Detecting Multiple Periods and Periodic Patterns in Event Time Sequences

Quan Yuan[†], Jingbo Shang[†], Xin Cao[‡], Chao Zhang[†], Xinhe Geng[†], Jiawei Han[†]

[†]Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801
{qyuan,shang7,czhang82,xgeng2,hanj}@illinois.edu

[‡]School of Computer Science and Engineering, The University of New South Wales, Australia 2052
xin.cao@unsw.edu.au

ABSTRACT

Periodicity is prevalent in physical world, and many events involve more than one periods, *e.g.,* individual's mobility, tide pattern, and massive transportation utilization. Knowing the true periods of events can benefit a number of applications, such as traffic prediction, time-aware recommendation and advertisement, and anomaly detection. However, detecting multiple periods is a very challenging task due to not only the interwoven periodic patterns but also the low quality of event tracking records. In this paper, we study the problem of discovering all true periods and the corresponded occurring patterns of an event from a noisy and incomplete observation sequence. We devise a novel scoring function, by maximizing which we can identify the true periodic patterns involved in the sequence. We prove that, however, optimizing the objective function is an NP-hard problem. To address this challenge, we develop a heuristic algorithm named Timeslot Coverage Model (TiCom), for identifying the periods and periodic patterns approximately. The results of extensive experiments on both synthetic and real-life datasets show that our model outperforms the state-of-the-art baselines significantly in various tasks, including period detection, periodic pattern identification, and anomaly detection.

1 INTRODUCTION

Periodicity, the phenomenon that an event occurs with regular time intervals, is prevalent in our life. Although a handful of methods have been proposed for periodicity detection [4, 9–11, 13, 17, 21, 22, 25, 27, 30, 36], most of them are unable to detect multiple periods — which are common for many real-life events. For instance, the precipitation rate in South America may increase every five years under the El Niño condition; female body temperature rises during ovulation periods; Kate, a graduate student in Computer Science, works on her homework at a CS Lab every evening and attends a lecture in the CS Lab every Wednesday morning (but her calendar is not known by others). By detecting the multiple periods involved in a target event, we can understand the reoccurring patterns of events more thoroughly, which will benefit a wide variety of applications, such as agricultural production, public policy-making, personal health condition tracking, and time-aware recommendation [38].

In this paper, we study the problem of detecting multiple periods from event observation sequence, where each observation indicates whether or not an event takes place at a particular timestamp. Figure 1 shows an example of an observation sequence, where the circles and crosses denote whether the event happens or not at corresponding timestamps. In addition to periods, we are also interested in periodic patterns so that we can infer a given event happens at what timestamps. This enables various applications in prediction and anomaly detection, *e.g.,* personal health assistant systems can predict the occurring times of symptoms if they could extract periodic patterns by analyzing the vital signs collected by wearable devices; periodicity has been demonstrated as an important factor underlying human mobility [5], which can be combined with other mobility patterns (*e.g.,* sequential patterns [41]) to predict human movements more accurately for advertisement and planning applications [26, 39, 42]; if the time of certain gatherings of people falls outside of a normal periodic pattern, this could raise a flag for police departments who are watching out for riots or protests.

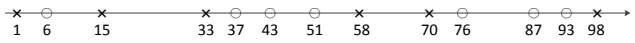


Figure 1: Observation sequence of an event

However, detecting multiple periods is non-trivial because:

- **Problem 1**: Multiple periods interweave. It is difficult to tell how many periods are involved in a given observation sequence and which period each observation belongs to.
- **Problem 2**: Observations are often incomplete due to limitations of data collection and inherent characteristics of the event, resulting in less evidence for periodicity detection. As shown in Figure 1, the observations are missing at many timestamps, such as 0, 2, 3, 4, 5, etc.
- **Problem 3**: Events rarely follow the patterns strictly, resulting in irregular observations, or even noise. There are two kinds of irregularities: false positives and false negatives. The former indicates that an event happens when it should not, *e.g.,* Kate attended a company's information session at the CS Lab on a Monday afternoon; the latter means that the event does not happen when it should, *e.g.,* Kate has no lecture to attend at the CS Lab on the

CIKM'17, November 6–10, 2017, Singapore.
 ISBN 978-1-4503-4918-5/17/11…$15.00
DOI: https://doi.org/10.1145/3132847.3133027

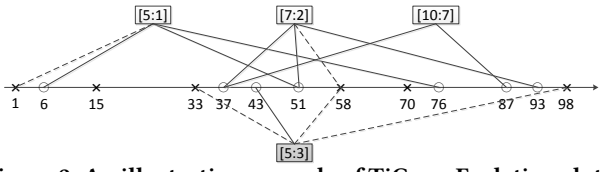


Figure 2: An illustrative example of TiCom. Each timeslot is connected to its covered timestamps, in particular, to timestamps with positive observations by solid lines and to timestamps with negative observations by dashed lines.

> Wednesday morning over spring break. The observations at timestamps 1, 43, and 58 in Figure 1 are all proved to be noise later. As we can see, the noise can contaminate observations and undermine a periodicity detection method.

Existing periodicity detection approaches [4, 9–11, 13, 17, 21, 22, 25, 27, 30, 36] can successfully detect a single period from observation sequences with high sampling rate and little noise, yet they cannot solve any of the three problems mentioned above. In this paper, we propose a generic approach that can address all three problems to effectively detect multiple periods from a noisy and incomplete observation sequence.

We assume that an observation sequence of a periodic event is generated by a mixture of periodic patterns with particular numbers of repetitions, contaminated by noise and missing observations to a certain extent. To represent the patterns of an event, we introduce the concept of **timeslot** - composed of a period l and an offset i, denoted by $[l : i]$. After segmenting a sequence with a timeslot's period and overlaying the pieces, if the timestamp lies in the offset of the segment, the timestamp is said to be covered by the timeslot. A timeslot can be either positive or negative: if positive, then an event that follows its pattern will always occur at covered timestamps; if negative, the event will never occur. Figure 2 shows an example, if an event follows periodic pattern with period 5 and positive timeslot $[5 : 1]$, the event happens at timestamps 1, 6, 11, etc.

As discussed in Problem 1, multiple periods may interweave with each other. If an event follows patterns of different periods, then it is possible that a timestamp is covered by both a positive timeslot of one pattern and a negative timeslot of another pattern. We observe that *the timestamp of each regular positive observation should be covered by at least one positive timeslot, whereas the timestamp of each regular negative observation should not be covered by any positive timeslots.* In Kate's case, everyday evening and Wednesday morning are two positive timeslots. Assuming Kate strictly follows the two periodic patterns, when Kate is at lab (positive observation), she is either doing homework or attending lectures (positive timeslots); when she is not at lab (negative observation), then this is neither her regular homework time nor regular lecture time. Incomplete observations also make detecting multiple periods challenging (Problem 2). In addition, as discussed in Problem 3, events do not always strictly follow periodic patterns. Telling whether an observation is regular is difficult. To cope with the problem caused by irregular observations, noise and incomplete observations, we aim to find the set of timeslots that can cover as many positive observations and as few negative observations as possible. Then we return their corresponding periods as the detection results.

Based on this premise, we develop a scoring function to measure the qualities of a given set of timeslots. Two aspects have taken into account: how well it can cover the timestamps of positive observations and how well it can avoid the timestamps of negative observations. For example, in Figure 2, compared with timeslot set $\{[5 : 3]\}$, the set $\{[5 : 1], [7 : 2], [10 : 7]\}$ has better quality because it covers more timestamps of positive observations and fewer timestamps of negative ones. If the latter set is the optimal solution to the scoring function, then the positive observations not covered by it (*e.g.,* at timestamp 43) and the negative observations covered by it (*e.g.,* at timestamps 1 and 58) are false positive and false negative noise, respectively. As shown in Section 4, we prove that finding the timeslots which maximize the scoring function is an NP-hard problem by a reduction from the vertex cover problem on tripartite graphs [12]. To solve the problem, we propose an effective and efficient heuristic algorithm named **Ti**meslot **Co**verage **M**odel (TiCom), which iteratively selects the timeslot with the maximum score as result until the overall score does not increase. We evaluate the performance of TiCom on both synthetic and real-life datasets. Our experimental results demonstrate that our method outperforms state-of-the-art baselines significantly on various tasks, including period detection, positive timeslot detection and anomaly detection.

