A mathematical framework for understanding perceptual bistability

Arnau Parrilla Gibert

A thesis presented for the degree of
Computational Mathematics and Data Analytics

Supervised by Alexandre Hyafil

Universitat Autònoma de Barcelona

Centre de Recerca Matemàtica

June 2022

Contents

Abstract

The study of human perception is still very unexplored and could contribute significantly to the understanding of the brain, especially for having more tools to fight against mental illness. Here I have used the framework of statistical inference to pursuit a better understanding of how perception takes place in the human brain. More precisely I studied if bistable perception can be understood using a probabilistic graphical model (PGM) with an algorithm to approximate the inference process on this PGM and see which algorithm betters adjusts the brain's computations.

In this work we have concretely focused on the perception of bistable stimulus and study them through the behaviour of Gibbs Sampling and Mean Field algorithms in a PGM that represents this stimuli. These are two sampling methods that could recreate the brains perceptual computations due to having behaviors very similar to those we would expect to see in a human being.

An other focal point in this work has been on the proper modelling that takes into consideration all the hypothesis and theoretical frame that are behind it. Setting a proper framework with which to work in the future.

The studies along this line could help to the understanding of certain mental illnesses like schizophrenia, where psychotic episodes have been linked to faulty perceptual inference.

Keywords: neuroscience; bistable perception; inferential approximation; sampling algorithms; Necker cube

1 Introduction

At least since 1925 there has been the idea that the brain performs statistical inference. The first to develop this idea was Helmholtz in 1925 and it has been studied since then.

Statistical inference is the process of analysing data to infer properties of an underlying probability distribution [1]. The idea is to infer properties of a population applying methods that analyze the sample data in order to estimate population parameters. The basic and most important assumption in statistical inference is that each individual within the population of interest has the same probability of being included in a specific sample.

Still the fundamental idea behind our brain's learning process is weighting the impact of new evidence. This is what is known as Bayesian Inference.

Bayesian inference is a method of statistical inference in which Bayes' theorem is used for updating the probability of a hypothesis as more evidence or information becomes available.

The Bayes' theorem is:

$$P(H/E) = \frac{P(E/H)P(H)}{P(E)} \tag{1}$$

Where in bayesian inference we denote [2]:

1. P(H), the *prior probability*, is the estimated probability before the new evidence is obtained E. Where H is the hypothesis whose probability might be affected by the data.

2. P(H/E), the *posterior probability*, is the probability of the hypothesis after we have observed the evidence.

3. P(E/H), it gives the compatibility of the evidence E given the hypothesis H. Where H is known as the *likelihood*.

4. P(E), this factor often known as integrated marginal likelihood or marginal likelihood function, and is the same for all the possible hypothesis. Its also often referred as the *model evidence*.

It is important to note that the posterior probability P(H/E) only depends on the numerator factors P(H/E) and P(H). Therefore the posterior probability is proportional to its prior probability (*i.e.* its inherent likeliness) and in a second recompilation of new data P(H/E) will become our newly acquired likelihood when taking into account newer evidence.

This is a very coherent model for understanding how the brain could update its belief of certain events depending on the cumulative sensory information (its important to note that the model of the world is already known). In Figure 1 we can see an example.

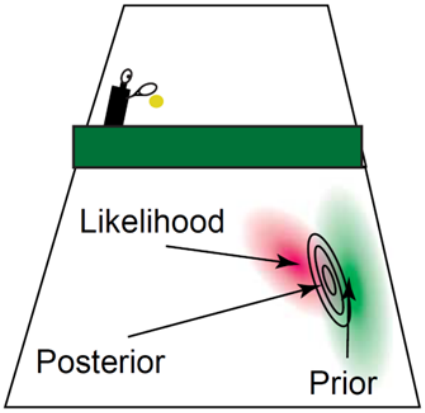

Figure 1: This image represents the idea behind the brain's Bayesian inference process. Having a prior probability $p(P) = \mathcal{N}(P, \mu_p, \epsilon^2)$ where μ_p is its mean and ϵ^2 the error for our normally distributed prior, that represents the initial idea of where the ball will land. A likelihood $p(V/P) = \mathcal{N}(P, \mu_v, \sigma^2)$ as a result of the visual information with its respective mean μ_v and variance σ^2. Finally the posterior $p(P/V)$ is where the hypothetical ball would land. [3]

Here we will focus on the understanding of visual perception. Visual perception is the ability to perceive our surroundings through the light that enters our eyes [4] and we will study it using probabilistic inference. We will derives the probability distribution of one or more random variables of having a specific value or set of values usually conditioned on some observations (the sensory information). Nonetheless, animals and humans are limited in order to perform properly this task perfectly (*i.e* they do not have infinite neurons), but still there are some observations that points into studying the brain in this direction and different hypothesis that would make models based on this idea viable.

Continuing along the same lines, the brain should have a model that tries to represent the structure of the world, specifically we hypothesized to be a probabilistic graphical model (PGM) and with it perform the inferential process.

A probabilistic graphical model (PGM) or structured probabilistic model is a probabilistic model where a graph expresses the conditional dependence structure between random variables [5]. For example the graph represented in Figure 2, A and B are two events that individually increase the probability of a third event C. They do not affect each other, thus $P(A/B) = P(A)$ and $P(B/A) = P(B)$ *i.e.* A and B are independent, this means that $P(A, B) = P(A)P(B)$. But if we observe that C occurs, if A also happens to occur ($A = 1$ and $C = 1$), then the probability of B is reduced since its positive relationship with C is less necessary to explain the occurrence of C and vice-versa with A. Now the two events A and B are conditionally negatively dependent, *i.e.* in general $P(A, B/C) \neq P(A/C)P(B/C)$.

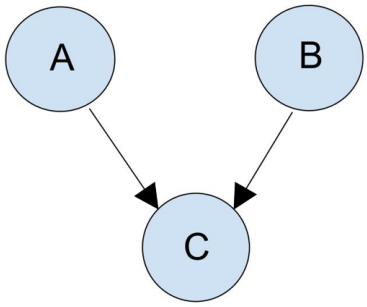

Figure 2: Probabilistic graphical model representing the hierarchical dependency between nodes.

With our PGM will have to take into consideration not only the properties perceived by our sense, since the ones that we can act upon usually are not directly observed. These are the hidden or latent variables. For example: when picking up an object, what we really perceive is an increasing tactile pressure in our hands and some images from the reflected light. But the real three dimensional properties of the object are never perceived (this is the latent part *i.e.* the hidden variable that we would have to infer).
As we have said taking into account all these variables would be impossible (we do not have enough neurons/computational power). Nonetheless, depending on the behavioural goal, lots of these variables (*i.e.* variables that represents all the properties of the object) are irrelevant. For example: when picking up a frying pan, the handle plays a very important role but the shadows are unimportant to perform this task. Thus we can discard the idea that the brain performs exact inference since the models that seems to represent would be computationally intractable. Even if it was feasible, we would need a lifetime to gain all the information that would allow us to make a complete statistical model.

For these reasons our brain has to use the "blessing of abstraction" [6] (*i.e.* make assumptions that will limit what can be represented, manipulate and learn). Moreover, all of the sensory information that our brain relies to perform inference has uncertainty, due to physical factors, neural noise ... All this is in agreement with the fact that the brain should perform inference on some kind of probabilistic model. We hypothesised to by a PGM used to approximate Bayesian Inference.

1.1 Encoding

The brain does not only obtain information and leave it untouched but transforms it in order to make it more useful. It is very important for the recording of this information not to be lineal, since the variables that are relevant for our task are usually mixed with the none relevant ones. For this reason our

PGM needs to be able to make nonlineaer embedings and thus separated those variable.
We also need nonlinearities to perform ethological tasks [1] since in order to separate behaviorally relevant variables from nuisance variables it is required to perform complex nonlinear computations.
To accomplish this purpose, it has been seen whit their success and the fact that they reproduce the hierarchical organization of the cortex. That the most efficient way to accomplish it is using a "deep" cascade of nonlinear transformations (Although theoretically it can be achieved by using just a simple network with only one layer of nonlinearities, since "this architecture is a universal function approximator for both feedforward nets and recurrent nets" [7][8]). This might be because parameters are easier to learn with the imposed architecture of deep models that matches better natural inputs [9].

Moreover, the relation between the stimulus and neural activity patterns has also to be encoded. Since probability distributions in the real world are very complex *Xaq Pitkow et al.* [6]) hypotheses that: "The brain simplifies the world by assuming that not every variable necessarily interacts with all other variables. Instead, there may be a small number of important interactions". These variables and their interactions can be represented by a graph (notice that this would be a sparsely connected graph, but still each variable could be connect to dozens/hundreds of other variables) and describe it mathematically as a probabilistic graphical model (PGM).

To recap, the idea that perception relies in some kind of probabilistic computations has become very popular over the recent years. But is very complicate to understand how cortical neural populations represents the real world and then performs these probabilistic operations.

Several theoretical work suggest that probabilistic representations are present from low-level sensory ares to high-level areas [10]. According to this view, the neural dynamics implements some form of probabilistic message passing (*i.e.* neural sampling, probabilistic population coding, etc.)

This is represented in Figure 3 where the external world is made of high-level features like objects and beings that gradually decompose into lower-level features. These are precisely the features that are represented by our senses. Then the sensory areas perceive the lowest-level features and the brain has to perform an inverse operation using its internal model of the exterior world *i.e.* the PGM to reconstruct more complex objects.

[1] Ethology is the scientific study of animal behaviour, usually with a focus on behaviour under natural conditions, and viewing behaviour as an evolutionarily adaptive trait.

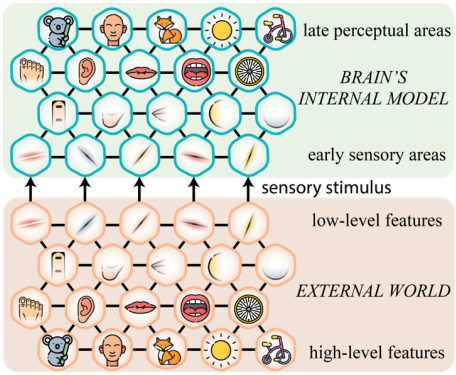

Figure 3: Probabilistic graphical model representing how the real world objects can be decomposed into lower level features (bottom part). This are the ones perceived by our senses with whom our brain has to reconstruct the object by using its own model of the world (top part). [10]

Here we will focus on the idea of probabilistic inference where all the representations throughout perception are probabilistic (they all represent probability distributions over some features, Figure 4). Thus, that the brain represents the external world as a PGM where the activity of the different neuronal populations (nodes) represents probability distributions of certain features and the different connections between them, encodes the statistical relationships between variables.

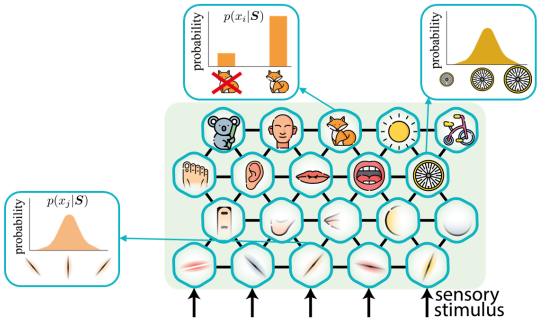

Figure 4: Representation that at each node of the graph representing a feature we associated a probability distribution, from lower level features to higher. This feature node is represented with a set of neurons (*i.e.* neural population that reacts to this task).[10]

The question of how neural activity can represent a probabilistic graphical model is explained because neurons with similar stimulus sensitivities spread the information with their spiking activity (*i.e.* a node in our PGM is a group of neurons that activates on the same stimuli), meaning that the probability distribution of a specific feature (node) is represented by a set of neurons that all react on

the same stimuli.

Therefore, with the right experiments we will be able to see how well our approach explains the brain's behaviour and neurophysiology. Will focus on probabilistic graphical models using the arguments of [6] mentioned before.
It is important to highlight that we will not focus on the question of how the brain learns good models, although it is fundamental. We will only focus on how the computations might look like once the PGM is learnt.

We therefore hypothesize:

1. The model is already learned and we will just focus on the inference part.
2. The brain makes some approximation in order to be able to make inference.
3. The brain's model uses a undirected probabilistic graphical model (undirected PGM).
4. Each node of the graph represents a probability distribution.
5. In our case the probability distribution of the nodes will represent different depths. For this reason our variables will have a binary domain which leads to the use of the Ising Model 3.1.

2 Message-Passing to use inference in a PGM

They are a group of algorithms that performs approximate inference in a probabilistic graphical model. Based on passing iteratively the information about probability distribution between the different nodes connected in the graph. They are called this way since the information that the nodes exchanges is defined as messages. More precisely Message-Passing algorithms are iterative algorithms that factorizes a global function of many variables into the product of simpler local functions. This will be the group of algorithms on which we will focus here and i future work (like mean field, loopy belief propagation ...). In figure 5 it can be seen how a message passing algorithm works in a PGM.

