Universiteit Utrecht

Machine learning phases of active matter

Finite size scaling in the Vicsek model by means of a Principle Component Analysis and Neural Networks

[**Matthijs de Koning** `m.d.dekoning@students.uu.nl`]

Abstract

The interest in understanding the group motion of living systems provides a breeding ground for a plethora of active matter models in statistical physics. The Vicsek model (VM), a minimal model of self-propelled particles in which their tendency to align with each other competes with perturbations controlled by a noise term, captures this behaviour of collective motion. In this thesis the machine learning tools Principal Component Analysis (PCA) and Neural Networks (NN) have been used to detect order-disorder phase transitions in the VM. PCA was able to construct an order parameter even in the presence of limited and inherently noisy data. The NN detected critical points of phase transitions for systems greater than 1000 particles, but struggled to find phase transitions in smaller systems. The finite size scaling found the critical noise value $\eta_c(\infty) = 2.11 \pm 0.25$ without the use of a NN and $\eta_c(\infty) = 2.28 \pm 0.16$ with the use of a NN. Furthermore, critical exponents $\beta = 0.3$, $\gamma = 2.1$ and $\nu = 0.9$ were extracted.

Supervisors:

Prof. Dr. Ir. Marjolein Dijkstra
Debye institute

Prof. Dr. René van Roij
Institute for theoretical physics

Bachelor Thesis
12-6-2020

Department of Physics

Contents

1 Introduction

This thesis is about a machine learning analysis of active matter systems. Active matter are systems in which the constituents are self-propelling. Motion of self-propelling agents is a fascinating feature that can be observed on many different scales. Examples in nature include schools of fish [1], flocks of birds [2], pedestrian crowds [3], insect swarms [4] and bacterial colonies [5]. Active systems are not in equilibrium, because individual particles take up energy from their surroundings in order to move and therefore kinetic energy is added to the system. As a result the laws of equilibrium thermodynamics are no longer valid to describe these systems.

The interest in understanding this behaviour of living systems provides a breeding ground for a plethora of models and studies in statistical physics [6]. In view of the widespread interest and applications of the phenomenon it is desirable to understand one of the simplest possible models capable of capturing the main features of active matter motion in a nontrivial manner. In this respect, the Vicsek Model (VM), proposed by Vicsek et al. in 1995 [7], stands out because of its minimal character. In this model, identical point particles move at constant velocity and interact locally by trying to align their direction with that of their neighbors. Curiously, even in the presence of noise, individual particles line up and collective motion emerges provided the density of particles is high enough or, equivalently, if the noise is weak enough.

Machine learning, on the other hand, already explored as a tool in condensed-matter research [8], provides a complementary paradigm to the above situation. The ability of modern machine learning techniques to classify, identify, or interpret massive data sets foreshadows their suitability to provide physicists with similar analyses on the exponentially large data sets embodied in condensed-matter systems. Therefore, our research question will be: How well are machine learning architectures, such as Principle Component Analysis (PCA) and Neural Networks (NN), able to identify phase transitions and order parameters in the VM?

The outline of this thesis is as follows. In section 2 we provide the necessary theoretical background. The VM is introduced together with the main results from previous investigations. The important features of PCA and neural networks are discussed as well. In section 3, we present the simulation method. Finally, we discuss our simulations and we present an analysis of our results using PCA and NNs. In section 5, we present our conclusion and provide an outlook for future research.

2 Theoretical essentials

2.1 The Vicsek Model

A simple model for active matter was introduced in 1995 by Vicsek [7]. It consists of N point particles confined to a 2D square cell of dimensions $L \times L$. Every particle moves with a constant absolute velocity v, in a direction that is described by an angle $\theta(t)$ relative to the x-axis at time t. In addition to the usual mechanics induced by the positions and velocities of the particles, the model has only one extra rule: at time step t the direction of motion of a particle is determined by the averaged direction of motion of all the neighbouring particles within a distance r of that particle with some additional noise. After a certain time Δt we update the angle of the i^{th} particle as

$$\theta_i(t + \Delta t) = \langle \theta_i(t) \rangle_r + \Delta\theta_i \tag{1}$$

where

$$\langle \theta_i(t) \rangle = \arctan\left(\frac{\frac{1}{N_b(i)+1} \sum_{j=1}^{N_b(i)+1} \sin(\theta(t))}{\frac{1}{N_b(i)+1} \sum_{j=1}^{N_b(i)+1} \cos(\theta(t))} \right)$$

denotes the average direction of all the neighbouring particles of i (including particle i) within a circle of radius r surrounding particle i. $N_b(i)$ denotes the number of neighbouring particles of i. In Eq. 1, $\Delta\theta_i$ is a random number sampled from a uniform distribution $[-\eta/2, \eta/2]$. This implies that at every time iteration for every particle a certain amount of noise, $\Delta\theta_i$, is added to the system. The value of $\Delta\theta_i$ is separately generated for every particle and time step. From this point on we will refer to η as the noise. After the particle has aligned with its surrounding particles, it will change its position according to

$$\vec{x_i}(t + \Delta t) = \vec{x_i}(t) + \vec{v_i}\Delta t. \tag{2}$$

$\vec{v_i}$ is related to θ by $\vec{v_i} = [v\cos\theta_i, v\sin\theta_i]$. The free parameters of the system are the absolute velocity v, noise η and density $\rho = N/L^2$. Furthermore, in this thesis we use r and Δt as our unit of length and time, and we keep the dimensionless absolute velocity $v\Delta t/r = 0.03$ fixed.

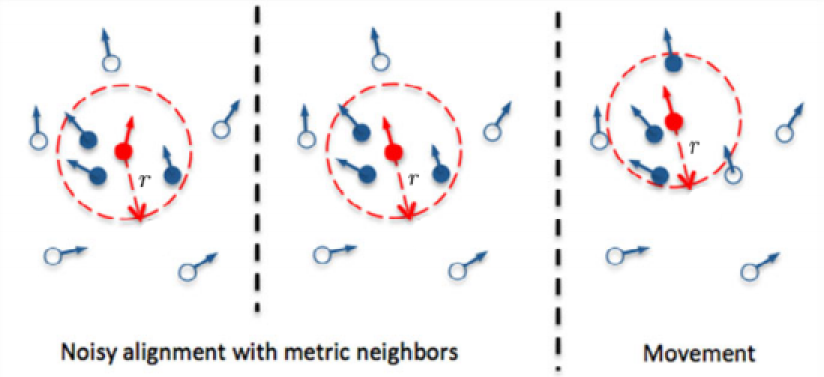
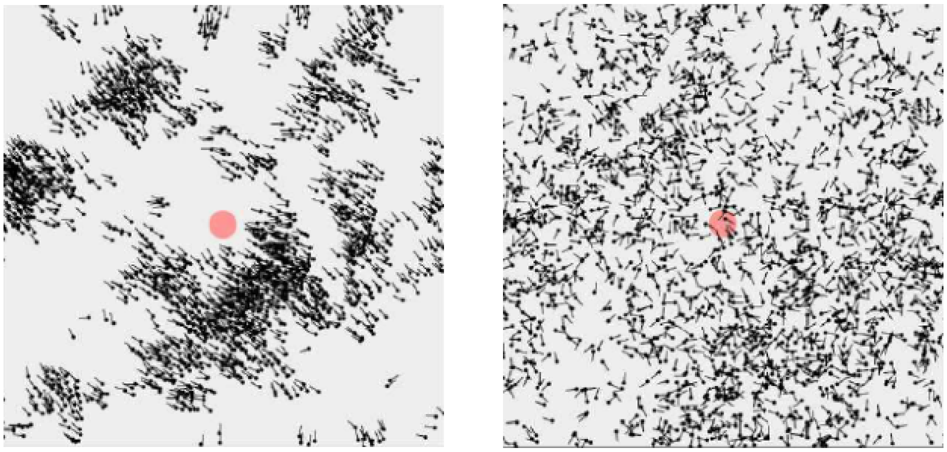


Figure 1: A visualisation of the Vicsek algorithm. Picture taken and adjusted from [9].



(a) Snapshot of an ordered phase. (b) Snapshot of a disordered phase.

Figure 2: Examples of different phases in the VM. The red dot indicates the size of the interaction area. Pictures are taken from [14].

