Statistics and Its Interface Volume 6 (2013) 53–64

Collusion set detection using a quasi hidden Markov model[*]

Zhengxiao Wu[†] and Xiaoyu Wu

In stock market, a collusion set is defined as a group of individuals or organizations who act cooperatively with an intention of manipulating security price. Collusion-based malpractices impose large costs on the economy, but few techniques have yet been developed for collusion set detection.

In this article, we propose a quasi hidden Markov model (QHMM) approach. In particular, we consider the transactions as a marked point process with hidden states, and we calculate the class conditional probabilities to identify the malicious transactions. The detection algorithms associated with the model are recursive, hence suitable for online monitoring and detection. The QHMM approach has several advantages over the existent methods. For example, it incorporates the transaction times into the model naturally, and the model parameters can be estimated from the data systematically. We illustrate the models with examples and the QHMM performs well in our numerical experiments.

AMS 2000 subject classifications: Primary 62M99, 60K35; secondary 62P99.

Keywords and phrases: Collusion set, Fraud detection, Hidden Markov model, Quasi hidden Markov model.

1. INTRODUCTION

In financial industry, there are individuals and organizations who exploit dishonest or deceitful means to realize personal gains. Examples include money laundry, embezzlement, security fraud, to name a few (see [13] and the references therein). Different financial crimes have different criminal strategies and they may vary from time to time. In some cases, individuals or organizations use *modus operandi* of collusion to achieve illegal financial gains. In this study, we focus on one particular type of collusion-based financial crimes – circular trading.

In circular trading, unscrupulous colluders reach mutual agreements to manipulate stock with an intention of influencing the price. They circulate large amount of shares within a short period of time to create a false impression of heavy demand. Other investors are often tricked by the fraudulent information and rush for the stock. The price rises dramatically as a result. When the price reaches a predetermined level, the collusion set sells their shares and realizes substantial profits. As the high price does not reflect the true value of the stock, and due to the large-scale selling, the price typically crashes down to its original level or even lower, resulting in great loss to the other investors.

Circular trading damages the fairness and integrity of the market, and imposes large costs to the economy. To facilitate fair practices, financial regulators and authorities have enforced laws and guidelines to regulate participants in stock market activities. However, it is believed that most collusion sets remain undetected due to the large size of the trading database, number of participants and complexity of trading behaviors. Any individual transaction in circular trading is often superficially legal. Malpractices can only be detected when the relevant transactions are appropriately grouped together, but the evidence of such activity is often hidden deeply inside the databases and can be extremely difficult to obtain. On the other hand, victims of the malpractice are rarely aware of them being victimized [2]. Without obvious victims, regulators face difficulties in obtaining information during the investigation.

There have been only a few statistical studies on collusion set detection. In 2000, Palshikar and Bahulkar [9] proposed a pattern recognition based approach. They applied fuzzy logic tools to specify and identify the trading patterns. As trading strategies vary from time to time and trader to trader, intensive market investigations and considerable domain knowledge are required to fine-tune their model constantly, which imposes great limitations on the method.

Palshikar and Apte (2006) suggested another approach – using a graph-clustering algorithm to detect the collusion set [8]. The intuition is that malicious transactions have certain characteristics. For example, the malicious transactions typically have large trading volumes, and their buyers and sellers are from the same group of participants [1]. Palshikar and Apte proposed to pre-process the data by summarizing the transactions in a specific time window. The cumulative trading volume between each pair of traders is recorded. Palshikar and Apte then considered the graph with the traders as the vertices. The dissimilarity between the vertices is defined so that a large cumulative trading volume leads to a small dissimilarity between each pair of traders. Clustering algorithms are applied and the cluster found is identified as

[*]This work is supported by MOE grant R-155-000-105-112.
[†]Corresponding author.

Table 1. Partial malicious transactions on DSQ security

Date	Trade time	Buying Client	Selling Client	Volume	Price (Rs)	Value (Rs)
15-Feb-00	11:14:07	Hulda	Hulda	50000	1529	96.45 mil
15-Feb-00	11:14:27	Hulda	Hulda	50000	1529	96.45 mil
08-Mar-00	13:46:08	Hulda	DSQ H	25000	2492	62.3 mil
08-Mar-00	13:46:52	Hulda	DSQ H	25000	2485	62.125 mil
08-Mar-00	13:47:22	Hulda	DSQ H	10000	2486	24.86 mil
08-Mar-00	13:47:40	Hulda	DSQ H	25000	2487	62.175 mil
08-Mar-00	13:48:08	Hulda	DSQ H	15000	2493	37.395 mil

the collusion set. Palshikar and Apte showed that this procedure is effective when the collusion set is present in the studied time window and the input parameters of the clustering algorithms are well-chosen. However, the procedure outputs a collusion set candidate even if there is no collusion set in the investigated transactions. The procedure does not give a probabilistic statement on the uncertainty of the collusion set found. The method is designed to be retrospective, not for online monitoring and real time detection. Furthermore, the data are not fully utilized in the procedure as the input of the clustering algorithm is just a data summary. In particular, the timestamp of the transactions are ignored in [8].

However, since the traders in a collusion set tend to trade intensively in a short period of time to maximize the impact on the security price, it is conceivable that the trading times also contain crucial information. From October 1999 to March 2001, three entities in India formed a collusion set and circulated large amount shares of a security called DSQ Industries. Table 1 is an excerpt from the court orders [11, 12] released by the Security and Exchange Board of India (SEBI). One can see from Table 1 that several malicious trades are temporally clustered. This temporal information is lost when the transaction times are discarded in the detection procedure.

In this article, we propose to model the raw data (i.e., the transactions) as a marked point process [4], hence the information can be extracted from trading time. We do not observe if a transaction is malicious or not, thus the marked point process we considered have certain hidden states. Our initial attempt of using the well-known hidden Markov model (HMM) [10] or the nonhomogeneous hidden Markov models (NHMM) [5] did not succeed because the assumptions the HMM and NHMM are quite restrictive for this problem. Hence we apply the more general quasi hidden Markov model (QHMM) framework ([17] and [18]) and build a QHMM for the collusion set detection problem. The detection algorithms associated with the model are recursive; hence they are suitable for online monitoring and detection. The parameters of the model can be estimated from the data by the maximum likelihood principle or the moment methods; whereas many graph-clustering algorithms apply ad hoc methods to determine the input parameters. The QHMM approach provides a soft classifier [7] in the sense that it explicitly estimates the class conditional probabilities and then performs classification based on estimated probabilities. On the other hand, the graph-clustering algorithm proposed in [8] is not satisfactory for collusion set detection because it does not make an inference on the important question regarding whether a collusion set is present or not.

The rest of the paper is organized as follows: Section 2 introduces the QHMM framework and proposes a QHMM for collusion set detection; we carry out numerical studies in Section 3 with simulated data and real data; and Section 4 summarizes the conclusion and discusses extensions. Appendix A describes the forward-backward algorithm and the Viterbi algorithms associated with a QHMM; Appendix B defines a measure of dissimilarity used in the numerical examples.

2. QHMM APPROACH

QHMM is a broad class of models which subsume HMM and NHMM. In a QHMM, the observation O_t at time t can depend on not only the current hidden state q_t, but also the previous observations $O_1, \ldots, O_{t-1}$. Another attribute of the QHMM is that the hidden state space could be time-varying. The forward-backward algorithm and the Viterbi algorithm associated with a QHMM are derived in [17]. They provide a flexible framework for modeling partially observed processes. In this section, we briefly review the QHMM, propose a QHMM for collusion set detection, and then finally illustrate the QHMM approach with two examples.

