Master of Science in Mathematical Statistics

Dissertation:

Introduction to fast Super-Paramagnetic Clustering

Lionel Yelibi

Supervisor: A/Prof. T. Gebbie

Department of Statistical Sciences,

University of Cape Town, Rondebosch

(Dated: July 20, 2019)

Declaration

I declare that: “Introduction to fast Super-Paramagnetic Clustering” is my own work and that all the sources that I have used or quoted have been indicated and acknowledged by means of complete references.

Lionel Yelibi

Acknowledgments

The authors thank Etienne Pienaar, Nic Murphy and Micheal Gant for discussions and comments. LY would like to thank Fangqiang Zhu for first introducing him to the elegance of Ising models and their simulation. Computations were performed using facilities provided by the University of Cape Town's ICTS High Performance Computing team: hpc.uct.ac.za

Abstract

We map stock market interactions to spin models to recover their hierarchical structure using a simulated annealing based Super-Paramagnetic Clustering (SPC) algorithm. This is directly compared to a modified implementation of a maximum likelihood approach to fast-Super-Paramagnetic Clustering (f-SPC). The methods are first applied standard toy test-case problems, and then to a dataset of 447 stocks traded on the New York Stock Exchange (NYSE) over 1249 days. The signal to noise ratio of stock market correlation matrices is briefly considered. Our result recover approximately clusters representative of standard economic sectors and mixed clusters whose dynamics shine light on the adaptive nature of financial markets and raise concerns relating to the effectiveness of industry based static financial market classification in the world of real-time data-analytics. A key result is that we show that the standard maximum likelihood methods are confirmed to converge to solutions within a Super-Paramagnetic (SP) phase. We use insights arising from this to discuss the implications of using a Maximum Entropy Principle (MEP) as opposed to the Maximum Likelihood Principle (MLP) as an optimization device for this class of problems.

Keywords: maximum likelihood, Potts Models, unsupervised learning, clustering, maximum entropy

Contents

I. Introduction

We consider the problem of unsupervised statistical learning for the classification of financial trading and investment strategies. Concretely, we consider Potts model [73] based [7, 8] methods optimized for performance [24, 32, 33, 45] using a Maximum Likelihood Estimation (MLE) approach based on the ground-state Noh Ansatz [56] compared to the finite-temperature approach to select configurations based on the susceptibility [7, 8].

We compare the cluster configuration from the fast clustering algorithms based on the ground state configurations [32, 33] (Sec. G 2) with those based on finite temperature Simulated Annealing (SA) based Monte-Carlo Markov-Chain (MCMC) [36, 68] (see Sec. G 1) to generate the full dendrogram of cluster configurations [7, 8]. It was shown that the clustering structure of financial assets are time horizons dependent in [10], and given a proper translation of correlations into the Euclidean distance can be represented using Minimal Spanning Trees (MST) [39].

The Super-Paramagnetic Clustering (SPC) method originally developed in the early 90s is a universal data clustering method [7] and has acquired a certain popularity for its implementation of the Maximum Entropy Principle (MEP) [38] where no assumptions are made about the distributions found in the data, and the number of clusters is revealed rather than predefined. It can be used in any environment as long as the features are embedded in an appropriate similarity metric. SPC has been applied to chemical data using a sequential method and the Tanimoto similarity measure [60], the detection and classification of spiking activity in neuronal tissue in [62], the identification of regions of the brain with shared functionality in [67], yeast genes profiles in [23], histone modification data [41], and image segmentation in [1].

Marsili and Giada [24] were able to developed an efficient maximum likelihood clustering method for high dimensional data using the same spin-model inspired approach used in the SPC method. However, given the ill-posed nature of clustering they chose to evaluate the likelihood model L_c for robustness relative to different optimization methods in [25]. The method is then applied to the detection of clusters of assets, and financial markets states in [45] to uncover collective behavior of agents in a complex systems. Hendricks, Gebbie, and Wilcox created a GPU based Parallel Genetic Algorithm (PGA) implementation to maximize L_c in near real time clustering of market states in for quantitative trading applications [32].

In this work, some of which was previously implemented in [75], we explore the relationship between the SPC method of Domany-Wiseman-Blatt [7] as compared to that of Giada-Marsili [24] and we are able to confirm that the likelihood method is super-paramagnetic and conforms well to the entropy method when dimensionality is sufficiently high (see Sec. VI). We are also able to further optimize the PGA implementation [32]. However, what we really demonstrate is the variety of clustering problems that the SPC can quickly and easily handle, but with the advantage of leveraging a mature and well-understood foundational theoretical framework from statistical mechanics that has many, if not most, viable alternative algorithms, as

special cases. We are of the view that SPC type models grounded in the maximum entropy principle and within the general framework of energy-based machine learning offer a variety of research and development opportunities for building better and faster unsupervised learning algorithms.

The dissertation is organized as follows: Section II discusses a brief overview of Erdos-Renyi's Random Graphs (see Sec. II B), and their connection to the Potts models as special cases. We describe the inhomogeneous Potts Model as a data-clustering engine (see Sec. II C). In Section II D, the implementations with the SPC algorithm (Sec. II D 1) followed by maximum likelihood methods (Sec. II D 2). We then discuss the characteristics of the Potts model which motivate a departure toward maximum likelihood methods in Sec. II G 1, and we propose a fast, and inexpensive solution in Sec. II F. We then proceed to discussion validations procedures linking maximum likelihood methods to Super-paramagnetic clustering in Sec. II G. Section III goes over our choice of similarity metric (Sec. III A), data-preprocessing (Sec. III B), and time-series noise filtering (Sec. III C). Section IV provides toy test cases, and Section V stock market applications. In Section VI a summary analysis of the results and their implications, and finally in Section VII the conclusion, and potential directions for further research on similar topics are mentioned.

We promote the idea that a promising future research direction would be quantized spin-models and ultimately building unsupervised learning algorithms that more effectively accommodate state-interference and phase information within some likelihood method.

II. Potts Clustering

A. Inference in Statistical Mechanics, and Machine Learning

Techniques imported from statistical mechanics have by their flexibility been employed in problems of inference in domains going from mainstream physics such as the study of phase transitions in spin glasses systems. These systems are composed of spins which we consider random variables and their known symmetric pairwise interaction terms. Within this framework we are tasked with inferring the distribution of states of spins, and other variables which compress and capture sufficient statistical information from these systems. The topology of these systems is induced by the pairwise interactions. This poses theoretical challenges because exact solutions are typically intractable and impossible: The number of possible arrangements is a combinatorial optimization problem of the N^N order. In light of that fact, Mean-Field models have been developed [7, 24] which have in the limit of large systems allowed the determination of the critical temperature of the ising model [11], the Potts model [7], and the Giada Marsili likelihood model [24]. Additional challenges are computational because the simulation and optimization of such systems are rendered difficult: indeed the objective function of such systems, the Hamiltonian, has a rough landscape with many meta-stable fixed points. MCMC techniques like SA were developed to deal with these issues where the objective function is tempera-

ture constrained (this need not be the only constraint), the space of energy levels searched and the energy distribution of the system at the given temperature is recovered. Equivalently the space of spin configurations is searched and the appropriate distribution of states (the number of clusters, and their sizes) is inferred from the data. Within a given phase of the system the state (temperature) of maximum entropy is the most stable (optimal): In the Super-Paramagnetic framework this need not be the critical temperature although the optimal temperature will not be far from it.

A1	N data points mapped to N discrete spins $S \ : \ s_i = 0, ..., N$ on a fully connected graph.
A2	D features mapped to the interaction terms J_{ij}
A3	the Hamiltonian $H(J, S)$ is a function of the spin configuration S, and the interaction terms J
A4	Determine $\arg\min_H \ P(H\|S, J, T)$ which maximizes $\sum -P \ln P$ the entropy leading to $P(H) \propto e^{-H/T}$.

TABLE A: The optimization routine for spin systems such as as the Potts Model. The objective of the algorithm is the estimation of the distribution of energy levels for a given system.

"Modern" machine learning methods such as Neural Networks are fully included in the framework of Spin Glass models. The system consists of fully connected layers of neurons: input, hidden, and output layers. Feed-Forward Neural Networks are typically employed in the resolution of supervised learning problems such as prediction and classification. Restricted Boltzmann Machines (RBM), first invented by Smolensky [65], and popularized by a fast algorithm [34], are an example of such systems where the input are the features of the data points, the hidden layers are the representation or learned features of the data, and the output layers are predictions on the categorical or numerical variables (the states). The interaction terms are the weights assigned to the edges linking input to hidden layers (hidden layers are considered input layers to the output layers). The hidden layers neurons capture the collection of pairwise interaction terms through an activation function, and the objective function of the system is an energy (Hamiltonian) function which computes the error under an inverse temperature constraint. This temperature is typically near the "Curie" critical temperature for ising models (and RBMs) at which the entropy is maximal, and the system can reach its ground state (lowest) energy.

B1	D features mapped to D input layer neurons with states v_i, and connected to a hidden layer of N neurons with states h_i
B2	Weight matrix W mapping v to h neurons through an activation function.
B3	The output variable $y_i = f(\sum w_i x_i)$ with x_i the state of neurons at the previous (hidden) layer, and f an activation function.
B4	The Hamiltonian $H(y_i^*, y_i \vert W, v, h) = \sum [y_i^* - y_i]^2$ the residual sum of squares (RSS)
B5	Determine $\arg\min_H \ P(W)$ which maximizes $\sum -P \ln P$ the entropy leading to $P(W) \propto e^{-H(W)/T}$.

TABLE B: The optimization routine for a typical feed-forward Neural Network. The algorithm minimizes the residual sum of squares over the training set, and maximize the entropy of the distribution over the weights as to allow the possibility of noisy data.

Similarities between Table (A) and (B) illustrate how the inference process of Spin Glass models are intricately linked to that of Neural Networks. The interaction terms, and representation states are central to the inferential procedure: In our context, Spin Glass models like the Potts model consider the distance (or the strength) between data points as interaction terms, while the Neural Network described above maps interactions terms between individual features of the data. In both cases we are looking to find representations of shared states between the data points through the interaction terms. This concept has been further illustrated by the link between Renormalization Group Theory and Deep Neural Networks in [52] where the equivalency of both framework was shown: The minimization of the change in Free Energy when a system is traveled between spin lattices is equivalent to minimizing the Kullback-Leibler (KL) Divergence thus ensuring that the learned representations, and the distribution of states from one layer to another remains faithful and the fixed-points of the system are preserved. The preservation of this symmetry which, under increasing scale, maintains the invariance of the system fixed-points not only allows the effective compression of information thus reducing the complexity of interactions, and the study simpler versions of such systems.

B. Random Graphs

A graph is a mathematical model which formalizes pairwise relationships between objects [69]. Graphs are popular in complexity sciences because they provide a framework to model large systems of interacting components by representing the system directly through the pair-wise relationships between the components. The field has seen the rise of many models, each with their own assumptions and various

nuance, almost all are of the generative form based on the premise of bottom-up causation. One feature that is useful in our context is that one can observe certain types of emergent dynamics *i.e.* "phase transitions" [11].

A general class of models called the Random Cluster model [1] was developed by Fortuin-Kasteleyn in 1972 [19]. It is a "random" graph generated by a probability distribution

$$W(N) = \frac{q^{C_N}}{Z} \prod_{\langle i,j \rangle}^{N} P_{ij}^{n_{ij}} (1 - P_{ij})^{(1-n_{ij})} \tag{1}$$

where N is a given edge configuration (or adjacency matrix of n_{ij}), C_N is the number of clusters given N, P_{ij} is the probability of nodes i and j being connected, and Z is the normalization constant (also called the partition function in statistical mechanics [66]). The adjacency matrix is linked to the probabilities P_{ij} by picking a value P such that if $P_{ij} > P$ then $n_{ij} = 1$ or 0 otherwise. W is essentially the probability of the graph being connected given an adjacency matrix N.

If we now set $q = 1$, the random cluster model reduces to a Erdos-Renyi's random graph [14]. Given the existence of only one class on the graph, bonds are linked independently from their respective states (i.e. they all have the same state) with equal probability $P_{ij} = \frac{1}{n}$ with n the number of nodes on the graph [14]. An important generalization of this idea is the Barabási-Albert model [4]; this model works slightly differently: it starts with a low number of connected nodes m_0, adds new nodes one at a time, one new node is able to connect to $m < m_0$ nodes, and every time a node i is connected its degree k_i increases. The probability of a node connecting to another is $P_{ij} = \frac{k_i}{\sum k_j}$. This means as a node i makes connections it becomes "popular" and succeeding nodes have a higher likelihood of connecting to that same node: this is the principle of "preferential attachment" which hopes to explains how some social networks are formed. These models are all based on a generative model that builds on microscopic causal relationships from the system components to the bulk.

The Fortuin-Kasteleyn random cluster model is closely related to the Potts model via its distribution

$$P(S, N) = \prod_{\langle i,j \rangle} \left[(1 - P_{ij})(1 - n_{ij}) + P_{ij} n_{ij} \delta_{s_i, s_j} \right] \tag{2}$$

which is the conditional probability of the spin configuration S given the edge configuration N [13]. The marginal probability W is recovered by summing over all spin configurations. The major difference the Potts model brings is entropy maximization which assumes an exponential Boltzmann distribution of edges connections $P_{ij} = 1 - e^{-J_{ij}}$ with J_{ij} as the pairwise probabilities. The strength J_{ij} captures the closeness between nodes, and, with the clusters membership, defines the topology of the graph. It's a central variable of the model. This is similar to the Bianconi-

[1] See Grimmett [28] and references therein.

Barabási model [6] which introduces a fitness η_i which plays a related role as an add-on to the Barabási-Albert model [4].

C. The Potts Model

The Ising model [11] simulates the existence of phase transitions in large systems of interacting particles. The model consists in the representation of a n-Dimensional plane. Ising's PhD Thesis [11] solved the 1D problem, which showed no phase transition, while Onsager provided an analytical solution using a transfer-matrix method [58] for the 2D case. If we consider observations in our data sets as nodes with edges which link nodes together it becomes natural to consider the data-set in the context of a Potts model [73]. An edge is active or inactive with probability dependent on the distance between two nodes. The collection of nodes and edges form the graph which is navigated for clustering. Every node can be assigned, for example, a +1 or -1 spin (for the Ising model). Interactions are permitted by randomly changing the spin values in the graph, and then accepting or rejecting new configurations is implemented using the Swendsen-Wang MCMC algorithm at every step [68].

The Potts model [73] is a generalization of the Ising [11] model allowing the system to accept a higher q spin values instead of 2. The parameter q can be compared to the K value in K-means used to fix the number of clusters. q is the maximum number of classes: it must be chosen to be big enough to avoid clusters forcefully merged together. The only inconvenience to a relatively high q is the additional computational cost needed to perform the statistics after the system reaches thermal equilibrium.

The model is governed by a Hamiltonian Energy [2] equation [73]

$$H_S = \sum_{\langle i,j \rangle} J_{ij}(1 - \delta_{s_i,s_j}) \tag{3}$$

with: $S = [s_i, ..., s_N]$ the spin vector assigned to our data, spins $s_i \in [1, ..., q]$, and N nodes. The Kronecker delta which is 1 for equal spins and 0 otherwise. For data embedded in a metric space the Euclidean distance function $d_{ij} = ||x_i - x_j||$ is computed between two nodes.

d_{ij} is fed to the strength function which, in turn, measures similarity. Many models for strength exist but their central feature is they must decrease with distance. This is typically achieved with a function of the type $e^{-d_{ij}}$ or a power law as seen

[2] The more general Potts Hamiltonian can be contrasted with one of its special cases in the form of the energy of a Boltzmann machine with bias $\boldsymbol{b}$ and weight matrix W for features $\boldsymbol{x}$: $E_{b,W} = -\boldsymbol{b}^T\boldsymbol{x} + \boldsymbol{x}^T W \boldsymbol{x}$ with partition function $Z_S = \sum_x e^{-E_{b,W}(x)}$ [2, 35, 59, 64]

in [7]:

$$J_{ij} = \frac{1}{\hat{K}} \exp\left\{ -\frac{1}{2}\left[\frac{d_{ij}}{a}\right]^2 \right\} \tag{4}$$

where $\hat{K}$ is the average number of neighbors per node, and a is a local length scale: the average d_{ij} of all nearest-neighbors.

There are alternative choices for local characteristic length scale[7]. We only report the results obtained with the previous definition, and note that the adjustments to a are problem dependent: higher values of a ensure the 1$^{\text{st}}$ phase of the simulation is ferromagnetic while lower values start the simulation in the super-paramagnetic (SP) phase.

The objective is to compute averages of thermodynamic quantities after simulating the system at a given temperature for a set number of MCMC iterations M until thermal equilibrium.

The magnetization m of the system is given by

$$m(S) = \frac{qN_{max}(S) - N}{(q-1)N}$$

This quantity, which ranges from 0 to 1, expresses how dominated the system is by the largest cluster. The order parameter of the system is the average magnetization $\langle m \rangle$, and its variance $\chi_{\frac{T}{N}} = \langle m^2 \rangle - \langle m \rangle^2$ is called the susceptibility density. Both can be used to detect a phase transition: $\langle m \rangle$ dives down while χ peaks at every transition.

The first simulation serves to uncover the existence of a critical temperature T_c at which a first transition occurs. At $T < T_c$ all spins are strongly correlated, $\langle m \rangle \approx 1$ and all have the same state: It is called spontaneous magnetization (ferromagnetic phase). At $T = T_c$ the single cluster breaks down into smaller ones (SP-phase). Furthermore, inside the temperature range where the SP-phase exists, a system can go through additional transitions: These reflect the different hierarchical structures present in the data. Finally at $T \gg T_c$ we go through a final transition into complete disorder (Paramagnetic phase): The energy H_S is high, all clusters dissolve, and $\langle m \rangle \approx 0$.

For a quantity A the thermal average will be

$$\langle A(S) \rangle = \sum A(S)P(S). \tag{5}$$

Here each S represents a single MCMC step. If M is large enough, Eqn. (5) is equivalent to the arithmetic mean $\langle A(S) \rangle \approx \frac{1}{M}\sum_{i=1}^{M} A(S)$. The probability of a system being in a particular state (referring to the energy of the system H_S) is:

$$P(S) = \frac{e^{-H_S/T}}{Z} \tag{6}$$

where $e^{-H_S/T}$ is the Boltzmann factor, Z is the partition function $Z = \sum_S e^{-H_S/T}$ and the normalization constant of the Gibbs-Boltzmann distribution.

Numerically, we use a mean-field mode of the Hamiltonian such that $H_S = \frac{1}{N}\sum_{\langle i,j\rangle} J_{ij}(1-\delta_{s_i,s_j})$. The motivation being that high levels of H lead to Boltzmann factors close to 0, Z also ≈ 0 which by definition makes the computation of $P(S)$ impossible, also the value of H_S impacts the temperature range explored.

1. Maximum Entropy

We briefly remind ourselves of the MEP [38]. We define a statistical mechanical system as an ensemble of objects each in their respective micro-states (spin values s_i) so that the resulting in microscopic state of the entire system $S = [s_i, ..., s_N]$ can be used to derive parameters which characterize the distributions for macroscopic variables of interests (here the internal energy H_S). We assume that at equilibrium, thermodynamic systems obey conservation of energy which sets the constraints of the system such that on average H_S is a constant, and then from Eqn. ((5)) it follows that:

$$\langle H_S\rangle = \sum H_S P(H_S), \text{ and } \sum P(H_S) = 1. \tag{7}$$

We then consider that the distribution representative of the energy of the system as the one which incorporates our constraints and assumes nothing else. This maximizes the Shannon's Entropy as defined by:

$$S = -\sum P(H_S)\ln(P(H_S))\ ^3 \tag{8}$$

The problem can then be reduced to a Lagrange optimization task for which the exponential family of distributions is a well known solution (see Eqn. (6)) with the inverse temperature as its Lagrange multiplier [38].

D. Super-Paramagnetic Clustering (SPC)

1. A Maximum Entropy Method

We define a neighbor on a lattice to be a node located in the vicinity of another node such that a node $s_{i,j}$ will have neighbors $s_{i+1,j}, s_{i-1,j}, s_{i,j+1}$, and $s_{i,j-1}$. A neighborhood generated using these rules is valid for a 2D lattice with a fixed J for all nodes (the interaction strength is said to be homogeneous). This is the original method used in simulating Ising/Potts models of ferromagnets. As a generalization to the problem of data clustering, we will consider a neighborhood which emerges from the inhomogeneous interaction strength J_{ij}. Every neighborhood is a mini-graph, and their aggregation constitutes a graph whose topology is determined by the matrix J_{ij}: For two nodes to be neighbors they, each, must be included in their respective K-nearest neighbors.

[3] with $P(H_S) \geqslant 0$, and $P(H_S)\ln(P(H_S)) = 0$ for $P(H_S) = 0$.

We first define the neighborhood of size K. This is implemented following steps in Table (C) below.

C1 Pick a K appropriate for the case, and build the `nodenext` matrix which is the array containing the locations of every node's neighbors. Throughout this dissertation we use $K = 10$ except when otherwise explicitly stated.

C2 Build the MST, and add its edges to `nodenext` thus making every graph connected regardless of K

TABLE C: Setting the neighborhood size K for SPC: The neighbor determines the scope of the algorithm. It effectively cancels the pairwise interaction strengths of the spins outside the spins respective neighbors thus producing a speed-up in the computation of the Hamiltonian.

The graph is traveled via nodes, and edges can be set active or inactive with probability

$$p_{ij} = 1 - e^{-\frac{J_{ij}}{T}\delta_{s_i,s_j}} \tag{9}$$

The next array is called `links`. It is the adjacency matrix where the activation status of edges is stored such that $\texttt{links}_{ij} = 1$ if $p_{ij} >$ rand, and 0 otherwise.

The original Hoshen-Kopelman (HK) algorithm [36] is the standard for labeling clusters in many statistical mechanics problems. The 2D version is mostly restricted to two neighbors per node: $s_{i-1,j}$, and $s_{i,j-1}$. SPC (Sec. G 1) deals with problems where J_{ij} is not fixed, and K can be large so we apply the extension of HK to non-lattice environments found in [3].

The clusters are labeled using the extended HK algorithm (See Table D below).

D1 Initialize the label counter at 0, create two arrays: `nodel`, a 1xN array which stores the labels, and `nodelp` an empty list where the roots are recorded. HK is inspired by the Union-Find algorithm which, like its name says, "finds" the root class of nodes, and "unites" nodes belonging to the same class. The concept is applied similarly in SPC, nodes' labels are stored in `nodel`, and once clustering is done, `nodelp` is used to re-assign every nodes to its root class.

D2 Check for activated edges, if none or if any are occupied but none of those link to already labeled nodes: start a new cluster by storing the current label counter in `nodel`, and `nodelp`, then increase the label counter.

On the other hand if active edges link to labeled nodes: Find the root of the node, and its labeled neighbors. This is achieved by first locating the labels stored in `nodel`, then using `nodelp` to recover the root associated with these labels. We then pick the smallest root, store it as the label of the present node in `nodel`, and we replace the other roots in `nodelp`.

D3 Sequentialize `nodelp`

D4 Relabel `nodel` with the true roots using `nodelp`

TABLE D: Labeling: Implementing the extended Hoshen-Kopelman (HK) algorithm: HK reads data from a matrix of edges indicating spins pairwise associations and proceeds by creating the disconnected components (clusters) (also see Appendix 2)

Once the graph is fully constructed, the next step is implement the Swendsen-Wang MCMC algorithm (See C1 in E below.

E1 The resulting clusters are all flipped independently from each other: they each get assigned a new spin value between 0 and q.

TABLE E: Flipping Clusters using Swendsen-Wang MCMC

Finally, the spin-spin correlation G_{ij} is the average probability of two spins being in the same cluster is computed in two steps (See Table (F)).

F1 At every MCMC step c_{ij}, the two-point connectedness, is incremented if i and j are clustered together.

F2 Once the simulation ends for the temperature explored we compute

$$G_{ij} = \frac{(q-1)c_{ij}+1}{q}$$

TABLE F: Compute the Spin-Spin Correlation G_{ij}

The spin-spin correlation G is probably the most important quantity as it is used to build the final clusters. The threshold θ for which two nodes are members of the same cluster is picked to'be higher than $\frac{1}{q}$ but less than $1-\frac{2}{q}$. The bounds on that range are explained by the distribution of G_{ij} which peaks at those two values: They are respectively the peak inter and intra cluster correlations [4]. It is typical to use $\theta = 0.5$ as it does not significantly affect the results in previous examples [7].

2. A Maximum-Likelihood Method

Based on an analysis of the spectral properties of stock market correlations matrices, Noh [56] makes the following statistical *ansatz*: let's assume an existing market hierarchy where individual stocks dynamics are dependent on clusters of similar stocks. This can be illustrated by a simple model as follows:

$$x_i = f_i + \epsilon_i \tag{10}$$

where x_i are the stock's features, f_i the cluster-related influence, and ϵ_i the node's specific effect.

