Noname manuscript No.
(will be inserted by the editor)

On solving the SPL problem using the concept of Probability Flux

Asieh Abolpour Mofrad · Anis Yazidi · Hugo Hammer

Received: date / Accepted: date

Abstract The Stochastic Point Location (SPL) problem [20] is a fundamental learning problem that has recently found a lot of research attention. SPL can be summarized as searching for an unknown point in an interval under faulty feedback. The search is performed via a Learning Mechanism (LM) (algorithm) that interacts with a stochastic Environment which in turn informs it about the direction of the search. Since the Environment is stochastic, the guidance for directions could be faulty. The first solution to the SPL problem, which was pioneered two decades ago by Oommen, relies on discretizing the search interval and performing a controlled random walk on it. The state of the random walk at each step is considered to be the estimation of the point location. The convergence of the latter simplistic estimation strategy is proved for an infinite resolution, i.e., infinite memory. However, this strategy yields rather poor accuracy for low discretization resolutions. In this paper, we present two major contributions to the SPL problem. First, we demonstrate that the estimation of the point location can significantly be improved by resorting to the concept of *mutual probability flux* between neighboring states along the line. Second, we are able to accurately track the position of the optimal point and simultaneously show a method by which we can estimate the error probability characterizing the Environment. Interestingly, learning this error probability of the Environment takes place in tandem with the unknown location estimation. We present and analyze several experiments discussing the weaknesses and strengths of the different methods.

Asieh Abolpour Mofrad
Dept. of Computer Science, OsloMet - Oslo Metropolitan University
Oslo, Norway.
E-mail: asieh.abolpour-mofrad@oslomet.no

Anis Yazidi
Dept. of Computer Science, OsloMet - Oslo Metropolitan University
Oslo, Norway.
E-mail: anis.yazidi@oslomet.no

Hugo Hammer
Dept. of Computer Science, OsloMet - Oslo Metropolitan University
Oslo, Norway.
E-mail: hugo.hammer@oslomet.no

Keywords Stochastic Point Location (SPL) · Mutual Probability Flux · Flux-based Estimation Solution (FES) · Last Transition-based Estimation Solution (LTES) · Stochastic Learning Weak Estimation (SLWE) · Estimating Environment Effectiveness

1 Introduction

Stochastic Point Location (SPL) is a fundamental optimization problem that was pioneered by Oommen [20] and ever since has received increasing research interest [28, 12]. A Learning Mechanism (LM) attempts to locate a unique point λ^* in an interval whilst the only assistance comes from the information provided by a random Environment (E) which informs it, possibly erroneously, whether the location is to the left or to the right of the point. The probability of receiving correct response from Environment is basically fixed and unknown. The SPL problem, which was addressed by Oommen and few others [20, 28, 12, 14, 22, 21], is indeed a general optimization framework where a large class of optimization problems could be modeled as an instantiation of it, see [31] for a survey of all the reported solutions to the SPL.

The assumption that the parameter or point location in the SPL setting does not change over time is not the case in many real-life dynamic systems such as web-based applications [11]. Indeed, the probability of receiving correct response from Environment might be unknown and even non-stationary. Sliding window [13] is a traditional strategy for estimation in non-stationary Environments. However, choosing the appropriate window size would be crucial. When the window size is too small, the estimation will be poor. Contrarily, if the window size is rather large, the estimation will be degraded.

It is worth mentioning that Continuous Point Location with Adaptive Tertiary Search (CPL-ATS) strategy [23] is another method of solving SPL which systematically and recursively searches for sub-intervals that λ^* is guaranteed to locate in, with an arbitrarily high probability. A series of guessing which starts with the mid-point of the given interval estimates the point location and repeats until the requested resolution is achieved. The given interval is partitioned into three sub-intervals where three LA work in parallel in each sub-interval and at least one of them will be eliminated from further search. So, it is crucial in CPL-ATS to construct the partition and elimination process. This method is further developed into the CPL with Adaptive d-ary Search (CPL-AdS) Strategy [24] where the current interval is partitioned into d sub-intervals, instead of three. The larger d results in faster convergence, but the decision table of elimination process becomes more complicated. An extension of the CPL-AdS scheme, which could also operate in non-stationary environments, is presented in [12]. The decision formula is proposed to modify the decision table in [24] to resolve certain issues of original CPL-AdS scheme.

In [35] an SPL algorithm based on Optimal Computing Budget Allocation (OCBA), named as SPL-OCBA, is proposed. SPL-OCBA employs OCBA and the historical sample information to find the location of a target point. Zhang et al. [33] integrated SPL with Particle Swarm Optimization (PSO)- which is a popular swarm intelligence algorithm- in a noisy Environment, in order to alleviate the

impacts of noise on the evaluation of true fitness and increase the convergence speed.

In order to fasten the SPL scheme, the work reported in [26] proposes to use the last two transitions of the SPL to decide whether to increase or decrease the step size. Intuitively, two suggestions from the Environment in a row for going left or right will increase the step size. On the other hand, the step size is decreased whenever the SPL oscillates between two sates; this might be an indication that the optimal point is located between those two states.