Because of the minimal character of the VM a lot of extensions exist. Some examples are: a VM with repulsion [10], a VM with the added effects of a surrounding fluid [11] and a VM that takes into account individualistic behaviour [12]. Furthermore, there exists a subtle variation of the VM regarding the manner of how the updating rule is performed and how the noise is applied. It is possible to first update the direction of motion of a particle before moving (backward update) or vice versa (forward update), and one can choose to apply noise only to the angular part of the velocity vector (angular noise) or to the vector as a whole (vectorial noise). Combination of these schemes give rise to different behaviour. More information on how the different updating rules compare can be found in ref. [13]. In this thesis angular noise and a backward updating scheme is chosen, as is used in the original Vicsek paper. It is logical to start applying machine learning techniques to the first and arguably most basic VM.

All studies about the VM, and variations there-of, show that an order-disorder phase transition occurs. When the noise is sufficiently low the individual particles are able to align with their neighbours and a collectively coordinated ordered phase occurs. See figure 2a. When the noise is steadily increased this process of alignment becomes harder and harder until a point is reached, the critical point, at which the phase of the system changes to a disordered gas-like phase. See figure 2b. The noise value that is associated with the critical point of the system is called the critical value of the noise η_c. In literature it is found that the critical value of the noise depends on the density of the system and on the absolute velocity of the individual particles [9, 14, 15]. In ref. [15] the critical value of the noise $\eta_c = 2.14$ has been found for $\rho = 4$ and $v = 0.03$.

The quantities of interest are the instantaneous absolute value of the average normalised velocity

$$v_a = \frac{1}{Nv} | \sum_{i=1}^{N} \vec{v_i} |, \tag{3}$$

its fluctuations (or susceptibility)

$$\chi = L^2(\langle v_a^2 \rangle - \langle v_a \rangle^2) \tag{4}$$

and the critical value of the noise η_c. The easiest way to capture the phase transition from disorder to collective motion is to monitor v_a as a function of the noise η. v_a is approximately zero if the direction of the motion of the individual particles is random (disordered phase), and is one when the directions of all the individual particles line up (ordered phase). Thus we consider v_a as an order parameter. From calculating the susceptibility χ as a function of the noise η an estimate for the critical value of the noise η_c can be made. The susceptibility χ will reach a maximum at η_c for at this value the fluctuations of the order parameter will be the largest.

2.2 Finite size scaling

Finite size scaling is a method that extracts values for critical exponents. Critical exponents are numbers assigned to a particular system to classify it based on their thermodynamic properties around their respective critical point. Take for example the Ising model and liquid crystals in LCD screens. At first sight, they are two entirely different systems, but around their respective phase transitions they can be described by the same models, e.g. the have the same values for their critical exponents. This is useful for physicists, because one can derive properties from one system by performing measurements on the other. The critical exponents that we are interested in are: γ, β and ν. They can be obtained by plotting equations 5 and 6 as a function of $|\eta - \eta_c|L^{1/\nu}$ for different system sizes L.

$$\tilde{\chi}(L, |\eta - \eta_c|) = L^{-\gamma/\nu}\chi(|\eta - \eta_c|L^{1/\nu}) \tag{5}$$

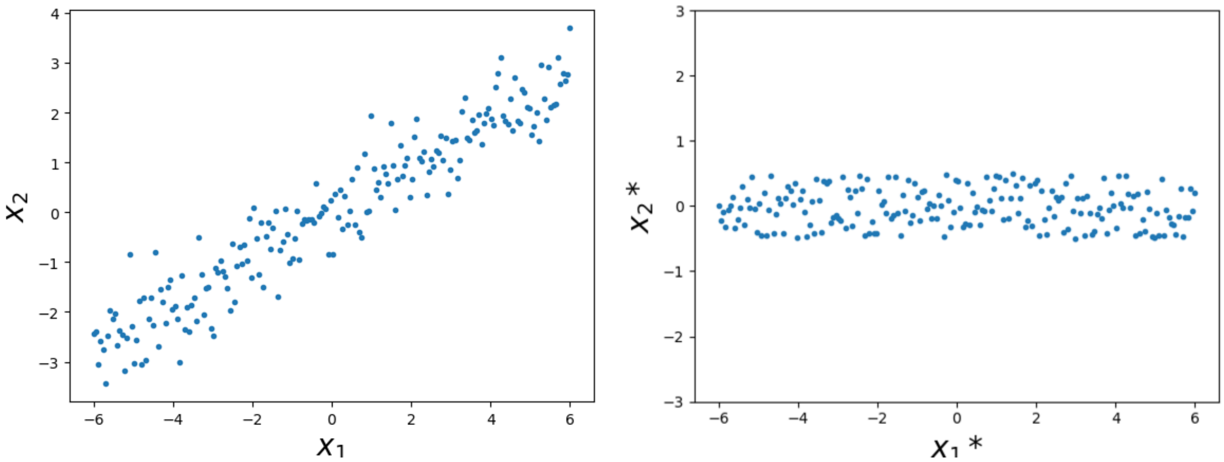
$$\tilde{v_a}(L, |\eta - \eta_c|) = L^{-\beta/\nu}v_a(|\eta - \eta_c|L^{1/\nu}) \tag{6}$$

$|...|$ denotes the absolute value, $\chi = L^2(\langle v_a^2 \rangle - \langle v_a \rangle^2)$ is the susceptibility, v_a the order parameter of the VM and η_c is the value of the critical point. χ and v_a are calculated directly from simulations and are data, together with L, that is known. η_c and the critical exponents γ, β and ν are unknowns. $\tilde{\chi}$ and $\tilde{v_a}$ in equations 5 and 6 are so called scaling functions. Their shapes are unknown, but some important properties are not. We know what these properties are, because of the way that they are defined. So by design, the scaling functions are not dependent on the system size. This means that measurements of $\tilde{v_a}$ for different system sizes should all fall on the same curve if we plot them together in one graph. However - and this is the crucial point - this will only happen when we use the one correct combination of the critical exponents γ, β, ν and the correct value of the critical noise η_c.
Finally, studies about the VM have performed finite size scaling analysis for a wide range of variations of system parameters. Ref. [15] found that the values of the critical exponents depend on the density ρ and absolute velocity v. For the values $\rho = 4$ and $v = 0.03$, which will also be used in this thesis, critical exponents $\gamma = 2.16(9)$, $\beta = 0.31(3)$ and $\nu = 0.93(3)$ have been found.

2.3 Machine Learning

2.3.1 Principal Component Analysis

Principal component analysis is a statistical technique that analyses a data set involving a substantial number of correlated variables. The goals of PCA are: to extract the important information from the data, reduce the dimensionality of the data and analyse the structure between the observations and the variables. In order to achieve these goals a transformation to a new orthogonal basis is performed. The vectors of this new basis are called principal components. The principal components are linear combinations of the original variables and the weights of the new variables are determined by the eigenvectors of the original data matrix. The first principal component is the direction in the data set which maximises the variance. The variance is the eigenvalue of that principal component. The second component, again, will maximise the variance in the remaining data set, but under the constraint that it is orthogonal to the first component. The third will maximise the variance under the constraint that it is orthogonal to both the first and the second component. The other components are calculated analogously. A natural result of this procedure is that most variance of the data will be expressed in the first few components. Because humans can not conceptualise high dimensional spaces, compressing the data in a 2 or 3 dimensional space makes the data easier to visualise. An example of PCA analysis is illustrated in figure 3.



(a) Original data **X**. Note that most variance is along the positive sloping diagonal.

(b) Transformed data **X*** plotted along the principal components.

Figure 3: Example of PCA analysis of a 2-dimensional toy dataset.