In [24] Giada, and Marsili formally develop a Potts model using Noh's idea, and in [32] Hendricks, Gebbie, and Wilcox solved the optimization problem using a PGA for unsupervised learning for quantitative trading. Let's consider a group of N observations, embedded in a space with dimensionality D as the features, and as with SPC (Sec. G 1), every observation is assigned a spin value. One version of the ansatz models the observation features such that

$$x_i = g_{s_i}\eta_{s_i} + \sqrt{1-g_{s_i}^2}\epsilon_i \tag{11}$$

[4] For $q = 20$ results in $\rho_{sa} = 0.05$, and $\rho_{sb} = 0.9$. Uncorrelated nodes have $\rho_{ij} \approx \rho_{sa}$, and correlated nodes $\rho_{ij} \approx \rho_{sb}$.

where x_i is one feature, g_{s_i} the intra-cluster coupling parameter [5] , η_{s_i} the cluster-related influence, and ϵ_i the observation's specific effect, and measurement error. A covariance analysis yields additional terms such as n_s the size of cluster s, and c_s the intra-cluster correlation [6].

We explicitly mention that $n_s < c_s < n_s^2$ must be enforced: the lower bound is required because g_s is undefined for values of $c_s \leq n_s$, and the upper bound requires a strict inequality because Eqn. (13) is undefined when $c_s = n_s^2$. We introduce a Dirac-delta function [7] to model the probability of observing data in a configuration S close to criticality:

$$P = \prod_{d=1}^{D} \prod_{i=1}^{N} \left\langle \delta(x_i - (g_{s_i}\eta_{s_i} + \sqrt{1 - g_{s_i}^2}\epsilon_i)) \right\rangle . \tag{12}$$

This joint likelihood is the probability of a cluster configuration matching the observed data for every observations, and for every feature. The log-likelihood derived from P can be thought of the Hamiltonian of this Potts system:

$$L_c = \frac{1}{2} \sum_{s:n_s>1} \ln \frac{n_s}{c_s} + (n_s - 1) \ln \frac{n_s^2 - n_s}{n_s^2 - c_s} . \tag{13}$$

The sum is computed for every feature, and represents the amount of structure present in the data. The value of L_c is indirectly dependent on spins via the terms n_s , and c_s.

A-priori advantages of this method over industry standard alternatives: First, that L_c is completely dependent on C_{ij}, and the dimensionality of the dataset only plays a part in computing C_{ij}, and Second, it is unsupervised: There are no preset number of clusters. Clustering configurations are randomly generated, and that which maximizes L_c provides us with the number of clusters, and their compositions.

Further modification of the model can be made to reduce the Hamiltonian to that of the standard K-means algorithm [42]:

$$H_{KM} = \sum_{s:n_s>0} (n_s - \frac{n_s}{c_s}) \tag{14}$$

The f-SPC algorithm (Sec. 3) uses a PGA to find the global optimum of the likelihood L_c (13).

The principles of our GA are given in Table (G) below.

---

[5] The thermal average $\langle g_s \rangle$ can be used to reconstruct data-sets sharing identical statistical features of the original time-series by using Eqn. (11) [24]

[6] Here $n_s = \sum_{i=1}^{N} \delta_{s_i,s}$, $c_s = \sum_{i=1}^{N} \sum_{j=1}^{N} C_{ij}\delta_{s_i,s}\delta_{s_j,s}$, and $g_s = \sqrt{\frac{c_s - n_s}{n_s^2 - n_s}}$ [24, 32].

[7] Let $y_i = x_i - \left(g_{s_i}\eta_{s_i} + \sqrt{1 - g_{s_i}^2}\epsilon_i\right)$, and $\delta(y)$ a Dirac delta function of y which is 1 when $y = 0$, and 0 otherwise.

G1 **Generate Populations**: Generate the populations as a set of randomly generated Potts configuration with spin values ranging form 0 to N, 2.)

G2 **Evaluate Fitness**: Use the computation of L_C

G3 **Select the Best Individuals**

G4 **Mutate**: A set number of individuals in the populations are mutated

G5 **Recombine**: The parent and child generations are recombined and again selection of the best individuals takes place.

G6 **Iterative Convergence**: Repeat 2.) to 5.) until sufficient convergence has been achieved.

TABLE G: f-SPC PGA Implementation: This Genetic Algorithm has no crossing step where parents would be mated. The mutations are the main genetic diversity operator. Mutations and Likelihood computations are evaluated in parallel.

The original PGA algorithm implemented in [22] contained a mating step involving a bespoke cross-over function, and a restriction: Only parents with the same number of clusters could be mated, and the resulting children should maintain the same characteristic. The enforcement of this rule restricted clusters merging and splitting through mutations only. Our f-SPC implementation removes this intermediary step. This was implemented in order to decrease the computational cost [8]

In addition to the diversity of individuals present in the population, mutations serve as GA diversity operators: They increase the genetic diversity, and send the system onto another path toward higher local maxima. We used six equally weighted types of mutations: i) *New*: A complete new individual, ii) *Split*: a random cluster split into two, iii) *Merge*: two clusters merged at random, iv) *Swap*: two spin labels are exchanged, v) *Scramble*: where a sequence of spins have their labels re-assigned in reverse order, and, vi) *Flip*: cluster (spin) labels are randomly re-assigned using the total cluster number (See SW Table E).

At last, the simulation has converged once the fitness of the best individual hasn't increased for a pre-determined number of iterations called "Stall Generations". It should also be noted that although we did not recode a CUDA implementation for direct GPU implementation of our refined PGA algorithm this can be implemented using a modified version of the original CUDA implementation [32].

[8] The algorithm is able to double the number of generations from 250 to 500 for a 5 mins simulation (Iris see Sec. IV C): $L_c = 100,$ and without the cross-over function $L_c = 105$.

E. Enhancing Performance

We review the literature on implementations of maximum likelihood methods based on Giada and Marsili's Likelihood optimization.

It is important to make clear that the Potts Hamiltonian H_S in its ground state (the pure ferromagnetic Potts Model: $\forall J_{ij} > 0$) meaning its lowest energy, has all spins found in one giant cluster, and happens at $T = 0$. One should however be careful as not to confuse the state at $T = 0$, and the ground state which aren't necessarily equivalent but are the same for the Potts Hamiltonian. In [40], the Anti-ferromagnetic Potts Model ($J_{ij} < 0$ is allowed) is considered, and is found to only be relevant at the ground state $T = 0$: the lowest energy displays, in this case, two clusters instead of one: The energy/mining sector in 1 cluster and the rest of the market in a second one.

The Potts Hamiltonian has a trivial behavior: Approximately, a ground state at $T = 0$, excited states as the temperature is increased, and the energy goes up until it caps at its maximum (i.e. N spins for N clusters). Because of this nature it does not lend itself to traditional maximization/minimization methods as those would inevitably result in the trivial ferromagnetic and paramagnetic solutions. It is why SA can be considered the preferred methods for such systems. It then follows that the Potts Hamiltonian isn't a classical optimization cost function for which one's sole interest would be minimum/maximum values within standard machine learning frameworks. L_c on the other hand meets these requirements: Marsili and Giada explain in [25] that L_c shows a "non-trivial" clustering structure: At low β (i.e. $\beta = \frac{1}{T}$), the system is at high T and in a state equivalent to the Paramagnetic phase in which L_c is always zero. As beta increases, and a phase transition has happened the clusters configuration at very high beta (or $T = 0$) the systems still shows a non-trivial cluster configuration unlike in the original Potts Hamiltonian case. This motivates further the use of L_c as a clustering objective function as it can be maximized by a variety of optimization routines. Finally Marsili and Giada show that L_c is a "well behaved" function with solutions consistent given their likelihood values: Configurations with close likelihood values will show high clustering similarity which isn't guaranteed by alternative data clustering methods, and their associated objective functions. It is then legitimately motivated to build and/or use less expensive methods around L_c whose solutions will be consistent with the ones obtained using expensive algorithms.

We review in more details the previous attempts at building such heuristics in the literature. In [24, 25, 45] Marsili and Giada mention 3 implementations of optimizations methods they use with L_c:

- A Metropolis based SA. They note that the original Swendsen-Wang based SA does not work with L_c: modifications to the clusters are accepted with probably depending on the temperature. Thus restricting the trajectories allowed, and allowing the exploration of different regions of the space.
- A Merging Algorithm (MR): All spins start in their own clusters, and are successively merged in a greedy fashion: Every single merging combination is

considered and at each iteration the next candidate chosen is the one which maximizes the increase in new formed cluster's likelihood.

- Deterministic Maximization (DM): Clusters can be split as well as merged for maximum likelihood increased which greatly expends the search compared to MR.

Building on the above work, a metropolis SA and MR were implemented in [50] where sectors and states of the South African equity market where identified. This was followed by a first attempt at the construction of a GA in [22] as a fully unsupervised alternative to SA. The GA created for L_c optimization involved a "bitflip" (see Sec. II D 2 for definition) mutation function, and a cross-over function which fixed the number of clusters between parents so as to create children with the same number of feasible clusters. This was a serial GA and given the noncompetitive execution times a PGA was built and used in [12, 32, 33] as an interpretation of the Deterministic Maximization. Given the lack of public implementation of MR, and DM one can only guess the exact method the authors had in mind. In particular what made this GA different is the cross-over specifically designed for L_c. Instead of using the typical single-point, or two-point cross over functions. Cieslakiewicz argues that the performance of these algorithms is greatly improved if their components incorporate domain knowledge from the problem they are tasked to solve which, in this case, is L_c's maximization. He then argues for a hybrid cross-over function which is made of a combination of the single point cross-over for which the segments are picked after two stages:

- The single point locations are randomly picked, and the parents left and right segments of the split location are evaluated using (18)
- For both left and right segments the child receives the parents segment with highest likelihood if cross-over probability is higher, and the lesser segment if not.

This produces children which are validated at the cluster level, and re-validated at the configuration level once mutations have been applied and their new likelihood computed. This avoids introducing bad candidates whose likelihood would be low in comparison to that of the group at its current iteration.

In this work we have replicated a CPU-based version of the previously mentioned PGAs, and we now want to move toward an algorithm in the same vein as MR, and the Louvain community detection algorithm [9] (See CBMA Sec. II F) which despite its serial nature provides low computing resources consumption and drastically low execution times. We tested the two PGA implementations and CBMA against three data sets when possible:

- BRICS Data V B
- NYSE Data V A

- US intra-day 5mins Bar Data V C

We ran those algorithms on the following machine:

- Model: Dell XPS 15
- CPU: Intel Core i7-6700 HQ @ 2.60GHz (we use 7 core out of 8)
- RAM: 16GB
- Python ver.: 3.7
- Operating System: Windows 10

f-SPC Algorithms	BRICS	Kaggle S&P 500 1day Bar	Kaggle US Stocks 5mins Bar
Nodes / Edges	226/51k	447/200k	1.2k/1.5M
CBMA	7.6/2s	377/7.7s	1815/94s
PGA w/ Cross-Over	5.61/1051s	193.63/6.08 Hrs	N/A
PGA w/o Cross-Over	6.33/1207s	218.86/9.78 Hrs	N/A

TABLE H: Table of algorithm execution cost comparison: The CBMA has the best performance whereas the PGA comes last. Important to note that the PGA in [32] was GPU-based written in C++ and had a much better performance. Nevertheless CBMA is able to reach faster execution times than Hendricks et al. PGA making it the fastest, and least resource intensive L_c optimizer we are aware of.

The amount of data available to scientists to test models and numerical simulations hasn't ceased to increased. We are now living in the "big data" era and the need to algorithm to efficiently handle large data sets has become ubiquitous. The data sets in [9] are networks which range from tens to millions of nodes with edges numbers from tens to billions. One example of such network is the website of the University of Notre Dame nd.edu (325k nodes/1M edges). This network has a similar number of edges to our US intra-day 5mins Bar Data (1.2k nodes/1.5M edges) but a much larger number of nodes (which is explained by the power-law degree distribution of real world networks). The Louvain algorithm takes 3s (on a bi-Opteron 2.2k with 24GB of RAM) to optimize its modularity, and while not totally comparable, our algorithm took 171s ($\approx$ 3mins) thus making it very much usable for live trading however showing improvement are possible.

It's important to note some of the nuances between correlation-based clustering and community detection: whereas as we previously mention real network are power-law degree distributed, one can argue that correlation matrices are fully connected

networks with N nodes for N^2 edges which mean the performance of network clustering algorithms and correlation-based algorithms with scale differently with size.

In Table (H) we show the execution times and the likelihood achieved by our implementation of the PGA, and CBMA but one must acknowledge the difficulty in making comparisons to other implementations in the literature. As far as we are aware no replications and likelihood comparison have been done using different implementations on identical data sets. This is critical because the algorithms in [22, 32, 33] attempt to implement DM but do not compare their performance (be it computational time or maximum likelihood attained) to Marsili and Giada's DM. Furthermore in [25] have established a hierarchy of algorithms:

$$MR \ll DM \ll SA \tag{15}$$

They argue that maximum likelihoods attained with SA are often superior to DM and MR solutions. In our project we have shown in Table (H) that our CBMA (MR) likelihoods are superior to our PGA solutions which would violate this order. This may be due to our own inability to properly implement a DM algorithm however as previously argued there are no benchmark data sets available for replication studies. This motivates us to argue that unless one is capable of producing DM implementations which outperform algorithms such as CBMA their additional computational cost is not fully justified.

F. Community Base Merging Algorithm (CBMA)

One can approach the clustering problem with algorithms implementing top-down or bottom-up methods. Top-down methods are divisive and consist in starting with a single cluster as initial condition and splitting (or partitioning) the graph in additional clusters iteratively while minimizing the cost. On the other hand, bottom-up methods initially start with each observations in their own clusters, and proceed to merge them iteratively. The so called "Louvain" algorithm [9] is agglomerative and implements the later bottom-up approach to "community detection" in graphs. It is in spirit very similar to the Merging Algorithm (MR) developed by Marsili and Giada in [25].

The method proposed in Sec. II D 2 allows for all sorts of mutations and is insensitive to initial conditions. At every step a new generation of individuals is mutated, evaluated, and a group of the best candidates survives until the next algorithm's iteration. This methods has its disadvantages which are the following:

I1	It is convergence based: Assuming the existence of multiple local maxima it is designed to try and navigate around these "sub-optimal" solutions on its way to a potential global maximum. However there is no certainty and it is just assumed that the algorithm stops once the convergence criteria is met
I2	Because it applies random mutations at every generations, the population size, the number and diversity of mutations, and the number of generations all have an impact of the final result.
I3	Parallel implementation: This requires loading and evaluating the entirety of the mutated population at every iteration. This has has a computational but also a memory cost as it requires loading the data (i.e. the correlation matrix) on every cpu. The computation of the Likelihood itself is inexpensive but multiprocessing adds cpu-overhead cost. This was avoided by using a GPU-based PGA in [32].

TABLE I: Disadvantages of the PGA algorithm in Sec. II D 2

Here we start with all N spins in N cluster, and we iteratively merge clusters either at random or in a greedy fashion.

MR's implementation requires computing the ΔL_c: Let's consider three clusters $C_1,\ C_2,\ and,\ C_3$ with $C_3 = C_1 + C_2$ where the addition sign means clusters $C_1,\ and,\ C_2$ are merged. Marsili and Giada define two cases for ΔL_c :

$$\Delta L_c = L_c(C_3) - \max(L_c(C_1), L_c(C_2)) \tag{16}$$

$$\Delta L_c = L_c(C_3) - (L_c(C_1) + L_c(C_2)) \tag{17}$$

Whereas in Eqn. 16 C_3 would be a better cluster than any of $C_1,\ and,\ C_2$ we opt to go with the more restrictive definition in Eqn. 17 by requiring that the new cluster be better than the combination of the two individual sub-clusters.

We have slightly modified Eqn. 13 by removing the sum so as to only compute the likelihood of individual clusters

$$L_c = \frac{1}{2}\left[\ln\frac{n_s}{c_s} + (n_s - 1)\ln\frac{n_s^2 - n_s}{n_s^2 - c_s}\right] . \tag{18}$$

and the objective at every iteration is to maximize ΔL_c over every possible moves.

As an implementation we propose a Community Based Merging Algorithm (CBMA) which mimics in a closer way the state of the art community detection algorithms such as Louvain [9].

J1 The Correlation Matrix C_{ij} is mapped to a graph $G(V, E)$ whose nodes V are the stocks, and edges E are weighted by the cross-correlations.

J2 We add a second attribute named n_s (see Sec. II D 2) to the graph which stores a cluster size, and naturally initial n_s values are set to 1.

J3 We create a list `tracker` which stores lists of labels: each list represents a cluster, and the labels inside the list are the nodes inside the given cluster.

TABLE J: Mapping the data set correlation matrix allows to reduce its size at every iteration. This however requires a way to keep track of the clusters compositions which we do by using the tracker array.

The spin array S which contains the nodes labels (or spin values). Having `tracker` and S may sound redundant but they serve different purpose. At every iteration both `tracker` and S are reduced in size however `tracker` keeps track of the clusters composition while S keeps track of the original root label: the first label merged.

K1 The first pass of the algorithm is the most expensive: we create an array `candidates` which stores the root and leaf labels, and the corresponding ΔL_c. The first pass consist in applying a "merge" function which merges one label to a list of labels, and we do this for the N initial labels resulting in about $\frac{N^2}{2}$ operations.

K2 The next stage consist in iterating the following steps: Look at `candidates` and find out the highest ΔL_c, and unless it is bigger than 0 continue to the next step.

K3 From `candidates` get the root and leaf labels, and find their respective indices in S. Find out the smaller index, and use as the root index, and the bigger one as the leaf index. In `tracker`, get the root, and leaf clusters by using the previously found indices. Add the labels in the leaf to the root cluster, and delete the leaf cluster from `tracker`.

K4 The tracker is now updated, and we proceed to deleting the rows in `candidates` where the root and leaf labels are found.

K5 We update the graph by merging the leaf to the root label: this is achieved by updating the edges weights of the root label with the sum of the leaf's and its own edges weights. The self-correlation becomes the intra-cluster correlation, we also update the root's n_s value by adding the leaf's n_s value, and we remove the leaf label from the graph.

K6 The next step consists in generating the new spin array S by extracting the nodes' labels from G.

K7 **Iterative Convergence**: Run the merge function but this time only merging the root label to the remaining labels in S, and finally update `candidates`. It not needed to loop over the entire array like in 1) because the other labels and their associated weights haven't changed. We iterate by repeating this process, and picking the highest ΔL_c until a local maximum is reached.

TABLE K: CBMA's main routine consists in merge clusters for maximum increase in likelihood, and by iteratively reduce the size of the search space by removing merged labels from the data. The newer clusters are then merged against the remaining labels until a local maximum is reached.

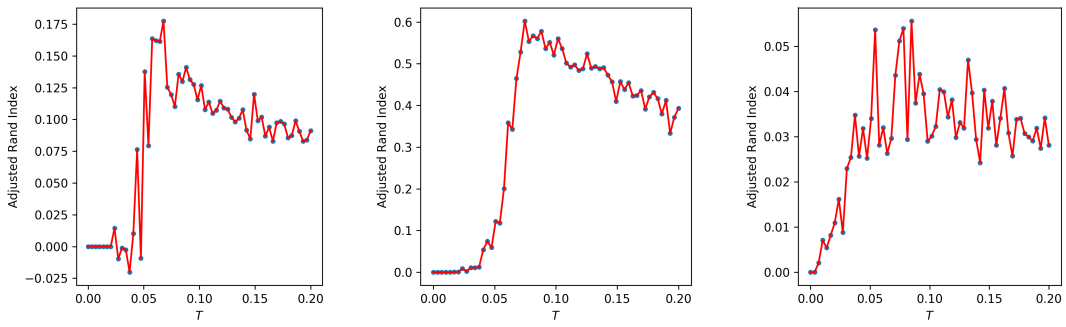
The CBMA operates in a similar fashion to a comprehensive grid search over the space of clusters. It completes in $N-1$ or once a maximum is achieved, and the most expensive step is the first one. It loops over a unique array S of labels, iteratively reduces the size of the array, tests the new root labels on the remaining labels, and keeps track of the labels merged at previous steps. By proceeding this way the

CBMA significantly decreases its cost both in term of time, and CPU resources: it is serialized, and has no obvious parallelization need.

G. Super-Paramagnetic Phase Validation

In [25] it is shown that MR, DM, and SA cluster solutions have significant overlap between them. The goal of this project was to effectively validate the existing link between the solutions obtained using L_c and the original Potts Hamiltonian H_S. We proceeded by comparing clustering configurations obtained using the two models for the data set in Sec. V A.

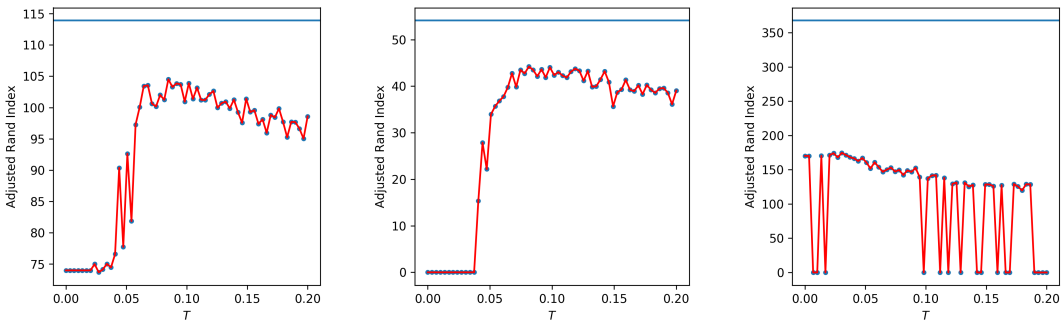
Shown in Fig. (1a) , Fig. (1b) , and Fig. (1c) are the Adjusted Rand Indices as functions of temperature for our three cases. We compute the ARI at every temperature taking f-SPC as the true classification, and SPC as the candidates. Maximal ARI values are respectively 0.175, 0.6, and 0.05 for temperatures $T = 0.068$, $T = 0.071$, and $T = 0.08$. These temperature values are all located in the SP-phase where the susceptibility is non-zero confirming the claim in [24] that the maximization of L_c should recover a clustering configuration of the a system in the vicinity of the phase transition. We also note that despite SPC neighborhood search restrictions the Normalized f-SPC solution has the highest similarity to its SPC's counterparts. The "Market Mode" case of f-SPC has a large mixed cluster, and the RMT one has a its largest cluster mixed with securities from every economic sector. This would require further analysis but we think the main difference between that and SPC's equivalent is this cluster which could be of low correlation.



(a) Market Mode (b) IMN (c) RMT

FIG. 1: In figures (a), (b) and (c) find the ARI (See Sec. IV) for the following cases : (a), with a market mode (See Sec. III C), (b) de-noising with IMN (see Sec. III C 2) and (c) when a RMT method is used to clean the correlation matrix (See Sec. III C 1). The ARI index expresses configuration similarity on [0,1] [37]. Blue dots represent ARI values, and the red line the curve liking them all. We looked at NYSE S&P500 447 Stocks Data. In all 3 cases we compare the f-SPC method (See Sec. II D 2) to each of SPC candidates (See Sec. V A). This demonstrated that in all 3 cases the maximum likelihood candidates are close to solutions recovered within the super-paramagnetic phase.

We push further the analysis by considering L_c as a clustering quality evaluator. As was demonstrated in [25] L_c is a consistent objective function which if maximized can discriminate between clustering algorithms. Similarly to the "Silhouette Coefficient", and the "Calinski-Harabaz Index" which are methods used to evaluate the clusters definition when the ground truth class is not known, L_c plays a similar role.



(a) Market Mode (b) IMN (c) RMT

FIG. 2: S& P500 (Sec. V A) N = 447 Stocks, D = 1249 trading days: We computed the Likelihood [24] L_c (13) of every SPC solutions for all temperatures (red curve, and blue dots), and Likelihood L_c of f-SPC's solution (blue horizontal line). in a) the Market Mode case, in b) the Normalized case, and in c) the RMT case. Every f-SPC solutions has higher likelihood than the SPC entire temperature range in every case. f-SPC solutions are composed of clusters with higher correlation than SPC candidates.

We test for this by evaluating all SPC candidates for their L_c values, and we add a horizontal line on each plot indicating the respective f-SPC's L_c. In every case, SPC's L_c start low at low T, reaches a maximum at intermediate T and decreases slowly at T increases into Paramagnetic territory. This is yet another confirmation that higher L_c values are located in a intermediate temperature regime which coincides with the SP-phase when the system is critical. SPC's L_c maxima are 105, 43, and 170 respectively in Fig. (2a) Fig. (2b) Fig. (2c), and we observe that solutions recovered using f-SPC all have higher likelihoods than SPC's. Based on the result in this dissertation, and in [25] one could argue that f-SPC produces better clustering candidates than SPC at least in this case.

