Inferring the Periodicity in Large-Scale Internet Measurements

Oded Argon
School of Electrical Engineering
Tel-Aviv University, Israel

Yuval Shavitt
School of Electrical Engineering
Tel-Aviv University, Israel

Udi Weinsberg
Technicolor
Palo Alto, USA

***Abstract*—Many Internet events exhibit periodical patterns. Such events include the availability of end-hosts, usage of inter-network links for balancing load and cost of transit, traffic shaping during peak hours, etc. Internet monitoring systems that collect huge amount of data can leverage periodicity information for improving resource utilization. However, automatic periodicity inference is a non trivial task, especially when facing measurement "noise".**

In this paper we present two methods for assessing the periodicity of network events and inferring their periodical patterns. The first method uses Power Spectral Density for inferring a single dominant period that exists in a signal representing the sampling process. This method is highly robust to noise, but is most useful for single-period processes. Thus, we present a novel method for detecting multiple periods that comprise a single process, using iterative relaxation of the time-domain autocorrelation function. We evaluate these methods using extensive simulations, and show their applicability on real Internet measurements of end-host availability and IP address alternations.

I. Introduction

Human behavior often follows periodical patterns as a result of work habits, weekends and even yearly holidays. These patterns directly affect the way Internet resources are consumed, e.g., creating peak bandwidth hours, availability of hosts and resources, and mobility patterns. As a result, network operators often engineer their networks to accommodate these periodical changes in various ways.

For example, excessive traffic during peak hours may result in congestion, which impact user satisfaction. Network engineers commonly overcome this using two simultaneous links: a low cost link with sufficient capacity for most of the day, and a more expensive spill-over link with a usage based cost. Alternatively, it is now becoming increasingly common to perform traffic shaping during peak hours [1], [2].

Another example is the availability of end-hosts and their IP addresses assignment, the first is mostly determined by human habits, while the latter is often an engineered process of the serving ISPs [3]. Both have implications on peer-to-peer applications [4], online fraud detection [5], and on content distribution networks [6], that need to know which host is available and via which IP address it can be reached.

Although it is important to detect these periodical patterns and understand their effect on network resources, most patterns are not exposed by network operators, or not even deliberately engineered. Measurement efforts that attempt to discover and analyze them perform repeated measurements using various techniques, generate huge amounts of data, and post-process it for extracting insightful information. Such measurements can be viewed as a sampling process of the actual behavior. However, even though some periodical patterns are intuitive, detecting the samples that follow periodical patterns within large datasets is a non trivial task, particularly with the existence of intrinsic measurement "noise".

This paper presents efficient methods for detecting periodical patterns in large-scale Internet measurements data. We first convert the measurement data into a canonical signal, and then apply period inference methods for extracting its periodical patterns. We use a common frequency-domain method for robustly inferring a single dominant period. Due to its limitations when detecting multi-period signals, we present a novel iterative, yet a more time-consuming time-domain method for extracting all periods that comprise the signal.

A major challenge that does not exist in related frequency inference techniques [7], [8] is that our methods cannot assume that the signal is indeed periodic. Therefore, our methods first determine whether periodical patterns exist, and if they do, infer their period length.

The major contributions of this paper are as follows:

- We present the concept of periodicity in Internet measurement data, pointing out the difficulties of multiple period inference and noise factors.
- Using a signal processing technique called Power Spectral Density estimation on a signal constructed from the measurements, we show that it is most useful for detecting a single dominant period.
- We present a novel time-domain iterative method that is capable of robustly inferring all periods.
- We study the operation regimes of each method, focusing on network measurements data.
- Using real-world data, we detect multiple periods that align with human behavior and IP allocation patterns.

II. Signal Construction

The first phase is to construct a signal that represents the process being investigated. Consider a sequence S of N discrete samples, $S = \{s_1, \ldots, s_N\}$, where $s_i \in C$ and C is a set of possible values. We focus on two types of processes:

1) Dual-state processes, namely $|C| = 2$. Alternatively, processes may have multiple states which are quantized to two values.

2) A processes with multiple states, but we are interested in the point where the state changes and model this with two values that alternate at each state change.

The input samples S are converted to a canonical signal x_n, $\{x_1, \ldots, x_N | x_i = \pm 1\}$. For dual-state processes, C contains two possible values, $C = \{c_1, c_2\}$, making construction of x_n straightforward:

$$x_n = \begin{cases} 1 & s_n = c_1 \\ -1 & s_n = c_2 \end{cases} \quad (1)$$

For the alternating process, let $C = \{c_1, \ldots, c_K\}$, where K is the number of possible values. The signal x_n is represented using the same canonical notation, so that it keeps its value while the probe process contiguously samples the same value, and inverses when the sample results in a different value:

$$x_n = \begin{cases} 1 & \text{n=1,} \\ x_{n-1} & \text{if } s_{n-1} = s_n, \\ -x_{n-1} & \text{otherwise} \end{cases} \quad (2)$$

III. Process Properties

This section details the operation regions of our model and inference methods.

A. Number of Periods

The simplest classification of a process can be either periodic, e.g., with daily or weekly period, or non-periodic. However, some processes may exhibit multiple periods. For example, consider a cellphone tower that is next to a large corporate office [9]. During workdays the amount of traffic it carries exhibits a daily period during peak hours, while during weekends the traffic goes almost to zero. Although both patterns exist simultaneously, the weekly pattern is actually an "interference" to the daily period, because it creates imperfections in the daily pattern. The weekly pattern is perfect, unless the study is sufficiently long that it manages to include yearly patterns that 'harm' some instances of the weekly pattern, due to yearly holidays for example.

Fig. 1a depicts such a simulated signal, exhibiting a daily pattern (with non-symmetric duty-cycle), a weekly pattern, and a monthly pattern. Notice that the weekly patterns are observed due to a disturbance in the daily pattern (1 in every 7 days is different), and similarly, the monthly patterns are simply imperfections in the weekly pattern.

When multiple periods exist, the expected outcome is highly subjective. One may argue that the longest period (the monthly in the above example) is the most significant, because its periodical pattern is more perfect than the others. More commonly, the shortest period (the daily) may be considered more important, since it is the most dominant (contains the highest amount of "energy", in signal processing jargon) and already includes other periods (the weekly and monthly periods are harmonics of the daily period). Finally, one may want to infer all of the existing periods.