In summary, the contributions of this paper are four-fold:

- We propose a general framework for periodicity detection. To the best of our knowledge, this is the first work that can detect multiple true periods - with periodic patterns - of incomplete and noisy observation sequence of an event.
- We prove that maximizing the proposed scoring function is an NP-hard problem.
- We design an effective and efficient heuristic solution TiCom to find positive timeslots based on the scoring function.
- The experimental results on both synthetic and real-life datasets show that TiCom outperforms state-of-the-art baselines significantly on various tasks.

The rest of this paper is organized as follows. In Section 2, we review related work. Section 3 provides the definition of notations and formulates the task. We introduce the optimization problem and the heuristic algorithm in Section 4. Experimental results are presented in Section 5. Finally, Section 6 concludes this paper.

2 RELATED WORK

2.1 Periodic Pattern Mining

Given a time sequence of observations, periodic pattern mining aims to discover periodic patterns with a given period. There are two types of periodic patterns: full periodic and partial periodic [28]. The difference lies in whether all pattern points (full) or parts of them (partial) exhibit periodicity. In this paper, we try to discover multiple partial periodic patterns with different periods for an event.

A number of research work has been done [14, 15, 23, 31, 33, 34] on mining periodic patterns from sequential data. Specifically, Ozden *et al.* propose to discover cyclic association rules that reoccur in every cycle of the sequence. In reality, however, time sequences can seldom meet the requirement of which the pattern is perfectly repeated. Han *et al.* [14, 15] relax the strong requirement on the perfect repetition of the pattern and present several methods of efficient mining in frequent partial periodic patterns from itemset sequences, based on Apriori and max-subpattern hit set properties. The partial periodic patterns are later extended in many studies [31, 33–35]. Despite the inspiring results obtained by these pioneer studies, they

are all based on the assumption that a periodic pattern is a certain subsequence that appears frequently in the given sequence.

2.2 Single Period Detection

Extensive methods have been proposed to detect periods from time sequence data. Fast Fourier Transform (FFT) based approaches and AutoCorrelation Function (ACF) based approaches are the most popular. Given a time sequence, FFT maps it to frequency domain and uses the inverse of the frequency with strongest power as the predicted period, while ACF estimates how similar a sequence is to its previous sequence with different lags and then uses the lag that maximizes ACF as the predicted period. Specifically, Glynn *et al.* [13], and Ahdesmaki *et al.* [2] detect periodicity by Lomb-Scargle periodogram [20], a variation of Fourier Transform using least-squares fitting of sinusoidal curves. Berberidis *et al.* [4] propose to use ACF to detect the candidate periods for binary sequences. Elfeky *et al.* consider the cases where the sequences consist of observations of multiple events [9]. They also employ ACF to detect the period of each event [9–11]. However, FFT cannot detect exact periods, especially large periods, due to the increasing width of the FFT bins [30]. It also suffers from spectral leakage problems. ACF calls for a carefully selected significance threshold for periods. Some proposals combine FFT and ACF for periodicity detection [17, 30], but they are still ineffective in handling partial periodic patterns and low-sampling rate, which are inherent problems in both FFT and ACF. Other techniques are employed for periodicity detection, such as seasonal-trend decomposition [6], chi-squared test [21], max subpattern tree [27], suffix tree [25], pattern combination [22], projection [36], etc. Nevertheless, all these methods work well for sequential data that have a relatively high sampling rate. When the sampling rate is low, the performance of these methods is limited. To address such problem, Junier *et al.* [16] detect periodic patterns based on that the timestamps at which an event happened tend to align, when the timeline is wrapped around a periodic solenoid. Li *et al.* [18, 19] segment the sequence into small chunks according to all possible periods, overlay them, and find the period w.r.t. which the distribution of observation is highly skewed. Yuan *et al.* [40] extract periods of individuals' visits by a Bayesian generative model, based on the observations that the gap time between two consecutive visits is roughly a multiple of the period.

2.3 Multiple Period Detection

Several methods have been proposed to detect multiple periods in time sequences. Dongen *et al.* [29] leverage Lomb-Scargle method to find a set of frequencies with highest power. For each frequency, they compute the standard deviation by temporarily subtracting the corresponding sinusoid from the time sequence. Parthasarathy *et al.* [24] combines FFT and ACF to detect multiple periods. Each time a period is detected, its harmonic component is subtracted from the original signal by a comb filter. Nevertheless, none of these can extract periodic patterns. Other studies exploits the support of the occurrences of periods. Yang *et al.* [37] first calculate the supports of prime periods, then check composite periods that are multiples of those valid prime periods. Xu *et al.* [32] calculate the time intervals between any two timestamps at which an event happened. Then they select the time intervals as periods if its support is greater than a threshold. However, the threshold is difficult to set. This method cannot address the low-sampling rate problem. Note that some other studies [9, 17] detect multiple periods for multiple events, rather than a single event.

3 PROBLEM SETTING AND OBJECTIVE

The problem's input is a sequence of observations $O = \{o_0, o_1, ..., o_{n-1}\}$ with size n, where $o_t = 1$ if it is observed that the event happens at timestamp t, $o_t = -1$ if it does not happen at t, or $o_t = 0$ if the observation at t is missing or unknown. Based on the observation values o_t, we group timestamps t into **positive timestamps set** $T_O = \{t|o_t = 1\}$, **negative timestamps set** $F_O = \{t|o_t = -1\}$ and **missing timestamps set** $M_O = \{t|o_t = 0\}$. A timestamp t is positive, negative, or missing, depending on the set it belongs to. Note that $o_t = -1$ (negative observation) and $o_t = 0$ (missing observation) are different: the former means we observe that the event does not happen at t; the latter says we do not know whether the event happens at t due to incomplete sampling. Consider the event that Kate visits the CS Lab. Suppose the observation sequences is $O = \{..., o_{21} = 1, o_{22} = 0, o_{23} = -1, ...\}$, from which we know that Kate stayed in the lab at timestamp 21 but not at timestamp 23 (*e.g.,* she was in her dorm) and we do not know where she was at timestamp 22 because the corresponding observation is missing.

Definition 1. *We define a* ***timeslot*** s_i *as a pair of a period* l *and an offset* i*, denoted by* $[l : i]$*. Timestamp* t *is* ***covered by*** *timeslot* $s_i = [l : i]$ *if* $t \bmod l \equiv i$*.*

The timestamps covered by $s_i = [l : i]$ constitute the set of positive timestamps of s_i: $P_{s_i} = \{t|t \bmod l \equiv i\}$. The timestamps not covered by s_i form the set of negative timestamps of s_i: $N_{s_i} = \{t|t \bmod l \not\equiv i\}$.

Definition 2. *We define a* ***periodic pattern*** p *with period* l *of an event as a sequence of* l *consecutive observations* $\{p_0, ..., p_i, ..., p_{l-1}\}$*, where* $p_i \in \{0, 1\}$*.*

Each observation p_i in pattern p with period l corresponds to a timeslot $s_i = [l : i]$, and $p_i = 1$ means the event always happens at timestamps covered by s_i, and $p_i = 0$ means the event always does not happen at timestamps covered by s_i. We call the timeslot s_i a **positive** (or **negative**) **timeslot** if $p_i = 1$ (or 0). Based on the values of p_i, we can group the timeslots s_i of pattern p into the set of positive timeslots $S_p^+ = \{s_i|p_i = 1\}$ and the set of negative timeslots $S_p^- = \{s_i|p_i = 0\}$. Kate's visits at the CS Lab follow a periodic pattern p with period 24 hours. Suppose Kate does her homework between 7:00 pm to 9:00 pm in the CS Lab every day, then the positive timeslots are $[24 : 19]$, $[24 : 20]$ and $[24 : 21]$, and we have $S_p^+ = \{19, 20, 21\}$. The remaining timeslots are included in S_p^-. Timestamps 20 and 44 are covered by timeslot $[24 : 20]$, and $P_{[24:20]} = \{20, 44, 68, ...\}$

Definition 3. *Given the observation sequence* O *and a pattern* p *with period* l*, we say that* l *is a* ***time period*** *of* O *if* $\forall s_i = [l : i] \in S_p^+$ *we have* $o_t = 1$ *for each* $t \bmod l \equiv i$*.*

The definition means that if O follows p, the values of observations are 1 if their corresponding timestamps are covered by the positive timeslots of p. Here we only consider the observations of timestamps covered by positive timeslots. l being a period of O does not mean that $\forall s_i \in S_p^-$ we have $o_t = -1$ for each $t \bmod l \equiv i$. This is due to the interwoven multiple periodic patterns in the sequence: one observation with timestamp covered by the negative timeslots

of a pattern is not necessarily negative because its timestamp might be covered by the positive timeslots of another pattern.