In [8], SPL is modified in accordance with the classical Random Walk-based Triple level Algorithm (RWTA), where Environment provides three kinds of responses, i.e, right, left or unmoved.

A generalization of the hierarchical SPL scheme [28] to the case of deceptive Environment was proposed in [34]. In order to deal with the deceptive nature of the Environment and still be able to estimate the optimal location, the original tree structure found in [28] was extended by a symmetric tree rooted at the root node and it was shown that the SPL will converge to a leaf node in that symmetric tree in case the Environment is deceptive, while it will converge to the leaf node in the original tree if the Environment is informative i.e., not deceptive.

There is a wide range of scientific and real-life problems that can be modeled as the instances of SPL problem, such as adaptive data encoding, web-based applications, etc. [11]. In [7], Granmo and Oommen presented an approach for solving resources allocation problems under noisy Environment using a learning machine that is basically an SPL. The basic SPL version is used to determine the probability of polling a resource among two possible resources at each time instant. The scheme was also generalized to handle the case of more than one material using an hierarchical structure. The paradigm has been applied to determining the optimal polling frequencies of a web-page and to solving sampling estimation problems with constraints [6].

In [30], it is proposed to apply the SPL paradigm to solve the stochastic root finding problem which is a well-known stochastic optimization problem. The classical solution to solve this problem is based on stochastic approximation. Yazidi and Oommen show that it is possible to model the problem as variant of the SPL with adaptive d-ary search.

Recently, Yazidi et al. [29] show that quantiles can be estimated using an SPL type search. The scheme has computational advantages as it uses discretized memory and it is able to adapt to dynamic environments. Another recent application of the SPL [23] is estimating the optimal parameters of Distance Estimation Functions (DEF). Distance Estimation (DE) [10] is a classical problem where the aim is to estimate an accurate value for the real (road) distance between two points which is typically tackled by utilizing parametric functions called Distance Estimation Functions. The authors use the Adaptive Tertiary Search strategy [23], to calculate the best parameters for the DEF. The proposed method uses the current estimate of the distances, the feedback from the Environment, and the set of known distances, to determine the unknown parameters of the DEF. It is suggested that SPL is a better way to determine DEF parameters rather than the traditional Goodness of-Fit (GoF) based paradigm [10].

SPL can also be used to find the appropriate dose in clinical practices and experiments [16].

A possible interesting application, which we focus on in our ongoing research, is to determine the difficulty level of a cognitive training method by SPL. One of the key challenges, faced by many learning methods, is to find the cognitive level of a participant in order of designing suitable level of training. To the best of our knowledge, in most legacy methods, alternating between different training levels and scenarios is simply done by increasing the difficulty if the task is managed, once or over a set of repeated iterations, or by decreasing/fixing the difficulty level if the task is not managed. This problem could be modeled by SPL with certain conditions, such as non-stationary point location, since the manageable difficulty level will change as time goes for trained participant, and unknown certainty/probability of the results. Because there are many factors that might affect the response to a training test that are not related to the real ability of the participant. For instance, Titrated delayed matching-to-sample (TDMTS) method, which is used by behaviour analysts, could easily be modeled as a SPL problem. TDMTS can be used to study important variables for analyzing short-term memory problems [1].

Spaced Retrieval Training (SRT) [3] is also a method of learning and retaining target information by recalling that information over increasingly longer intervals; a method which is especially used for people with dementia [2]. For progressive diseases like dementia, it is so important to estimate the ability level, i.e. point location in SPL, as quickly as possible, since the ability will be rapidly modified during time, affected by training, disease, and patient's condition.

This paper is partially based on our previous work published in [18]. In [18], we show that the SPL problem can be solved by introducing two key multinomially distributed random variables and tracking them using the Stochastic Learning Weak Estimator (SLWE) method. SLWE [25] figures among the most prominent estimators for non-stationary distributions. We proposeed to integrate the SLWE as the inherent part of a more sophisticated and accurate solution for the SPL. The recursive updated form of the SLWE makes it a viable strategy in our problem since the tracked distribution in the case of SLWE is updated incrementally. Therefore, our strategy for estimation of point location revolves around tracking the distribution at each time step and estimating the point based upon it. We applied different statistical operators: maximum, expectation, and median on the estimated probability vectors to obtain our estimates. The results indicate that, the estimates obtained from these methods are smoother than those obtained from legacy SPL solutions and can track the changes more efficiently. The results, also, confirm that using the concept of mutual probability flux between states, according to which transitions are considered as the events of multinomially distributed random variable, is a superior alternative to [20]. We name the contribution as Flux-based Estimation Solution (FES). In the simulation part of initial work reported [18], Environment effectiveness fixed to $p = 0.7$ and the resolution fixed to $N = 16$. It was shown there that the estimated error reduced up to 75%. In the current paper, we do not fix the resolution and consider the case where we can tune the resolution. A new contribution in this paper is to introduce the Last Transition-based Estimation Solution(LTES). This estimator is much simpler than FES and in the case that we have no constraint on the resolution, LTES could estimate the point location equally well with FES.