In either case, in order to be able to distinguish between periods, there must be a clear difference between them. For example, a yearly pattern with three days off in every year will be almost impossible to separate from a weekly pattern with two days off in every week.

We propose two methods: one for detecting the most reoccurring period using frequency domain analysis and a more complicated time-domain analysis for inferring all periods.

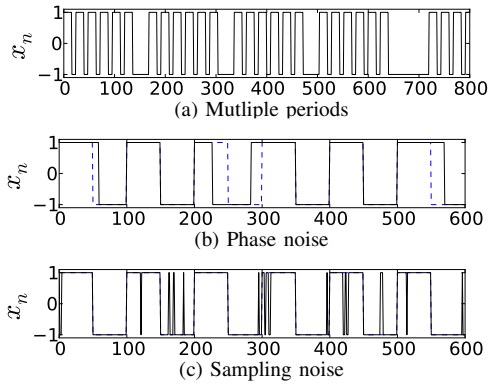


Fig. 1: Examples of x_n given phase noise, sampling noise and multiple periods

B. Duty-Cycle and Alternations

Two basic parameters of a periodic square signal are its *duty-cycle* and number of *alternations* per period. A simple signal has a single alternation, meaning it changes states only once per period. The duty-cycle of such a signal is the percent of time that the signal is in one state. A symmetric duty-cycle means that in each period the first half of the signal is one state and the other half it is in the second state.

The sampled process may have a non-symmetric duty-cycle, meaning that the change between states may occur anywhere within the period. This is common in human related behavioral patterns, for example, peak hours exhibit a daily pattern, but typically last only a few hours, making the duty-cycle not more than 0.25. Since we seek to find the periodicity of these processes, our methods make no assumption on the duty-cycle.

A perfect single-period signal (without noise) has a single alternation inside each period, i.e., a single zero-crossing. When noise exists, x_n may have more than one zero-crossing within a period, however, this should be filtered out by the inference methods. In signals with multiple periods, each period, except for the shortest one, is bound to have more than a single alternation. For example, Fig. 1a is comprised of two periods, a "short" one, which has a single alternation and a duty cycle of roughly 66%, and a "long" one that has multiple alternations and a completely non-symmetric duty-cycle: for each 6 repeats of the fast periodic pattern, it has a short duration of the fixed state "-1".

C. Noise Models

We include in our model two types of noise that are a common result of discrete sampling. The first type is when the sampling process exhibits jitter, i.e., it misses the exact time of a change that occurred in the sampled process. This

is common due to not frequent enough sampling, and causes x_n to have a delayed response to the real change. Since this delayed response is not likely to be consistent, x_n will exhibit variability in the periods lengths. Fig. 1b depicts such a signal, having cycles with wider or narrower periods than the real one (dashed lines).

We refer to this type of noise as *phase noise*, where the skewing of the phase in the resulting signal depends on the distance between the sampling and the actual event. Given that f_s is the sampling rate, assumed to be at least at Nyquist rate, i.e., twice the sampled frequency [10], the error in the period inference is at most $\pm 1/f_s$; $+1/f_s$ occurs when a sample is immediately after the real change and the following sample is right before the real change, thus missing until the next sample, and $-1/f_s$ occurs when a sample is right before the real change, thus missing it until the next sample, and the sample afterwards is immediately after the following change.

Phase noise can also be the result of jitter in the process itself. For example, the exact peak-hour time that causes a link to become congested is not consistent. Furthermore, the sampling process itself is often not accurate, and may exhibit different intervals between samples. The only important aspect to maintain is that the sample process is performed in at least Nyquist frequency, i.e., twice the frequency of the process, so that it does not misses actual changes [10].

The second type of noise occurs due to mistakes in the sampling, e.g., a sampling process of the load on a link incorrectly reported that the link is congested even though it was not. We refer to this type of noise as *sampling noise*.

The result of sampling noise on x_n differs depending on the sampled process. In dual-state processes, x_n will have wrong values for each wrong sample. We expect that only a few contiguous samples will be incorrect, thus the effect on x_n is local and relatively short, given a sufficiently high f_s. Fig. 1c provides an example to sampling noise (3% of the samples are wrong, up to two contiguous mistakes).

On the other hand, when sampling alternating processes, contiguous sampling mistakes may have a more global effect. If the incorrect sample resulted in a single value, then the result is a local noise in x_n, since right after the incorrect samples, the correct sample is made, and x_n returns to the correct form. However, if there were two (or any even number of) mistakes that resulted in two different incorrect values , then once returning to the correct value, x_n is inverted relative to what it would be without the mistakes. Contiguous sampling of two different and incorrect values should be a very rare case, and we assume that in the case of alternating signals, special care is taken to assure the accuracy of the sampling process, so that this case is avoided.

Notice that sampling noise is a special form of the common amplitude noise. When the sampling process experience an amplitude noise that is high enough for incorrect classification of the sampled value, it translates into a sampling noise according to our definition.

IV. Period Inference Methods

In this section we present two methods for inferring the periodicity of the sampled signal. The first method uses Power Spectral Density (PSD) estimation in the frequency domain for finding the most energetic period. We then present a novel method, which we call Mutliple Period Estimation (MPE), that infers all periods, using iterative partitioning of the peaks observed in the Autocorrelation Function (ACF).

PSD returns the inferred period, $\hat{P}$, and a confidence value ξ, that quantifies the probability that the signal is indeed periodic with the inferred period. In case of MPE, multiple pairs $(\hat{P}, \xi)$ are returned, one for each inferred period.

We note that intuitively, simple statistical inference methods can be applied. For example, it is possible to create a histogram of the times between alternations in x_n, and consider the peaks as representing half of the period. Such a method, however, assumes a duty-cycle of 0.5, and cannot capture multiple periods. Furthermore, averaging and smoothing is required for the method to handle noise well. Thus, we use techniques that are more complicated, but are known to have good properties for our problem domain.

A. Method A: Power Spectral Density

PSD is a method for estimating the power that each frequency of a signal holds (power spectrum). The basis for spectral density estimation of a signal x_n is the Discrete Fourier Transform (DFT), that converts the time-domain signal into the frequency domain.