Suppose Kate goes to class in the CS Lab at 9:00 am every Wednesday. Thus, in addition to pattern p with period 24, the observation sequence also follows pattern p' with period 24×7. As a result, although the observation at 9:00 am on Wednesday should be negative according to p, it might be positive due to p'.

We may find that if O has a period l (*e.g.,* 24), then it will follows the multiples of l (*e.g.,* 48) as well, because the positive timeslots of the pattern of l (*e.g.,* $[24:20]$) can be replaced by a set of timeslots with greater periods (*e.g.,* $[48:20]$ and $[48:44]$). Intuitively, we do not want to report the redundant period 48 for O. Here we define the containment relationship between timeslots.

Definition 4. *A timeslot $s'_i = [l', i']$ is **contained by** timeslot $s_i = [l, i]$ if $P_{s'_i} \subseteq P_{s_i}$, or equivalently, $l' \bmod l \equiv 0$ and $i' \bmod l \equiv i$.*

For example, $[48:20]$ is contained by $[24:20]$, but $[168:87]$ is not. Based on the definitions above, we describe the objective of our periodicity detection problem as follows:

Problem Objective. Given an observation sequence O and a set of candidate positive timeslots $S = \{[l, i]\}$, we aim to detect the periods of true positive timeslots hidden in O from S, where no positive timeslot is contained by another.

Here we use $S = \{[l, i]\}$ to encode the prior knowledge of the event periodicity, *e.g.,* Kate may visit the CS Lab in afternoon or evening with possible periods 1 day and 1 week. After we find the optimal timeslots set, we report their periods as the detected periods, and the timeslots with the same period form a pattern of the event. If no prior knowledge is available, we can set a reasonable large enough upper bound period l_{max} (*e.g.,* 24×31, or $n/2$) and use the timeslots of all periods ranging from 2 to l_{max} as S.

4 PROPOSED MODEL

4.1 Model Formulation

Assume that the observation sequence O strictly follows several periods, *i.e.,* an event always or never happens predictably according to these periods and there is no missing observation. We observe that the timestamps of positive observations must be covered by at least one positive timeslot of a period, and the timestamps of negative observations must not be covered by a positive timeslot of any period. This can be explained as follows. Consider the probability that the event happens at time t, *i.e.,* $P(o_t = 1)$. Suppose t is covered by a set of timeslots S and the event happening at $s \in S$ follows a Bernoulli trial with success probability p_s. Then we have:

$$P(o_t = 1) = 1 - \prod_{s \in S} (1 - p_s) \tag{1}$$

Given $o_t = 1$, if all timeslots $s \in S$ are negative, *i.e.,* $p_s = 0$, $P(o_t = 1)$ is 0, then it contradicts the observation result and thus t must be covered by a positive timeslot. Similarly, the probability of an event not happening at t is calculated as follows:

$$P(o_t = -1) = \prod_{s \in S} (1 - p_s) \tag{2}$$

If any timeslot $s \in S$ is positive, $P(o_t = -1) = 0$, which contradicts $o_t = -1$. Thus t must not be covered by any positive timeslots. Based on the observation, we formalize our model as follows:

Periodicity detection problem. Given a sequence O and a candidate timeslot set S, we aim to find a set of positive timeslots $S^* \subseteq S$ such that 1) no timeslot in S^* is contained by another, and 2) the following scoring function is maximized:

$$f_O(S^*) = (1 - \alpha) \frac{|T_O \cap (\bigcup_{s \in S^*} P_s|)}{|T_O|} + \alpha \frac{|F_O \cap (\bigcap_{s \in S^*} N_s)|}{|F_O|} \tag{3}$$

The first part of Equation 3 measures how well the positive observations are covered by the candidate timeslots, and the second part measures how well the negative observations are not covered by any candidate timeslots. The two parts are balanced by a weighing parameter $\alpha \in [0, 1]$. Ideally, if we have no noise and missing observations, then the score of S^* is 1.

Equation 3 can be transformed as follows:

$$\begin{aligned} f_O(S^*) = &\quad (1 - \alpha) \tfrac{|\bigcup_{s \in S^*} (T_O \cap P_s)|)}{|T_O|} + \alpha \tfrac{|F_O - \bigcup_{s \in S^*} (F_O \cap P_s)|}{|F_O|} \\ = &\quad (1 - \alpha) \tfrac{|\bigcup_{s \in S^*} TP_s|)}{|T_O|} - \alpha \tfrac{|\bigcup_{s \in S^*} FP_s)|}{|F_O|} + \alpha, \end{aligned} \tag{4}$$

where $TP_s \doteq T_O \cap P_s$, and $FP_s \doteq F_O \cap P_s$.

We can better understand the scoring function using the Venn diagram in Figure 3, which represents the example illustrated in Figure 2. There are 4 candidate positive timeslots, whose positive timestamp sets are represented by ellipses. Note that different P_s of timeslot $s \in S$ may overlap with each other (*e.g.,* timestamp 51 belongs to both $P_{[5:1]}$ and $P_{[7:2]}$), and they may intersect with sets T_O, M_O, and F_O. To maximize the scoring function 4, we try to find a subset of timeslots $S^* \subseteq S$ such that: 1) the ratio of the grid-pattern area ($\bigcup_{s \in S^*} TP_s$) against the area of T_O is maximized, and 2) the ratio of the dotted-pattern area ($\bigcup_{s \in S^*} FP_s$) against the area of F_O is minimized. From the figure, we can find that S^* should include timeslots $\{[5:1], [7:2], [10:7]\}$, but not timeslot $[5:3]$.

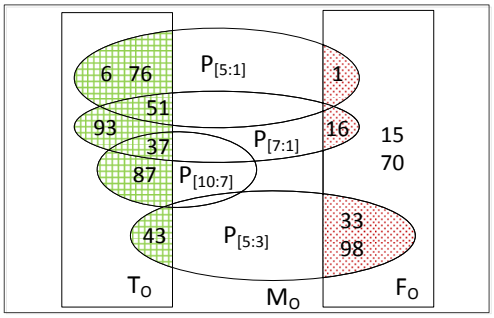


Figure 3: The Venn Diagram of the scoring function

Maximizing the score function enables us to addresses the problems of interleaving periods, low-sampling rate and noise in the following ways. 1) the detected positive timeslots may involve multiple periods. Thus it is able to detect multiple periods; 2) we map the positive or negative timestamps to positive timeslots, rather than checking the observations at all timestamps. Therefore, it is robust to the low-sampling problem; 3) we aim to cover more positive observations and fewer negative observations. Consequently, our model is less sensitive to noise, which may only account for a small fraction of the observations.

4.2 Problem Complexity

Theorem: Our periodicity detection problem as defined in Definition 6 is NP-hard.

Proof: We prove this by a reduction from the NP-hard minimum vertex cover problem in a tripartite graph [12]. In any tripartite graph, we can always partition the vertices into 3 sets s.t. there is no edge between the nodes from the same partition. The vertex cover problem aims to find a smallest set of vertices s.t. all edges will be covered by selected vertices.

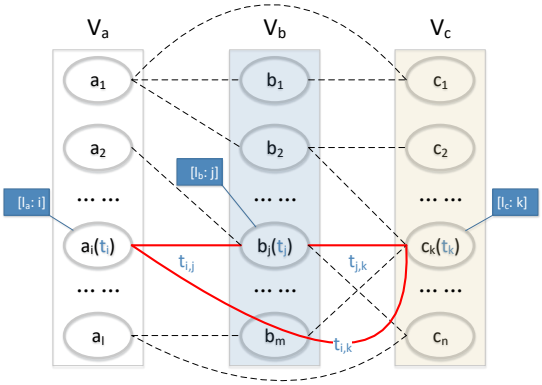


Figure 4: NP-proof mapping

Given a tripartite graph $G = (V, E)$, and its three tripartition vertex sets V_a, V_b, and V_c, we map this tripartite graph to an instance of our problem as follows:

First, we find the first 3 primes greater than $|V|$ and denote them as l_a, l_b, and l_c, respectively. According to the density of primes, finding all these primes should be of complexity $O(|V|)$ [3]. Then, for the i^{th} vertex a_i in V_a, we create a timeslot $[l_a : i]$. For the j^{th} vertex b_j in V_b, we create a timeslot $[l_b : j]$. For the k^{th} vertex c_k in V_c, we create a timeslot $[l_c : k]$. Note that i, j, and k start from 1.