The Environment effectiveness, i.e. probability of correct answer, is unknown and might vary over time. As the second contribution of this paper, we estimate the probability in tandem with the unknown location estimation.

The remainder of this paper is organized as follows. In Section 2, the SPL problem is defined formally. Section 3 is devoted to presenting our solution for both estimating the point location as well as the Environment effectiveness probability. In this perspective, Section 3.1 introduces the concept of mutual probability flux which is formally proved to be a stronger method compared with the last visited state of the Markov Chain. In Section 3.2, we introduce our estimation approach reckoned as Flux-based Estimation Solution (FES) that is based on a subtle usage of the concept of flux probability. We show that the LTES is a special case of the FES method, and a comparison between the LTES with the FES method is provided at the end of this part. Section 3.3 deals with the related fundamental problem of estimation of the Environment effectiveness. To evaluate the behavior of estimators, extensive simulation results based on synthetic data are presented and discussed in Section 4. Experiments based on real-life data related to online tracking of topics are presented in Section 5. Finally, we drew final conclusions in Section 6.

2 Stochastic Point Location Problem in a Dynamic Setting

This problem considers that the learning mechanism (LM) moves within $[0, 1]$ interval and attempts to locate a point ($0 \leq \lambda^*(n) \leq 1$) that may changes over time n. The Environment E is considered to be informative;

LM receives the right direction to the point location with probability $p^*(n) > 0.5$. This probability of receiving a correct response, which reflects the "effectiveness" of the Environment, is unknown by LM and assumed to be varying.

As aforementioned, we intend to track $\lambda^*(n)$ in an efficient manner. We follow the model presented in [20] and discretize the interval and perform a controlled random walk on it, characterized by $\lambda(n)$. More precisely, we subdivide the unit interval into $N + 1$ discrete points

$$\{0, 1/N, 2/N, \cdots, (N-1)/N, 1\},$$

where N is called the resolution of the learning scheme. Let $\lambda(n)$ be the current location at time step n:

- If E suggests increasing $\lambda(n)$:
  $\lambda(n+1) = \min(\lambda(n) + 1/N, 1)$
- If E suggests decreasing $\lambda(n)$:
  $\lambda(n+1) = \max(0, \lambda(n) - 1/N)$

Hereafter, the binary function $E(n, i)$ stands for the Environment answer at step n and location $\lambda(n) = i/N$, where $E(n, i) = 1$ refers to the Environment suggestion to increase $\lambda(n)$ and $E(n, i) = 0$ refers to the Environment suggestion to decrease $\lambda(n)$. Let Z be an integer value between 0 and $N - 1$, based on above rules, if $Z/N \leq \lambda^*(n) < (Z + 1)/N$ at time n we have:

$$\begin{aligned} Pr(E(n,i)=1) &= p^*(n)\ if\ \ 0 \le i \le Z \\ &= q^*(n)\ if\ Z < i \le N \\ Pr(E(n,i)=0) &= q^*(n)\ if\ \ 0 \le i \le Z \\ &= p^*(n)\ if\ Z < i \le N \end{aligned} \tag{1}$$

Where $q^*(n) = 1 - p^*(n)$.

Based on the results presented in [20], in the stationary case in which $\lambda^*(n) = \lambda^*$, this random walk will converge into a value arbitrarily close to λ^*, when $N \to \infty$ & $n \to \infty$. However, the above asymptotic results are not valid for the non-stationary SPL. Practically, we might experience some constraints, both on time $n \le T$ and on the resolution $N \le R$. Throughout the rest of this paper, we pursue better estimates for $\lambda^*(n)$ than $\lambda(n)$.

3 Estimation Strategies

In this section, we first show the superiority of the Last Transition-based Estimation Solution(LTES) over the last location estimate. Then, a multinomially distributed random variable is considered. We track its probability distribution with SLWE method [25] and estimate the $\lambda^*(n)$ from the estimated distributions. Then, we explain how we can estimate the probability $p^*(n)$ using the estimation of $\lambda^*(n)$.

In [18], we showed that tracking probability distribution for different state transitions, instead of the point locations, yields a better performance. The reason is that the estimation by Markov chain will have many transitions around the true and unknown $\lambda^*(n)$. In the following, we prove that using the concept of mutual probability flux is a stronger tool for solving the SPL problem than using the current point location. In the proof, we consider the static case, i.e. $\lambda^*(n) = \lambda^*$.

3.1 Superior Accuracy with the Concept of Mutual Probability Flux

For simplicity, let $x_i = i/N$ for $i = 0, 1, \ldots, N$. So, the Markov chain states will be the possible value of x_i for $0 \le i \le N$ which belongs to the set of values $\{0, 1/N, 2/N, \cdots, 1\}$. Suppose π_i be the stationary (or equilibrium) probability of the chain being in state x_i. Then, the equilibrium probability distribution vector will be $\mathit{\Pi} = [\pi_0, \pi_1, \cdots, \pi_N]^T$.