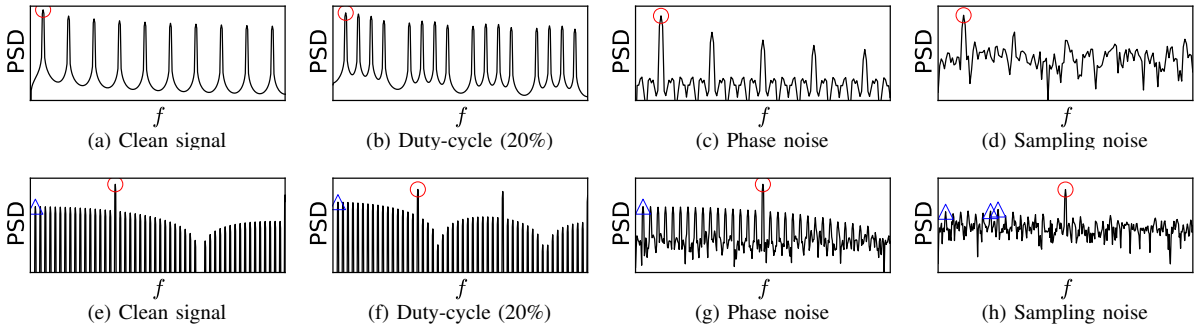
Before applying DFT, we normalize the signal in order to remove any DC (corresponding to zero frequency) artifacts. This is particularly important for signals with non-symmetric duty-cycle, that have a non-zero mean. Let μ denote the mean value of x_n, i.e., $\mu = \frac{1}{N}\sum x_n$, we normalize the signal $\hat{x}_n$ using $\hat{x}_n = x_{n+1} - \mu,\ n = 0, ..., N-1$. Notice we also shifted the signal to make it zero-based, allowing simpler DFT computation. The DFT of $\hat{x}$ is then computed using:

$$X_k = \sum_{n=0}^{N-1} \hat{x}_n e^{-2\pi k n i/N} \qquad k = 0, 1, ..., N-1 \tag{3}$$

The power of each frequency is computed simply using the squared amplitude of each complex component in the DFT. For computing the PSD, we apply Welch's average method, a method that uses segmentation, windowing and averaging for improving the statistical properties of the resulting spectral estimates [11]. Using PSD, the computation of the *fundamental frequency* of the signal, which is the one that holds the most energy, is straightforward – the frequency that matches the highest peak. We use it for inferring the period (inverse of the frequency) of the signal by computing:

$$\hat{P} = \left(\arg\max_k \{|X_k|\} \cdot \frac{f_s}{N} \right)^{-1} \tag{4}$$

PSD provides all the frequencies that comprise the signal, including their harmonics (multiplications). However, since our signal is not a composition of periodical sine waves, but



(a) Clean signal (b) Duty-cycle (20%) (c) Phase noise (d) Sampling noise

(e) Clean signal (f) Duty-cycle (20%) (g) Phase noise (h) Sampling noise

Fig. 2: Power Spectral Density (PSD) functions of a simulated signal with single period (a–d) and two periods (e–f)

rather a noisy square wave, the signal energy is spread across many frequencies. Theoretically, the "interesting" periods can be inferred by iteratively selecting the highest peak with a frequency smaller than the last detected peak, since higher frequencies are a result of harmonics of already inferred periods or noise. However, when looking for secondary peaks, even low levels of noise result in peaks that are indistinguishable from those we seek to find.

Fig. 2 shows the PSD of a signal with a single period (top plots, 100 samples per cycle, 15 cycles) and a signal with two periods (bottom plots, zoomed, second period is 10 cycles of the first period, with added 100 samples of -1 between each cycle). For each type of signal, the figure shows its PSD with no changes, with non-symmetric duty-cycle (20%), with added phase noise (10% of alternations, jitter of at most 2 samples) and with added sampling noise (10% of the samples, at most 2 incorrect samples).

In all plots the max peak (marked with a red circle) that corresponds to the fundamental frequency of the signal is easily inferred, regardless whether noise exists. Fig. 2e shows that two periods and no noise are correctly detected (the red circle and blue triangle), and Fig. 2f shows robustness to duty-cycle asymmetry, which is a result of the signal normalization (note that all other higher-frequency peaks are harmonics of the two inferred periods).

However, Fig. 2c and Fig. 2d show that phase noise and sampling noise result in a significant amount of secondary peaks, rendering separation between real periods and noise almost impossible (e.g., Fig. 2d vs. Fig. 2h). Moreover, as a result of noise, the harmonics of secondary periods exhibit higher peaks than the matching fundamental frequency, making the inferred period incorrect (e.g., a bi-daily period is detected instead of the correct daily period).

Given the above, we use PSD for the detection of a single period, a task that suits many network monitoring applications. Since it is easily and efficiently implemented (using Fast Fourier Transform), this method is quite useful and, as we show in Sec. V, is very robust to noise.

Computing the period confidence, ξ, is done by summing the energy of the inferred frequency and its harmonics (since the energy of the frequency is divided amongst all harmonics). We then normalize it using the energy of the complete signal. Let k be the index of the peak that corresponds to the inferred period $\hat{P}$, we denote by M the set of harmonics of $\hat{P}$, i.e., $M = \{n \cdot \hat{P}\},\ n = 1..\lfloor \frac{N}{k} \rfloor$. We then compute ξ using:

$$\xi = \frac{\sum_{m \in M} |X_m|^2}{\sum_n |X_n|^2} \tag{5}$$

When multiple peaks are detected, it can either be a result of noise or existence of multiple periods. In this case we perform the method described next, which is capable of extracting all periods that comprise the signal.