2.1 Definition of QHMM

We let O_t and q_t denote the observation and the hidden state at time t respectively. The hidden state q_t takes values in a possibly time-varying finite discrete space $\mathbf{S_t} = \{S_1, S_2, \ldots, S_{N_t}\}$. The observations $\{O_t\}$ and the hidden states $\{q_t\}$ are said to form a QHMM if the conditional probability distributions satisfy $P(O_{t+1}, q_{t+1}|O_{1:t}, q_{1:t}) = P(O_{t+1}, q_{t+1}|O_{1:t}, q_t)$, where $O_{1:t}$ denotes $(O_1, O_2, \ldots, O_t)$ and $q_{1:t}$ denotes $(q_1, q_2, \ldots, q_t)$. Note that the assumptions of the conventional HMM and NHMM imply $P(O_{t+1}, q_{t+1}|O_{1:t}, q_{1:t}) = P(O_{t+1}, q_{t+1}|q_t)$, hence HMM and NHMM are special cases of the QHMMs.

Several algorithms facilitate the statistical inferences on a QHMM. The forward algorithm computes the likelihood function $P(O_{1:T})$ of the model. Combining with a backward procedure, it leads to the forward-backward algorithm, which calculates the probability of being in state S_i at time t given the observation sequence, i.e., $P(q_t = S_i|O_{1:T})$. The Viterbi algorithm finds the most likely hidden state sequence. We include these algorithms in Appendix A for completeness.

In a conventional HMM, the parameters of the model typically are estimated by the maximum likelihood principle via the EM algorithm [3]. However, in general the EM algorithm is not applicable to a QHMM. Fortunately, since the forward algorithm efficiently computes the likelihood function of a QHMM, most standard optimization procedures are applicable to find the Maximum Likelihood Estimators (MLE) of the parameters. For instance, we use the Nelder-Mead simplex method [6] in the numerical studies.

2.2 A QHMM for collusion set detection

We model the normal transactions and malicious transactions as two independent marked point processes, hence the observations are a mixture of these two processes. Let γ be the intensity of the normal transactions, thus the waiting time between two consecutive normal trades follows an exponential distribution with mean $1/\gamma$. On the other hand, motivated by the malicious transaction pattern in Table 1 (we can see that there are two malicious trade clusters in Table 1, one on 15-Feb-00 and the other one on 08-Mar-00), we allow the malicious transactions to have two possible intensities. They are with intensity ϵ when no malicious trade cluster is active and the intensity increases to $\epsilon + \lambda$ when a malicious trade cluster is active. The first malicious trade in a cluster always activates the malicious trade cluster. We let p be the probability of a malicious transaction deactivating the cluster, hence eventually a malicious trade cluster in our model dies off. For simplicity we assume there is at most one active malicious trade cluster at a time. From now on, the statements "a collusion set is active" and "a malicious trade cluster is active" are used interchangeably. Circular trading often occurs in thinly traded securities because they are easy to manipulate. Thus γ is typically small, whereas λ is typically large because the collusion set trades frequently when it is active.

In practice, we do not observe whether a trade is malicious or not, or whether there is an active collusion set. Hence we need to make inferences on the hidden states based on the observed data at hand. The QHMM framework naturally comes in.

The O_t and q_t are defined as follows. Suppose we have a data set containing n transactions. Transaction t ($1 \leq t \leq n$) has the observation $O_t = (\tau_t, R_t)$, which represents the transaction time and other trading information observed respectively. We can consider R_t as a vector containing "spatial" information such as trading volume, price, buyer information, seller information, etc. The hidden state is defined to be $q_t = (A_t, C_t, J_t)$. Both A_t and C_t are indicator variables: $A_t = 1$ if there is a collusion set active at time t, and $A_t = 0$ if otherwise; $C_t = 1$ if transaction t is malicious and $C_t = 0$ if otherwise.

Table 2. The hidden states of ten trades

t	C_t	A_t	J_t
1	0	0	0
2	1	1	2
3	0	1	2
4	1	0	2
5	0	0	2
6	0	0	2
7	1	1	7
8	0	1	7
9	1	1	7
10	0	1	7

The definition of J_t is more involved. We let J_t be the index of the first malicious transaction from the most recent malicious trade cluster, up to transaction t; and $J_t = 0$ if there is no malicious trade up to transaction t. Hence J_t can take values in $\{0, 1, 2, \ldots, t\}$.

To illustrate the meaning of (A_t, C_t, J_t), suppose a data set contains ten trades and four of them, trades $2, 4, 7, 9$ are malicious; and there are two malicious trade clusters, where 2 and 4 form the first malicious trade cluster, and 7 and 9 form the second malicious trade cluster; the second cluster is still active. Table 2 gives their hidden states. The hidden state C_t needs no explanation. $A_1 = 0$ because there is no active collusion set; $A_2 = A_3 = 1$ because malicious trade 2 activates the collusion set and normal trade 3 does not change the status of A; and $A_4 = 0$ because malicious trade 4 deactivates the cluster. Similarly $A_7 = A_8 = A_9 = A_{10} = 1$ because malicious trade 7 activates the second cluster and no trade deactivates it, so the second cluster is still active. $J_1 = 0$ since there is no malicious trade up to 1; $J_2 = 2$ because the first malicious trade in the most recent malicious cluster up to $t = 2$ is trade 2 itself. Then it is easy to see that $J_3 = J_4 = J_5 = J_6 = 2$; and $J_7 = 7$ since the first malicious trade in the most recent malicious cluster up to $t = 7$ is trade 7 itself.

With a given $J_t \neq t$, one can compare the current trading information R_t with R_{J_t}. The intuition is that if R_t is similar to R_{J_t}, (i.e., the dissimilarity between R_t and R_{J_t} is small, see Appendix B for a definition of dissimilarity), then it is likely that the current trade is malicious. On the other hand, if the dissimilarity between R_t and R_{J_t} is large, R_t is not likely to be malicious. The role of J_t will become clearer in the examples.

To fully determine the QHMM, it suffices to specify the conditional distribution of $P(O_{t+1}, q_{t+1}|O_{1:t}, q_t)$, which satisfies

$$\begin{aligned} &P(O_{t+1}, q_{t+1}|O_{1:t}, q_t) \\ &\quad = P(O_{t+1}|O_{1:t}, q_{t+1}, q_t)P(q_{t+1}|O_{1:t}, q_t). \end{aligned}$$

For simplicity we assume that

$$P(O_{t+1}|O_{1:t}, q_{t+1}, q_t) = P(O_{t+1}|O_{1:t}, q_{t+1})$$

and

$$P(q_{t+1}|O_{1:t}, q_t) = P(q_{t+1}|q_t).$$

These conditional distributions are discussed below in two cases and each case has several scenarios.