1. *Free-Energy Validation*

We now define the Helmholtz Free Energy F for a thermodynamic process of an isolated system.

$$F = U - TS \tag{19}$$

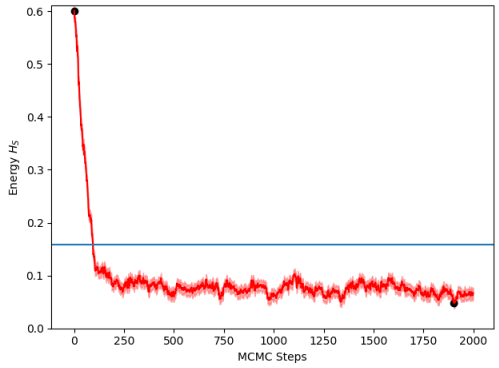
where U is the internal energy of the system (see Eqn. (3), and (13)), T is the temperature of the heat bath or reservoir in contact the system, and S is the entropy.

$$F = -T \ln Z \tag{20}$$

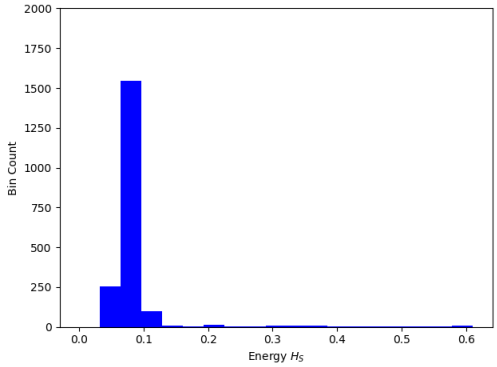
The free energy can also be computed using (20) with Z as the partition function like in Eqn. (6).

We consider an isothermal process a system exchanging energy with a reservoir at constant T by absorbing heat until its own temperature converges to that of the reservoir. For processes such as the one just described Eqn. (20) tells us that some of the energy needed for the system to be realized can be spontaneously transferred from the reservoir by heating "TS". For systems on which no work is done $\Delta F \leq 0$ and thermal equilibrium is reached if the free energy reaches a minimum.

Using Mean Field Models in [7], and [24] It was shown that the free energy reaches a local minimum within the Super-Paramagnetic or Clustered Ferromagnetic Phase, and a maximum at the Paramagnetic Phase transition. We argue that the temperature at which the previously mentioned minimum happens is synonymous to the heat-bath inside of which the system is in its "lowest level".



(a) Evolution of the internal energy at $T = 0.13$ as MCMC steps are made. There is no convergence per se to a unique value, but distributional oscillations around a level.



(b) Histogram of the distribution of the internal energy at $T = 0.13$ with the number of bins calculated using (21)

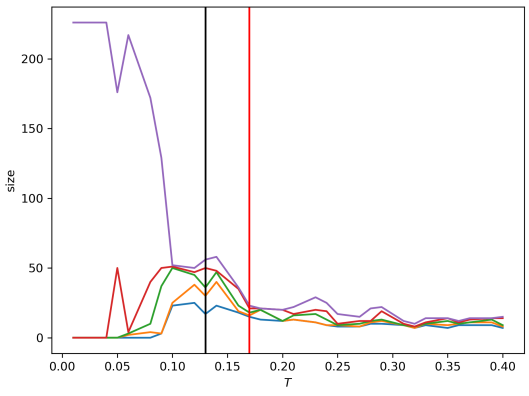
FIG. 3: BRICS Data: SPC's internal energy at $T = 0.13$. On the left, the energy as it converges, and on the right its binned distribution.

Eqn. (20) requires computing Z whereas Eqn. (19) needs the average energy $\langle H \rangle$, and the entropy S. Although we don't show it these two methods agree. At any given temperature the system doesn't converge to a specific energy level but displays a distribution (see Fig. (3a)). The task at hand is now about picking a number of bins for our MCMC simulation which is consistent and not arbitrary. We borrow a "low bias" methods from [30] which follows:

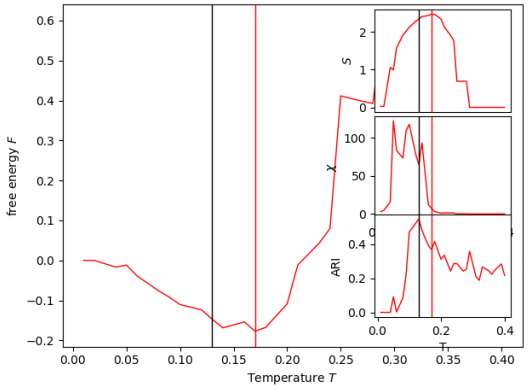
$$k_X = \text{round}\left\{\frac{\epsilon}{6} + \frac{2}{3\epsilon} + \frac{1}{3}\right\} \tag{21}$$

where k_X is the number of bins, and $\epsilon = \sqrt[3]{8 + 324n + 12\sqrt{36n + 729n^2}}$ with n as the number of samples, here the total number of MCMC steps. Because we fix n

to 2000 the number of bins remains set for every temperatures, and the bin edges are set on the minimum and maximum possible energies depending on the problem. The Hamiltonian H_S minimum energy is always 0 (for the ferromagnetic case), and in the case of the BRICS data the maximum was ≈ 0.61 (see Fig. (3b)). We then determine the distribution energy levels by picking the k_X bin centers which we compute by taking the mean of the distribution inside each bin. Once obtained we can now compute Z, the Boltzmann distribution of energies, which we use to compute the thermal average Energy $\langle H \rangle$, the entropy S using Eqn. (8) and the free energy F.



(a) BRICS DATA, Cluster sizes as function of Temperature for SPC solutions. The vertical lines respectively show the temperatures of maximum ARI, and minimum Free Energy / maximum entropy.



(b) BRICS DATA, Free Energy as function of Temperature for SPC solutions (in red). Insets: (top right) Entropy, (center right) Susceptibility, (bottom right) ARI. The vertical lines respectively show the temperatures of maximum ARI, and minimum Free Energy / maximum entropy.

In Figures (4a), and (4b) we show the results of a SPC simulation on the BRICS data (See Sec. V B): We show the free energy as a function of temperature for the SPC simulation, and we also plot the Adjusted Rand Index of the f-SPC solutions against the SPC ones. We quickly describe the behavior of the free energy which decreases and reaches a minimum at $T \approx 0.17$, and a maximum at $T \approx 0.25$. A quick look at Fig. (4a) shows that before $T \approx 0.17$ the giant cluster is breaking down inside the Super-paramagnetic Phase until their sizes are comparable and seemingly stable. After $T \approx 0.17$ The clusters sizes are unstable, start decreasing and it's become impossible to significantly distinguish clusters which signals a transition into the Paramagnetic Phase. More importantly the ARI curve peaks at $T \approx 0.13$, close to the minimum free energy temperature within the Super-paramagnetic Phase thus revealing that f-SPC's algorithm and objective function minimizes the free energy (maximizes entropy) of the system as it maximizes L_c.

2. Sci-Kit learn: Varying Density Clusters

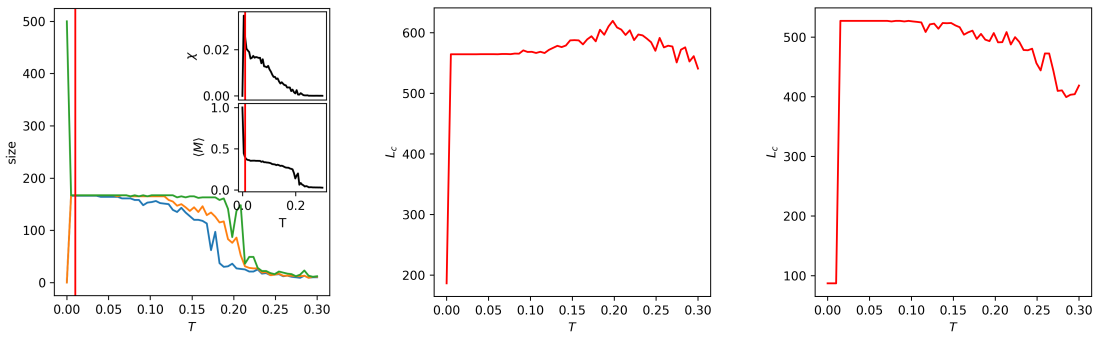
The problem consists of 3 clusters using `Sci-kit learn` samples generator [9] with $N = 500$, and $\sigma = 0.25, 0.5, 1$. We observed two cases of this problem: a 3D case, and a 500D case.

Figure Fig. (5a) respectively show the susceptibility, and the clusters size as a functions of temperature. At $T = 0.01$, in the SP-phase, we observe 3 clusters in Fig. (5a) of size 167, 166, and 166 with an ARI of 1.

We follow this with L_c's solution which scores 1460.47, an ARI of 0.20, while the expected likelihood was 599.91. Yet again our solution's likelihood is higher than our expectation, and the real clusters are divided in smaller ones. In comparison K-Means and DBSCAN respectively achieve ARI of 1, and 0.8. In light of this, we decided to try again the same problem but with the dimensionality set to $D = 500$. As expected SPC's results do not change. However in this instance f-SPC's solution quickly converges to the best classification. A further investigation was done by simulating both the 3D, and 500D cases using SPC, and computing the likelihood L_c for every configurations. The assumption being that the maximum likelihood should be found within the temperature range where the system is in the SP-phase.

---

[9] B. Thirion, G. Varoquaux, A. Gramfort, V. Michel, O. Grisel, G. Louppe, J. Nothman, 'make_blobs', 2017. [Online]. Available: `http://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_blobs.html`. [Accessed: 12-Jun-2018]



(a) Size vs T (b) 3D Case: L_c vs T (c) 500D Case: L_c vs T

FIG. 5: in a) 3D blobs (Sec. II G 2): Cluster sizes vs Temperature T using SPC (Sec. II D 1). 1 giant cluster at $T = 0$, and 3 stable clusters from $T = 0.007$ to $T \approx 0.05$. The transition into the Paramagnetic Phase is asynchronous for each clusters because of their differing densities: At $T \approx 0.05$ cluster 1 starts dissolving, followed by cluster 2 at $T \approx 0.11$, and finally, cluster 3 at $T \approx 0.19$. Insets: (in a) above right) Susceptibility $\chi(T)$, and (in a) below right) Average Magnetization $\langle M \rangle$ at $T = 0.01$. χ peaks at $T = 0.007$ into the SP-phase, remains stable until $T \approx 0.09$, and then slowly decreases to 0 into the Paramagnetic Phase. in b) the likelihood $L_c(T)$ of SPC solutions for 3D clusters, and 500D in c). When D is low, L_c is stable until $T \approx 0.05$ which is the temperature at which the transition into the Paramagnetic Phase occurs. Where we would expect a decrease in L_c, we see an increase as T goes up, a maximum is reached around $T \approx 0.20$ which as can be seen from a) $\chi \approx 0$ which signals the Paramagnetic Phase of the system. The 500D case in c), on the other hand, peaks within the SP-phase until $T \approx 0.10$ which is the temperature at which the transition into the Paramagnetic Phase occurs. Once $T > 0.10$, a net decrease in L_c happens, and as T goes up the slope of L_c remains negative as expected.

Figures (5b) and (5c) respectively show the likelihood as functions of temperature. We notice that in the 500D case in Fig. (5c), the maximum likelihood is found at temperatures $T < 0.15$ within the SP-phase, and the L_c monotonously decreases at higher temperatures. The opposite happens in Fig. (5b) where the best classification doesn't correspond to the maximum likelihood of L_c which in this case is found at high temperatures $T \approx 0.25$ which by looking at χ in Fig. (5a) means we are effectively in the paramagnetic phase. We provide additional comments in the discussion section of this dissertation.

III. Data Pre-Processing

A. The Distance Function

The wide variety of problems our clustering methods can tackle necessitates a careful choice of pairwise distances if we are to properly identify shared behavior.

We will proceed by using the Euclidean distances whenever we assume independence of the features, and the Pearson correlations otherwise especially for problems where the features consist of time-series.

This has implications for both algorithms such that we use the Euclidean distance or the Pearson correlation distance for SPC, and for f-SPC, we use the Pearson correlation matrix, and the similarity matrix, which is the Euclidean distance matrix on $[0, 1]$, and subtracted from 1.

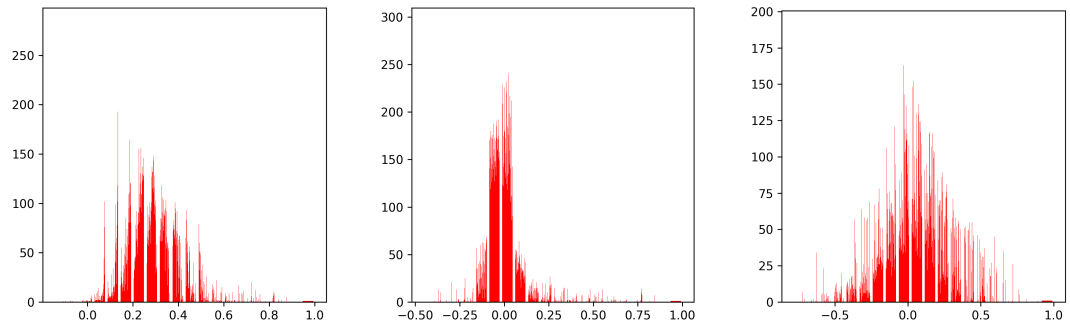
We note that from [40] that our Eqn. (4) can be modified to incorporate negative correlations, but the authors explain this only affects the results at the ground state (*i.e.* $T \approx 0$).

B. Scaling

Raw data sets often contain features on different order of magnitude of scales, outliers, and missing data which can have significant impact on Machine Learning algorithms. One way to deal with these issues is through normalization of the features. This was achieved using the Min-Max Scaling technique which puts all features on a 0 to 1 scale by performing the following operation:

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}} . \tag{22}$$

Scaling has significant effects on the feature space: one example is seen in Fig. (10a), and Fig. (10b) which respectively show the unscaled and scaled plots of the 3 wines problem. The unscaled data set has two classes completely inseparable whereas scaling the data effectively dissociates all three classes with minimal overlap.



(a) Market Mode (b) IMN (c) RMT

FIG. 6: Distribution of Pearson correlations of daily returns for 447 publicly traded companies on the S&P 500 stock exchange from 8/13/2012 to 8/11/2017 (Sec. V A). The "Market Mode" (Sec. III C) : Noisy markets like in a) are highly correlated with most $\rho_{ij} > 0$. The noise is cleaned by removing the "Market Mode" either by IMN (Sec. III C 2) in b) or RMT methods (Sec. III C 1) in c), and produces distributions centered around 0.

C. Noise

The next and final pre-processing task consist in removing any noise present in our data. This is especially important for financial market time-series which exhibit extreme randomness and possibly chaotic behavior. Stock market correlation matrices are noisy, and positively skewed Fig. (6a) which translates into what is referred as the "Market Mode". We consider an intermediary step which consist in "removing" the market mode, thus ensuring we are able to recover the underlying correlation structures, if any, present in the system.

1. Random Matrix Theory (RMT)

We follow the predictions of RMT in [70] by assuming that stock market returns are IID random variables with zero mean and unit variance. These assumptions lead us to the conclusion that stock market correlations should all be zeros, and if the assumptions are indeed true, RMT predicts that the eigenvalues of the random matrices are Wishart distributed such that:

$$P(\lambda) = \frac{Q}{2\pi} \frac{\sqrt{(\lambda_{max} - \lambda)(\lambda - \lambda_{min})}}{\lambda} \tag{23}$$

where $Q = \frac{D}{N}$, and $\lambda_{min/max} = 1 + \frac{1}{Q} \pm 2\sqrt{\frac{1}{Q}}$.

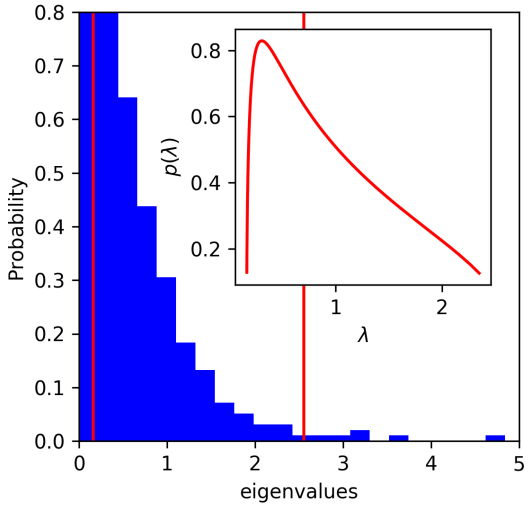


FIG. 7: The Eigenvalue distribution of the Correlation Matrix of 1249 daily returns for 447 publicly traded companies in the S&P500 (Sec. V A).The two vertical red lines delimit the wishart range $\lambda_{min} = 0.16$ and $\lambda_{max} = 2.55$: The eigenvalues located inside the Wishart range (see Sec. III C 1) are noise whereas the ones outside aren't. Inset: (red curve) We show that the computed Wishart PDF of a random matrix (using Eqn. (23)) fits well the eigenvalue distribution of our correlation matrix only inside the wishart range.

Shown in Fig. (7) is the distribution of eigenvalues of the correlation matrix of our stock market data Sec. V A. As can be observed the eigenvalues inside the Wishart range are responsible for the noise whereas those outside of the range are potentially correlated signal which shouldn't be discarded.

We consider the eigenvalues $\lambda > \lambda_{max}$ represent the linear, and 1st order relations between time-series while it is unclear what those on the left side ($\lambda < \lambda_{min}$) of the Wishart distribution are. The linear signals are the signals shared by assets at the sectoral level.

The RMT "Market Mode" removal method is implemented in the five following steps bellow in Table L:

L1	Compute the correlation matrix C_{ij}
L2	Extract the eigenvalues and eigenvectors from C_{ij}
L3	Select the eigenvalues (and the corresponding eigenvectors) found outside the Wishart Range.
L4	Reconstruct the data using the compressed signal: Let X be our data, W the matrix of eigenvectors found outside the Wishart Range, and $Z = W'.X$ the compressed data. The reconstructed data is then $X = W.Z$
L5	Re-compute the correlations C_{ij} from the reconstructed data.

TABLE L: Implementation of RMT Noise removal methods

We tested different time-series lengths ranging from 89 to 1249, and we note that the size of the Wishart Range increases with dimensionality, and the lower left tail decreases on the other hand. The higher the dimensionality the easier it is to rule out eigenvalues as random signals, and the more important the biggest eigenvalues are to the noise-less data reconstruction [10]. An example of a cleaned correlation matrix resulting from this method Fig. (6c).

2. Iterative Matrix Normalization (IMN)

Another "Market Mode" removal method [57] IID random variables with zero mean and unit variance.

The Iterative Matrix Normalization "Market Mode" removal method is implemented in the three following steps bellow in M:

[10] In [24], The authors achieve a similar result by using the model in Sec. (II D 2) confirming that "noise cleaning" mainly affects the small eigenvalues of stock market correlation matrices.

M1	Compute the Covariance Cov
M2	Standardize Cov across rows then columns for a set number of iterations (i.e. 500) or until a convergence criteria is met.
M3	Extract the correlation matrix C_{ij} from Cov

TABLE M: Implementation of Noise Removal via Iterative Matrix Normalization

We observe in Fig. (6b) that the distribution of correlations is now centered around 0.

IV. The Data test-cases

The following examples are used as a stress test for both methods. We obtained both synthetic, and real data which enabled us to discuss the features of each models.

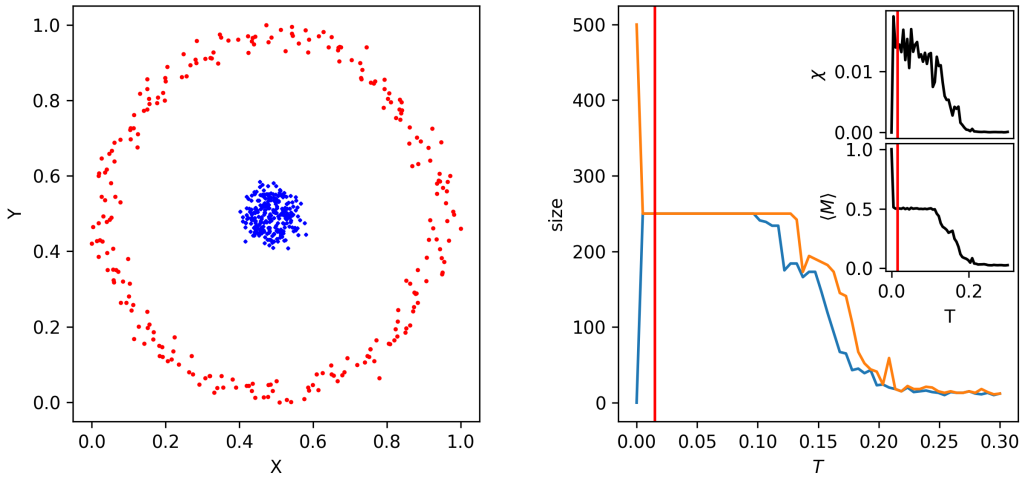
As a comparison tool we use the Adjusted Rand Index (ARI) [37] which given two classifications measures their similarity. The ARI operates on a $[-1, 1]$ scale with positive values signifying increasing similarity. Where a true classification exists we will use the ARI to measure the quality of clustering of both methods but also industry standards such as "K-Means" [42], and "DBSCAN" [15]. Using SPC (Sec. G 1) we cluster a temperature range which we then compare against the L_c (13) solution recovered. We then select the SPC temperature with the highest ARI for a closer comparison with the L_c solution in the stock market case where a true classification is not available.

For visualization, where possible, we provide the plots or we make use of a non-linear dimensionality reduction package called UMAP [51]. The graph of the MST is also provided as it is a faithful representation of clusters on a 2D plane. The MST takes in the graph of our data, and find the unique shortest path linking every nodes.

A. *Sci-Kit learn*: Concentric Circles

Our first problem is the identification of two concentric circles on a 2D plane using `Sci-kit learn` [61] samples generator [11] with $N = 500$, 0.5 for the noise parameter, and the 2 dimensions represent the X and Y coordinates of the observations.

[11] B. Thirion, G. Varoquaux, A. Gramfort, V. Michel, O. Grisel, G. Louppe, J. Nothman, 'make_circles', 2017. [Online]. Available: `http://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_circles.html`. [Accessed: 12-Jun-2018]

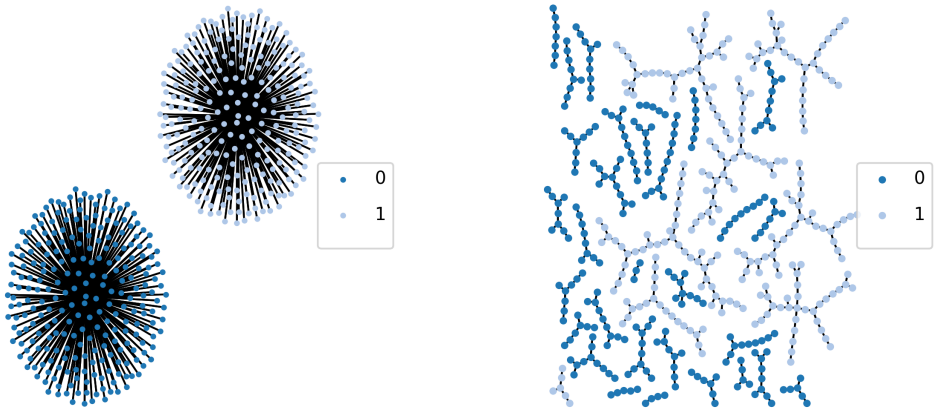


(a) Scatter Plot (b) Cluster size vs Temperature T

FIG. 8: in a) Two circle (Sec. IV A) shaped 2D clusters each of size $N = 250$ such that the blue points have higher density than the red ones. in b) Cluster size vs Temperature T using SPC (Sec. II D 1). As T increases, the giant component successively breaks down: at $T = 0.007$ we can observe 2 clusters which remain stable until $T = 0.10$. Insets: (in a) above right) Susceptibility $\chi(T)$ at $T = 0.01$. χ peaks around $T = 0.007$, remains stable until $T \approx 0.10$ inside the SP-phase, then dives down toward 0 for $T > 0.10$. (in a) below right) The Average Magnetization $\langle M \rangle(T)$ at $T = 0.01$. $\langle M \rangle$ starts at 1 for $T = 0$ then remains stable at $\langle M \rangle = 0.5$ inside the SP-phase from $T = 0.01$ to $T = 0.10$. This stability only occurs when clusters have uniform or identical densities, and are linearly separable. Once $T > 0.10$, $\langle M \rangle$ goes down to 0 inside the Paramagnetic Phase.

Judging by observing Fig. (8b) , and we see no overlap between the two clusters present in the data, and we expect to recover close to perfect clusters after applying the algorithms.

We obtained the susceptibility as a function of temperature in Fig. (8a). Within the SP-phase at $T = 0.01$ we observe two clusters in figure Fig. (9a) both contain 250 nodes with an ARI of 1. Once the temperature gets relatively high, near $T \approx 0.15$, the system is deemed at "high energy", and the clusters dissolve almost simultaneously. Unlike this particular example, this does not generally happen with real data where clusters have different densities.