We know that, the Markov chain is an instantiation of the birth-death process[1]. It is also known that, such a process is a time reversible Markov chain, i.e. satisfies the detailed balance equation:

$$\pi_i M_{i,j} = \pi_j M_{j,i} \quad \text{for all } i \ne j$$

where $M_{i,j}$'s are transition probabilities. For a complete overview about time reversibility, we refer the reader to an excellent book by Kelly [15]. The following simple proof shows time reversibility of our Markov chain.

If $|i - j| > 1$ for $0 \le i, j \le N$, i.e. x_i and x_j are not adjacent, then the detailed balance equation is obviously true. For a given i, we can divide the states into two

[1] Since the only possible transitions are moving one state to the left or right.

parts, $L = \{x_k | k \leq i\}$ and $R = \{x_k | k > i\}$. Since the Markov chain is a birth-death chain, the only passage between the two parts is the transition x_i to x_{i+1} or x_{i+1} to x_i. The flow from L to R is $\pi_i M_{i,i+1}$ and from R to L is $\pi_{i+1} M_{i+1,i}$. Since Π is stationary, the total flow must be 0, which concludes what is desired:

$$\pi_i M_{i,i+1} = \pi_{i+1} M_{i+1,i}. \tag{2}$$

Let x_i^+ denotes the event according to which the Markov chain makes a transition from x_i to x_{i+1} or from x_{i+1} to x_i. The informed reader would observe the latter event can be related to the concept of flux probability [19, Chapter 8.4]. In fact, in the literature, the flux probability between two neighboring states x_i and x_{i+1} is given by $M_{i,i+1}\pi_i$ which represents the absolute probability of observing a transition from x_i to x_{i+1}. We can see that the probability of x_i^+ can be described as the sum of two flux probabilities; namely the flux probability corresponding to transiting from x_i to x_{i+1}, and the flux probability of transiting in the opposite direction from state x_{i+1} to x_i. In other words, the probability of the event x_i^+, which is shown by π_i^+, equals to the following sum

$$\pi_i^+ = M_{i,i+1}\pi_i + M_{i+1,i}\pi_{i+1}. \tag{3}$$

We call this quantity as *mutual probability flux* between states x_i and x_{i+1}. In the light of this explanation, we call $\Pi^+ = [\pi_0^+, \pi_1^+, \cdots, \pi_{N-1}^+]^T$ the *mutual flux probability vector* between two neighboring states.

Now we intend to investigate the relation between Π and Π^+. Let $x_Z \leq \lambda^* < x_{Z+1}$ and $e = \frac{p}{q} > 1$[2]. As a result of equation (2) and referring to relations in (1), the following balance equations hold.

$$\pi_i = e.\pi_{i-1} \quad \text{whenever} \quad i \leq Z \tag{4}$$

In the case $i \leq Z$ we have $M_{i,i+1} = p$ and $M_{i+1,i} = q$.

$$\pi_i = \frac{\pi_{i-1}}{e} \quad \text{whenever} \quad i > Z+1 \tag{5}$$

In the case $i > Z+1$ we have $M_{i,i+1} = q$ and $M_{i+1,i} = p$. Finally, since we have $M_{Z,Z+1} = p$ and $M_{Z+1,Z} = p$:

$$\pi_{Z+1} = \pi_Z \tag{6}$$

These relations show that values are increasing from π_0 to π_Z and decreasing from π_{Z+1} to π_N; and therefore, π_Z and π_{Z+1} take the maximum value.

Let $\lambda_{\text{tr}}(n)$ be the mean of last two states, i.e. $\lambda_{\text{tr}}(n) = \frac{\lambda(n-1)+\lambda(n)}{2}$. In this case, π_i^+ would be the stationary probability of $\lambda_{\text{tr}}(n)$ chain being in transition x_i^+. We can easily see that the probabilities for $\Pi^+ = [\pi_0^+, \pi_1^+, \cdots, \pi_{N-1}^+]^T$ have higher probabilities around the λ^*.

Whenever $i < Z$, using equation (3), we have

$$\pi_i^+ = p\pi_i + (1-p)\pi_{i+1} = p\frac{q}{p}\pi_{i+1} + q\pi_{i+1}.$$

and therefore

$$\pi_i^+ = 2q\pi_{i+1} \quad \text{whenever} \quad i < Z. \tag{7}$$

[2] Suppose the Environment is stationary; $\lambda^*(n) = \lambda^*$, $p^*(n) = p > 0.5$, and $q^*(n) = q = 1-p$

In the case $i = Z$ we have:

$$\pi_Z^+ = p\pi_Z + p\pi_{Z+1} = 2p\pi_Z, \tag{8}$$

and finally whenever $i > Z$

$$\begin{aligned} \pi_i^+ &= q\pi_i + p\pi_{i+1} = q\pi_i + p\tfrac{q}{p}\pi_i, \\ \pi_i^+ &= 2q\pi_i \quad \text{whenever} \quad i > Z. \end{aligned} \tag{9}$$

Up to this point, we have showed the relation between Π and Π^+. Now, to show the convergence of Π^+, we just need to prove π_Z^+ is greater than π_i^+ for $i \neq Z$ (i.e. $i < Z$ and $i > Z$).