PCA is most effective, like all other unsupervised learning algorithms, with a large data set **X**. PCA has scores of applications that span a wide variety of fields. Because of this, the structure of **X** varies quite a bit depending on the context in which it is used. In our application of physical feature extraction, **X** consists of the information contained in the final configuration of a VM simulation for typical values of η and ρ. The input data matrix **X** then consists of n observations described by m variables, such that its dimension is $n \times m$. In this thesis a variety of combinations of the velocities of every particle is used to describe the configuration of a system. These are the variables m. See for example figure 5. Furthermore, a standard data preprocessing procedure when dealing with PCA is to create a centered and column-wise zero mean data matrix. This is achieved by subtracting the column average from every column entry in $\mathbf{X}$ such that

$$\mathbf{X}_{ij}^{\text{processed}} = \mathbf{X}_{ij} - \frac{1}{m}\sum_{k=1}^{n} X_{kj}. \tag{7}$$

Principle component analysis has three important outputs: the principal components of the data matrix (the eigenvectors)[1], the variance along the respective components (the eigenvalues) and the transformed data matrix, which we call $\mathbf{X*}$. We can find these objects by solving the eigenvalue problem:

$$\mathbf{X}^T\mathbf{X}\vec{w}_i - \lambda_i \vec{w}_i = 0, \tag{8}$$

where we have used the preprocessed data matrix. This will give us the set of eigenvectors $\vec{w}_i$ and their corresponding eigenvalues λ_i. It is useful to order the values of λ_i in a decreasing fashion and order the eigenvectors $\vec{w}_i$ in a $m \times m$ matrix $\mathbf{W}$, accordingly. The higher the value of λ_i, the more variance is along that component. Usually, one refers to the normalised eigenvalues

$$\tilde{\lambda_i} = \frac{\lambda_i}{\sum_{k=1}^{m} \lambda_k} \tag{9}$$

or better known as the *explained variance ratio*, which quantifies the ratio of the total variance in the data along a specific eigenvector $\vec{w}_i$. Because of the normalisation one can immediately observe the percentage of the entire variance in the data along a certain component. The transformed data set $\mathbf{X*}$ then follows from projecting the original data set onto the new basis $\{\vec{w}_i\}$:

$$\mathbf{X*} = \mathbf{XW} \tag{10}$$

.

2.3.2 Neural Network

A neural network (NN) is a model which depend on many parameters, also called weights and biases, whose number depends on the architecture of the NN. The NN is trained by adjusting the weights and biases in such a way that the correct output is generated by the NN according to the labelled data. When a NN is trained it can be employed to classify unlabelled data. A NN loosely resembles the network of neurons that makes up the human brain (See the right hand side of figure 4 for an example of the architecture of a NN). It consists of a certain amount of interconnected layers with in each layer a set of neurons

[1] Formally the principal components are the variables of the transformed matrix $\mathbf{X*}$, and are a linear combination of the original variables of $\mathbf{X}$ where the weights are determined by the eigenvectors. Here we use the terms principal components and eigenvectors exchangeably for they roughly indicate the same thing; the direction with the most variance.

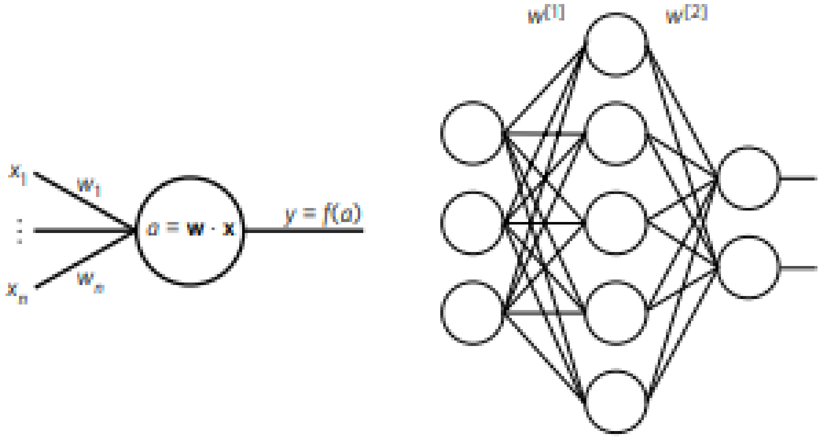


Figure 4: A neuron and an example of a neural network with 1 hidden layer.

that pass information forward. A neuron does this by considering all the input values x_i, a bias b and the values of the weights w_i of the connections to the neurons of the previous layer. The weights are a unique value that quantifies the strength of the connection between two neurons. A neuron from one particular layer is connected to every other neuron of the previous and the next layer. The neuron calculates $\sum_i x_i w_i + b$. This result is then evaluated on an activation function and passed to the next layer.

The activation function works like a gate. It determines if a neurons input is relevant for the task of the NN and if the neuron should pass that information onto the next layer. It can be as simple as a step function that turns the neuron output on and off, depending on a threshold. A rectified linear function is used in this thesis. This activation function is a piece-wise linear function that will output the value of the input if it is positive, and output zero otherwise. The same algorithm is performed in every neuron. The exception being the input layer neurons, these neurons only pass the value of the input data to the next layer.

The weights, together with the biases, are the free parameters that need to be trained. This process of training a model with labelled data is called supervised machine learning and the tweaking of the weights is done with back-propagation. Back-propagation is an algorithm that minimises the amount of error in the predictions of our NN. It achieves this by a gradient descent approach: calculating the gradient of the loss-function with respect to the weights and updating the weights such that the loss-function is minimised. The loss-function is a quantification of the amount of error in the NN. One can visualise the process of minimising the loss function of the NN as a hiker in a mountainous region looking to find the lowest point. When the hiker is in this area its horizontal coordinates are a combination of different values of the weights and the vertical coordinate is the value of the loss-function. The higher the hiker is on the mountain the higher the value of the loss function. The higher the value of the loss function the higher the amount of error in your NN. The shape of the loss-function (i.e. the mountainous area) is different for every problem. In order to be able to optimise the NN it needs a feedback to learn from, therefore a loss-function must be chosen. This can be a challenging problem as the function must capture the properties of the problem and must be motivated by concerns that are important to the project. In this thesis we use a logarithmic loss function (or log loss in short). This function suits best to our classification problem. It is defined as:

$$J = -t \log y + (t-1) \log 1 - y) \tag{11}$$

where t is a binary indicator (0 or 1) of whether observation o belongs to one class or the other and y the model's predicted probability that observation o is of one class or the other. Each predicted probability is compared to the actual class output value (0 or 1) and a score is calculated that penalises the probability based on the distance from the expected value. The penalty is logarithmic, offering a small score for small differences (0.1 or 0.2) and an enormous score for a large difference (0.9 or 1.0).

To find the best values for the weights w_i and bias b that minimises J, we update w_i and b according to:

$$w_{new} = w_{old} - \alpha \frac{\partial J}{\partial w} \tag{12}$$

and

$$b_{new} = b_{old} - \alpha \frac{\partial J}{\partial b} \tag{13}$$

with α defined as the learning rate. The learning rate can be changed to optimise the gradient descent. If α is too large, the gradient descent may overshoot the minimum and may fail to converge to a minimum value of J. However, when α is too small, the minimisation may take too long.

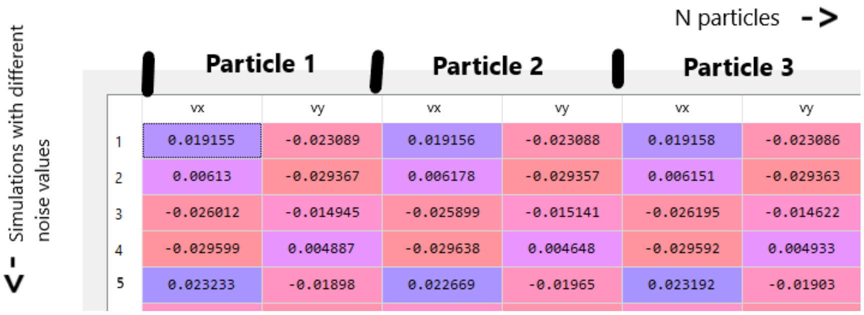
3 Methodology

3.1 Introduction to the method used

The goal of this thesis is to use machine learning to locate a phase transition in the VM, and assess its performance. The Vicsek model as described in section 2.1 has been used. Locating a phase transition means in this context performing a PCA analysis to try and reproduce the order parameter v_a as a function of the noise η, and applying a NN to try and find values of the critical values of the noise η_c. With values for η_c for different system sizes N a finite size scaling analysis can be performed to extract values of the critical exponents. Furthermore, in order to check if the critical noise values that the NN finds are reasonable the susceptibility χ as a function of the noise η is calculated.
The NN consists of 100 hidden layer neurons with a rectified linear activation function. A learning rate of $\alpha = 0.001$ with a stochastic gradient descent learning algorithm is used. Here we used a confusion scheme taken from [16] to pinpoint the critical point. It works as follows. We have data that depends on the noise parameter that lies in the range (a, b), and we assume that there exists a critical point $a < c < b$ such that the data can be classified into two groups. However, we do not know the value of c. We propose a critical point c', and train a network by labelling all data with noise parameter smaller than c' with label 0 and all the data with a higher noise parameter than c' label 1. Next, we evaluate the performance of the neural network on the entire data set and refer to its total performance, with respect to the proposed critical point c', as $P(c')$. We will find that the function $P(c')$ has a universal W-shape, with the middle peak at the correct critical point c. Finally, for the finite scaling analysis we use two methods to find a first estimate of the critical noise values. With these values we can tweak the values of the exponents and the critical value of the noise by hand to create the best overlap of data around the critical point. We use the NN confusion scheme to find critical noise values and we can take the noise values related to the peaks in the susceptibility χ as function of the noise η plots.
For PCA, the data matrix as explained in section 3.2 is used. See also figure 5. We used the v_x and v_y of the particles, because they contained the relevant information of the motion. The positional data of the particles would not have added information in the analysis of the velocity dependent order parameter.