(a) SPC (b) f-SPC

FIG. 9: in a) 2 Circles (Sec. IV A): The MST of the (SPC Sec. II D 1) Solution at $T = 0.01$ shows two subtrees each representing the two clusters in the data, and in b) with the f-SPC (Sec. II D 2) Solution, a high number of clusters are found: There is no misclassification however the 2 original clusters are pieced apart

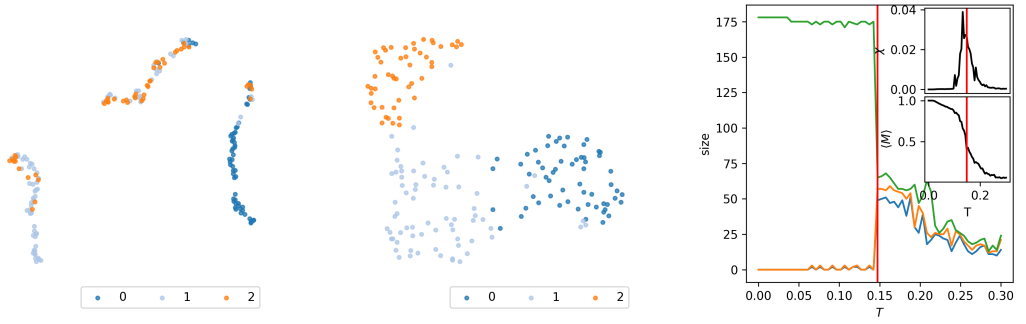
On this data, f-SPC runs for 10000 generations maximizing L_c to 639 while the real classification scores 317. The f-SPC configuration is presented in Fig. (9b) with an ARI of 0.085. In comparison K-Means and DBSCAN respectively achieve 0.16, and 1. K-Means has low clustering quality despite specifying the correct number of clusters. This is due to its inability to deal with non-spherical and non-Gaussian shaped clusters. Despite the high likelihood, Fig. (9b) shows a high number of clusters. The clusters are not mixed and ultimately we fail to recover the initial two clusters.

B. *Sci-Kit learn*: Wine data

The second problem consists of a data set containing three clusters: $N = 178$, and $D = 13$. It is a reputed easy problem illustrating the importance of Normalizing/Standardizing features. There are 59, 71, and 48 samples respectively for class 1, 2 and 3, and the data is generated using `Sci-kit learn` loader [12]. the 13 features are quantities extracted from a chemical analysis of 3 types of italian wines: One Alcohol, Malic acid, Ash, Alcalinity of ash, Magnesium, Total phenols, Flavanoids,

[12] D. Cournapeau, F. Pedregosa, O. Grisel 'load_wine', 2007-2010. [Online]. Available: `http://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_wine.html`. [Accessed: 12-Jun-2018]

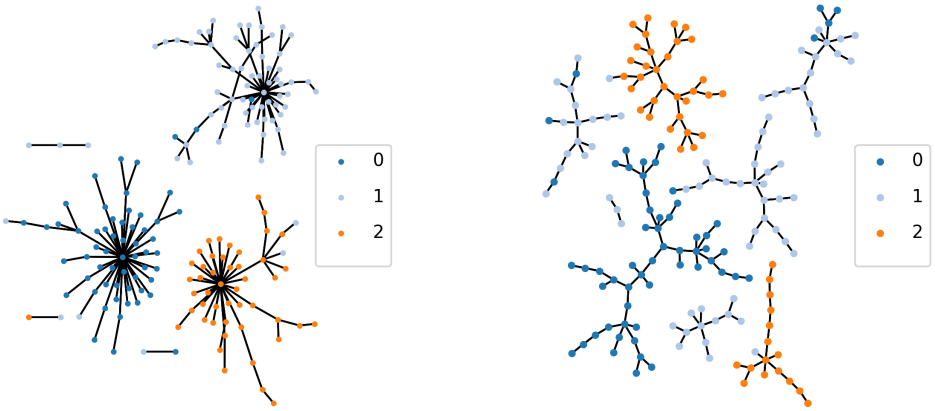
Nonflavanoid phenols, Proanthocyanins, Color intensity, Hue, OD280/OD315 of diluted wines, and Proline.



(a) Raw Data (b) Normalized Data (c) Clusters size vs Temperature T

FIG. 10: in a) 3 wines (Sec. IV B), 2D plot of dimensionality reduction of the 13 features using UMAP [51]. No scaling and/or normalizing done to the features produces 3 clusters: Wines of type 1 and 2 are found in the same clusters while Wines in cluster 0 remain isolated. in b) we rescaled the 11 features using the MinMax method Sec. III B. The MinMax Scaler spreads out the original clusters, and the Wine 1 and 2 clusters are now linearly separable. in c) Clusters size vs Temperature T. From $T = 0$ to $T \approx 0.14$, The ferromagnetic Phase with one giant cluster, then from $T \approx 0.14$ to $T \approx 0.20$, the SP-phase with 3 clusters which all start dissolving once $T > 0.20$. Insets: (in a) above right) Susceptibility $\chi(T)$, and (in a) below right) Average Magnetization $\langle M \rangle$ at $T \approx 0.15$. χ peaks at $T = 0.12$ into the SP-phase, decreases, and peaks one last time at $T \approx 0.17$ to transition into the Paramagnetic Phase.

At first sight in Fig. (10a), 2 clusters are merged whereas once the features have been normalized Fig. (10b) the 3 clusters occupy separate regions of the space. Each cluster has one extremity close to its neighboring cluster which may induce some misclassification error, and because of this we expect to recover 3 imperfect clusters.



(a) SPC (b) f-SPC

FIG. 11: In a) 3 wines (Sec. IV B) MST of SPC's solution at $T = 0.147$: The 3 largest clusters respectively contain most of the observations from the original wine groups except for few unclassified or misclassified samples. in b) f-SPC's solution: There are 7 clusters: 1 for Wine 0, 4 for Wine 1, and 2 for Wine 2.

Between $T = 0.147$ and $T \approx 0.22$ we observe three clusters, and the best classification recovered in Fig. (11a) provides the MST of the SPC's solution with an ARI of 0.82.

Figure Fig. (11b) presents L_c's solution with a likelihood of 83.94, an ARI of 0.51, and an expected likelihood of 66.97. As with the circle problem our solution's L_c is higher than our expectation, and it has 7 clusters instead of 3. 1 cluster contains observations of cluster 0, while clusters 1 and 2 are split in smaller ones without much misclassification. In comparison K-Means and DBSCAN respectively achieve ARI of 0.85, and 0.42.

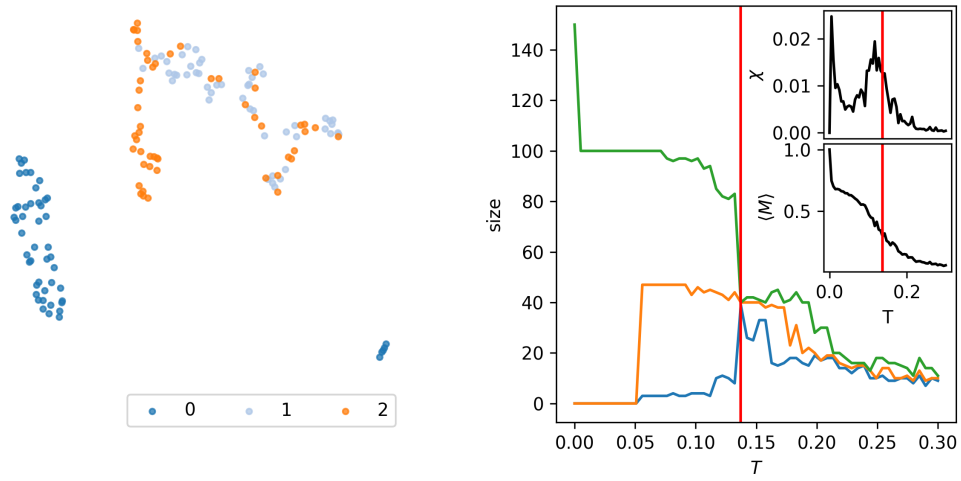
C. *Sci-Kit learn*: Fishers Iris data

Fisher' Iris Data using `Sci-kit learn` loader [13] which includes individuals from 3 species: Iris Setosa, Virginica, and Versicolor. $N = 150$, $D = 4$, and there are 50 nodes per cluster.

As we can see in Fig. (12b), It's one of the more challenging toy problems because two of the three clusters, Virginica, and Versicolor, are not linearly separable. We

[13] D. Cournapeau, F. Pedregosa, O. Grisel 'load_iris', 2007-2010. [Online]. Available: `http://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_iris.html`. [Accessed: 12-Jun-2018]

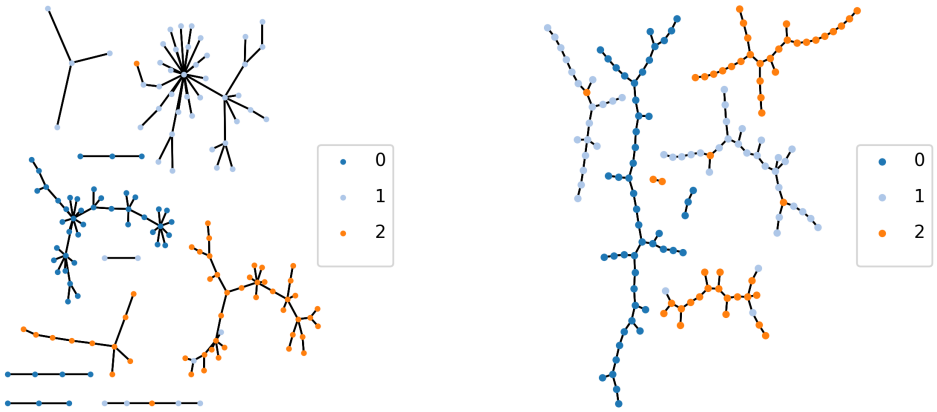
set $K = 7$, and observe two phases in Fig. (12a): for $0.05 < T < 0.137$ there are 2 clusters. The largest contains the Virginica, and Versicolor nodes while the smaller one includes most Seratosa nodes. The 2^{nd} phase transition occurs at right before $T = 0.137$, and is followed by the separation of most Virginica nodes into their own cluster. This SPC solution Fig. (13a) has an ARI of 0.65.



(a) Scatter Plot (b) Cluster sizes vs Temperature T

FIG. 12: In a) Iris 3 species (Cluster 0 for "Setosa", 1 for "Versicolor", and 2 for "Virginica") (Sec. IV C) : 2D plot of dimensionality reduction of 4 MinMax Scaled features. The Setosa, Versicolor, and Virginica clusters are respectively clusters 0, 1, and 2. Clusters 1 and 2 are not linearly separable whereas Cluster 0 is. in b) Cluster sizes vs Temperature using SPC Sec. II D 1: 1 Cluster starting at $T = 0$, 2 cluster at $T = 0.05$, and 3 clusters at $T \approx 0.14$. Around $T \approx 0.16$, the system transitions into the Paramagnetic Phase, and clusters start dissolving. Insets: (in a) above right) Susceptibility $\chi(T)$, and (in a) below right) Average Magnetization $\langle M \rangle$ at $T \approx 0.14$. χ peaks first at $T = 0.007$, and a second time at $T \approx 0.12$: at each transition one or more clusters detach from the giant component.

The L_c solution in Fig. (13b) has an ARI of 0.627, a likelihood of 132, and our expected L_c is 104. In comparison K-Means and DBSCAN respectively achieve ARI of 0.73, and 0.52. Similarly to the precedent examples, we recover five large clusters: Cluster 1 contains Seratosa individuals, while the Virginica, and Versicolor clusters are split into 4 smaller ones with minimal misclassification.



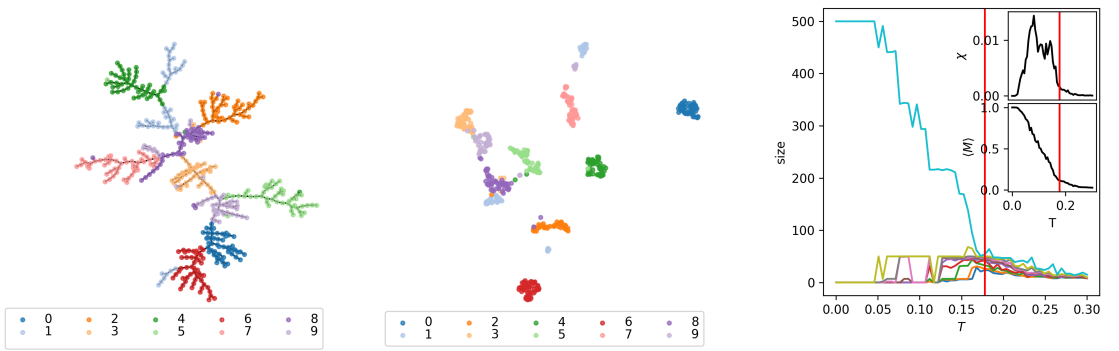
(a) SPC (b) f-SPC

FIG. 13: in a) Iris 3 species (Sec. IV C) : MST of SPC's solution at $T = 0.137$. 3 large clusters representing the original Iris species (Cluster 0 for "Setosa", 1 for "Versicolor", and 2 for "Virginica"), and 6 smaller ones. in b) f-SPC's solution after 10000 generations. 5 large clusters: 1 for Setosa, 2 for Versicolor, and 2 for Virginica.

D. *Sci-Kit learn*: MNIST digits

The hand-written digits dataset, generated with `Sci-kit learn` loader [14], is mainly used to test classification algorithms in supervised learning but we are interested in how well both SPC, and f-SPC deal with the nonlinear nature of handwriting. The data contains 10 classes of digits ranging from 0 to 9. The full set has close to 2000 nodes from which we select 500, and 50 of each class. The images are 8 by 8, and $D = 64$.

[14] D. Cournapeau, F. Pedregosa, O. Grisel 'load_digits', 2007-2010. [Online]. Available: `http://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_digits.html`. [Accessed: 12-Jun-2018]

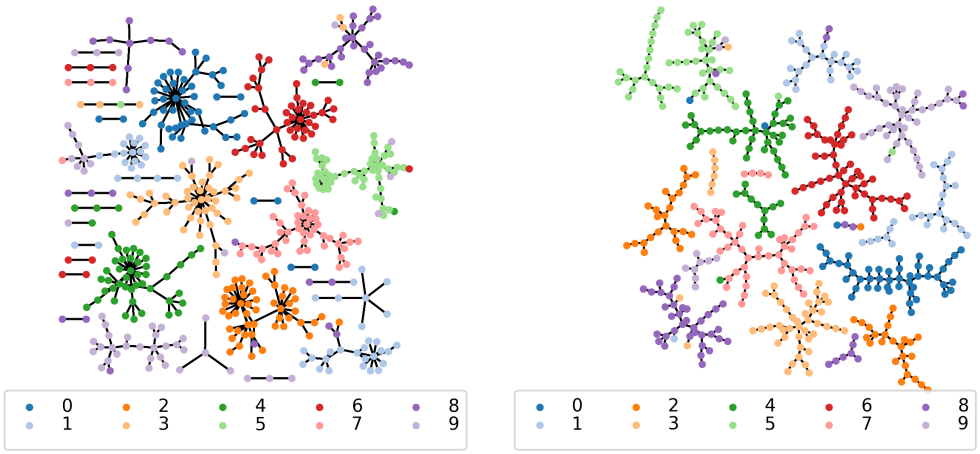


(a) MST (b) Scatter Plot (c) Cluster sizes vs Temperature

FIG. 14: in a) MNIST hand-written digits (Sec. IV D) : 2D plot of dimensionality reduction of the 64 features using UMAP [51]. $N = 500$. 10 classes from 0 to 9. in b) the MST: Overall numbers of the same digit class are close. in c) Cluster sizes vs Temperature T using SPC (Sec. II D 1) at $T \approx 0.18$. Insets: (in a) above right)Susceptibility $\chi(T)$, and (in a) below right) Average Magnetization $\langle M \rangle$ at $T \approx 0.18$.

The MST in Fig. (14a), and the UMAP [51] plot in Fig. (14b) show us all digit classes are linearly separable, the data contains outliers especially "1"'s and "9"'s which may be the result of different writing styles.

We see that the big cluster breaks down in multiple steps Fig. (14c) due to the differing densities of clusters present in the data. Fig. (14c) shows that at $T = 0.178$ right after the final χ peak the configuration's clusters in Fig. (15a) has an ARI $= 0.75$, and show no significant misclassification.



(a) SPC (b) f-SPC

FIG. 15: in a) MNIST hand-written digits (Sec. IV D): MST of SPC's solution at $T \approx 0.18$. 10 classes recovered: Cluster 8 is split in two, and some observations from cluster 9 & 1 are found in one mixed cluster. There are non-linearities in how digits are drawn which may explain the closeness of 9s and 1s. in b) f-SPC's solution: 10 classes recovered: 1 cluster for 0s and 6s, 3 for 1s, 2 for 2s, 3s, 4s, 5s, 7s, 8s and 9s.

L_c's solution in Fig. (15b), after 25k generations, has a likelihood of 149.47, an ARI of 0.747, and an expected L_c of 135. Once again we encounter similar results as with the previous cases with the higher likelihood, and the number of clusters. The L_c solution has close to 20 clusters, and while there is one main cluster per digit which is the case for digits 0 and 6, and mostly for 3, 7, 8, and 9, the digits 1, 2, 4, and 5 are all split in two clusters. We explain this by the inconsistent nature of hand-writing which produces different writing styles. In comparison K-Means and DBSCAN respectively achieve ARI of 0.56, and 0. There are many reasons why DBSCAN fails this problem: DBSCAN classifies some observations as noise into one cluster, it also has issues tackling problems with clusters of different densities.

V. Stock Market Datascience

A. *Kaggle*: NYSE Data

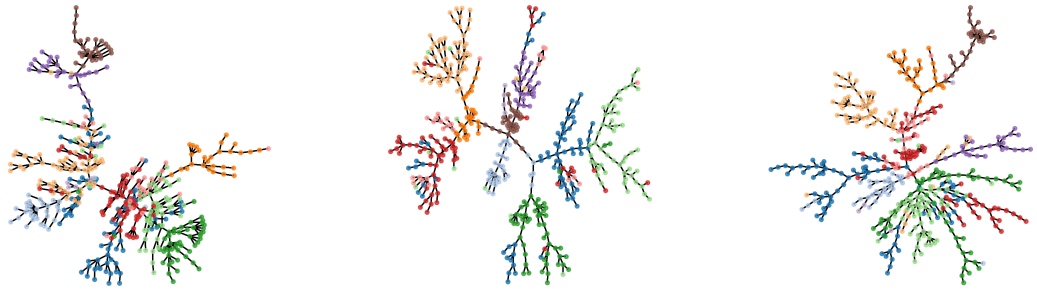
We obtained publicly available NYSE stock market data on Kaggle[15]. The original data contains daily open, high, low, closing prices, and volume from 8/13/2012

[15] C. Nugent 'S&P 500 stock data - Historical stock data for all current S&P 500 companies', 2017-2018. [Online]. Available: `https://www.kaggle.com/camnugent/sandp500`. [Accessed:

to 8/11/2017. Because not all stocks traded for the whole duration we only select the stocks which did for the last 1250 days ($\approx$ 5 years) which left us with 447 stocks, and furthermore we, in this case, were interested in a time horizon of 5 years in trading days from 8/23/2012 to 8/11/2017. We consider the daily trading closing prices which are use to compute the daily returns such that :

$$r(t) = \ln(P_{t+1}) - \ln(P_t) \tag{24}$$

The final data set has a time-series Length $D = 1249$. Using Eqn. (24) we consider three cases for the correlation matrix: i.) the full correlations, ii.) denoising using IMN (See Sec. III C 2), and iii.) cleaning the matrix using a RMT method (See Sec. III C 1).



(a) Market Mode (b) IMN (c) RMT

FIG. 16: in a) S&P500: Market Mode Correlation-based MST of 447 stocks over 1249 trading days (Sec. V A). in b) The Market Mode was removed using IMN (Sec. III C 2), and in c) using RMT (Sec. III C 1). Colors refer to GICS sectors (See footnote 21)

Financial markets are perpetually evolving living ecosystems, and this is illustrated in the lack of available true sectoral classification of publicly traded companies. In the process of clustering stock market data [16], we wanted to compare the results of our algorithms with industry standard classification but we faced the following difficulties:

We consider the following industry classifications [17]: The New York Stock Exchange (NYSE) uses the Global Industry Classification Standard (GICS) [18] (which

01-Dec-2017]

[16] A very nice review of clustering methods applied to financial datasets is available at [46]

[17] There is no consensus industry classification used in the financial services industry.

[18] The GICS counts 11 sectors, 24 industry groups, 68 industries, 157 sub-industries, and is updated annually. For more details on their hierarchical industry classification system `https://www.msci.com/documents/10199/4547797/GICS+Structure+effective+Sep+1%2C+2016.xls`

we use here). The National Association of Securities Dealers Automated Quotations (NASDAQ) and the London Stock Exchange (LSE) both use the Industry Classification Benchmark (ICB)[19].Industry classifications have sectoral, industrial and sub-industrial levels. Although commonalities exists one is left to determine the equivalences when information aggregation is required across different markets.

GICS, and ICB are static classifications which are updated at irregular intervals (i.e. GICS every year, ICB from weekly to yearly updates). The focus of these companies is to provide long term structural trends of financial markets. As such they lose their usefulness if one wants to consider the impact of rare events such as financial crashes which significantly alter the behavior of businesses. They also do not consider how the diversification of investments and activities affect their respective classifications. The case of Amazon can be argued to illustrate the idea behind the Adaptive Market Hypothesis (AMH) [43]: Amazon's GICS' sector is Consumer Discretionary. GICS uses this sector to classify companies whose activity they deem "most sensitive to economic cycles" [20]. It is unclear what is meant by "sensitive" in this instance as there are many possible interpretations, and this sector is very heterogeneous. Perhaps it highlights the adaptive nature of Amazon's business interests which started first as an order-to-delivery e-commerce bookstore but based on Fig. (16a) is now closest to the Information Technology sector.

The life cycle of publicly traded companies can be short. Firms go public and private at relatively high frequency when compared to biological evolution on a human timescale as motivated by Farmer in [18]. The inclusion or exclusion of individuals in an ecosystem can and should have an impact on its structure based on how important the individuals are to the groups. When we looked for GICS data for our time-series, a number of companies had gone private since Aug 2017, and GICS classification had been updated without reflecting these new changes for these companies. Gathering data on these companies which translated into the newer nomenclatures was thus rendered more difficult. Yet again illustrating the need for expert-free unsupervised methods.

Finally, while as previously stated, GICS and ICB intend on providing data which capture long term trends. Financial markets are populated with participants (i.e. pension funds, high frequency trader, asset managers etc...) each holding a diverse set of objectives, who do not necessarily operate on the same time scales or have the same investment horizons. If one goal is to provide comprehensive analyses of the multiple existing dynamics in markets, tools which capture these trends, and methods which subsequently find relations between them should be prioritized.

This motivates us to argue that the highly dynamic nature of financial markets renders the use of static classifications problematic to a certain extent.

---

[19] The ICB counts 10 industries, 19 super-sectors, 41 sectors, 114 sub-sectors, and is For details on their hierarchical industry classification system `https://www.ftse.com/products/downloads/ICB_Rules.pdf`

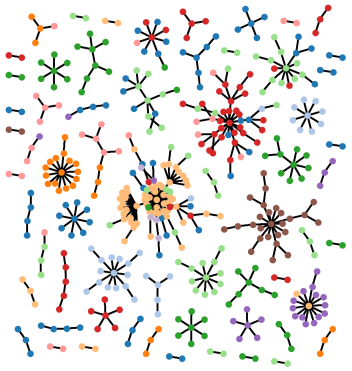
[20] A description of GICS sector is available at `https://www.msci.com/documents/10199/4547797/GICS+Sector+definitions-Sep+2016.pdf`

We use GICS's 11 sectors as the "true" economic sectors of the US financial market. These include Consumer Discretionary (74 stocks), Consumer Staples (31 stocks), Energy (28 stocks), Financials (62 stocks), Health Care (51 stocks), Information Technology (IT) (59 stocks), Industrials (58 stocks), Materials (26 stocks), Real Estate (26 stocks), Telecoms (4 stocks), and Utilities (28 stocks)[21]. Although as previously mentioned we do not believe this classification to be valid, here we make use of it as benchmark.

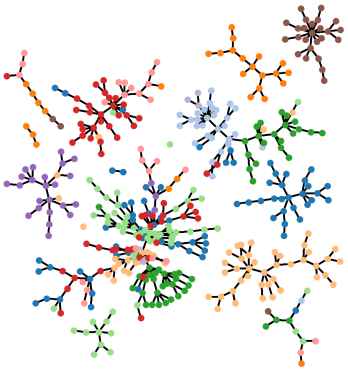
Looking at MSTs in figures (16a), (16b), and (16c), and aided by the GICS classification as legend, we notice nodes belonging to the same economic sectors are mostly located in proximity of each other as one would expect in a static world or over time-scales where the static model is a reasonable approximation.

We report SPC results in figures (17a), (17c), and (17e) respectively at $T = 0.081$,$T = 0.071$, and $T = 0.129$ for the full (K=5), normalized, and RMT cases.

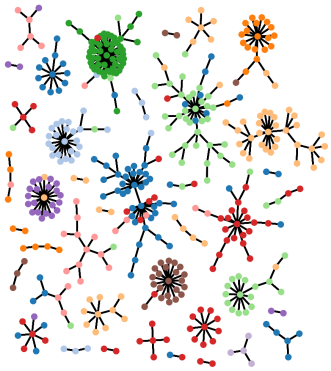
[21] Colors used for the 11 GICS economic sectors: Consumer Discretionary (royal blue), Consumer Staples (sky blue), Energy (orange) , Financials (beige), Health Care (dark green), Information Technology (light green), Industrials (red), Materials (pink), Real Estate (purple), Telecom (magenta), Utilities (brown).



