Y = f(x)$, where f is an unknown function that we wish to learn. In the Bayesian approach, anything that is unknown is random. So we treat this unknown function f as a function-valued random variable, i.e. as a random function. Randomness for a Bayesian means that the particular variable takes a certain value with a certain probability, and that probability is dictated by the probability distribution of this random variable. So the random function, f, will take a certain shape with a given probability, i.e. the function will attain a value at a given value (x) with a given probability, s.t. $f(x)$ is a random variable. This probability is given by the probability distribution of this random function f.

Definition C.1.1. *We aim to learn the unknown function*

$$f : \mathcal{X} \to \mathbb{R}, \quad Y = f(X),$$

where X denotes a random variable (r.v.) taking values $x \in \mathcal{X}$, and Y is another r.v. , using a training set

$$\mathcal{D} = \left\{(x_i, y_i)\right\}_{i=1}^{N},$$

where each observation y_i is the realized value of Y_i at $X = x_i$.

Since f is unknown, we treat it as a random function and place a probability distribution over the space of all candidate functions. Such a distribution is provided by a *Stochastic Process* (Dudley, 1989; Doob, 1990).

Remark C.1.1. *A sample path or sample function of a stochastic process is a particular realisation of the process (Lamperti, 1977; Rogers and Williams, 2000; Florescu, 2014), i.e. a sample function comprises one set of variables $X(t)$ for all $t \in \mathcal{T}$, (where the deterministic variable T might be discrete or continuous).*

In my learning of the function $f(\cdot)$, I choose to model the random function $f(\cdot)$ as a sample function of a *Gaussian process* (GP) (Dudley, 1989; Rasmussen and Williams, 2006). A GP offers minimal constraints on its sample functions and it is easy to compute. The only constraint is that any finite subset of random variables - that are realisations of a sampled function at chosen inputs - is jointly, a multivariate Normal. Thus, the collection of random variables $Y_1, \ldots, Y_N$ that are outputs of $f(\cdot)$ - modelled as a sample function of the GP - realised at the inputs $x_1, \ldots, x_N$, are jointly distributed as a multivariate Normal distribution. Here, $x_1, \ldots, x_N$ are the design points in the training set that is available. So $Y_1, \ldots, Y_N$ are jointly multivariate Normal.

Definition C.1.2. *For the function $f(\cdot) \sim GP(\mu(\cdot), K(\cdot,\cdot))$, the joint pdf of N realisations $Y_1, \ldots, Y_N$ of $f(\cdot)$, is the multivariate Normal density with parameters that are the mean vector $\boldsymbol{\mu}$ and the covariance matrix $\boldsymbol{\Sigma}$.*

Here, the mean vector and covariance matrix are induced upon by the mean function $\mu(\cdot)$ and covariance function $K(\cdot,\cdot)$ of the GP. Thus, $\boldsymbol{\Sigma} = [Cov(Y_i, Y_j)]$. Then for $\boldsymbol{Y} := (Y_1, \ldots, Y_N)^T$, the joint $f_{\boldsymbol{Y}}(\boldsymbol{y})$ of the N realisations $Y_1, \ldots, Y_N$ of $f(\cdot)$ is the density:

$$f_{\boldsymbol{Y}}(\boldsymbol{y}) = \frac{\exp\left(-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{y}-\boldsymbol{\mu})\right)}{\sqrt{(2\pi)^N |\boldsymbol{\Sigma}|}}$$

I will learn the model parameters of this multivariate Normal density. This is the likelihood, i.e. the probability of the observables given the data. Then I will invoke priors on the model parameters and compute the posterior probability density of the model parameters given the training data $\{(x_i, y_i)\}_{i=1}^N$. I will undertake MCMC to perform Bayesian inference on the model parameters given this data.

I will start by standardising the data on the output variables, using the sample mean and sample standard deviation. Then the mean vector $\boldsymbol{\mu}$ can be considered as a null vector. Standardising also renders the covariance matrix $\boldsymbol{\Sigma}$ the correlation matrix, s.t. $\boldsymbol{\Sigma} = [corr(Y_i, Y_j)]$.

Definition C.1.3. *We standardise the outputs variables $Y_1, \ldots, Y_N$ which have values $\{y_i\}_{i=1}^N$ s.t.:*

$$\tilde{y}_i = \frac{y_i - \bar{y}}{s_y}, \qquad \bar{y} = \frac{1}{N}\sum_{i=1}^N y_i, \quad s_y = \sqrt{\frac{1}{N-1}\sum_{i=1}^N (y_i - \bar{y})^2}.$$

Having zero-mean, unit-variance data allows us to set the GP prior mean vector to

$$\boldsymbol{\mu} = \mathbf{0},$$

and interpret the covariance matrix $\boldsymbol{\Sigma}$ *as the correlation matrix with entries*

$$\boldsymbol{\Sigma} \;=\; [\mathrm{Corr}(Y_i, Y_j)], \qquad i, j \in \{1, \ldots, N\}.$$

Now we want to know how to populate this $\boldsymbol{\Sigma}$. For a non-diagonal element, the correlation between y_i and y_j, we can learn this value directly from MCMC. However, if we have a sample size $N = 20$, which is a small dataset, we have 20 diagonal elements but 190 distinct off-diagonal elements - it is clear that learning all such 190 parameter values using MCMC is infeasible, i.e. the proposed method of learning seems impossible, even for a small dataset.

Consider that we are at the input $X = x_i$ with the random variable $Y_i = f(x_i)$, and then we move from x_i to x_j where the realised output is the variable $Y_j = f(x_j)$. Then the distance between the i-th and j-th input values will decide how the correlation between these outputs changes. We parametrise the correlation between Y_i and Y_j as a function of x_i and x_j - to be precise, as the distance between x_i and x_j. The farther the distance, the less the correlation, i.e. the correlation between Y_i and Y_j is a declining function of the difference between x_i and x_j.

Definition C.1.4. *$corr(Y_i, Y_j)$ is parametrised as a function $K(\cdot, \cdot)$ of x_i and x_j's. K is called a correlation kernel, which is a declining function of the distance between x_i and x_j in a simple, i.e. stationary model of the GP.*

I have worked only with a kernel that takes the Euclidean distance between x_i and x_j as its input, i.e. with a stationary kernel.

I then had to determine a length scale parameter to learn how quickly the correlation between Y_i and Y_j decreases with the distance between x_i and x_j.

Definition C.1.5. *We can model these correlations via a stationary kernel function K with a single length-scale hyperparameter ℓ:*

$$\boldsymbol{\Sigma} \;=\; [K\big(|x_i - x_j|;\, \ell\big)].$$

The Squared exponential kernel (SQE), uses a Gaussian-like form with the distance between x_i and x_j as its input. It is one type of covariance kernel which uses the Euclidean distance between x_i and x_j as its input. Notice that in general, the distance between the differences between x_i and x_j does not necessarily have to be chosen as Euclidean distance.

SQE is defined as:

$$K(|x_i - x_j|; \ell) = \exp\Big(-\tfrac{1}{2}\,|x_i - x_j|^2/\ell^2\Big)$$

so that when $|x_i - x_j| = \ell$, the correlation decays to $e^{-1/2}$.

Definition C.1.6. *Likelihood is defined on the model parameter ℓ, in the available data:*

$$p(\boldsymbol{y} \mid \ell) = \frac{1}{(2\pi)^{N/2}\,|\boldsymbol{\Sigma}(\ell)|^{1/2}} \exp\!\left(-\tfrac{1}{2}\,\boldsymbol{y}^{\top}\boldsymbol{\Sigma}(\ell)^{-1}\,\boldsymbol{y}\right).$$

To complete the Bayesian model, I will typically place a Normal prior on ℓ s.t.

$$\ell \;\sim\; \mathcal{N}(\mu_\ell, \sigma_\ell^2).$$

and will perform posterior with MCMC: at iteration t, propose ℓ^* (e.g. from a Normal proposal density), compute the Metropolis–Hastings acceptance ratio

$$\alpha \;=\; \frac{p(\boldsymbol{y} \mid \ell^*)\,p(\ell^*)}{p(\boldsymbol{y} \mid \ell^{(t-1)})\,p(\ell^{(t-1)})}.$$

I will accept ℓ^* with probability $min(1, \alpha)$. After I spot the convergence of the MCMC trace, I will then summarise ℓ by its posterior mean $\hat{\ell}$ within the 95% highest posterior density interval, to predict the expectation and variance of Y_* at $X = x_*$. Rasmussen and Williams (2006) presents such a closed-form mean and variance of the multivariate Normal predictive density: for a new test input x_*, the predictive distribution of the output y_* is given as:

$$\mu_* \;=\; k_*^{\top}\boldsymbol{\Sigma}(\hat{\ell})^{-1}\,\boldsymbol{y}, \qquad \sigma_*^2 \;=\; k_{**} - k_*^{\top}\boldsymbol{\Sigma}(\hat{\ell})^{-1}k_*,$$

where

$$k_* = \left[K(|x_* - x_1|; \hat{\ell}), \ldots, K(|x_* - x_N|; \hat{\ell})\right]^{\top}, \quad k_{**} = K(|x_* - x_*|; \hat{\ell}) = K(0; \hat{\ell}).$$

Rescaling back by $\bar{y}$ and s_y yields the final predicted mean and variance for Y_*.

C.2 Discussion: singularity in $\boldsymbol{\Sigma}$

Definition C.2.1. *Any $N \times N$-dimensional matrix $\mathbf{C} = [C_{ij}]$ is positive semi-definite if $\sum_{i=1}^{N}\sum_{j=1}^{N} a_i a_j C_{ij} \geq 0$, where $i, j \in \{1, 2, \ldots, N\}$, $\forall a_i, a_j \in \mathbb{R}$.*

Now, variance of the weighted sum of the N random variables $X_1, \ldots, X_N$ is always ≥ 0, i.e. $var(\sum_{i=1}^{N} a_i X_i) = \sum_{i=1}^{N}\sum_{j=1}^{N} a_i a_j Cov(X_i X_j) \geq 0$, $\forall a_i, a_j \in \mathbb{R}$. Thus, the matrix $\boldsymbol{\Sigma} = [Cov(X_i, X_j)]$ is positive semi-definite, by Definition C.2.1. Indeed, $\boldsymbol{\Sigma}$ being a covariance matrix, is a positive definite symmetric matrix. When the outputs are standardised, $\boldsymbol{\Sigma} = [Cov(Y_i, Y_j)]$ reduces to $\boldsymbol{\Sigma} = [Corr(X_i, X_j)]$. Here, the standardisation is done with the sample mean and sample standard deviation derivation of the sample of values of the output that live in the training data.

To compute $\boldsymbol{\Sigma}$, I model its ij-th element as $K(x_i, x_j)$, where $K(\cdot, \cdot)$ is a declining function of the Euclidean distance between x_i and x_j, as done (for example) in the choice of an SQE kernel: $Corr(Y_i, Y_j) = K(x_i, x_j) = \exp(-(x_i - x_j)^2/2\ell^2)$. This holds $\forall i, j \in \{1, \ldots, N\}$. In my learning of the length scale hyperparameter ℓ of the SQE kernel, there may exist some proposed value of ℓ that causes $\boldsymbol{\Sigma}$ to violate

positive definiteness. Hence, when choosing ℓ, one should be careful. Indeed, a judicious seed becomes relevant, about which I need to glean some information using the available training data. Since I have decided to choose SQE to compute my $\mathbf{\Sigma}$ matrix, it would be helpful that I examine key information, such as the average values of $(x_i - x_j)$, which goes into my numerator of the exponential term. I could then consider the order of magnitude of the parameters I want to learn, namely ℓ, chosen to be closer to the true value. Also, I undertake a trick (Rasmussen and Williams, 2006) of adding a very small number to the diagonal elements of the correlation matrix, to guard against falling prey to violating positive definiteness. (Once I make the prediction on the output realised at a test input, I will un-standardise the predicted output using the sample mean and standard deviation employed earlier to standardise the output values).

Inversion of the $\mathbf{\Sigma}$ matrix can be achieved using Cholesky decomposition (Benoît, 1924; Press et al., 1992).

Definition C.2.2. *Every symmetric, positive definite matrix* $\mathbf{A}$ *can be decomposed into a product of a unique lower triangular matrix* $\mathbf{L}$ *and its transpose:*

$$\mathbf{A} = \mathbf{L}\mathbf{L}^T$$

where $\mathbf{L}$ *is the Cholesky factor of* $\mathbf{A}$ *(Golub and Van Loan, 1996; Horn and Johnson, 1985; Trefethen and Bau, 1997; Schabauer et al., 2010), and can be interpreted as a generalized square root of* $\mathbf{A}$*, as described in Cholesky decomposition .*

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{21} & a_{31} & ... \\ a_{21} & a_{22} & a_{23} & ... \\ a_{31} & & ... & \\ ... & & ... & ... \\ a_{n1} & ... & ... & a_{nn} \end{pmatrix} = \begin{pmatrix} \ell_{11} & 0 & 0 & ... & \\ \ell_{21} & \ell_{21} & 0 & ... & \\ ... & & ... & & \\ & ... & & & \\ \ell_{n1} & \ell_{n2} & ... & ... & \ell_{nn} \end{pmatrix} \begin{pmatrix} \ell_{11} & \ell_{21} & \ell_{31} & ... & \ell_{n1} \\ 0 & \ell_{22} & \ell_{32} & ... & \\ ... & & ... & & \\ & ... & & & \\ 0 & 0 & ... & ... & \ell_{nn} \end{pmatrix}$$

Then the diagonal of $\mathbf{L}$ *is:*

$$\ell_{ii} = \sqrt{a_{ii} - \sum_{j=i}^{k-1} \ell_{kj}^2}$$

and for the elements below the diagonal, i.e. for ℓ_{ik}*,* $i > k$ *we have*

$$\ell_{ik} = \frac{1}{\ell_{kk}}(a_{ik} - \sum_{j=1}^{k-1} \ell_{ij}\ell_{kj})$$

Then for

$$\mathbf{\Sigma} = [K(x_i, x_j)],$$

using

$$\mathbf{\Sigma} = \mathbf{L}\mathbf{L}^T,$$

we get

$$(\mathbf{\Sigma})^{-1} = (\mathbf{L}\mathbf{L}^T)^{-1}$$

$$(\mathbf{\Sigma})^{-1} = (\mathbf{L}^T)^{-1}\mathbf{L}^{-1}$$

C.3 Implementation

We now apply our GP learning on the datasets:

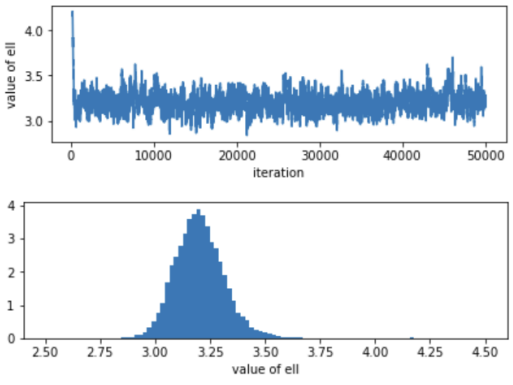
1. The first dataset is $\{(x_i, \sin(0.9x_i))\}_{x=1}^N$, where $N = 100$, with $x \in [-5, 5]$ $\forall i = 1, \ldots, N$. We refer to this dataset as a "simple dataset" below.

2. The second dataset is a real-world, publicly-available data comprising values of power consumption in Tetouan city (Salam and El Hibaoui, 2018).

C.3.1 Simple data set

We set up a training set $\{(x_i, \sin(0.9x_i))\}_{x=1}^N$, for N=100. We set a test dataset of size 50, and make predictions at the test inputs.

For the seed value of parameters $\ell^{(seed)} = 5$, and proposed jump scale $\sigma_\ell = 0.02$, we undertake 50,000 MCMC iterations:



The 95% Highest Posterior Density Regions of ell is [2.98953579 3.42919971]

Figure C.1: Figure showing results of learning the model parameter ℓ using RW and a training set of size 100. The plot of the trace of ℓ is in the top panel and the histogram that approximates the marginal posterior of ℓ given the training set, is at the bottom. This learning of ℓ has been undertaken with a prior variance $\sigma_\ell^2 = 0.02^2$. The 95% HPD can be calculated using this marginal of ℓ.

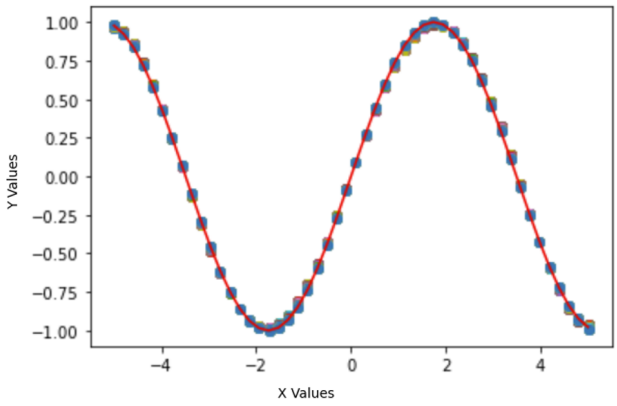


Figure C.2: The red line joins the true values of the outputs, and the blue rectangles show the predicted output at each test input, for the ℓ learnt with MCMC.

C.3.2 Real-world dataset: Tetouan city data

For an application to real-world data, I chose the dataset on Power consumption in Tetouan city (Salam and El Hibaoui, 2018). This data is relevant to the power consumption of three different power-distribution networks of Tetouan City in northern Morocco. There are 9 attributes in this dataset: Date; Time; Temperature; Humidity; Wind Speed of Tetouan city; General diffuse flows; Power consumption of zone 1; zone 2; and zone 3 of Tetouan city, respectively. As we can appreciate, there is a relationship between temperature and power consumption. Hence we could set temperature X to be our predictive variable and average power consumption of the three zones in this city, Y becomes the response variable in the model in which we regress Y against X, completely non-parametrically, i.e. the functional relationship between the input X and output Y is not assigned a pre-chosen parametric form, but is learnt.

Since every parameter in this data has been recorded every 10 minutes, we could tidy up the figures to achieve data on a daily basis, and also standardise the response variable, namely the average power consumption of three zones, Y, in order to study the hyperparameter ℓ in SQE. I used Cholesky decomposition to calculate the inversion of our covariance matrix $\mathbf{\Sigma}$. I used this publicly available dataset as my training dataset of sample size 300, and my test dataset is of size 64.

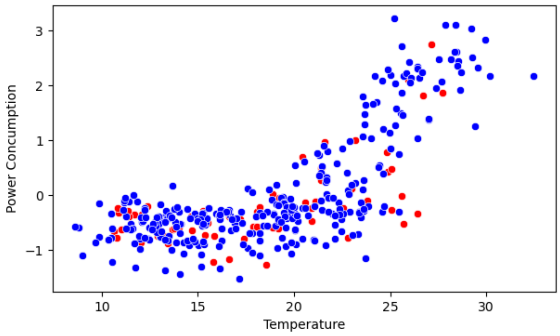


Figure C.3: Training data and test data visualisation: the (standardised) average power consumption in zones 1, 2 and 3 of Tetouan city, is plotted against temperature (in degrees Celsius).

By choosing our prior to being Normal prior, $\pi_0(\ell) \sim \mathcal{N}(20, 2^2)$, with seed values of parameters $\ell^{(seed)} = 20$, and proposed jump scale $\sigma_\ell = 0.2$, we run for the 50,000 iterations:

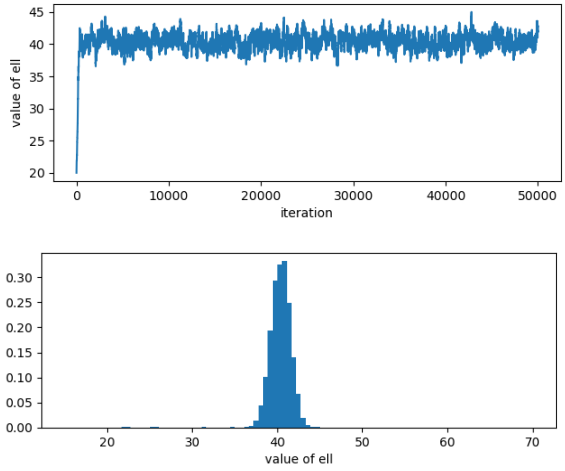


Figure C.4: Using MCMC with a Normal prior and Normal proposal, the plots show the values and histogram of learning ℓ, with $\sigma_\ell = 0.2$. The 95% HPD is (38.2089275942.68057076) according to the distribution of ℓ.

We can then see how our predictions work.

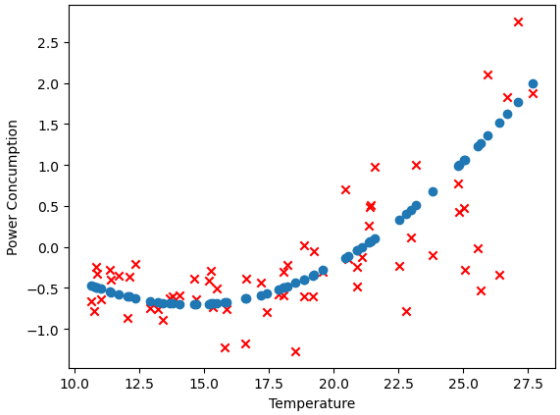


Figure C.5: The red crosses are the true values of the output at the test inputs, and the blue dots show the mean of the predicted output at each test point at all learnt ℓ in Figure C.4.