3.2 Data generation

We perform simulations of the Vicsek model and collect the final configuration for many different value of noise η. We use N particles in a box of size $L \times L$ of density $\rho r^2 = 4$. All the simulations were done for $v\Delta t/r = 0.03$ and sufficient time was used to equilibrate the system, i.e. the particles interact with each other to lose their correlations from their initial configuration. The time it took to equilibrate was different for every system and was guessed on the basis of visualisation. We employ periodic boundary conditions in both directions. Data is generated for a number of different system sizes, but in this thesis we will focus on systems of: $N = 100$, $N = 1000$, $N = 3000$ and $N = 10000$. An example of a data table can be seen in figure 5. The columns of this matrix consists of the v_x and v_y component of all the particles present in that particular simulation. The values for v_x and v_y that are used are the



	Particle 1		Particle 2		Particle 3	
	vx	vy	vx	vy	vx	vy
1	0.019155	-0.023089	0.019156	-0.023088	0.019158	-0.023086
2	0.00613	-0.029367	0.006178	-0.029357	0.006151	-0.029363
3	-0.026012	-0.014945	-0.025899	-0.015141	-0.026195	-0.014622
4	-0.029599	0.004887	-0.029638	0.004648	-0.029592	0.004933
5	0.023233	-0.01898	0.022669	-0.01965	0.023192	-0.01903

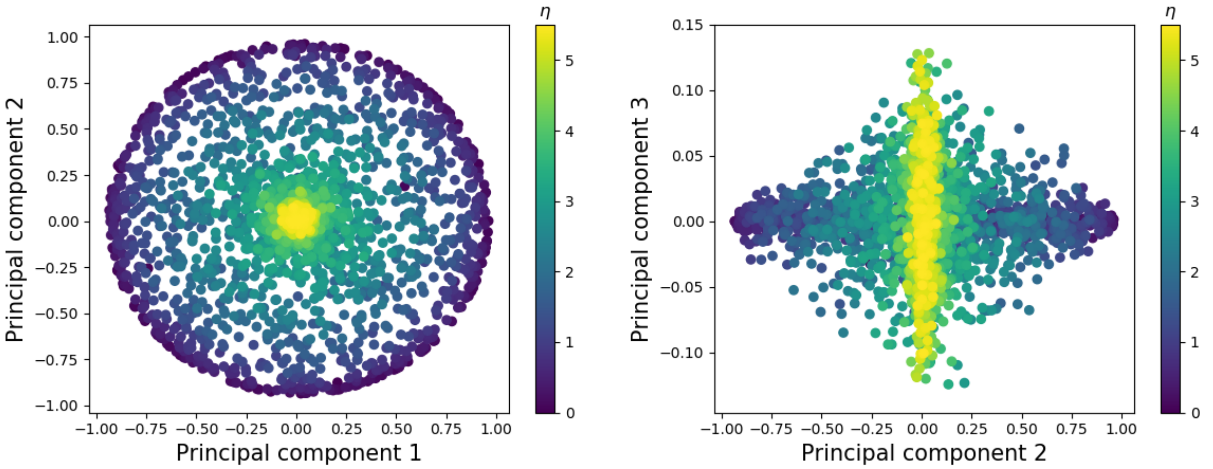
Figure 5: Part of a typical data matrix used in this thesis

values of the final state of the system. Every row in the matrix is a separate Vicsek model simulation. The simulations differ in the value of the noise parameter η. However, some data matrices contain multiple simulations for the same noise parameter to increase the amount of the data available for analysis. We chose to use one configuration per simulation to ensure uncorrelated data.

4 Results and Discussion

4.1 PCA

We start with a PCA for the system size $N = 1000$ with a fixed density $\rho r^2 = 4$. The data set used is that of section 3.2, where the noise parameter η is varied. The same analysis is performed for $N = 100$, $N = 3000$ and $N = 10000$, but as they gave roughly the same results it is clearer to highlight one system and add comments along the way. The resulting transformed data is plotted in figure 6. Every point stands for a simulation of the VM for a particular value of the noise η.

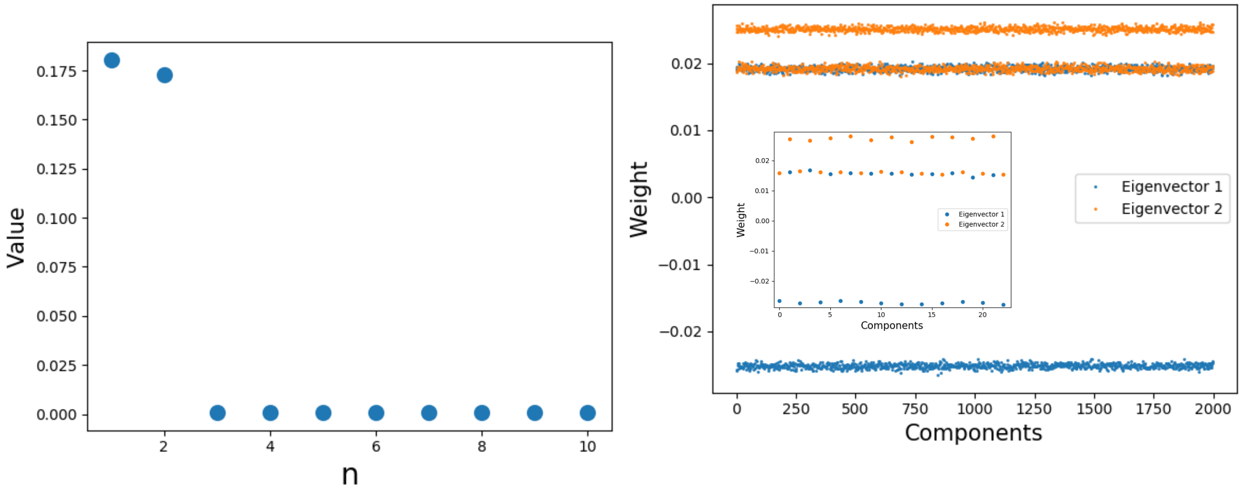


(a) First and second principal component

(b) Second and third principal component.

Figure 6: Transformed data for $N = 1000$ plotted along different principal components.

From figure 6a it is evident that there is a ring of low noise simulations and a cluster in the middle where all the high noise simulations reside. The more distinct the two clusters are, the sharper the transition between the ordered and disordered phase. Figure 6b shows the projection on the second and third principal component. Here the approach of ref. [17] for the Ising model was tried, where the other eigenvectors contained information about the variation of the order parameter. For the PCA on the VM a similar result could not be found. Furthermore, the 10 largest eigenvalues and the corresponding eigenvectors of the 2 largest eigenvalues are plotted in figure 7. Interesting to note here is that only the first couple of eigenvalues seem to contain most of the variance in the data. The amount of variance in these first 2 eigenvalues roughly adds up to 40%. Nonetheless, we will argue that this 40% variance is able to identify the important physics every time. We start with our discussion of figure 6. If we assume that the first principal component plots the net y-direction of all the particles in a simulation and the second principle component plots the net x-direction of all the particles in a simulation, the shape can then be explained as follows. When we examine the simulations for the lower half of the noise values we find that collective motion occurs. Every particle in the system groups together and moves collectively in one direction. This direction can be in any direction. Furthermore, the maximum alignment one can have



(a) Eigenvalue λ_n of principle component n.