Second, for each edge connecting a vertex a_i in V_a and a vertex b_j in V_b, we create a positive timestamp s.t. it is only covered by timeslots $[l_a : i]$ and $[l_b : j]$ but not by $[l_c : k]$. It is very important to observe that, due to the Chinese Remainder Theorem [7], we can always find a unique timestamp $t_{i,j}$ $(0 \le t_{i,j} < l_a \times l_b \times l_c)$ to represent this edge, such that

$$\begin{cases} t_{i,j} \bmod l_a \equiv i \\ t_{i,j} \bmod l_b \equiv j \\ t_{i,j} \bmod l_c \equiv 0 \end{cases}$$

Similar equations will be solved when considering the edges connecting a vertex from V_b to a vertex from V_c and the edges connecting a vertex from V_a to a vertex from V_c.

Third, for each vertex a_i in V_a, we create a negative timestamp t_i s.t. it is only covered by $[l_a : i]$ but not by $[l_b : j]$ and $[l_c : k]$. Again, according to the Chinese Remainder Theorem, we can always find such a timestamp t_i satisfying:

$$\begin{cases} t_i \bmod l_a \equiv i \\ t_i \bmod l_b \equiv 0 \\ t_i \bmod l_c \equiv 0 \end{cases}$$

Similarly, we do this for each vertex in V_b and V_c.

Finally, we have $T_O = |E|$ positive timestamps and $F_O = |V|$ negative timestamps. By this mapping, given a solution for the minimum vertex cover problem in the tripartite graph G, in the corresponding periodicity detection problem, all positive timestamps and the fewest negative timestamps are covered. According to Equation 4, this set of timeslots has the best score.

Algorithm 1: TimeSlot Coverage Model (TiCom)

Input: O, α, S
Output: S^*: the set of positive timeslots

$S^* = \{\}$;
$P_{S^*} = \{\}$;
$s = argmax_{s \in S - S^*}((1-\alpha)\frac{|TP_s - P_{S^*}|}{|T|} - \alpha\frac{|FP_s - P_{S^*}|}{|F|} + \alpha)$;
while $S \neq \Phi$ *and* $score^+_O(s) > 0$ **do**
  $S^* = S^* \cup \{s\}$;
  $S = S - \{s\}$;
  $P_{S^*} = P_{S^*} \cup \{P_s\}$;
  Remove all $s' \in S^*$ that is contained by s;
  $s = argmax_{s \in S - S^*}((1-\alpha)\frac{|TP_s - P_{S^*}|}{|T|} - \alpha\frac{|FP_s - P_{S^*}|}{|F|} + \alpha)$;
return S^*;

We choose a value for α such that $(1-\alpha)\frac{1}{T_O} > \alpha$. Let S^p be a set of timeslots covering all positive timestamps, and S^n be a set of timeslots not covering all positive timeslots. Based on Equation 4, we have $f_O(S^p) - f_O(S^n) \ge (1-\alpha)\frac{1}{T_O} - \alpha > 0$, because there is at least 1 positive timestamp not covered by S^n, and at most all negative timestamps covered by S^p but not by S^n. Hence, the best set of timeslots in the mapped periodicity detection problem must cover all positive timeslots and the fewest negative timeslots, and this is exactly the optimal solution for the vertex cover in the tripartite graph G.

In summary, we can reduce the the vertex cover problem in a tripartite graph $G = (V, E)$ which is NP-hard to an instance of our problem, where we have $|V|$ timeslots, an observation sequence of length $l = l_a \times l_b \times l_c \in O(|V|^3)$, and a suitable α value. Thus we complete the proof that our problem is NP-hard.

4.3 Solution

We propose out algorithm TimeSlot Coverage Model (TiCom) that can iteratively select the set of positive timeslots to maximize the scoring function in Equation 4. In each iteration, we select the timeslot that can increase the score most. Let S^* and P_{S^*} be the sets of the selected positive timeslots and the timestamps covered by the timeslots in S^*, respectively. Then the additional score brought by timeslot $s \in S$ is calculated as follows:

$$\begin{aligned} score^+_O(s) &= f_O(S^* \cup \{s\}) - f_O(S^*) \\ &= (1-\alpha)\frac{|TP_s - P_{S^*}|}{|T|} - \alpha\frac{|FP_s - P_{S^*}|}{|F|} \end{aligned} \quad (5)$$

We repeat this process until no timeslot can be found to increase the score $f_O(S^*)$. Each time a timeslot is selected, we remove $s' \in S^*$ that is contained by s. Note that we do not need to check if s is contained by the timeslot $s'' \in S$. If we do so, s will not be selected, as $P_s \subseteq P_{s''}$, $score^+_O(s) = 0$.

The algorithm of TiCom is summarized in Algorithm 1. For each timeslot $s = [l : k]$, by brute force, it takes n/l to calculate the score. In the worst-case scenario, we iterate $|S|$ times. Each time a timeslot is selected, we re-calculate the score of the remaining timeslots. Thus, the total time complexity is $O(\sum_{s=[l:k] \in S} |S| n/l)$.

Since S can contain at most all $\frac{n(n+1)}{2}$ possible timeslots, we have the following inequality:

$$\sum_{s=[l:k]\in S} |S|n/l \leq \sum_{l=1}^{n}\sum_{k=0}^{l-1} \frac{n(n+1)}{2} \cdot n/l = \sum_{l=1}^{n} n^2(n+1)/2$$

As a result, the worst-case time complexity of the brute-force algorithm is $O(n^4)$.

The bottle-neck of TiCom's efficiency is when each time a timeslot is selected, we need to re-calculate the score for every remaining timeslot. We can optimize the algorithm as follows. For each candidate timeslot $s' \in S$, we record $|TP_{s'}|$ and $|FP_{s'}|$. Each time a timeslot s is selected, for each timestamp $t \in P_s - P_{S^*}$, we reduce $|TP_{s'}|$ or $|FP_{s'}|$ by one if $o_t = 1$ or -1 for every timeslot $s' \in \{s'' = [l'', k'']|t \bmod l'' \equiv k'', s'' \in S - S^*\}$. Then, instead of re-calculating score for every candidate timeslot, we only need to update score $score^+_O(s')$ of timeslot s' whose $|TP_{s'}|$ or $|FP_{s'}|$ is modified due to the selection of s. During optimization, every timestamp will be added and removed exactly once, and thus the number of modifications to those TP and FP is no more than the total occurrences of timestamps in timeslots, which is $\sum_{s=[l:k]\in S} n/l$. Again, when S contains all $\frac{n(n+1)}{2}$ possible timeslots, the worst-case time complexity is $O(n^2)$.

Note that some wrong timeslots with longer periods might be selected by TiCom. This is because as the number of observations is fixed, fewer observations will be covered by a timeslot of a larger period. Also, even a few false positive observations will significantly increase the scores (first part of the right hand side of Equation 6). Thus, the scores of these timeslots are more sensitive to noise. To address this problem, we set a threshold score for timeslots with the same period. Specifically, given an observation sequence, we can generate a random permutation, which exhibits no periodicity and thus no timeslot is a positive timeslot. For each permutation, we calculate the score of every timeslot $s = [l : k]$ and keep track of the largest score for the period l. We repeat the process 100 times. For each period l, use the 99^{th} largest score as the threshold δ_l of the timeslots with period l. The timeslots $[l : k]$ whose scores are smaller than δ_l will be excluded from S^*.

5 EXPERIMENTS

In this section, we evaluate the proposed method on both synthetic and real-life datasets.