Appendix D

Recovery Trajectories of Stroke Patients Playing Exer-games

D.1 Game Airplane

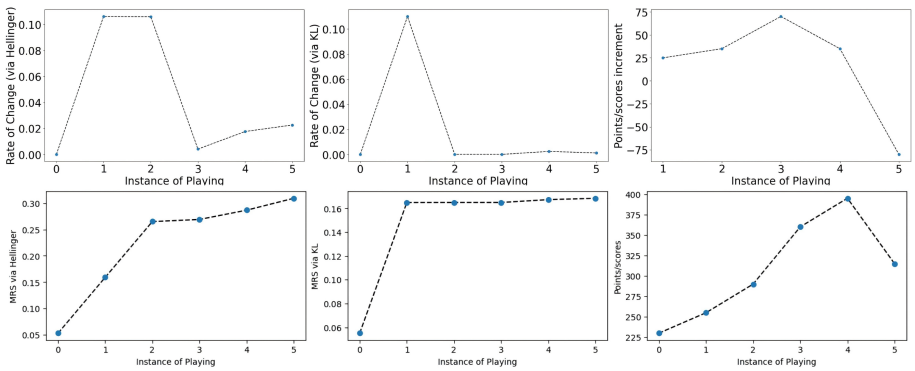


Figure D.1: The comparison of computing scores of Patient ID3071 playing Game Airplane 6 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

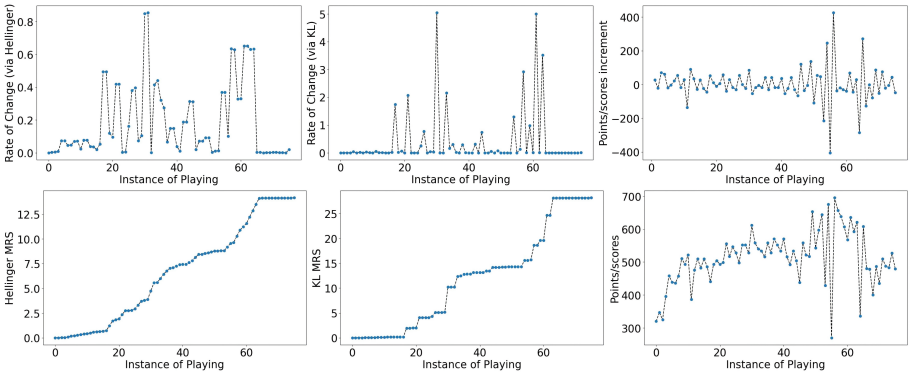


Figure D.2: The comparison of computing scores of Patient ID3085 playing Game Airplane 26 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

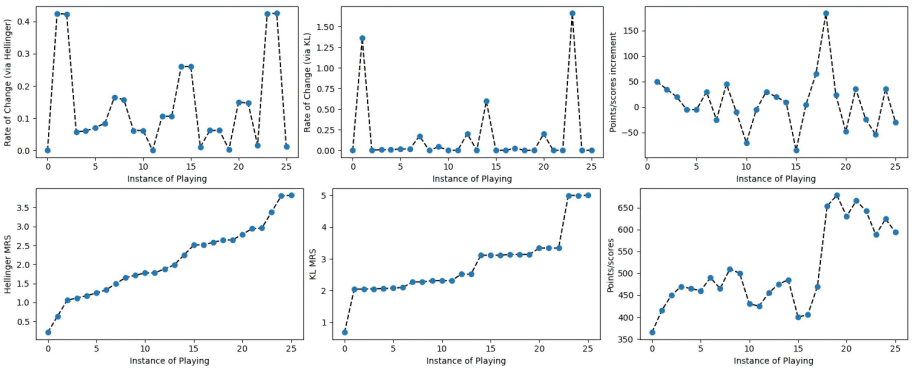


Figure D.3: The comparison of computing scores of Patient ID3147 playing Game Airplane 26 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

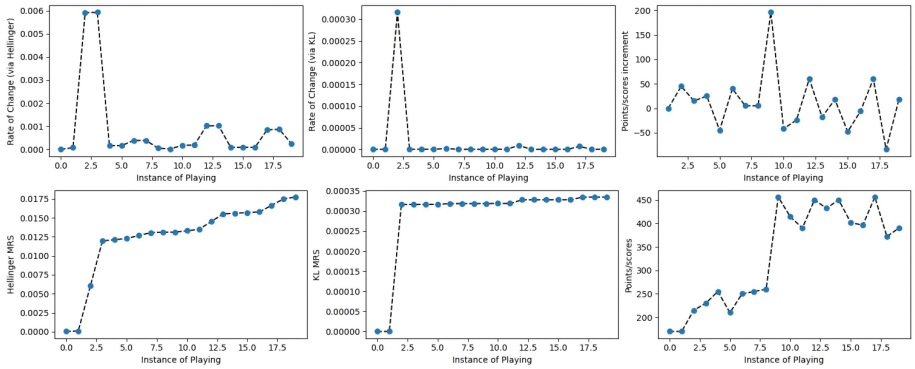


Figure D.4: The comparison of computing scores of Patient ID3327 playing Game Airplane 20 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

D.2 Game Animal

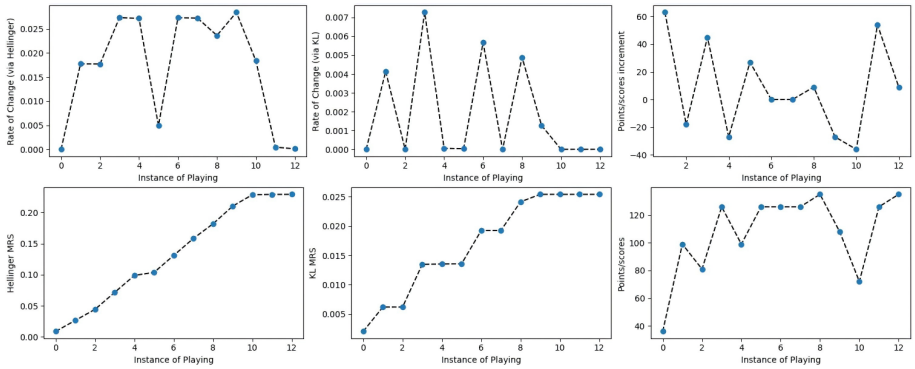


Figure D.5: The comparison of computing scores of Patient ID3071 playing Game Animal 13 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

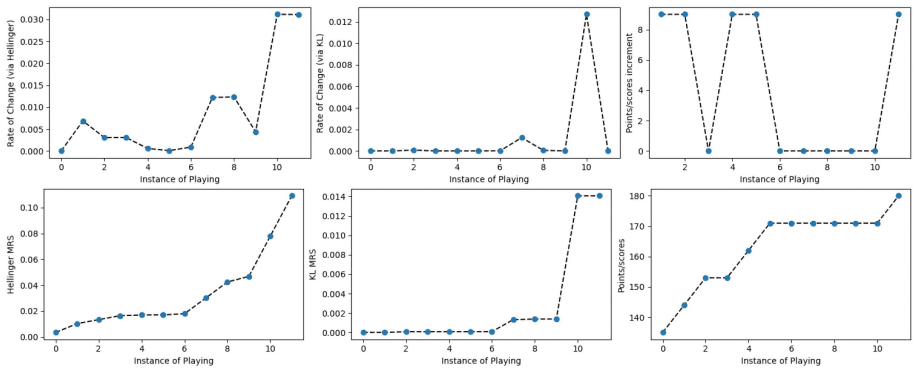


Figure D.6: The comparison of computing scores of Patient ID3085 playing Game Animal 12 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

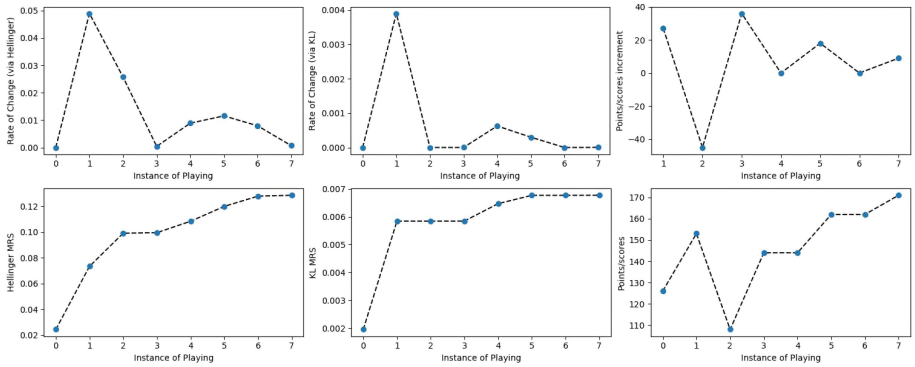


Figure D.7: The comparison of computing scores of Patient ID3147 playing Game Animal 8 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

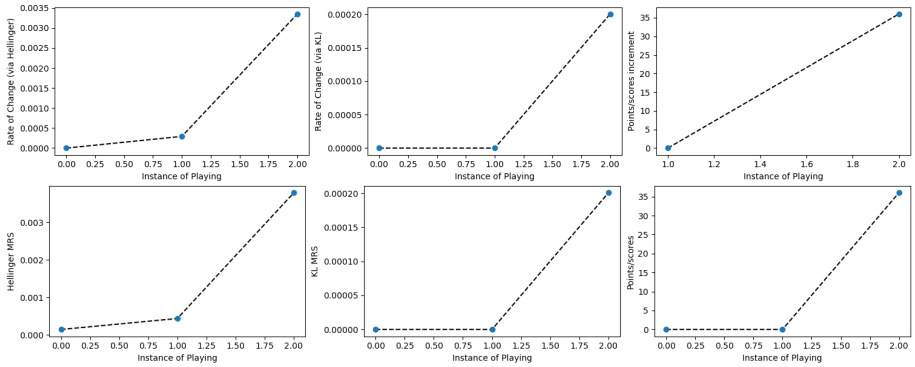


Figure D.8: The comparison of computing scores of Patient ID3165 playing Game Animal 3 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

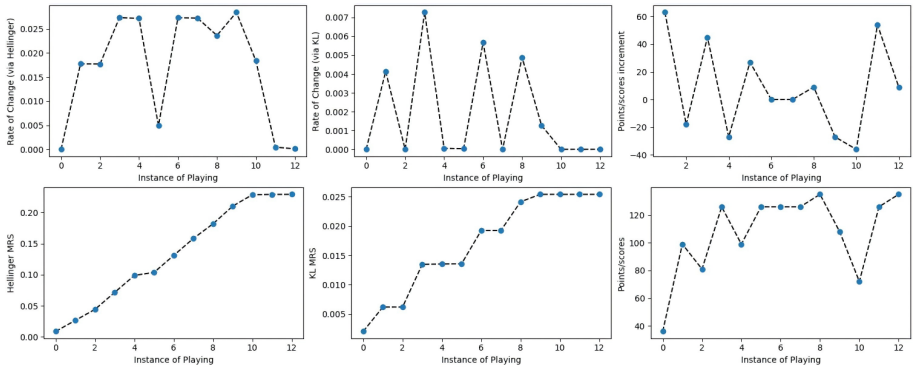


Figure D.9: The comparison of computing scores of Patient ID3462 playing Game Animal 21 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

D.3 Game Catch

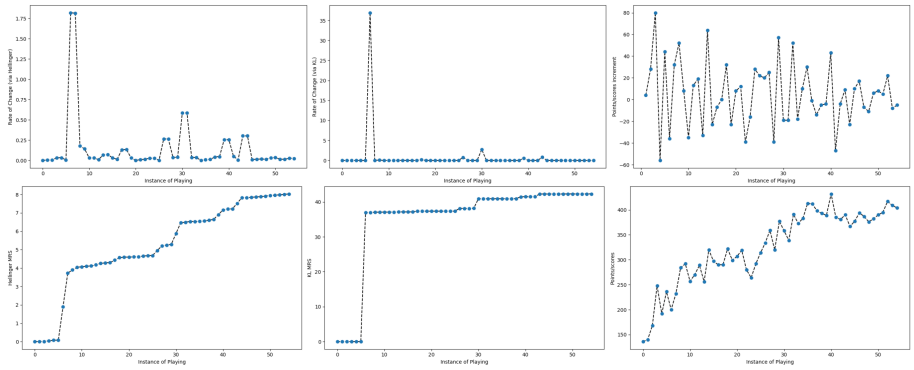


Figure D.10: The comparison of computing scores of Patient ID3070 playing Game Catch 55 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

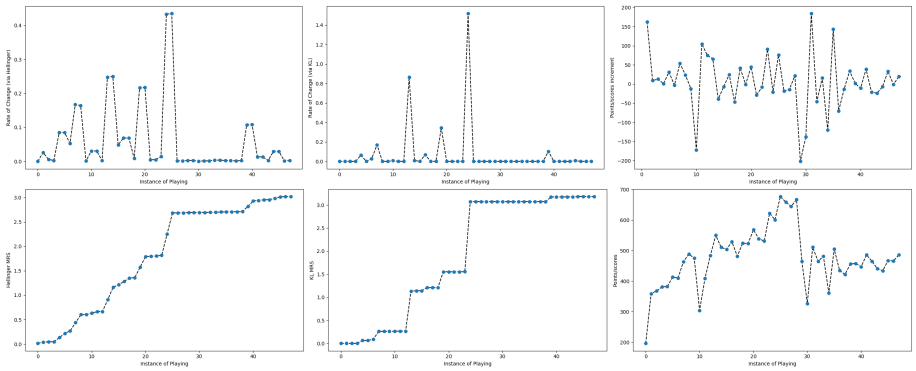


Figure D.11: The comparison of computing scores of Patient ID3071 playing Game Catch 48 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

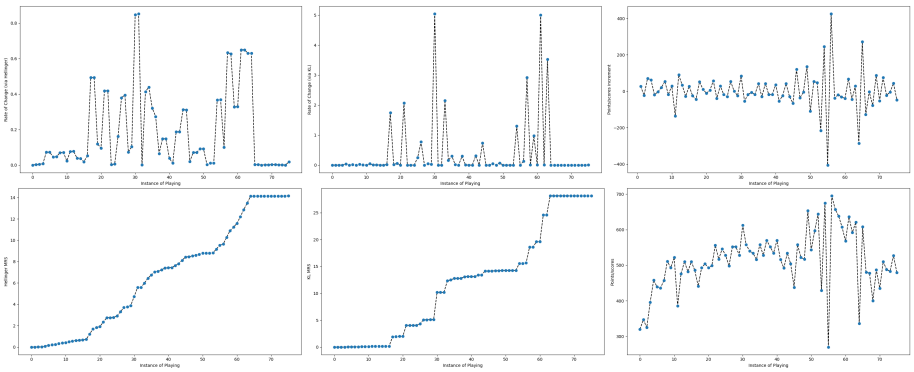


Figure D.12: The comparison of computing scores of Patient ID3085 playing Game Catch 76 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

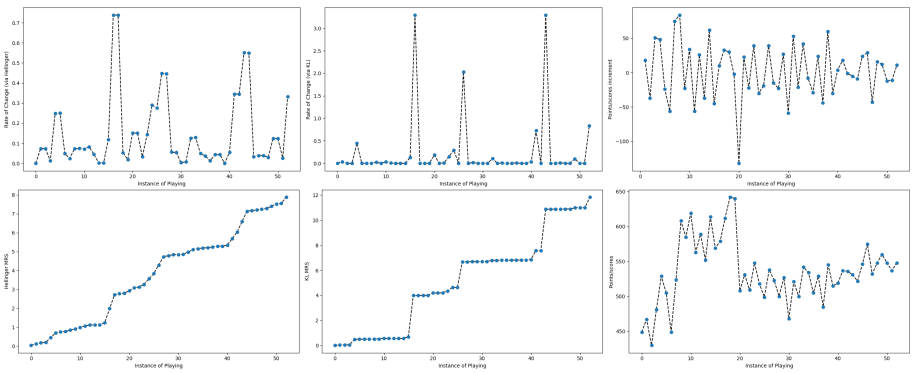


Figure D.13: The comparison of computing scores of Patient ID3147 playing Game Catch 53 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

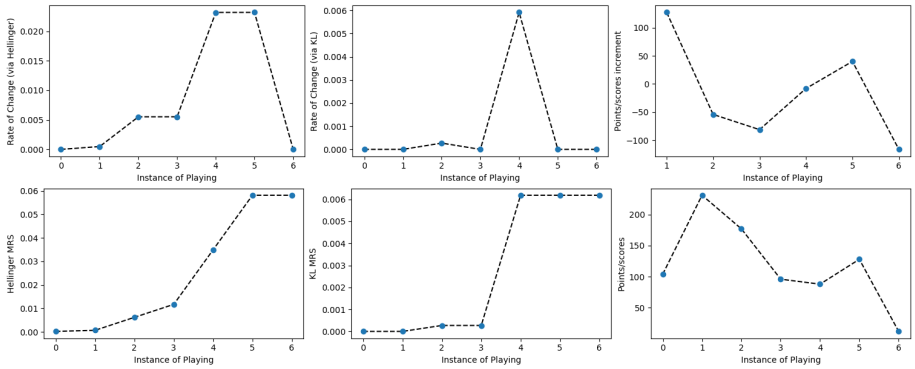


Figure D.14: The comparison of computing scores of Patient ID3165 playing Game Catch 7 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

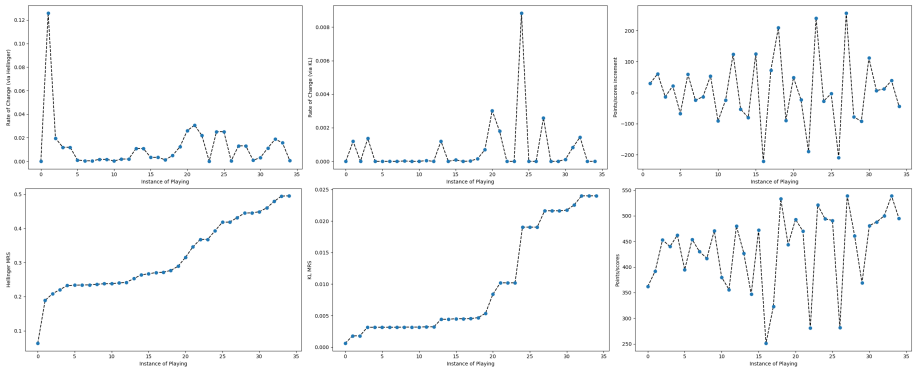


Figure D.15: The comparison of computing scores of Patient ID3311 playing Game Catch 35 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

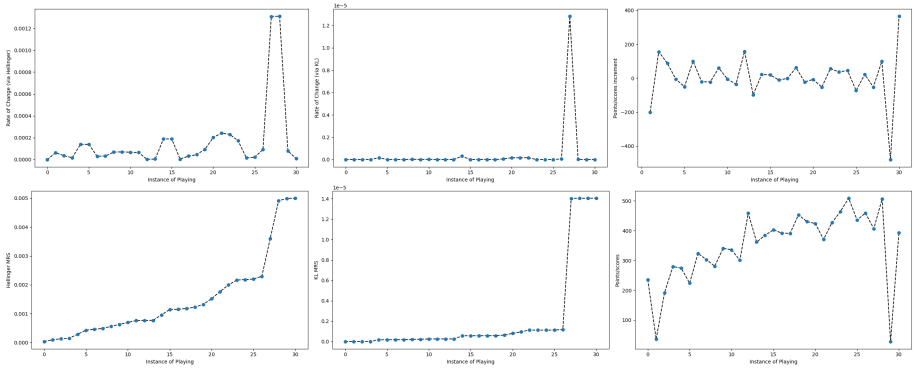


Figure D.16: The comparison of computing scores of Patient ID3327 playing Game Catch 32 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

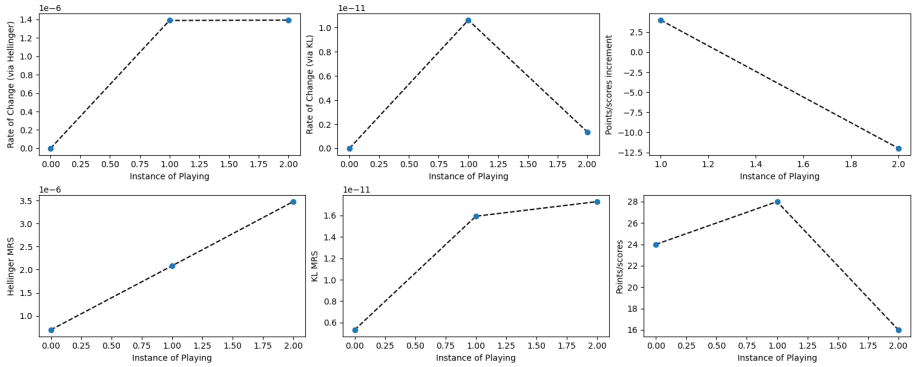


Figure D.17: The comparison of computing scores of Patient ID3379 playing Game Catch 4 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

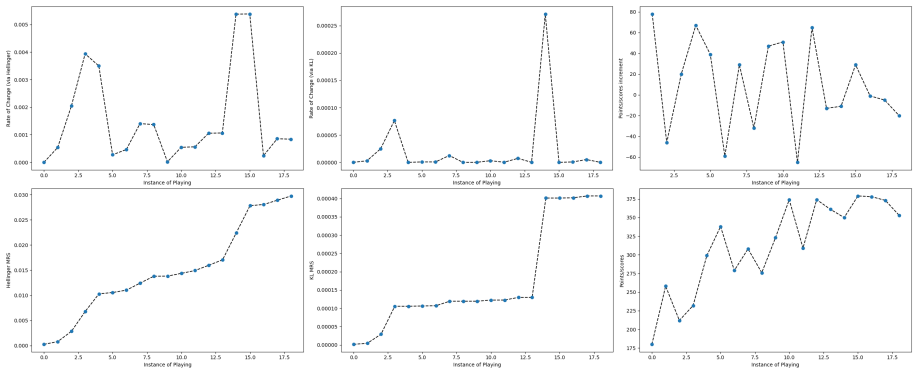


Figure D.18: The comparison of computing scores of Patient ID3430 playing Game Catch 22 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