1. $A_t = 0$, i.e., no collusion set is active at time τ_t. In this case, the next trade $t+1$ is either a normal trade (scenario (a)), or a malicious trade that initiates a malicious trade cluster (scenario (b)). As the malicious trade cluster is initiated with intensity ϵ and the normal trade occurs with intensity γ, it is not hard to see that the next trade $t+1$ is a normal trade with probability $\frac{\gamma}{\epsilon+\gamma}$, and is malicious with probability $\frac{\epsilon}{\epsilon+\gamma}$. The waiting time $\tau_{t+1} - \tau_t$ follows an exponential distribution with parameter $\epsilon + \gamma$. Thus we can model the transition probabilities in these two scenarios:
   (a) Transaction $t+1$ is normal, hence there is still no active collusion set at time τ_{t+1}. And J_{t+1} takes the same value as J_t, thus in this scenario, $A_{t+1} = 0$, $C_{t+1} = 0$ and $J_{t+1} = J_t$. Therefore, $P(q_{t+1}|q_t) = P(A_{t+1} = 0, C_{t+1} = 0, J_{t+1} = J_t|q_t) = \frac{\gamma}{\epsilon+\gamma}$, and in this scenario we define $P(O_{t+1}|O_{1:t}, q_{t+1}) = P(\tau_{t+1}, R_{t+1}|O_{1:t}, A_{t+1} = 0, C_{t+1} = 0, J_{t+1} = J_t) = (\epsilon + \gamma) \exp[-(\tau_{t+1} - \tau_t)(\epsilon + \gamma)] f_1(R_{t+1})$, where $f_1(\cdot)$ is the "spatial" distribution for a normal trade.
   (b) Transaction $t+1$ is a malicious transaction that starts an active malicious trade cluster, i.e. $A_{t+1} = 1$, $C_{t+1} = 1$ and $J_{t+1} = t+1$. Hence $P(q_{t+1}|q_t) = P(A_{t+1} = 1, C_{t+1} = 1, J_{t+1} = t+1|q_t) = \frac{\epsilon}{\epsilon+\gamma}$, and we define $P(O_{t+1}|O_{1:t}, q_{t+1}) = P(\tau_{t+1}, R_{t+1}|O_{1:t}, A_{t+1} = 1, C_{t+1} = 1, J_{t+1} = t+1) = (\epsilon + \gamma) \exp[-(\tau_{t+1} - \tau_t)(\epsilon + \gamma)] f_2(R_{t+1})$, where $f_2(\cdot)$ is the "spatial" distribution for a first malicious trade.
2. $A_t = 1$, i.e. there is one malicious trade cluster active when transaction t is executed. The presence of an active collusion set introduces an extra intensity λ, hence the waiting time $\tau_{t+1} - \tau_t$ follows an exponential distribution with parameter $\lambda + \epsilon + \gamma$. Similarly, the next trade $t+1$ is a normal trade with probability $\frac{\gamma}{\epsilon+\gamma+\lambda}$, and it is malicious with probability $\frac{\epsilon+\lambda}{\epsilon+\gamma+\lambda}$. There are three scenarios in this case.
   (a) Transaction $t+1$ is normal, then it must be a normal trade settled between malicious trades because the collusion set is still active at time τ_{t+1}. One can see that in this scenario, $A_{t+1} = 1$, $C_{t+1} = 0$ and $J_{t+1} = J_t$. Hence, $P(q_{t+1}|q_t) = P(A_{t+1} = 1, C_{t+1} = 0, J_{t+1} = J_t|q_t) = \frac{\gamma}{\epsilon+\gamma+\lambda}$, and $P(O_{t+1}|O_{1:t}, q_{t+1}) = P(\tau_{t+1}, R_{t+1}|A_{t+1} = 1, C_{t+1} = 0, J_{t+1} = J_t) = (\epsilon + \gamma + \lambda) \exp[-(\tau_{t+1} - \tau_t)(\epsilon + \gamma + \lambda)] f_1(R_{t+1})$, as in scenario (a) in the first case, $f_1(\cdot)$ is the "spatial" distribution for a normal trade.
   (b) Transaction $t+1$ is malicious, and it does not deactivate the collusion set (this occurs with probability $1 - p$). In this scenario, $A_{t+1} = 1$, $C_{t+1} = 1$ and $J_{t+1} = J_t$. Hence $P(q_{t+1}|q_t) = P(A_{t+1} = 1, C_{t+1} = 1, J_{t+1} = J_t|q_t) = \frac{(1-p)(\epsilon+\lambda)}{\epsilon+\gamma+\lambda}$, and we define $P(O_{t+1}|O_{1:t}, q_{t+1}) = P(\tau_{t+1}, R_{t+1}|A_{t+1} = 1, C_{t+1} = 1, J_{t+1} = J_t) = (\epsilon + \gamma + \lambda) \exp[-(\tau_{t+1} - \tau_t)(\epsilon + \gamma + \lambda)] f_3(R_{t+1}|R_{1:t}, J_t)$, where $f_3(R_{t+1}|R_{1:t}, J_t)$ is the conditional distribution of a malicious trade given $R_{1:t}$ and J_t. In our examples, $f_3(R_{t+1}|R_{1:t}, J_t)$ takes a simple form of $f_3(R_{t+1}|R_{J_t})$. In particular, it depends on a dissimilarity measure between R_{t+1} and R_{J_t}.
   (c) Transaction $t+1$ is malicious, but it deactivates its collusion set (this occurs with probability p). In this scenario, $A_{t+1} = 0$, $C_{t+1} = 1$ and $J_{t+1} = J_t$. Hence $P(q_{t+1}|q_t) = P(A_{t+1} = 0, C_{t+1} = 1, J_{t+1} = J_t|q_t) = \frac{p(\epsilon+\lambda)}{\epsilon+\gamma+\lambda}$ and $P(O_{t+1}|O_{1:t}, q_{t+1}) = P(\tau_{t+1}, R_{t+1}|A_{t+1} = 0, C_{t+1} = 1, J_{t+1} = J_t) = (\epsilon + \gamma + \lambda) \exp[-(\tau_{t+1} - \tau_t)(\epsilon + \gamma + \lambda)] f_3(R_{t+1}|R_{1:t}, J_t)$

Therefore, the above discussions give all the possible conditional probability transactions $P(O_{t+1}, q_{t+1}|O_{1:t}, q_t)$. Hence the QHMM is fully determined if we specify $f_1(\cdot)$, $f_2(\cdot)$ and $f_3(\cdot|\cdot)$.

2.3 Examples

We illustrate the QHMM proposed in 2.2 with two examples.

2.3.1 Example 1

This toy example is to illustrate the motivation of our QHMM approach. We imagine that the trading information R_t is literately spatial. In particular, let $R_t = (x_t, y_t)$ be a point in a region, say, a 10 by 10 square. Hence x_t and y_t can be considered as the "longtitude" and the "latitude". Recall that $f_1(\cdot)$ is the "spatial" distribution for a normal trade, $f_2(\cdot)$ is the "spatial" distribution for a first malicious trade and $f_3(R_{t+1}|R_{1:t}, J_t)$ is the conditional distribution of a malicious trade given $R_{1:t}$ and J_t. We set both $f_1(\cdot)$ and $f_2(\cdot)$ to be the uniform distribution in this square, and let $f_3(R_{t+1}|R_{1:t}, J_t) = f_3(R_{t+1}|R_{J_t}) = 1/(2\pi d) \exp\{-\frac{(x_{t+1}-x_{J_t})^2+(y_{t+1}-y_{J_t})^2}{2d}\}$.

Hence according to the QHMM described in 2.2, the normal trades form a homogeneous Poisson process in this square with intensity γ. On the other hand, the collusion set is activated by a malicious trade (this first ma-

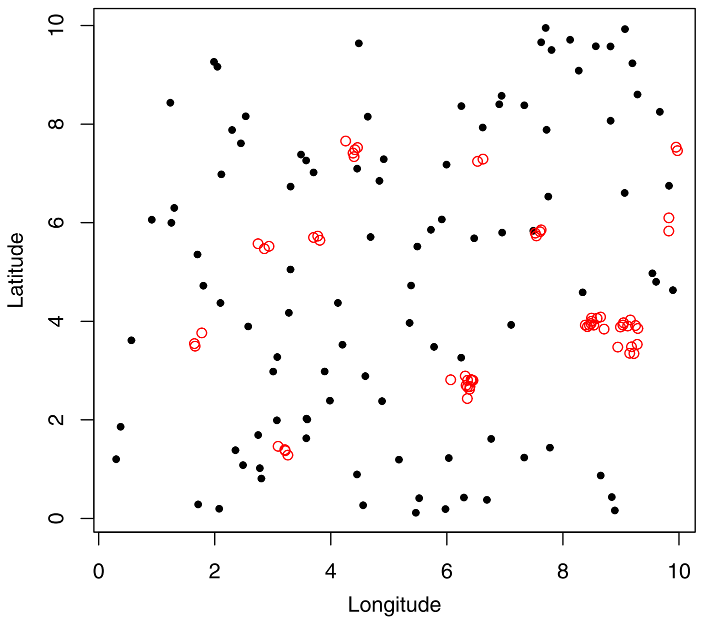

Figure 1. The "trades" in Example 1. The dots are the normal trades and the circles are malicious trades.

licious trade acts like an ancestor) and the ancestor intensively (with intensity $\epsilon + \lambda$) generates malicious trades (offspring) in its neighborhood with a bivariate Gaussian distribution until it is killed (The ancestor's death is triggered by giving birth to its last child). A new ancestor will appear somewhere according to the location distribution $f_2(\cdot)$ after an exponential waiting time with parameter ϵ. Figure 1 depicts a possible realization. The dots are the normal trades and the circles are malicious trades. A natural question is whether the QHMM can identify the malicious trades in this setting. The simulation studies on this example will be carried out in section 3.1.