(b) Weights of the components of eigenvectors 1 and 2. The inset shows a zoom in of the data.

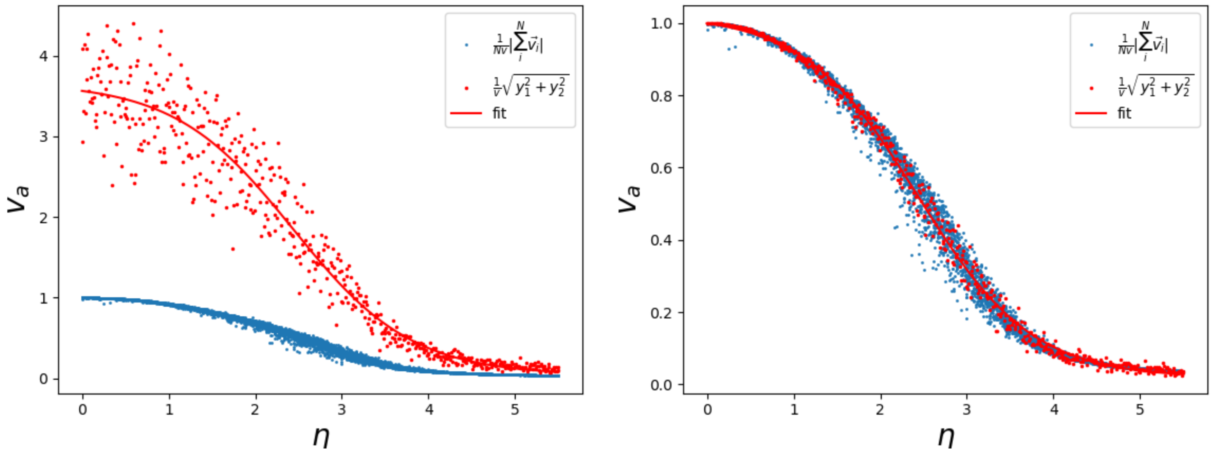
Figure 7: Eigenvalues and eigenvectors output from PCA.

in a system is when the directions of all the particles align. This explains the boundary of the circular shape, because that is where the completely ordered systems are plotted. The points in the plot, which represents the simulations which contain more noise, are closer to the centre than the low noise systems, because the high noise simulations do not (or have much less) have a net directional order. This is because all the directions of the individual particles cancel each other. No net directional order will occur in the y-direction or in the x-direction and the point which corresponds to this particular high noise simulation will be placed in the centre of the plot.

Furthermore, the eigenvectors that PCA returns further strengthens this reasoning. PCA finds that all particles are equivalent, but on close inspection of figure 7b one sees that the first eigenvectors finds a relatively high value for all v_x inputs and a low value for all the v_y inputs of the particles. While the second eigenvectors has exactly the opposite characteristic, every v_y has a high importance and every v_x has a relatively low importance. PCA splits all the input data in a v_x favoured eigenvector and a v_y favoured eigenvector. This can roughly be interpreted as an average of all the x components and an average of the y components over all the inputs. In order to check if this reasoning is correct we can compare a calculated order parameter against the results of PCA. The usual order parameter for the VM and the one we used is defined as the absolute value of the sum over all the velocity vectors of the particles. This can be decomposed as a sum over the separate components, see equation 14.

$$v_a = \frac{1}{Nv}\Big|\sum_{i=1}^{N} \vec{v_i}\Big| \simeq \frac{1}{Nv}\sqrt{(\sum_{i=1}^{N} v_{i,x})^2 + (\sum_{i=1}^{N} v_{i,y})^2} \tag{14}$$

In order to compare the PCA data with the calculated order parameter we need to construct an order parameter from the outputs of PCA. Although the shape of the eigenvectors suggest



(a) PCA order parameter without filtered eigenvectors

(b) PCA order parameter with filtered eigenvectors

Figure 8: PCA order parameter compared to the calculated order parameter. The fit serves as guide to the eye.

a straightforward substitution into equation 14 for the sum over all the v_x and for the sum over all the v_y. Only using the eigenvectors alone is not sufficient. The eigenvectors just form a basis and to give them physical meaning one has to project the original data onto this new basis. A natural way to do this, is by defining $y_1 = \vec{x} \cdot \hat{w}_1$ and $y_2 = \vec{x} \cdot \hat{w}_2$, with $\vec{x}$ a row from the original input data matrix $\mathbf{X}$ and $\hat{w}_1 = \vec{w_1}/|w_1|$ and $\hat{w}_2 = \vec{w_2}/|w_2|$ the first and second normalised eigenvector. From here it is possible to substitute y_1 and y_2 for $\sum_i^N v_{i,y}$ and $\sum_i^N v_{i,y}$ respectively in equation 14. The results can be seen in figure 8a. Already interesting to note here is that the shape of the red PCA data roughly resembles the shape of the blue calculated order parameter data. Only the PCA order parameter data has trouble converging to $v_a = 1$ for low values of η. This suggests that somehow the data is not correctly normalised, however on the other side of the noise spectrum the large noise data points do neatly converge to $v_a = 0$. Furthermore, the large spread of the data around the fit curve hints at the presence of noisy data. Look at figure 7b again, here we see that the eigenvectors not purely represent only v_y or only v_x. For example, every uneven component of eigenvector 2, which represent the v_x entries of all the particles in the system, have a non-zero value. When in the definition of y_1 the dot product between x and w_1 is performed all these non-zero values of the v_x are added as well. This could explain the noise and wrong normalisation for v_a around the low noise regime. To illustrate the PCA order parameter in the absence of noise, one can manipulate the data such that the v_x entries in the second eigenvector and the v_y entries in the first eigenvector have a value of zero. The result can be seen in figure 8b. Now the PCA order parameter perfectly coincides with the calculated order parameter.

The same analysis is performed for the other system sizes. The most important results can be summarised as follows. In the transformed data plots (like figure 6) we found that the larger systems, $N = 3000$ and $N = 10000$, have a smaller and more distinct cluster in the centre.