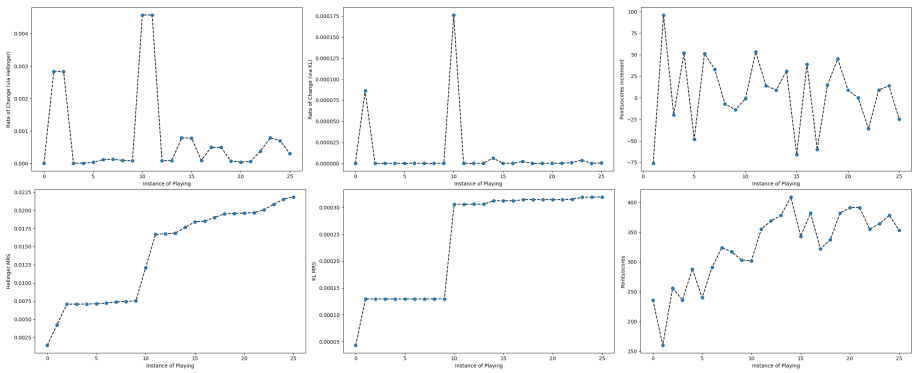


Figure D.19: The comparison of computing scores of Patient ID3431 playing Game Catch 26 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

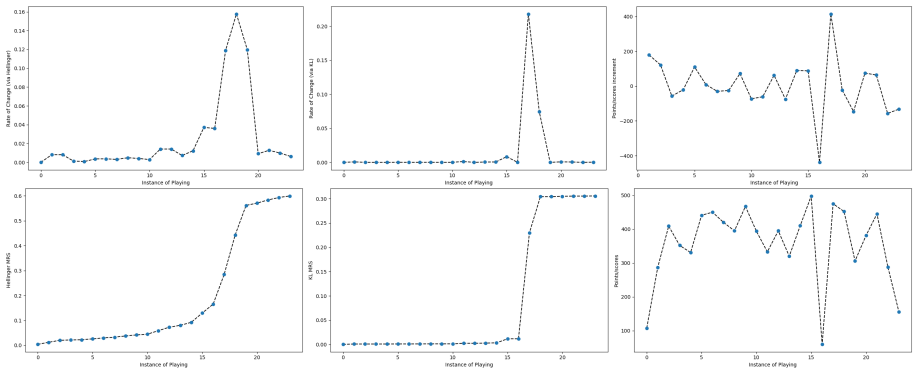


Figure D.20: The comparison of computing scores of Patient ID3462 playing Game Catch 24 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

D.4 Game Colour Block

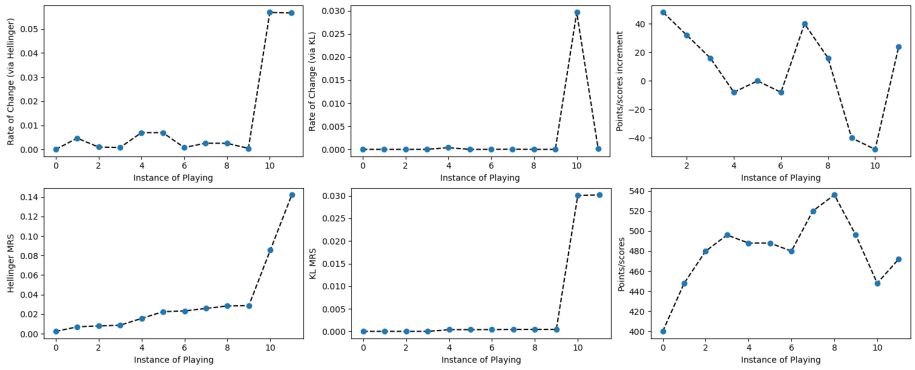


Figure D.21: The comparison of computing scores of Patient ID3085 playing Game Colour Blocks 12 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

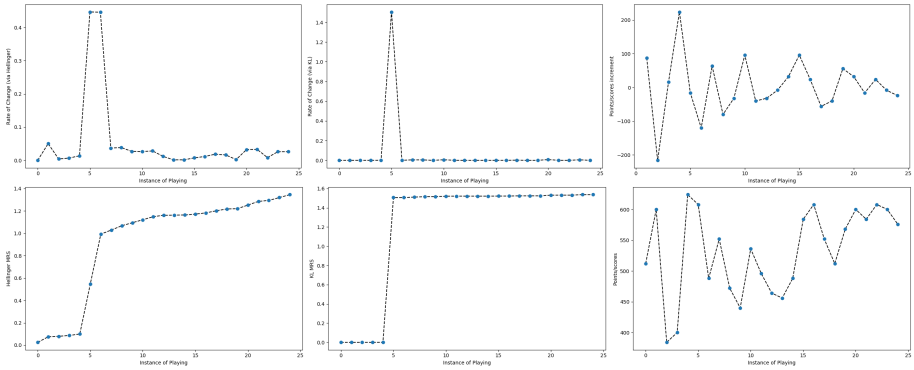


Figure D.22: The comparison of computing scores of Patient ID3147 playing Game Colour Blocks 25 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

D.5 Game Colour Clouds

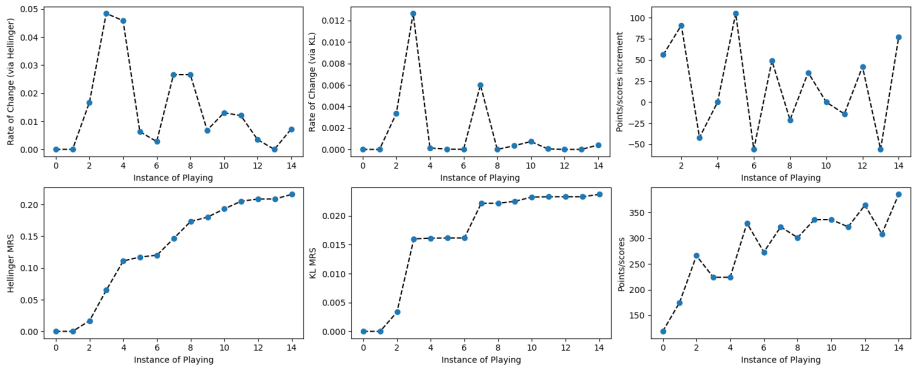


Figure D.23: The comparison of computing scores of Patient ID3085 playing Game Colour Clouds 14 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

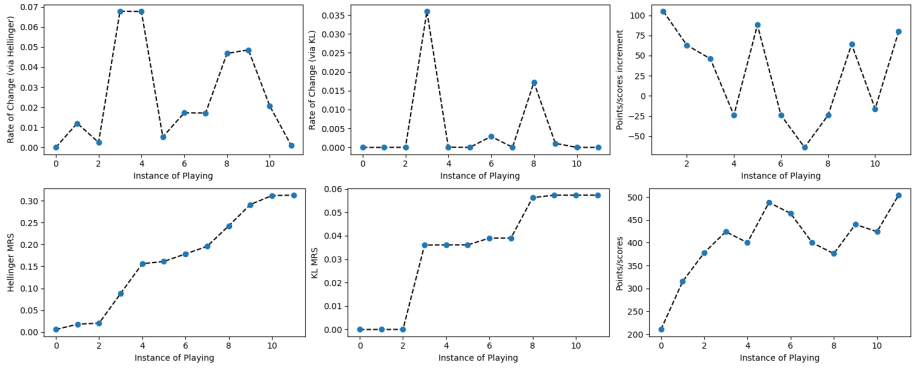


Figure D.24: The comparison of computing scores of Patient ID3147 playing Game Colour Clouds 12 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

D.6 Game Firefly

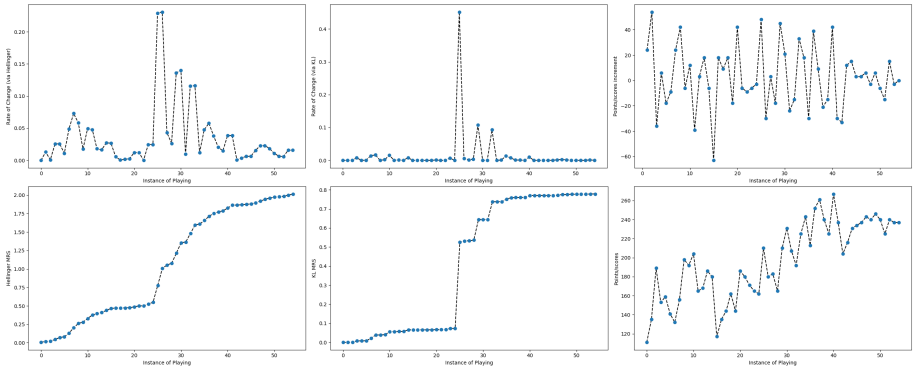


Figure D.25: The comparison of computing scores of Patient ID3070 playing Game Firefly 55 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

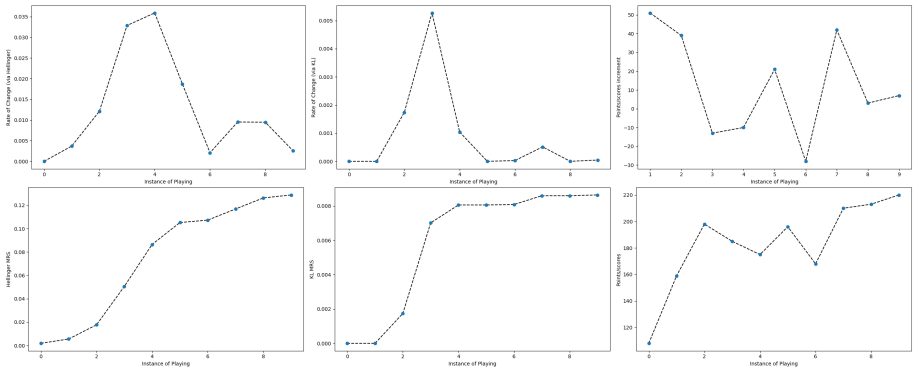


Figure D.26: The comparison of computing scores of Patient ID3071 playing Game Firefly 10 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

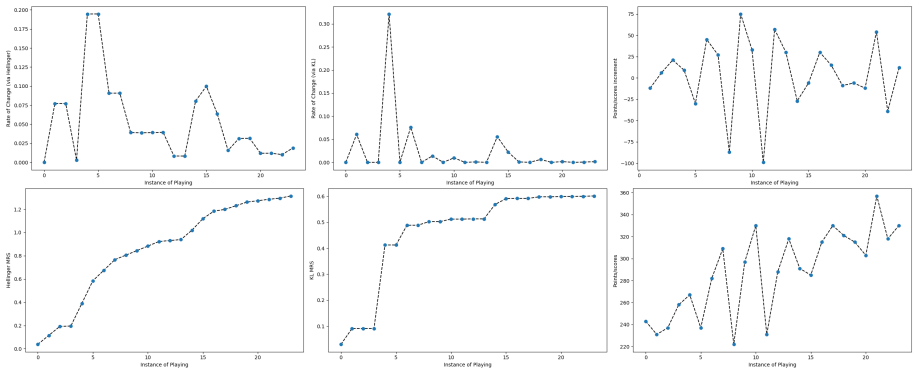


Figure D.27: The comparison of computing scores of Patient ID3085 playing Game Firefly 24 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

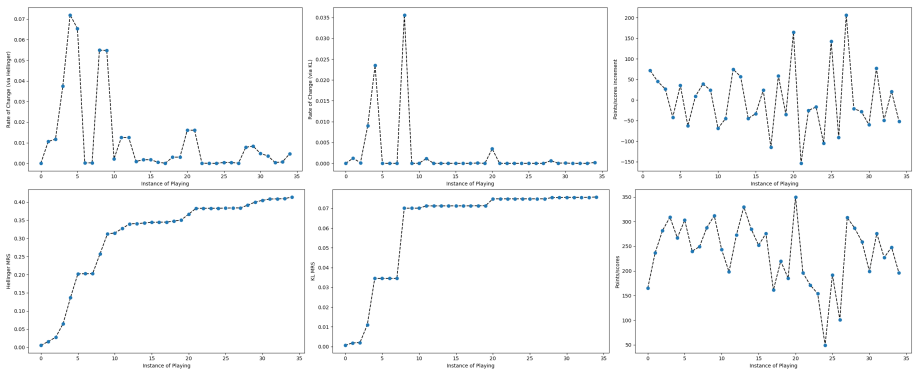


Figure D.28: The comparison of computing scores of Patient ID3311 playing Game Firefly 35 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

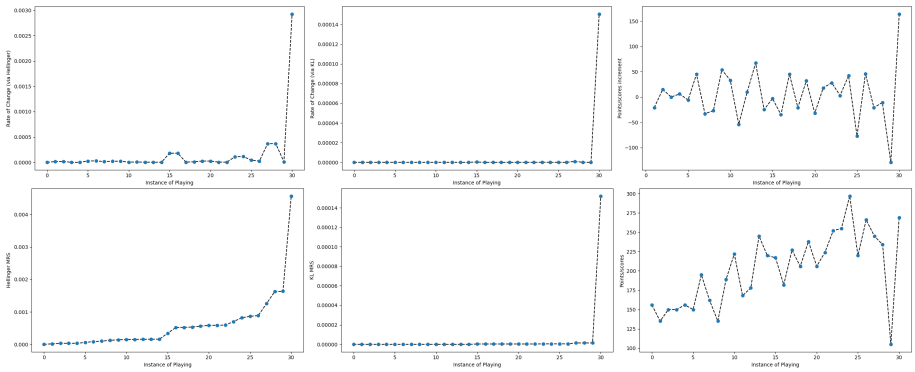


Figure D.29: The comparison of computing scores of Patient ID3327 playing Game Firefly 32 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

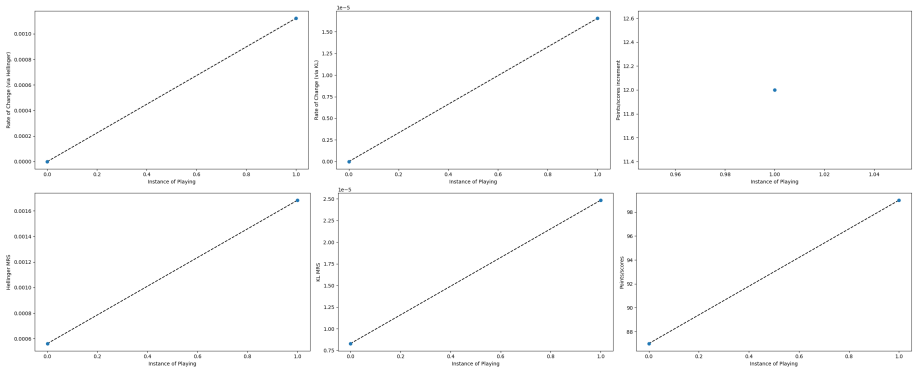


Figure D.30: The comparison of computing scores of Patient ID3379 playing Game Firefly 2 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

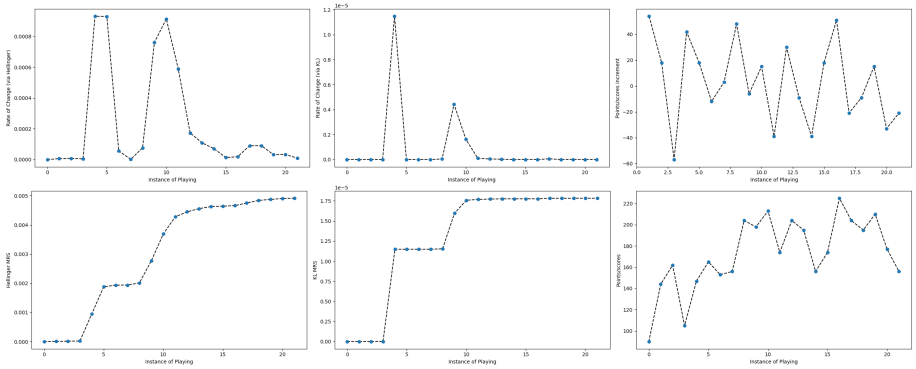


Figure D.31: The comparison of computing scores of Patient ID3430 playing Game Firefly 22 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

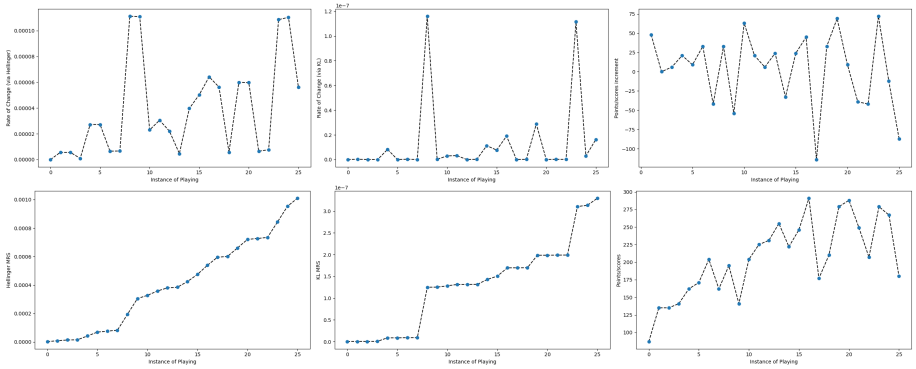


Figure D.32: The comparison of computing scores of Patient ID3431 playing Game Firefly 26 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

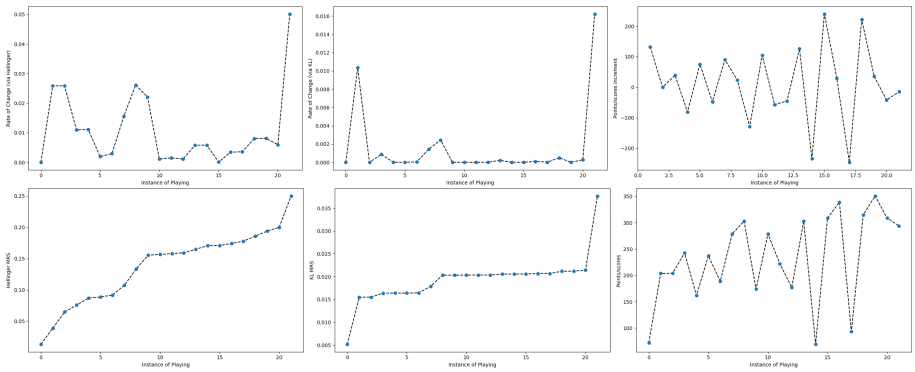


Figure D.33: The comparison of computing scores of Patient ID3462 playing Game Firefly 22 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

D.7 Game Follow

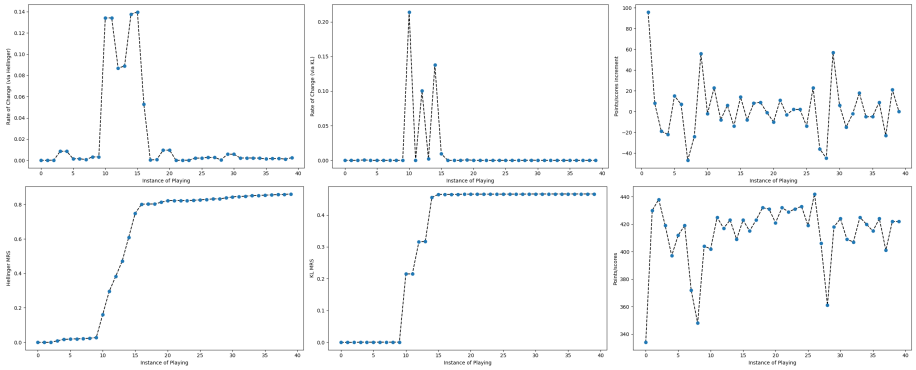


Figure D.34: The comparison of computing scores of Patient ID3070 playing Game Follow 40 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

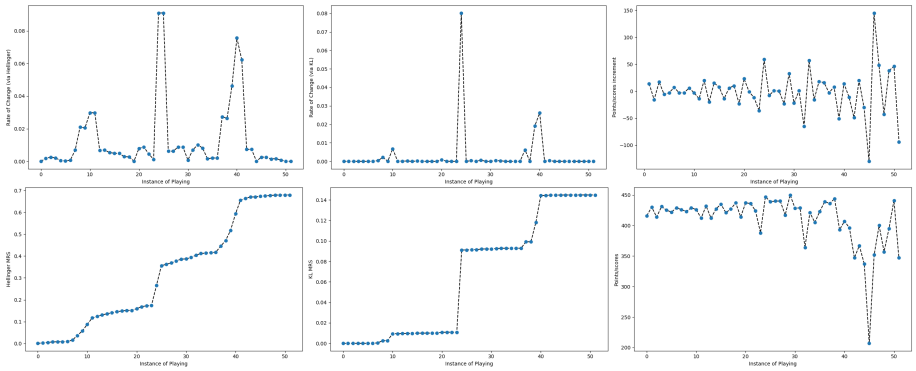


Figure D.35: The comparison of computing scores of Patient ID3085 playing Game Follow 52 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