B. Method B: Multiple Period Estimation

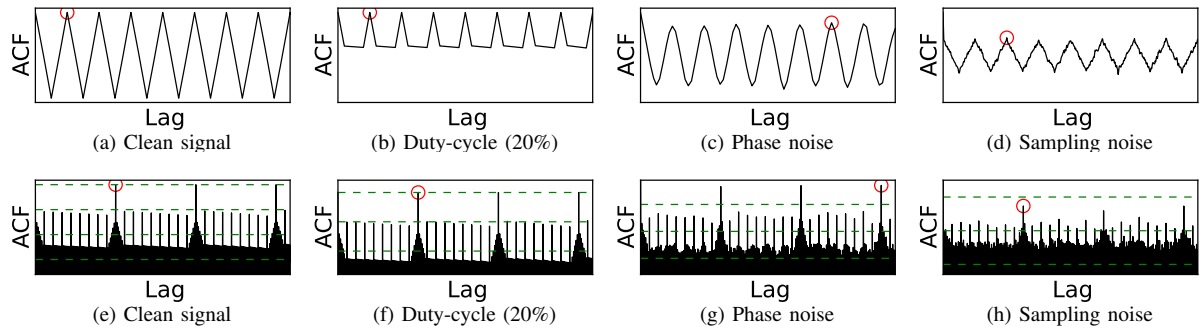
Similar to DFT, the autocorrelation function (ACF) is an averaging method, only it operates in the time domain. ACF measures how well a signal is correlated with a shifted version of itself. More formally, the normalized ACF of a discrete signal x_n can be defined as:

$$R_n = \frac{\sum_{m=1}^{N} x_m x_{m-n}}{N-n}, \quad n = 0..N-1 \tag{6}$$

where R_n is the normalized ACF of lag n. Since we only use this form of normalized ACF in the paper, we refer to it simply using the term ACF. For periodic signals, the ACF is periodic with the same period.

Notice that although the ACF is normalized high shifts (large n) use only a small portion of the signal, thus are less accurate than low shifts, indicating the importance of a sufficiently long signal. We further evaluate this issue in Sec. V, and show that MPE is robust to the signal length.

A key strength of ACF, that makes it useful for finding repeating patterns, is that it smooths both sampling and phase noise, since these types of noise affect only small sections of the signal. Fig. 3 plots the ACFs of a signal with a single period (upper plots) and a signal with three periods (lower plots), each with different types of noise and duty-cycle. In the single period plots, the periodic pattern is clearly visible. Fig. 3c shows that phase noise causes the ACF to lose its linearity, while sampling noise, depicted in Fig. 3d lowers the peak value. The non-symmetric 20% duty-cycle in Fig. 3b cuts the lower parts of the ACF, since there is no lag that results in an "inverted-phase", which causes the negative peaks in a 50% duty-cycle signal. However, the periodical pattern in all variations is still clear.



(a) Clean signal (b) Duty-cycle (20%) (c) Phase noise (d) Sampling noise

(e) Clean signal (f) Duty-cycle (20%) (g) Phase noise (h) Sampling noise

Fig. 3: Autocorrelation functions of a simulated signal comprised of a single period (a–d) and three periods (e–f)

ACF by itself is commonly used for inferring periodicity, e.g., inferring the pitch of musical and human speech signals, however it is still known to be unreliable [8]. For example, consider the round markers in Fig. 3, depicting the maximal peak, showing that different maximal peaks are selected, corresponding to different inferred periods.

Instead, we extend the usage of ACF for extracting multiple periods that comprise the signal. The basis is the observation that different periods have different peak levels in the ACF, while peaks belonging to the same period have roughly the same value. Looking at the bottom plots in Fig. 3 (depicting only a portion of the signal), there is an obvious separation between 3 different levels of peaks (the dashed horizontal lines were added as illustrative aid); the top ones correspond to the longest period, which is the most "perfect", the mid-peaks correspond to a shorter period and the bottom region (that are actually peaks) correspond to the shortest period (that has imperfections due to the longer periods). The reason is that the more perfect a period is, the higher the corresponding peaks will be, in all the shifts that match the period.

Consider the following strict definition of a periodic signal with period τ:

$$\exists \tau,\ s.t.\ \ \forall t,\ \ f(t) = f(t+\tau) \tag{7}$$

which holds when there is a single period and no noise. Whenever multiple periods exist or there is noise in the signal, we need to relax three aspects of this definition. First, the equality should be for peaks that belong to the same period. Second, $f(t)$ and $f(t+\tau)$ needs to be only roughly the same, and not precisely equal. Third, τ, which represents the distance in lags between peaks, does not have to be precise, but can vary (to some extend) between different peaks.

Alg. 1 lists the pseudo-code of MPE. First, accounting for the separation of periods and relaxing the equality of $f(t)$ and $f(t+\tau)$, MPE partitions the ACF peaks into *slices* (line 4), so that each slice contains peaks belonging to different periods. Since we do not know apriori how to slice the ACF, this is an iterative process, trying each time a coarser partitioning. Accounting for the variations in τ, MPE computes, for each slice that has a sufficient number of peaks, a histogram (PDF kernel [12]) of the intervals (*gaps*) between peaks (lines 6–10). If there is a significant mode (higher than the given probability

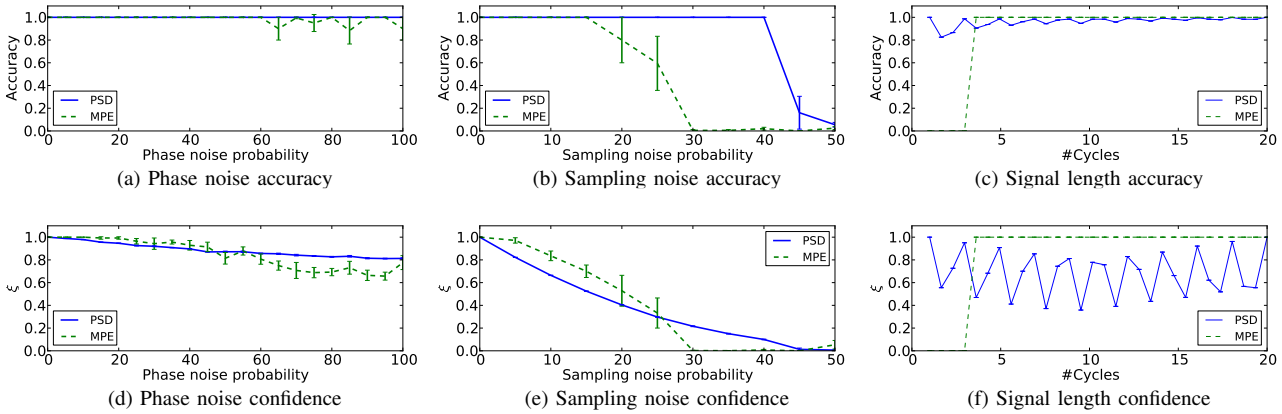
Algorithm 1 Multiple Period Estimation (MPE) algorithm