2.3.2 Example 2

We suggest to model $f_1(\cdot)$, $f_2(\cdot)$ and $f_3(R_{t+1}|R_{J_t})$ for real transaction data as follows. Based on our empirical analysis (see also [1]), the large trading volume is an important characteristic of malicious trades. Hence we let $f_1(R_t) = f_1(v_t)$ and $f_2(R_t) = f_2(v_t)$ be two different Log-normal distributions, where v_t is the trading volume recorded in R_t. The idea is that when the trading volume is large, $f_1(v_t)$ should be small and $f_2(v_t)$ should be large. This setup characterizes the intuition that when a trade has a large trading volume, it is more likely to be a malicious trade.

We set $f_3(R_{t+1}|R_{J_t}) = 1/(2\pi d)\exp\{-\frac{D(R_t,R_{J_t})^2}{2d}\}$, where $D(\cdot,\cdot)$ is a dissimilarity measure between transaction records. There is a great flexibility in defining such kind of dissimilarity measures. Our definition involves the transaction counterparty and the cumulative trading volume. The details can be found in Appendix B.

3. NUMERICAL EXPERIMENTS

3.1 Example 1

We carry out simulation study on example 1 which is introduced in Section 2.3.1, to illustrate how statistical inferences are made based on the QHMM framework.

3.1.1 Data generation

To further illustrate the model, we outline how the data shown in Figure 1 are simulated. We carry out the simulation by generating the normal trades and the malicious trades separately, and then merge the two sequences as the observed trades.

Algorithm 1:

1. Simulate the normal trades. Each normal trade is independent and uniformly distributed in the square. The time interval between two consecutive normal trades is exponentially distributed with parameter γ (i.e., with mean $1/\gamma$).
2. Simulate the malicious trades.
   (i) Simulate the first malicious trade (the ancestor) in a malicious trade cluster: its location is uniformly distributed in the square; and the waiting time before an ancestor appears is exponential with mean $1/\epsilon$. The collusion set is activated when the ancestor appears.
   (ii) Simulate the next malicious trade in the active malicious trade cluster: the location of the next malicious trade follows a bivariate Gaussian distribution, centered at its ancestor, with variance parameter d; the waiting time before the next malicious trade appears is exponential with mean $1/(\epsilon + \lambda)$. The occurrence of this malicious trade has a probability p to deactivate the collusion set. If the collusion set is deactivated, go to (i), otherwise, repeat (ii).
3. Combine the two sequences and sort the sequence according to the trading time.

We set $\epsilon = 0.1$, $\gamma = 1$, $\lambda = 20$, $p = 0.2$ and $d = 0.01$ and generate 100 normal trades and 61 malicious trades. They are plotted in Figure 1 and Figure 2. In Figure 1, the malicious trades are depicted as circles whereas in Figure 2 all the trades are depicted as dots. So Figure 1 represents the underlying truth and Figure 2 shows what we observe.

We intentionally set the parameters to be rather extreme, so that the signal is strong, in the sense that the cluster pattern is obvious. In fact, our eyes can pick out many of the clusters in Figure 2. We would like to test how the model and the algorithms work under this simple setting.

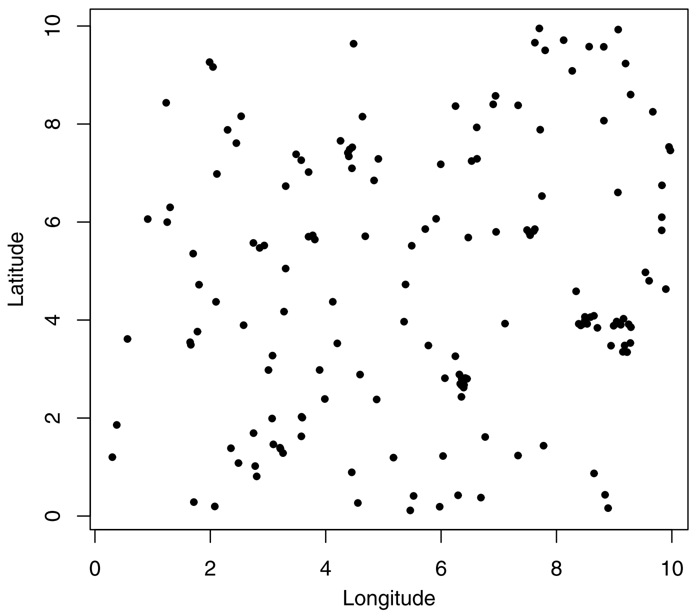

Figure 2. All the "trades" in Example 1.

3.1.2 Search for MLE

Given a set of parameters, the forward algorithm can compute the likelihood of the model $P(O_{1:T})$ efficiently. Hence finding the parameters that maximize the likelihood function becomes a standard optimization problem. We apply the Nelder-Mead simplex method and find the MLEs of the model are $\hat{\epsilon} = 0.16$, $\hat{\gamma} = 1.25$, $\hat{\lambda} = 21.51$, $\hat{p} = 0.21$ and $\hat{d} = 0.01$. They are not far away from the true parameters.

3.1.3 Conditional probabilities

Our goal is to compute the class conditional probabilities, and classify the trades based on them. The forward-backward algorithm suffices for this purpose. We take the MLEs as the input parameters and run the forward-backward algorithm, the output are the joint conditional probabilities $P(A_t, C_t, J_t|O_{1:n})$ for all possible combinations of A_t, C_t, J_t, $1 \leq t \leq n$. Actually, they are more than what we need since our main interest is the marginal conditional probabilities $P(C_i = 1|O_{1:n})$ for all i, $1 \leq i \leq n$ (Recall that C_t is the indicator of a trade being malicious).

We calculate these marginal conditional probabilities and plot them in Figure 3. The dots are the normal trades and the circles are the malicious trades. One can see that the two classes are well separated by the conditional probabilities. In particular, the conditional probabilities are close to 0 for all normal trades, and above 0.9 for all malicious trades. If we set the cut-off probability to be 0.5, in other words, we classify a trade t as malicious when $P(C_i = 1|O_{1:n}) > 0.5$, and as normal when otherwise, then we would have discovered the underlying truth precisely.

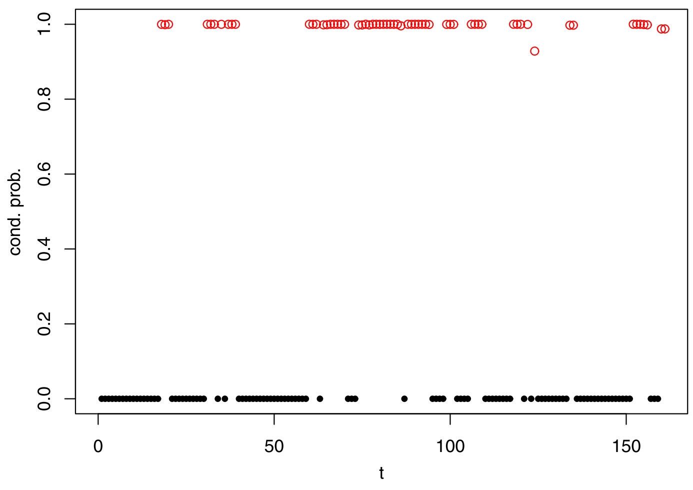

Figure 3. Output of the forward-backward algorithm in Example 1: $P(C_t = 1|O_{1:n})$ for all t, $1 \leq t \leq n$, i.e., the conditional probabilities of each trade being malicious. The dots are the normal trades and the circles are malicious trades.