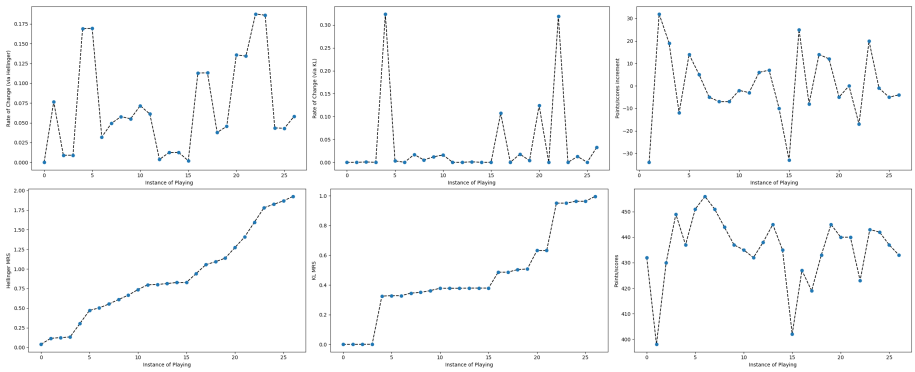


Figure D.36: The comparison of computing scores of Patient ID3147 playing Game Follow 27 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

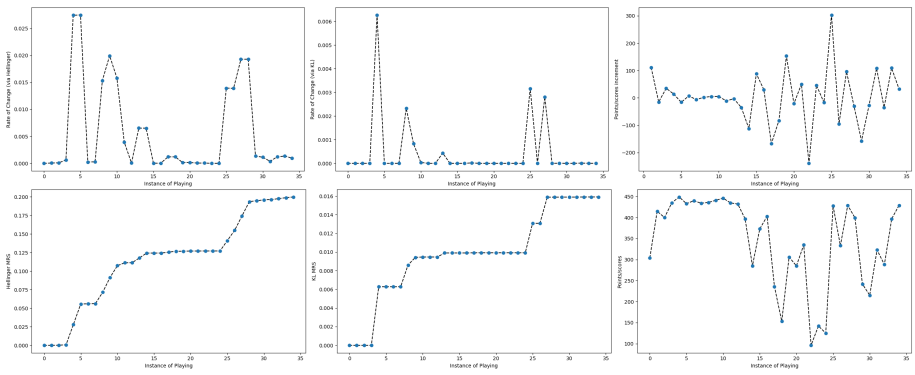


Figure D.37: The comparison of computing scores of Patient ID3311 playing Game Follow 35 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

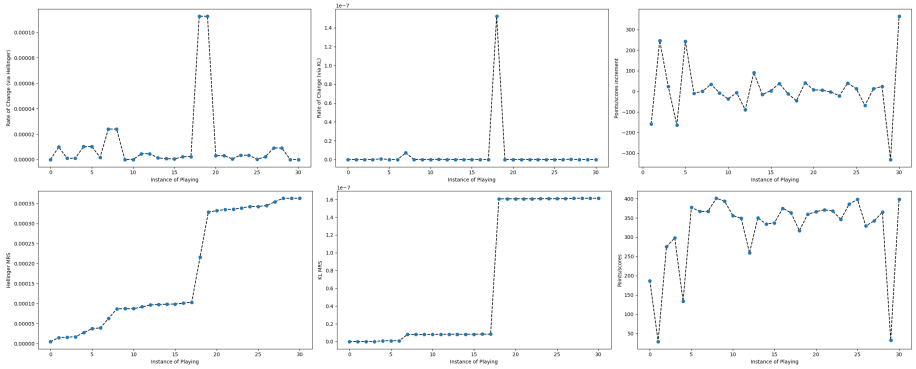


Figure D.38: The comparison of computing scores of Patient ID3327 playing Game Follow 32 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

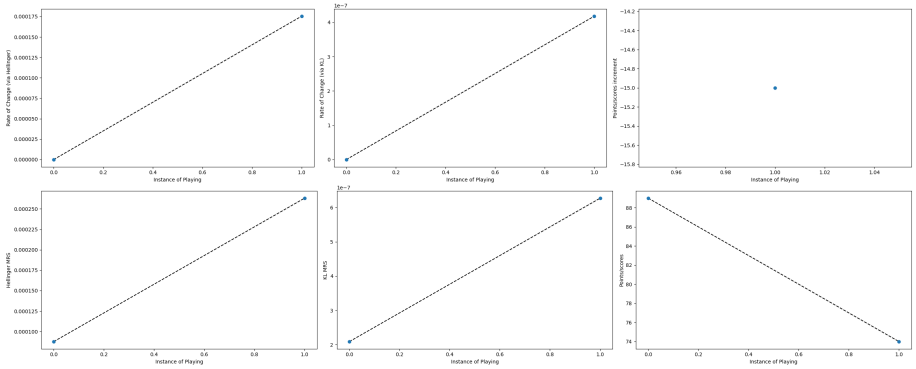


Figure D.39: The comparison of computing scores of Patient ID3379 playing Game Follow 2 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

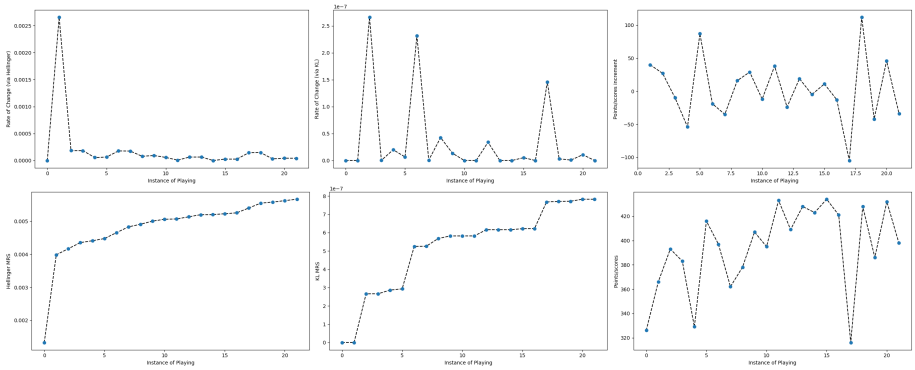


Figure D.40: The comparison of computing scores of Patient ID3430 playing Game Follow 22 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

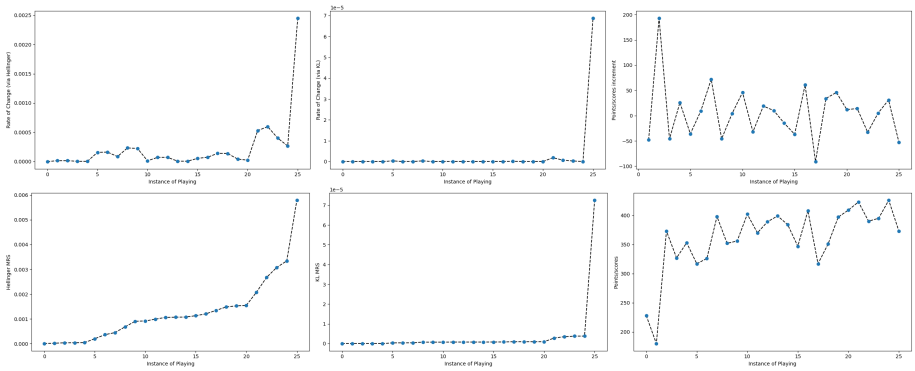


Figure D.41: The comparison of computing scores of Patient ID3431 playing Game Follow 26 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

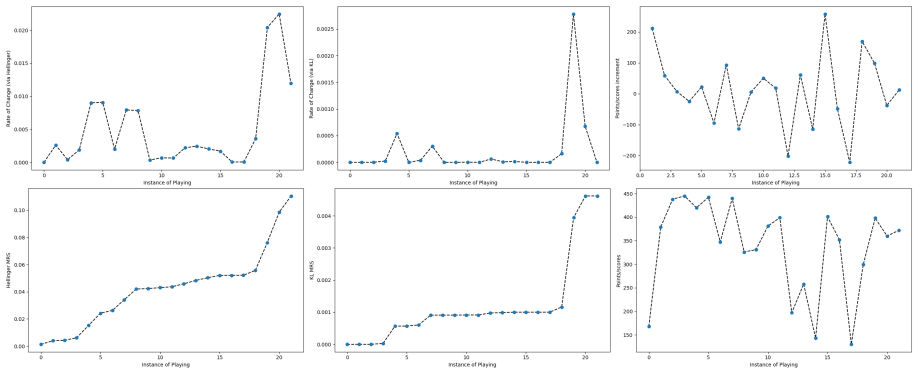


Figure D.42: The comparison of computing scores of Patient ID3462 playing Game Follow 22 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

D.8 Game Forest Leaves

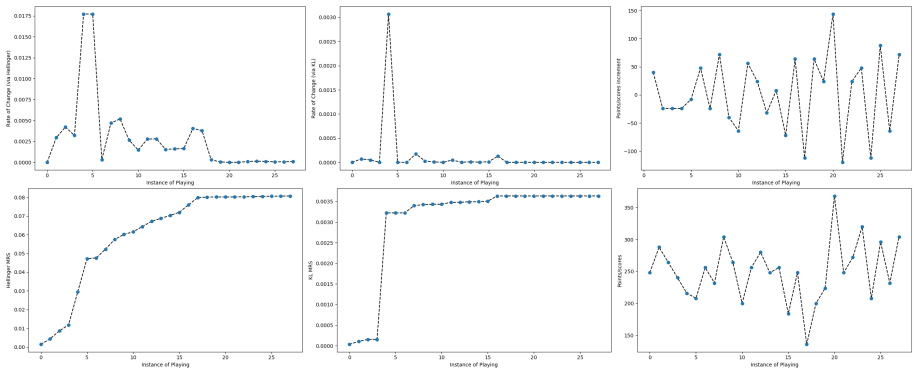


Figure D.43: The comparison of computing scores of Patient ID3085 playing Game Forest Leaves 28 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

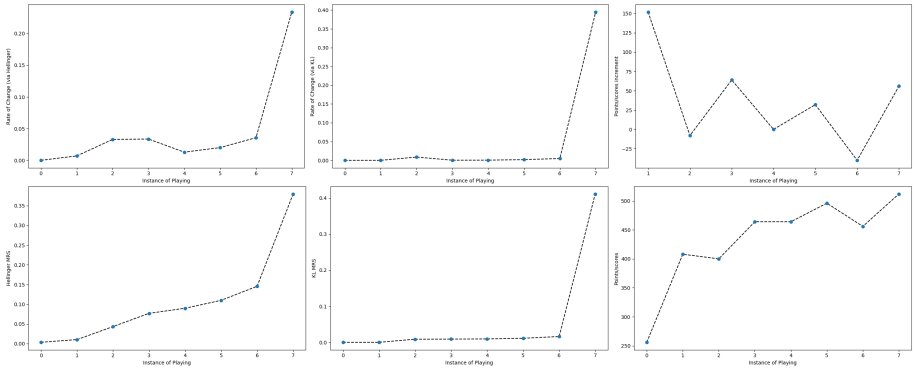


Figure D.44: The comparison of computing scores of Patient ID3147 playing Game Forest Leaves 8 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

D.9 Game Izzy the Bee

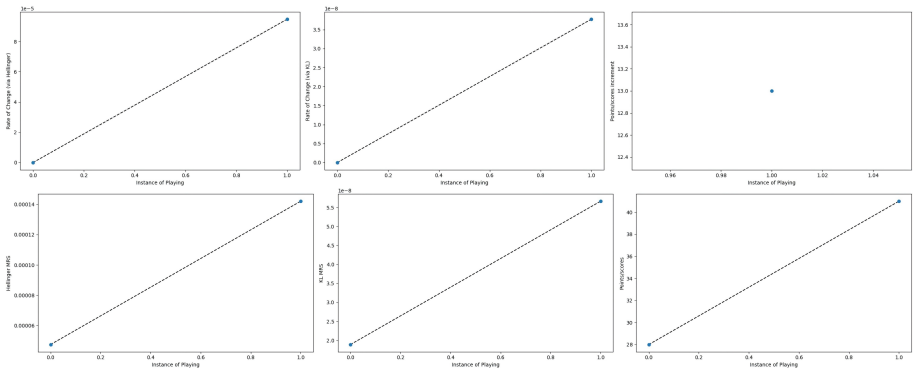


Figure D.45: The comparison of computing scores of Patient ID3011 playing Game Izzy the Bee 2 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

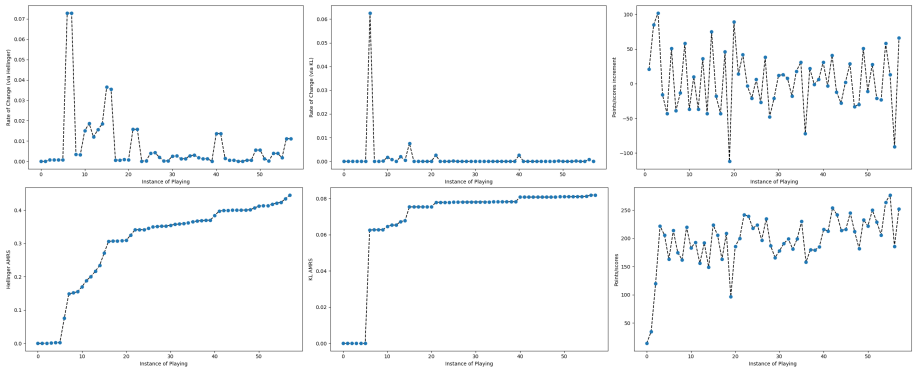


Figure D.46: The comparison of computing scores of Patient ID3070 playing Game Izzy the Bee 58 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

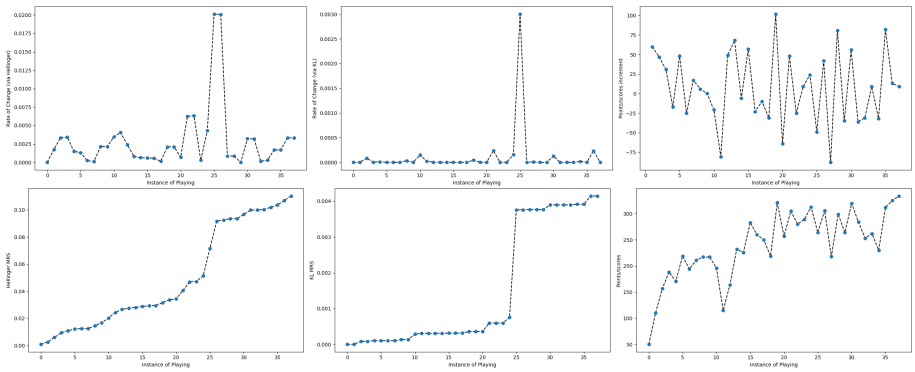


Figure D.47: The comparison of computing scores of Patient ID3071 playing Game Izzy the Bee 38 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

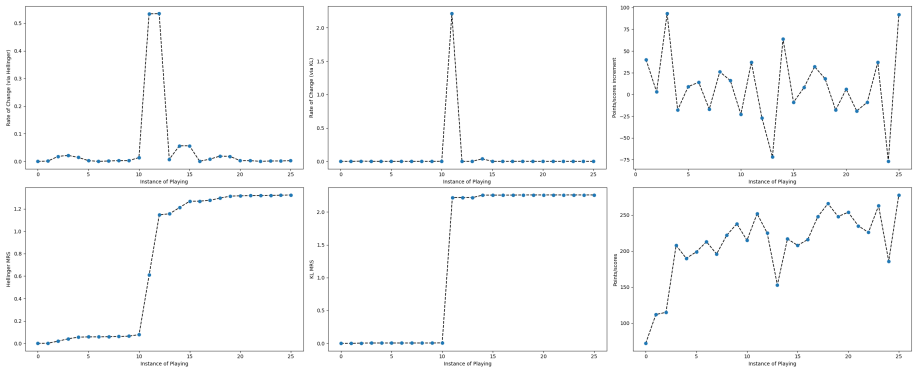


Figure D.48: The comparison of computing scores of Patient ID3085 playing Game Izzy the Bee 26 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

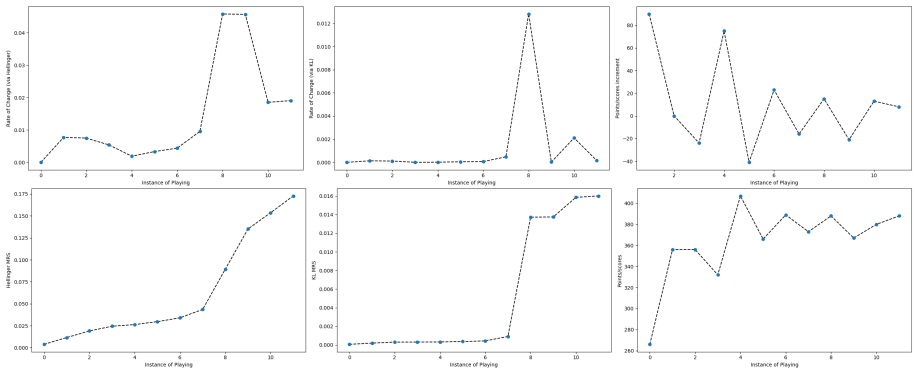


Figure D.49: The comparison of computing scores of Patient ID3147 playing Game Izzy the Bee 12 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

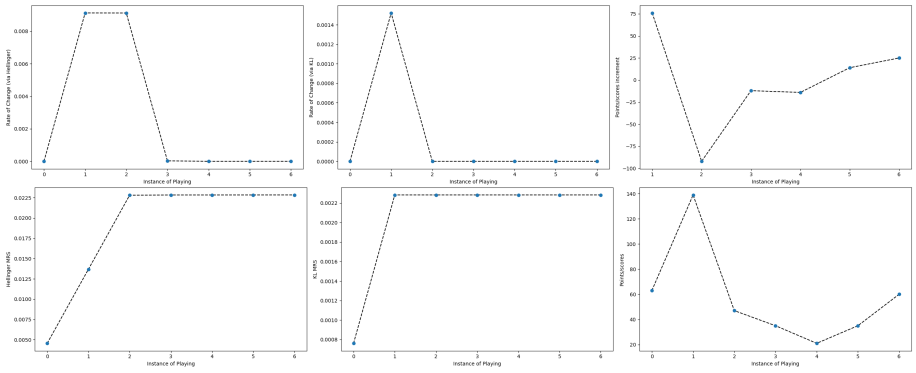


Figure D.50: The comparison of computing scores of Patient ID3165 playing Game Izzy the Bee 7 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

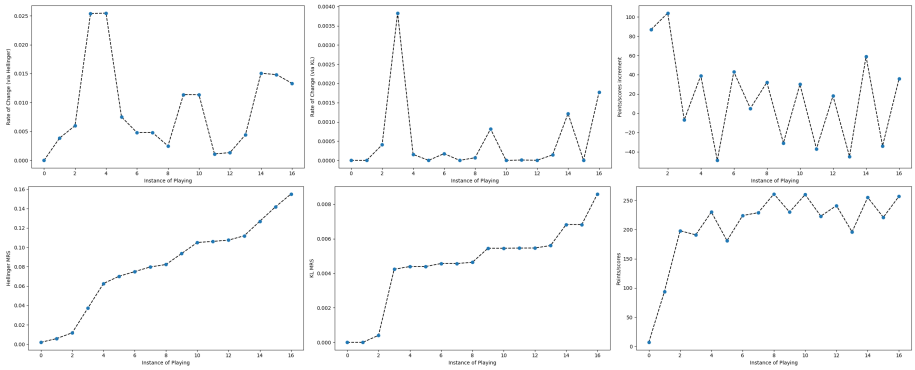


Figure D.51: The comparison of computing scores of Patient ID3311 playing Game Izzy the Bee 17 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

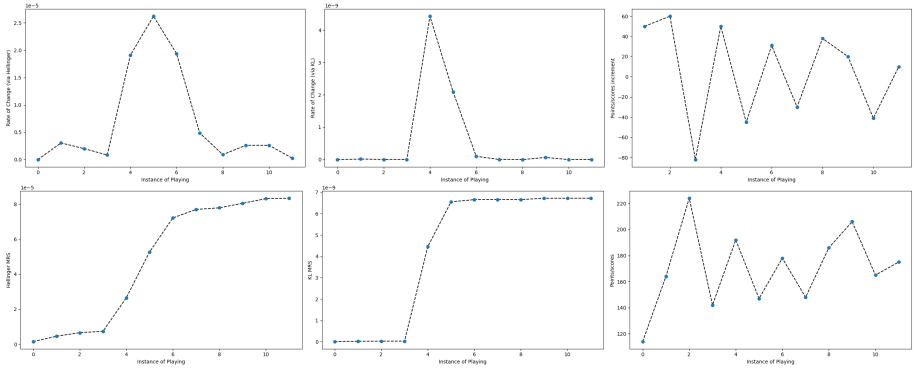


Figure D.52: The comparison of computing scores of Patient ID3327 playing Game Izzy the Bee 12 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

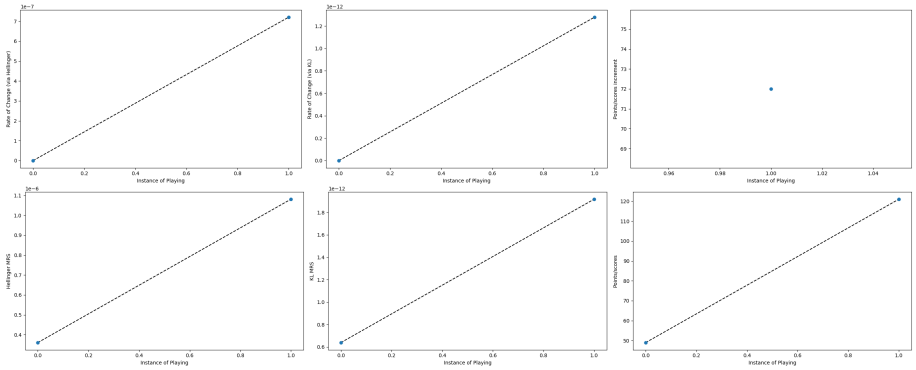


Figure D.53: The comparison of computing scores of Patient ID3379 playing Game Izzy the Bee 2 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