Input: ACF of a discrete signal x_n
Output: A set of $(\hat{P}, \xi)$ for each inferred period

1: $scale \leftarrow MAX\_SLICES$, $periods \leftarrow \emptyset$
2: **while** $scale > 0$ **do**
3: Partition the ACF y-axis to $scale$ equal-size slices
4: **for** slice **in** slices **do**
5: Find the ACF peaks within the slice
6: $N \leftarrow$ number of peaks within the slice
7: **if** $N \geq MIN\_PEAKS$ **then**
8: Compute the gaps between the peaks
9: Compute the gaps PDF with width = $1/f_s$
10: $G \leftarrow$ tallest mode in the gap PDF
11: **if** probability of $G > MIN\_PROB$ **then**
12: $p \leftarrow \left(G \cdot f_s^{-1}\right)$
13: $gaps \leftarrow$ number of gaps in G
14: $e_{gaps} \leftarrow \min(1, \lceil signal\_length/p \rceil - 1)$
15: $\xi^* \leftarrow gaps/e_{gaps}$
16: **if** $p \notin periods$ **then**
17: $periods \leftarrow periods \bigcup (p, \xi)$
18: **else if** previous ξ is smaller than ξ^* **then**
19: replace existing ξ with ξ^*
20: **if** all peaks are in the same slice **then**
21: break
22: $scale \leftarrow scale - 1$
return $periods$

MIN_PROB), then it is considered a valid period (lines 12–20). If all signal peaks fell into the same slice, then the algorithm terminates (lines 21–22). Otherwise, it repeats the above process for a coarser partitioning. For each inferred period, its confidence, ξ, is calculated by counting the number of gaps that fall into the tallest mode bin, and normalizing it by the number of expected gaps in a perfect signal with the inferred period (lines 13–16). This can later be refined if a different partitioning result in more peaks per slice (lines 19–20). In a perfect signal, all of the peaks that correspond to a given period would fall in the same bin, thus the resulting ξ will be one. When noise or multiple periods exist, the peaks may shift between slices, hence ξ will be lower than 1.

Looking back at the bottom plots of Fig. 3, it is possible to see that MPE can infer three different periods, by detecting peaks in different slices (marked by the dashed lines), even when noise exists.



(a) Phase noise accuracy (b) Sampling noise accuracy (c) Signal length accuracy

(d) Phase noise confidence (e) Sampling noise confidence (f) Signal length confidence

Fig. 4: Simulation results of increasing phase noise, sampling noise and number of sampled periods

Parameters. MPE requires setting several parameters that affect its period detection ability and inference mistake. The resolution of slicing the peaks (MAX_SLICES) is a trade-off between the ability to separate similar periods and the robustness to noise. Fine partitioning has the ability to distinguish periods that are very similar, but makes the noise margins smaller. Notice that this parameter does not imply that the number of expected periods needs be known a-priori; it only needs to be higher than the number of periods that are expected to be detected.

Similarly, the width of the gap PDF bins (line 10) countermeasures noise but also determines the mistake that is introduced to the inferred period. Small bins improve accuracy but are susceptible to noise and also result in lower confidence values because possibly a few gaps are contained in the detected mode. We use a bin width of $1/f_s$, which already encapsulates the mistake in the inferred period – higher sampling rate, implies lower inference mistake. By using the sampling rate as the bin width, we ensure that the period inference mistake is at most the mistake already introduced by the sampling process.

MIN_PROB and MIN_PEAKS determine the ability to detect periods that do not exhibit a clearly dominant gap. Setting a low value for these parameters can help detect noisy periods, however, it may also result in inferring incorrect periods that managed to pass the thresholds.

Efficiency. The most time consuming task in MPE is the computation of the ACF, which is naively computed in $O(N^2)$, or more efficiently in $O(NlogN)$ using FFT. Additional improvements can be made if the signal is significantly longer than the expected period. In such a case, only a portion of the signal that matches the length of the expected period should be autocorrelated. MAX_SLICES also impacts the running time, affecting both the number of iterations, and the number of PDFs per iteration. However, since the number of expected periods is commonly low, we expect MAX_SLICES to be in roughly the same order of magnitude.

V. Simulation

In this section we evaluate the results of the methods on synthetic signals. We first compare the two methods for signals with a single period, and evaluate their performance when facing noise. We then study the ability of MPE to detect multiple periods and explore its operation limits.

A. Simulating Noise

Recall that we consider two types of noise – phase and sampling noise. Simulating phase noise is achieved by varying the exact time of alternations (zero crossings) in x_n. To this end, we define Pr_{PH} as the probability of a zero-crossing to suffer phase noise and N_{PH} as number of samples relative to the selected sample, that the zero-crossing should be move to.

Similarly, simulating sampling noise is achieved by selecting random samples with uniform probability Pr_{SM} at which the sampling mistake is performed, and inverting the value for N_{SM} contiguous samples.

We perform separate simulations for each type of noise, by varying its probability. We set N_{PH} and N_{SM} to use normal distributions, and repeat each simulation 10 times.

B. Single Period Estimation

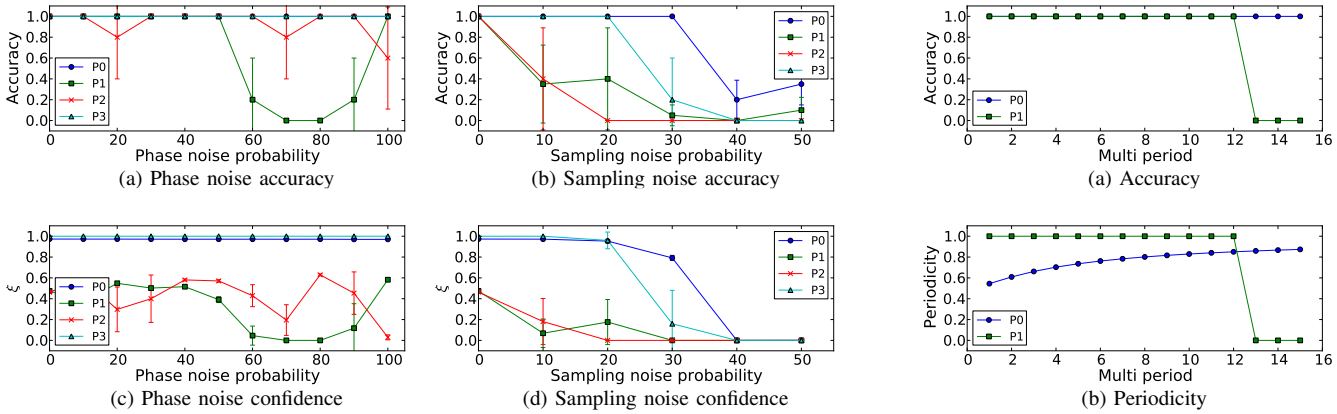
Denote by P the period we seek to infer, and $\hat{P}$ the inferred period. We define the accuracy of the inferred period as:

$$Accuracy = 1 - \frac{|\hat{P}-P|}{\max\{\hat{P},P\}} \tag{8}$$

An accuracy of 1 indicates that there is no mistake. As the mistake increases the accuracy goes down to zero. This definition aligns with that of the confidence ξ, where 1 is most confident and the value decays as the confidence is lower.

The simulated signal is built of 15 cycles with a period $P = 100$ (total length of 1500 samples). We first validated that changing the duty-cycle of the signal has no effect on the algorithm results, and found that both PSD and MPE result in accurate inference and a perfect confidence (both equal to 1).

When simulating noise, we use a symmetric duty-cycle (50%) and set $N_{PH} \sim NORM(5,1)$ and $N_{SM} \sim NORM(1,0)$



(a) Phase noise accuracy (b) Sampling noise accuracy

(c) Phase noise confidence (d) Sampling noise confidence

Fig. 5: Simulation results of MPE accuracy and confidence showing phase noise and sampling noise with 4 periods

(a) Accuracy

(b) Periodicity

Fig. 6: Simulation results of period ratio using 2 periods signal

(at most 1 incorrect sample). We note that other noise models can also be utilized, and we plan to study the effect of more correlated noise models in future work. Fig. 4 plots the inference error and confidence for varying Pr_{PH}, Pr_{SM} and signal length. The vertical error bars illustrate the variance. Fig. 4a shows that the phase noise has very little effect on the accuracy of both methods, with PSD being completely robust to it. The confidence of both, depicted in Fig. 4d, decreases as the phase noise increases, but remains mostly above 0.5.

As expected, sampling noise has a greater impact on both methods. Fig. 4b shows that MPE is affected above 20%, whereas PSD is more robust, starting to exhibit lower accuracy only above 40% of noise. The confidence, shown in Fig. 4e, also exhibits relatively low values for both methods. PSD confidence is low because the sampling noise spreads the energy into many different frequencies, hence the overall energy of the harmonics is low. Similarly, MPE suffers from peak gaps "falling" into different bins in the PDF, thus the number of peaks in the same highest mode becomes lower as the noise increases. If sampling noise is known to exist in the dataset, the confidence can be improved by increasing the bin width, but resulting in a lower inference accuracy.

The robustness of the methods to the signal length is shown in Fig. 4c and Fig. 4f. Fig. 4c shows that both methods result in an accurate inference. MPE starts with zero accuracy due to the value of MIN_PEAKS, mandating sufficient periods before detecting a period as valid. PSD exhibits a chainsaw pattern because the computation of the period depends on whether the signal length is a complete multiplication of the period. Thus, when complete multiplications of periods are sampled, the value is perfectly correct.

Fig. 4f shows that MPE results in a perfect confidence, regardless of the length. PSD exhibits a similar chainsaw pattern because when the inferred period is slightly incorrect, the harmonics are not aligned with that period, thus their energy is not accumulated, resulting in a low confidence. In any case, the value is above 0.3 at all times, thus we use 0.3 as a threshold for the confidence in the evaluation in Sec. VI.

C. Multiple Period Estimation

We evaluate MPE's ability for inferring multiple periods, by constructing a signal with 4 periods – daily, weekly, monthly and yearly periods. A dominant gap is selected with $MIN\_PROB = 0.5$, and the finest slicing resolution is $MAX\_SLICES = 10$. Under these typical settings, the values enable sufficient separation and robustness to noise, while extracting periods with clear dominance.

Fig. 5 shows the accuracy and confidence for each resulting period (P0 being the shortest and P3 the longest), when facing increasing phase noise and sampling noise. Fig. 5a shows that MPE is robust to phase noise, except for the weekly period (P1). Note that when the phase noise is very high the accuracy improves again, which is a result of the signal completely inverting its phase. Fig. 5c shows that the confidence of the two extreme periods ($P0$ and $P3$) is high, while the two middle periods ($P1$ and $P2$) is relatively low. The baseline confidence of the latter two is 0.5 since they have ACF peaks that reside in several slices, thus even though the accuracy is high, the confidence is relatively low, further indicating that a relatively low confidence value is sufficient to get accurate results.

Fig. 5b and Fig. 5d show that sampling noise has an even larger impact on the results when multiple periods need to be detected. However, since the confidence drops rapidly with the accuracy (reaching almost 0), it is clear which periods can be trusted and which cannot. Overall, when multiple periods need to be detected, the sampling process should be relatively accurate so that sampling mistakes are not common.

Finally, we measure the effect of the ratio between periods on MPE's results. To this end, we simulate a signal with 2 periods, $P0$ and $P1$, $(P0 < P1)$, and change their ratio by increasing the number of cycles of $P0$ for each appearance of $P1$. Fig. 6a shows that the two periods are correctly inferred (the confidence resulted in the exact same values), until reaching 13 cycles of $P0$, which causes $P1$ to be completely undetected by MPE.

In order to understand these results, we introduce a *Peri-*

oditicy parameter, which is the average of the peak values that correspond to the selected bin in the gap PDF (notice that all these peaks come from the same slice). This value captures how "perfect" the period is, i.e., a high peak value (close to 1) implies very few interruptions in the ACF, while low values indicate that the periodicity is interrupted. Fig. 6b shows that the periodicity of $P0$ starts in a low value, since for every other cycle, it is interrupted by $P1$. However, as the ratio between periods increases, the periodicity of $P0$ increases, i.e., their peak values raise. Once the peak values of $P0$ surpasses 0.8, the peaks shift into the bin of $P1$, making both periods look like a single period, thus $P1$ is not detected as a separate period. Increasing MAX_SLICES can provide better separation between the periods (but at a cost of performance and smaller noise margins). For example, setting $MAX\_SLICES = 15$, produces the two periods until 17 cycles of $P0$ for every $P1$ cycle.