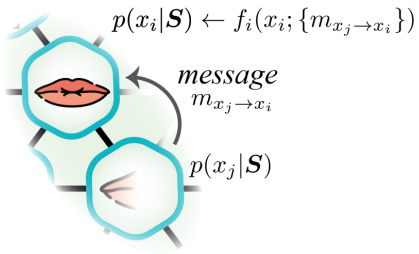

Figure 5: In order to compute the new posterior for a concrete feature its done by integrating all the messages of the different variables coming from neighbouring regions, then messages are passed throughout the graph until we have a coherent interpretation of what we are perceiving. In this image x_i is the node variable, S the new perceived data and $m_{x_j \to x_i}$ the new updated message that the node x_j sends to x_i

But as we have said and also commend in Section 3 exact inference is not feasible (it would be if we had a directed graph with tree structure 3) but since we have a PGM (*i.e.* undirected) exact inference is completely discarded.
For this reason we need to use algorithms that approximates the message passing procedure in our PGM *i.e.* approximate the Bayesian inference updating. For our PGM representing the exterior world and brain's model the message passing algorithm will have the form as in figure 6:

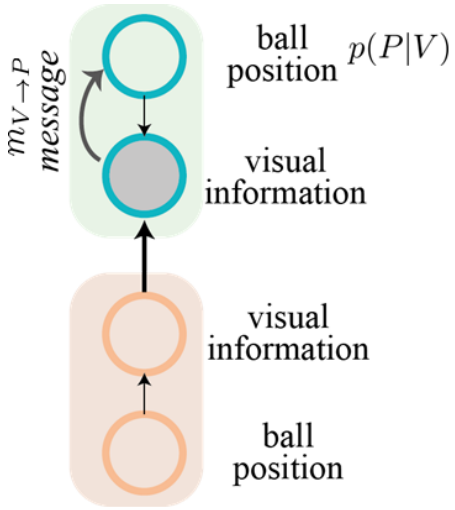

Figure 6: This figure represents how message passing works with the model structure defined in Figure 3. Where to infer the ball position the first layer nodes directly connected to the external stimuli passes the messages to the others (note that the real structure of the external world does not play any role). Where $P(P/V) = N(P; m, V)$ is the posterior probability and the message is defined as $m = \frac{\sigma^2}{\sigma^2+\epsilon^2}\mu_P + \frac{\epsilon^2}{\epsilon^2+\sigma^2}\mu_V$ noting that if the uncertainty of the likelihood increases it will also increase the weight of the prior.

This complex undirected graph/network updates its probabilities as more information is collected (i.e. new stimulus). This process is known as Bayesian updating.

What we will do is to study different Message-Passing algorithms and analyse how each of these models behaves and see if there are significant differences between them and whether is coherent for our brain to perform it. In future work this analysis will be compared with experimental data.

3 Mathematical framework

We want to represent the joint distribution $p(\vec{x}/\theta)$ of a specific state of the graph $\vec{x} = (x_1, x_2, ..., x_n)$ and use this distribution to infer one set of variables given another with the input data. In our specific case what is going to be represented by the graph is bistable stimuli.
Bistable stimuli is when we can perceive an object into two different configurations, for example the Necker cube 4.1. The idea is that depending on the previous bias and sensed data we will infer one state or the other. In this case $\vec{x}$ corresponds to the observed & latent data and θ the connection/relation between the observed nodes, such as if they are in different depth or attached to be in the same one.

As previously said in Section 1.1 we make the hypothesis that the model can be represented as an undirected graph following the network model in Figure 3. Since undirected graphs do not have any topological order (*i.e.* tree structure) we can not use the chain rule in order to compute the exact joint probability[11]. For this reason, first we will see how it would be in the case of having tree structure and then generalize it for undirected graphs. Thus the join probability is:

$$p(x_1, ..., x_n) = \prod_i p(x_i/x_{\pi_i}) \tag{2}$$

Where π_i denote the parents of the node i. We also need to assume that a node only depends on its immediate parents and not on all its predecessors in the ordering:

$$x_i \perp x_{\tilde{\pi}_i}/x_{\pi_i} \tag{3}$$

Nonetheless, we can not use the inference and learning methodologies of directed graphical models (*i.e.* exact inference), we only have exact inference algorithms when the PGM has tree structure, which is not the case here because we assume that our representation of the world include many complex interactions between variables.
In an undirected graph, the equivalent idea of conditioning with their parents is to condition against the adjacent nodes (in a directed graph we condition against the parent nodes as they are the ones from whom we received information. This for an undirected graph corresponds to conditioning against all neighbouring nodes since every node receive information from all of them). For example in Figure 7, would be $x_i/x_x, x_y, x_k, x_z$.

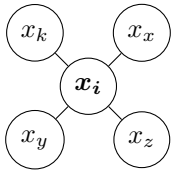

Figure 7: Simple MRF where x_i is connected *i.e.* shares information with all its adjacent nodes.

This corresponds to the idea of using maximal cliques to represent the relationship between nodes. Allowing us to perform the inference and learning process with an undirected graphical model.

Definition 3.1 *A clique is a subset of vertices of an undirected graph such that every two distinct vertices in the clique are adjacent[12].*

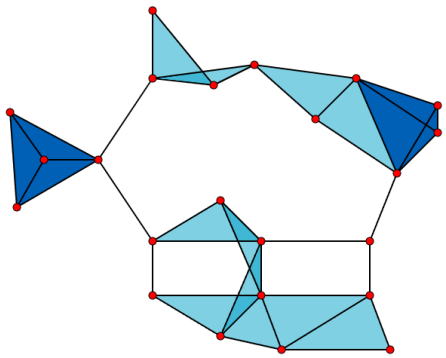

Figure 8: The 11 light blue triangles form maximal cliques. The two dark blue 4-cliques are both maximum and maximal.[13]

Definition 3.2 *A maximal clique is a clique that cannot be extended by including one more adjacent vertex, meaning it is not a subset of a larger clique. A maximum clique (i.e.., clique of largest size in a given graph) is therefore always maximal, but the converse does not hold.[14]*

This probabilistic graphical models theory where we are building up on, is based on the theory form [15].

Now we will redefine the inferential process in our undirected graph structure. Instead of associating conditional probabilities to each node, we associate potential functions or factors to each maximal clique in the graph. These new tool we allow us to compute the joint probability but now using the clique potentials. We denote the potential function for a clique c by $\Psi_c(x_c/\theta_c)$. For example in Figure 9 the maximal cliques would be $\{1,2,3\},\{2,3,4\},\{3,5\}$

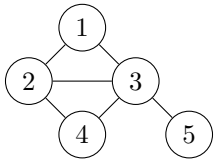

Figure 9: Representation of a UGM with 3 maximal cliques.

Then the joint distribution is proportional to the product of the clique potentials.

$$p(x/\theta) \propto \prod_{c \in C} \psi_c(x_c/\theta_c) \tag{4}$$

In the case of Figure 9 would be:

$$p(x/\theta) \propto \psi_{123}(x_1, x_2, x_3)\psi_{234}(x_2, x_3, x_4)\psi_{35}(x_3, x_5)$$

Basically we associate a potential function for each clique c with $\Psi_c(x_c/\theta_c)$ and compute the joint probability by performing the product of the potentials, finally normalize them.
For example in Figure 9 $p(x/\theta) = \frac{1}{Z(\theta)}\psi_{123}(x_1, x_2, x_3)\psi_{234}(x_2, x_3, x_4)\psi_{35}(x_3, x_5))$.
Where $Z(\theta)$ is the result of summing $p(x/\theta)$ over all the different possible samples (*i.e.* if our fives x_i would have a binary domain all the different samples that we could have would be 2^5. Notice that obtaining the normalization constant is usually computationally very costly).
This concepts are formalized by the Hammersly-Clifford Theorem.

Theorem 1 (Hammersly-Clifford) *A positive distribution $p(x) > 0$ satisfies the conditionally independence CI properties of an undirected graph G iff p can be represented as a product of factors, one per maximal clique, i.e.:*

$$p(x/\theta) = \frac{1}{Z(\theta)} \prod_{c \in C} \psi_c(x_c/\theta_c)$$

where C is the set of all the maximal cliques of G, and $Z(\theta)$ is the partition function given by:

$$Z(\theta) = \sum_{x} \prod_{c \in C} \psi_c(x_c/\theta_c)$$

Now we can restrict the parameterization to the edges of the graph since we have an undirected graphical model, also known as Markov Random Field (MRF). Thus it meets the Pairwise Markov property that given an undirected graph $G = (V, E)$, a set of random variables $X = (X_v)_{v \in V}$ indexed by V one of the necessary condition in order to form a Markov Random Field with respect to

G is to satisfy the Pairwise Markov property saying that any two non-adjacent variables are conditionally independent given all other variables:

$$X_u \perp X_v | X_{V \setminus \{u,v\}}$$

This implies that we can restrict the parameterization to the edges of the graph, rather than the maximal cliques (*i.e.* a pairwise MRF). In Figure 9, we get:

$$p(x/\theta) \propto \psi_{12}(x_1, x_2)\psi_{13}(x_1, x_3)\psi_{23}(x_2, x_3)\psi_{24}(x_2, x_4)\psi_{34}(x_3, x_4)\psi_{35}(x_3, x_5)$$

Using a pairwise MRF, we express the probability:

$$p(x/\theta) \propto \prod_{s \sim t} \psi_{st}(x_s, x_t) \tag{5}$$

Where $s \sim t$ denotes all the pairs connected by an edge. Multiplied by the normalization constant $\frac{1}{Z(\theta)}$. Which is obtained by marginalizing over all the variables.

We have defined how to compute the joint probability $p(x/\theta)$ for an undirected graph. Defining a potential function for each clique *i.e.* a set of nodes that are fully connected between them. Assigning them a function ψ_c that enables us to work as if they are just one node.
Now we need to define the potential function that we are going to use. Since one of the hypothesis that we made is that our variables will have a binary domain and we are only interested into modelling two different perceived depths. This leads us to the Ising Model.

3.1 Ising model

In our case of studying the Necker cube perception 4.1 we will model which face is perceived to be at the front, more precisely the depth of each vertex. For this reason the values for the nodes of the cube will be $x_i \in \{-1, 1\}$ representing if the node is perceived to be at the bottom or at front.
This brings us to the Ising Model. The Ising Model is an example of a MRF with their variables x_i having a binary set of possible values. It arose from statistical physics and was originally used for modelling the behaviour of magnets. Specifically ferro-magnet and anti-ferromagnet materials. For example, in ferro-magnetism the spins of atoms tend to align in the same direction (*i.e.* the coupling between them leads us to this outcome) in order to be able to simulate that, the model consists of discrete variables that represent magnetic dipole moments of atomic "spins" that can be in one of two states (+1 or -1). Then the idea is to see into which state $\vec{x}$ do they converge. This also can be applied to model the perceived depth of individual nodes in the Necker cube.

To do so, first we create a graph in a 2D or 3D grid, then connect the neighboring variables and define the following pairwise clique potential:

$$\psi_{st}(y_s, y_t) = \begin{pmatrix} e^{w_{st}} & e^{-w_{st}} \\ e^{-w_{st}} & e^{w_{st}} \end{pmatrix} \tag{6}$$

Note: Since the main use of the Ising model has been in physics we use the standard vector notation.

Where w_{st} is the coupling strength between the nodes s and t. If we set $w_{st} = 0$ this means that the two nodes are not connected in the graph. We also assume that the weight matrix M is symmetric (*i.e.* $w_{st} = w_{ts}$), meaning that the coupling strength between two nodes is always the same (note that we work with a factorial graph and do not differentiate any direction for the connection between two nodes). We will also assume that the connections between two nodes always have the same strength $w_{st} = J \geq 0$ where J represent their coupling. In order to have some intuition on how the model works we will describe the two main cases of the magnets model.

For the case of $J > 0$, this means that all the weights are positive. Then the spins are likely to be in the same state as their neighbours, this corresponds to ferromagnet materials. If the weights are sufficiently strong, the corresponding probability distribution will have two modes, the state with all $+1's$ and the state with $-1's$. This are called the ground states of the system and the same idea can be applied for studding the depths of nodes.
Therefore, we assume that the brain represents the Necker cube as an Ising Model with positive coupling (*i.e.* the nodes are connected by $J > 0$). Depending on the coupling strength J the Ising Model can converges into two different ground states representing the two possible perceptions of the object. We are

interested in knowing with which values of J this is more probable to happen. Note that if the model does not converge into any ground state (*i.e.* the probability is distributed homogeneously) we would see a plain image, what we say it to be "proper perception of the object" *i.e.* without bistability.