Case 1: $i < Z$: Based on equations (4), (5), and (6), we know that $\pi_Z > \pi_i$ for $i < Z$. We also know that $2q < 1$, when $p > 1/2$; and therefore, $2q\pi_i < \pi_Z$. However, we showed that $2q\pi_i = \pi_i^+$, and as a result:

$$\pi_i^+ = 2q\pi_{i+1} < \pi_Z < 2p\pi_Z = \pi_Z^+$$

Case 2: $i > Z$: Again, we observe that $\pi_Z > \pi_i$ for $i > Z$. So, we have

$$\pi_i^+ = 2q\pi_i \leq \pi_i \leq \pi_Z < 2p\pi_Z = \pi_Z^+.$$

Thus, we have proved that $\pi_i^+ < \pi_Z^+$ for $i \neq Z$, which means that, the transition has higher probabilities at π_Z^+ and lower values at other locations.

Since π_Z^+ is greater than π_Z, we expect that the $\lambda_{\mathrm{tr}}(n)$ estimator, or LTES, performs better than $\lambda(n)$. This can be investigated by comparing the expected estimation error of SPL and LTES. Let E_{SPL} be the expected estimation error for SPL and let E_{LTES} be the expected estimation error for the LTES. For the sake of simplicity, we suppose that λ^* is in the middle of the interval $[Z/N, (Z+1)/N]$ which means $\lambda^* = \frac{Z+(Z+1)}{2N}$.

$$\begin{aligned} \mathrm{E_{SPL}} &= \sum_i \pi_i \mid \lambda^* - x_i \mid \\ &= \sum_i \pi_i \mid \frac{Z+(Z+1)}{2N} - \frac{i}{N} \mid \\ &= \sum_{i \neq Z} \pi_i \mid \frac{Z+(Z+1)}{2N} - \frac{i}{N} \mid + \pi_Z \mid \frac{Z+(Z+1)}{2N} - \frac{Z}{N} \mid \end{aligned} \tag{10}$$

On the other hand for the LTES we have

$$\begin{aligned} \mathrm{E_{LTES}} &= \sum_i \pi_i^+ \mid \lambda^* - x_i^+ \mid \\ &= \sum_i \pi_i^+ \mid \frac{Z+(Z+1)}{2N} - \frac{i+(i+1)}{2N} \mid \\ &= \sum_{i \neq Z} \pi_i^+ \mid \frac{Z+(Z+1)}{2N} - \frac{i+(i+1)}{N} \mid + \pi_Z^+ \mid \frac{Z+(Z+1)}{2N} - \frac{Z+(Z+1)}{2N} \mid \\ &= \sum_{i \neq Z} 2q\pi_i \mid \frac{Z+(Z+1)}{2N} - \frac{i+(i+1)}{2N} \mid \end{aligned} \tag{11}$$

As for large N, $\frac{2i+1}{2N} \approx \frac{i}{N}$, we can write

$$\mathrm{E}_{\mathrm{LTES}} = \sum_{i \neq Z} 2q\pi_i \mid \frac{Z + (Z+1)}{2N} - \frac{i}{N} \mid .$$

From the above equations we get:

$$\begin{aligned} \mathrm{E}_{\mathrm{SPL}} &> \sum_{i \neq Z} \pi_i \mid \frac{Z + (Z+1)}{2N} - \frac{i}{N} \mid \\ &> 2q(\sum_{i \neq Z} \pi_i \mid \frac{Z + (Z+1)}{2N} - \frac{i}{N} \mid) \\ &= \mathrm{E}_{\mathrm{LTES}} \end{aligned} \tag{12}$$

The last inequality is due to the fact that $2q < 1$.

Therefore we conclude that the expected estimation error for LTES is smaller than SPL for large enough N. The results in section 4 confirm the discussion above.

3.2 Flux-based Estimation Solution (FES)

Let $X^+(n)$ denote a multinomially distributed variable over the possible transitions $x_i^+, i = 0, \ldots, N-1$; where the concrete realization of $X^+(n)$ at time step n is $\lambda_{\mathrm{tr}}(n)$. Please note that the distribution of $X^+(n)$ can be explained using the mutual flux probability vector $\Pi^+(n)$. The portion of transitions defined as $P(X^+(n) = x_i^+) = \pi_i^+(n), i = 0, \ldots, N-1$.

The SLWE method estimates the probabilities

$$\Pi^+(n) = [\pi_0^+(n), \pi_1^+(n), \ldots, \pi_{N-1}^+(n)]^T$$

by maintaining a running estimate $S(n) = [s_0(n), s_1(n), \cdots, s_{N-1}(n)]^T$ of $\Pi^+(n)$ where $s_i(n)$ is the estimate of $\pi_i^+(n)$ at time n. The updating rule is (the rules for other values of $s_j(n), j \neq i$, are similar):

$$\begin{aligned} s_i(n+1) &\leftarrow \alpha s_i(n) + (1-\alpha) \text{ when } \lambda_{\mathrm{tr}}(n) = x_i^+ \\ &\leftarrow \alpha s_i(n) \text{ when } \lambda_{\mathrm{tr}}(n) \neq x_i^+ \end{aligned} \tag{13}$$

$0 < \alpha < 1$ is a user-defined parameter for updating the probability distribution. The intuition behind the updating rule is that if $\lambda_{\mathrm{tr}}(n) \neq x_i^+$ we should decrease our estimate $s_i(n)$ which is given by the second part of the updating rule. Similarly, if $\lambda_{\mathrm{tr}}(n) = x_i^+$ we should increase our estimate which is given by the first part of the updating rule.