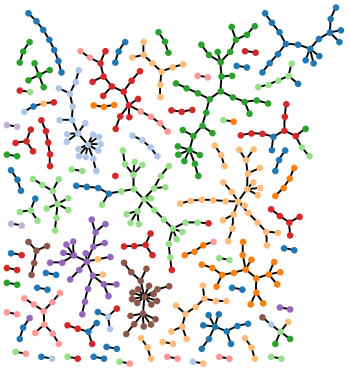
(a) Market Mode - SPC



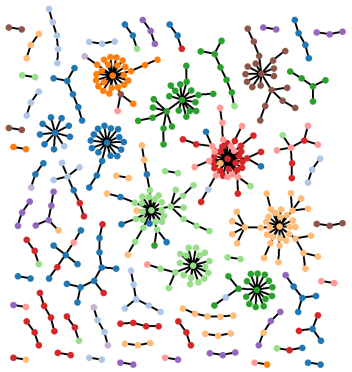
(b) Market Mode - f-SPC



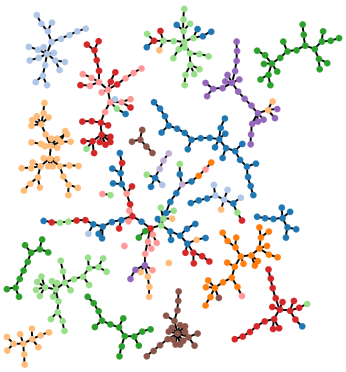
(c) IMN - SPC



(d) IMN - f-SPC



(e) RMT - SPC



(f) RMT - f-SPC

FIG. 17: S&P500: $N = 447$ stocks traded over 1249 days (Sec. V A). in a), c), and e) SPC's solution at $T = 0.081$, $T = 0.071$, and $T = 0.119$ respectively for the Full "Market Mode" sample Correlation Matrix, the iteratively normalized (Sec. III C 2) , and Noise cleaned RMT (Sec. III C 1) cases. And in b), d), and f) the f-SPC's solutions after 25k generations. Colors refer to GICS sectors (See footnote 21).

We briefly mention again one of SPC's features which consists in linking a node to its closest neighbor based on the spin-spin correlations. Using the condition $\theta > 0.5$ we construct a graph but in the case where a node has no correlations meeting our condition, it is linked to its neighbor of highest spin-spin correlation. This feature forces SPC to produce graphs without isolated nodes. At the same time, and because of this fact, we consider small size clusters are equivalent to noisy, insufficiently correlated, or unclassified observations.

SPC solutions recover GICS information as seen by their respective ARI: 0.317, 0.479, and 0.33. The solution with highest number of noisy or unclustered stocks is Fig. (17a), The financial sector is merged with many other stocks from other sectors, whereas most industries are found in one or two clusters. The complexity goes down when we move to Fig. (17c) where every sector have mostly separated into their own unique cluster, and Fig. (17e) which gives a similar picture although with more smaller unclassified clusters present.

L_c results were simulated for 25k generations, and we obtained L_c values of 113.92, 54.13, and 367.93 respectively for the Full Fig. (17b), the Normalized Fig. (17d), and the RMT Correlations Fig. (17f). Their economic GICS information recovered via the ARI were, following the same order, 0.25, 0.35 and 0.41. While Fig. (17b) has the smallest number of clusters 15, we find clusters which mostly contain firms from single industries such as the financial, utilities, Real Estate, and Energy sectors. The other clusters more or less mixed including a very large one which we could refer to as the "market". Recall that Fig. (6a) shows the correlations of the "Market Mode" are mostly positive, and one can easily infer this kind of result. Fig. (17d) and Fig. (17f) provide a cleaner pictures of the market: in Fig. (17d) there is no large "market" cluster and every industry is mostly represented in their own respective clusters. Firms, previously found in the "market" are now for most of them located in clusters representative of their respective industries. Similar situation in Fig. (17f) except the industry sectors have a better definition while a large mixed cluster remains present similarly to Fig. (17b).

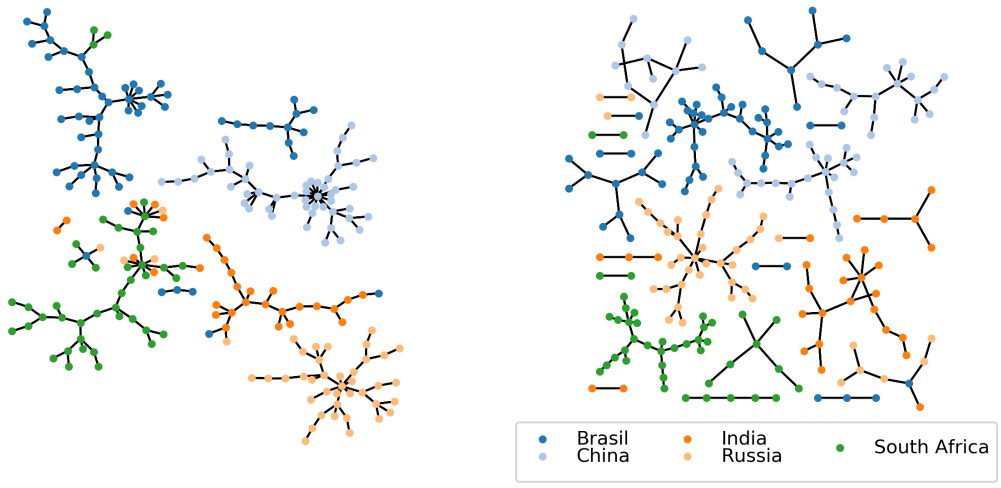
Th neighborhood search SPC performs constrains the scope of the optimization. In SPC's case, SA can only minimize the Hamiltonian H_S over the neighborhoods necessitates an additional decision in picking the neighborhood size K which acts as a hyper-parameter. The likelihood L_c is optimized over the whole range of observations effectively removing such need, and making the optimization fully unsupervised. This also means that there exists a possibility that nodes which wouldn't cluster together, because of neighborhood limitation, would in this particular case. It is unclear which is the best way to proceed.

Our second goal is to explore clustering differences which arise in our 3 cases. In Fig. (17a), and Fig. (17b) The biggest clusters have significant overlap with economic sectors except for a few large ones such as the financials cluster which also houses stocks from other sectors. The picture gets cleaner once we look at Fig. (17c), and Fig. (17d) where we now have less mixing in most clusters, and finally in Fig. (17e), and Fig. (17f) some of the clusters such as the Real Estate, Utilities, Health Care, and Consumer Staples found in the Normalized case are split. We

noticed a similar result in Sec. IV C where the L_c solution had a higher number of clusters, but they were essentially subgroup within the ones found by SPC.

B. BRICS data

We obtained publicly available BRICS (Brazil, Russia, India, China and South Africa) stock market data [22]. The original data contains daily closing prices of 226 stocks:Brazil (60), China (50), India (30), Russia (43), and South Africa (43). We will refer to BRICS as a way of listing the mentioned countries in the previously given specific order. The window spans 2005 to 2015 from which we retained the last 5 years of daily trading. The data set suffers from a missing data problem which we would make it impossible to compute correlation matrices. We deal with the problem by using the time-series missing-data which consists on computing correlations only on overlapping sections of time-series. The resulting correlation matrix is then made positive definite, and cleaned using IMN (See Sec III C 2).



(a) SPC BRICS (b) FSPC BRICS

FIG. 18: 226 BRICS stocks. Data cleaned using IMN (See Sec. (III C 2)). in a) SPC solution at $T = 0.12$, and in b) CBMA's solution with $L_c = 7.56$

Using MR the Likelihood's local maxima reached was 7.56, with 24 clusters: very little mixing, and the 5 countries are concentrated inside 1 to 3 clusters per country

[22] C. Nugent 'S&P 500 stock data - Historical stock data for all current S&P 500 companies', 2017-2018. [Online]. Available: `https://www.kaggle.com/camnugent/sandp500`. [Accessed: 01-Dec-2017]

(see Fig. (18b)). This confirms our expectations which were that same country stocks should mostly belong to the same clusters, and the existence of multiple clusters per country as evidence of meso-scale industry/sectoral level classification.

We continue with the SPC result given for $T = 0.12$ in Figures (18a), and (4a) with 9 clusters. The ARI between the SPC and f-SPC solutions is 0.5. Globally speaking the same clusters are recovered except for their size being slightly smaller for the MR solution, and clusters "purity" is higher in the MR candidate potentially (stocks which belong to different countries do not mix). In both candidates Brazil financial market is divided in 2 clusters which upon a more detailed cluster analysis could reveal industry (or sector) economic subdivisions.

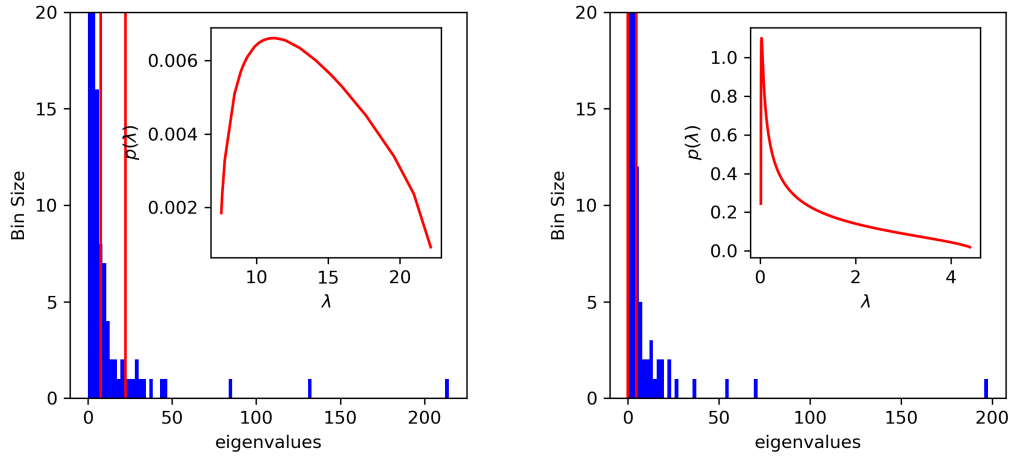
C. Kaggle Intraday US Stock Market data

Similarly to Sec. V A, we collected publicly available US stock market data on Kaggle[23]. The original data contains daily open, high, low, closing prices, and volume for daily, and intraday data.

1. Signal-to-Noise ratio's impact on data clustering.

We previously discussed in Sections III C 1 and VI how RMT allows us to identify two spectrum of eigenvalues: one which contains mostly noise, and it Wishart distributed and another with most of the information located on the upper tail of the eigenvalue distribution. We also discussed how the noise-to-signal ratio plays an important role in the ability of maximum likelihood based methods such as f-SPC to achieve good solutions. We would like here to compare clusters extracted from correlation matrices spanning identical time windows but with different time-series lengths. We consider 1224 securities (1103 stocks, and 121 Exchange Traded Funds (ETF)) traded on all us stock exchanges (NYSE, NASDAQ, and NYSE MKT) from 2017-11-17 to 2017-12-06. We use two different bar sizes: 1-hour bars, with time-series length $D = 88$, and 5-mins bars with $D = 973$. We mention again the Signal-to-Noise ratio q which for both data-sets is respectively 13.9, and 1.25. The correlation matrices eigenvalue distributions are shown in (19a) and (19b).

[23] B. Marjanovic 'U.S.-based stocks and ETFs trading on the NYSE, NASDAQ, and NYSE MKT.', 2017. [Online]. Available: `https://www.kaggle.com/borismarjanovic/daily-and-intraday-stock-price-data/`. [Accessed: 17-Sep-2018]



(a) 1hour bar (b) 5 mins bar

FIG. 19: Kaggle US stock Market Data: a) 1Hour Bar data and b) 5Min Bar data. The spectrum of eigenvalues is plotted against the bin-count distributions. The vertical red lines delimit the Wishart range where RMT predicts noisy eigenvalues are located. Insets: in a) and b) are respectively plotted the density function of the Wishart range for both data sets. Data sets with high signal-to-noise ratio have power-law distributed Wishart eigenvalues.

with 31, and 12 with upper tail sum 666, and 735, and lower tail sum 0.56, and 205, Wishart range sum 557, and 284. The shape tends toward that of an heavy tail distribution as Q goes down. Recent research suggest that the correlation matrices of Deep Neural Network fully connected final hidden layers exhibit similar behavior (see [47]) as these models are trained. Here one could conjecture that low Q correlation matrices, and power-law like noise eigenvalue spectrum distribution are characteristics of "highly informative" data-sets.

We look at the descriptive statistics of the solutions obtained with MR (see Sec. II F) respectively for the 5-mins bar, and 1-hour bar data: Likelihood: 613, and 790. Number of clusters: 209, and 234. Average cluster size: 5.85, and 5.23. Median cluster size: 4 for both data-sets. The ARI between the two configuration is 0.21. We notice that despite the difference in time-scale both solutions share a certain amount of similarity expressed by the ARI value. Based on the average and median clusters sizes, it can be said that the distribution of cluster size is indeed impacted by noise: Noisier data sets have higher number of clusters, and smaller cluster sizes.

2. Short-Term Emergent Dynamics

We consider 1 Hour bar data for 1186 securities traded from 2017-06-26 to 2017-12-06 composed of 1060 stocks, and 126 ETFs. Correlation matrices are usually

computed over years of historical data in the financial industry, here we split the time-series in 4 consecutive mini-series. We do so because rolling-windows by design prolong trends whereas we are interested in detecting different trends on shorter, albeit noisier, windows as an attempt to illustrate their existence and compare them to longer term trends which are reflective of the economic sectors of the economy. This is a not so direct attempt at illustrating the existence of "Trend Following", and "Mean Reverting" strategies in financial markets. Markets are said to be populated with "chartists" and "fundamentalists" which translate into the former "emergent" and the latter "corrective" macroscopic dynamics. The 5 windows are the following:

0) 2017-06-26 - 2017-12-06
1) 2017-06-26 - 2017-08-04
2) 2017-08-07 - 2017-09-15
3) 2017-09-15 - 2017-10-26
4) 2017-10-26 - 2017-12-06.

TABLE N: US stock market Time-series data: windows 0 is split into 4 sub-windows 1, 2, 3, and 4.

We look at the descriptive statistics of the solutions obtained with MR (see Sec. II F) respectively for a) through e): Likelihood: 434, 517, 488, 404 and 488. Number of clusters: 210, 210, 218, 232 and 215. Average cluster size: 5.64, 5.64, 5.44, 5.11 and 5.52. At first sight every windows share similar overall statistics. A better perspective is given in Fig. (20) showing the heatmap of ARIs between every windows. The individual mini-series cross ARIs are all close to 0 (light color), and clearly don't share much similarity except with the window 0 which spans the entire time-range and overlaps them every single one of them. Thus confirming that short term trends exists, and are non-stationary.

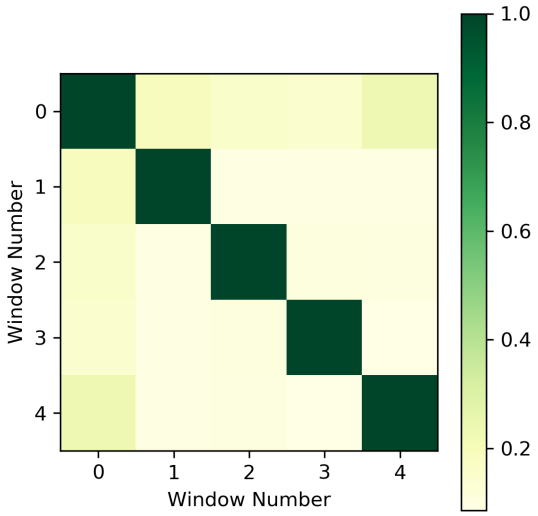


FIG. 20: Heat Map of Cross ARI values between all windows in Table N : The small windows do not overlap except with the largest window (window 0) which includes them all. Short trends do exists but they quickly disappear.

D. Financial Crash: Historical Analysis.

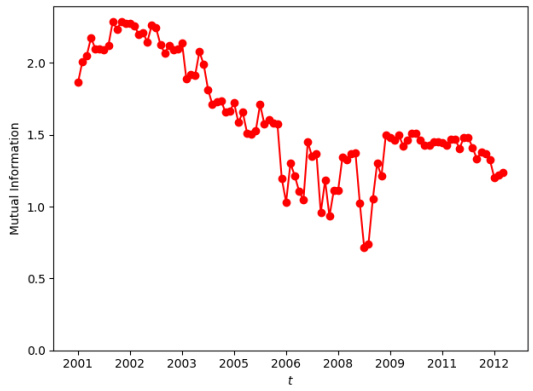
We use the same US stocks market data obtained in Sec. V C for which we slice dates starting in 1997 until the end of 2012. The resulting data set is comprised of 120 stocks: unfortunately it is not a complete picture of the US stock market of that era as stocks which were not trading in 2017 were not included in this data set (unlike in [55]). 100 correlation matrices each spanning 1000 trading days are computed, and we proceed to cluster these matrices using the CBMA (see Sec. II F). We don't need here the successive phase transitions observed in this data but we are more interested in dynamic evolution of the clustering structure. For this the CBMA, given its very fast execution time, allows to efficiently process multiple data-sets..

In Fig. (21) we show the temporal evolution of 3 Clustering "sklearn" clustering measures:

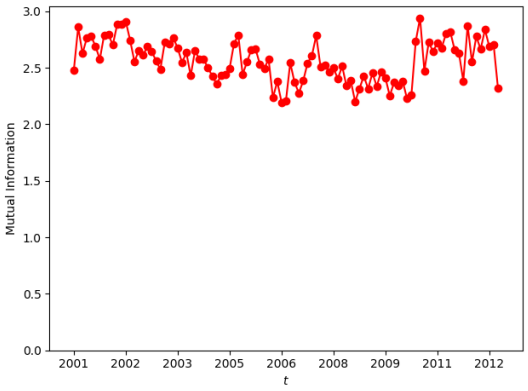
1. The Mutual Information (MI)
2. The Adjusted Mutual Information (AMI)
3. The Adjusted Rand Index (ARI)

We proceed by computing the clustering persistence between a window at time t_i against t_{i-1}. Our expectations are that in stable market conditions persistence should remain high and stable whereas in the occurrence of extreme events such as

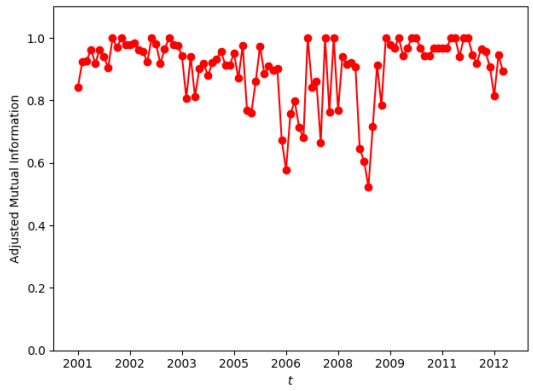
a critical transition in physical systems or a crash in a financial market, it would show a discontinuity. This was similarly implemented in [55] with the ARI, and a different network-based data clustering method was used. Information Theoretic measures such as the pairwise spin-spin mutual information were previously successfully demonstrated in [5, 29, 49] to peak at the critical temperature T_c similarly to the magnetic susceptibility χ. Additionally in [31] Harré and Bossomaier show that's also the case but with stock prices time-series around Market crashes and they are followed by what appears to be phase transitions just like what we can observe in statistical mechanics. Our objective here is to investigate once more this problem using our own tools.



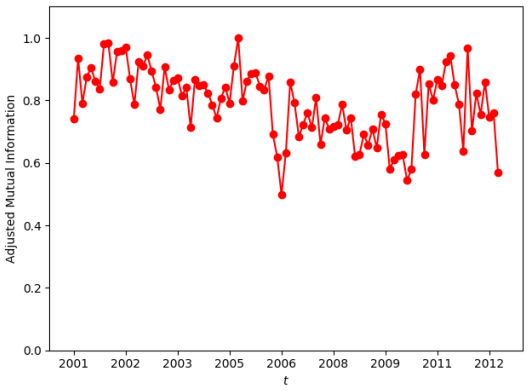
(a) MM - MI



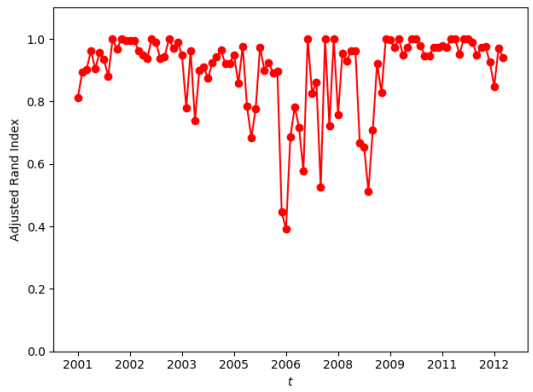
(b) RMT - MI



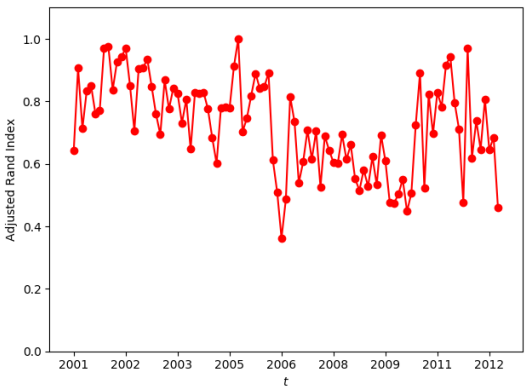
(c) MM - AMI



(d) RMT - AMI



(e) MM - ARI

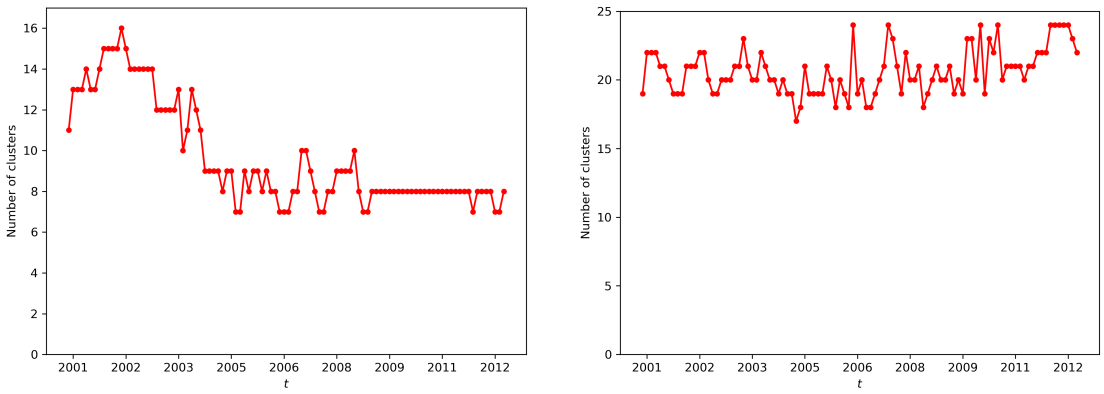


(f) RMT - ARI

FIG. 21: Temporal clustering 100 Correlation matrices using CBMA (See Sec. II F). Each red dot represents a measure between two 1000 days windows. We used the Mutual Information (MI), the Adjusted Mutual Information (AMI), and the Adjusted Rand Index (ARI). On the left we measure dynamical cluster similarity of the original data with the Market Mode, and on the right without after de-noising using RMT (See Sec. III C 1.)

Figure (21) shows us a collection of clustering measures computed on two data-sets: On the right we have the temporal clustering measures (MI, AMI, and ARI) of the 100 cleaned (see Sec. III C 1) rolling windows correlation matrices, and on the left we have clustered the same data while keeping the "Market Mode" (or the noise) in. In Figures (21b), (21d), and (21f) we can observe a relative stability (ignoring monthly fluctuations) of the measures which respectively remain between [2.5-3], [0.5-1], and [0.4-1] for MI, AMI, and ARI. In Figures (21a), (21c), and (21e) the MI, AMI, and ARI respectively range between [0.8-2.2], [0.5-1], and [0.4-1]. All our measures, except in Fig. (21b), show a negative jump around the year 2006 followed by a series of large fluctuations until another jump, this one positive, occurs in 2010. From 2010 until the end of 2012 there are no significant changes.

We now discuss the clustering measure tools. There is no significant difference between the AMI, and the ARI in our example. However MI displays, in both the market mode and RMT cases, a dynamic which is not necessarily trivial to interpret. Whereas, in the RMT case, AMI, and ARI display an overall decreasing trend, MI does so but to a lesser extent which could be interpreted as a sort of "global" stability. Before continuing onto the market mode case we briefly mention that one major difference between Adjusted and Non-Adjusted clustering measures is their insensitivity to the size of the data-set and the number of clusters. Adjusted "for chance" measures mainly compute cluster similarity whereas Non-Adjusted measures such as MI are affected by the number of clusters. One clearly sees the effect this has on the measures by looking at Figures (21a), and (21c) also aided by Fig. (22a). Finally Fig. (22b) shows that the number of clusters extracted from the "cleaned" data remains stable between 20 and 25 throughout the entire period (1997-2012) as was similarly determined in [55]. It's possible this may be one reason why MI in the RMT case is slightly uninformative: The underlying structure of the financial markets, or the fundamental number of sectors/industries didn't change much despite the market fluctuations, and crashes. On another hand, Figures (21a), and (22a) show that between 2002 and 2004 the number of clusters in the US financial market has roughly halved.