We can write the unnormalized log probability of the Ising model:

$$\log p(x) = J \sum_{i<j} M_{ij} x_i x_j + CONST \tag{7}$$

$$\log p(x) = \frac{J}{2} \sum_{i,j} M_{ij} x_i x_j = \frac{J}{2} x^t M x + CONST \tag{8}$$

Where $x_i \in \{-1, 1\}$. The factor $\frac{1}{2}$ appears because we sum each edge twice. The potential function ψ_{st} conveniently uses an exponential function. This allows us to readjust the product $p(y/\theta) \propto \prod_{c \in C} \psi_c(y_c/\theta_c)$ into a summation by taking the logarithm (note that ψ_{st} is represented by $\exp(M_{ij} x_i x_j)$ being M the matrix that contains all the different couplings w_{st} between nodes). Using the result of equation 8 and the exponential function to remove the logarithm we get $p(x)$:

$$\begin{aligned} p(x) =& \frac{1}{Z} \exp(\frac{J}{2} x^t M x) \\ & or \\ p(x) =& \frac{1}{Z} \exp(\frac{J}{2} \sum_{i,j} M_{ij} x_i x_j) \end{aligned} \tag{9}$$

Where $\theta = M$ and the normalizing constant Z is the result of summing $p(x) \propto \exp(\frac{J}{2} x^t M x)$ over all 2^D vectors *i.e.* the potential functions of all the states that we have (where D is the dimension of our vectors).

4 Modelling a PGM for a bistable stimuli

We want to have a PGM that allows us to represent how the brain models the external world (*i.e.* approximates the exterior world distribution). Therefore, we want to find which models are the most likely to take place in our brain. For this reason, we need to use some experiments that will enable us to distinct between models and see how well do they adjust the brain's computations. We used the case of bistable perception since we have some intuitions on how do we perceive these cases where bistability appears. Thus we will compere the different results using most known object for this case being that the Necker cube.

4.1 The Necker cube

The Necker cube is an optical illusion published by Louis Albert Necker in 1832. Its a simple cube with lines without orientation where we can interpret that the frontal face is either the upper right side or the lower left side (see Figure 10).

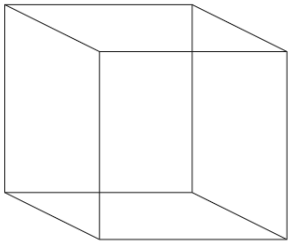

Figure 10: Representation of the Necker cube [13]

In the PGM of the Necker cube we will use different algorithms to approximate the posterior distribution of how do we perceived. All of these algorithm are dynamical algorithms. Therefore it will allow us to study their dynamics like the evolution of the model, the number of iterations to obtain a stable posterior approximation, how and when they transition to one state to an other, the stability of the different posteriors ... Then compare these results with human experimental data and ideally have some algorithm that would seem to point in the good direction regarding the understanding of the brain's computations *i.e.* knowing how to approximate some brain process. This could be useful to apply it in the study of certain diseases. For example, in schizophrenia, where psychotic episodes have been linked to faulty perceptual inference [16].

4.2 Modelling the perceptual inference of the Necker cube

In this section the idea is to see how the real world model, the perceived stimulus and brain's model relates to each other as mentioned in Figure 3 (*i.e.* the 3D representation of the world that the brain tries to infer). Then we will simplify

it to have a PGM that enables us to work easier with the Ising Model. Still this previous contextualization will allow us to see what each variable represents and have a better global understanding.

The perception model is represented in Figure 11 where the left cube corresponds to the real world model and the factorial graph to the observed inputted stimulus and brain's own internal model. In our factorial graph, we have two types of nodes. Unobserved nodes that correspond to features in the 3D space represented by empty nodes *i.e.* our latent variables (depth of nodes) and the observed nodes that correspond to features of the sensory stimuli (*i.e.* image features, represented by grey nodes). Each empty node, represents a probability distribution of the corresponding feature, *i.e.* a distribution that corresponds to the depth of the point in the (unobserved) 3D space. Finally the empty square nodes (*i.e.* factors linking the three elements) correspond to the probabilistic association between nodes. For example, if we observe a horizontal line, it makes it very likely that the unobserved depths for the two points connected by this line is the same. On the contrary if we observe a vertical line between two nodes would meant that the two points are probably on different depth. If there is no factor connecting the two nodes we removes the association between them and their depth becomes independent. In Figure 11 we could also have a final scene node that represents the proportion of dots that appears to be on the front of the image (*i.e.* which side is the one that we visualize as being at the front of the cube).

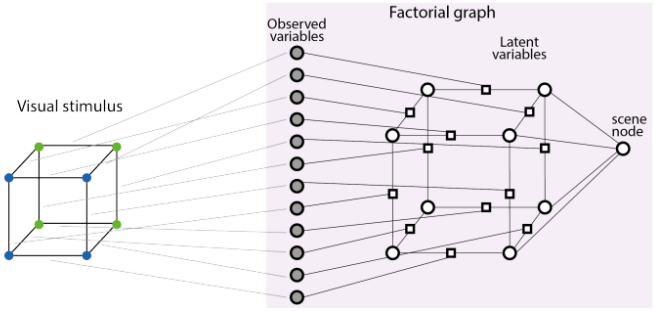

Figure 11: Factorial graph for the Necker cube

To model the state probability with equation 9, by default each node is equally probable to be at the front or back, thus we set $\theta_i = 0$ (*i.e.* this means that we do not have any previous bias). If two latent variables i and j corresponding to the cubes nodes, are connected by a horizontal line, their factor is $\theta_{ij} = J > 0$ (this means that they are likely to be on the same depth). Whereas when they are connected by a vertical line $\theta_{ij} = -J$ (likely to be on different depths). Finally if two nodes are not connected $\theta_{ij} = 0$.
We should also add more observed nodes (grey nodes) in order to represent the size that the nodes have on the image and with it the factor that connects these

observed variables to the corresponding latent node (empty nodes) (this factor is not represented in Figure 11 neither the observed nodes to avoid having a cluttering image). Still we would represent it by $\theta_{ij} = K > 0$, meaning that the larger the dot in the 2D image is more likely to correspond to a dot on the front in the underlying 3D structure. Nonetheless, this is only to recreate properly the perception framework, specially for future work to study the different effects of each element in the graph. Here we will just focus on the cube alone, but now being able to have in mind this lower level of abstraction.

5 Approximating the inference process in the Necker cube

After having defined our graphical model and its parameters, now we can see how to apply the different inference algorithms on it. As we have said, we want to analyse the performance of these algorithms on how do they obtain a probability distribution of the perceived state and whether this distribution is able to represent properly bistability. With it we are able to obtain the required number of iterations for the algorithms to have a stable posterior and compare their performance with human data experiments, seeing their differences and how do they compare to the data of real humans.

But first we will simplify the representation of our model since it will allow us to work better with our algorithms. Some modification of the previous model in Figure 11 have to be made in order to obtain an undirected graphical model that satisfies the properties that we require. Simplifying the cube's model by just having the 8 latent nodes with values $x_i \in \{-1, 1\}$ and the values of the edges that connects them, as seen in Figure 12. This structure allows us to define a vector state $\vec{x} \in \{-1, 1\}^8$. Thus we will just focus in the relationship between the latent variables. This much simplified representation of the PGM will enable us to work easier with the different algorithms and also ensure us that the PGM has the required properties that some algorithm like Gibbs requires as we will mention latter 5.2.

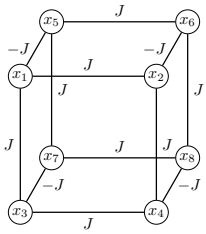

Figure 12: Graphical model that defines a state of the Necker cube, where $x_i \in \{1, -1\}$ and $\theta_{ij} = \pm J$

This can be more easily understood with an example, if we define a state with the configuration in Figure 13:

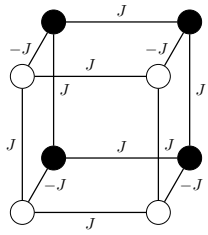

Figure 13: Cube representing a state where the black dots corresponds to $x_i = -1$ and white dots corresponds to $x_i = 1$ (*e.g.* bottom left face on top)

And by using the equation 9 the probability distribution of the state in Figure 13 would be:

$$\begin{gathered} \vec{a} = (1,1,1,1,-1,-1,-1,-1)^T \\ P(x = \vec{a}) = \frac{1}{Z}\exp(\sum \theta_{ij} x_i x_j) = \\ \frac{1}{Z}\exp(4J(1)(1) + 4J(-1)(-1) + 4(-J)(1)(-1)) = \frac{1}{Z}\exp(12J) \end{gathered} \tag{10}$$

5.1 Exact inference

Before analysing any of the algorithms, first we will study the true posterior of the different states of the cube. In this case we have only 256 states so the true posterior probability can be computed, but not for more complex models (this could also allow us to simplify a more complex model in the future).
When performing exact inference in the Necker cube, first we need to define which are going to be the different states, since we have eight different nodes and all of them have two possible values being $x_i \in \{1, -1\}$ this means that we have $2^8 = 256$ different states.
In order to simplify our model by making use of the different symmetries in the cube, we will redefine these 256 states gathering together the states where we can pass from one to the other using its symmetry. And regrouping them by the proportion of positive and negative nodes x_i (*i.e.* how meany nodes x_i we have in the cube with the same value, how many are $x_i = 1$ and $x_i = -1$) this is represented by μ defined as:

$$\begin{gathered} x_i \in \{1, -1\} \\ \mu = \sum_{i=1}^{4} x_i - \sum_{i=5}^{8} x_i = f(x) \end{gathered} \tag{11}$$

By using this definition we have the situation where the two opposite states that we are interested on (the one that corresponds on having one state in the front and the other in the back and vice-versa). Having the same number of nodes *i.e.* four $x_i = 1$ and $x_i = -1$, being these the two main perception for the cube's bistability with values $\mu = 8$ and $\mu = -8$. Instead of having 256 combinations of

$\vec{x}$ we have $\mu \in \{-8, -6, -4, -2, 0, 2, 4, 6, 8\}$ regrouping the cube states by their μ value.
Even though, for each value of μ we still have different cube configurations depending on how the x_i with the same sign are distributed, thus this will also have to distinct them. For this reason, in order to be able to use the potential that the symmetry of the cube has. Allowing us to regroup the states if they are symmetric, we will redefine its structure as follows:

$$\begin{aligned}
y_i &= x_i, \quad i \leq 4 \\
y_i &= -x_i, \quad i \geq 5 \\
\mathcal{V}_{ij} &= |\theta_{ij}| \\
p(\vec{x}) &= \exp(\sum_{i,j \leq 4} \theta_{ij} x_i x_j + \sum_{i,j \geq 5} \theta_{ij} x_i x_j + \sum_{\substack{i \leq 4, \\ j \geq 5}} \theta_{ij} x_i x_j) \\
p(\vec{x}) &= \exp(\sum_{i,j \leq 4} \mathcal{V}_{ij} y_i y_j + \sum_{i,j \geq 5} \mathcal{V}_{ij} (-y_i)(-y_j) + \sum_{\substack{i \leq 4, \\ j \geq 5}} -\mathcal{V}_{ij} y_i (-y_j))
\end{aligned} \tag{12}$$

$$p(\vec{x}) = \exp(\sum_{\substack{i,j \\ i \leq j}} \mathcal{V}_{ij} y_i y_j) \tag{13}$$

Before we have defined μ in equation 11 where $\mu \in \{-8, -6, -4, -2, 0, 2, 4, 6, 8\}$ but now we have:

$$\mu = \sum_{i=1}^{8} y_i \tag{14}$$

since $y_i = x_i, i \leq 4$ and $y_i = -x_i, i \geq 5$ implying that $\mu \in \{0, 2, 4, 6, 8\}$.
Notice that now our model does not separate the two main faces of the cube *i.e.* without distinguishing two different depth in the same image (all the connections $\mathcal{V}_{ij}$ are equal). Yet the general graph model still represents bistability since the cube can converge into two different extreme states (see Appendix A) that will model bistable perception. For this reason we are able to work using this higher level of abstraction.
Now we need to see how many different states of the cube we could have, regroup them and find the normalization constant. Then for each class we can define their exact probability formula, since the factor from the summation of equation 9 will always have the same value for all the states that are symmetric. We will denote $k_C = \sum_{i,j} \mathcal{V}_{i,j} y_i y_j = \vec{x}^t \theta_{ij} \vec{x}$ (note that all the states from a class will have the same probability *i.e.* the same k_C), thus we can define the exact probability as:

$$p(c) = \frac{\alpha_c}{Z} \exp(k_C J) \tag{15}$$

Where α_c is how many times the same pattern can appear in our cube *i.e.* how many states of the 256 are symmetric, implying that they are in the same class.

The cardinal of $x \in \{-1, 1\}^8$ that maps onto class C is discussed in much more detail in the Annex A.

All the connections between nodes x_i will have the same value (in our case $\mathcal{V}_{ij} = \theta_{ij} = J$) as previously seen in page 17. Being much easier to see the symmetry between states.