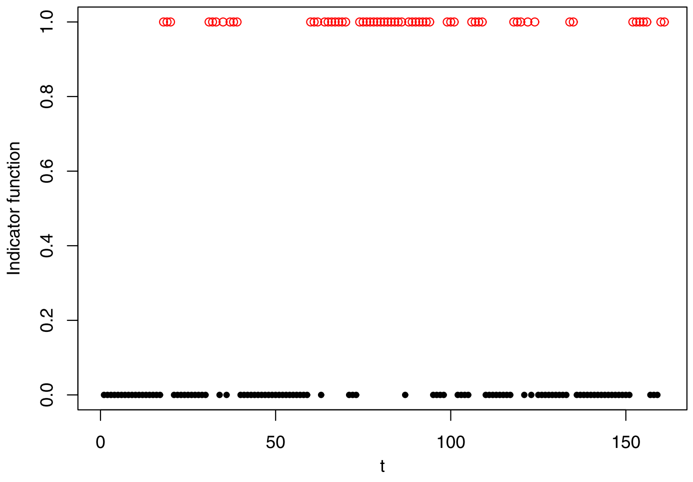

Figure 4. Output of the Viterbi algorithm in Example 1: trade t has a indicator equal to 1 if it is recognized as malicious by the Viterbi algorithm, and 0 if otherwise. The dots are the normal trades and the circles are malicious trades.

3.1.4 The most likely sequence

The Viterbi algorithm gives us the most likely hidden state sequence $\{q_t^*\}$ (see Appendix A.2), where $q_t^* = (A_t^*, C_t^*, J_t^*)$. We are interested in the indicators of a trade being malicious, i.e., $\{C_t^*\}$. Figure 4 plots $\{C_t^*\}$ for $1 \leq t \leq n$. The normal trades are in dots and the circles are the malicious trades. Again, the output of the Viterbi algorithm provides a perfect classification for this toy example.

The good performance of the algorithms is not too surprising because we have a strong signal in the simulation setup and the normal trades and malicious trades are generated according to our proposed model. Nevertheless, the re-

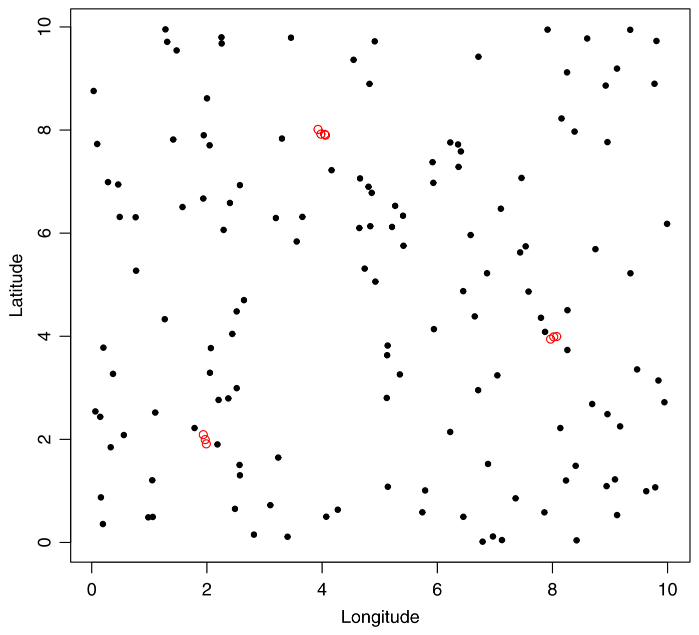

Figure 5. The "trades" in the robustness check experiment where the cluster not being centered at the ancestor. The dots are the normal trades and the circles are malicious trades.

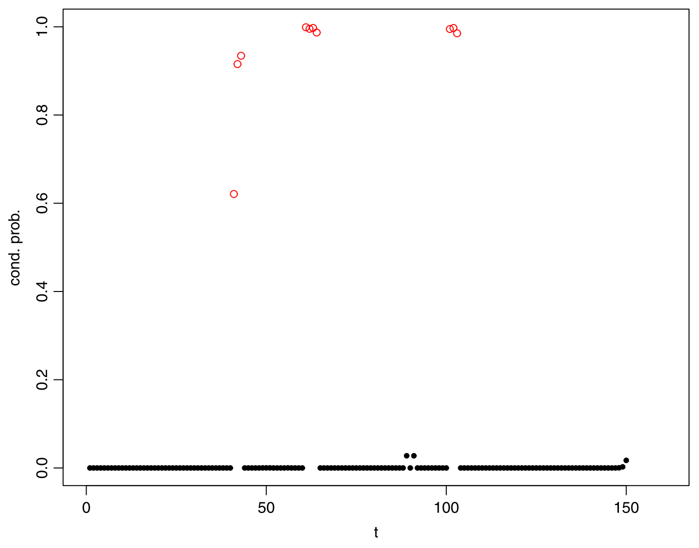

Figure 6. Output of the forward-backward algorithm in the robustness check experiment where the cluster is not centered at the ancestor: $P(C_t = 1|O_{1:n})$ for $1 \le t \le 150$, i.e., the conditional probabilities of each trade being malicious. The dots are the normal trades and the circles are malicious trades.

sults are encouraging as they illustrate that the QHMM approach can be very powerful. In particular, we can see from Figure 1 that several normal trades are spatially clustered, but they are not identified as malicious by the QHMM approach because the transaction times are taken into account. On the other hand, a detection procedure without considering the timestamps, such as the graph-clustering approach proposed in [8], is expected to give more false positives.

Because the programming associated with the QHMM approach is nontrivial, this simple example also helps us confirm that our codes are in order. One only needs to make minor changes of the codes to perform the data analysis for Example 2.

3.2 Robustness check

In order to test the robustness of the QHMM approach, we generate the trades differently from our proposed model, and then run the QHMM algorithms on the simulated data to see how they perform.

3.2.1 Cluster not centered at the ancestor

In this experiment, the succeeding malicious trades in a malicious trade cluster do not follow a bivariate Gaussian distribution centered at their ancestor. We simulate 150 trades on the 10 by 10 square. We let the trades be equally spaced in time. Three malicious trade clusters are generated on three 0.2 by 0.2 small squares uniformly. The small squares are respectively centered at (2,2), (4,8) and (8,4). The first cluster contains trades 41, 42, 43; the second cluster consists of trades 61, 62, 63 and 64; and the third cluster contains trades 101, 102, 103. The simulated data are depicted in Figure 5, where the dots are the normal trades and the circles are the malicious trades.

We fit the same QHMM as in Example 1 on this data set. The MLEs are $\hat{\epsilon} = 0.19$, $\hat{\gamma} = 1.07$, $\hat{\lambda} = 1.94$, $\hat{p} = 0.87$ and $\hat{d} = 0.004$. The MLEs are fed into the forward-backward algorithm, and the conditional probabilities $P(C_i = 1|O_{1:n})$, $1 \le i \le 150$ are calculated. The results are shown in Figure 6. Again one can see that malicious trades and the normal trades are well separated by the conditional probabilities. A cut-off probability of 0.5 would lead to a classification that is the same as the underlying truth. The QHMM approach works well in this simulation study. Actually, the assumption of the cluster centering at the ancestor is expected to be quite robust, when the malicious trades are highly clustered in space-time. This is because when the malicious trades are near to each other, then they must be close to the first malicious trade in the cluster, i.e., its ancestor.

3.2.2 When there is no malicious trades

This simulated data set consists of 150 normal trades and no malicious trades. They are generated on the 10 by 10 square with parameter $\gamma = 1$. The QHMM as in Example 1 is fitted on this data set. The MLEs are $\hat{\epsilon} = 0.0001$, $\hat{\gamma} = 1.02$, $\hat{\lambda} = 2.05$, $\hat{p} = 0.51$ and $\hat{d} = 0.59$. The MLEs are used as the input parameters for the forward-backward algorithm, and the conditional probabilities $P(C_i = 1|O_{1:n})$, $1 \le i \le 150$ are calculated and depicted in Figure 7.