VI. Evaluation

We evaluate our methods on two real-world Internet processes that capture the dynamics of end-hosts – the availability of end-hosts and the alternation of allocated IP addresses. These patterns has implications on various network applications. For example, malicious host identification [13], network forensic analysis and blacklisting require tracking infected hosts over time using their IP addresses [14], [15], [3].

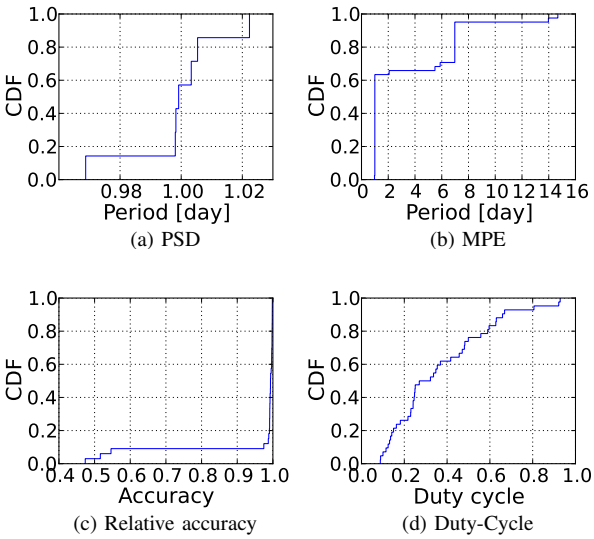
A. Dataset

The dataset for evaluation is obtained from passive sampling of the measuring hosts of DIMES [16], a community-based Internet measurements system. DIMES utilizes hundreds of software agents installed on user PCs, each has a unique ID, which is associated with the machine it is installed on.

When a machine is online and connected to the Internet, its agent performs a set of measurement scripts and reports the results back to a central server. These results, alongside with the routable IP address of the machine are reported roughly every 30 to 60 minutes. This variance is a result of the length of the measurement scripts, or due to short term network, end-host or server failures.

We used data collected over three months in 2010. Overall, 1804 agents provided 8.6 million reports, using 7611 different routable IPs from 432 unique autonomous systems. Using this dataset, we build two datasets for evaluation – *Availability* and *Alternations*. The availability dataset tracks whether an agent's machine is online or not. We consider an agent as "offline" after 3 hours has passed since its last report. The alternation dataset tracks the changes in routable IP addresses of the agents during its "online" time-frames, henceforth *windows*.

We carefully filter this data to reduce various measurement artifacts. If an agent exhibits more than 20 IP alternations in a given online window, we discard its data, since we empirically found that this it is most likely a measurement artifact. We also filter windows with IPs spanning multiple ASes, since our goal is to capture the alternations in IP addresses allocated to the agent by its ISP, and not human mobility. After applying these filters 1784 agents remain.



(a) PSD (b) MPE (c) Relative accuracy (d) Duty-Cycle

Fig. 7: Cumulative distribution of availability periods

Constructing the canonical signal from the availability dataset is straightforward, marking online windows as 1 and offline as -1. For the alternation dataset, the first IP we observe for an agent is marked as 1, and whenever we encountered a change in the IP address we inverse the sign of the signal. Both datasets exhibit low levels of phase-noise due to small variations in the intervals between agent reports. The alternation dataset also exhibits a few sampling mistakes, which result from rare measurement artifacts [16].

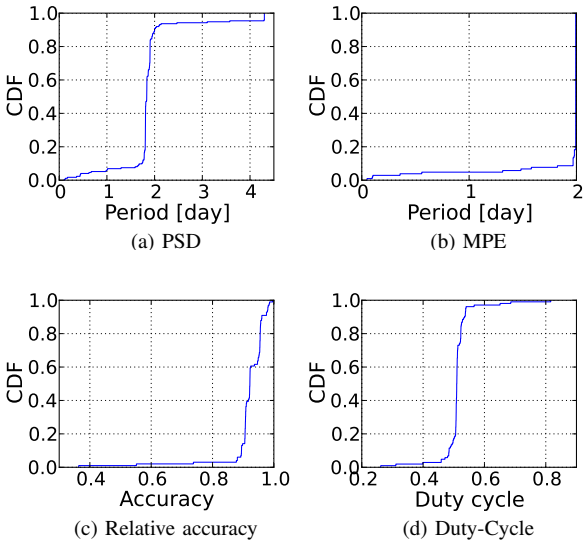
B. Results

We ran the PSD and MPE on both datasets. We consider an agent as periodic when we detect periods with $\xi > 0.3$, within signals that contain at least 4 cycles, with the latter being imposed to further increase the confidence of the findings.

Availability. Fig. 7 depicts the results of applying the methods on the availability dataset. Using PSD we found 82 agents that exhibit periodical patterns and using MPE we found 51. Fig. 7a shows that PSD inferred a daily pattern with relatively small mistake. MPE shown in Fig. 7b managed to detect weekly patterns (7 days) and even a few bi-weekly patterns (14 and 15 days). We note that these weekly and bi-weekly patterns are secondary periods, i.e., each of the agents that exhibited one of them also had a daily pattern. Fig. 7c plots the relative accuracy of PSD and MPE (using Eq. 8), and shows that the two methods agree on over 90% of the periods.

For inferring the duty-cycle we simply count the amount of online vs. offline time in signals of agents that exhibit periodical patterns. Fig. 7d shows a wide range of duty-cycles, which is the result of capturing different behaviors of agents.

Alternations. Using the alternations dataset, we found 174 agents that exhibit a periodical pattern using PSD and 131 agents using MPE. Fig. 8 shows that MPE resulted in a perfect 2 days period, whereas PSD resulted in slightly less accurate inference of roughly 2 days period. Thus, the resulting relative accuracy, shown in Fig. 8c, is mostly above 0.9. The inferred



(a) PSD (b) MPE

(c) Relative accuracy (d) Duty-Cycle

Fig. 8: Cumulative distribution of IP alternation periods

duty-cycle shown in Fig. 8d is 0.5 for almost 90% of the agents, meaning that their IP address is replaced roughly every day, which is a common DHCP default lease time [4].