To visualise how the 256 cube states are represented with our new definition of equation 12 we will use the same indexes domain for x_i as in Figure 12 and the same notation to visualize $x_i = \pm 1$ as in Figure 13, where the black dots represents $x_i = -1$ and $x_i = 1$ for the white ones. To facilitate the step of computing the state probability with equation 9, a green line will represent that the connection between two nodes contributes with a positive value to the summation of equation 9 being $J = 1$ (this mean that the two neighbouring vertex have the same value $x_i x_j = +1$) and a red line will represent that the connection between the two nodes contributes negatively with $J = -1$ (neighbouring vertex with different values $x_i x_j = -1$).

We will separate our 256 states in 22 different classes where as previously said each value of μ corresponds in changing the value of a specific number of vertex. Then we will separate the ones with the same μ by how these vertex are distributed in the cube. The detailed regrouping with its arguments and analysis is found at Appendix A

We have 22 different classes but if we consider different the ones with the same symmetry and the opposite values of x_i in total we have a number of 14 classes, with all the states of the cube are defined. The states identifications for the 8 symmetric cube positions *i.e.* the same as the 8 first ones but with opposite values of x_i are designated in the reverse order of the 8 first. Counting from the 15 to the 21 state being 21 the other extreme state where all the nodes have the same value of x_i (but now being -1). Finally the normalization constant Z can also be computed by summing all the probabilities, also found in Annex A.

The graphics showing the exact probability distribution for the 22 states are in the Annex C.
By analysing these images we can conclude that when our Necker cube model begins to have some coupling bistability instantly emerge and increasing as the coupling also increases. At the beginning it does so very quickly, but rapidly stabilises.
As our model is not very complex we have been able to perfom exact inference in the different states, but for the case of a more complex PGM this would not be possible (for our computers and our brain). Therefore, we are now going to introduce a couple of algorithms that will help us to approximate the inference process and study more generally its behaviours and properties in order to be able to easily generalise our analysis with a more complex PGM for the Necker cube in the future.

5.2 Gibbs sampling

The Gibbs sampling algorithm is a Markov Chain Monte Carlo (MCMC) algorithm. MCMC algorithms are a class of algorithms that allow us to sample from a probability distribution constructing a Markov Chain using the dependencies between variables. Thus we need to sample from a proposal distribution that depends on the current state $\vec{x}^t$ (*i.e.* $q(\vec{x}/\vec{x}^t)$), then the sequence of samples $\vec{x}^1, ..., \vec{x}^n$ forms a Markov Chain. A Markov chain or Markov process is a stochastic model describing a sequence of possible events in which the probability of each event depends only on the state attained in the previous event. There are two properties of a Markov Chain that if are satisfied the sample sequence can reach a unique invariant distribution. These are:

- Irreducible: a finite Markov Chain is irreductible if it only has one class. In this context a class is a set of nodes where we can transition from one to the other (*i.e.* there is a path that always connects two nodes), this is the case with the cube's undirected graph. Thus the Necker cube can be considered of just having one class.
- Aperiodic: if the states that forms the chain are aperiodic *i.e.* there is a path of k steps that begins in the class i and ends also at i. It is straightforward to see that our cube's graph is also aperiodic.

Thus, since we have an aperiodic and irreducible Markov Chain the theorem from [17] guarantees us the existence and uniqueness of an invariant distribution.

Invariant distribution: An invariant distribution of a MC on the classes C = {1,2 ...,m} is a vector $\pi = (\pi_1, ..., \pi_m)$ with no negative components that:

$$P\pi = \pi \quad \text{and} \quad \sum_{i \in S} \pi_i = 1 \tag{16}$$

With MCMC algorithms we can sample from a large distributions and they also scale well with the dimensionality of the sample space. Its is important to note that successive samples are highly correlated, so if we want to obtain and independent sample we can discard most of the sequence and just retain every M sample (if M is large enough the samples will be independent).

Another element to take into account is under which circumstances the algorithm will converge to the desired distribution. Basically a Markov Chain has to satisfy two properties:

- Ergodicity that is the require that for $m \to \infty$, the distribution $p(\vec{x}^m)$ converges always to the same invariant distribution, irrespective of the chosen initial distribution $p(\vec{x}^0)$.

- A distribution is invariant if at each step it remains constant. If we have and invariant distribution we call it equilibrium, thus the second propriety is that the Chain has an equilibrium state.

Therefore the Markov Chain that defines our algorithm is an undirected graphical model that connects the different states and meets these properties.
The hypothesis is that the brain uses Gibbs on states, not classes. But we will use the 22 classes defined above to study the Gibbs dynamics.
To see that the distribution is invariant, first we define that a chain would homogeneous when the transitional probabilities have the same values for all m, now with this definition we can see that:

If we define the transition probabilities as:

$$T_m(\vec{x}^m, \vec{x}^{m+1}) \equiv p(\vec{x}^{m+1}|\vec{x}^m) \tag{17}$$

If the transition probabilities are the same for all m (homogeneous Markov Chain):

$$p(\vec{x}^{(m+1)}) = \sum_{\vec{x}^{(m)}} p(\vec{x}^{(m+1)}|\vec{x}^{(m)})p(\vec{x}^m) \tag{18}$$

Then the converged distribution $p^*(z)$ is invariant *i.e.* once the algorithm reach it, more iterations will not change $p^*(z)$:

$$p^*(\vec{x}) = \sum_{\vec{x}'} T(\vec{x}', \vec{x})p^*(\vec{x}') \tag{19}$$

5.2.1 Gibbs sampling algorithm

Gibbs is a form of MCMC that runs on the probability distributions defined by a PGM. It consists on choosing randomly an initial state and then iterate from that state. Replacing the value of one variable that we sample conditioning its distribution against the value of the others. The sampling can be done cycling through the variables choosing randomly the ones to be updated as in equation 22 or in a specific order like in algorithm 1. Here we have the general form of the algorithm from [15].

Algorithm 1 Gibbs sampling algorithm [15]

Require: Initial state $\{\vec{x}_i : i = 1, ..., M\}$
  for $t = 1, ..., T$: **do**
    Sample $\vec{x}_1^{t+1} \sim p(\vec{x}_1|\vec{x}_2^t, \vec{x}_3^t, ..., \vec{x}_M^t)$
    Sample $\vec{x}_2^{t+1} \sim p(\vec{x}_2|\vec{x}_1^{t+1}, \vec{x}_3^t, ..., \vec{x}_M^t)$
    ...
    Sample $\vec{x}_j^{t+1} \sim p(\vec{x}_j|\vec{x}_1^{t+1}, ..., \vec{x}_{j-1}^{t+1}, \vec{x}_{j+1}^t, ..., \vec{x}_M^t)$
    ...
    Sample $\vec{x}_M^{t+1} \sim p(\vec{x}_M|\vec{x}_1^{t+1}, \vec{x}_2^{t+1}, ..., \vec{x}_{M-1}^{t+1})$
  end for

In our case, with the 256, the Gibbs sampling algorithms is initialized in a random state/class of the cube $\vec{x}$ (*i.e.* assigning randomly values of x_i between $-1, 1$ for the eight nodes of the cube). Then choose randomly one of the nodes and with p_t the probability of the Ising Model 9 change its value or not.

$$x_{it}^{t+1}/x_{it}^{t} \sim Bin(x_{it}^{t+1}, p_t) \tag{20}$$

Where p_t is the normalized probability between the probabilities of changing or not the values of the nodes.

$$\begin{aligned} p_t(x_i) &\sim \exp(\frac{J}{2}\sum_{(i,j)}\theta_{ij}x_i x_j) \\ p_t(x_i) &\sim \exp(\frac{J}{2}\vec{x}^T\theta_{ij}\vec{x}) \\ k(x) &:= \frac{1}{2}\vec{x}^T\theta_{ij}\vec{x} \end{aligned} \tag{21}$$

Thus the probability for updating the value of x_i is:

$$\begin{aligned} p(x_i^{t+1} \neq x_i^t) &= \frac{\exp(Jk(x^{t+1}))}{\exp(Jk(x^{t+1}) + \exp(Jk(x^t))} = \frac{1}{1+\exp(-(k(x^t) - k(x^{t+1})))} \\ p(x_i^{t+1} \neq x_i^t) &= \sigma(J(k(x^{t+1}) - k(x^t))) \end{aligned} \tag{22}$$

This process is iterated n times, counting how many of it we pass through each class and finally normalize each class probability by dividing it with the total number of iterations. Note that the higher the value of k a state has, the more probable is this state. The algorithm is 2.

Algorithm 2 Necker's Gibbs sampling algorithm

Require: Initial state $\vec{x} \in \{-1, 1\}^8$

for $t = 1, ..., n$: **do**

- Choose x_i with $p = \frac{1}{8}$ the node to be changed.

- With p_t defined in equation 22 change the value of x_i or not.

- See in which of the 22 classes corresponds our new $\vec{x}$ and add 1 to the counter of this class.

- Divide all our class counters by the number of iterations n.

end for

Note: The first iterations are very noisy since they have a strongly dependence with the initialization value. For this reason the first iterations of the algorithm

are omitted, this is represented with the `burn_in` parameter.

This Gibbs simulation gives us the approximate posterior probabilities. The simulations results are found at Appendix D.
From the images in Appendix D we can see that as soon as the coupling starts to occur the probability concentrates very quickly at the extremes *i.e.* the cube's bistability will be much easier to appears. Since we have made a sampling of 100.000 iterations our results are practically identical to the exact inference. Thus our results do not show bistability since we have explored all the cube states, seeing all the possible perceptions of the cube. But this is very costly and probably the brain does not perform this large number of iterations. Then if we reduce it like in Appendix D, the bast majority of probability is found on one of the two extreme cases being these the only to be explored. Giving rise to bistability.

5.2.2 Gibbs analysis

Previously instead of using transition with the complete 254 states space, we have studied at the level of classes and now we can use it to obtain the transition matrix between them (*i.e.* the transition matrix of our Markov Chain). This will be a transition matrix of 22×22 where the transition probabilities will be the same as the one we have defined above in equation 22 but now using the exact value of k_C for each class (Annex A) when defining our σ function. To do so we multiple the probability of transitioning from a class to a new one *i.e.* $\sigma(k_C[j] - k_C[i])$ by how many of the eight nodes of the cube can give us the new class j from our current class i if we change its value, equation 23.

$$p(c'/c) = \sum_{x' \text{ in class } c'} p(x', x) = \sigma(J(k_{C'} - k_C)) \cdot \#(x' \text{ in class } c'/x \text{ in class } c) \tag{23}$$

This information is encoded in a matrix of 22×22 defined as C *i.e.* $C_{ij} = \#(x' \text{ in class } c'/x \text{ in class } c)$. In Annex B we can find C encoding the information that if we change the value of one node from a class to which new class will takes us. Counting how many times we can jump from one class i to another j. Within a class we can change any of the eight nodes of the cube, therefore from one class we have eight different transitions into new ones.

Then we construct the transition matrix as following:

$$T[i, j] = (\frac{C[i, j]}{8})\sigma(k_C[j] - k_C[i]) \tag{24}$$

Now that we have the transition matrix we can analyze the dynamics of our Gibbs sampling algorithm for a given J.
As we have previously said, we have an irreducible an aperiodic Markov Chain and the theorem [17] guaranties the existence and uniqueness of an invariant distribution. The previous theorem result's as a consequence of Perron Frobenius' Theorem for irreducible and aperiodic matrices with non-negative entries.

Which guarantees that 1 is a simple and dominant eigenvalue of the matrix T and that eigenvector with eigenvalue 1 has all the entries of the same sign. Therefore, to find the invariant distribution of T we only need to find the only eigenvector of eigenvalue 1 and normalize it so that the components are positive and sum to 1.

Using the transition matrix T, we can find the distribution for the $k-th$ iterations with:

$$p(x^k = x) = Tp(x^{k-1} = x) = T^l p(x^{k-l} = x) = ... = T^k p(x^0 = x) \tag{25}$$

Since we have the exact transition matrix T and $\forall x^0$ there is only one invariant distribution where our algorithm will "converge" *i.e.* a $k-th$ iteration where the samples will be independent, thus the distribution approximation does not change with further sampling. Being that the eigenvector with the largest eigenvalue of the diagonalized matrix D (with $\lambda = 1$). By noting at what speed the second largest eigenvector tends to zero (note that this will be $|\lambda| < 1$), we will have the number of iterations in which the algorithm gives us the invariant distribution (*i.e.* our sampling will be independent) corresponds to the only eigenvector $\mathbf{x}$ of eigenvalue one. This argument comes as the result of diagonalizing the matrix T and see that if we take the limit $k \to \infty$ only one eigenvector of the diagonal matrix D will remain (the one with $\lambda = 1$). The velocity at which the others tend to zero will be faster, the smaller is their value:

$$\begin{aligned} T &= R^T D R \\ T^k &= R^T D^k R \\ T^k &\xrightarrow{k\to\infty} R^T D^k R \end{aligned} \tag{26}$$

Due to our transition matrix depending on the value of J. We can analyse the convergence time depending of its value (determined by the second largest λ). Defining τ as the $\lambda^k = \exp(\frac{-k}{\tau})$, this represents how λ tends towards zero as a function of k, rewriting it we obtain $\tau = -k \cdot \log(\lambda)$ that give us the order of how many iterations our sampling needs be independent *i.e.* converges to the desired distribution. We can also analyse the direct effect of J to the value of the second largest eigenvalue and the theoretical distribution to which it should converge with the normalized eigenvector.