One can see from the Figure 7 that all the conditional probabilities of the trade being malicious are near zero. In

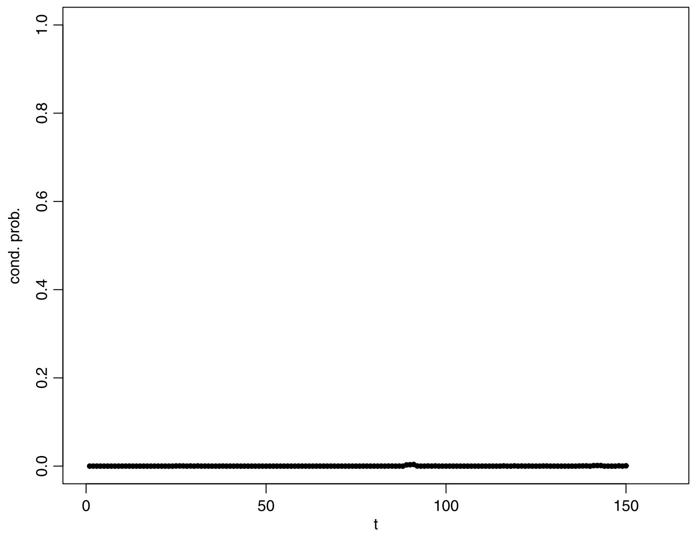

Figure 7. Output of the forward-backward algorithm when there are no malicious trades: $P(C_t = 1|O_{1:n})$ *for* $1 \le t \le 150$*, i.e., the conditional probabilities of each trade being malicious.*

fact, the largest one is only of 0.0037. Hence it strongly indicates that there is no malicious trade in the data set. Nevertheless, a graph-clustering approach as in [8] would report a collusion set candidate in this scenario.

3.3 Example 2

As mentioned in Section 2.3.2, the assumptions on $f_1(\cdot)$, $f_2(\cdot)$ and $f(R_{t+1}|R_{J_t})$ in this example are based on our empirical studies on real transaction data. In particular, we acquire tick-by-tick data on AsiaTiger from Singapore Exchange (SGX). Asia Tiger Group is a Singapore-based investment holding company mainly engaged in the manufacturing of office equipment. The tick-by-tick data consist of transaction time, volume, price, value in SGD and trade number. Partial transactions on AsiaTiger can be found in Table 3. In this section, we first carry out numerical experiments on simulated datasets, and then real data sets are used to test our method.

In practice, when training data sets are available, the model parameters can also be estimated based on the training data. Hence in this study, we adopt two parameter estimation methods: maximum likelihood estimation and estimation from a training set.

3.3.1 Simulation study

The simulation scheme in this example is similar to Example 1. Normal and malicious trades are generated separately and combined to form a complete testing data set. We generate 25 testing data sets. In each data set generation, we set $\epsilon = 0.05$, $\gamma = 4$, $\lambda = 30$ and $p = 0.4$ and every data set contains 200 normal trades, and the number of malicious trades varies. Instead of simulating the "longitude" and "latitude" as in Example 1, we generate trading volume, buyer ID and seller ID. In particular, the information is used to calculate the dissimilarities between trades.

1. Trader ID. We assume 150 traders are involved. For each normal transaction, buyer and seller ID are uniform random numbers between 1 to 147. And three traders $\{148, 149, 150\}$ form a collusion set. Hence buyer and seller IDs of malicious transactions are randomly generated from these three.
2. Trading volume. Our empirical studies shows that the trading volumes of AsiaTiger follow approximately a Log-normal distribution with $\mu = 3.95$ and $\sigma^2 = 1.2996$. We use the same set of parameters to generate trading volumes for normal transactions. Trading volumes of malicious transactions are generated using Log-normal distribution with $\mu = 7$ and $\sigma^2 = 0.05$.

With the same setup, we also generate a training set. The training data set consists of 2,000 normal transactions and 70 malicious transactions. We assume that we know which trades are malicious in the train data set, hence we can estimate the model parameters from the training set, say, by the method of moments.

We test our model on these 25 simulated data sets. Parameters are estimated using both the training data set and MLE method. The definition of dissimilarity measure we used in this experiment can be found in Appendix B. Forward-backward algorithm and Viterbi algorithm are run to identify the malicious transactions. Table 4 shows the results from the forward-backward algorithm (we set the cut-off probability to be 0.75 for conservativeness), where specificity is the percentage of normal transactions that are correctly identified, and sensitivity is the percentage of the malicious transactions that are correctly identified. The results on the detection errors (false positive plus false negative) of the Viterbi algorithm are shown in Table 5. Both algorithms classify most of the trades correctly.

The training data estimations have a comparable performance with the MLEs. However, we note that both estimation methods have their own merits and limits. The training data estimates are robust and easy to compute, but a training data set is not always available in practice. Especially one rarely knows the properties of the malicious trades. While the MLE is computational more expensive, and most optimization procedures do not guarantee a global maximizer.

3.3.2 An experiment with real data

We would like to perform similar experiments on real data. However, the buyer and seller information of the transactions are kept strictly confidential by the stock exchanges. The only exception to our awareness is the DSQ security mentioned in Section 1. SEBI provides the malicious transactions information under the request of India court. We attempted to acquire the normal transactions of DSQ security from SEBI but did not succeed.

Table 3. Partial transactions on AsiaTiger

Date	Time	Counter Name	Volume	Price	Value	Trade No.
2-Nov-09	091318	AsiaTiger	45000	0.32	14400	1004069
2-Nov-09	091318	AsiaTiger	100000	0.32	32000	1004068
2-Nov-09	091318	AsiaTiger	5000	0.32	1600	1004067
2-Nov-09	091512	AsiaTiger	5000	0.32	1600	1004451
2-Nov-09	092210	AsiaTiger	95000	0.32	30400	1005064
2-Nov-09	092210	AsiaTiger	50000	0.32	16000	1005065
2-Nov-09	092604	AsiaTiger	55000	0.32	17600	1005509
2-Nov-09	092820	AsiaTiger	5000	0.32	1600	1005734

Table 4. Sensitivity and specificity of detections using the forward-backward algorithm on 25 simulated data sets

	MLE		Training data	
Data Set	Specificity	Sensitivity	Specificity	Sensitivity
1	100.00%	100.00%	100.00%	100.00%
2	99.00%	88.89%	100.00%	88.89%
3	99.00%	100.00%	99.00%	100.00%
4	99.00%	80.00%	99.00%	80.00%
5	100.00%	100.00%	100.00%	100.00%
6	100.00%	100.00%	100.00%	100.00%
7	99.00%	100.00%	99.00%	100.00%
8	98.50%	100.00%	99.00%	100.00%
9	100.00%	100.00%	100.00%	100.00%
10	100.00%	100.00%	100.00%	100.00%
11	99.50%	100.00%	99.50%	100.00%
12	100.00%	100.00%	100.00%	100.00%
13	100.00%	100.00%	99.50%	100.00%
14	97.50%	88.89%	98.00%	100.00%
15	100.00%	100.00%	100.00%	100.00%
16	98.50%	100.00%	100.00%	100.00%
17	100.00%	100.00%	100.00%	100.00%
18	100.00%	83.33%	100.00%	83.33%
19	99.00%	100.00%	99.00%	100.00%
20	100.00%	100.00%	100.00%	100.00%
21	100.00%	60.00%	99.50%	60.00%
22	100.00%	100.00%	100.00%	100.00%
23	100.00%	88.89%	100.00%	88.89%
24	100.00%	100.00%	100.00%	100.00%
25	100.00%	100.00%	100.00%	100.00%

Table 5. Summary of Viterbi algorithm's performance on 25 simulated data sets

Data Set	1	2	3	4	5	6	7	8	9	10	11	12	13
Detection Err (MLE)	2	0	5	5	0	0	2	4	0	0	5	0	0
Detection Err (Training)	0	2	2	5	0	0	2	2	0	0	3	0	1
# of Transactions	209	206	209	204	204	204	210	209	205	206	206	205	204
Data Set	14	15	16	17	18	19	20	21	22	23	24	25	
Detection Err (MLE)	7	1	6	0	1	2	0	2	0	1	0	0	
Detection Err (Training)	7	1	0	0	1	2	0	3	0	1	0	0	
# of Transactions	208	214	204	205	208	204	204	208	208	208	205	206	