(a) MM - Number of clusters as a function of time (b) RMT - Number of clusters as a function of time

FIG. 22: Financial Market temporal cluster sizes. Each red rot represents the number of clusters of one 1000 trading day window cluster configuration for 120 stocks. On the left we have counted the number of clusters for the original data with the Market Mode, and on the right with the market mode removed using RMT (see Sec. III C 1). The Market Mode is characterized by a small number of clusters (average $\approx$ 10) compared to the cleaned data (average $\approx$ 21). The Market Mode also shows approximately twice the number of clusters between 2001 and 2004 as the following period between 2004 and 2012.

We offer an analogy to motivate our reasoning: Consider a financial market mapped to a spin system, where spins are the random variables, the states occupied by the spins are the sub-states of the financial market as a whole, and the sufficient statistic, its average thermal energy $\langle H_S \rangle$, is a Hamiltonian as similarly defined by Eqn. (3). Recall that in the SPC framework the system goes through phase transitions during which the number of clusters (or sub-states) changes as it goes from one order to another. A slight clarification must be made here when the "order-to-disorder" term is used: In Statistical Mechanics it is often the case of a system going from the Paramagnetic Phase (disordered phase) where all spins are unclustered to the Ferromagnetic Phase where all spins are found in one giant cluster. SPC is a departure from this simple case where the SP-Phase can exhibit multiple stages before reaching the giant cluster stage going from high to low temperatures. It is trivial to observe that social systems only exists in clustered states at all time, so one should be careful when talking about "disorder-to-order" as the system is always ordered.

What is of interest here is the loss of current, or gain of new, order as a transition happens: Unlike AMI and ARI, MI captures the change in the number of sub-states, and expresses it not only by a discontinuity but also by a drop in the MI level where it settles after the jump. This change can also be interpreted as a decrease in the entropy of sub-states distribution: Imagine switching from a Portfolio (the financial market) of 16 to one of 8 assets (see Fig. (22a)). It is however left to rigorously determine what a decrease in entropy signifies in term of financial stability or risk

management. The RMT methods are industry standards used to clean correlation matrices which are then used to build investment portfolios. One may argue that the underlying structure of the market remained roughly stable (see Fig. (22b)) but we argue some information gets left out by using these methods or as one could put it " there is order (information) within the chaos".

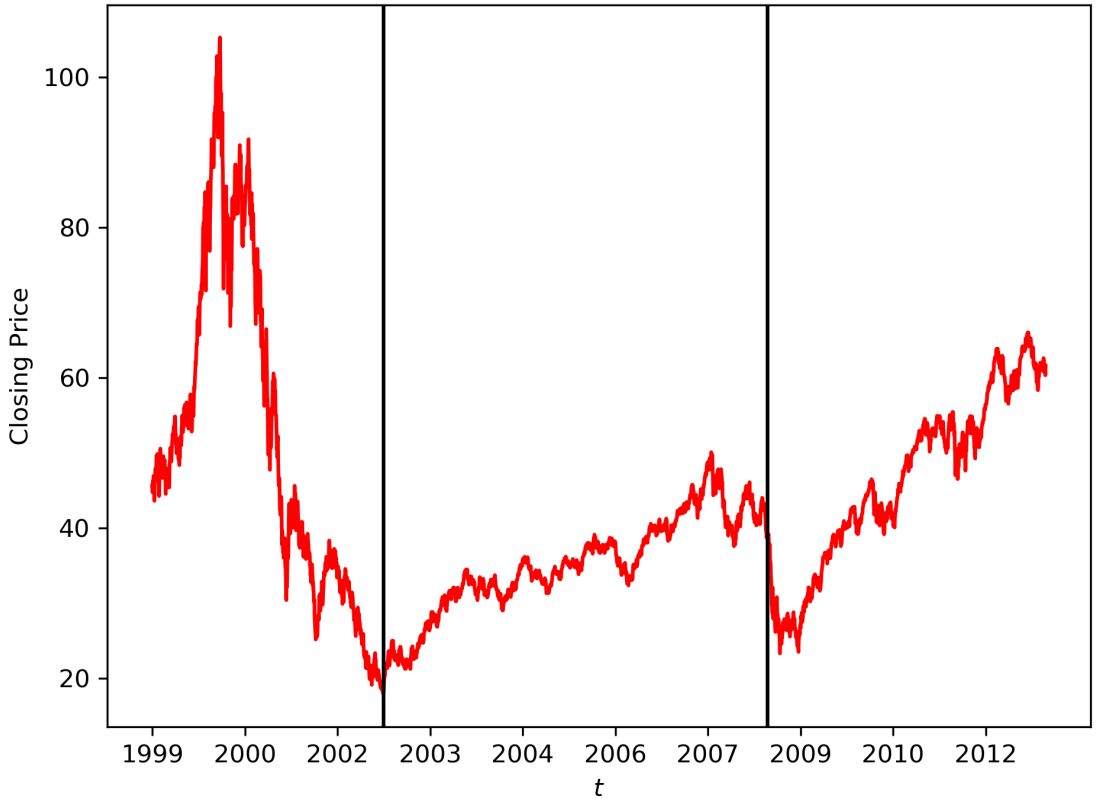


FIG. 23: Daily closing prices for Invesco's QQQ ETF tracking the NASDAQ 100 Index. This index is one of the longest running indices included in our data set. It captures the overall price trends of many tech companies stocks in the market. We are able to identify 3 Phases separated by 2 vertical black lines: (starting from the left) from 1999 to 2003: the Dot-Com bubble, from 2003 to 2008: Post-Dot-Com Pre-Lehman Brothers, and from 2008 onward: Post-Lehman Brothers.

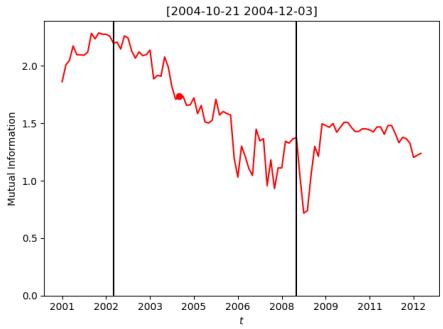
To further our analysis we look at the Invesco QQQ ETF which tracks the NASDAQ 100 Index. The NASDAQ 100 index is composed of 103 securities from the top 100 largest non-financial companies publicly traded on the NASDAQ stock exchange. A better choice would have been an index tracking the S&P500 however the ones found in our data-sets all recorded data as far as 2006 which wouldn't give a complete picture of the historical events of the early 2000s. The QQQ which itself only started trading in 1999 (and not 1997) isn't perfect by any means, and is skewed toward large market capitalization, should still allow us to get a good

enough approximation of the events. We plotted in Fig. (23) the historical closing prices of the QQQ from its inception to the end of the 2012. We also added two black vertical lines which respectively signal the bottom of the "dot-com" bubble on 2002-10-09, and the day Lehman Brothers filed for bankruptcy on 2008-09-15. We use these two lines as demarcation of 3 potential phases in that historical period of the US financial market:

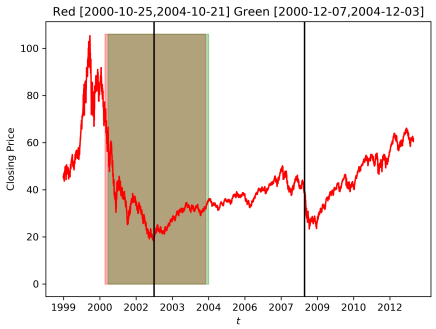
1. The Pre-Dot-Com Bubble bottom Phase
2. The Post-Dot-Com Bubble / Pre-Lehman Brothers Phase
3. The Post-Lehman Brothers Phase

We finally attempt a description of the system as it moves through these phases in Fig. (24) where plotted: (On the left) the market mode MI, and (on the right) QQQ's price as functions of time. We also add to both plots the same historical markers shown in Fig. (23), and (on the right) we added the overlapping windows used to compute MI: the MI data point shown (on the left) corresponds to the MI computed between the green window and the red one.

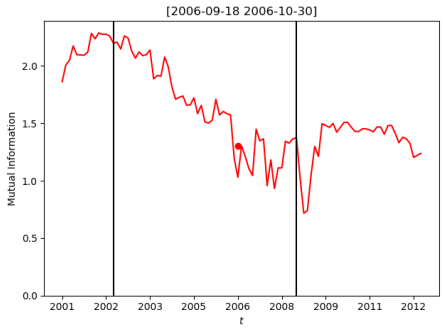
1. Figures (24a), and (24b) show that between 2002 and 2006 MI is steadily decreasing, while the rolling window spans a period which includes the descent from the dot-com bubble high, its bottom, and post-dot-com bubble recovery as the number of clusters progressively goes from 16 the maximum over the entire window to 8.
2. Figures (24c), (24d): As we move away from the Dot-Com's bubble bottom, the rolling window now only includes data incorporating a mostly upward trending market signified by the increasing QQQ price. The number of clusters however oscillates between 7 and 10 which is translated by a fluctuating MI between 1 and 1.5
3. Figures (24e), (24f): Once Lehman Brothers bankruptcy was made official, the following months characterized by the market's MI plunging to 0.8 its lowest value in the entire window, and the number of clusters becomes stable at 8 for the remainder of the window. Figures (24g), and (24h) show that the MI value is back to a stable (although with a slight decreasing trend) pre-2006 level of 1.5. It pays to notice that despite the 2008 Great Financial Crisis (GFC) the number of clusters, unlike what happened with the end of the dot-com bubble, didn't change even as the rolling spanned the QQQ's high Pre-Lehman, its bottom, and the upward trending market which followed the crash. By looking at 100 trading days Harre et al [31] are able to detect, using their pairwise Mutual Information between time-series a surge shortly after April 2002 which they consider an anomaly because not associated with any historical market event. The dot-com's bottom happened on 2002-10-09 which leads us to believe that our simulation may have shed some light on



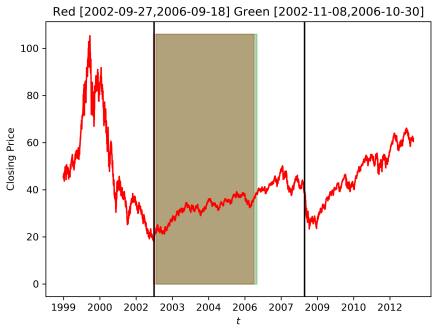
(a) MI - Transition from Dot-Com to Pre-Lehman



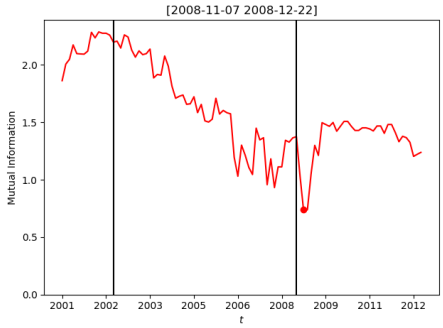
(b) Overlapping window Dot-Com to Pre-Lehman



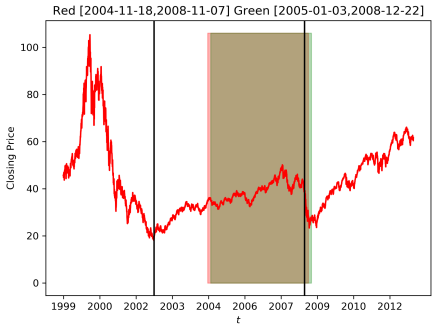
(c) MI - Pre-Lehman Phase



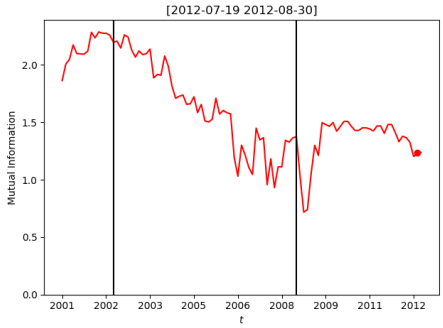
(d) Window Pre-Lehman Phase



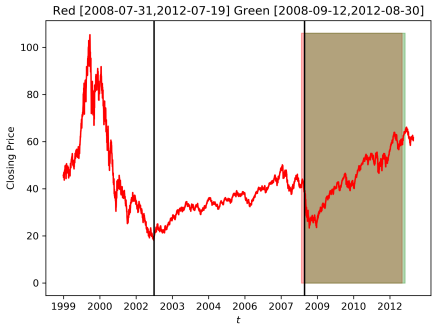
(e) MI - Overlapping window Pre to Post-Lehman Phase



(f) Overlapping window Pre to Post-Lehman Phase



(g) MI - Post-Lehman Phase



(h) Window Post-Lehman Phase

FIG. 24: On the left the dynamic Mutual Information of cluster configurations. See Sec. V D for commentary.

a phase transition of the market from 16 clusters to an average of 8. This may be a sign that those two bubbles, if one may call them this way, are different in nature. A quick historical assessment is the absence of Bail-out as businesses failed during the dot-com bubble, as opposed to the 2008 GFC for which the US government intervened to save Bear Stearns, Fannie Mae and Freddie Mac, and American International Group (AIG) which are major financial institutions involved in the Subprime Mortgage crisis.

Information theoretic tools are able to detect a crash has happened however one would be interested in developing tools which peak before the crisis occurs. In [5, 31] Mutual Information, and Transfer Entropy methods are both used to study critical transitions. Transfer entropy is seen to peak pre-Phase transition, however tested on historical data the tool also exhibits peaks where crashes do not happen leaving room for further research on the subject matter.

VI. Discussion

In this dissertation, we were able to successfully implement SPC, and f-SPC, and we tested those methods on synthetic, and real data. If there exists significantly different structures within the data, SPC will exhibit multiple transitions within the SP-phase. As the temperature is varied, the couple Susceptibility χ and Average Magnetization $\langle m \rangle$ signal the occurrence of phase transitions. The spin-spin correlation G is indirectly linked to the interaction strength J and the densities found in the data. The method has the advantage of being unsupervised for the most part, it does not necessitate *a-priori* knowledge of the number of clusters, and makes no assumption about the distributions of the data. While SPC performs a neighborhood search, which requires picking a value for K, it does not affect the simulation significantly for large data sets; as was previously seen in [8]. The parameter θ is set to 0.5 and helps decide clusters membership. We clustered at every temperature within a pre-determined range such that we do not need to identify "clustering temperatures T_{clus}" like in [7] but we do so at the expense of additional computational cost. We note that in the literature there are modifications of the Potts model clustering which automate parameter selections for the clustering temperature T_{clus}, the local length scale a, and the cluster membership threshold *theta* through validation based calibration. [54].

Once we have a hierarchy of configurations, such as in Fig. (14c), one needs to select appropriate clusters representative of the different regimens of the SP-phase. This is an easy task as can be seen in Fig. (10c) when the number of cluster is low, and the clusters have similar densities which establishes a stable phase for a relatively wide temperature range. On the other hand if the number of clusters is high, and the data is composed of clusters of different densities, the susceptibility, which tracks the variance of biggest cluster has its limitations [40]. As the size of the data-set increases the susceptibility is useful as a tool to locate the final transition and lowest clustering level.

f-SPC only requires the correlation matrix and is completely unsupervised. The randomly generated population is diversified at every iteration by applying as many as 7 different mutations. It's a fast and deterministic algorithm (at least in its CBMA version) while SPC is MCMC based and requires statistical averages. The computation time is affected by the order of the observations in the data [9]. We noticed that ordering our data based on the order of one of the observations' closest "neighbors" produced better results which should motivate further exploration of potential heuristics dealing with this issue. The CBMA alleviates these issues: we have essentially migrated from a Genetic Algorithm-like model by removing the cross-over function, and borrowing from agglomerative graph-based data clustering methods. We can confidently say CBMA is a serious contender in the realm of the state-of-the-art correlation based clustering.

L_c measures the quality of cluster configurations: its value is computed from the clusters sizes n_s and the intra-cluster correlations c_s. The optimization is global which, as opposed to SPC, avoids the need to determine a neighborhood size K. There exist problems where choosing a sufficiently big K has an non-trivial impact on SPC's solutions. One such example would be the existence of a relatively low density and sparse cluster in a data set mostly composed of high density clusters. Low values of K would fail to recover the low density clusters which would remain unclassified whereas this isn't an issue for f-SPC which would perform much better.

f-SPC results are consistent for high dimensionality data-sets. if we consider the metric used to evaluate the noise in correlation matrices $q = \frac{N}{D}$ as the ratio of the dimension over the number of observations in the limit of $N \to \infty$. We recall that in [70], q encodes the noise level of the eigenvalues of the sample covariance matrix. q values for our problems are 250 for the two circles, 13.69 for the wines, 166.66 for the 3D blobs, 1 for the 500D blobs, 37.5 for Fisher's Iris, 7.81 for the MNIST digits, and 0.35 for the NYSE Kaggle data. The L_c results consistent with SPC were the 500D blobs, MNIST, and the NYSE stock data which all confirm that a low q is necessary to compute appropriate correlation matrices. We want q to be as small as possible, and if possible close to 0. This is not always the case, and we have tested ways to de-noise the correlation matrix (Sec. III C 2, and III C 1) in the case of financial time-series but it is unclear at this time what would the solutions be in other cases. In [24] Marsili and Giada derive L_c, and along the way they assume that $D \to \infty$ which in turn means one has to consider the finite size effects of the method. Fig. (5b) and Fig. (5b) tell us that if we were to visualize the L_c's objective surface, depending on dimensionality of the problem we could face a "rough" space. One could consider L_c as a sort of modularity function just like in the Network Science literature. One major Network Science problem is the efficient detection of communities inside networks. Similar to our work, cluster configurations are the input of the modularity function Q which, through diverse heuristics, is maximized. However it is well known [26] to Network Scientists that modularity objective surfaces are degenerate: many significantly different clustering results have similar modularity, or in our case, higher likelihood than the true clustering.

We compare this to the modus operandi of SPC: The generative model of the SPC

is the Gibbs-Boltzmann distribution which not only validates clusters locally using Eqn. (9). It is a bottom up approach as opposed to global optimization methods which are top down. One assumes that there exists multiple realizations (micro-states) of the generative model, the so called "equivalence classes" which are valid representation of the data. In order to link micro and macro-state one could pick any one micro-state translating into the desired macro-state: Maximum Likelihood methods essentially achieve this feature by searching the space of solutions for any candidates meeting the global objective. We argue that in complex systems, the existence of equivalence classes as illustrated by the degeneracy of clustering objective surfaces leads to the Maximum Entropy principle [38] as an alternative optimization device. The generative model generates equivalence classes (among which are included maximum likelihood candidates) each with differing probabilities, and one then needs to probabilistically combine them to achieve some sort of representative weighted average.

Or as motivated in Sec. II G 1 There may not be a representative average, but a collection of states arising in nature with probability.

We suspect a way to deal with cases where D is small compared to N, and indirectly $q \gg 1$, would be a modification of L_c by adding an additional term acting as a regularizer which could account for the number of clusters. Our rationale follows that L_c as an objective function is degenerate with multiple spin configurations whose likelihood are equal or very close. This degeneracy comes from, if we assume the minimum number size of clusters to be 2 (no singletons), the number of possible configurations $(\frac{N}{2})^N$ which for a case $N = 100$ would be on the other of 10^{169}.

Finally one is left to decide which de-noising method is deemed optimal and as a consequence which clustering one prefers. The assumptions in both methods have their validity and should be carefully considered. Whereas IMN (Sec. III C 2) consider the covariance matrix as IID normal random variables, RMT (Sec. III C 1) predicts a spectrum of random matrices eigenvalues exists which is pure noisy signal. The noise is removed by reconstructing the data without the noisy eigenvalues whose number increases with dimensionality. We suspect a proper way of deciding which method is optimal is the implementation of such methods as bases of trading strategies.

VII. Conclusion

In this dissertation we have presented two unsupervised data clustering algorithms inspired by the Potts Model [73]. Using SA (SPC Sec. II D 1) optimizes the Potts Hamiltonian which at every temperature explored results in a hierarchy of cluster configurations. We show that the parameter-free Marsili and Giada's (Sec. II D 2) maximum likelihood methods implemented with a modified version of Hendricks et al. Parallelized Genetic Algorithm recover solutions similar to those found in the SP-phase Fig. (1). This was done by comparing the SPC solutions to the f-SPC one using the ARI [42]. By comparing the Likelihood of SPC solutions to f-SPC one we showed that f-SPC have higher likelihood which prompted an additional

discussion on implication for statistical inference in complex systems.

We also provided additional validation, and connection between the Free energy F minimization of thermodynamic systems, and the maximization of the Likelihood L_c in Sec. II G 1 where we have shown that the SPC cluster solution with the highest ARI when compared to f-SPC's maximum likelihood solution is the one located at a temperature very close to the minimum free energy. This temperature is the critical point signaling a Super-Paramagnetic to Paramagnetic transition, and provides additional evidence that not only as claimed in [24, 48] maximizing L_c brings the systems in its ground state, and near criticality.

The methods were tested both on toy test cases, and real stock market time-series data illustrating their universality provided an appropriate similarity metric is selected such as the Euclidean distance of the Pearson correlation coefficient. We showed that the results are similar to the 11 standard GICS economic sectors however the differences in the number of clusters, and their composition should be cause for concerns with respect to the use of GICS classification for risk management purposes.

Building on the work presented in this dissertation we would like to perform cluster analysis of stock market intra-day time-series. The last thirty years have seen a magnificent increase in technological power which enabled simultaneous trading on multiple time scales. The so called "High Frequency Trading" paradigm increased tenfold the amount of stock market data, and has significantly impacted the market participants behaviors at shorter time scales. It is therefore natural to consider new species of market participants to exist, and that they come with different objectives to fulfill than longer timescales participants. We conjecture this would be reflected in clusters of stocks disconnected from their economic sectors. We assume traders use all information available to make decisions however the rate of economic information released about publicly traded firms can range from once a month to once a year. This rate is significantly lower than that of high frequency trading which leads us to think it is therefore impossible for high frequency traders to trade based solely on economic information alone, and we suspect different objectives may be at play.

One logical next step is what we call Dynamical Cluster Analysis (DCA): Events such as financial crises like the one which preceded the 2008 Great Recession can be investigated at the intra-day scale. Here again we conjecture shocks to the system irremediably affect strategies, and clustering structures are less persistent with time as in [55]. Ultimately some sort of quantification of clustering on different temporal scales could be useful towards probing potential hierarchical causal affects given that different effective theories may dominate at different scales [71].

We have so far worked with changes in price returns as our factor model. The derivation and formulation of the Giada-Marsili L_c allows for multivariate clustering: We can use F by N by N Correlation matrices where F is the number of factors we want to include. Another challenge however would be that at this time we are not aware of an implementation of multi-factor correlation based SPC.

It is notoriously difficult to obtain trading data linked to individual market participants accounts. This kind of data would be extremely useful to directly, not only

study traders' behavior, but begin to understand the kind of ecosystem a financial market is. Unfortunately one is only left with the possibility of proxy studies through the dynamics of the traded securities. One alternative approach is to create simulated trading agents, and cluster them using our unsupervised methods using technical trading strategies available in the literature.

Finally, one should consider a complete rewrite of f-SPC as a GPU-based Parallelized Genetic Algorithm to take advantage of modern computing power available at our disposal and further reduce the computation time.

Appendix

A. The Giada-Marsili Asset Pricing Model

We restate Ansatz model in its Eqn. (11) version.

$$\xi_i(d) = g_{s_i}\eta_{s_i}(d) + \sqrt{1 - g_{s_i}^2}\epsilon_i(d)$$

We briefly discuss factor models ubiquitous in the finance industry and literature within the Arbitrage Pricing Theory framework [53] as an attempt to provide explainability to assets' returns. Let us consider the following model:

$$\xi_i(d) = \alpha_i + \epsilon_i(d) + \sum_j \beta_{ij} f_j \tag{A1}$$

where β_i is a constant coefficient of random factor's f_i importance for the asset return, ϵ_i the stock random effect similarly in Eqn. (11), and α_i is the unexplained component of the stock return. Popular single and multi-factor models are typically the Capital Asset Pricing Model (CAPM) [63] and the Fama-French 3-factor Model (F&F) [16]. In the CAPM framework asset returns are modeled the following way:

$$\xi_i(d) = \xi_f + \alpha_i + \epsilon_i(d) + \beta_i(\xi_M - \xi_f) \tag{A2}$$

with $\alpha_i = 0$ [24], and the single factor $f = \xi_M - \xi_f$ the excess return between the market ξ_M and risk free asset ξ_f returns. CAPM's beta $\beta = \rho_{iM}\frac{\sigma_i}{\sigma_M}$ with ρ_{iM} the correlation of asset i to the market M, σ_i and σ_M are respectively standard deviations of asset i, and market M. The CAPM essentially explains that excess returns above the market are conditioned on the risk of asset i over that of the market as a whole and leads to what is known as the Efficient Market Hypothesis (EMH). CAPM is a great but overly simplistic model which may not provide the best fit to data given that in practice $\alpha_i \neq 0$ thus highlighting the fact that the market, to a certain extent, isn't efficient. Overcoming CAPM shortcomings requires one to come up with more complex multi-factor models, and the F&F is one popular example:

$$\xi_i(d) = \xi_f + \alpha_i + \epsilon_i(d) + \beta_{iM}(\xi_M - \xi_f) + \beta_{iS}\xi_{SMB} + \beta_{iv}\xi_{HML} \tag{A3}$$

with SMB the "Small [market capitalization] Minus Big" and HML the "High [book-to-market ratio] Minus Low" which respectively capture the excess returns of small caps over big caps assets, and of the value stocks over growth stocks. [25] This model is an improvement over CAPM and it's easy to understand, just by their

[24] which makes sense if we believe the Efficient Market Hypothesis (EMH) [44]

[25] historical factor values can be found at `http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html`

definitions alone, how the additional factors could bring more explainability power to asset pricing models. There are no absolute stochastic system per se by that we mean processes are deterministic up to a point where their degrees of freedom retain explanatory power. What we call randomness is the unexplained residual, and thus there is no such thing as "noise". Factor models are an attempt at identifying the degrees of freedom in Asset Pricing models and one cannot help but wonder about the topology of the asset returns space: The identity, and perhaps the natural number of degree of freedoms which capture and compress all information. Limits to factor models in general and the F&F in particular include:

Factors f_j are random variables but identical for the entire collection of assets and the betas β_{ij} are estimated from data for every assets. It was however shown that The F&F factors are country specific and not universal[27]. Fama and French also introduced two additional factors [17] which yet again improved their model. That being said the model seem not be universal and its performance in the UK stock market has been criticized [20]. Thus it highlights that the composition of portfolios relative to the factors matters. Furthermore there exists alternatives such as Characteristic Based Models (CBM) which provide better out-of-sample performances than factor models for US and South Africa data [72]. There exists no strict requirement for factors characteristics if not that they typically capture macroeconomic dynamics, and should have economic meaning. We discuss the SMB factor as an example of the limitation of such approach: we recall that SMB is computed by ranking assets according to their market capitalization. Assets are then separated into 2 groups: big caps, and small caps. And the excess returns between these two groups are computed.