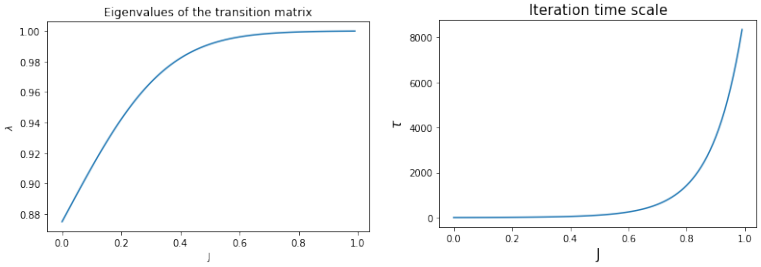

Figure 14: *Right*: plotting the value of the second larges λ for each J. *Left*: plot of the τ for the different values of J.

In the previous images 14 we see the directed effect that the value of J has on both the second larges λ and the theoretical number of iterations to converge. We see that when the coupling between nodes increases *i.e.* the value of J the required number of iterations to obtain a significant result increase exponentially. Specially between the values for $J \in [0.5, 1]$. This behaviour is easily interpreted when seeing the value of the corresponding second larges λ since for the same values of J where the number of iterations start to increase exponentially, they are practically equal to 1. Therefore needing a larger value of k in equation 26 to tend towards zero. So we conclude that as more coupling we have the more time it is required to converge.
From here we should see whether the convergence times for the Gibbs sampling algorithm are coherent enough to be implemented by the brain and how well our results adjust the experimental data. It seems that when the coupling is very strong (seeming to be the case the Necker plain image 10) the amount of iterations to explore all the posterior space is extremely costly and probably the brain does not perform an amount this large. Thus bistability is explained very naturally as the brain does not have enough time to explore all the posterior space.

5.3 Mean Field algorithm

The mean field algorithm is a variational message passing algorithm where we define a functional (a mapping that takes a function as the input and returns the value of that functional as the output) and express it as an optimization problem where the quantity to optimize will be the functional.

Since we have a Bayesian model with its latent variables Z and observed variables X as seen in section 1 and 3. We want to find and approximation for the model evidence $p(X)$ (very hard to compute it exactly since usually requires performing very costly integrals, in our model we are able to do so, but typically is not the case) and the posterior distribution p(Z/X). The log marginal probability $\log(p(X))$ can be decomposed as in equation 27, the arguments behind this outcome are explained in Chaper 10 from [15]:

$$\log(p(X)) = \mathcal{L}(q) + KL(q||p) \tag{27}$$

where

$$\begin{aligned} \mathcal{L} &= \int q(z) \log(\frac{p(X,Z)}{q(Z)}) dZ \\ KL(q||p) &= -\int q(z) \log(\frac{p(Z|X)}{q(Z)}) dZ \end{aligned} \tag{28}$$

The equations 28 are Kullback-Leibler Divergence score (KL) quantifies how much one probability distribution differs from another probability distribution. Here are represented for a continuous domain but in our case we will change the integrals for summations since we are working with discrete parameters. Although we have a discrete domain the true posterior distribution is normally still intractable. Thus we have to restrict the family of distributions $q(Z)$. We have previously assume when using our potential functions ψ. That Z partitions in disjoint groups Z_i, where $i = 1, ..., M$ and q factorizes with respect to these groups: $q(Z) = \prod_i^M q_i(Z_i)$. Then if we substitute q(Z) and minimize equation 27 what we obtain is $\log(q_j^*(Z_j)) = \mathbb{E}_{i \neq j}[\log(p(X,Z))] + CONST$ (this procedure can also be found in [15] Chapter 10). More generally for the case of a graph:

The joint distribution corresponding to an undirected graph

$$p(x) \propto \prod_{ij} \psi_{ij}(x_i, x_j) \tag{29}$$

$$\log(q_j^*(x_j)) = \mathbb{E}_{i \neq j}[\sum_i \log(\psi_{ij}(x_i, x_j))] + CONST \tag{30}$$

In our case the probability of changing the value at one of the nodes is q_j. If we readjust equation 30 for our specific case of the Necker's Ising model we get:

$$\begin{aligned} \log(q_j(x_j)) &= \sum_{i \in N(j)} \sum_{x_i \in -1,1} \log(\psi_{ij}(x_i, x_j) q_i(x_i)) + CONST \\ p(x) &\propto \exp(\sum_{i,j} \theta_{ij} x_i x_j) \\ \log(q_j(x_j)) &= \sum_{i \in N(j)} \sum_{x_i \in -1,1} \log(\psi_{ij}(x_i, x_j) q_i(x_i) + CONST \\ &= x_j J \sum_{i \in N(j)} \sum_{x_i \in -1,1} x_i q_i(x_i) + CONST \end{aligned} \tag{31}$$

where for $x_i \in \{-1, 1\}$

$$q(x_i = 1) - q(x_i = -1) = 2q(x_i = 1) - 1$$

then obtaining

$$\log(q_j(x_j)) = x_j J \sum_{i \in N(j)} (2q(x_i = 1) - 1) + CONST \tag{32}$$

$$q_j(x_j = 1) = \frac{\exp(J \sum_{i \in N(j)} 2q(x_i = 1) - 1)}{\exp(J \sum_{i \in N(j)} 2q(x_i = 1) - 1) + \exp(-J \sum_{i \in N(j)} 2q(x_i = 1) - 1)} \tag{33}$$

$$q_j(x_j = 1) = \frac{1}{1 + \exp(-2J \sum_{i \in N(j)} 2q(x_i = 1) - 1)} \tag{34}$$

$$q_j(x_j = 1) = \sigma(2J \sum_{i \in N(j)} 2q(x_i = 1) - 1) \tag{35}$$

Note: That the value of q_j is now the probability for the node j of being equal to one.

This equation is use recursively until the distribution q converges. First we set randomly the value for all $q_j \in [0, 1]$ and then we change the value of each node n times using equation 35. Algorithm 3

Algorithm 3 Mean Field algorithm

Require: Initial state $\vec{q} = {x_i}^8$ where $x_i \in [0, 1]$

for $t = 1, ..., n$: **do**
- Change with equation 35 the value of each q_i

end for

The results of the simulations are in Annex E.

5.3.1 Mean Field algorithm analysis

When analysing the Mean Field algorithm through equation 35 we are going see where our dynamical system series converges. It is important to realise that the values $\forall q_i$ will be the same since all the nodes in the cube are symmetric. Where q is the probability that the node has the value 1 and $1 - q$ for -1. ($q = q_i(x_i = 1) \forall i$). With that said, notice that equation 35 can be rewritten as:

$$q = \sigma(2J \cdot 3(2q - 1)) = \sigma(6J(2q - 1)) \tag{36}$$

This corresponds to a dynamical system of the form:

$$f(q, J) = \sigma(6J(2q - 1)) \tag{37}$$

Thus we can study their behaviour by analysing its fixed points [18] (*i.e.* when $f(x) = x$) and see whether they act as attractors or repulsors in our dynamical system. Since we have a one dimensional function this can easily be seen by plotting the function and see where it intersects with the line $f(x) = x$. Here we have some examples for different values of J:

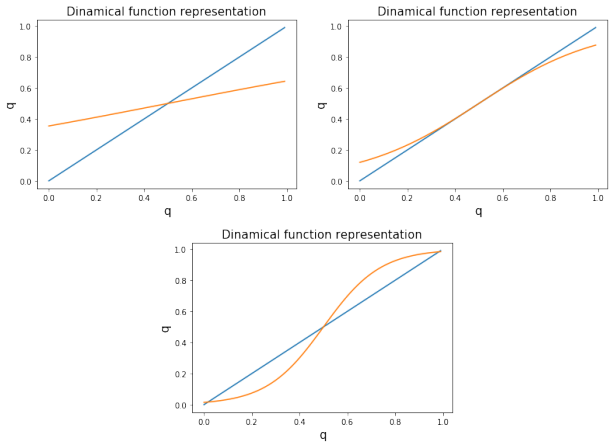

Figure 15: The *top-left* plot shows us the behaviour of our dynamical system for $J = 0.1$ where it can be seen that the point at $q = 0.5$ will act as an attractor node and our system will always gave us the same result. The *top-right* image shows the behaviour for $J = 0.33$ being that when the dynamical system has only one solution acting as an attractor. Finally the *bottom* image with $J = 0.7$ where there are three fix points the two on the extremes acting as attractors and the point for $q = 0.5$ repulsor.

Here we have a dynamical system $f(q, J)$ parameterized by J giving rise to a family of dynamical systems. It is clear that depending on the values of J the fix points and their behaviour changes as we can see in figure 16. Thus if for a specific value of J^* the fix point changes in number or typology, it is said that the system presents a bifurcation in J^*. In our case we have a different number of fixed point depending on the value of J but $q = 0.5$ is always a fixed point and is the one that corresponds to the true posterior $p(x_i)$.

To analyse this we have to look at the derivative of $f(q, J)$ [19]. We have a transcritical bifurcation [19] at $q = 0.5$ and $J^* = \frac{1}{3}$ since it meets the properties:

$$\begin{aligned} &F'_{J^*}(q^*) = 1 \quad \text{i} \quad F''_{J^*}(q^*) = 0 \quad \text{i} \qquad F'''_{J^*}(q^*) \neq 0 \\ &\partial_J \partial_q F(q^*, J^*) = \frac{\partial}{\partial_J}(\frac{\partial}{\partial_x}) \neq 0 \end{aligned} \tag{38}$$

At this point two branches of fixed points (one stable and the other unstable) intersect and exchange their character (the stable fixed point branch becomes unstable fixed points and vice versa). This value J^* is known as the critical value and in our case is $J = \frac{1}{3}$ as seen in equation 39.

$$\frac{\partial f}{\partial q} = 1$$
$$\frac{\partial f}{\partial q} - 1 = 0$$
$$12J \cdot \sigma(6J(2q-1))(1-\sigma(6J(2q-1))) - 1 = 0$$
$$12J \cdot \frac{1}{1+\exp(6J(2q-1))} \cdot \left(1 - \frac{1}{1+\exp(6J(2q-1))}\right) = 1$$

When $q = 0.5$ occurs the bifurcation since all properties mentioned are fulfilled 38

$$6J(1-\frac{1}{2}) = 0$$
$$J = \frac{1}{3} \tag{39}$$

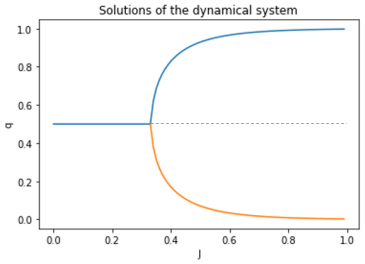

Figure 16: This plots shows the solutions that our dynamical system $f(q, J)$ depending on the value of J. Note that when $J \leq \frac{1}{3}$ we only have one stable solution being $q = 0.5$ *i.e.* we do not have bistability. But for larger values of J bistability appears having two solutions in or dynamical system, corresponding to the two different perceived depth in our cube (note that the two solutions always sums to 1)

If we analyse the mean field behaviour in the Necker cube. The "ideal" probability of q is $q = 0.5$ this means that the probability for a node of being on one side or another is the same. In our system $f(q, J)$ when $J \leq \frac{1}{3}$ there are two possible convergence points that will act as attractor nodes and give rise to bistability, the third point $q = 0.5$ will acts as a repulsor. However, for the value of $J \leq \frac{1}{3}$ the point $q = 0.5$ changes its behaviour acting as an attractor node. Therefore not being bias towards seeing one face or the other and acting as an attractor node, in this case we do not have bistability.

Note that $\forall q_i$ have the same value and for this reason this does not make any distinction between the depths of the two possible perceived faces since it takes into consideration the entire cube. But the behaviour still corresponds to the same idea. Thus we can also study the Necker using this model.

Therefore, for all the value of $J > \frac{1}{3}$ our system always has three fix points, the point $(0.5, 0.5)$ will act as a repulsor and the system depending on the initial value of q will converge to one of the two extremes. If $q_{init} > 0.5$ to the extreme more bias is towards $q = 1$ and vice versa for $q_{init} < 0.5$. If $J \leq \frac{1}{3}$ we have one fix point being $(0.5, 0.5)$ with attractor behaviour, implying that there is no place for bistability.
From here we should see with the experimental data whether this modelling of the Necker cube bistability is well adjusted by the Mean Field algorithm. If we would need to add more variables to the PGM or that with the results obtained the hypothesis for the brain using this algorithm is completely discarded.