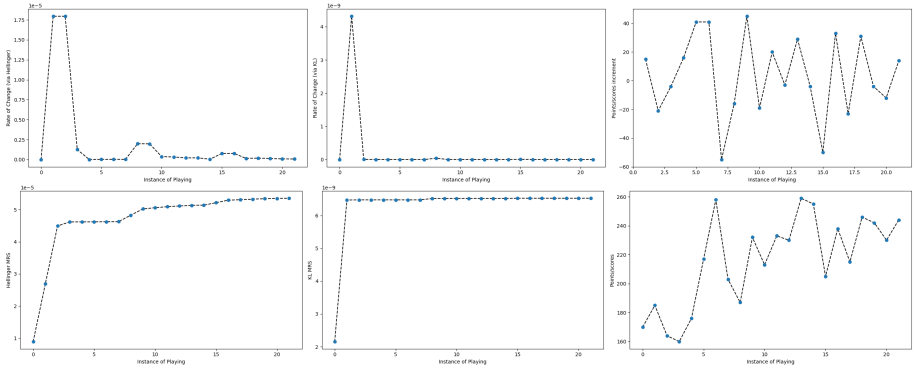


Figure D.54: The comparison of computing scores of Patient ID3430 playing Game Izzy the Bee 22 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

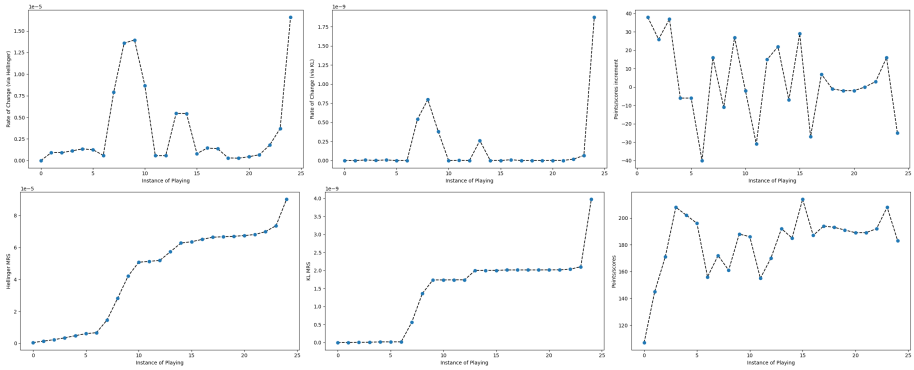


Figure D.55: The comparison of computing scores of Patient ID3431 playing Game Izzy the Bee 26 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

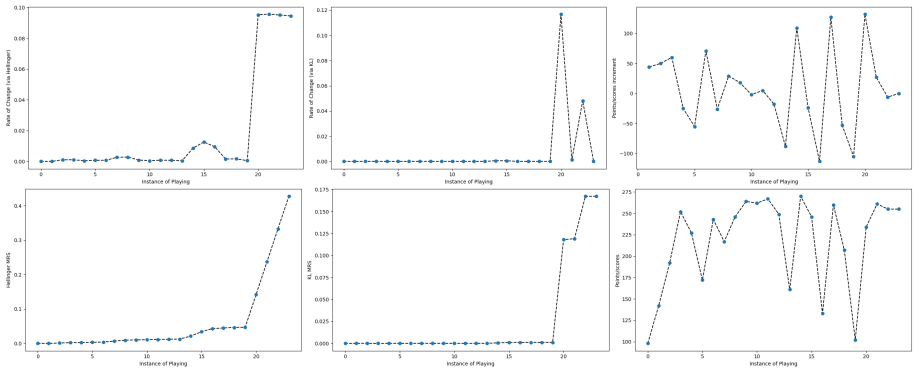


Figure D.56: The comparison of computing scores of Patient ID3462 playing Game Izzy the Bee 24 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

D.10 Game Seansons

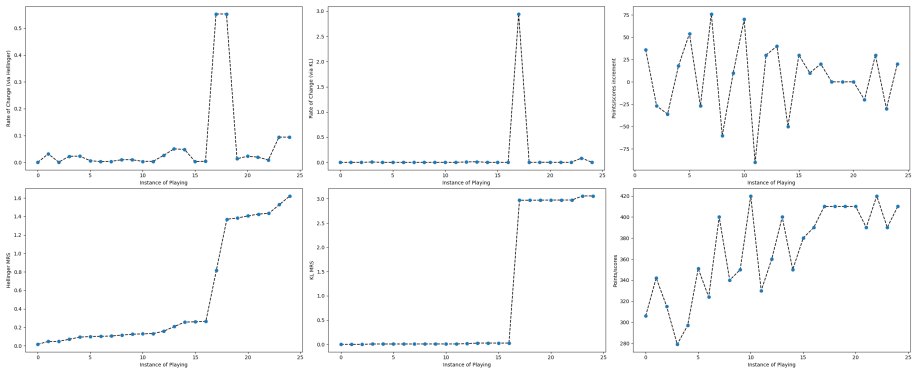


Figure D.57: The comparison of computing scores of Patient ID3147 playing Game Seasons 25 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

D.11 Game Star Find

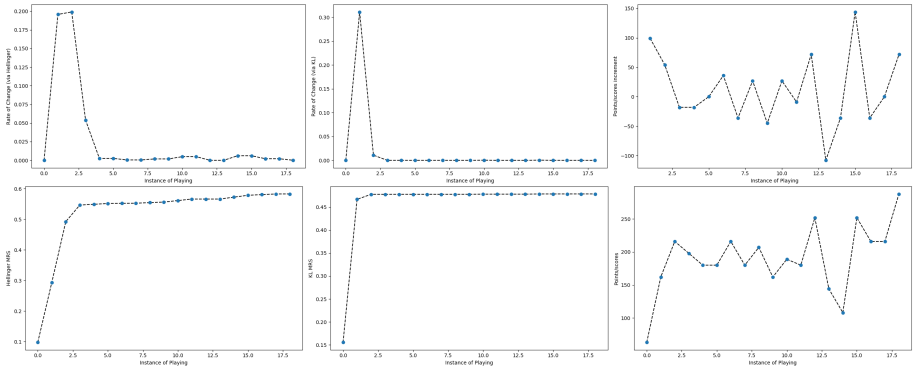


Figure D.58: The comparison of computing scores of Patient ID3071 playing Game Star Find 19 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

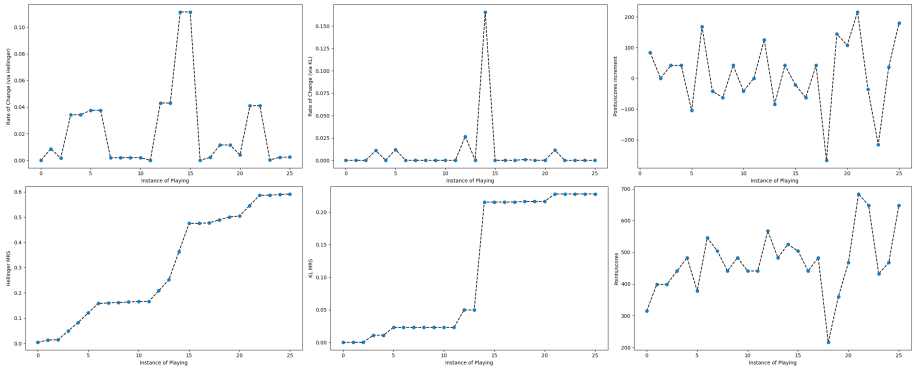


Figure D.59: The comparison of computing scores of Patient ID3147 playing Game Star Find 26 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

D.12 Game Jugger

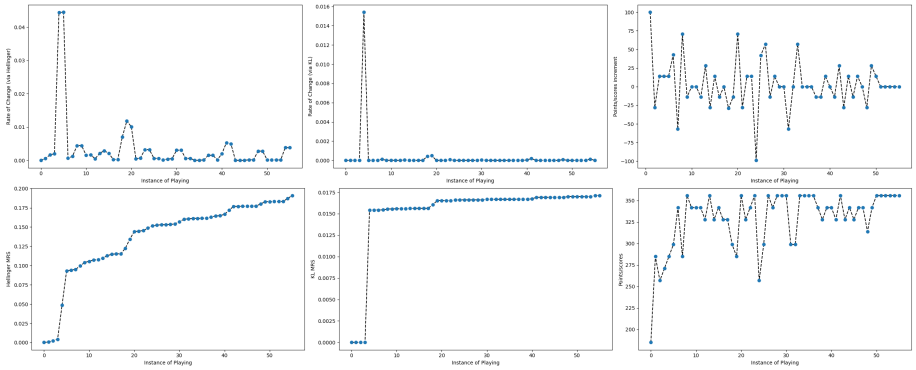


Figure D.60: The comparison of computing scores of Patient ID3070 playing Game Jugger 56 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

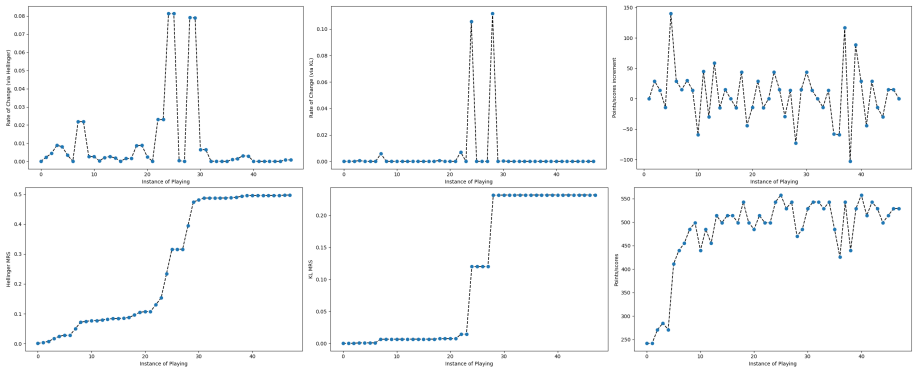


Figure D.61: The comparison of computing scores of Patient ID3071 playing Game Jugger 48 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

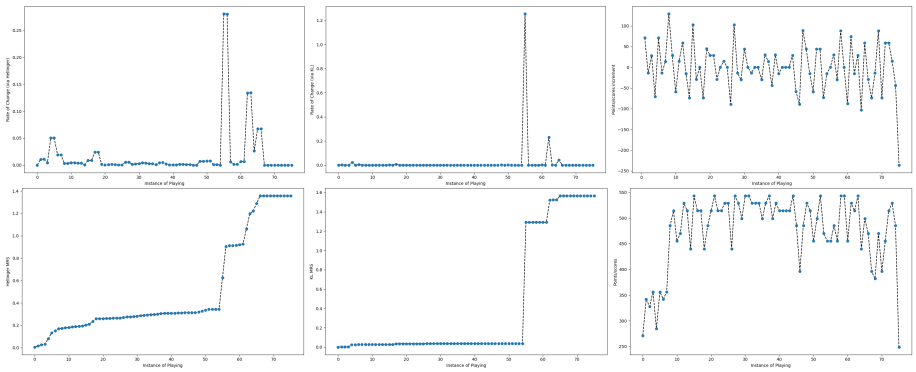


Figure D.62: The comparison of computing scores of Patient ID3085 playing Game Jugger 76 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

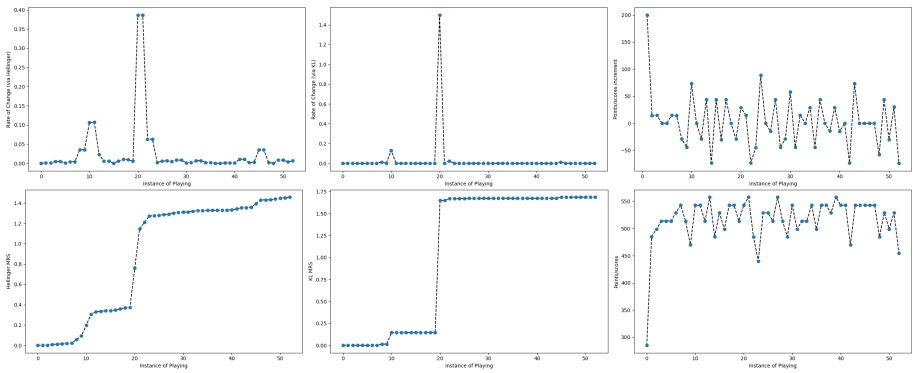


Figure D.63: The comparison of computing scores of Patient ID3147 playing Game Jugger 53 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

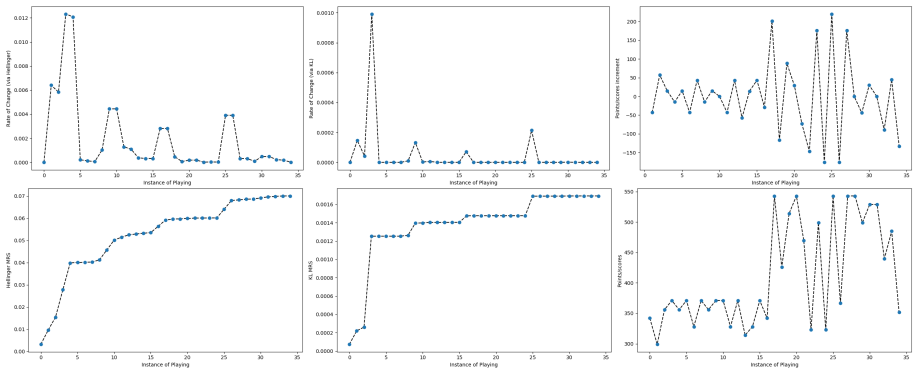


Figure D.64: The comparison of computing scores of Patient ID3311 playing Game Jugger 35 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

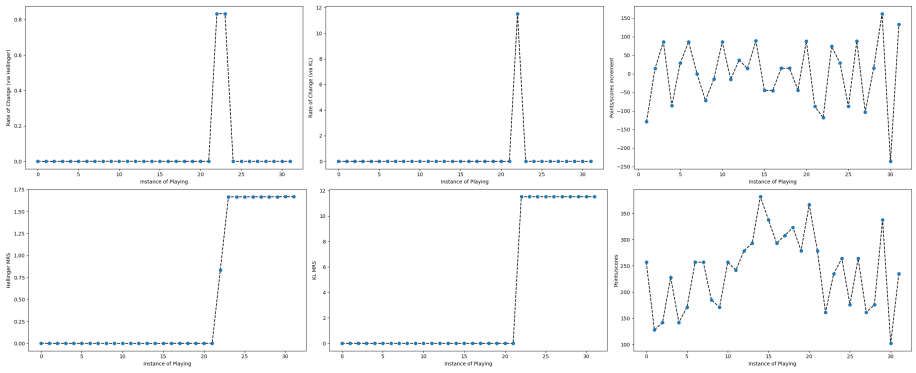


Figure D.65: The comparison of computing scores of Patient ID3327 playing Game Jugger 32 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

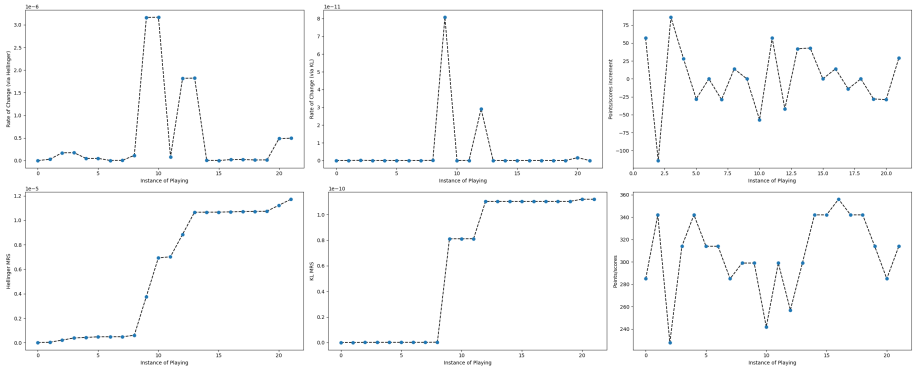


Figure D.66: The comparison of computing scores of Patient ID3430 playing Game Jugger 22 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

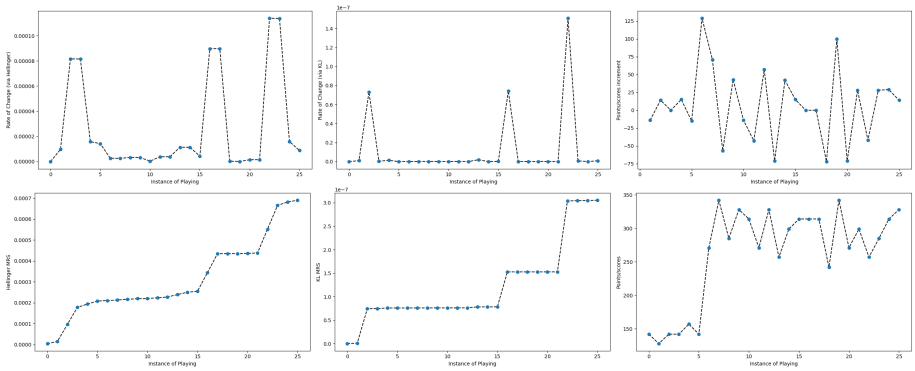


Figure D.67: The comparison of computing scores of Patient ID3431 playing Game Jugger 26 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

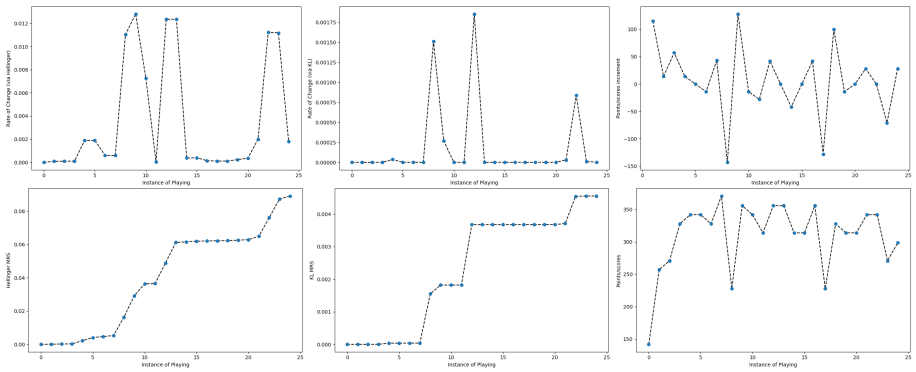


Figure D.68: The comparison of computing scores of Patient ID3462 playing Game Jugger 25 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

D.13 Game Matchday Striker

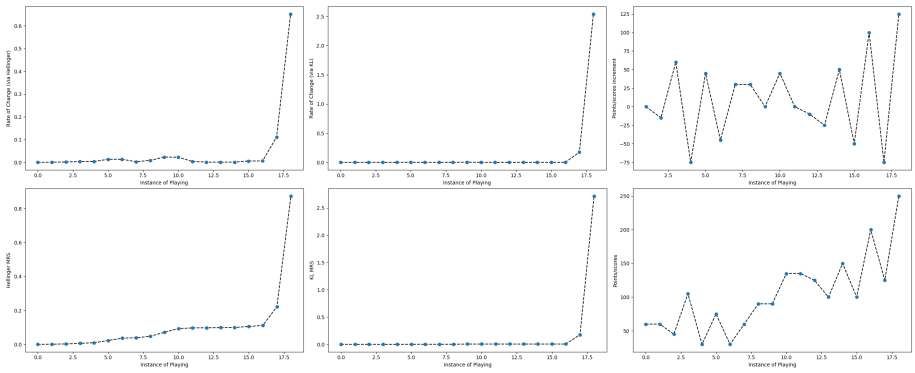


Figure D.69: The comparison of computing scores of Patient ID3071 playing Game Matchday Striker 19 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

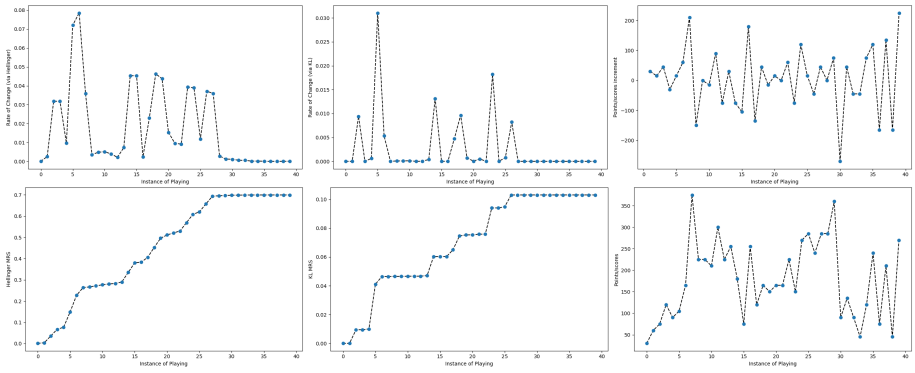


Figure D.70: The comparison of computing scores of Patient ID3085 playing Game Matchday Striker 40 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