This evaluation shows the strengths of PSD and in particular of MPE for analyzing periodical patterns in large-scale measurements. Without imposing any assumptions on the dataset, the methods were able to both detect the agents that exhibit periodical patterns, and extract the corresponding periodicity. MPE was even able to recover multiple periodicities that exist in the dataset, which enables better understanding of the underlying patterns.

VII. Related Work

Monitoring networks and behavioral patterns is a key aspect of network management. Thompson *et al.* [17] measured two OC-3 trunks for 7 days and observed a daily period with varying duty-cycles in the volume of bytes, number of flows, number of packets, TCP traffic, etc. Willkomm *et al.* [9] studied datasets of a cellular network operator, exhibiting a clear daily load periodical pattern. Leland *et al.* [18] studied the self-similarity of Ethernet traffic, and showed daily cycles in some of their datasets.

Signal processing was previously used in network applications, e.g., wavelets in flow and SNMP data [19], spectral density in wireless networks [20], denial of service attacks [21], and in bottleneck detection [22]. These methods commonly focus on a single period. A method that uses iterative refining of PDF kernel estimation was presented in [23], however, in the context of detecting multiple congested links along a path.

Inference of multiple periods exists in music and voice signals [8] and emitter positioning [7], however, these usually operate on small datasets and assume domain-specific signal and noise parameters, making them highly tailored to their tasks. In this paper we studied and proposed robust methods that are suited to network measurements applications and are efficient so that they can be applied to large-scale data.

VIII. Conclusion

Understanding behavioral patterns in the vast amounts of data originating from Internet monitoring systems is a key challenge for making better use of resources and designing better systems. This paper presents two methods for detecting and inferring such periodical patterns and shows their ability to both detect periodical patterns and infer their periodicities without imposing assumptions on the obtained samples.

Although some of these results may seem intuitive, e.g., a work-day pattern of availability, applications that care about periodical patterns, cannot easily detect these patterns. The amount of data renders any manual detection impossible, and the inherent measurements noise makes simple comparison methods inaccurate, particularly in the existence of multiple periods. The proposed methods overcome these issues and perfectly fit noisy measurements data.

References

[1] P. Kanuparthy and C. Dovrolis, "DiffProbe: Detecting ISP service discrimination," in *Infocom*, 2010.
[2] Y. Zhang, Z. M. Mao, and M. Zhang, "Detecting Traffic Differences in Backbone ISPs with NetPolice," in *IMC*, 2009.
[3] Y. Xie, F. Yu, K. Achan, E. Gillum, M. Goldszmidt, and T. Wobber, "How dynamic are IP addresses?" in *SIGCOMM*. ACM, 2007.
[4] M. Khadilkar, N. Feamster, M. Sanders, and R. Clark, "Usage-based DHCP lease time optimization," in *IMC*. ACM, 2007.
[5] C. Walgampaya, M. Kantardzic, and R. Yampolskiy, "Real time click fraud prevention using multi-level data fusion," in *WCECS*, 2010.
[6] V. Valancius, N. Laoutaris, L. Massoulié, C. Diot, and P. Rodriguez, "Greening the Internet with Nano Data Centers," in *CoNEXT*, 2009.
[7] R. Schmidt, "Multiple emitter location and signal parameter estimation," *IEEE Trans. on Antennas and Propagation*, vol. 34, no. 3, Mar. 1986.
[8] A. de Cheveigné and H. Kawahara, "YIN, a fundamental frequency estimator for speech and music," *The Journal of the Acoustical Society of America*, vol. 111, no. 4, pp. 1917–1930, 2002.
[9] D. Willkomm, S. Machiraju, J. Bolot, and A. Wolisz, "Primary users in cellular networks: A large-scale measurement study," in *DySPAN*, 2008.
[10] R. J. Marks, II, *Introduction to Shannon sampling and interpolation theory*. Springer-Verlag New York, Inc., 1991.
[11] P. Welch, "The use of fast fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms," *IEEE Trans. on Audio and Electroacoustics*, 1967.
[12] D. Scott, *Multivariate Density Estimation*. John Wiley, 1992.
[13] Y. Zhao, Y. Xie, F. Yu, Q. Ke, Y. Yu, Y. Chen, and E. Gillum, "Botgraph: large scale spamming botnet detection," in *NSDI*. USENIX, 2009.
[14] A. Ramachandran and N. Feamster, "Understanding the Network-Level Behavior of Spammers," in *SIGCOMM*. ACM, 2006.
[15] C. Wilcox, C. Papadopoulos, and J. Heidemann, "Correlating spam activity with IP address characteristics," in *GI*, 2010.
[16] Y. Shavitt and E. Shir, "DIMES: Let the internet measure itself," *ACM SIGCOMM CCR*, vol. 35, no. 5, pp. 71–74, 2005.
[17] K. Thompson, G. J. Miller, and R. Wilder, "Wide-area internet traffic patterns and characteristics," *IEEE Network*, pp. 10–23, 1997.
[18] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson, "On the self-similar nature of ethernet traffic (extended version)," *IEEE/ACM Trans. on Networking*, vol. 2, no. 1, pp. 1–15, 1994.
[19] P. Barford, J. Kline, D. Plonka, and A. Ron, "A signal analysis of network traffic anomalies," in *IMW*, 2002.
[20] C. Partridge, D. Cousins, A. W. Jackson, R. Krishnan, T. Saxena, and W. T. Strayer, "Using signal processing to analyze wireless data traffic," in *WiSE*, 2002.
[21] A. Hussain, J. Heidemann, and C. Papadopoulos, "A framework for classifying denial of service attacks," in *ACM SIGCOMM*, 2003.
[22] X. He, C. Papadopoulos, J. Heidemann, U. Mitra, and U. Riaz, "Remote detection of bottleneck links using spectral and statistical methods," *Computer Networks*, vol. 53, 2009.
[23] S. Katti, D. Katabi, C. Blake, E. Kohler, and J. Strauss, "Multiq: Automated detection of multiple bottleneck capacities along a path," in *IMC*. ACM, 2004, pp. 245–250.