6 Conclusions

Here we have seen the different hypotheses about the brain performing statistical inference and the coherent interpretation that are behind it. Why the brain could use some statistical method to approximate inference due to the difficulty to perform the perception process using exact inference (*i.e.* we do not have enough neurons) and how the exterior world information could structure in a model from higher to lower level features, being that the ones perceived by our sense. With which then the brain has to reconstruct the real world model using its own (remember that we have not focus on how the brain learned this model but on how would be its computations once it is learnt).

Then we have constructed the mathematical framework to model the PGM in coherence with our hypothesis and interpretations. Here we have specially focus on the proper understanding and modelling of the PGM and why we have concretely made these hypotheses. Due to the required length of the work, all the different modifications and extensions of the model to further study perception are left for future work. But will be studied using this framework.

Within this project we can conclude that with our PGM bistability seems to appear very naturally and always when the coupling strength between node increases, becoming very strong when the value of J gets larger. For experiments like the Necker cube, we as humans usually experiment bistability. Thus if the coupling of the Necker image is very large (its continuous lines seems to point into that direction) and our brain uses an algorithm similar to Gibbs sampling. The fact that bistability appears is explained because our brain presumably will not perform a large enough sampling in order to explore the entire posterior distribution since the Necker classes probabilities distribution is concentrated on the two extreme cases as we have seen in exact inference and in our simulations. Therefore, with a smaller amount of iterations we can not explore the entire posterior space (*i.e.* all the classes) probably getting stuck into one of the two base states. This is what gives rise to bistability. In future work we should see with experimental data if depending on the time that a subject sees an image

the perception of the cube changes *i.e.* the inferential process continues. Then compare the results with a Gibbs simulation.
As for the Mean Field algorithm, it has a different behaviour than Gibbs where the probability for each node on being in one face or the other is what represents bistability. If they are aligned on one bias state (*i.e.* not being equally probable to be on both faces $q_i \neq 0.5$).
The Mean Field algorithm converges into a stable distribution faster than Gibbs. Working well for cases where we have a very light coupling (not presenting bistability), but when J increases an passes its critical value bistability appears no matter number of iterations. We should see with experimental data if we also experience bistability basically depending on the coupling between variables and if we have some kind of critical value for it. Or we are in the other case where the brain performs an other kind of algorithm where we do not have sufficient time to perform properly the inferential process.

In the future work we will have to compare them with different experimental data and make some modifications on the algorithm to see which one adjust better our brain's behaviour (like adding some bias towards ones of the two states *i.e.* $\theta_i \neq 0$). In addition, it will be necessary to increase the complexity of the PGM in order to study the weight and importance of other variables like the size of the perceived nodes, the effects that previous biases could have or how we could make our model more accurate by also simulating neuronal noise. As well as study bistability with a larger range of experiments that will allow us to generalise and give weight to our observation. Taking into consideration new characteristics and phenomenons of perception.

Most importantly, future work should focus on how our models and algorithms adjust the experimental data. Comparing the results, for different numbers of iterations, with the true posterior distribution. In order to see which modifications we should do to have better approximation and with it a better understanding of our brain.

All the code can be found add:
`https://github.com/arnauparrilla/Bistable-Perception`.

7 References

[1] Wikipedia contributors, *Statistical inference — Wikipedia, the free encyclopedia*, [Online; accessed 2-June-2022], 2022. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Statistical_inference&oldid=1086351221.

[2] ——, *Bayesian inference — Wikipedia, the free encyclopedia*, https://en.wikipedia.org/w/index.php?title=Bayesian_inference&oldid=1090145649, [Online; accessed 31-May-2022], 2022.

[3] K. P. Körding and D. M. Wolpert, "Bayesian decision theory in sensorimotor control," *Trends in Cognitive Sciences*, vol. 10, no. 7, pp. 319–326, 2006, Special issue: Probabilistic models of cognition, ISSN: 1364-6613. DOI: https://doi.org/10.1016/j.tics.2006.05.003. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1364661306001276.

[4] Interaction Design Foundation, *Visual perception*. [Online]. Available: https://www.interaction-design.org/literature/topics/visual-perception#:~:text=Visual%5C%20perception%5C%20is%5C%20the%5C%20ability,are%5C%20perceived%5C%20exclusively%5C%20through%5C%20vision..

[5] Wikipedia, the free encyclopedia, *Graphical model*, 2021. [Online]. Available: https://en.wikipedia.org/wiki/Graphical_model.

[6] X. Pitkow and D. Angelaki, "Inference in the brain: Statistics flowing in redundant population codes," *Neuron*, vol. 94, pp. 943–953, Jun. 2017. DOI: 10.1016/j.neuron.2017.05.028.

[7] Cybenko, 1989.

[8] Hornik, 1991.

[9] Montúfar, 2014.

[10] Alex Hyafil, *World wide theoretical neuroscience seminar: Alex hyafil, december, 22, 2021*. [Online]. Available: https://www.youtube.com/watch?v=VCx-ollnJ6E.

[11] K. P. Murphy, *Machine learning : a probabilistic perspective*. Cambridge, Mass. [u.a.]: MIT Press, 2013, ISBN: 9780262018029 0262018020. [Online]. Available: https://www.amazon.com/Machine-Learning-Probabilistic-Perspective-Computation/dp/0262018020/ref=sr_1_2?ie=UTF8&qid=1336857747&sr=8-2.

[12] Wikipedia, the free encyclopedia, *Clique (graph theory)*, 2021. [Online]. Available: https://en.wikipedia.org/wiki/Clique_(graph_theory).

[13] ——, *A vietoris–rips complex of a set of 23 points in the euclidean plane*, 2021. [Online]. Available: https://en.wikipedia.org/wiki/Clique_(graph_theory)#/media/File:VR_complex.svg.

[14] Weisstein, Eric W, *Maximal clique*, [From MathWorld–A Wolfram Web Resource]. [Online]. Available: https://mathworld.wolfram.com/MaximalClique.html.

[15] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*, 4. Springer, 2006, vol. 4.

[16] R. Jardri and S. Denève, "Circular inferences in schizophrenia," *Brain*, vol. 136, no. 11, pp. 3227–3241, Sep. 2013, ISSN: 0006-8950. DOI: 10.1093/brain/awt257. eprint: https://academic.oup.com/brain/article-pdf/136/11/3227/13795765/awt257.pdf. [Online]. Available: https://doi.org/10.1093/brain/awt257.

[17] Carles Barril Basil, *Sistemes dinàmics*, (Theorem 1 from page 6, Sistemes dinàmics, Modelització i Simulació, UAB, March 22, 2021).

[18] ——, *Sistemes dinàmics*, (Sistemes dinàmics, Modelització i Simulació, UAB, March 25, 2021).

[19] ——, *Sistemes dinàmics*, (Bifuracacions de punts fixos, Sistemes dinàmics, Modelització i Simulació, UAB, March 11, 2021).

A Necker cube states

First we will define a bases state from which we will make the modifications to differentiate them. The bases state of the cube are when the values of all the variables x_i are unmodified, having two of them. One when all x_i are set to 1 and the other to -1. Thus for each defined class we have two cases representing the same symmetric state. That corresponds on changing the values of the same x_i, but from one bases state or the other.

Case where $\mu = 8$ and $\mu = -8$
For $\mu = 8$, this means that we do not change the value of any x_i we have:

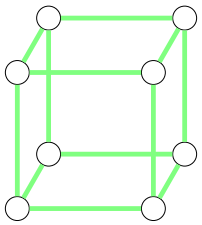

Figure 17: **Class 0** This state of the cube corresponds on having all the vertex x_i unmodified. We only have one combination of this configuration and the probability of the state is: $p(\text{Class } 0) = \frac{1}{Z} exp(12J)$ since we have 12 positive connections between nodes.

We have two classes of the cube that are represented with this configuration the previous one and the one with all $x_i = -1$ and $\mu = -8$ (*i.e.* the two opposite states of the cube) having both the exact same probability. The total probability for this class is $p(\mu = 8) = 2\frac{1}{Z}\exp(12J)$.

Case where $\mu = 6$ and $\mu = -6$

For $\mu = 6$, this means that we only modify the value of one vertex being opposite of the others.

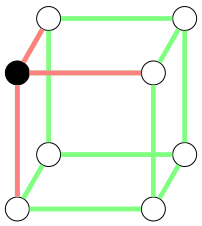

Figure 18: **Class 1** This state corresponds on modifying the value of just one of the variables x_i. We have 8 different combinations since the variable that is changed could be any of the eight x_i of the cube. The probability of this state is: $p(\text{Class } 1) = \frac{1}{Z} exp(\sum 9J - 3J) = \frac{1}{Z} exp(6J)$ with $\mu = 6$

In total we have 16 different states of the cube represented by this configuration the eight represented in Figure 18 where we have all the variable positive and one negative and the opposite state with all the variables being negative and one positive having all the same probability. Thus the total probability of for this class is $p(\mu = 6) = 16\frac{1}{Z}\exp(6J)$

Case where $\mu = 4$ and $\mu = -4$

To obtain $\mu = 4$ we have to change the value of two variables that will be opposite to the rest. If we modify the value of two variables, these two variables modified can have three different positions with respect the others.

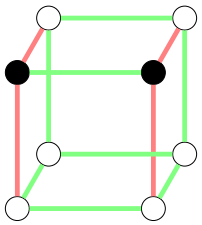

Figure 19: **Class 2** This state corresponds to modifying the value of two variables x_i where the variables are next to each other. We have 12 different combinations of this configuration since there are 12 edges in the cube. The probability of this state is: $p(\text{Class 2}) = \frac{1}{Z} exp(\sum 8J - 4J) = \frac{1}{Z} exp(4J)$

In total we have 24 different states represented by this configuration the twelve represented in Figure 19 and the opposite state where the two changed variables are positive and the rest negative.

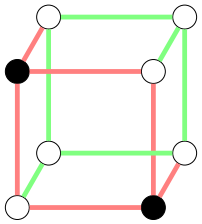

Figure 20: **Class 3** This state corresponds to modifying the value of two variables x_i where the changed variables are in the same face but their edges are not connected. We have 12 different combinations of this configurations since we have 6 faces and 2 feasible positions for each face. The probability of this state is: $p(\text{Class 3}) = \frac{1}{Z} exp(\sum 6J - 6J) = \frac{1}{Z} exp(0J) = \frac{1}{Z}$

In total we have 24 different states represented by this configuration the twelve represented in Figure 20 and the opposite state where the two changed variables are positive and the rest negative.

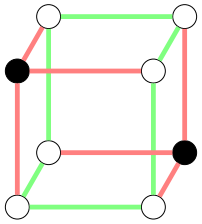

Figure 21: **Class 4** This state corresponds to modifying the value of two variables x_i where the changed variables are in two different faces of the cube. We have 4 different combinations of this configuration since the cube has 4 diagonals. The probability of this state is: $p(\text{Class 4}) = \frac{1}{Z} exp(\sum 6J - 6J) = \frac{1}{Z} exp(0J) = \frac{1}{Z}$

In total we have 24 different states represented by this configuration the twelve

represented in Figure 21 and the opposite state where the two changed variables are positive and the rest negative.

The total probability for $\mu = 4$ and $\mu = -4$ is the sum of the three states above being:

$$p(\mu = \pm 4) = 8\frac{1}{Z} + 24\frac{1}{Z} + 24\frac{1}{Z}\exp(4J) \tag{40}$$

Case where $\mu = 2$ and $\mu = -2$

For the cases where $\mu = 2$ the value of three variables has to be changed. These three variables have three different positions with respect the others.

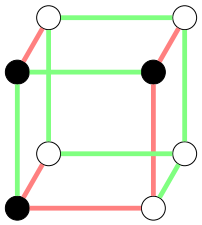

Figure 22: **Class 5** This state corresponds to modifying the value of three variables x_i where the changed variables are on the same face of the cube. We have 24 different combinations of this configuration, 4 different positions in each face for the 6 faces of the cube. The probability of this state is: $p(\text{Class 5}) = \frac{1}{Z}exp(\sum 7J - 5J) = \frac{1}{Z}exp(2J)$

In total we have 48 different states represented by this configuration the 24 represented in Figure 22 and the other 24 opposite states where the three changed variables are positive and the rest negative.

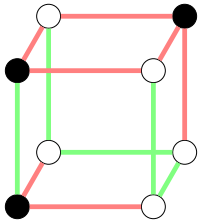

Figure 23: **Class 6** This state corresponds to modifying the value of the three variables x_i where two of them are directly connected and the other is in a different face (*i.e* forming a diagonal plane with respects the cubes perspective). We have 24 different combinations of this configuration, 4 different positions in each face for the 6 faces of the cube. The probability of this state is: $p(\text{Class 6}) = \frac{1}{Z}exp(\sum 5J - 7J) = \frac{1}{Z}exp(-2J)$

In total we have 48 different states represented by this configuration the 24 represented in Figure 23 and the other 24 opposite states where the three changed variables are positive and the rest negative.