It is worth mentioning that in [25], $X(n) = X$, i.e. it is not modeled as a function of time and as a result $\Pi(n) = [\pi_0, \pi_1, \cdots, \pi_N]^T$ is time-invariant. The theorems and results are also proven in the asymptotic case when $n \to \infty$ which is in contradiction with the non-stationary assumption for Environment. It is discussed that in practice the convergence takes place after a relatively small value of n. For instance, if the Environment switches its multinomial probability vector after 50 steps, the SLWE could track this change. However, we prefer to use the notation in a way that the point location, and thereafter, the multinomially probability vector are clearly shown to be non-stationary. SLWE converges weakly,

independently of α value, however the rate of convergence is a function of α. Based on previous section where we showed $\pi_i < \pi_Z,\ i \neq Z$, and as $S(n)$ converges to $\Pi^+(n)$, we are able to estimate the point location, $\lambda^*(n)$, by finding the maximum probability, i.e.

$$\begin{aligned} z &= \arg\max_i(s_i(n)) \\ \lambda_{\max}(n) &= x_z^+ \end{aligned} \tag{14}$$

Note that the maximum value refers to a pair that LM transits to the most. For non-unique z, the last visited pair with the max probability value is chosen. See Algorithm 1.

As $n \to \infty$, and for appropriate choices of $\alpha \to 1$; $S(n) \to \Pi^+(n)$. Thus, equation (14) reduces to $z = Z$, as we know that π_Z^+ is the largest component in the vector $\Pi^+(n)$. Then, the error will be ≈ 0 as time goes to infinity.

As a side remark, if $\alpha \leq 0.5$ and $\lambda_{\text{tr}}(n) = x_i^+$, then $s_i(n) \geq 0.5$ for event x_i^+. In other words, $\lambda_{\max}(n) = \lambda_{\text{tr}}(n)$ if $\alpha \leq 0.5$. Because of this, we set $\alpha > 0.5$ in our simulations to avoid repeating the same estimation.

Algorithm 1: Estimation of $\lambda^*(n)$ by FES

input : $N,\ T,\ i = \lfloor N/2 \rfloor,\ E(n,i),\ X(n),\ \alpha$

initialization

$\lambda(0) = x_i,\ S(0) = [s_0(0), s_1(0), \ldots, s_{N-1}(0)]^T = [1/N, 1/N, \ldots, 1/N]^T$

begin

  for $n = 1$ *to* T **do**

    $j = i - (-1)^{E(n,i)}$

    if $j \leq 0$ *or* $j \geq N$ **then**

      $j = i$

    $i = j$

    $\lambda(n) = x_i$

    $x_i^+ = \frac{\lambda(n)+\lambda(n-1)}{2}$

    $S(n) = \alpha S(n-1) + (1-\alpha)\mathcal{I}_i$ `/* ` $\mathcal{I}_i = [0, 0, \cdots, 1, \cdots, 0]^T$`; a vector of size` N `with 1 at` i`th position and 0 elsewhere. */`

    $\lambda_{\max}(n) = x_z^+$ where $z = \arg\max_i(s_i(n))$

output: $\lambda(n),\ \lambda_{\max}(n)$

3.2.1 LTES as a Special Case of FES

The informed reader would remark that the FES scheme needs to keep track of the maximum component of the mutual flux probability vector. For each component, the middle point of the corresponding pair of states is used as an estimate of the point location. A special case of the FES method is to operate without memory, and in this case, the maximum component of the mutual flux probability vector will simply correspond to the middle point of the last visited pair of states. This is also true regarding Algorithm 1, where we see that if we replace α by 0, then FES reduces to the LTES algorithm.

A potential strength of LTES (λ_{tr}) is that we only need to tune the parameter, namely N, while the FES estimator (λ_{max}) contains two parameters N and

α. However, both parameters are related to how rapidly the estimator adjust to changes in the Environment. This suggests that if we are able to tune over N, the LTES approach and Oommen's method could perform equally well as the more sophisticated algorithm with weak estimation.

In the following, we show how to estimate the $p^*(n)$ using λ_{tr} and weak estimators. The Oommen estimate i.e. $\lambda(n)$ can not be a basis for estimation of $p^*(n)$. The reason is that we increase or decrease the probability by comparing the estimation of point location $\hat{\lambda}(n)$ and Environment suggestion $E(n,i)$ at point $\lambda(n)$. In Oommen's method since $\hat{\lambda}(n) = \lambda(n)$ the probability estimation always would be 0.5.