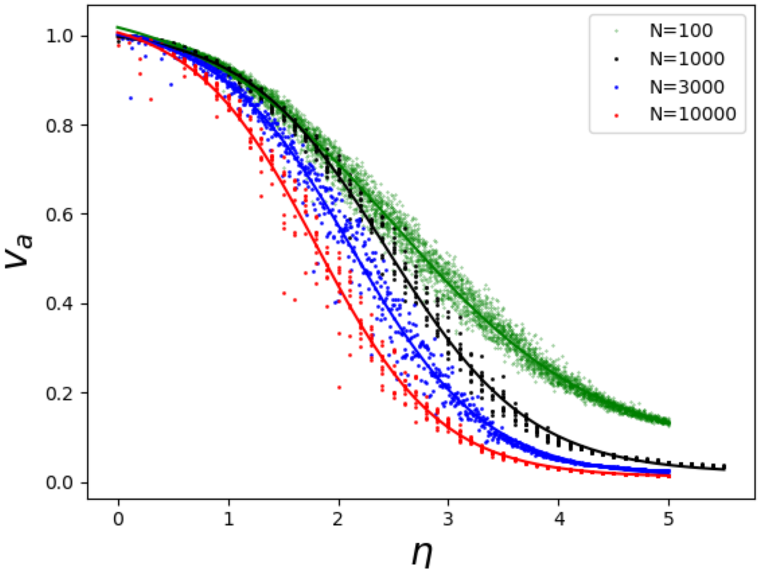


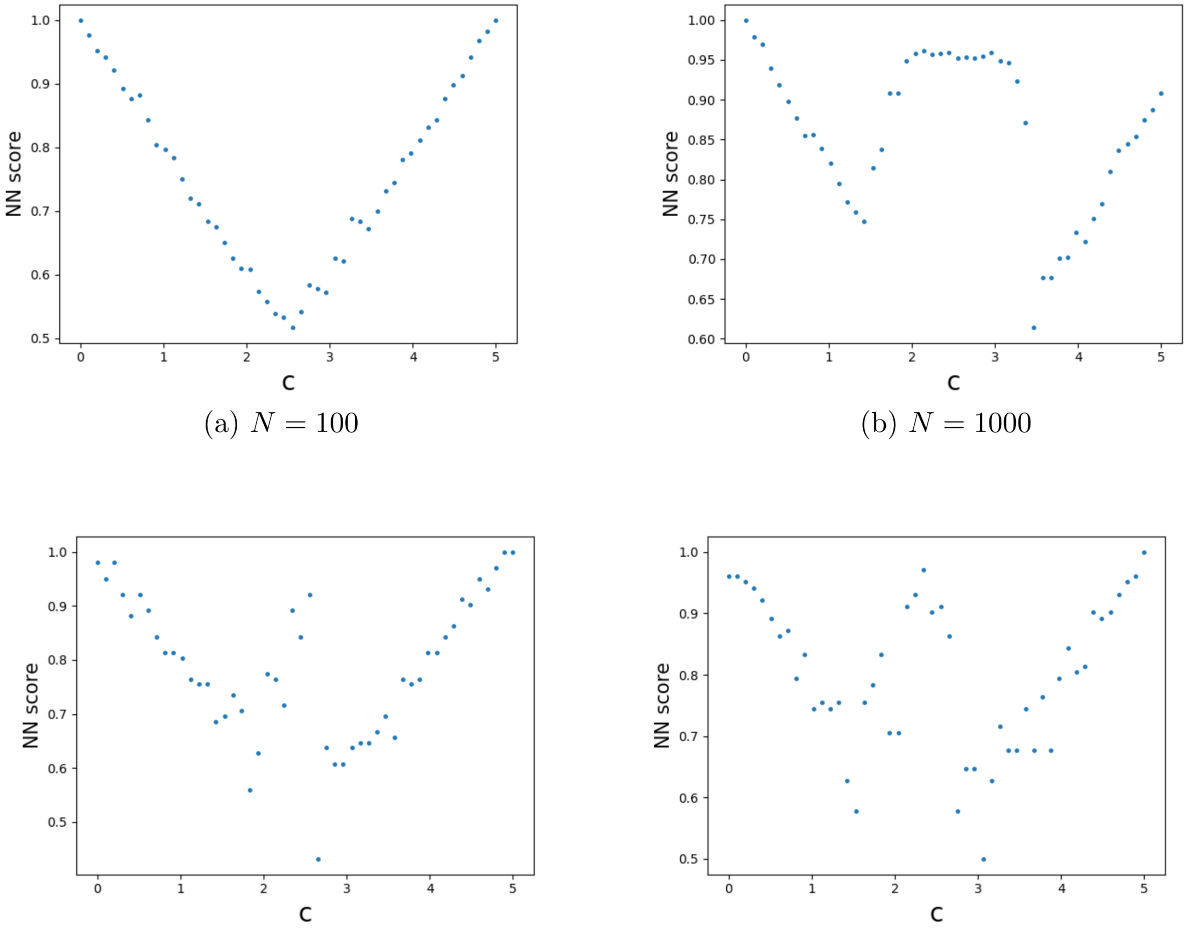
Figure 9: Order parameter for system size $N = 100$, $N = 1000$, $N = 3000$ and $N = 10000$ as a function of the noise. Fit serves as guide to the eye.

The smaller system $N = 100$ had a comparatively more spread out centre. Furthermore, the large system sizes had less points in the low noise regions. This can be explained by the differences in the shape of the order parameter as a function of the noise. See figure 9. The first thing we see is that for $N = 100$ in the high noise regime the order parameter does not converge to 0. Instead, the order parameter keeps a low, but non-zero value. This means that even at significant amount of noise there remains a certain degree of order in that VM. We also see this in the transformed data plot where the centre (high noise regime) is more spread out. For systems with more particles this characteristic is not present. Furthermore, one can also see that the larger systems fall off earlier from a completely ordered phase. This is because the density is kept constant at $\rho = 4$. When the amount of particles is increased the width of the box L is increased too. However the interaction radius r of the particles stays constant. As a result of this the particles interact with a smaller percentage of the area inside the box and complete alignment is harder. Even the smallest noise is than able to disorder the system. We also see this in the transformed data plots where there are less points in the low noise regime.

Finally, we found that the shape of the PCA order parameter depends on a combination of noise step size in the data and the system size N. For certain combinations PCA was able to find an order parameter that perfectly overlaps with the calculated order parameter before removing the noise from the data. Why this phenomena happens is not yet clear, this has to be examined more closely.

4.2 NN

The results of the confusion scheme for the method and neural network described in section 3.1 can be seen in figure 10.



(a) $N = 100$

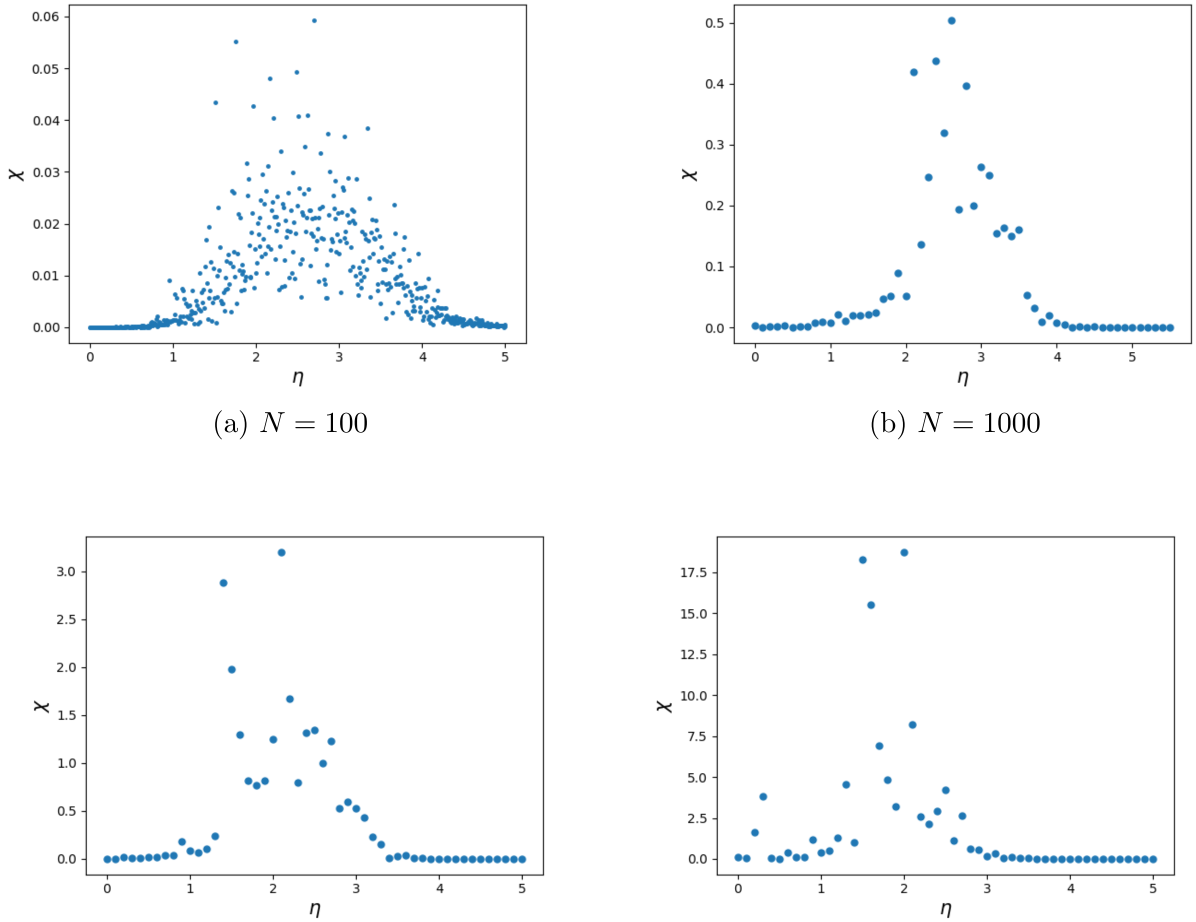
(b) $N = 1000$

(c) $N = 3000$

(d) $N = 10000$

Figure 10: Confusion scheme results for $N = 10000$, $N = 3000$, $N = 1000$ and $N = 100$.