5.1 Experimental Settings

We generate synthetic datasets (*i.e.,* observation sequences) according to a set of parameters including periods and periodic patterns. Since the periods and periodic patterns are known, we are able to check how well different methods can recover these groundtruths for given testing sequences.

5.1.1 Data Generation. We use synthetic datasets to test the effectiveness of the proposed method, where the datasets are generated by the follow steps:

Step 1: We randomly select a set of $|L|$ periods $L = \{l_j\}_{j=0}^{|L|-1}$ from a uniform distribution in the range of [10:100), and for each period $l_j \in L$, we randomly generate a set of positive timeslots S_j^+ with a randomly selected size $|S_j^+| \in [1, |L| - 1]$;

Step 2: We fix a *number of repetitions* m, and set $n = m \times l_{max}$ as the length of time sequences, where l_{max} is the largest period in L;

Step 3: We generate observations $\{o_t\}_{t=0}^{n-1}$ according to the mixture of selected positive timeslots $\bigcup_{l_j \in L} S_{l_j}^+$. Specifically, we set $o_t = 1$ if $\exists l_j \in L$ st. $t \bmod l_j \in S_j^+$, and set $o_t = -1$ otherwise.

Step 4: We fix a *sampling rate* η, and set $o_t = 0$, $t \in [0, n-1]$ (missing observation) with probability $(1 - \eta)$.

Step 5: We fix a *noise ratio* β, and negate each non-zero observation (*i.e.,* making 1 as -1 and -1 as 1) with probability β.

By default, we have three periods 24, 32, and 48 with according positive timeslot sets {[24 : 10], [24 : 11], [24 : 14], [24 : 15], [48 : 40], [48 : 41]}, {[32 : 17]}, and {[48 : 40], [48 : 41]}, and set the values of n, α, β and η as 300, 0.6, 0.1 and 0.1, respectively [1].

5.1.2 Methods for Comparison.

Fast Fourier Transform (FFT): We employ Fast Fourier Transform to find the frequencies with the greatest power, and use their inverse as the predicted periods. To determine the number of periods, we employ the widely adopted 99% confidence approach [17, 18, 30].

Histogram and Fourier Transform (Histogram): We apply FFT on the histogram of gap time between positive observations, and report the periods with 99% confidence w.r.t. spectral power.

AUTOPERIOD [30]: We adapt the method proposed in [30] for multiple period detection. Specifically, we first apply Fourier Transform to identify the ranges of candidate periods, within which return the periods with power peak as results.

Autocorrelation and Fourier Transform (ACFFFT): We first calculate the autocorrelation (ACF) of different gap time, then retain the ones with 99% confidence w.r.t. ACF as candidates for further validation using FFT: only those whose frequencies are 99% confidence w.r.t. spectral power will be returned as results.

Discrepancy-based method (Discrepancy [18]). We calculate the discrepancy score for each candidate period. To detect multiple periods, each time we report the period with greatest score, and remove the observations belonging to the period. We iterate the process until no period with positive score can be found.

Lomb-Scargle Multiple Period Searching (LS [29]). We first apply Lomb-Scargle method to find a set of frequencies with highest power, and then for each frequency compute the standard deviation by temporarily subtracting its sinusoid from the time sequence.

Composite Period Searching (CPS [37]). This method first calculates the support of each prime-length period. All the periods with supports greater than a threshold are reported as results.

Interval Support based Method (IS [32]). This method calculates the gap time between every pair of positive observations, and reports the periods with support greater than the threshold.

Timeslot Coverage Model (TiCom). Our proposed model.

CPS and IS require a support threshold. We fix it as the optimal value 0.1 by grid search under the default parameter setting. TiCom takes no candidate timeslots but the upper bound period as input.

5.1.3 Tasks and Evaluation Metrics. For each synthetic observation sequence, we know all true periodic patterns (including true periods and positive timeslots) it involves. Based on the patterns, we can tell whether an observation is regular or noise. With the availability of the groundtruth information, we evaluate the performance of our proposed method by the following three tasks:

[1] The code is available at http://www.quan-yuan.com/datacode.html

Period Detection. We check whether a method can correctly detect all true periods for an observation sequence.

Positive Timeslot Detection. In addition to period detection, a good method should be able to understand the periodic patterns of observation sequence. Thus, the second task evaluates how well a method can detect all true positive timeslots.

Anomaly Detection. We are interested in seeing whether a method can successfully separate all irregular observations from regular ones. An observation is regarded as anomaly if it is covered by positive timeslots but the value is negative, or it is not covered by any positive slots but the value is positive (See Section 1).

We use the average F-1 measure of 100 trials as the evaluation metric. Note that the periodogram-based methods, namely, FFT, Histogram, AUTOPERIOD, ACFFFT, and LS, cannot infer the periodic patterns. Thus, they are incapable of detecting positive timeslots and anomalies. When evaluating the performance, we follow previous study [18] and set the upper bound period l_{max} to be 99.

5.2 Effectiveness Study

An good method should be able to perform well regardless of the choice of parameters. We change the value of a parameter each time, and examine the effectiveness of different methods.

Performance w.r.t. sampling rate η. We vary η and plot the results of all methods performed on different tasks in Figure 5(a), 5(b), and 5(c), respectively. We can find that our proposed method TiCom always achieves the best F-1 scores in all three tasks under all parameter settings, even when the sampling rate is low. This is because our model maps observations into positive timeslots, and find the set of timeslots that can maximize the overall score function. In comparison, the F-1 scores of other baselines are much worse. Specifically, the support-based method CPS and IS can hardly detect any true periods or timeslots when sampling rate is smaller than 0.25. As η increases, the performance of IS enjoys a rapid growth, because a greater sampling rate will make the supports more reliable. However, IS, even at its peak, still performs inferior to that of TiCom, and starts to drop after η=0.45, because some wrong periods start to have enough supports to be returned. Similarly, when η becomes larger (Figure 5(c)), CPS tends to return prime-length periods as the results, which however are not the true periods. The Fourier transform based methods FFT, Histogram and AUTOPERIOD can better detect periods with the increase of η, because more observations are available for period detection. The performance of ACFFFT increases at $\eta = 0.1$, and stays stable afterwards. This is because calculating ACF calls for enough number of observations. When the observations are sufficient, the performance still hits a ceiling because ACF is designed for sequences with only 1 period. When multiple periodic patterns interweave, the positive observations generated by a periodic pattern will disturb the sequence alignment of other patterns. The F-1 score of Discrepancy decreases when η increases, because Discrepancy favors the periods in which the positive observations and negative observations can be separated. Thus, as more observations become available, this method tends to report longer periods (smallest common multiple of possible periods). However, such long periods are not desirable as they can provide little information about the occurring patterns of an event.

Performance w.r.t. noise ratio β. We vary β and plot the performance of different methods for the three tasks in Figure 5(d), 5(e), and 5(f), respectively. Figure 5(d) and 5(e) show that with the ratio of noise increasing, the effectiveness of all methods in detecting periods and timeslots decreases and reaches the bottom at β=0.45. Our model TiCom achieves the best performance all the time, because it selects timeslots that can cover more positive observations but fewer negative ones. As a result, the influence of noise is mitigated. Among the baselines, the performance of ACFFFT drops the fastest, because the ACF-based model is sensitive to noise, which hinders the sequence alignment. FFT, Histogram and AUTOPERIOD reach their peaks at β=0.25 or 0.35, probably because the spectral leakage problem is suppressed by the noise at these points, but the time series is overwhelmed by noise afterwards. The performance of Discrepancy keeps stable and starts to decline at $\beta = 0.35$, succumbing its robustness to noise. CPS and IS fail to detect any true periods or timeslots, because the support-based methods are sensitive to noise. For anomaly detection (Figure 5(f)), however, the performances of these two methods are fairly decent when β is large (their curves often overlap with each other). We found they can only detect one wrong positive timeslots, and thus they tend to classify most of the observations as anomalies. With the increase of β, there are more irregular observations in the sequences, thus these two methods have good recall values that bring the F-1 score up.

Performance w.r.t. number of repetitions m. We vary m and plot the results of different methods on the three tasks in Figure 5(g), 5(h) and 5(i). TiCom dominates with perfect results from the beginning to the end, because it cares about the percentage of covered observations, rather than the exact number. In addition, with m increases, the performance of Discrepancy decreases on the tasks of period detection and timeslots detection. The reasons are the same as the experiments with varying the sampling rate η.