The data set used in this experiment is a combination of selected transactions from AsiaTiger and malicious transactions from DSQ security. For normal trades, 206 transactions on AsiaTiger, dated from 2 Nov to 6 Nov 2009, are chosen. We are not able to obtain the trader ID, so they are simulated as described in Section 3.3.1. For malicious trades, eight malicious transactions on DSQ security are selected to form two malicious trade clusters. IDs are assigned according to information provided in the law order. As two securities are from different countries, trading volumes are

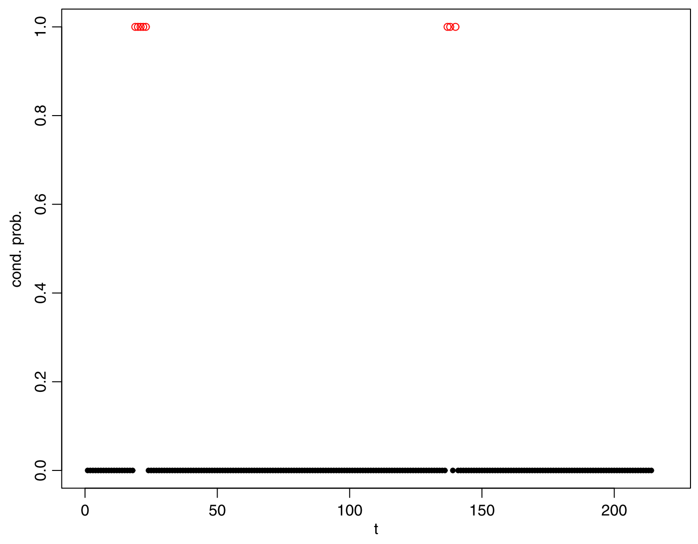

Figure 8. Output of the forward-backward algorithm in Example 2: $P(C_t = 1|O_{1:n})$ for all t, $1 \leq t \leq n$, i.e., the conditional probabilities of each trade being malicious. The dots are the normal trades and the circles are malicious trades.

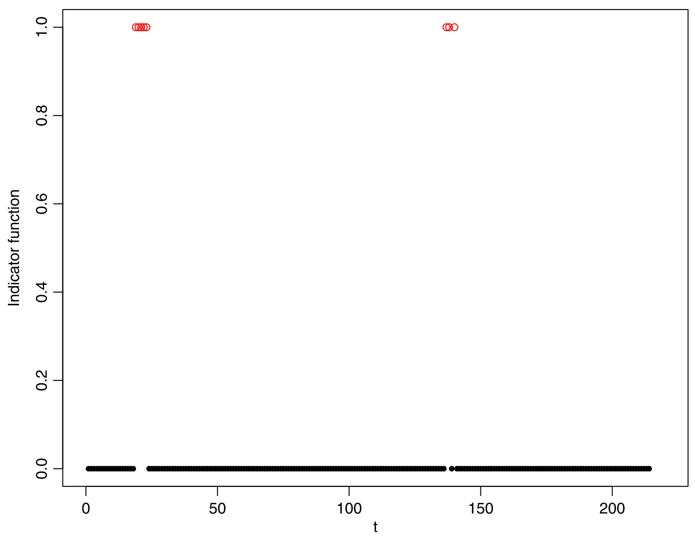

Figure 9. Output of the Viterbi algorithm: trade t has a indicator equal to 1 if it is recognized as malicious by the Viterbi algorithm, and 0 if otherwise. The dots are the normal trades and the circles are malicious trades.

Table 6. Parameters estimated by MLE

	γ	λ	ϵ	p	d
MLE	4.3707	34.6935	0.0432	0.4083	400.1777

adjusted according to the stock price and the currency exchange rate. More details can be found in [14].

We also construct a training data set consisting of 400 transactions on AsiaTiger and 20 transactions on DSQ security. The parameters of the Log-normal distributions (for the trading volume) are estimated from training data set. For normal transactions, $\hat{\mu} = 3.7641$ and $\hat{\sigma}^2 = 1.8217$; for malicious transactions, $\hat{\mu} = 8.4705$ and $\hat{\sigma}^2 = 0.1573$.

We estimate the remaining parameters by the maximum likelihood estimates. The results are shown in Table 6. As expected, when the collusion set is not active, it has a small intensity $\epsilon = 0.0432$; when it is activated, the intensity increases about 800 times, and it is about 8 times larger than the intensity of the normal trades. Using these parameters as input, the forward-backward algorithm and the Viterbi algorithm separate the malicious trades and normal trades very well. Figure 8 depicts the output of the forward-backward algorithm. The conditional probabilities almost look like indicator functions. All the malicious trades (in circles) have conditional probabilities close to 1, and all the normal trades (in dots) have conditional probabilities near 0. Figure 9 is the output from the Viterbi algorithm; it is almost identical to Figure 8. So both algorithms discover the underlying truth.

Our model performs excellently for this experiment. A possible reason for the perfect detection is that the signal in the data is really strong, in other words, the malicious transactions on DSQ security are too evident. This might be why this collusion set was caught while most collusion sets are undetected.

4. DISCUSSION

In this article, we propose a QHMM approach for collusion set detection. We model the security transactions as a marked point process so that the trading times can be naturally incorporated into the model. Then because the stochastic process is only partially observed, to make inferences on the hidden states, we set it up as a QHMM to make use of the forward-backward algorithm and Viterbi algorithm. These algorithms are recursive, hence our model is suitable for online monitoring and detection.

We demonstrate the proposed QHMM model with two examples. The first example is mainly for the purpose of illustration. The trades are considered as points in a region, and the normal trades and malicious trades follow their own distributions. In particular, the malicious trades tend to be clustered. Replacing the Euclidean distance in example 1 with a dissimilarity measure, we get our example 2 which is designed for real transaction data. In fact, example 1 is more than a toy example. Similar models are used for earthquake declustering [15, 16].

The QHMM approach explicitly estimates the conditional probabilities of each transaction being malicious, hence inferences are made on the basic question whether there are malicious trades in the transactions; whereas the graph-clustering approach proposed in [8] ignores such a question and always outputs a collusion set candidate, even when no collusion set exists in the data.

Our study is a first attempt to model the circular trading by stochastic processes. Though our simulation results and experiments with real data look promising, we are aware that they are under a rather fortunate scenario because the signal is strong. It would be interesting to see how our approach would work with real and large scale data.

APPENDIX A. ALGORITHMS ASSOCIATED WITH QHMM

In this section, we describe the forward-backward algorithm and the Viterbi algorithm associated with a QHMM. The detailed derivation of these algorithms can be found in [17].

A.1 Forward-backward algorithm

We define the forward variables $\alpha_t(i)$ as,

$$\alpha_t(i) = P(O_{1:t}, q_t = S_i), \quad 1 \le i \le N_t.$$

Then it is not hard to see that, for $1 \le i \le N_{t+1}$

$$\alpha_{t+1}(i) = \sum_{j=1}^{N_t} P(O_{t+1}, q_{t+1} = S_i | O_{1:t}, q_t = S_j)\alpha_t(j).$$

Hence one can compute $\alpha_t(i)$ inductively after setting $\alpha_1(i) = \pi(O_1, q_1 = S_i)$, $1 \le i \le N_1$, where $\pi(\cdot,\cdot)$ is the initial distribution of (O_1, q_1). The forward variables can be used in the computation of the likelihood of the model: $P(O_{1:T}) = \sum_{i=1}^{N_T} \alpha_T(i)$.

In a similar manner, the backward variables $\beta_t(i)$ are defined as,

$$\beta_t(i) = P(O_{t+1:T} | O_{1:t}, q_t = S_i), \quad 1 \le i \le N_t.$$

Note that the recursive relationship for $1 \le i \le S_t$,

$$\beta_t(i) = \sum_{j=1}^{N_{t+1}} P(O_{t+1}, q_{t+1} = S_j | O_{1:t}, q_t = S_i)\beta_{t+1}(j)$$

holds. Hence setting $\beta_T(i) = 1$, $1 \le i \le S_T$, the remaining $\beta_t(i)$ can be computed backward in time inductively.