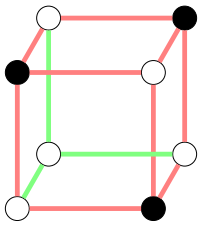

Figure 24: **Class 7** This state corresponds to modifying the value of the three variables x_i where we change the three variables around an unmodified vertex. Since we have 8 vertex, there are 8 different combinations of this configuration. The probability of this state is: $p(\text{Class 7}) = \frac{1}{Z} exp(\sum 3J - 9J) = \frac{1}{Z} exp(-6J)$

In total we have 16 different states represented by this configuration the 8 represented in Figure 24 and the other 8 corresponding to the opposite states where the three changed variables are positive and the rest negative.

The total probability for $\mu = \pm 2$ is the sum of the three states studied above:

$$p(\mu = \pm 2) = 48\frac{1}{Z}\exp(2J) + 48\frac{1}{Z}\exp(-2J) + 16\frac{1}{Z}\exp(-6J) \qquad (41)$$

Case where $\mu = 0$

For $\mu = 0$ is the case that does not have and opposite state (*i.e.* the exactly same state but with the variables having the opposite value). To obtain $\mu = 0$ we need to change the value of 4 variable (*i.e.* half of the variables we have), therefore can not have an opposite state. We can arrange this four variables in 6 different configurations.

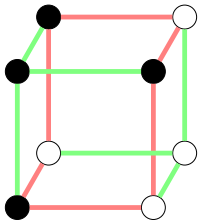

Figure 25: **Class 8** This state corresponds to modifying the values of the variables x_i around a vertex that also has its value changed. Since we have 8 vertex, there are 8 different combinations of this configuration. The probability of this state is: $p(\text{Class 8}) = \frac{1}{Z} exp(\sum 8J - 4J) = \frac{1}{Z} exp(4J)$

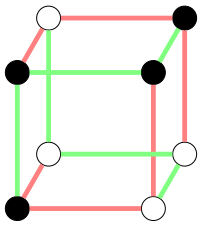

Figure 26: **Class 9** This state corresponds to modifying the four variables x_i having two adjacent faces with three variables connected. We have 24 different combinations of this configuration, 4 different positions in each face for the 6 faces of the cube. Thus the probability of the states is: $p(\text{Class 9}) = \frac{1}{Z} exp(\sum 6J - 6J) = \frac{1}{Z} exp(0) = \frac{1}{Z}$

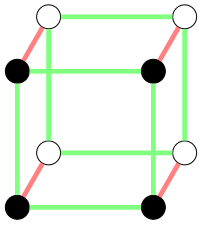

Figure 27: **Class 10** This state corresponds to modifying the value of four variables x_i in the same face. Since we have 6 faces, there are 6 different combinations of this configuration. The probability of this state is: $p(\text{Class 10}) = \frac{1}{Z} exp(\sum 8J - 4J) = \frac{1}{Z} exp(4J)$

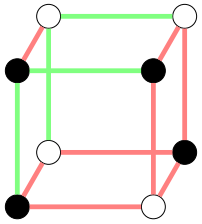

Figure 28: **Class 11** This state corresponds to modifying the value of three variables x_i on the same face and the other in the vertex that do not connect it to the rest. We have 24 different combinations of this configuration, 4 different positions in each face for the 6 faces of the cube. Thus the probability of the states is: $p(\text{Class 11}) = \frac{1}{Z} exp(\sum 4J - 8J) = \frac{1}{Z} exp(-4J)$

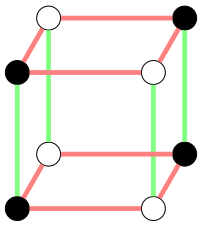

Figure 29: **Class 12** This state corresponds to modifying the value of the four variables where they are connected in pairs. Since we have 12 edges in the cube this means that we have 6 pairs of edges that satisfy this property $i.e$ 6 different combinations of this configuration. The probability of the states is: $p(\text{Class 12}) = \frac{1}{Z} exp(\sum 4J - 8J) = \frac{1}{Z} exp(-4J)$

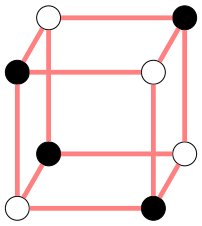

Figure 30: **Class 13** This state corresponds to modifying the value of the four variables where they all of the modified variables are not connected by any edge. We only have 2 combinations of this configuration. The probability of this state is: $p(\text{Class 13}) = \frac{1}{Z} exp(\sum -12J) = \frac{1}{Z} exp(-12J)$

The total probability for $\mu = 0$ is the sum of these six states:

$$p(\mu = 0) = 6\frac{1}{Z}\exp(4J) + 8\frac{1}{Z} + 24\frac{1}{Z} + 24\frac{1}{Z}\exp(-4J) + 6\frac{1}{Z}\exp(-4J) + 2\frac{1}{Z}\exp(-12J) \tag{42}$$

Normalization constant.

$$\begin{aligned} Z = 2\exp(12J) + 16\exp(6J) + 30\exp(4J) + 48\exp(2J) + 48\exp(-2J) + \\ +30\exp(-4J) + 16\exp(-6J) + 2\exp(-12 * J) + 64 \end{aligned} \tag{43}$$

B Matrix C

$$\begin{pmatrix}
0 & 8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 3 & 3 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 2 & 0 & 0 & 0 & 4 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 2 & 0 & 0 & 0 & 2 & 2 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 2 & 0 & 0 & 0 & 0 & 6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 2 & 1 & 0 & 0 & 0 & 0 & 1 & 2 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 2 & 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 3 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 3 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 2 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 2 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 3 & 0 & 1 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 2 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6 & 0 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 2 & 2 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 4 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 3 & 3 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 8 & 0
\end{pmatrix} \tag{44}$$

C Exact probability distributions

The following images corresponds to the normalized exact probability defined before, for the 22 states.

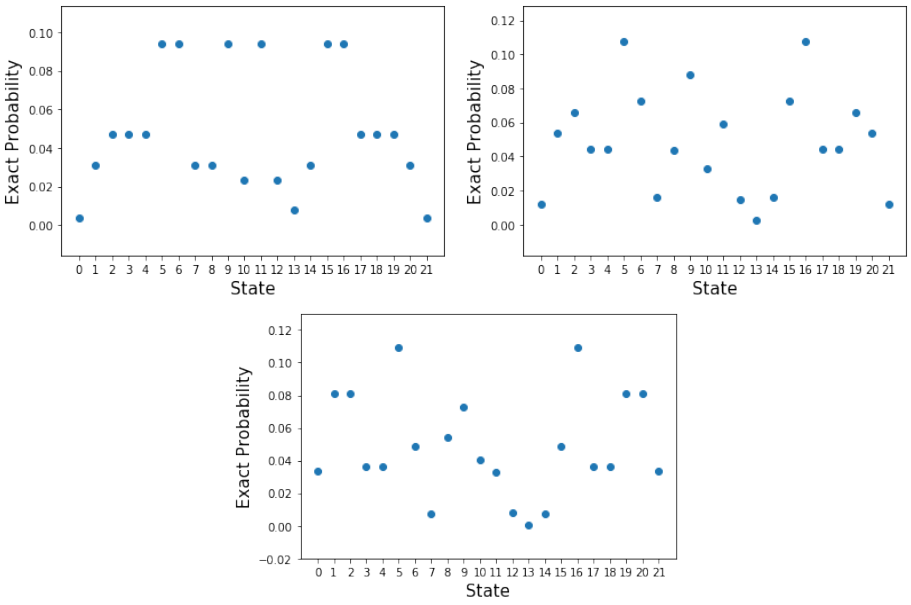

Figure 31: The three images corresponds to the exact probabilities of J = 0, J = 0.1, J = 0.2 respectively.

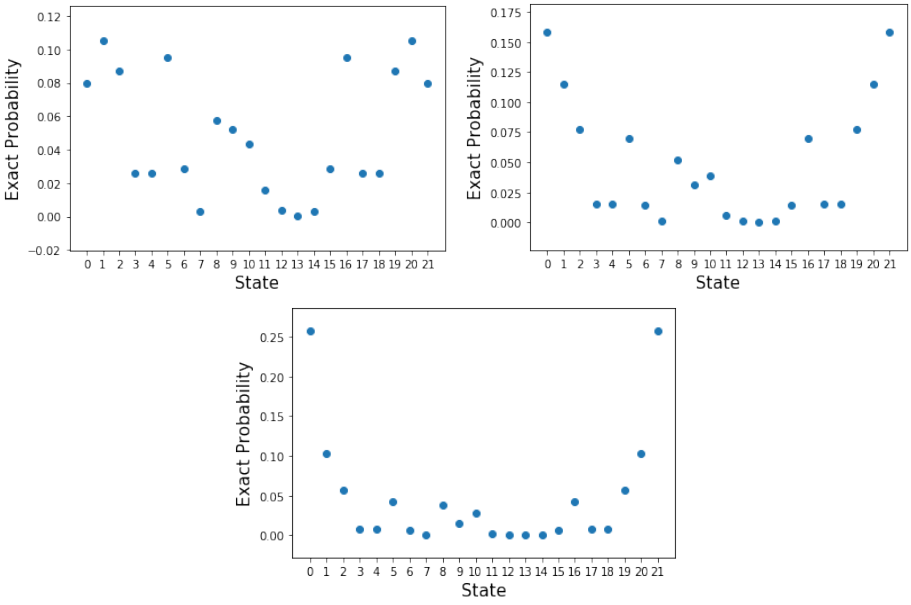

Figure 32: The three images corresponds to the exact probabilities of J = 0.3, J = 0.4, J = 0.5 respectively.

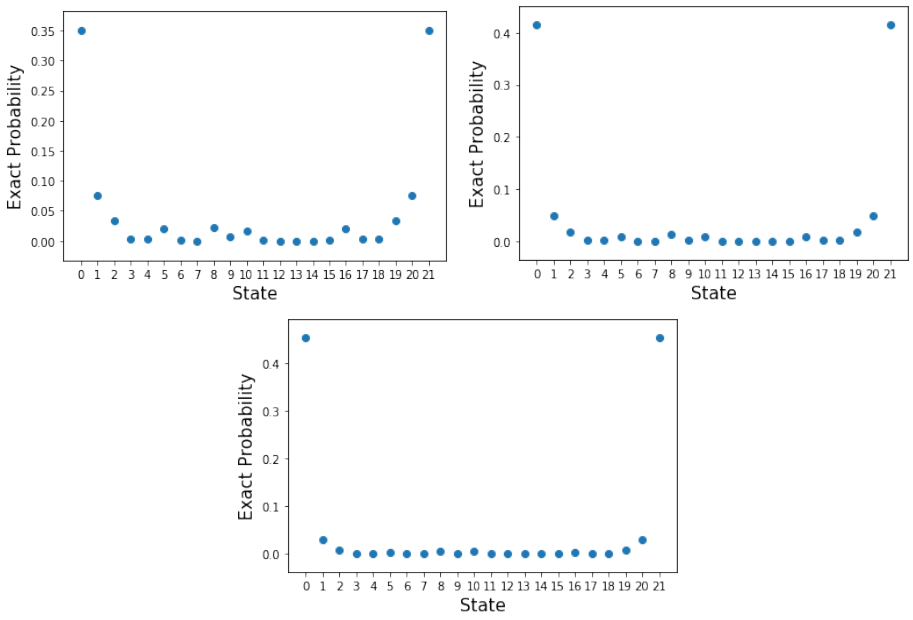

Figure 33: The three images corresponds to the exact probabilities of J = 0.6, J = 0.7, J = 0.8 respectively.

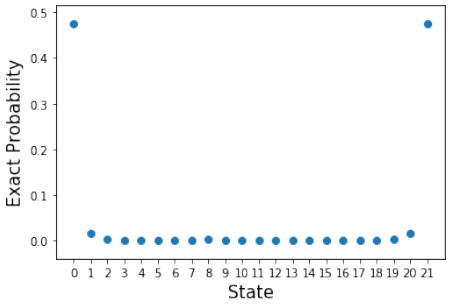

Figure 34: The image corresponds to the exact probabilities of J = 0.9.

D Gibbs simulation

The following plots corresponds to the mean with its respective 95% confidence interval for 100.000 iterations of the Gibbs algorithm assigning the approximate probability to each of the 22 states defined before. It is important to note that the confidence intervals (represented with a blue vertical line) are usually not visible due to the scale. When we can not appreciate them is because they are practically zero.

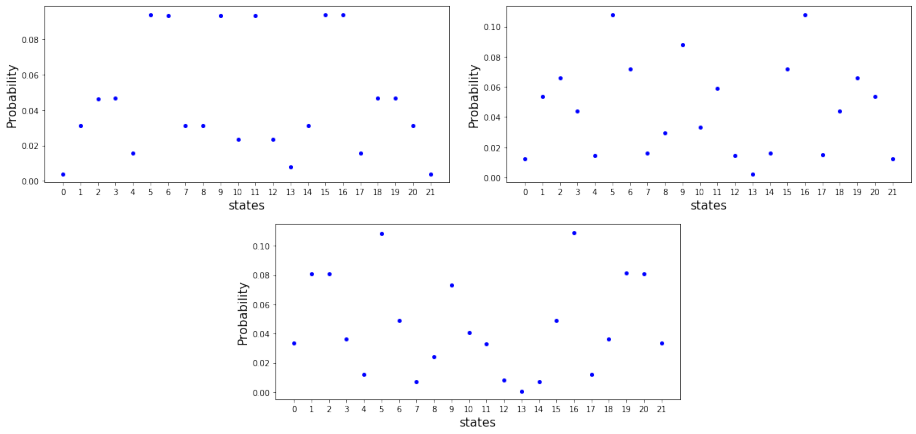

Figure 35: The three images corresponds to the simulated probabilities of J = 0, J = 0.1, J = 0.2 respectively.