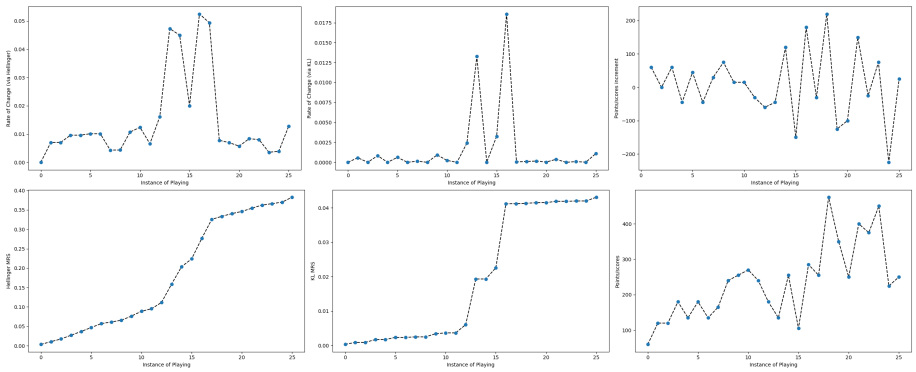


Figure D.71: The comparison of computing scores of Patient ID3147 playing Game Matchday Striker 26 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

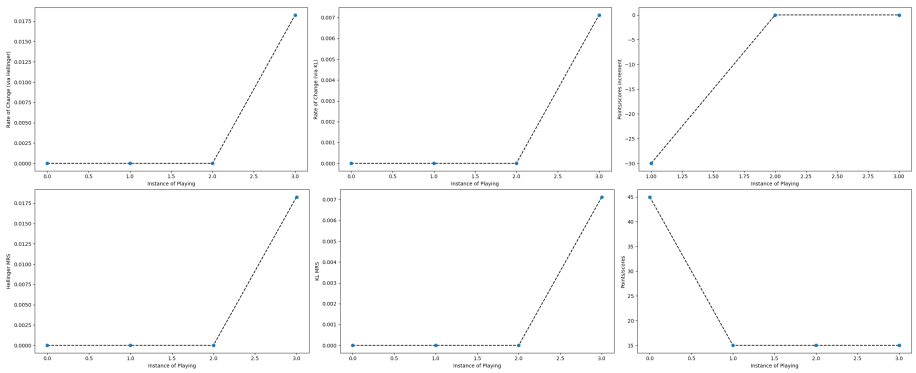


Figure D.72: The comparison of computing scores of Patient ID3165 playing Game Matchday Striker 4 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

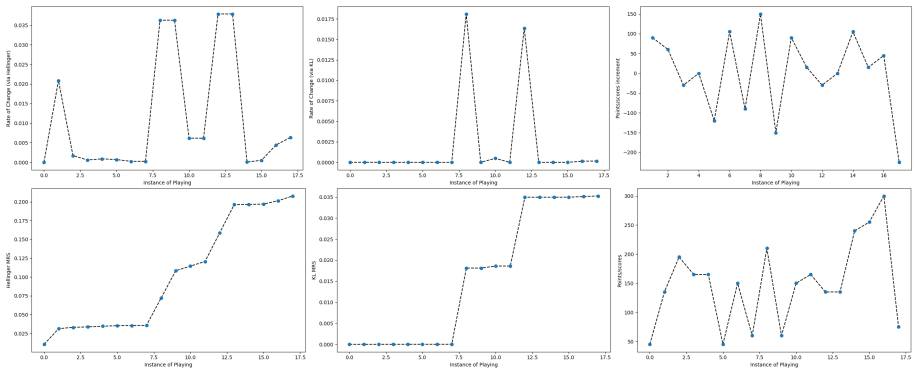


Figure D.73: The comparison of computing scores of Patient ID3311 playing Game Matchday Striker 18 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

D.14 Game Nimble

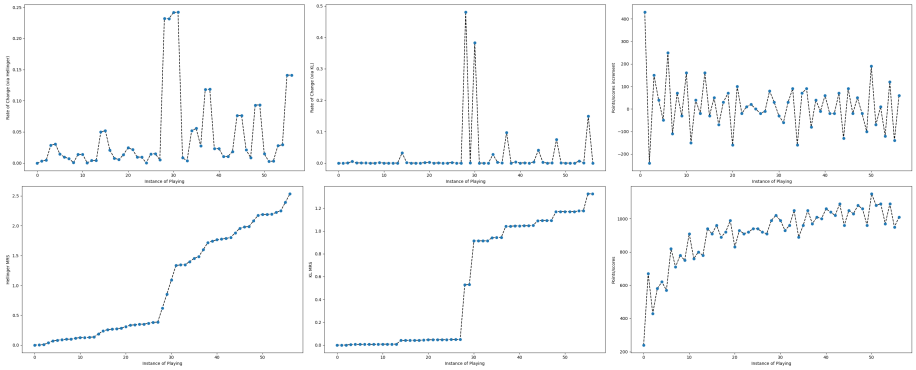


Figure D.74: The comparison of computing scores of Patient ID3070 playing Game Nimble 57 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

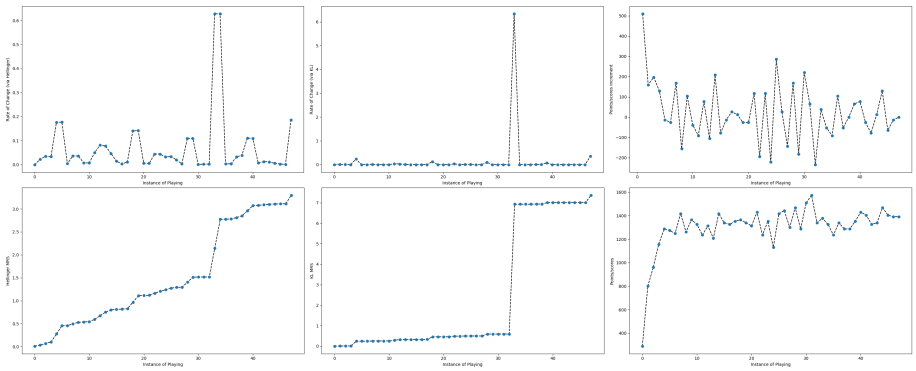


Figure D.75: The comparison of computing scores of Patient ID3071 playing Game Nimble 48 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

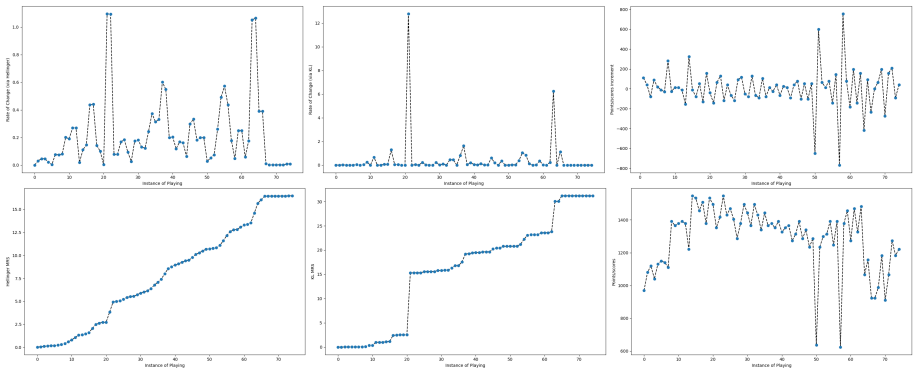


Figure D.76: The comparison of computing scores of Patient ID3085 playing Game Nimble 75 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

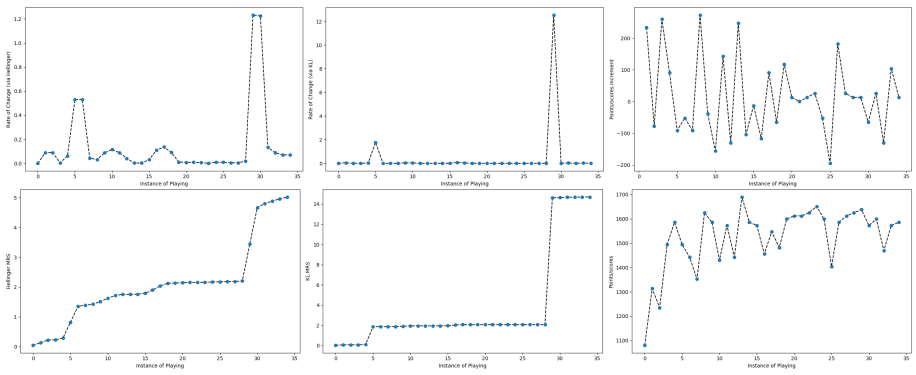


Figure D.77: The comparison of computing scores of Patient ID3147 playing Game Nimble 35 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

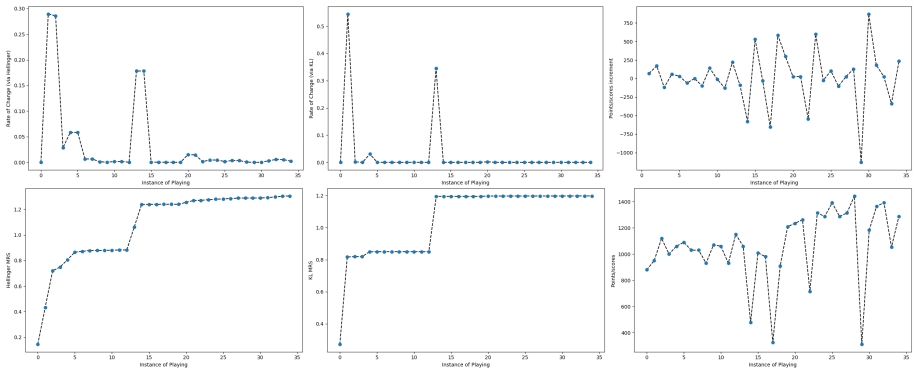


Figure D.78: The comparison of computing scores of Patient ID3311 playing Game Nimble 35 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

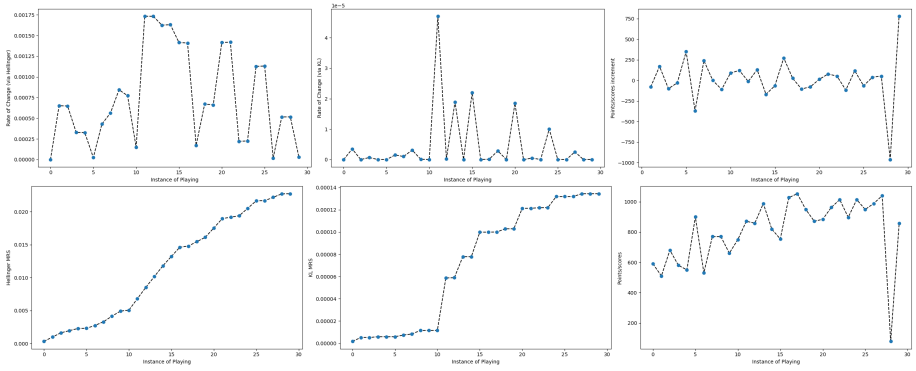


Figure D.79: The comparison of computing scores of Patient ID3327 playing Game Nimble 31 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

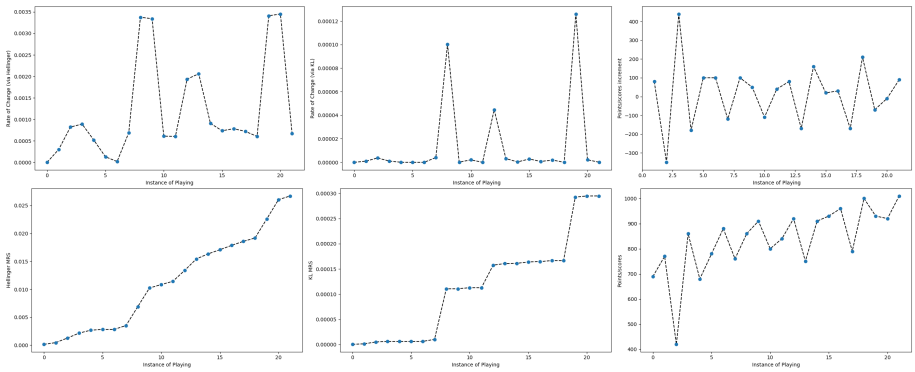


Figure D.80: The comparison of computing scores of Patient ID3430 playing Game Nimble 22 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

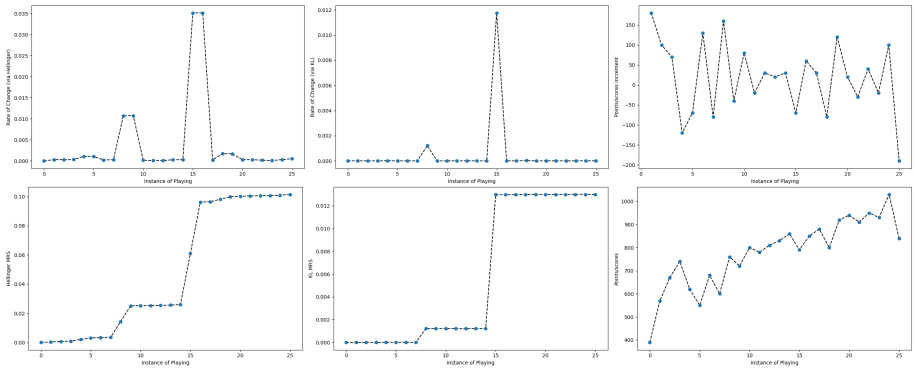


Figure D.81: The comparison of computing scores of Patient ID3431 playing Game Nimble 26 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

D.15 Game Piano

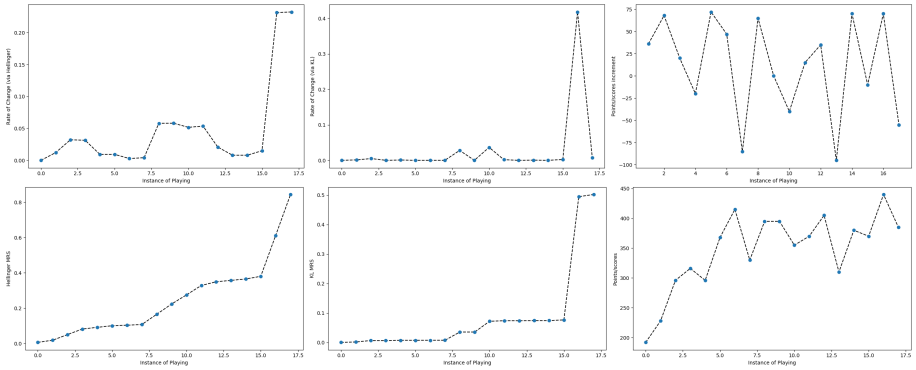


Figure D.82: The comparison of computing scores of Patient ID3071 playing Game Piano 18 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

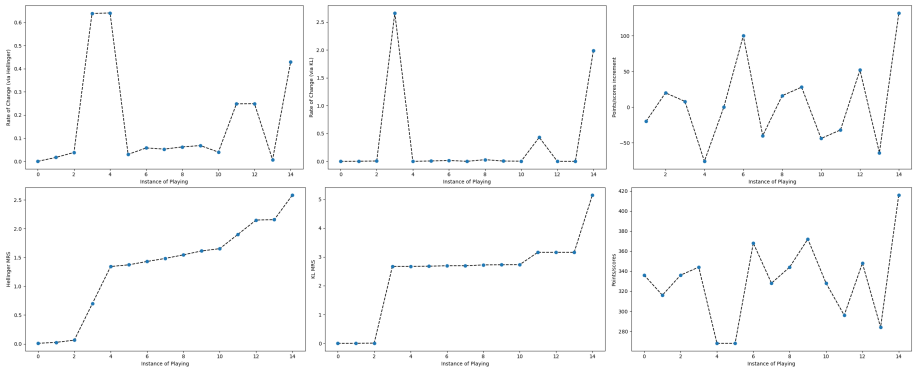


Figure D.83: The comparison of computing scores of Patient ID3085 playing Game Piano 15 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

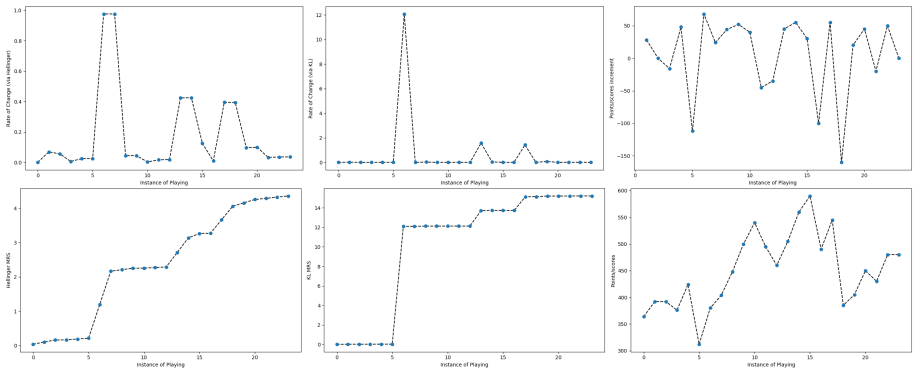


Figure D.84: The comparison of computing scores of Patient ID3147 playing Game Piano 24 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

D.16 Game Space Ship

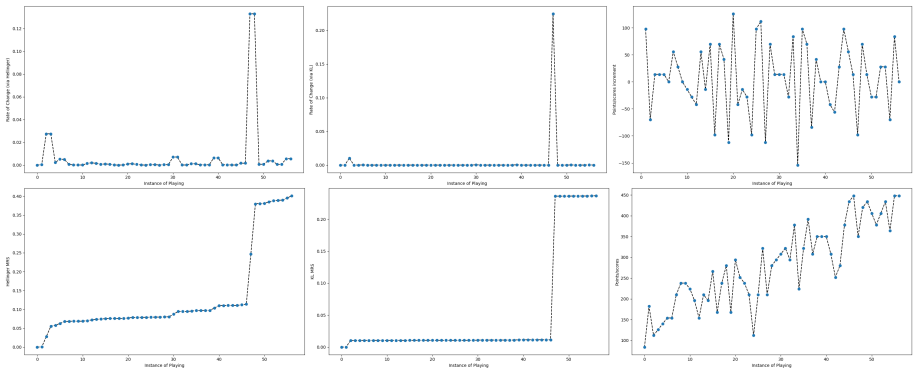


Figure D.85: The comparison of computing scores of Patient ID3070 playing Game Space Ship 58 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

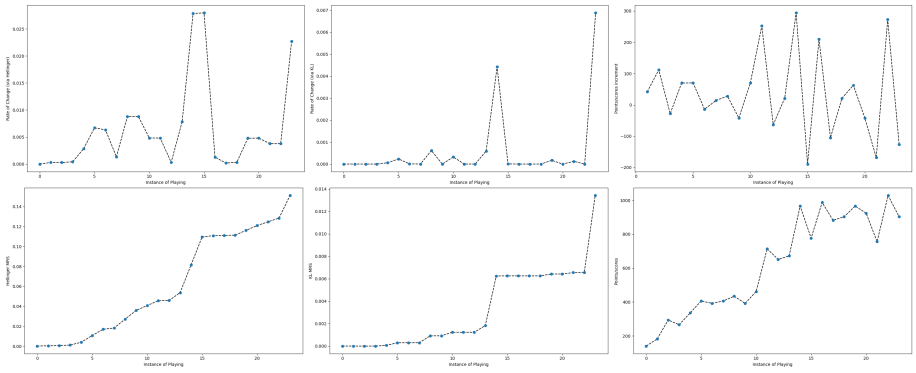


Figure D.86: The comparison of computing scores of Patient ID3071 playing Game Space Ship 24 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

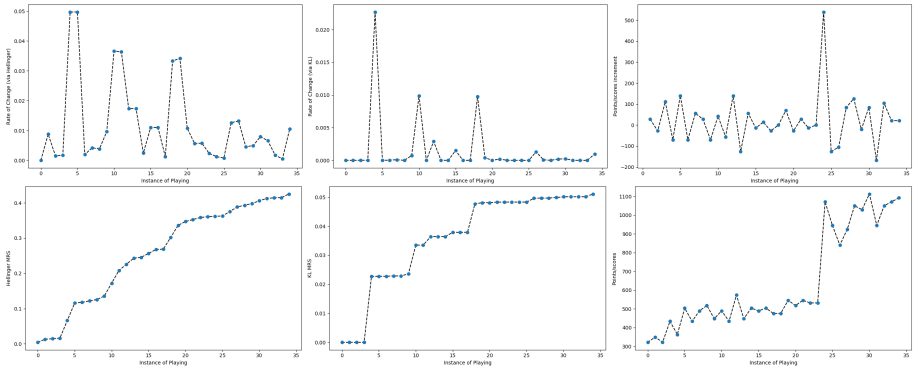


Figure D.87: The comparison of computing scores of Patient ID3085 playing Game Space Ship 35 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

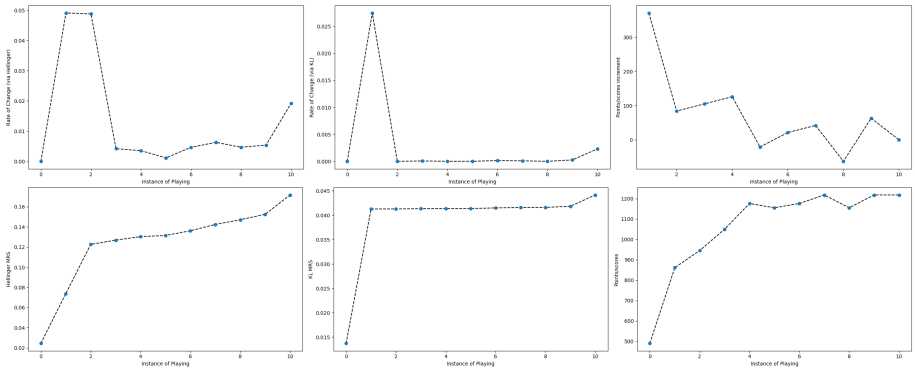


Figure D.88: The comparison of computing scores of Patient ID3147 playing Game Space Ship 11 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