Here we roughly observe the predicted 'W'-shape for $N = 10000$, $N = 3000$ and $N = 1000$, but not for $N = 100$. The results can be understood as follows. Assume that the system has two different phases. One for the values below the critical point c and one above c, and that the NN is able to find and distinguish them. When we set $c' = a$, and train the NN with these labels the NN will learn that only one class exists and chooses to assign that corresponding label to all data and thus correctly predicts 100% of the data. An analogous analysis applies to $c' = b$, but now every data point has the other of the two labels. When $c' = c$, the correct labelling, the NN will correctly learn to distinguish the two different phases and is able to assign the right label to both sides of the critical point and again performs perfectly. When $a < c' < c$, the NN will see data of the same phase in the ranges from a to c' and from c' to c, but having different labels. As a result, the NN will get "confused" and perform worse. The



(a) $N = 100$

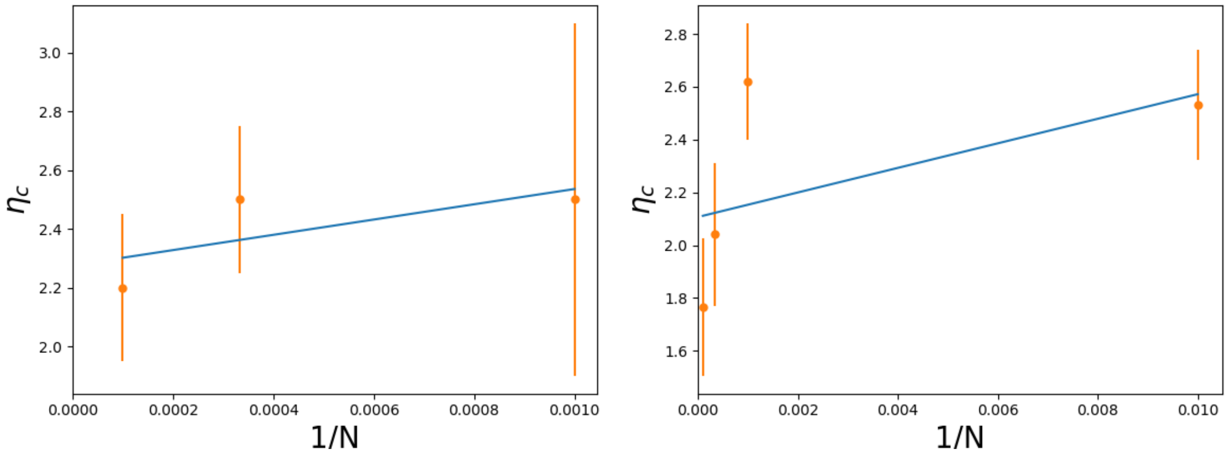
(b) $N = 1000$

(c) $N = 3000$

(d) $N = 10000$

Figure 11: Susceptibility χ as a function of the noise for several system sizes N.

same reasoning holds for $c < c' < b$. We see this behaviour for the $N = 3000$, $N = 10000$ and roughly for $N = 1000$ system sizes, but not for the smaller $N = 100$ system. All the $N = 100$ data looks so similar according to the NN that it isn't able to distinguish between 2 different phases and no clear critical point can be found. However when we examine figure 9 it is clear that a phase transitions exists. Apparantly the NN finds that it does matter whether v_a goes to 0 at large noise or not. We found this not only for $N = 100$, but for several other systems smaller than $N = 1000$. Furthermore, note that the NN accuracy in the middle peak of the 'W'-shape for the $N = 1000$ system stays the same between approximately $c = 2$ and $c = 3$. This signifies that the NN thinks that all these values for c could work equally well as a critical point for that particular system size. In order to check if the NN results are reasonable we can use the order parameter plots as a function of the noise for these particular system sizes. This can serve as an indication of the behaviour of these respective systems. The order parameter can be seen in figure 9. Here we see that the smallest system size $N = 100$ has less of a distinctive phase transition than the larger systems. The fact that the value of the order parameter does not converge to 0 highlights this (no complete disorder is



(a) Critical noise analogue η_c taken from figure 10.

(b) Critical noise analogue η_c taken from figure 11.

Figure 12: Critical noise analagoues η_c plotted as a function of the inverse system size $1/N$.

present in the system). On the contrary, we see that the other systems have a more distinct phase transition because the order parameter reaches zero. However, the amount of order in the $N = 3000$ and $N = 10000$ systems decreases earlier when increasing the noise value compared to $N = 1000$. As a result of this we can split up the curve in figure 9 up in two parts. Data which can be categorised in the slope and the data that makes up the tail for low noise. The NN probably makes use of this asymmetry, because the critical noise value which is found lies at the boundary of the low order simulations in the order parameter plot. Finally, the $N = 1000$ data can be explained with a similar reasoning. Where we split up the curve in in figure 9 for $N = 3000$ and $N = 10000$ in two parts we can split the curve for $N = 1000$ up in three: the ordered simulations for low noise, the disordered simulations for high noise and the slope for the middle ranges of the noise. Note that the order parameter curve for $N = 1000$ is symmetric and consists of roughly equal amount of ordered and disordered data points. When the NN tries to classify this in two distinct phases it is not able to find one concrete line that distinguishes the data, but finds a whole range of suitable points. This is because the NN somehow notices that the important parts are the order and disordered clusters and that the data points from the slope only serve as a cross-over between these two phases, and are unimportant. Therefore the NN detects that every noise value in the cross-over region could equally work as well for value of the critical noise.

From the plots in figure 10 we can crudely read of values of the critical noise of the respective system sizes. With this information we can perform a finite size scaling analysis, and it is possible to estimate the critical value of the noise in the thermodynamic limit by using

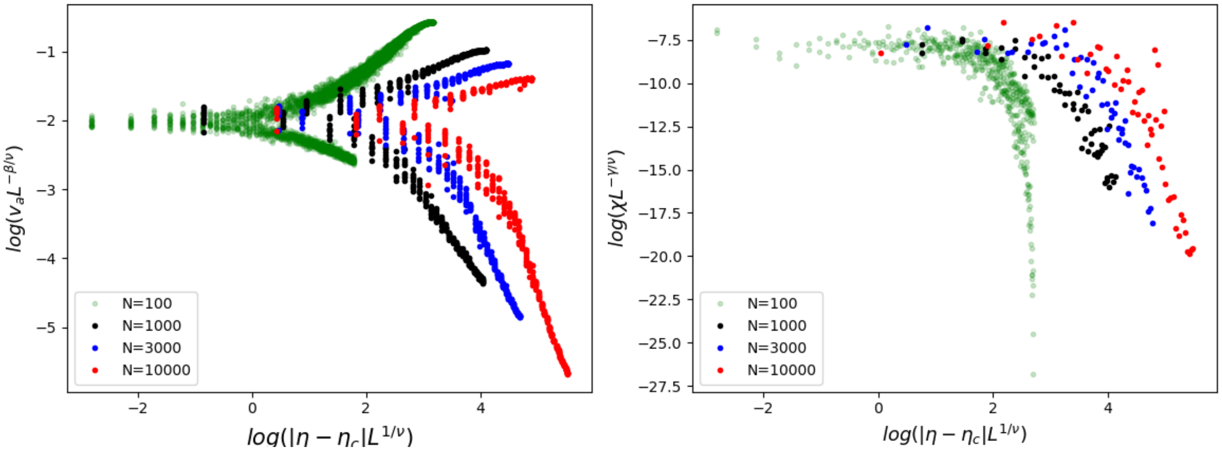
$$\eta_c' = \eta_c + kN^{-1}. \tag{15}$$

If we take the critical values of the noise from figure 10 and plot them as a function of $1/N$ we can estimate the fit parameters η_c and k. We use $1/N$, because we are interested in the thermodynamic limit $1/N \to 0$. This is because most systems consists of around $N \sim 10^{23}$

particles so we are interested in the values of that limit. The results are shown in figure 12a. The y intercept of the fit for the NN critical noise values is $\eta_c(\infty) = 2.28 \pm 0.16$.
To check this result a finite size scaling was also performed by using the critical noise values read of from the susceptibility plots as a function of the noise. See equation 4. Around the critical point high fluctuations in the order parameter should be expected. Results can be seen in figure 11. Every point represents an average over 10 different VM simulations. If we read of the critical noise values from the plot of figure 11 and use them to perform a similar finite size scaling analysis as is done with the NN we find that $\eta_c(\infty) = 2.11 \pm 0.25$. The plot is showed in figure 12b. Note that for $N = 100$ no clear peak is found.