3.3 Estimation of Environment Effectiveness Probability

To estimate $p^*(n)$ based on the estimation of $\lambda^*(n)$, we use simple binomial weak estimator. Let $\hat{\lambda}(n)$ be the estimation of $\lambda^*(n)$. We adjust over γ which is the parameter for binomial weak estimator. See Algorithm 2. Since the probability assumed to change over $[0.5, 1]$, the initial guess of the probability is set to $\hat{p}(0) = 0.75$.

- If ($\lambda(n) < \hat{\lambda}(n)$ and $E(n,i) = 1$) OR ($\lambda(n) > \hat{\lambda}(n)$ and $E(n,i) = 0$):
$$\hat{p}(n) = 1 - \gamma(1 - \hat{p}(n-1))$$
- Else if ($\lambda(n) < \hat{\lambda}(n)$ and $E(n,i) = 0$) or ($\lambda(n) > \hat{\lambda}(n)$ and $E(n,i) = 1$):
$$\hat{p}(n) = max(0.5, \gamma\hat{p}(n-1)) \tag{15}$$
- Else if ($\lambda(n) = \hat{\lambda}(n)$):
$$\hat{p}(n) = \hat{p}(n-1)$$

Basically, the probability $\hat{p}(n)$ increases by a multiplicative parameter γ if the Environment direction $E(n,i)$ agrees with the estimation of point location, $\hat{\lambda}(n)$, and vice verse; the opposite probability $(1 - \hat{p}(n))$ increases by a multiplicative factor γ if they disagree. Since we know that $p^*(n)$ change over $[0.5, 1]$, we restrict our estimations to this domain by setting the lower bound 0.5 in equation (15).

Algorithm 2: Estimation of $p^*(n)$

input : $N,\ T,\ E(n,i),\ \lambda(n),\ \hat{\lambda}(n),\ \gamma$

initialization

$\hat{p}(0) = 0.75$

begin

  for $n = 1$ *to* T **do**

    if *(*$\lambda(n) < \hat{\lambda}(n)$ *AND* $E(n,i) = 1$*) OR (*$\lambda(n) > \hat{\lambda}(n)$ *AND* $E(n,i) = 0$*)* **then**

      $\hat{p}(n) = 1 - \gamma(1 - \hat{p}(n-1))$

    else if *(*$\lambda(n) < \hat{\lambda}(n)$ *AND* $E(n,i) = 0$*) OR (*$\lambda(n) > \hat{\lambda}(n)$ *AND* $E(n,i) = 1$*)* **then**

      $\hat{p}(n) = \max(0.5, \gamma\hat{p}(n-1))$

    else

      $\hat{p}(n) = \hat{p}(n-1)$

output: $\hat{p}(n)$

4 Experimental Results

In this section, we resort to simulation experiments to evaluate the performance of the estimators suggested in this paper. As mentioned before, both $\lambda^*(n)$ and $p^*(n)$, which are not known by LM, could be either constant or dynamic. In this regard, there are many possibilities to define the Environment in which two general types of Environments are considered. Those Environments can show the characteristics of estimators in the best manner.

- Both $\lambda^*(n)$ and $p^*(n)$ change after a fixed amount of time. So their values are fixed for a while until a sharp change happens. We use the sample abbreviation SWITCH-1000-10000 for this type, which means $\lambda^*(n)$ changes after 1000 steps and $p^*(n)$ changes after 10000 steps. The next value of $\lambda^*(n)$ is randomly chosen from $[0, 1]$, and for $p^*(n)$ the random value is chosen from $[0.5, 1]$.
- Both $\lambda^*(n)$ and $p^*(n)$ vary gradually as continuous functions of time. We consider the changes as sine functions. A sample abbreviation for this type would be SINE-1080-10080, which means that $\lambda^*(n)$ has a period of 1080 and $p^*(n)$ has a period of 10080. More precisely, $\lambda^*(n) = 0.5 + 0.5\sin((n/540)\pi)$ where the sine argument changes by $\pi/180$ radians every 3 steps. Therefore, period equals $3 \cdot 360 = 1080$. Moreover, $p^*(n) = 0.75 + 0.25\sin((n/5040)\pi)$ where the sine argument changes by $\pi/180$ radians every 28 steps; so the period equals $28 \cdot 360 = 10080$.

The key aspects of presented estimators can be discussed through eight cases that highlight the salient features of our scheme. For the sake of clarity, these cases are classified into seven headings which are introduced briefly in the following.