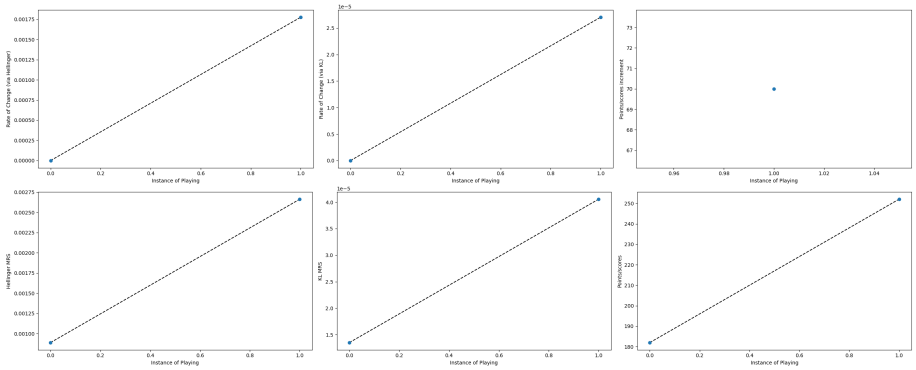


Figure D.89: The comparison of computing scores of Patient ID3165 playing Game Space Ship 2 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

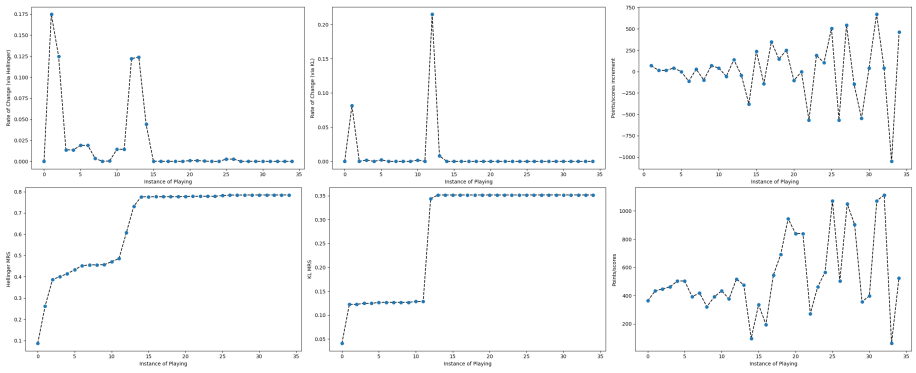


Figure D.90: The comparison of computing scores of Patient ID3311 playing Game Space Ship 35 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

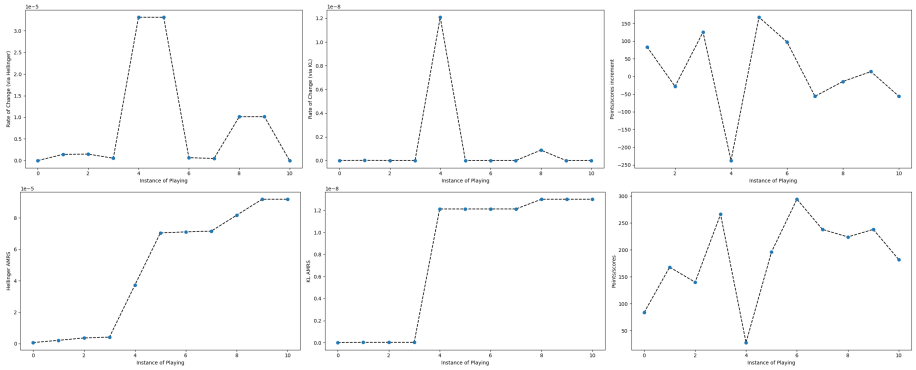


Figure D.91: The comparison of computing scores of Patient ID3327 playing Game Space Ship 12 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

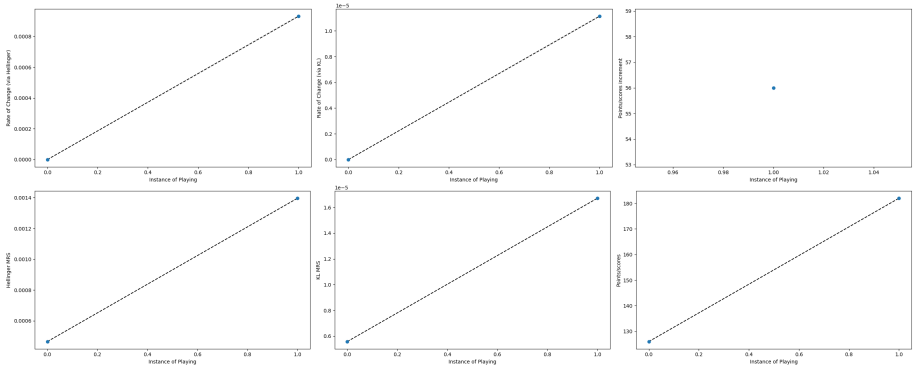


Figure D.92: The comparison of computing scores of Patient ID3379 playing Game Star Find 2 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

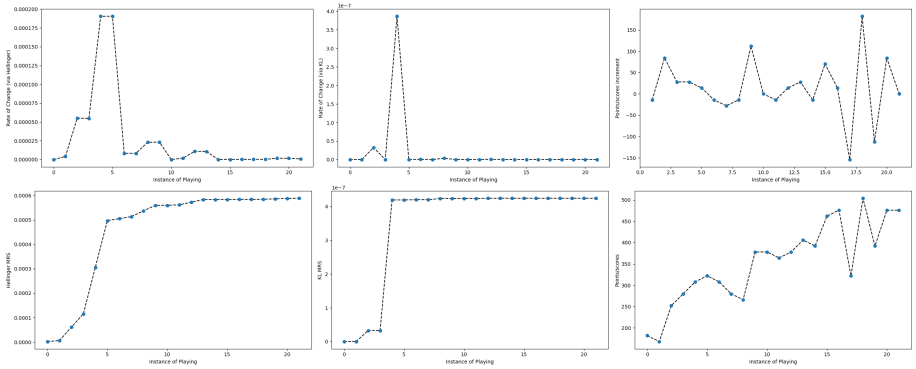


Figure D.93: The comparison of computing scores of Patient ID3430 playing Game Space Ship 22 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

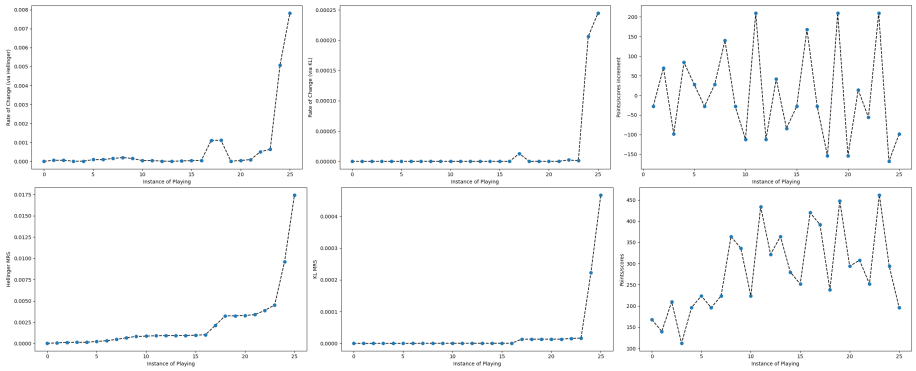


Figure D.94: The comparison of computing scores of Patient ID3431 playing Game Space Ship 26 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

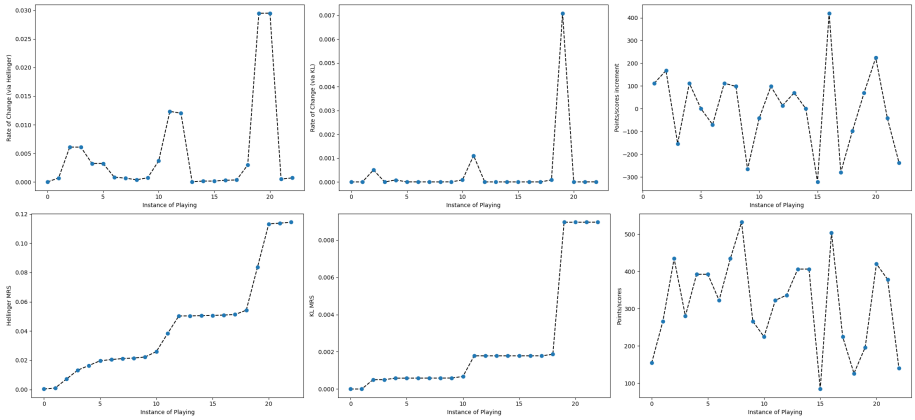


Figure D.95: The comparison of computing scores of Patient ID3462 playing Game Star Find 23 times. The first column plots are the calculated rate of MRS and MRS via Hellinger distance, and the middle column plots are the calculated rate of MRS and MRS using Kullback–Leibler divergence. The plots in the 3rd column are the rate of change of the points/scores provided by *MIRA*, and the points/scores themselves.

Bibliography

Ugo Ala, Rosario Michael Piro, Elena Grassi, Christian Damasco, Lorenzo Silengo, Martin Oti, Paolo Provero, and Ferdinando Di Cunto. Prediction of human disease genes by human-mouse conserved coexpression analysis. *PLoS Computational Biology*, 4(3):e1000043, 2008. doi: 10.1371/journal.pcbi.1000043.

L. A. N. Amaral, A. Scala, M. Barthélemy, and H. E. Stanley. Classes of small-world networks. *Proceedings of the National Academy of Sciences*, 97(21):11149–11152, 2000. doi: 10.1073/pnas.200327197.

T. W. Anderson. *An Introduction to Multivariate Statistical Analysis*. Wiley-Interscience, New York, 3 edition, 2003. ISBN 9780471360919.

Christophe Andrieu, Nando De Freitas, Arnaud Doucet, and Michael I. Jordan. An introduction to mcmc for machine learning. *Machine Learning*, 50(1-2):5–43, 2003.

Søren Asmussen and Peter W. Glynn. *Stochastic Simulation: Algorithms and Analysis*. Springer, 2007.

Paul Balister, Béla Bollobás, Anush Sarkar, and Mark Walters. Connectivity of random k-nearest-neighbour graphs. *Advances in Applied Probability*, 37(1):1–24, 2005.

Sudipto Banerjee, Bradley P. Carlin, and Alan E. Gelfand. *Hierarchical Modeling and Analysis for Spatial Data*. CRC Press, 2014.

Sudipto Banerjee, Bradley P. Carlin, and Alan E. Gelfand. *Hierarchical Modeling and Analysis for Spatial Data (2nd Edition)*. Chapman & Hall/CRC, 2015.

A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.

A.-L. Barabási, R. Albert, and H. Jeong. Scale-free characteristics of random networks: The topology of the world-wide web. *Physica A*, 281:69–77, 2000.

Danielle S. Bassett, Edward Bullmore, Beth A. Verchinski, Venkata S. Mattay, Daniel R. Weinberger, and Andreas Meyer-Lindenberg. Hierarchical organization of human cortical networks in health and schizophrenia. *Journal of Neuroscience*, 28(37):9239–9248, 2008. doi: 10.1523/JNEUROSCI.1929-08.2008.

E. Benoît. Note sur une méthode de résolution des équations normales provenant de l'application de la méthode des moindres carrés à un système d'équations linéaires en nombre inférieur à celui des inconnues (procédé du commandant cholesky). *Bulletin Géodésique*, 2:66–67, 1924. doi: 10.1007/BF03031308.

D. P. Berk, H. Popper, R. G. Krueger, J. Decter, G. Herzig, and G. R. Graw. Veno-occlusive disease of the liver after allogeneic bone marrow transplantation: possible association with graft-versus-host disease. *Annals of Internal Medicine*, 90(2):158–164, 1979. doi: 10.7326/0003-4819-90-2-158.

J.M. Bernardo. Intrinsic credible regions: An objective bayesian approach to interval estimation. *Sociedad de Estadístca e Investigación Operativa*, 14(2):317–384, 2005.

José M. Bernardo and Adrian F. M. Smith. *Bayesian Theory*. Wiley, 1994.

Christopher E Berndsen and Cynthia Wolberger. New insights into ubiquitin e3 ligase mechanism. *Nature structural & molecular biology*, 21(4):301–307, 2014.

Amrita Bhattacharya, Vaibhav Kumar Shukla, Nitin Kachariya, Parveen Sehrawat, Ashutosh Kumar, et al. Disorder in the human skp1 structure is the key to its adaptability to bind many different proteins in the scf complex assembly. *Journal of molecular biology*, 434(21):167830, 2022.

Anil Kumar Bhattacharyya. On a measure of divergence between two statistical populations defined by their probability distributions. *Bulletin of the Calcutta Mathematical Society*, 35:99–109, 1943.

William M. Bolstad. *Understanding Computational Bayesian Statistics*. Wiley, 2010.

Lyuba V. Bozhilova, Javier Pardo-Diaz, Gesine Reinert, and Charlotte M. Deane. Cogent: evaluating the consistency of gene co-expression networks. *Bioinformatics*, 37(13):1928–1929, 2020.

Marianna Capecci, Maria Gabriella Ceravolo, Francesco Ferracuti, Sabrina Iarlori, Andrea Monteriu, Luca Romeo, and Federica Verdini. The kimore dataset: Kinematic assessment of movement and clinical scores for remote monitoring of physical rehabilitation. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 27(7):1436–1448, 2019.

John A. Capra and Mona Singh. Predicting functionally important residues from sequence conservation. *Bioinformatics*, 23(15):1875–1882, 05 2007. doi: 10.1093/bioinformatics/btm270.

E. Carreras, H. Bertz, W. Arcese, J. P. Vernant, J. F. Tomas, H. Hagglund, and et al. Incidence and outcome of hepatic veno-occlusive disease after blood or marrow transplantation: a prospective cohort study of the european group for blood and marrow transplantation. *Blood*, 92(10):3599–3604, 1998.

George Casella and Edward I. George. Explaining the gibbs sampler. *The American Statistician*, 46(3):167–174, 1992.

Petrina P Casey, Anne Marie Feyer, and Ian D Cameron. Course of recovery for whiplash associated disorders in a compensation setting. *Injury*, 46(11):2118–2129, 2015.

Dalia Chakrabarty. *Learning in the Absence of Training Data*. Springer International Publishing, 2023. ISBN 9783031310119. URL https://books.google.co.uk/books?id=FnPLEAAAQBAJ.

Dalia Chakrabarty, Kangrui Wang, Gargi Roy, Akash Bhojgaria, Chuqiao Zhang, Jiri Pavlu, and Joydeep Chakrabartty. Constructing training set using distance between learnt graphical models of time series data on patient physiology, to predict disease scores. *PloS one*, 18(10):e0292404, 2023. doi: https://doi.org/10.1371/journal.pone.0292404.

Sarath Chandra Dantu, Nitin Nathubhai Kachariya, and Ashutosh Kumar. Molecular dynamics simulations elucidate the mode of protein recognition by s kp1 and the f-box domain in the scf complex. *Proteins: Structure, Function, and Bioinformatics*, 84(1):159–171, 2016.

Siddhartha Chib and Edward Greenberg. Understanding the metropolis-hastings algorithm. *The American Statistician*, 49(4):327–335, 1995.

R. Cohen, K. Erez, D. ben Avraham, and S. Havlin. Resilience of the internet to random breakdowns. *Physical Review Letters*, 85(21):4626–4628, 2000. doi: 10.1103/PhysRevLett.85.4626.

J. A. Coppell, P. G. Richardson, R. Soiffer, P. L. Martin, N. A. Kernan, A. Chen, et al. Hepatic veno-occlusive disease following stem cell transplantation: incidence, clinical course, and outcome. *Biology of Blood and Marrow Transplantation*, 16:157–168, 2010. doi: 10.1016/j.bbmt.2009.08.024.

S. Corbacioglu, E. Carreras, M. Ansari, A. Balduzzi, S. Cesaro, J. H. Dalle, et al. Diagnosis and severity criteria for sinusoidal obstruction syndrome/veno-occlusive disease in pediatric patients: a new classification from the european society for blood and marrow transplantation. *Bone Marrow Transplantation*, 53:138–145, 2018. doi: 10.1038/bmt.2017.161.

S. Corbacioglu, E. J. Jabbour, and M. Mohty. Risk factors for development of and progression of hepatic veno-occlusive disease/sinusoidal obstruction syndrome. *Biology of Blood and Marrow Transplantation*, 25(7):1271–1280, 2019. doi: 10.1016/j.bbmt.2019.02.018.

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*, pages 510–525. John Wiley & Sons, Ltd, 1991. doi: https://doi.org/10.1002/0471200611.biblio.

H. Cramér. *Mathematical Methods of Statistics*. Chpater 21. Princeton University Press, 1946. ISBN 9780691080048.

Imre Csiszár. Information-type measures of difference of probability distributions and indirect observation. *Studia Scientiarum Mathematicarum Hungarica*, 2:299–318, 1967.

Michael P. Cusack, Boris Thibert, Dale E. Bredesen, and Gabriel del Rio. Efficient identification of critical residues based only on protein structure by network analysis. *PLoS ONE*, 2(5):e421, 2007. doi: https://doi.org/10.1371/journal.pone.0000421.

S. Dalal, E. B. Fowlkes, and B. Hoadly. Risk analysis of the space shuflte: Prechallenger prediction of failure. *American Statistical Association*, 84(408):945–957, 1989.

Jesper Dall and Michael Christensen. Random geometric graphs. *Physical Review E*, 66(1):016121, 2002.

Alberto de la Fuente, Nan Bing, Ina Hoeschele, and Pedro Mendes. Discovery of meaningful associations in genomic data using partial correlation coefficients. *Bioinformatics*, 20(18):3565–3574, 2004. doi: 10.1093/bioinformatics/bth436.

Pierre Del Moral. *Feynman-Kac Formulae: Genealogical and Interacting Particle Systems with Applications*. Springer, 2004.

Antonio del Sol and Paul O'Meara. Small-world network approach to identify key residues in protein-protein interaction. *Proteins*, 58(3):672–682, 2005. doi: https://doi.org/10.1002/prot.20348.

Carl P. Dettmann. Isolation and connectivity in random geometric graphs with self-similar intensity measures. *Journal of Statistical Physics*, Journal of Statistical Physics 672:679–700, 2018.

Luc Devroye. *Non-Uniform Random Variate Generation*. Springer, 1986. doi: 10.1007/978-1-4613-8643-8.

Luc Devroye and Nicholas Fraiman. Connectivity of inhomogeneous random graphs. *Random Structures & Algorithms*, 45(3):408–420, 2014.

Nadezhda T Doncheva, Yassen Assenov, Francisco S Domingues, and Mario Albrecht. Topological analysis and interactive visualization of biological networks and protein structures. *Nat Protoc*, 7:670–685, 2012. doi: https://doi.org/10.1038/nprot.2012.004.

Joseph L. Doob. *Stochastic Processes*. John Wiley & Sons, New York, 1990. ISBN 978-0-471-52369-7.

M. Drakesmith, K. Caeyenberghs, A. Dutt, G. Lewis, A. S. David, and D. K. Jones. Overcoming the effects of false positives and threshold bias in graph theoretical analyses of neuroimaging data. *NeuroImage*, 118:313–333, 2015. doi: 10.1016/j.neuroimage.2015.05.011.

Richard M. Dudley. *Real Analysis and Probability*. Wadsworth and Brooks/Cole, Pacific Grove, CA, 1989. ISBN 0-534-10050-3.

Adrian H Elcock. Prediction of functionally important residues based solely on the computed energetics of protein structure. *Journal of Molecular Biology*, 312(4):885–896, 2001. ISSN 0022-2836. doi: https://doi.org/10.1006/jmbi.2001.5009.

Paul L. Erdos and Alfréd Rényi. On random graphs. i. *Publicationes Mathematicae*, 6(3–4):290–297, 1959. doi: 10.5486/PMD.1959.6.3-4.12. URL https://doi.org/10.5486/PMD.1959.6.3-4.12. Archived (PDF) from the original on 2020-08-07. Retrieved 2011-02-23.

Valery L Feigin, Michael Brainin, Bo Norrving, Sheila Martins, Ralph L Sacco, Werner Hacke, Marc Fisher, Jeyaraj Pandian, and Patrice Lindsay. World Stroke Organization (WSO): Global stroke fact sheet 2022. *Int J Stroke*, 17(4):479, 2022.

Younis M. S. Firwana, Mohd Khairul Izamil Zolkefley, Hasnetty Zuria Mohamed Hatta, Christina Rowbin, Che Mohd Nasril Che Mohd Nassir, Muhammad Hafiz Hanafi, Mohd Shafie Abdullah, Bilgin Keserci, Natasha A. Lannin, and Muzaimi Mustapha. Regional cerebral blood perfusion changes in chronic stroke survivors as potential brain correlates of the functional outcome following gamified home-based rehabilitation (intellirehab)—a pilot study. *Journal of NeuroEngineering and Rehabilitation*, 19(1):94, 2022.

Tali Fishman-Jacob and Moussa BH Youdim. A sporadic parkinson's disease model via silencing of the ubiquitin–proteasome/e3 ligase component, skp1a. *Journal of Neural Transmission*, 131(6):675–707, 2024.