Combining the forward procedure and the backward procedure, the probability of the hidden state is S_i at time t conditional on the observations is given by

$$P(q_t = S_i | O_{1:T}) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{N_t} \alpha_t(j)\beta_t(j)}.$$

A.2 Viterbi algorithm

The Viterbi algorithm uses dynamic programming to find the most likely hidden state sequence, namely, $\arg\max_{q_1,q_2,\ldots,q_T} P(q_1, q_2, \ldots, q_T | O_{1:T})$. We define the quantity $\delta_t(i) = \max_{q_1,\ldots,q_{t-1}} P(O_{1:t}, q_1, \ldots, q_{t-1}, q_t = S_i)$ and observe that

$$\delta_{t+1}(i) = \max_j \left[P(O_{t+1}, q_{t+1} = S_i | O_{1:t}, q_t = S_j)\delta_t(j)\right].$$

We also define $\psi_{t+1}(i) = \arg\max_j [P(O_{t+1}, q_{t+1} = S_i | O_{1:t}, q_t = S_j)\delta_t(j)]$ to keep track the index. Let $q_T^* = S_{\arg\max_j \delta_T(j)}$ and $q_t^* = S_{\psi_{t+1}(q_{t+1}^*)}$ for $t = T-1, T-2, \ldots, 1$, then one can show that $\{q_1^*, q_2^*, \ldots, q_T^*\}$ is the most likely hidden state sequence, i.e., $P(q_1^*, q_2^*, \ldots, q_T^* | O_{1:T}) = \max_{q_1,q_2,\ldots,q_T} P(q_1, q_2, \ldots, q_T | O_{1:T})$.

Table 7. Cumulative trading volume matrix, M

		Buyer ID				
		1	2	...	99	100
Seller ID	1	$m_{1,1}$	$m_{1,2}$	...	$m_{1,99}$	$m_{1,100}$
	2	$m_{2,1}$	$m_{2,2}$	...	$m_{2,99}$	$m_{2,100}$
	...	...	...	...	...	...
	99	$m_{99,1}$	$m_{99,2}$	...	$m_{99,99}$	$m_{99,100}$
	100	$m_{100,1}$	$m_{100,2}$	...	$m_{100,99}$	$m_{100,100}$

APPENDIX B. DISSIMILARITY

We let ID_i^b and ID_i^s denote the buyer and seller of trade R_i respectively. The intuition is that two transactions R_i and R_j are similar (hence suspicious) if the buyer ID_i^b and seller ID_i^s of R_i have a certain relationship with the buyer ID_j^b and seller $ID_j^s$of R_j. For example, they might be the same person, or there have been large volume tradings occured between them.

To this end, a matrix, M, is constructed to record and update the cumulative trading volumes (Table 7). In the matrix M, each entry represents the cumulative trading amount between two traders. For example, $m_{99,1}$ represents the cumulative trading amount sold from the trader with $ID = 99$ to the trader with $ID = 1$. The diagonal entries record the cumulative amount sold and bought by the same trader in a single transaction, which is highly suspicious for manipulation, and hence a penalty is imposed. In this study, we add ten times of the security's average trading volume to the diagonal entries. When a new transaction is executed, the matrix is updated by adding the trading volume to its corresponding entry.

The dissimilarity between transaction t and transaction J_t, the first malicious transaction executed by the most recent collusion set, is defined as: $D(R_t, R_{J_t}) = (\sum_{p=1}^{n_t}\sum_{q=1}^{n_t} M[p,q])/\{m_{ID_t^b, ID_{J_t}^b} + m_{ID_t^b, ID_{J_t}^s} + m_{ID_t^s, ID_{J_t}^b} + m_{ID_t^s, ID_{J_t}^s} + m_{ID_{J_t}^b, ID_t^b} + m_{ID_{J_t}^b, ID_t^s} + m_{ID_{J_t}^s, ID_t^b} + m_{ID_{J_t}^s, ID_t^s} + c\}$ where $M[p,q]$ is the entry on the p^{th} row and q^{th} column in the matrix, and n_t is the total number of traders, hence the numerator $\sum_{p=1}^{n_t}\sum_{q=1}^{n_t} M[p,q]$ is the summation of all entries in the cumulative trading volume matrix. The denominator consists of two parts. The first part is the summation of eight entries in the matrix. For example, $m_{ID_t^b ID_{J_t}^b}$ represents the cumulative trading amount until time τ_t sold from the buyer of transaction t to the buyer of transaction J_t. The second part of the denominator is a small constant c, which is added to ensure that the denominator does not equal to 0.

In this way, $D(R_i, R_j)$ is small if there have been large volume trading occured between ID_i^b, ID_i^s, ID_j^b and ID_j^s or for example, in the extreme case, if all four counterparties are the same.

Received 9 December 2011

REFERENCES

[1] Chakravarty, S. (2001). Stealth-trading: Which traders' trades move stock prices? *Journal of Financial Economics* **61** 289–307.

[2] Chapman, B. and Denniss, R. (2005). Using financial incentives and income contingent penalties to detect and punish collusion and insider trading. *The Australian and New Zealand Journal of Criminology* **38**(1) 122–140.

[3] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B* **39**(1) 1–38. MR0501537

[4] Engle, R. F. (2000). The econometrics of ultra-high-frequency data. *Econometrics* **68**(1) 1–22.

[5] Hughes, J. P. and Guttorp, P. (1994). A class of stochastic models for relating synoptic atmospheric patterns to regional hydrologic phenomena. *Water Resources Research* **30** 1535–1546.

[6] Lagarias, J., Reeds, J. A., Wright, M. H., and Wright, P. E. (1998). Convergence properties of the Nelder-Mead simplex method in low dimensions. *SIAM Journal of Optimization* **9**(1) 112–147. MR1662563

[7] Liu, Y., Zhang, H. H., and Wu, Y. (2011). Soft or hard classification? Large margin unified machines. *Journal of the American Statistical Association* **106** 166–177. MR2816711

[8] Palshikar, K. P. and Apte, M. M. (2005). Collusion set detection using graph clustering. In: *International Conference on Management of Data*, pp. 101–111.

[9] Palshikar, K. P. and Bahulkar, A. (2000). Fuzzy temporal patterns for analyzing stock market databases. In: *International Conference on Advances in Data Management*, pp. 135–142.

[10] Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. In: *Proceedings of the IEEE*, Vol. 77, No. 2.

[11] SEBI Order against M/S. Mittal Securities Private Limited dated 29 Sept, 2006. Order NO. WTM/GA/14/SD/5/07. http://www.sebi.gov.in.

[12] SEBI Order against Woodstock Securities Pvt. Ltd., Omega Equities Pvt. Ltd. and Woodstock Broking Pvt. Ltd. dated 14 Sept, 2006. Order NO. WTM/GA/15/ISD/6/07. http://www.sebi.gov.in.

[13] Sudjianto, A., Nair, S., Yuan, M., Zhang, A., Kern, D., and Cela-Diaz, F. (February 1, 2010). Statistical methods for fighting financial crimes. *Technometrics* **52**(1) 5–19. doi:10.1198/TECH.2010.07032. MR2752105

[14] Wu, X. (2010). Collusion set detection using a hidden Markov model, Technical report, National University of Singapore.

[15] Wu, Z. (2009). A cluster identification framework illustrated by a filtering model for earthquake occurrences. *Bernoulli* **15**(2) 357–379. MR2543866

[16] Wu, Z. (2010). A hidden Markov model for earthquake declustering. *Journal of Geophysical Research* **115** B03306.

[17] Wu, Z. (2011). Quasi hidden Markov model and its applications in multiple-change-point problems, Technical report, National University of Singapore.

[18] Wu, Z. (2011). Quasi hidden Markov model and its applications in cluster analysis of earthquake catalogs. *Journal of Geophysical Research* **116** B12316. doi:10.1029/2011JB008447.

Zhengxiao Wu
6 Science Drive 2
Department of Statistics and Applied Probability
National University of Singapore
Singapore, 117546
Singapore
E-mail address: stawz@nus.edu.sg

Xiaoyu Wu
80 Raffles Place
United Oversea Bank
Singapore 048624
Singapore
E-mail address: Wu.Xiaoyu@UOBgroup.com