Intuition and assumptions leading the inclusion of a potential factors are required. The number of factors is arbitrarily chosen based on the sample on which the betas are estimated. Stochastic dynamical systems like financial markets are non-stationary and it's therefore highly plausible that factors numbers are functions of the states of the markets. To motivate our claim we recall the risk factors are constructed using 6 portfolios at the intersection of 3 book-to-price categories: H, high; M, medium; and L, low, with 2 size categories: B, big; and S, small. The resulting designated portfolios are : HS, MS, LS, HB, MB, and LB [72].

$$\xi_{SMB} = \frac{1}{3}\big((\xi_{HS} + \xi_{MS} + \xi_{LS}) - (\xi_{HB} + \xi_{MB} + \xi_{LB})\big) \tag{A4}$$

$$\xi_{HML} = \frac{1}{2}\big((\xi_{HB} + \xi_{HS}) - (\xi_{LB} + \xi_{LS})\big) \tag{A5}$$

In the SMB and HML case, the division of assets in these 6 portfolios induce length scales on financial markets which are ultimately arbitrarily defined. In fact, there are no reasons to believe that there are only small and big market capitalization groups, or only high, medium, and low book-to-price ratios. That being said, and as previously mentioned, factors models have proven useful and our goal here is to argue that if there exist length scales in such systems they must be determined from the bottom-up and not top-down as risk factors were initially formulated.

These symptoms motivate the elaboration of adaptive asset pricing models within a framework where the nature and number of factors is state dependent. The Giada-Marsili asset pricing model provides a beginning attempt in this endeavor.Let's consider that Eqn. (11) reformulated the following way:

$$\xi_i(d) = \sqrt{1 - \sum_{S}^{L} g_{s_i}^2 \delta_{s,s_i} \epsilon_i(d)} + \sum_{S}^{L} g_{s_i} \eta_{s_i}(d) \delta_{s,s_i} \tag{A6}$$

where we only consider the case for which one asset belongs to one unique cluster[26], L is the number of clusters, and equivalently in the original APT framework the η_{s_i} are the factors f_{s_i} and they represent the average return in cluster s_i, and the g_{s_i} are the factor loadings β_{s_i}. The main differences with typical APT models are the following:

1. The factors loadings g_{s_i} are fully determined by the correlation matrix C_{ij} and the cluster structure (The Maximum Likelihood spin configuration) of the data through c_s and n_s. Instead of linear regression they are determined using the algorithms discussed in Sections II D 2, and II F, and are constant over the estimation period.

2. The factors $\eta_{s_i}(d)$ are dynamical random variables unique for all assets in cluster s_i whereas in the APT model in Eqn. (A1) the factors f_j are random variables constant for all assets. The number of clusters, and factors L is also a random variable constant over the estimation period[27] but dependent on the market state.

3. This model violates EMH given that one direct conclusion from it is the independence of asset returns from each others. Asset $\imath$ and $\jmath$ returns should be uncorrelated which in practice isn't empirically true.

4. The factors $\eta_{s_i}(d)$ could be modeled as random variables down from the distribution of daily average returns in the cluster s_i over the estimated period.

The Giada-Marsili Model essentially tells us that returns are best explained by dynamical factors, constant loadings, and a number of factors which captures the dynamical market contraction and expansion (the fluctuation in the number of clusters). We further recall that Eqn. (A6) is standardized, and given σ_i, and $\bar{\xi}_i$ respectively the standard deviation and mean return of asset i we give a final Giada-Marsili APT-like model:

$$\xi_i(d) = \sigma_i \left(\sqrt{1 - \sum_{S}^{L} g_{s_i}^2 \delta_{s,s_i} \epsilon_i(d)} + \sum_{S}^{L} g_{s_i} \eta_{s_i}(d) \delta_{s,s_i} \right) + \bar{\xi}_i \tag{A7}$$

[26] this needs not be the case, see Sec VI. in [24]

[27] L is dynamical over a rolling window. The Giada-Marsili Asset Pricing model could also be called a "L-factor Model"

B. Super-paramagnetic Clustering under increasing Noisy Data.

One theme of this dissertation was focused on data preparation by means of noise cleaning techniques especially in the cases where time-series were involved. We want to see how SPC behaves under varying noise levels. Before doing so one needs to carefully define what is meant by noise. Data clustering is the problem of identifying dense lumps, and we accordingly qualify noisy data sets as those with low density clusters.

The Ansatz in [24, 56] offers a powerful stochastic processes model for clustering purposes. We make use of the following equation (equivalent to Eqn. (11)) as a way of generating correlated time-series:

$$\xi_i(d) = \frac{\sqrt{g_{s_i}}\eta_{s_i}(d) + \epsilon_i(d)}{\sqrt{1 + g_{s_i}}} \tag{B1}$$

The process is as follow:

11 Define values for number of cluster C, and size of clusters s and obtain $N = s * C$ the number of time-series in the data-set. Pick a time-series length D

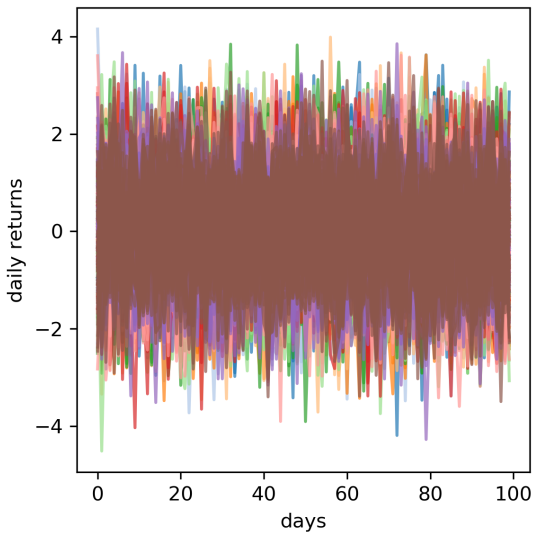
12 Create a list of array of spin-labels with the C labels

13 Create a CxD array $\eta \sim \mathcal{N}(0, 1)$, and another NxD array $\epsilon \sim \mathcal{N}(0, 1)$. η and ϵ respectively capture the daily cluster and stock random effects.

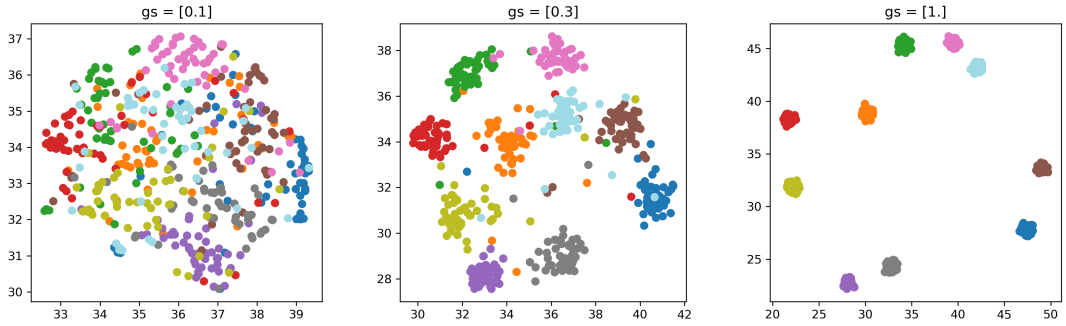
14 Pick a value for g_s per cluster: it is not needed that g_s be identical for every clusters. As a matter of fact, real noisy systems will various g_s values. However fixing g_s simplifies the process and increases interpretability.

15 Create a NxD array ξ and compute the daily returns using Eqn. (B1) by looping over the clusters, and the time-series within the clusters.

TABLE 1: Implementation of Noh Ansatz model of correlated time-series

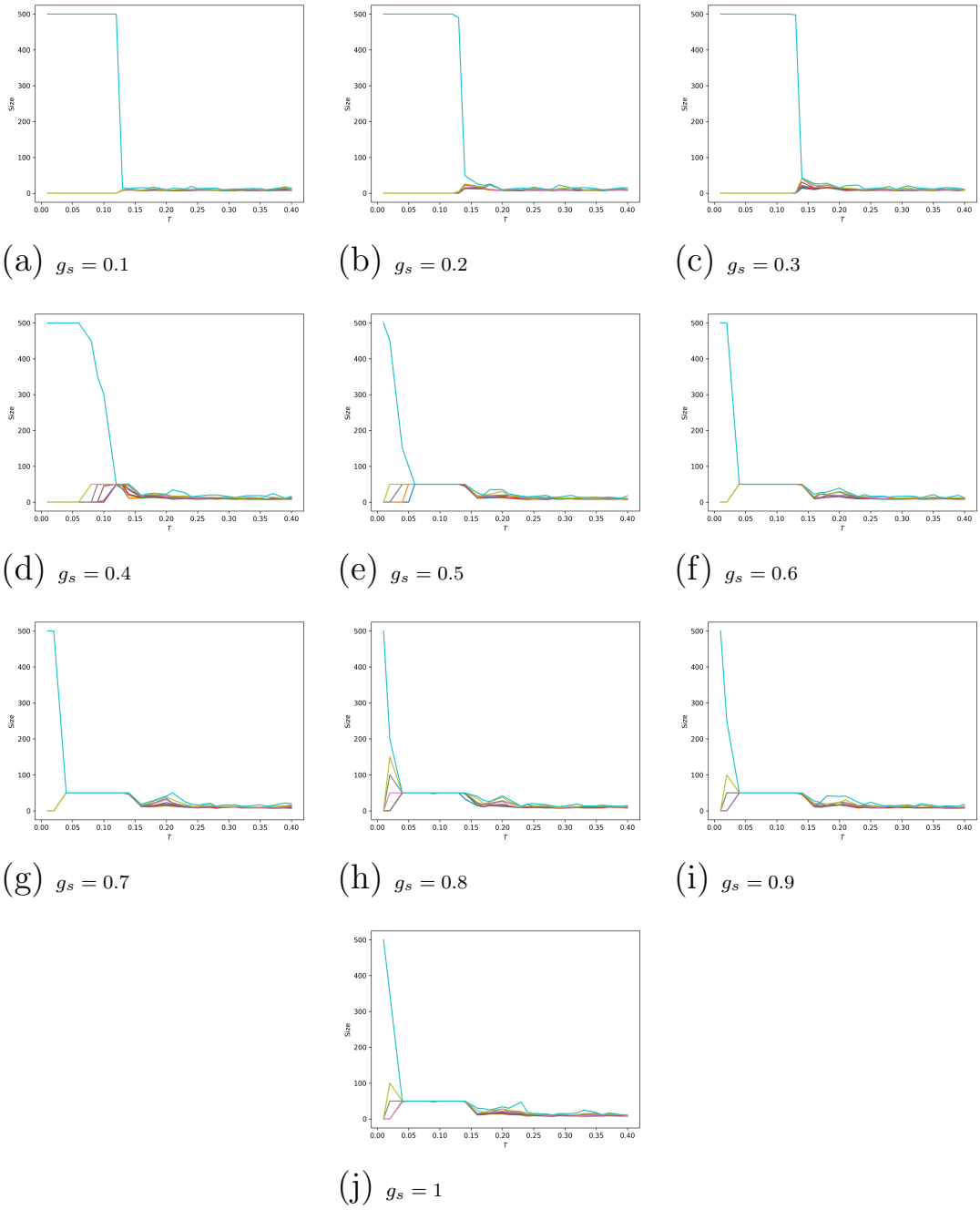


(a) 500 Normalized time-series daily returns



(b) $g_s = 0.1$ (c) $g_s = 0.3$ (d) $g_s = 1$

FIG. 25: in a) 500 time-series plotted: Normalized daily returns created using Table 1. in b), c), and d) the UMAP plot of the correlation matrices for the same data with increasing intra-cluster coupling parameter g_s from 0.1 to 1.



(a) $g_s = 0.1$ (b) $g_s = 0.2$ (c) $g_s = 0.3$

(d) $g_s = 0.4$ (e) $g_s = 0.5$ (f) $g_s = 0.6$

(g) $g_s = 0.7$ (h) $g_s = 0.8$ (i) $g_s = 0.9$

(j) $g_s = 1$

FIG. 26: 500 correlated time-series created using Table 1. for the same data with increasing intra-cluster coupling parameter g_s from 0.1 to 1. Very noisy data is characterized by an absence of, or small temperature region, for the Super-paramagnetic Phase. The SP-Phase increases in size as g_s goes up, and correlated timeseries separate from the giant cluster as early as $T = 0.05$.

In Fig. (25a) We plotted $N = 500$ simulated time-series of $D = 100$ with $C = 10$, and $s = 50$. Time-series visualizations lack interpretability especially given apparent

noise and chaos, and would necessitate a deeper statistical analysis. There exists however useful dimensionality reduction techniques that can capture the similarity (and dissimilarity) between observations, and UMAP (see [51]) is one such tool. UMAP allows us to plot the same data shown in Fig. (25a) but projected on a "learned" 2D manifold in Figures (25b), (25c), and (25d). The noise level is captured by g_s the intra-cluster coupling parameter: clusters with $g_s \to 0$ are spread out, and noisy whereas those with $g_s \to 1$ show increasingly high density, and strong correlation between its time-series. We train a UMAP model with an initial value for $g_s = 0.1$, and we iteratively create new data sets by increasing g_s from 0.1 to 1, but keeping the η, and ϵ matrices constant. Thus the random effects are kept the same but only the clusters couplings are varied and the noise level is the only tuned parameter. We show three examples for g_s values of 0.1, 0.3, and 1, and a gif of the entire range of g_s values is available at `https://github.com/tehraio/potts-model-clustering/blob/master/anigif.gif`.

Our objective is to first visualize how noise is manifested in data, and how it affects data clustering such as SPC. In Fig. (25b) It is easy to observe that at first ($g_s = 0.1$) the entire space is populated by data points spread out. The colors correspond to the cluster membership, and would signal to the observer the existence of clusters in the data however this is the kind of information which is usually missing, and which motivates unsupervised learning methods. As g_s is iteratively increased (see Figures (25c), and (25d)) we transform the newly created data onto the space learned using by UMAP, and the results are plotted. We immediately see that as g_s goes up the individual time-series increasingly get close to different centers, and we get a cleaner picture of an otherwise chaotic-looking data set.

We process these data-sets with SPC, and we show the results in Fig (26) where we plot the cluster sizes as a function of temperature clearly showing that for a low g_s (complete noise) there is virtually no SP-Phase, and the system transitions from order (ferromagnetic) to disorder (paramagnetic) whereas for values of $g_s \geq 0.4$ the SP-Phase is occur from $T \approx 0.05$ to $T \approx 0.15$. As the noise level goes down and $g_s \to 1$ the SP-Phase temperature range remains the same but the temperaure at which the clusters detach from the giant component are increasingly closer to T_c (first phase transition temperature), and their sizes remain stable until $T \approx 0.15$

C. Correlation Matrix for Time-Series with Missing Data

We now briefly explain how to compute a correlation matrix when dealing with time-series data which are plagued with missing values.

21	Create a NxN matrix of zeros
22	Pick a reference date range: $t_0 ... t_f$
23	Let ρ_{ij} be the Pearson Correlation Coefficient between r_i , and r_j for the time range where $r_i(t_0, ..., t_f) \cap r_j(t_0, ..., t_f)$ exists

TABLE 2: Computing Pearson Correlation Matrix with Missing Values.

D. Deriving the Potts Hamiltonian

We now give a short derivation of SPC (Sec. II D 1) objective function which is known as the Potts Hamiltonian (see Eqn. (3))

We first start from the likelihood of the Fortuin-Kasteleyn random cluster model (See Eqn. (2))

$$P(S) = \mathbf{Z^{-1}} \prod_{\langle i,j \rangle} \left[(1 - P_{ij}) + P_{ij} \delta_{s_i,s_j} \right] = \mathbf{Z^{-1}} e^{-H_S} \tag{D1}$$

We consider instead the log-likelihood of this distribution, and we drop the partition function

$$\ln(P(S)) \propto \sum_{\langle i,j \rangle} \ln \left(1 - P_{ij} + P_{ij} \delta_{s_i,s_j} \right) \tag{D2}$$

We then replace P_{ij} by its expression $P_{ij} = 1 - e^{-J_{ij}}$

$$\ln(P(S)) \propto \sum_{\langle i,j \rangle} \ln \left(1 - (1 - e^{-J_{ij}}) + (1 - e^{-J_{ij}}) \delta_{s_i,s_j} \right) \tag{D3}$$

$$\ln(P(S)) \propto \sum_{\langle i,j \rangle} \ln \left(e^{-J_{ij}} + \delta_{s_i,s_j} - e^{-J_{ij}} \delta_{s_i,s_j} \right) \tag{D4}$$

Let's collect the exponential terms

$$\ln(P(S)) \propto \sum_{\langle i,j \rangle} \ln \left(\delta_{s_i,s_j} + e^{-J_{ij}} (1 - \delta_{s_i,s_j}) \right) \tag{D5}$$

Let's now consider that we have two cases:

$$\delta_{s_i,s_j} = 1 \text{ with } \ln(P(S)) \propto \sum_{\langle i,j \rangle} \ln(1) = 0$$

$$\delta_{s_i,s_j} = 0 \text{ with } \ln(P(S)) \propto \sum_{\langle i,j \rangle} \ln(e^{-J_{ij}}) = \sum_{\langle i,j \rangle} -J_{ij}$$

which is equivalent to

$$\ln(P(S)) \propto \sum_{\langle i,j \rangle} -J_{ij}(\delta_{s_i,s_j} - 1) \tag{D6}$$

given that $P(S) \propto e^{-H_S}$ then $\ln(P(S)) \propto -H_S$ and it follows that

$$H_S = \sum_{\langle i,j \rangle} J_{ij}(\delta_{s_i,s_j} - 1) \tag{D7}$$

E. The Marsili-Giada coupling parameters

We present the derivation of the Giada-Marsili Likelihood and its parameters [28]The form of the price associated with the i-th stock can then be written as

$$X_i(t) = g_{s_i}\eta_{s_i} + \sqrt{1 - g_{s_i}^2}\epsilon_i \tag{E1}$$

where the cluster related influences are driven by η_{s_i} and the stock unique effects by ϵ_i, both can be treated as Gaussian random variables with unit variance and zero mean[29]. The relative contribution is controlled by the intra-cluster coupling parameter g_{s_i}. The Giada-Marsili Model encodes the idea that stocks that have something in common are in the same cluster this comes with the caveats that stock membership in clusters is mutually exclusive and that intra-cluster correlations are positive.

From Eqn. E1 we compute the covariance for the i-th and j-th stocks

$$\mathrm{E}[X_i(t)X_j(t)] = g_{s_i}^2 \mathrm{E}[\eta_{s_i}\eta_{s_j}] + (1 - g_{s_i}^2)\mathrm{E}[\epsilon_i\epsilon_j]. \tag{E4}$$

[28] See [21] for further reading.

[29] This form of the price model ensures that the self correlation of a stock is one and independent of the cluster coupling. This can be seen by computing the self correlation $E[X_i^2]$ and using that clusters and stock unique process are unit variance zero mean processes

$$\mathrm{E}[(g_{s_i}\eta_{s_i} + \sqrt{1 - g_{s_i}^2}\epsilon_i)^2] = g_{s_i}^2 + (1 - g_{s_i}^2) = 1. \tag{E2}$$

This is not a unique choice, another possible choice often used is

$$\mathrm{E}[(\frac{\sqrt{g_{s_i}}}{\sqrt{1 + g_{s_i}}}\eta_{s_i} + \frac{1}{\sqrt{1 + g_{s_i}}}\epsilon_i)^2] = \frac{1 + g_{s_i}}{1 + g_{s_i}} = 1. \tag{E3}$$

Using that the process for shared component, η_{si}, and the stock unique component, ϵ are unit variance zero mean noise

$$C_{ij} = g_{s_i}^2 \delta_{s_i s_j} + (1 - g_{s_i}^2)\delta_{ij}. \tag{E5}$$

The following cluster relations can be derived where n_s is then number of stock in the s-th cluster and c_s is the internal correlation of the s-th cluster given that clusters are mutually exclusive

$$n_s = \sum_{i=1}^{N} \delta_{s_i s}, \qquad c_s = \sum_{i,j=1}^{N} C_{ij}\delta_{s_i s}\delta_{s_j s}. \tag{E6}$$

It follows from Eqn E5 that to each s with $n_s \geq 1$ there corresponds a single eigenvalue $\lambda_{s,0} = g_s^2(n_s - 1) + 1$, and $n_s - 1$ eigenvalues $\lambda_{s,1} = 1 - g_s^2$. It can also be seen that from Eqn E5 for $s_i = s_j = s$ one find $C_{ij} \approx g_s^2$. Multiplying both sides of Eqn. E5 by $\delta_{s_i s}\delta_{s_j s}$ and summing of all i and j to find

$$\sum_{i,j} C_{ij}\delta_{s_i s}\delta_{s_j s} = \sum_{i,j} g_{s_i}^2 \delta_{s_i s_j}\delta_{s_i s}\delta_{s_j s} \ldots \tag{E7}$$

$$+ \sum_{i,j} (1 - g_{s_i}^2)\delta_{ij}\delta_{s_i s}\delta_{s_j s}, \tag{E8}$$

To sum out the delta functions over the clusters and stocks from

$$\sum_{i,j} C_{ij}\delta_{s_i s}\delta_{s_j s} = \sum_{i} (g_{s_i}^2 \delta_{s_i s} \sum_{j} \delta_{s_i s_j}\delta_{s_j s}) \ldots \tag{E9}$$

$$+ \sum_{i} ((1 - g_{s_i}^2)\delta_{s_i s} \sum_{j} \delta_{ij}\delta_{s_j s}), \tag{E10}$$

we use that $\sum_j \delta_{ij}\delta_{s_i s} = \delta_{s_i s}$, $\sum_j \delta_{s_i s_j}\delta_{s_j s} = n_s\delta_{s_i s}$ and $\sum_i \delta_{s_i s}^2 = \sum_i \delta_{s_i s}$ to find

$$\sum_{i,j} C_{ij}\delta_{s_i s}\delta_{s_j s} = g_s^2 n_s \sum_i \delta_{s_i s} + (1 - g_s^2) \sum_i \delta_{s_i s}. \tag{E11}$$

Upon using Eqn. E6 in the above

$$c_s = g_s^2 n_s^2 + (1 - g_s^2)n_s = g_s^2(n_s^2 - n_s) - n_s. \tag{E12}$$

This is re-arranged to find

$$g_s = \sqrt{\frac{c_s - n_s}{n_s^2 - n_s}} \tag{E13}$$

F. The Marsili-Giada likelihood function

We evaluate the probability of the data satisfying the model using that probabilities are multiplicative

$$P(X_1(1),\ldots,X_N(D)) = \prod_{d=1}^{D}\prod_{i=1}^{N} P(X_i(d)). \tag{F1}$$

The probability of being in a given state that satisfies the model is given as a delta function such that we sum over all N stocks and all D features (date-times) taking expectations $\langle\ldots\rangle_{\eta,\epsilon}$ over the random processes associated with the stock specific noise and the cluster specific noise

$$P = \prod_{d=1}^{D}\left\langle \prod_{i=1}^{N}\delta\left(X_i(d) - (g_{s_i}\eta_{s_i} + \sqrt{1-g_{s_i}^2}\epsilon_i)\right)\right\rangle_{\eta,\epsilon} \tag{F2}$$