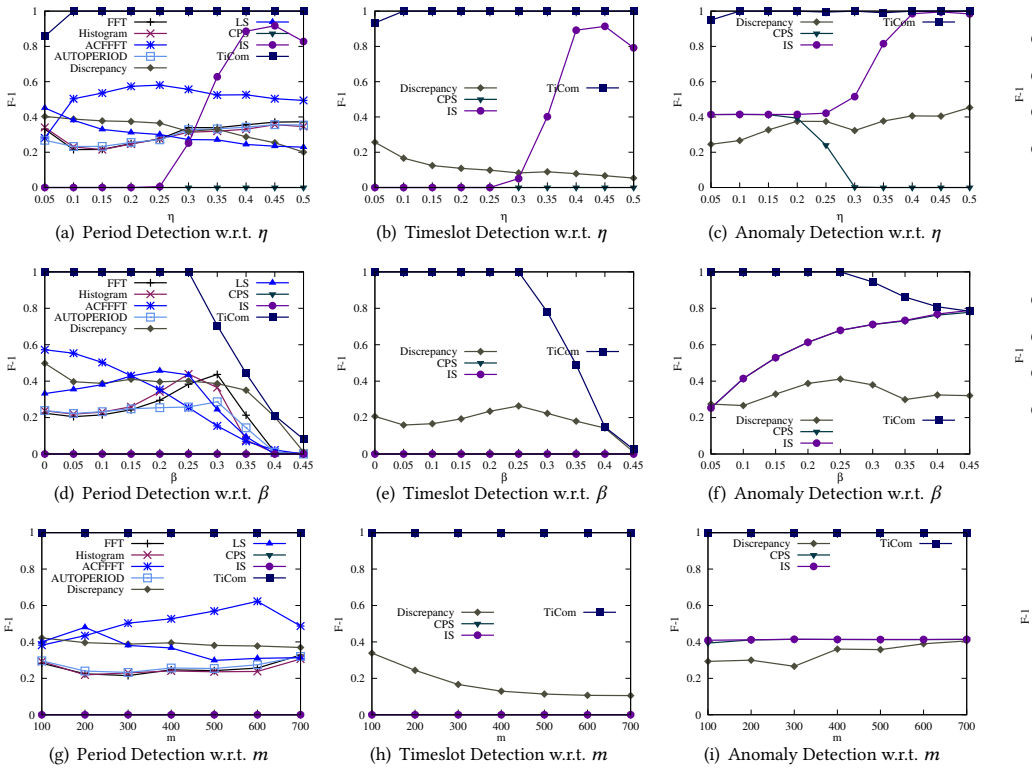
Performance w.r.t. number of periods $|L|$. We are interested in the capability that a method can discover different number of periods from the observation sequences. To examine this, we first randomly generate a set of periods L with size $|L|$ ranging from 10 to 100. For each period $l \in L$, we randomly select a number of positive timeslots M_l, and generate positive timeslots ranging from 0 to $l-1$, where $1 \leq M_l \leq min(\sqrt{l}, 5)$. When generating the set of positive timeslots of all periods $l \in L$, we guarantee that no positive timeslot is contained by another.

We vary the value of $|L|$ from 1 to 10, and plot the results of different methods on Figure 6. As we can see, when $|L| = 1$, both Discrepancy and TiCom achieve the best F-1 scores. This is because Discrepancy is a special case of TiCom when there is only a single period and $\alpha = 0.5$, even though the theoretical premises of the two methods are different. When $|L| > 1$, TiCom performs much better than Discrepancy. The results of other baselines are much worse, because IS and CPS are sensitive to sampling rate and noise, and the other methods are not designed to detect multiple periods.

Sensitivity to α. We vary α and plot the results in Figure 7. From the figures, we can find that TiCom can achieve good accuracy in a large range from $\alpha = 0.5$ to 0.9 with peaks at 0.6 and 0.7.

5.3 Efficiency Study

For many events like water level, precipitation rate and ocean temperature, the observation sequences can be very long and are only getting longer. Thus, a good periodicity detection method should be efficient and scale well to handle such data. In this

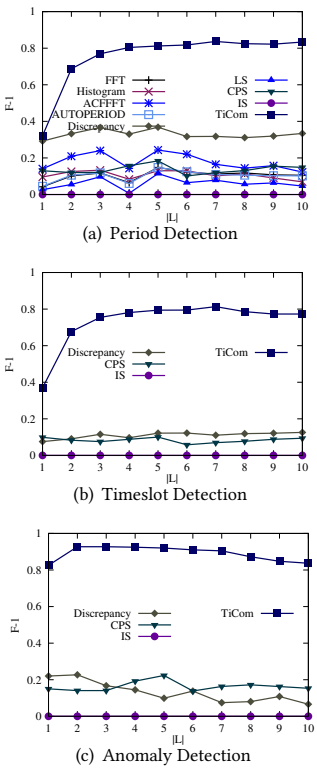


(a) Period Detection w.r.t. η (b) Timeslot Detection w.r.t. η (c) Anomaly Detection w.r.t. η

(d) Period Detection w.r.t. β (e) Timeslot Detection w.r.t. β (f) Anomaly Detection w.r.t. β

(g) Period Detection w.r.t. m (h) Timeslot Detection w.r.t. m (i) Anomaly Detection w.r.t. m

Figure 5: Varying parameter settings



(a) Period Detection

(b) Timeslot Detection

(c) Anomaly Detection

Figure 6: Varying $|L|$

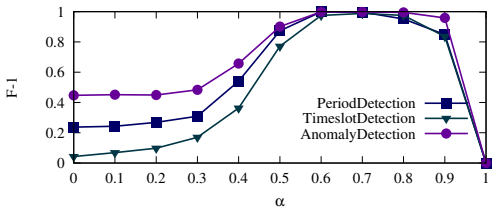
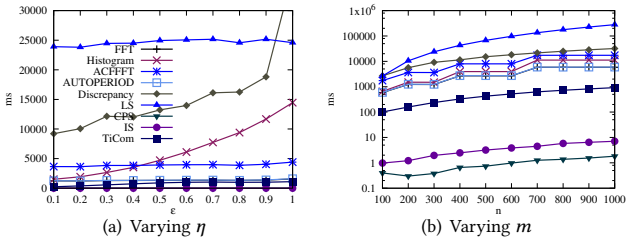


Figure 7: Results with various α

section, we examine the running time of different methods under different non-zero observations (controlled by η) and length of observation sequences (controlled by m). We increase η from 0.1 to 1.0, and increase n from 100 to 1000. Under each setting, we run each method 100 times, and plot the average running time. The efficiency is tested on an iMac with 3.2GHz quad-core Intel Core i5 CPU and 16G RAM.



(a) Varying η (b) Varying m

Figure 8: Average running time of different methods

From the two figures, we find the support-based methods IS and CPS are always the most efficient, but they can hardly detect any true periods or timeslots. Among the effective methods, TiCom is the most efficient. In addition, TiCom shows superior scalability. For example, the average running time of TiCom on m=100, 400, 700, 1000 are 101.11, 237.68, 625.42 and 919.96 ms. Our method also scales well w.r.t. η. Specifically, when η=0.1, 0.4, 0.7, 1.0, the average running time of TiCom is 240.96, 742.66, 1032.41, and 1099.04, respectively. Among the baselines, LS is the most time consuming: when m=1000, it takes 279,718.37 ms to finish a prediction. It is interesting to observe that TiCom runs even faster than FFT. A possible reason is that FFT needs additional time to calculate the 99% confidence spectral power. Although TiCom faces the same problem, FFT requires more computation time because it must extend the length of observations to a power of 2. As the extension is repeated 100 times, FFT becomes much slower.

5.4 Case Study on Real-life Data

We use human mobility data and tide height data to evaluate the effectiveness of the proposed method. For both experiments, the parameter α is set to 0.7 (Figure 7). Its value can be tuned when a development set is available.