4.3 Critical exponents

Finally we extracted values for the critical exponents β, γ and ν. We have plotted $\tilde{\chi}$ and $\tilde{v_a}$ as a function of $|\eta - \eta_c|L^{1/\nu}$ as is written down in equations 5 and 6. The results can be seen in figure 13. The data has been plotted with the values $\beta = 0.3$, $\gamma = 2.1$ and $\nu = 0.9$. These values represent the best overlap of data around the critical point and were determined by hand. Log-log plots have been used for clarity. As can be seen in the figures the data from the different systems sizes overlap around the critical point. The critical point is when $|\eta - \eta_c| \to 0$, so when the natural logarithm goes to $-\infty$. Because the deviation of the values of η_c from the real values of the critical noise, and because the step size of the noise is large, $|\eta - \eta_c|$ won't actually come close to reaching zero and thus the natural logarithm of that value won't come close to $-\infty$. We see this effect in the systems larger than $N = 100$, because larger step sizes of the noise were used there.



(a) log-log plot of $\tilde{v_a}$ as a function of $|\eta - \eta_c|L^{1/\nu}$

(b) log-log plot of $\tilde{\chi}$ as a function of $|\eta - \eta_c|L^{1/\nu}$.

Figure 13: $\tilde{\chi}$ and $\tilde{v_a}$ as a function of $|\eta - \eta_c|L^{1/\nu}$ for system sizes $N = 1000$, $N = 3000$ and $N = 10000$.

5 Conclusion

A PCA analysis of the VM is successfully able to find a phase transition. The transformed data showed that a low and high noise cluster is present in the system, and the eigenvectors and eigenvalues showed that it is able to detect an order parameter. Because for some combinations of N and noise step size the results of the PCA order parameter where very noisy, some PCA order parameters were better than others in reproducing the calculated order parameter. It is not yet clear why this happens, but most likely this behaviour can be attributed to the growth in the amount of noise in the data when the system size is increased. Furthermore, it is showed however that PCA was still able to find the correct shape of the order parameter in the noisy data. This was done by filtering out the noise from the PCA eigenvector outputs. Finally, it is worth mentioning that PCA found this result by only projecting less than 40% of the variance in the data. For some combinations of the system size N and noise step size this 40% was enough to find a perfect reproduction of the order parameter, for other systems the noise had to be removed by hand. If only 40% is used one could say that still a lot of information is embedded in the other principal components. This is probably not the case, because the VM is inherently a noisy system and the manner of how the data is chosen this noise is carried over into the PCA analysis. PCA had a hard time finding more complex results in this noisy environment. Still it would be interesting to see if there exists more information. This would have to be examined more closely.

The second part of this thesis consisted of an analysis of the VM with a NN confusion scheme. It was showed that for larger systems $N = 1000$, $N = 3000$ and $N = 10000$ a phase transition could be detected. From the characteristic 'W'-shape result of this analysis a value of the critical noise η_c could be read off and used for finite size scaling. For $N \to \infty$ the noise value $\eta_c(\infty) = 2.28 \pm 0.16$ is found. The large error is attributed to the informal way of guessing the critical value of the noise by eye. This value is in the same range as $\eta_c(\infty) = 2.11 \pm 0.25$, which was found from the susceptibility analysis. This, together with the knowledge of the literature value of $\eta_c(\infty) = 2.14$ for $\rho = 4$ and $\upsilon = 0.03$, shows that the NN confusion scheme is able to correctly find a phase transition considering a large uncertainty. However, interesting to note is that the NN struggled to find a phase transition for smaller system sizes where there was still order in the high noise regime. Finally, by using the critical values of the noise that the NN found for the different system sizes N, the critical exponents $\beta = 0.3$, $\gamma = 2.1$ and $\nu = 0.9$ were extracted. This is in almost perfect accordance with ref. [15] for the same VM parameters.

Although the amount of data that was used in this thesis is inherently noisy and sparse compared to the size of the data sets used in the literature, it is evident from the results that the machine learning techniques PCA and a NN confusion scheme are still able to detect phase transitions and reasonable values of the critical noise in the VM. Considering this, we can conclude that these machine learning algorithms are suitable for the identification of phase transitions and finding order parameters in the VM. It would be interesting to see how a combination of the methods, or a nonlinear dimensionality reduction method like an ISOMAP [18] could be used to improve the results. Finally, one should ask how the results in this thesis compare for different variations of the VM? Does changing parameters, adding external forces and individualistic behaviour make it harder to detect phase transitions or will it give new surprising insights? Only future research will tell.

References

[1] Simon Hubbard, Petro Babak, Sven Th Sigurdsson, and Kjartan G. Magnússon. A model of the formation of fish schools and migrations of fish. *Ecological Modelling*, 174(4):359–374, jun 2004.

[2] Craig W. Reynolds. FLOCKS, HERDS, AND SCHOOLS: A DISTRIBUTED BEHAVIORAL MODEL. *Computer Graphics (ACM)*, 21(4):25–34, aug 1987.

[3] Dirk Helbing, Illés Farkas, and Tamás Vicsek. Simulating dynamical features of escape panic. *Nature*, 407(6803):487–490, sep 2000.

[4] Erik M. Rauch, Mark M. Millonas, and Dante R. Chialvo. Pattern formation and functionality in swarm models. *Physics Letters A*, 207(3-4):185–193, oct 1995.

[5] Eshel Ben-Jacob, Inon Cohen, Ofer Shochet, Adam Tenenbaum, Andrs Czirk, and Tams Vicsek. Cooperative formation of chiral patterns during growth of bacterial colonies. *Physical Review Letters*, 75(15):2899–2902, 1995.

[6] John Toner, Yuhai Tu, and Sriram Ramaswamy. Hydrodynamics and phases of flocks.

[7] Tamas Vicsek, Andras Czirok, Eshel Ben-Jacob, Inon Cohen, and Ofer Shochet. Novel type of phase transition in a system of self-driven particles. Technical report, 1995.

[8] Juan Carrasquilla and Roger G. Melko. Machine learning phases of matter. *Nature Physics*, 13(5):431–434, may 2017.

[9] Francesco Ginelli. The Physics of the Vicsek model. *Eur. Phys. J. Special Topics*, 225:2099–2117, 2016.

[10] Dorilson S. Cambui, M. Godoy, and A. S. de Arruda. Finite-size effects in simulations of self-propelled particles system. *Physica A: Statistical Mechanics and its Applications*, 467:129–136, 2017.

[11] H Chaté, F Ginelli, G Grégoire, F Peruani, and F Raynaud. Modeling collective motion: variations on the Vicsek model. *Eur. Phys. J. B*, 64:451–456, 2008.

[12] Gabriel Baglietto, Ezequiel V Albano, and Julián Candia. Gregarious versus individualistic behavior in Vicsek swarms and the onset of first-order phase transitions. *Physica A*, 392:3240–3247, 2013.

[13] Gabriel Baglietto and Ezequiel V Albano. Nature of the order-disorder transition in the Vicsek model for the collective motion of self-propelled particles.

[14] M. Aldana, H. Larralde, and B. Vázquez. On the emergence of collective order in swarming systems: A recent debate. *International Journal of Modern Physics B*, 23(18):3459–3483, jul 2009.

[15] Dorilson S Cambui, Alberto S De Arruda, and Maurício Godoy. Critical exponents of a self-propelled particles system. *Physica A*, 444:582–588, 2016.

[16] Evert P.L. Van Nieuwenburg, Ye Hua Liu, and Sebastian D. Huber. Learning phase transitions by confusion. *Nature Physics*, 13(5):435–439, may 2017.

[17] Rebelo A. Unsupervised learning of physical models: Uses and limitations of principal component analysis. Master's thesis, Utrecht University, 2017.

[18] Joshua B Tenenbaum, Vin de Silva, and John C Langford. A Global Geometric Framework for Nonlinear Dimensionality Reduction. Technical report, 1995.