Ionuț Florescu. *Probability and Stochastic Processes*. John Wiley & Sons, Hoboken, NJ, 2014. ISBN 978-1-118-59320-2.

George E. Forsythe. Von neumann's comparison method for random sampling from the normal and other distribution. *Mathematic of Computation*, 26(120):817–826, 1972.

Fotos Frangoudes, Maria Matsangidou, Eirini C. Schiza, Kleanthis C. Neokleous, and Constantinos S. Pattichis. Assessing human motion during exercise using machine learning: A literature review. *IEEE Access*, 10:86874–86903, 2022.

Tom C. Freeman, Leon Goldovsky, Markus Brosch, Stijn van Dongen, Pierre Mazière, Russell J. Grocock, Shiri Freilich, Janet Thornton, and Anton J. Enright. Construction, visualisation, and clustering of transcription networks from microarray expression data. *PLoS Computational Biology*, 3(10):e206, 2007. doi: 10.1371/journal.pcbi.0030206.

Alan E. Gelfand and Adrian F. M. Smith. Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association*, 85(410):398–409, 1990.

Andrew Gelman. Prior distributions for variance parameters in hierarchical models. *Bayesian Analysis*, 1(3):515–533, 2006.

Andrew Gelman, John B. Carlin, Hal S. Stern, and Donald B. Rubin. *Bayesian Data Analysis*. Chapman & Hall, 1995.

Andrew Gelman, Aleks Jakulin, Maria Grazia Pittau, and Yu-Sung Su. A weakly informative default prior distribution for logistic and other regression models. *The Annals of Applied Statistics*, 2(4):1360–1383, 2008.

Andrew Gelman, John B. Carlin, Hal S. Stern, David B. Dunson, Aki Vehtari, and Donald B. Rubin. *Bayesian Data Analysis Third edition*. 2013.

Stuart Geman and Donald Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6):721–741, 1984.

Tadesse Ghirmai. Applying metropolis-hastings-within-gibbs algorithms for data detection in relay-based communication systems. *IEEE Signal Processing and Signal Processing Education Workshop (SP/SPE)*, 2015.

Adrian L. Gibbs and Francis Edward Su. On choosing and bounding probability metrics. *International Statistical Review*, 70(3):419–435, 2002. doi: 10.1111/j.1751-5823.2002.tb00178.x.

E. N. Gilbert. Random plane networks. *Journal of the Society for Industrial and Applied Mathematics*, 9(4):533–543, 1961.

Alexander P Giles, Carl P Dettmann, and Orestis Georgiou. Connectivity of soft random geometric graphs over annuli. *J Stat Phys*, 162:1068–1083, 2016.

Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, MD, 3rd edition, 1996.

Peter J. Green. Reversible jump markov chain monte carlo computation and bayesian model determination. *Biometrika*, 82(4):711–732, 1995.

P Gupta and PR Kumar. Critical power for asymptotic connectivity in wireless networks. *Stochastic Analysis, Control, Optimization and Applications: A Volume in Honor of W.H. Fleming*, pages 547–566, 1998.

Rebecca Hamer, Qiang Luo, Judith P Armitage, Gesine Reinert, and Charlotte M Deane. i-patch: interprotein contact prediction using local network information. *Proteins*, 78(13):2781–2797, 2010. doi: https://doi.org/10.1002/prot.22792.

Richard W. Hamming. Error detecting and error correcting codes. *The Bell System Technical Journal*, 29(2):147–160, 1950. doi: 10.1002/j.1538-7305.1950.tb00463.x.

Nurit Haspel and Filip Jagodzinski. Methods for detecting critical residues in proteins. *In Vitro Mutagenesis*, 1498:227–242, 2017. doi: https://doi.org/10.1007/978-1-4939-6472-7_15.

W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

Ernst Hellinger. Neue begründung der theorie quadratischer formen von unendlich vielen veränderlichen. *Journal für die reine und angewandte Mathematik*, 136: 210–271, 1909. doi: 10.1515/crll.1909.136.210.

Francisco J. Herrera, Jean Wong, and Frances Chung. A systematic review of postoperative recovery outcomes measurements after ambulatory surgery. *Anesthesia and Analgesia*, 105(1):63–69, 2007.

Thomas A Hopf, John B Ingraham, Frank J Poelwijk, Charlotta P I Schärfe, Michael Springer, Chris Sander, and Debora S Marks. Mutation effects predicted from sequence co-variation. *Nature Biotechnology*, 35(2):128–135, January 2017. ISSN 1546-1696. doi: 10.1038/nbt.3769. URL http://dx.doi.org/10.1038/nbt.3769.

Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, Cambridge, 1985.

Harold Hotelling. Relations between two sets of variates. *Biometrika*, 28(3/4): 321–377, 1936.

Ayman Jarrous and Benny Pinkas. Secure hamming distance based computation and its applications. In Michel Abdalla, David Pointcheval, Pierre-Alain Fouque, and Damien Vergnaud, editors, *Applied Cryptography and Network Security*, volume 5536 of *Lecture Notes in Computer Science*, pages 107–124, Berlin, Heidelberg, 2009. Springer. ISBN 978-3-642-01957-9. doi: 10.1007/978-3-642-01957-9_7.

Richard A. Johnson and Dean W. Wichern. *Applied Multivariate Statistical Analysis*. Pearson, Boston, 7 edition, 2019. ISBN 9780134995398.

Robbie P Joosten, Tim AH Te Beek, Elmar Krieger, Maarten L Hekkelman, Rob WW Hooft, Reinhard Schneider, Chris Sander, and Gert Vriend. A series of pdb related databases for everyday needs. *Nucleic acids research*, 39(suppl_1): D411–D419, 2010.

Tony Jung and K. A. S. Wickrama. An introduction to latent class growth analysis and growth mixture modeling. *Social and Personality Psychology Compass*, 2(1): 302–317, 2008.

Wolfgang Kabsch and Christian Sander. Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers: Original Research on Biomolecules*, 22(12):2577–2637, 1983.

Konstantinos F. Kantelis, Vassilios Asteriou, Aliki Papadimitriou-Tsantarliotou, Anthi Petrou, Lefteris Angelis, Petros Nicopolitidis, Georgios Papadimitriou, and Ioannis S. Vizirianakis. Graph theory-based simulation tools for protein structure networks. *Simulation Modelling Practice and Theory*, 121:102640, 2022. ISSN 1569-190X. doi: https://doi.org/10.1016/j.simpat.2022.102640. URL https://www.sciencedirect.com/science/article/pii/S1569190X22001137.

Leonid V. Kantorovich. Mathematical methods of organizing and planning production. *Management Science*, 6(4):366–422, 1939. doi: 10.1287/mnsc.6.4.366.

H Kasch, F W Bach, and T S Jensen. Handicap after acute whiplash injury: a 1-year prospective study of risk factors. *Neurology*, 56(12):1637–1643, 2001.

Maurice G. Kendall and Alan F. Stuart. *The Advanced Theory of Statistics: Vol. 2 – Inference and Relationship*. Charles Griffin & Company, 4th edition, 1979.

J.K. Kruschke. *Doing Bayesian Data Analysis*. 2014.

Solomon Kullback and Richard A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951.

P Kumar and P Gupta. The capacity of wireless networks. *IEEE Transactions on Information Theory*, 46(2):388–404, 2000.

R. Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tomkins, and E. Upfal. Stochastic models for the web graph. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science (FOCS)*, 2000.

John Lamperti. *Stochastic Processes: A Survey of the Mathematical Theory*. Springer-Verlag, New York, 1977. ISBN 978-3-540-90275-1.

Steffen L. Lauritzen. *Graphical Models*. Oxford University Press, 1996.

Joshua Y. Lee, David M. Walton, Paul Tremblay, Curtis May, Wanda Millard, James M. Elliott, and Joy C. MacDermid. Defining pain and interference recovery trajectories after acute non-catastrophic musculoskeletal trauma through growth mixture modeling. *BMC Musculoskelet Disord*, 21(1), 2020.

S. Lee, E. Lee, S. S. Park, S. M. Park, J. Jung, J. G. Min, and et al. Prediction and recommendation by machine learning through repetitive internal validation for hepatic veno-occlusive disease/sinusoidal obstruction syndrome and early death after allogeneic hematopoietic cell transplantation. *Bone Marrow Transplantation*, 57(4):538–546, 2022. doi: 10.1038/s41409-022-01583-z.

Jianhua Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151, 1991.

Yong-Qiang Liu, Xiao-Lu Wang, Xin Cheng, Yong-Zhi Lu, Gui-Zhen Wang, Xin-Chun Li, Jian Zhang, Zhe-Sheng Wen, Zhi-Liang Huang, Qin-Lei Gao, et al. Skp1 in lung cancer: clinical significance and therapeutic efficacy of its small molecule inhibitors. *Oncotarget*, 6(33):34953, 2015.

Steve W. Lockless and Rama Ranganathan. Evolutionarily conserved pathways of energetic connectivity in protein families. *Science*, 286(5438):295–299, 1999. doi: 10.1126/science.286.5438.295.

Guoqiang Mao and Brian D O Anderson. Towards a better understanding of large-scale network models. *IEEE/ACM Transactions on Networking*, 20(2):408–421, 2012.

Guillaume Marrelec, Alexandre Krainik, Hugues Duffau, Mélanie Pélégrini-Issac, Stéphane Lehéricy, Julien Doyon, and Habib Benali. Partial correlation for functional brain interactivity investigation in functional mri. *NeuroImage*, 32(1):228–237, 2006. doi: 10.1016/j.neuroimage.2005.12.057.

Angélica S. Mata. An overview of epidemic models with phase transitions to absorbing states running on top of complex networks. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 31(1):012101, 01 2021. ISSN 1054-1500. doi: 10.1063/5.0033130. URL https://doi.org/10.1063/5.0033130.

George B. McDonald, P. Sharma, David E. Matthews, Howard M. Shulman, and D. E. Thomas. Veno-occlusive disease of the liver after bone marrow transplantation: Diagnosis, incidence, and predisposing factors. *Hepatology*, 4(1):116–122, 1984. doi: 10.1002/hep.1840040121.

Karl Menger. Statistical metrics. *Proc Natl Acad Sci U S A.*, 28(12):535–537, 1942. doi: https://doi.org/10.1073/pnas.28.12.535.

Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953.

M. Mohty, F. Malard, M. Abecassis, E. Aerts, A. S. Alaskar, M. Aljurf, et al. Revised diagnosis and severity criteria for sinusoidal obstruction syndrome/veno-occlusive disease in adult patients: a new classification from the european society for blood and marrow transplantation. *Bone Marrow Transplantation*, 51:906–912, 2016. doi: 10.1038/bmt.2016.130.

Bengt Muthén and Linda K. Muthén. Integrating person-centered and variable-centered analyses: Growth mixture modeling with latent trajectory classes. *Alcoholism: Clinical and Experimental Research*, 24(6):882–891, 2000.

S Muthukrishnan, G Pandurangan, and R Rajaraman. Thresholds for connectivity in superposition of random key graphs on random geometric graphs. In *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 954–963, 2005.

Daniel S Nagin. Analyzing developmental trajectories: A semiparametric, group-based approach. *Psychological Methods*, 4:139–157, 1999.

Daniel S Nagin. *Group-Based Modeling of Development*. Harvard University Press, Cambridge, Massachusetts, United States, 2005.

Radford M. Neal. Probabilistic inference using markov chain monte carlo methods. *Technical Report CRG-TR-93-1, University of Toronto*, 1993.

M. E. Newman, D. J. Watts, and S. H. Strogatz. Random graph models of social networks. *Proceedings of the National Academy of Sciences of the United States of America*, 99(Suppl 1):2566–2572, 2002. doi: 10.1073/pnas.012582999. URL https://doi.org/10.1073/pnas.012582999.

M. S. Nikulin. Hellinger distance. In *Encyclopedia of Mathematics*. EMS Press, 2001. [Originally published in 1994].

Sadawi Noureddin, Miron Alina, Ismail Waidah, Hussain Hafez, and Grosan Crina. Gesture correctness estimation with deep neural networks and rough path descriptors. *2019 International Conference on Data Mining Workshops (ICDMW)*, pages 595–602, 11 2019.

Esther B Olasoji, Daniel K Uhm, Oluwole O Awosika, Sylvain Doré, Carolyn Geis, and Alexis N Simpkins. Trends in outpatient rehabilitation use for stroke survivors. *Journal of the neurological sciences*, 442:120383, 2022.

World Health Organization. International classification of functioning, disability and health(ICF), 2001.

Reza Haghighi Osgouei, David Soulsby, and Fernando Bello. Rehabilitation exergames: Use of motion sensing and machine learning to quantify exercise performance in healthy volunteers. *JMIR Rehabil Assist Technol*, 7(2):e17289, 2020.

Dejan Ozegovic, Linda J. Carroll, and J. David Cassidy. Does expecting mean achieving? the association between expecting to return to work and recovery in whiplash associated disorders: a population-based prospective cohort study. *Eur Spine J*, 18(6):893–899, 2009.

Adeline T M Paiement, Lili Tao, Sion L Hannuna, Massimo Camplani, Dima Damen, and Majid Mirmehdi. Online quality assessment of human movement from skeleton data. *Proceedings of the British Machine Vision Conference 2014*, 2014.

Guus Panken, Trynke Hoekstra, Arianne Verhagen, Maurits van Tulder, Jos Twisk, and Martijn W Heymans. Predicting chronic low-back pain based on pain trajectories in patients in an occupational setting: an exploratory analysis. *Scand J Work Environ Health*, 42(6):520–527, 2016.

Javier Pardo-Diaz, Lyuba V Bozhilova, Mariano Beguerisse-Díaz, Philip S Poole, Charlotte M Deane, and Gesine Reinert. Robust gene coexpression networks using signed distance correlation. *Bioinformatics*, 37(14):1982–1989, 02 2021. ISSN 1367-4803. doi: 10.1093/bioinformatics/btab041.

Javier Pardo-Diaz, Philip S Poole, Mariano Beguerisse-Díaz, Charlotte M Deane, and Gesine Reinert. Generating weighted and thresholded gene coexpression networks using signed distance correlation. *Network science (Cambridge University Press)*, 10(2):131–145, 2022.

Mathew D Penrose. On k-connectivity for a geometric random graph. *Random Structures & Algorithms*, 15(2):145–164, 1999.

Mathew D. Penrose. *Random geometric graphs*. Oxford University Press, 2003.

Mathew D. Penrose. Connectivity of soft random geometric graphs. *The Annals of Applied Probability*, 26(2):986–1028, 2016.

Andy D. Perkins and Michael A. Langston. Threshold selection in gene co-expression networks using spectral graph theory techniques. *BMC Bioinformatics*, 10(Suppl 11):S4, 2009. doi: 10.1186/1471-2105-10-S11-S4. URL https://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-10-S11-S4.

William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, Cambridge, England, 2nd edition, 1992. ISBN 0-521-43108-5.

Nilam Ram and Kevin J. Grimm. Growth mixture modeling: A method for identifying differences in longitudinal change among unobserved groups. *International Journal of Behavioral Development*, 33(6):565–576, 2009.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. 2006.

MIRA Rehab. https://www.mirarehab.com/.

Carrie Ritchie and Michele Sterling. Recovery pathways and prognosis after whiplash injury. *J Orthop Sports Phys Ther*, 46(10):851–861, 2016.

Christian P. Robert and George Casella. *Monte Carlo Statistical Methods*. Springer, 2004.

L. C. G. Rogers and David Williams. *Diffusions, Markov Processes, and Martingales: Volume 1, Foundations*. Cambridge University Press, Cambridge, 2000. ISBN 978-1-107-71749-7.

Colin F. Royse, Stanton Newman, Frances Chung, Jan Stygall, Rachel E. McKay, Joachim Boldt, Frederique S. Servin, Ignacio Hurtado, Raafat Hannallah, Buwei Yu, and David J. Wilkinson. Development and feasibility of a scale to assess postoperative recovery: The post-operative quality recovery scale. *Anesthesiology*, 113:892–905, 2010.

Reuven Y. Rubinstein and Dirk P. Kroese. *Simulation and the Monte Carlo Method*. Wiley, 2007.

Haneen T. Salah, Ibrahim N. Muhsen, Mohamed E. Salama, T. Owaidah, and Shahrukh K. Hashmi. Machine learning applications in the diagnosis of leukemia: Current trends and future directions. *International Journal of Laboratory Hematology*, 41(6):717–725, 2019. doi: 10.1111/ijlh.13089.

Abdulwahed Salam and Abdelaaziz El Hibaoui. Power Consumption of Tetouan City. UCI Machine Learning Repository, 2018. DOI: https://doi.org/10.24432/C5B034.

Chris Sander and Reinhard Schneider. Database of homology-derived protein structures and the structural meaning of sequence alignment. *Proteins: Structure, Function, and Bioinformatics*, 9(1):56–68, 1991.

Hannes Schabauer, Christoph Pacher, Andrew G. Sunderland, and Wilfried N. Gansterer. Toward a parallel solver for generalized complex symmetric eigenvalue problems. *Procedia Computer Science*, 1(1):437–445, 2010. ISSN 1877-0509. doi: 10.1016/j.procs.2010.04.047.

Brenda A Schulman, Andrea C Carrano, Philip D Jeffrey, Zachary Bowen, Elspeth RE Kinnucan, Michael S Finnin, Stephen J Elledge, J Wade Harper, Michele Pagano, and Nikola P Pavletich. Insights into scf ubiquitin ligases from the structure of the skp1–skp2 complex. *Nature*, 408(6810):381–386, 2000.

B. Schweizer and A. Sklar. *Probabilistic Metric Spaces*. Dover Publications, 2011.

Juliane Schäfer and Korbinian Strimmer. A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. *Statistical Applications in Genetics and Molecular Biology*, 4(1):Article 32, 2005. doi: 10.2202/1544-6115.1175.

N. C. Shah and S. Shenoy. Can defibrotide prophylaxis prevent sinusoidal obstruction syndrome following haematopoietic stem-cell transplantation? *The Lancet Haematology*, 10(5):e309–e311, 2023. doi: 10.1016/S2352-3026(23)00065-0.

P S Shenkin, B Erman, and L D Mastrandrea. Information-theoretical entropy as a measure of sequence variability. *Proteins*, 11(4):297–313, 1991. doi: 10.1002/prot.340110408.

Peter A. Stark, Paul S. Myles, and Justin A. Burke. Development and psychometric evaluation of a postoperative quality of recovery score: The QoR-15. *Anesthesiology*, 118:1332–1340, 2013.

Ji Guo Su, Xian Jin Xu, Chun Hua Li, Wei Zu Chen, and Cun Xin Wang. Identification of key residues for protein conformational transition using elastic network model. *The Journal of Chemical Physics*, 135(17):174101, 11 2011.

Nicholas Theis, Jonathan Rubin, Joshua Cape, Satish Iyengar, and Konasale M. Prasad. Threshold selection for brain connectomes. *Brain Connectivity*, 13(7): 441–455, 2023. doi: 10.1089/brain.2022.0082. URL https://www.liebertpub.com/doi/10.1089/brain.2022.0082.

Boris Thibert, Dale E Bredesen, and Gabriel del Rio. Improved prediction of critical residues for protein function based on network and phylogenetic analyses. *BMC Bioinformatics*, 6(213), 2005. doi: https://doi.org/10.1186/1471-2105-6-213.

Lloyd N. Trefethen and David Bau. *Numerical Linear Algebra*. SIAM, Philadelphia, PA, 1997.

L. N. Vaserstein. Markov processes over denumerable products of spaces, describing large systems of automata. *Problemy Peredači Informacii*, 5(3):64–72, 1969.

Cédric Villani. *Optimal Transport: Old and New*. Springer, 2008.

Bill Waggener. *Pulse Code Modulation Techniques*. Springer, 1995. ISBN 978-0-442-01436-0.

David Walton. A review of the definitions of 'recovery' used in prognostic studies on whiplash using an icf framework. *Disabil Rehabil*, 31(12):943–957, 2009.

David M. Walton, Dan Krebs, Dianna Moulden, Peter Wade, Lenerdene Levesque, James Elliott, and Joy C. Macdermid. The traumatic injuries distress scale: A new tool that quantifies distress and has predictive validity with patient-reported outcomes. *Journal of Orthopaedic and Sports Physical Therapy*, 46(10):920–928, 2016.

David M. Walton, James M. Elliott, Joshua Lee, Mohamad Fakhereddin, and Wonjin Seo. Identification of clinically-useful cut scores of the traumatic injuries distress scale (TIDS) for predicting rate of recovery following musculoskeletal trauma. *PLoS ONE*, 16(3), 2021.

Bernard M. Waxman. Routing of multipoint connections. *IEEE Journal on Selected Areas in Communications*, 6(9):1617–1622, 1988. doi: 10.1109/49.12889.

Joe Whittaker. *Graphical Models in Applied Multivariate Statistics*. Wiley, the UK, 1st edition, 2009.

Darren Wilkinson. Metropolis hastings mcmc when the proposal and target have differing support. *[Blog]Darren Wilkinson's blog, https://darrenjw.wordpress.com/*, 2012.

Chuqiao Zhang, Crina Grosan, and Dalia Chakrabarty. Individualised recovery trajectories of patients with impeded mobility, using distance between probability distributions of learnt graphs. *Artificial Intelligence in Medicine*, 157:103005, 2024. ISSN 0933-3657. doi: https://doi.org/10.1016/j.artmed.2024.103005.

Chuqiao Zhang, Sarath Chandra Dantu, Debarghya Mitra, and Dalia Chakrabarty. Identifying critical residues of a protein using meaningfully-thresholded random geometric graphs, 2025. URL https://arxiv.org/abs/2506.10015.