5.4.1 *Mobility Periodicity Detection.*

Reality Mining Dataset. This dataset contains the call logs, cell tower IDs and inferred where 106 students and faculty are at every hour [8]. For each hour, if the cellphone of an individual is within the range of the cell towers on campus, we get a positive observation. If the cellphone is within the range of other cell towers, we get a negative observation. Otherwise the observation is missing. We consider the users who have length of observation sequence greater than 200 (the greatest possible period). After pre-processing, there are 71 users, with average observation length 2327 (about 97

Table 1: Detected periods on reality mining data

user ID	FFT	AF	LS	DC	TC
8	24, 171	24, 171	24,166	24,12	24, 168
9	24, 171	24, 171	24, 12	24, 171	24, 168
15	24, 28	24, 28	24, 28	24, 169	168, 24
23	24, 12	24, 167	170, 24	24,12	24, 168
25	24, 171	24, 171	24, 173	24,12	24, 168

days), and average number of non-missing observations 1632.68. Although we do not know the true periods of an individual visiting the office, we do know 24 hours (1 day) and 168 hours (1 week) should be two meaningful periods. Note that the mobility of an individual may oscillate sometimes, *e.g.,* arrive office at 8:00 AM on some days, and arrive at 9:30 AM on some other days, and the mobility might be also influenced by other factors, *e.g.,* Daylight Saving Time. However, a good periodicity detection method should be robust to such fluctuations and shifts.

We perform the methods FFT, ACFFFT (AF), LS, Discrepancy (DC), and TiCom (TC) to detect the periods for each individual, and record the detection results if any method find both 24 and 168 as the closest-to-true-value periods. Note that the two periods are not known by any testing methods in advance. We found only the proposed TiCom can find such periods. The the top 2 periods of 5 randomly selected users with periods 24 and 168 being identified by TiCom are presented in Table 1. Among the five methods, TiCom can detect both 24 and 168 as true periods for 9 users, while none of other methods can find one. This demonstrates the effectiveness of our proposed model in multiple period detection.

We also illustrate the positive timeslots of periods 24 and 168 for user 8 that are detected by TiCom in Table 2 (for the timeslots of 168, we convert them into day:hour pairs). We can find that the user stays at campus everyday afternoon from 2:00 to 5:00PM. In addition, from Monday to Thursday, the user arrive a little earlier and leave a little late, but such pattern cannot be observed on Friday and weekends. The results again demonstrate the effectiveness of TiCom in discovering individuals' mobility patterns.

Table 2: Detected periodic patterns for user 8

Period	Positive Timeslots
24	14; 15; 16; 17
168	Mon:12,13,18; Tue:13,18; Wed:13,18; Thu: 13,19,20,21

Twitter Data. We randomly select 200 twitter users, and crawled the most recent 3,200 tweets. We map the geo-tagged tweets to 10*10 meters grids. We only focus on the users who posted more than 200 geo-annotated tweets. After filtering, 116 users are left. For each user, we find the grid that he/she visits most frequently, and transform the visiting records into observation sequences on an hourly-basis: if the user posted a tweet within the grid, we have a positive observation at the corresponding hour. If the tweet is posted in other grids, we have a negative observation. The average length of observation sequence is 29,628 (about 3 years and 4 months), and the average number of observations is 1,170.52.

In contrast to that on RM dataset, this time all methods can detect 24 and 168 for several users, probably because the average observation number and observation length are much larger. We randomly select 5 users for whom any of the testing methods can detect 24 and 168 as their periods, and list the detection results in Table 3. We find that the baseline methods report several periods rare in real world, such as 6, 21, 112. In addition, TiCom can identify 24 and 168 as the periods for 16 users, while the number is 11, 13, 14, and 0 for FFT, AF, LS, and DC, respectively.

Table 3: Detected periods on Twitter data

user ID	FFT	AF	LS	DC	TC
u_1	24, 8	24, 168	12, 24	24, 84	24, 168
u_2	168, 12	168, 24	168, 28	84, 96	48, 168
u_3	24, 168	24, 6	24, 112	24, 123	168, 48
u_4	24, 8	24, 168	8, 6	24, 84	24, 168
u_5	24, 12	24, 168	24, 168	21,12	24, 48

5.4.2 *Tide Cycle Detection.*

Tides are the rise and fall of sea levels caused by the combined effects of the gravitational forces exerted by the Moon and the Sun and the rotation of Earth [1]. Therefore they have multiple periods a.k.a. tide cycles. The primal period is about 12 hours and 25.2 minutes (principal lunar semi-diurnal); when it's new moon (roughly a lunar month), the tide's range is at its maximum due to the gravity of the Sun and Moon from the same side.

We collected the water level records of Boston, MA from July 01 to August 31, with 6 minutes as the sampling interval[2] (the tide levels are plotted in Figure 9). Then, we convert the observations into time sequence as follows: we first detect all peaks and bottoms, and then rank peak heights and bottom heights respectively in descending order. The median height of peaks h_p and the median height of bottoms h_b are used as the thresholds, *i.e.,* if the water level h_t at time t is larger than h_p, we have observation $o_t = 1$; if the water level h_t is smaller than h_b, we have $o_t = -1$; otherwise $o_t = 0$. After conversion, we get an observation sequence of length 14880, containing 282 non-zero observations. Our method is designed to detect periodic patterns from sequences made up of -1, 0, 1, which has some crucial applications in real life. Take our test data as an example, it's unsafe to sail when the tide range exceeds a threshold (*e.g.,* $h_p - h_b$). The detected periods and patterns can inform sailors of safe dates to go out into the ocean. Admittedly, some methods can deal with numeric observations, but we feed all methods with the same observation sequence for a fair comparison.

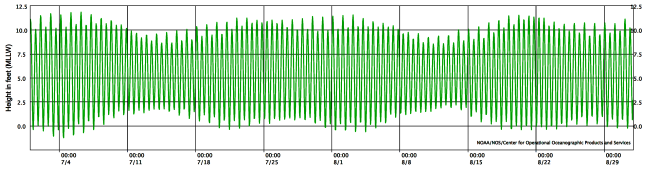


Figure 9: Tide level of Boston, MA (Jul. 01 to Aug. 31, 2016)

Again, we perform 5 methods FFT, AF, LS, DC and TC on the sequence. The detected periods and the numbers of detected periods of these methods are shown in Table 4. To make it easier to read, we convert the detected periods on a hourly- (hr) or daily- (d) basis.

From the table, we find the baselines FFT, AF, and LS report more than 15 periods, most of which are obviously incorrect, such as 1.8 hours and 2.5 hours. Calculating from the sampling rate (6 minutes), we have 12.4 hours as the closest period to the ground truth of 12 hours and 25.2 minutes. DC does successfully detect the

[2]The water level data is available at http://tidesandcurrents.noaa.gov/

Table 4: Tide cycle detection results

Method	Detected Periods	# Periods
FFT	2.5hr, 2.8hr, 4.1hr, 4.2hr, 4.9hr, ...	32
AF	1.8hr, 1.9hr, 2.3hr, 11.8hr, 11.9hr, ...	25
LS	2.5hr, 4.1hr, 4.2hr, 8.4hr, 8.5hr, ...	15
DC	12.4hr	1
TC	12.4hr, 28.5d, 29d	3

period of 12.4 hours (12 hours and 24 minutes), but it is the only period it finds. Our method TiCom reports 12.4 hours, 28.5 days and 29 days as the only 3 detected periods, which are close to the principal lunar semi-diurnal period and a lunar month. The results demonstrate that TiCom is effective in detecting multiple periods.

6 CONCLUSION

Many events involve with multiple periods. How to detect them and their corresponding periodic patterns is still an open problem. The interweaving periods, incomplete observations and noise make up three major challenges. To address them, we define a scoring function to measure how well an observation sequence can be generated by a set of periodic patterns. However, maximizing the scoring function is an NP hard problem. To this end, we develop a heuristic algorithm – Timeslot Coverage Model (TiCom), to identify periods and periodic patterns. Extensive experimental results on both synthetic and real-life datasets have demonstrated TiCom's superior effectiveness and efficiency against state-of-the-art baselines.

Several interesting directions exist for future exploration. First, in addition to noise and low sampling rate, time series may also involve shift bias, *e.g.,* Kate visits the CS Lab at 6:30 PM from January to March, but at 8:00 PM from April to May. We plan to handle the shift bias in our future work. Second, in our problem setting the times of observations are divided into integer bins. Support times in real numbers is also in our plan. Last, it would be promising to exploit the values of observations for period detection.