This takes on the form

$$\begin{aligned} P &= \prod_{d=1}^{D}\prod_{i=1}^{N}\int d\epsilon_i d\eta_{s_i}\exp\left[-\frac{1}{2}\sum_{k}^{N}\epsilon_k\delta_{ki}\epsilon_i\right] && \text{(F3)}\\ &\quad\times\exp\left[-\frac{1}{2}\sum_{p,q}^{N}\eta_{s_p}\eta_{s_q}\delta_{s_p s_i}\delta_{s_q s_i}\right] && \text{(F4)}\\ &\quad\times\delta\left(X_i(d) - g_{s_i}\eta_{s_i} - \sqrt{1-g_{s_i}^2}\epsilon_i\right) && \text{(F5)} \end{aligned}$$

This is simplified to the following form, where the sum over i stocks is converted to sums of the clusters s and the n_s stocks in each cluster

$$\begin{aligned} P &= \prod_{s=1}^{S}\prod_{d=1}^{D}\int d\eta_s e^{-\frac{1}{2}\eta_s^2}\prod_{i\in s}^{n_s}\int d\epsilon_i\exp\left[-\frac{1}{2}\epsilon_i^2\right] && \text{(F6)}\\ &\quad\times\delta\left(X_i(d) - g_s\eta_s - \sqrt{1-g_s^2}\epsilon_i\right) && \text{(F7)} \end{aligned}$$

The gaussian integral over the delta function is evaluated relative to the ϵ_i's using that $\prod\int f(x)\delta(ax-x_0) = \prod\frac{1}{|a|}f(x_0/a)$ over the n_s delta functions:

$$\begin{aligned} P &= \prod_{s=1}^{S}\prod_{d=1}^{D}\int\frac{d\eta_s}{(1-g_s^2)^{\frac{n_s}{2}}}e^{-\frac{1}{2}\eta_s^2} && \text{(F8)}\\ &\quad\times\prod_{i\in s}^{n_s}\exp\left[-\frac{1}{2}\frac{(g_s\eta_s - X_i)^2}{1-g_s^2}\right] && \text{(F9)} \end{aligned}$$

Expanding out the integrand and using $\prod_i e_i^A = e^{\sum_i A_i}$

$$P = \prod_{s=1}^{S}\prod_{d=1}^{D}\int \frac{d\eta_s}{(1-g_s^2)^{\frac{n_s}{2}}} e^{-\frac{1}{2}\eta_s^2} \tag{F10}$$

$$\times \exp\left[-\frac{1}{2}\sum_{i\in s}^{n_s}\frac{(g_s^2\eta_s^2 - 2g_s\eta_s X_i + X_i^2)}{1-g_s^2}\right] \tag{F11}$$

Expanding out the sum terms and evaluating where possible

$$P = \prod_{s=1}^{S}\prod_{d=1}^{D}\int \frac{d\eta_s}{(1-g_s^2)^{\frac{n_s}{2}}} e^{-\frac{1}{2}\eta_s^2} e^{-\frac{1}{2}\frac{n_s g_s^2 \eta_s^2}{1-g_s^2}} \tag{F12}$$

$$\times e^{\frac{g_s\eta_s}{1-g_s^2}\sum_{i\in s}^{n_s} X_i} e^{-\frac{1}{2}\frac{1}{1-g_s^2}\sum_{i\in s}^{n_s} X_i^2} \tag{F13}$$

This can be further simplified to

$$P = \prod_{s=1}^{S}\prod_{d=1}^{D}\int \frac{d\eta_s}{(1-g_s^2)^{\frac{n_s}{2}}} e^{-\frac{1}{2}\frac{1-g_s^2+n_s g_s^2}{1-g_s^2}\eta_s^2} \tag{F14}$$

$$\times e^{\frac{g_s\eta_s}{1-g_s^2}\sum_{i\in s}^{n_s} X_i} e^{-\frac{1}{2}\frac{1}{1-g_s^2}\sum_{i\in s}^{n_s} X_i^2} \tag{F15}$$

We now evaluate the guassian integral using that $\int e^{-x^2}dx = \sqrt{\pi/2}$ and hence that $\int e^{-ax^2+bx}dx = \sqrt{\frac{\pi}{2a}}e^{\frac{b^2}{4a}}$

$$P = \prod_{s=1}^{S}\prod_{d=1}^{D}\frac{\sqrt{\pi}}{(1-g_s^2)^{\frac{n_s}{2}}}\frac{(1-g_s^2)^{\frac{1}{2}}}{(n_s g_s^2 + (1-g_s^2))^{\frac{1}{2}}} \tag{F16}$$

$$\times \exp\left[\frac{g_s^2}{2(n_s g_s^2+(1-g_s^2))(1-g_s^2)}(\sum_{i\in s}^{n_s} X_i)^2\right] \tag{F17}$$

$$\times \exp\left[-\frac{1}{2}\frac{1}{1-g_s^2}\sum_{i\in s}^{n_s} X_i^2\right] \tag{F18}$$

Evaluate the product of all D times where $D >> 1$

$$P = \prod_{s=1}^{S}\left[\frac{\sqrt{\pi}}{(1-g_s^2)^{\frac{n_s}{2}}}\frac{(1-g_s^2)^{\frac{1}{2}}}{(n_s g_s^2 + (1-g_s^2))^{\frac{1}{2}}}\right]^D \tag{F19}$$

$$\times \exp\left[\frac{g_s^2}{2(n_s g_s^2+(1-g_s^2))(1-g_s^2)}(\sum_d^D\sum_{i\in s}^{n_s} X_i)^2\right] \tag{F20}$$

$$\times \exp\left[-\frac{1}{2}\frac{1}{1-g_s^2}\sum_d^D\sum_{i\in s}^{n_s} X_i^2\right] \tag{F21}$$

Using that $C_{ij} = \frac{1}{D}\sum_d X_i X_j$

$$\sum_d^D(\sum_{i\in s} X_i)^2 = \sum_{i,j=1}^{N}(\sum_d^D X_i X_j)\delta_{s_i s}\delta_{s_j,s} = Dc_s, \tag{F22}$$

and that the variance of the process in the s-th cluster can be computed from the trace[30]

$$\sum_{i\in s}\sum_{d}^{D} X_i^2 = DC_{ii} = \sum_{i\in s}^{n_s} DC_{ii} = Dn_s \tag{F26}$$

Substituting Eqn. F22, and Eqn. F26 in Eqn. F18

$$P = \prod_{s=1}^{S}\left[\frac{\sqrt{\pi}}{(1-g_s^2)^{\frac{n_s}{2}}}\frac{(1-g_s^2)^{\frac{1}{2}}}{(n_s g_s^2+(1-g_s^2))^{\frac{1}{2}}}\right]^D \tag{F27}$$

$$\times \exp^{-\frac{D}{2}\frac{n_s}{1-g_s^2}} \exp^{+\frac{D}{2}\frac{c_s}{1-g_s^2}\frac{g_s^2}{n_s g_s^2+(1-g_s^2)}} \tag{F28}$$

We re-write this as

$$P = \prod_{s=1}^{S}\frac{\pi^{\frac{D}{2}}(n_s g_s^2+(1-g_s^2))^{\frac{-D}{2}}}{(1-g_s^2)^{\frac{D}{2}(n_s-1)}} \tag{F29}$$

$$\times \exp^{-\frac{D}{2}\frac{1}{1-g_s^2}\left(n_s-\frac{c_s g_s^2}{n_s g_s^2+(1-g_s^2)}\right)} \tag{F30}$$

Then using that $P \propto e^{-DH_c}$ we can find that $H_c \propto \ln(P)$ from Eqn F28 and using that $\ln\prod_i A_i = \sum_i \ln(A_i)$ to find the log-likelihood function [Need to use $D >> 1$ and look at expansion $(g_s - g_s^*)$.]

$$\ln(P) = -\frac{D}{2}\sum_{s=1}^{S}\left[\ln(n_s g_s^2+(1-g_s^2)) \right. \tag{F31}$$

$$+ \quad \left.(n_s-1)\ln(1-g_s^2)\right] \tag{F32}$$

$$+ \quad \frac{D}{2}\sum_{s=1}^{S}[\ln(\pi)] \tag{F33}$$

$$- \quad \frac{D}{2}\sum_{s=1}^{S}\frac{1}{1-g_s^2}\left[n_s - \frac{c_s g_s^2}{n_s g_s^2+(1-g_s^2)}\right] \tag{F34}$$

[30] The trace of the correlation matrix for each cluster s can be verified from the eigenvalues

$$\sum_{i}^{N} C_{ii} = \sum_{s}\lambda_s \tag{F23}$$

$$= (n_s-1)(1-g_s^2) + n_s g_s^2 + (1-g_s^2) \tag{F24}$$

$$= n_s \tag{F25}$$

Using Eqn E13 we can substitute for g_s E13 to find the log-likelihood entirely in terms of n_s and c_s using that $(1-g_s^2) = \frac{n_s^2-c_s}{n_s^2-n_s}$ and $\frac{c_s}{n_s} = n_s g_s^2 + (1-g_s^2)$.

$$H_c = \frac{1}{2}\sum_{s:n_s>0}\left[\log\frac{c_s}{n_s} + (n_s-1)\log\frac{n_s^2-c_s}{n_s^2-n_s}\right] \tag{F35}$$

$$+ \quad \frac{1}{2}\sum_{s:n_s>0}\left[\ln(\pi) + n_s\right] \tag{F36}$$

The last term is a constant given that $\sum_{s:n_s>0} n_s = N$ where N is the number of objects. This is fixed for a given system. Hence the likelihood function required is

$$H_c = \frac{1}{2}\sum_{s:n_s>0}\left[\log\frac{c_s}{n_s} + (n_s-1)\log\frac{n_s^2-c_s}{n_s^2-n_s}\right] \tag{F37}$$

upto a constant $\frac{1}{2}(S\ln(\pi) + N)$.

G. The Algorithms

The algorithms implemented in this dissertation have been coded in python and are available on a github repository at [74].

1. SPC Algorithms

We provide a pseudo-code for the SPC [7] algorithm introduced in Sec. II D 1. Given a distance matrix, and a neighborhood of size K, the algorithm uses the Swendsen-Wang [68] MCMC method to optimize the thermodynamic system.

TABLE 1

Algorithm 1 SPC (Sec. II D, and [7]), and function "`runz`" in "super-paramagnetic-clustering.py" in [74]

1: **for** $k = 0$ to $k = M$ **do**
2: Create edge configuration matrix "`link`"
3: Create Swendsen-Wang clusters
4: Compute, and store thermodynamic quantities i.e. m, c_{ij}
5: **end for**

Hoshen-Kopelman [3] is the mechanism behind clusters discovery in SPC. It allows for the graph to be traveled via its nodes and neighborhoods while clustering the system locally.

TABLE 2

Algorithm 2 Extended Hoshen-Kopelman (Table D, and [3]), and [68]), and function "`eHK`" in "super-paramagnetic-clustering.py" in [74]

set label counter to 0, Initialize `nodel` Nx1 array
Create `nodelp`, an empty array
for $i = 0$ to $i = N$ **do**
  if node i isn't linked at all **then**
    Set label counter to i's label
     Store i's label to `nodelp`
    Increase label counter by 1
  else
    Find i's linked neighbors, and store their `nodelp` labels
    if None are labeled **then**
      Set label counter to i's label
       Store i's label to `nodelp`
      Increase label counter by 1
    else
      store the labels of the linked neighbors
      Store root of the labels of the linked neighbors
      Set min the smallest root label
      Set the `nodel` of i to min
      In `nodelp` change linked neighbors root labels to min
    end if
  end if
    {Make `nodelp` sequential}
  for $y = 0$ to $y =$ `len(nodelp)` **do**
    $n = y$
    while The root of n is less than n **do**
      Set n to the root of n
    end while
     Set the root of y to n
  end for
    { Relabel the labels with their roots }
  for $i = 0$ to $i =$ `len(nodelp)` **do**
     Find labels in `nodel` == i, & update them with their root in `nodelp`
  end for
end for

2. f-SPC Algorithms

We discussed f-SPC in Sec. II D 2, and here we provide a pseudo-code for the implementation of the parallelized genetic algorithm which generates clustering candidates, evaluates their likelihood L_c [24] using Eqn. (13), selects the best candidates and discards the others.

TABLE 3

Algorithm 3 f-SPC PGA (Sec. II D 2, and [32]), and "fast-SP-clustering.py" in [74]

```
Produce an initial population of N individuals
for (In parallel ) All individuals do
    evaluate fitnesses
end for
for G number of generations do
    Create offsprings by copying the entire population
    Mutate the offsprings
    for (In parallel ) All offsprings do
        Evaluate fitnesses
    end for
    Recombine parents and offsprings
    Select the N individuals as next population
end for
```

TABLE 4

Algorithm 4 Pseudo-code for a CBMA implementation (Sec. II F), and "cbma.py" in [74]

INPUT: Correlation Matrix, OUTPUT: Tracker
1st pass
Produce an initial population of N individuals
for i in N-1 **do**
  for j in [i+1, N) **do**
    merge labels i and j, store ΔL_c in candidates
  end for
end for
2nd Pass
for N-1 iterations **do**
  Select in candidates the two merged (root, and leaf) labels with max ΔL_c
  Stop if $\Delta L_c \leq 0$
  Update S, and tracker
  for i in labels not root **do**
    merge labels root and i, store ΔL_c in candidates
  end for
end for

[1] Abramov, A., Kulvicius, T., Wörgötter, F., and Dellen, B. (2010). Facing the Multicore-challenge. In Keller, R., Kramer, D., and Weiss, J.-P., editors, *Facing the Multicore-challenge*, chapter Real-time Image Segmentation on a GPU, pages 131–142. Springer-Verlag, Berlin, Heidelberg.

[2] Ackley, D. H., Hinton, G. E., and Sejnowski, T. J. (1985). A learning algorithm for boltzmann machines. *Cognitive Science*, 9(1):147 – 169.

[3] Al-Futaisi, A. and Patzek, T. W. (2003). Extension of Hoshen-Kopelman algorithm to non-lattice environments. *Physica A: Statistical Mechanics and its Applications*, 321(3):665 – 678.

[4] Barabási, A.-L. and Albert, R. (1999). Emergence of Scaling in Random Networks. *Science*, 286(5439):509–512.

[5] Barnett, L., Lizier, J. T., Harré, M., Seth, A. K., and Bossomaier, T. (2013). Information Flow in a Kinetic Ising Model Peaks in the Disordered Phase. *Phys. Rev. Lett.*, 111:177203.

[6] Bianconi, G. and Barabási, A.-L. (2001). Bose-Einstein Condensation in Complex Networks. *Phys. Rev. Lett.*, 86:5632–5635.

[7] Blatt, M., Wiseman, S., and Domany, E. (1996). Superparamagnetic Clustering of Data. *Phys. Rev. Lett.*, 76:3251–3254.

[8] Blatt, M., Wiseman, S., and Domany, E. (1997). Data Clustering Using a Model Granular Magnet. *Neural Computation*, 9(8):1805–1842.

[9] Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008.

[10] Bonanno, G., Lillo, F., and Mantegna, R. (2001). High-frequency cross-correlation in a set of stocks. *Quantitative Finance*, 1(1):96–104.

[11] BRUSH, S. G. (1967). History of the Lenz-Ising Model. *Rev. Mod. Phys.*, 39:883–893.

[12] Cieslakiewicz, D. (2014). *Unsupervised asset cluster analysis implemented with parallel genetic algorithms on the Nvidia CUDA platform.* PhD thesis.

[13] Edwards, R. G. and Sokal, A. D. (1988). Generalization of the Fortuin-Kasteleyn-Swendsen-Wang representation and Monte Carlo algorithm. *Phys. Rev. D*, 38:2009–2012.

[14] Erdos, P. and Rényi, A. (1960). On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, 5:17–61.

[15] Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. (1996). A Density-based Algorithm for Discovering Clusters a Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, pages 226–231. AAAI Press.

[16] Fama, E. F. and French, K. R. (1993). Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics*, 33(1):3 – 56.

[17] Fama, E. F. and French, K. R. (2015). A five-factor asset pricing model. *Journal of Financial Economics*, 116(1):1 – 22.

[18] Farmer, J. D. and Geanakoplos, J. (2009). The virtues and vices of equilibrium and the future of financial economics. *Complexity*, 14(3):11–38.

[19] Fortuin, C. and Kasteleyn, P. (1972). On the random-cluster model: I. Introduction and relation to other models. *Physica*, 57(4):536 – 564.

[20] Foye, J. (2018). Testing alternative versions of the Fama–French five-factor model in the UK. *Risk Management*, 20(2):167–183.

[21] Gebbie, T. (2017). Derivation of the Giada-Marsili Likelihood function. *WITS working document*.

[22] Gebbie, T., Wilcox, D., and Mbambiso, B. (2010). Spin, stochastic factor models, and a GA. *Southern African Finance Association Conference, Cape Town*.

[23] Getz, G., Levine, E., Domany, E., and Zhang, M. (2000). Super-paramagnetic clustering of yeast gene expression profiles. *Physica A: Statistical Mechanics and its Applications*, 279(1):457 – 464.

[24] Giada, L. and Marsili, M. (2001). Data clustering and noise undressing of correlation matrices. *Phys. Rev. E*, 63:061101.

[25] Giada, L. and Marsili, M. (2002). Algorithms of maximum likelihood data clustering with applications. *Physica A: Statistical Mechanics and its Applications*, 315(3):650 – 664.

[26] Good, B. H., de Montjoye, Y.-A., and Clauset, A. (2010). Performance of modularity maximization in practical contexts. *Phys. Rev. E*, 81:046106.

[27] Griffin, J. M. (2015). Are the Fama and French Factors Global or Country Specific? *The Review of Financial Studies*, 15(3):783–803.

[28] Grimmett, G. R. (2006). *The random-cluster model*, volume 333. Springer Science & Business Media.

[29] Gu, S.-J., Sun, C.-P., and Lin, H.-Q. (2008). Universal role of correlation entropy in critical phenomena. *Journal of Physics A: Mathematical and Theoretical*, 41(2):025002.

[30] Hacine-Gharbi, A., Ravier, P., Harba, R., and Mohamadi, T. (2012). Low bias histogram-based estimation of mutual information for feature selection. *Pattern Recognition Letters*, 33(10):1302 – 1308.

[31] Harré, M. and Bossomaier, T. (2009). Phase-transition–like behaviour of information measures in financial markets. *EPL (Europhysics Letters)*, 87(1):18009.

[32] Hendricks, D., Gebbie, T., and Wilcox, D. (2016a). Detecting intraday financial market states using temporal clustering. *Quantitative Finance*, 16(11):1657–1678.

[33] Hendricks, D., Gebbie, T., and Wilcox, D. (2016b). High-speed detection of emergent market clustering via an unsupervised parallel genetic algorithm. *South African Journal of Science*, 112(1/2):9.

[34] Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800.

[35] Hinton, G. E. and Salakhutdinov, R. R. (2006). Reducing the Dimensionality of Data with Neural Networks. *Science*, 313(5786):504–507.

[36] Hoshen, J. and Kopelman, R. (1976). Percolation and cluster distribution. I. Cluster multiple labeling technique and critical concentration algorithm. *Phys. Rev. B*,

14:3438–3445.

[37] Hubert, L. and Arabie, P. (1985). Comparing partitions. *Journal of Classification*, 2(1):193–218.

[38] Jaynes, E. T. (2003). *Probability theory: The logic of science*, volume II. Cambridge university press.

[39] Kruskal, J. B. (1956). On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem. *Proceedings of the American Mathematical Society*, 7(1):48–50.

[40] Kullmann, L., Kertesz, J., and Mantegna, R. (2000). Identification of clusters of companies in stock indices via Potts super-paramagnetic transitions. *Physica A: Statistical Mechanics and its Applications*, 287(3):412 – 419.

[41] Li, J. (2011). *Potts model clustering for discovering patterns of epigenetic marks*. PhD thesis, Rutgers University-Graduate School-New Brunswick.

[42] Lloyd, S. (1982). Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137.

[43] Lo, A. W. (2004). The Adaptive Markets Hypothesis. *The Journal of Portfolio Management*, 30(5):15–29.

[44] Malkiel, B. G. (2003). The efficient market hypothesis and its critics. *Journal of economic perspectives*, 17(1):59–82.

[45] Marsili, M. (2002). Dissecting financial markets: sectors and states. *Quantitative Finance*, 2(4):297–302.

[46] Marti, G., Nielsen, F., Bińkowski, M., and Donnat, P. (2017). A review of two decades of correlations, hierarchies, networks and clustering in financial markets. *arXiv preprint arXiv:1703.00485*.

[47] Martin, C. H. and Mahoney, M. W. (2018). Implicit Self-Regularization in Deep Neural Networks: Evidence from Random Matrix Theory and Implications for Learning. *arXiv preprint arXiv:1810.01075*.

[48] Mastromatteo, I. and Marsili, M. (2011). On the criticality of inferred models. *Journal of Statistical Mechanics: Theory and Experiment*, 2011(10):P10012.

[49] Matsuda, H., Kudo, K., Nakamura, R., Yamakawa, O., and Murata, T. (1996). Mutual information of ising systems. *International Journal of Theoretical Physics*, 35(4):839–845.

[50] Mbambiso, B. (2014). *Dissecting the South African equity markets into sectors and states*. LAP LAMBERT Academic Publishing.

[51] McInnes, L., Healy, J., Saul, N., and Groibberger, L. (2018). UMAP: Uniform Manifold Approximation and Projection. *The Journal of Open Source Software*, 3(29):861.

[52] Mehta, P. and Schwab, D. J. (2014). An exact mapping between the variational renormalization group and deep learning. *arXiv preprint arXiv:1410.3831*.

[53] Merton, R. C. et al. (1973). An intertemporal capital asset pricing model. *Econometrica*, 41(5):867–887.

[54] Murua, A. and Wicker, N. (2014). The Conditional-Potts Clustering Model. *Journal of Computational and Graphical Statistics*, 23(3):717–739.

[55] Musmeci, N., Aste, T., and Di Matteo, T. (2015). Risk diversification: a study of persistence with a filtered correlation-network approach. *Journal of Network Theory*

in Finance, 1(1):77–98.

[56] Noh, J. D. (2000). Model for correlations in stock markets. *Phys. Rev. E*, 61:5981–5982.

[57] Olshen, R. A. and Rajaratnam, B. (2010). Successive normalization of rectangular arrays. *Annals of Statistics*, 38(3):1638.

[58] Onsager, L. (1944). Crystal Statistics. I. A Two-Dimensional Model with an Order-Disorder Transition. *Phys. Rev.*, 65:117–149.

[59] Osogami, T. (2017). Boltzmann machines and energy-based models. *arXiv preprint arXiv:1708.06008*.

[60] Ott, T., Kern, A., Schuffenhauer, A., Popov, M., Acklin, P., Jacoby, E., and Stoop, R. (2004). Sequential Superparamagnetic Clustering for Unbiased Classification of High-Dimensional Chemical Data. *Journal of Chemical Information and Computer Sciences*, 44(4):1358–1364. PMID: 15272844.

[61] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python . *Journal of Machine Learning Research*, 12:2825–2830.

[62] Quiroga, R. Q., Nadasdy, Z., and Ben-Shaul, Y. (2004). Unsupervised spike detection and sorting with wavelets and Superparamagnetic clustering. *Neural Computation*, 16(8):1661–1687.

[63] Roll, R. and Ross, S. A. (1980). An empirical investigation of the arbitrage pricing theory. *The Journal of Finance*, 35(5):1073–1103.

[64] Rumelhart, D. E. and McClelland, J. L. (1987). Information Processing in Dynamical Systems: Foundations of Harmony Theory. Technical report, University of Colorado at Boulder.

[65] Rumelhart, D. E., McClelland, J. L., and PDP Research Group, C., editors (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations*. MIT Press, Cambridge, MA, USA.

[66] Sokal, A. D. (2005). The multivariate Tutte polynomial (alias Potts model) for graphs and matroids. pages 173–226.

[67] Stanberry, L., Murua, A., and Cordes, D. (2007). Functional connectivity mapping using the ferromagnetic Potts spin model. *Human Brain Mapping*, 29(4):422–440.

[68] Swendsen, R. H. and Wang, J.-S. (1987). Nonuniversal critical dynamics in Monte Carlo simulations. *Phys. Rev. Lett.*, 58:86–88.

[69] West, D. B. et al. (2001). *Introduction to graph theory*, volume 2. Prentice hall Upper Saddle River.

[70] Wilcox, D. and Gebbie, T. (2007). An analysis of cross-correlations in an emerging market. *Physica A: Statistical Mechanics and its Applications*, 375(2):584 – 598.

[71] Wilcox, D. and Gebbie, T. (2014). Hierarchical Causality in Financial Economics. *https://ssrn.com/abstract=2544327*.

[72] Wilcox, D. L. and Gebbie, T. J. (2015). On pricing kernels, information and risk. *Investment Analysts Journal*, 44(1):1–19.

[73] Wu, F. Y. (1982). The Potts model. *Rev. Mod. Phys.*, 54:235–268.

[74] Yelibi, L. (2017). Potts Model Clustering. `https://github.com/tehraio/potts-model-clustering`.

[75] Yelibi, L. and Gebbie, T. (2018). Fast Super-Paramagnetic Clustering. *arXiv:1810.02529*.