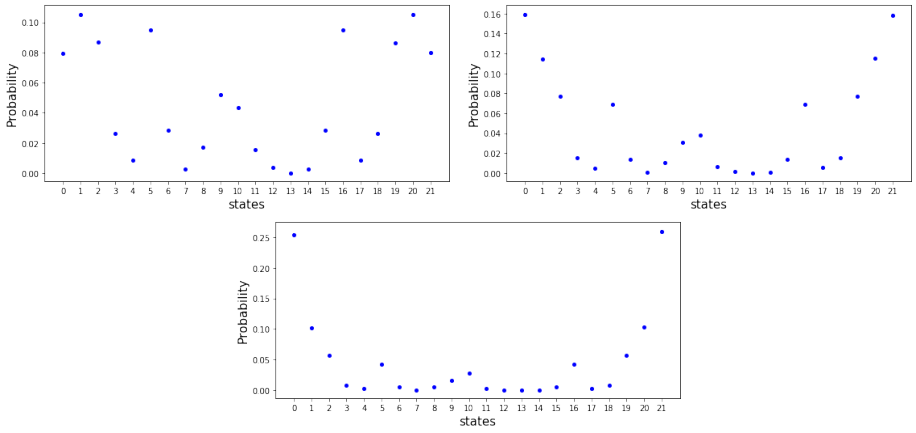

Figure 36: The three images corresponds to the simulated probabilities of J = 0.3, J = 0.4, J = 0.5 respectively.

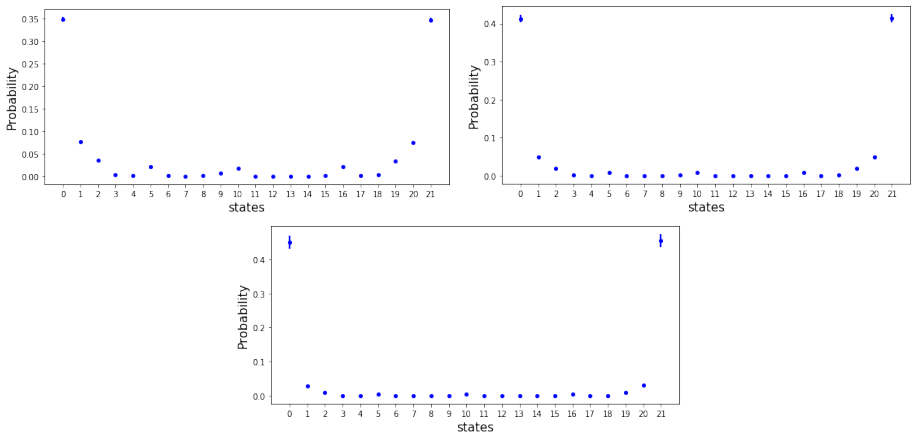

Figure 37: The three images corresponds to the simulated probabilities of J = 0.6, J = 0.7, J = 0.8 respectively.

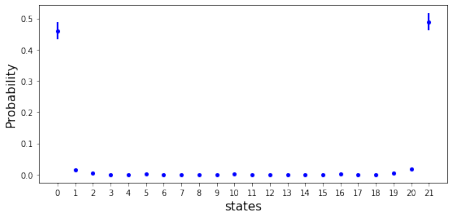

Figure 38: The image corresponds to the simulated probabilities of J = 0.9.

Here we have more simulations but for 5000 iterations.

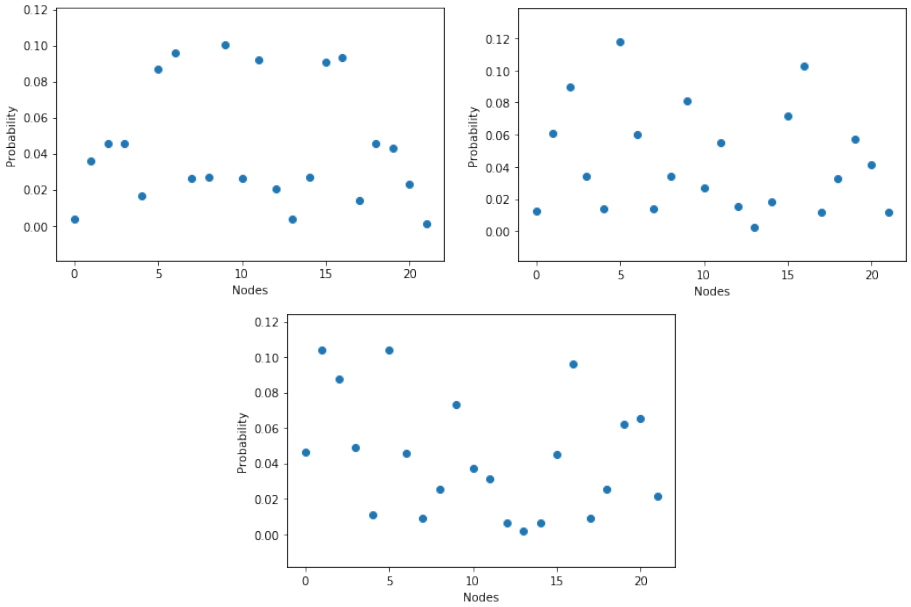

Figure 39: The three images corresponds to the simulated probabilities of J = 0, J = 0.1, J = 0.2 respectively.

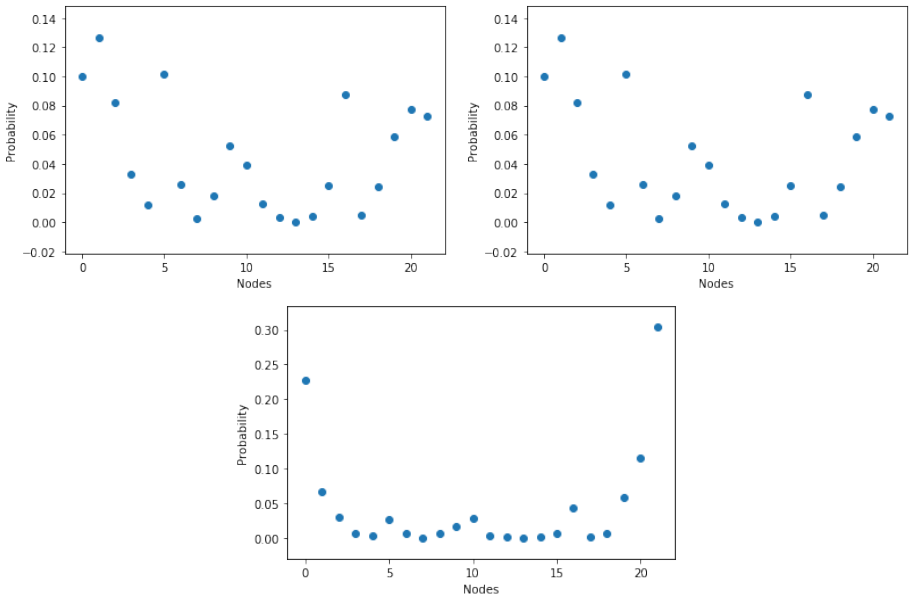

Figure 40: The three images corresponds to the simulated probabilities of J = 0.3, J = 0.4, J = 0.5 respectively.

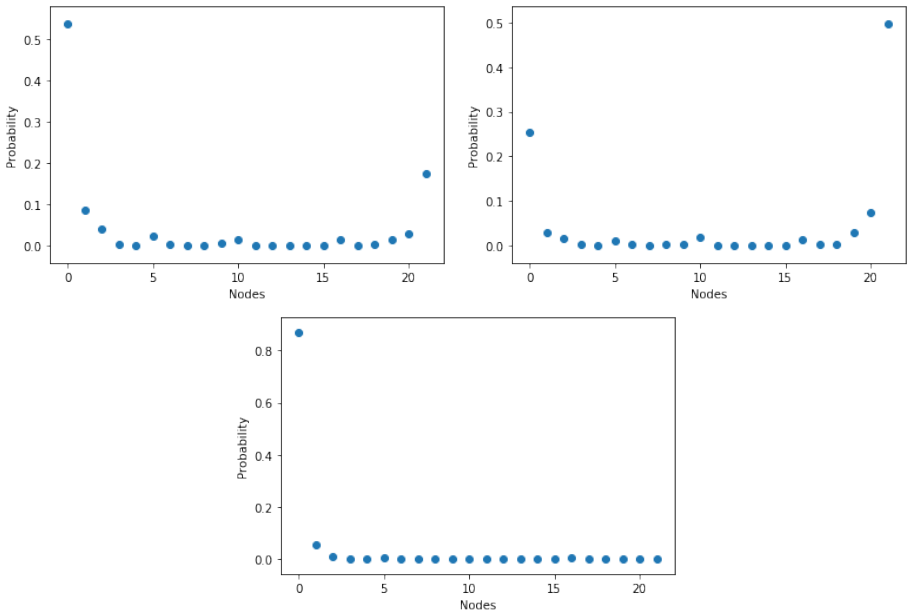

Figure 41: The three images corresponds to the simulated probabilities of J = 0.6, J = 0.7, J = 0.8 respectively.

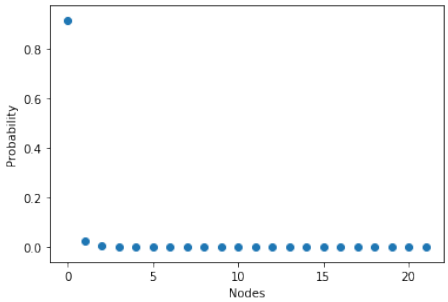

Figure 42: The image corresponds to the simulated probabilities of J = 0.9.

E Mean Field simulation tables

In these tables we have two different simulation for the Mean Field algorithm of $n = 5000$ iterations. We have not made more simulations since as we have previously said the exact values of our dynamical system can be obtained by solving the dynamical system as we have done. This is only to notice that for the exact same value of J, when $J > \frac{1}{3}$ depending on the simulation (*i.e.* the initialization value) the distribution will converge into one of the two attractor nodes.

Probability of node $q_i = 1$

Value of J	q1	q2	q3	q4	q5	q6	q7	q8
0.0	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
0.1	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
0.2	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
0.3	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
0.4	0.17	0.17	0.17	0.17	0.17	0.17	0.17	0.17
0.5	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07
0.6	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96
0.7	0.017	0.017	0.017	0.017	0.017	0.0171	0.017	0.017
0.8	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
0.9	0.0047	0.0047	0.0047	0.0047	0.0047	0.0047	0.0047	0.0047

Probability of node $q_i = 1$

Value of J	q1	q2	q3	q4	q5	q6	q7	q8
0.0	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
0.1	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
0.2	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
0.3	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
0.4	0.829	0.829	0.829	0.829	0.829	0.829	0.829	0.829
0.5	0.929	0.929	0.929	0.929	0.929	0.929	0.929	0.929
0.6	0.966	0.966	0.966	0.966	0.966	0.966	0.966	0.966
0.7	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017
0.8	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
0.9	0.0047	0.0047	0.0047	0.0047	0.0047	0.0047	0.0047	0.0047

For 300 iterations:

Probability of node $q_i = 1$

Value of J	q1	q2	q3	q4	q5	q6	q7	q8
0.0	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
0.1	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
0.2	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
0.3	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
0.4	0.829	0.829	0.829	0.829	0.829	0.829	0.829	0.829
0.5	0.929	0.929	0.929	0.929	0.929	0.929	0.929	0.929
0.6	0.0336	0.0336	0.0336	0.0336	0.0336	0.0336	0.0336	0.0336
0.7	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017
0.8	0.0088	0.0088	0.0088	0.0088	0.0088	0.0088	0.0088	0.0088
0.9	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99

Probability of node $q_i = 1$

Value of J	q1	q2	q3	q4	q5	q6	q7	q8
0.0	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
0.1	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
0.2	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
0.3	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
0.4	0.171	0.171	0.171	0.171	0.171	0.171	0.171	0.171
0.5	0.929	0.929	0.929	0.929	0.929	0.929	0.929	0.929
0.6	0.034	0.034	0.034	0.034	0.034	0.034	0.034	0.034
0.7	0.983	0.983	0.983	0.983	0.983	0.983	0.983	0.983
0.8	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
0.9	0.0047	0.0047	0.0047	0.0047	0.0047	0.0047	0.0047	0.0047