7 ACKNOWLEDGEMENTS

This work was sponsored by the U.S. Army Research Lab. under Cooperative Agreement No. W911NF-09-2-0053 (NSCTA), National Science Foundation IIS-1017362, IIS-1320617, and IIS-1354329, HDTRA1-10-1-0120, and Grant 1U54GM114838 awarded by NIGMS through funds provided by the trans-NIH Big Data to Knowledge (BD2K) initiative (www.bd2k.nih.gov), and MIAS, a DHS-IDS Center for Multimodal Information Access and Synthesis at UIUC. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing any funding agencies.

REFERENCES

[1] Tide, from wikipedia. https://en.wikipedia.org/wiki/Tide.
[2] M. Ahdesmäki, H. Lähdesmäki, A. Gracey, and O. Yli-Harja. Robust regression for periodicity detection in non-uniformly sampled time-course gene expression data. *BMC bioinformatics*, 8(1):233, 2007.
[3] J. Barkley Rosser and L. Schoenfeld. Approximate formulas for some functions of prime numbers. *Illinois Journal of Mathematics*, 6:6z–9z, 1962.
[4] C. Berberidis, W. G. Aref, M. Atallah, I. Vlahavas, and A. K. Elmagarmid. Multiple and partial periodicity mining in time series databases. In *ECAI*, volume 2, pages 370–374, 2002.
[5] E. Cho, S. A. Myers, and J. Leskovec. Friendship and mobility: user movement in location-based social networks. In *Proc. KDD*, pages 1082–1090. ACM, 2011.
[6] R. B. Cleveland, W. S. Cleveland, and I. Terpenning. Stl: A seasonal-trend decomposition procedure based on loess. *Journal of Official Statistics*, 6(1):3, 1990.
[7] C. Ding, D. Pei, and A. Salomaa. *Chinese remainder theorem: applications in computing, coding, cryptography*. World Scientific, 1996.
[8] N. Eagle and A. Pentland. Reality mining: sensing complex social systems. *Personal and ubiquitous computing*, 10(4):255–268, 2006.
[9] M. G. Elfeky, W. G. Aref, and A. K. Elmagarmid. Periodicity detection in time series databases. *TKDE*, 17(7):875–887, 2005.
[10] M. G. Elfeky, W. G. Aref, and A. K. Elmagarmid. Warp: time warping for periodicity detection. In *Proc. ICDM*, pages 8–pp. IEEE, 2005.
[11] M. G. Elfeky, W. G. Aref, and A. K. Elmagarmid. Stagger: Periodicity mining of data streams using expanding sliding windows. In *Proc. ICDM*, pages 188–199. IEEE, 2006.
[12] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.
[13] E. F. Glynn, J. Chen, and A. R. Mushegian. Detecting periodic patterns in unevenly spaced gene expression time series using lomb–scargle periodograms. *Bioinformatics*, 22(3):310–316, 2006.
[14] J. Han, G. Dong, and Y. Yin. Efficient mining of partial periodic patterns in time series database. In *Proc. ICDE*, pages 106–115. IEEE, 1999.
[15] J. Han, W. Gong, and Y. Yin. Mining segment-wise periodic patterns in time-related databases. In *Proc. KDD*, pages 214–218, 1998.
[16] I. Junier, J. Hérisson, and F. Képès. Periodic pattern detection in sparse boolean sequences. *Algorithms for Molecular Biology*, 5(1):1, 2010.
[17] Z. Li, B. Ding, J. Han, R. Kays, and P. Nye. Mining periodic behaviors for moving objects. In *Proc. KDD*, pages 1099–1108. ACM, 2010.
[18] Z. Li, J. Wang, and J. Han. Mining event periodicity from incomplete observations. In *Proc. KDD*, pages 444–452. ACM, 2012.
[19] Z. Li, J. Wang, and J. Han. eperiodicity: Mining event periodicity from incomplete observations. *TKDE*, 27(5):1219–1232, 2015.
[20] N. R. Lomb. Least-squares frequency analysis of unequally spaced data. *Astrophysics and space science*, 39(2):447–462, 1976.
[21] S. Ma and J. L. Hellerstein. Mining partially periodic event patterns with unknown periods. In *Proc. ICDE*, pages 205–214. IEEE, 2001.
[22] M. A. Nishi, C. F. Ahmed, M. Samiullah, and B.-S. Jeong. Effective periodic pattern mining in time series databases. *Expert Systems with Applications*, 40(8):3015–3027, 2013.
[23] B. Özden, S. Ramaswamy, and A. Silberschatz. Cyclic association rules. In *Proc. ICDE*, pages 412–421. IEEE, 1998.
[24] S. Parthasarathy, S. Mehta, and S. Srinivasan. Robust periodicity detection algorithms. In *Proc. CIKM*, pages 874–875. ACM, 2006.
[25] F. Rasheed, M. Alshalalfa, and R. Alhajj. Efficient periodicity mining in time series databases using suffix trees. *TKDE*, 23(1):79–94, 2011.
[26] S. Shang, D. Guo, J. Liu, and J.-R. Wen. Prediction-based unobstructed route planning. *Neurocomputing*, 213:147–154, 2016.
[27] C. Sheng, W. Hsu, and M. Li Lee. Mining dense periodic patterns in time series data. In *Proc. ICDE*, pages 115–115. IEEE, 2006.
[28] G. Sirisha, M. Shashi, and G. P. Raju. Periodic pattern mining–algorithms and applications. *Global Journal of Computer Science and Technology*, 13(13), 2014.
[29] H. Van Dongen, E. Olofsen, J. Van Hartevelt, and E. Kruyt. A procedure of multiple period searching in unequally spaced time-series with the lomb–scargle method. *Biological Rhythm Research*, 30(2):149–177, 1999.
[30] M. Vlachos, S. Y. Philip, and V. Castelli. On periodicity detection and structural periodic similarity. In *Proc. SDM*, volume 5, pages 449–460. SIAM, 2005.
[31] W. Wang, J. Yang, and P. S. Yu. Meta-patterns: revealing hidden periodic patterns. In *Proc. ICDM*, pages 550–557. IEEE, 2001.
[32] B. Xu, Z. Ding, and H. Chen. Mining multiple periods in event time sequence. In *Advances in Services Computing*, pages 278–288. Springer, 2015.
[33] J. Yang, W. Wang, and P. S. Yu. Infominer: mining surprising periodic patterns. In *Proc. KDD*, pages 395–400. ACM, 2001.
[34] J. Yang, W. Wang, and P. S. Yu. Infominer+: mining partial periodic patterns with gap penalties. In *Proc. ICDM*, pages 725–728. IEEE, 2002.
[35] J. Yang, W. Wang, and P. S. Yu. Mining asynchronous periodic patterns in time series data. *TKDE*, 15(3):613–628, 2003.
[36] K.-J. Yang, T.-P. Hong, Y.-M. Chen, and G.-C. Lan. Projection-based partial periodic pattern mining for event sequences. *Expert Systems with Applications*, 40(10):4232–4240, 2013.
[37] W. Yang and G. Lee. Efficient partial multiple periodic patterns mining without redundant rules. In *Proc. COMPSAC*, pages 430–435. IEEE, 2004.
[38] Q. Yuan, G. Cong, Z. Ma, A. Sun, and N. M. Thalmann. Time-aware point-of-interest recommendation. In *Proc. SIGIR*, pages 363–372, 2013.
[39] Q. Yuan, G. Cong, Z. Ma, A. Sun, and N. M. Thalmann. Who, where, when and what: discover spatio-temporal topics for twitter users. In *Proc. KDD*, pages 605–613, 2013.
[40] Q. Yuan, W. Zhang, C. Zhang, X. Geng, G. Cong, and J. Han. Pred: Periodic region detection for mobility modeling of social media users. In *Proc. WSDM*, pages 263–272. ACM, 2017.
[41] C. Zhang, J. Han, L. Shou, J. Lu, and T. F. L. Porta. Splitter: Mining fine-grained sequential patterns in semantic trajectories. *PVLDB*, 7(9):769–780, 2014.
[42] C. Zhang, K. Zhang, Q. Yuan, L. Zhang, T. Hanratty, and J. Han. Gmove: Group-level mobility modeling using geo-tagged social media. In *Proc. KDD*, pages 1305–1314, 2016.