The first section 4.1 presents the initial settings when both $\lambda^*(n)$ and $p^*(n)$ change moderately. Cases SWITCH-1000-1000 and SINE-1080-1080 are presented in this part in Fig. 1 and Fig. 2 respectively. Next, in section 4.2 the effect of faster changes in $\lambda^*(n)$ and $p^*(n)$ are addressed through cases SWITCH-100-100 (Fig. 3) and SINE-360-360 (Fig. 4). In the third section 4.3 the effect of changing rate of $p^*(n)$ on estimating $\lambda^*(n)$ in SWITCH dynamic is examined. To do so, the changes of $\lambda^*(n)$ are fixed on 1000, and two alternative cases SWITCH-1000-100 (Fig. 5) and SWITCH-1000-10000 (Fig. 6) are compared with SWITCH-1000-1000 (Fig.1). Additionally, in Fig. 7 a trace plot for tracking $\lambda^*(n)$ via LTES (λ_{tr}) is presented and the behavior of the estimator is discussed through three cases SWITCH-1000-100, SWITCH-1000-1000, and SWITCH-1000-10000. Table 1 summarizes the choices of tuning parameters resulting into the minimum error for λ_{tr} and $\lambda_{\max}$ to the SWITCH cases. The fourth section 4.4 focuses on the SINE dynamic and presents the results for the effect of changing rate of $p^*(n)$ on estimating $\lambda^*(n)$. Similarly, the period of sine function at $\lambda^*(n)$ are fixed on 1080, and two alternative cases SINE-1080-360 (Fig. 8) and SINE-1080-10080 (Fig. 9) are compared with SINE-1080-1080 (Fig. 2). Table 2 summarizes the same data as Table 1 for SINE cases. Fifth section 4.5 is devoted to study the effect of relation between $\lambda^*(n)$ and $p^*(n)$ dynamics on the estimators. Fig. 10 depicts tracking $\lambda^*(n)$ throughout the two scenarios SINE-1080-1080 and SINE-1080-1080-Shift where the second scenario has a shift in the phase of $\lambda^*(n)$. The differences in the tracking performance is discussed in detail. Table 3 is assisting the discussion in this section. Estimation of Environment effectiveness is addressed in the last two sections. In section 4.6, Fig. 11 and Fig. 12 present the estimation error for

various SWITCH and SINE cases respectively. Moreover, Table 4 summarizes the choices of tuning parameters resulting into the minimum error while Environment effectiveness is estimated. In order to compare the effect of tuning parameters, in Table 4, the estimation error for $p^*(n)$ with $N = 5$ and $\alpha = 0.9$ are reported as well. Finally, section 4.7 analyses the results of estimation of Environment effectiveness through tracking curves depicted in Fig. 13 and Fig. 14.

It is worth mentioning that there are two main other approaches to solve the SPL problem which we do not compare with here. The first approach was pioneered by Yazidi et al. [28] and is based on arranging the search space into a tree structure. The second main approach is the CPL-ATS strategy [23,24] and is based on diving the search interval into d sub-intervals and then recursively eliminating at least one sub-interval, thus shrinking the search space. We did not compare with these methods because in contrast to our solution and to Oommen's original SPL solution [20], much more queries are required per iteration. In fact, when it comes to the hierarchical solution [28], three queries are required in the case of a binary tree structure while the CPL-ATS strategy requires as many queries as the number d of sub-interval. Therefore, it would be inappropriate to compare against our method and Oommen's original SPL which use only one query per iteration. Furthermore, the CPL-ATS strategy suffers from the fact that is not suitable for dynamic Environment as it eliminates irreversibly parts of the search space at each epoch. Before proceeding to the experimental results, it is necessary to clarify some general issues regarding the reported data and figures. First, some figures shows the estimation error for a variety of tuning parameters N and α.

Below, we refer to this as "error plots". In all experiments we have considered $\alpha \in [0.6, 0.7, 0.8, 0.9, 0.95, 0.99]$, however, in the sake of clarity, we only depict $\alpha \in [0.6, 0.9, 0.95]$ cases in the error plots.

Along with $\lambda_{\max}(n)$ estimation, $\lambda_{\text{med}}(n)$ and $\lambda_{\exp}(n)$ estimations are presented in [18] respectively as the median and expectation of probability vector. Formally, $\lambda_{\text{med}}(n)$ and $\lambda_{\exp}(n)$ are defined by

- the expected value of the $X^+(n)$ at step n

$$\lambda_{\exp}(n) = \sum_{i=0}^{N-1} x_i^+ s_i(n), \tag{16}$$

- the median of the $X^+(n)$ at step n:

$$\begin{aligned} &\lambda_{\text{med}}(n) = x_z^+ \text{ where } z \text{ is the index satisfying:} \\ &\sum_{i=0}^{z} s_i(n) \geq 0.5 \text{ and } \sum_{i=z}^{N-1} s_i(n) \geq 0.5. \end{aligned} \tag{17}$$

Intuitively, it makes sense to estimate $\lambda^*(n)$ by the most visited transition which is given by $\lambda_{\max}(n)$. However, if the system varies rapidly, the probability vector estimate $S(n)$ will be quite poor. In such a case, taking the expectation might be a more robust alternative, as given by $\lambda_{\exp}(n)$.

Although the main proposal of this paper is $\lambda_{\text{tr}}(n)$ and $\lambda_{\max}(n)$, in order of comparison, we include error plots for $\lambda(n)$, $\lambda_{\text{med}}(n)$ and $\lambda_{\exp}(n)$.

The presented plots in section 4.5, show estimation error of $p^*(n)$ as a function of tuning parameter γ. Since the main objective of this paper is to track $\lambda^*(n)$,

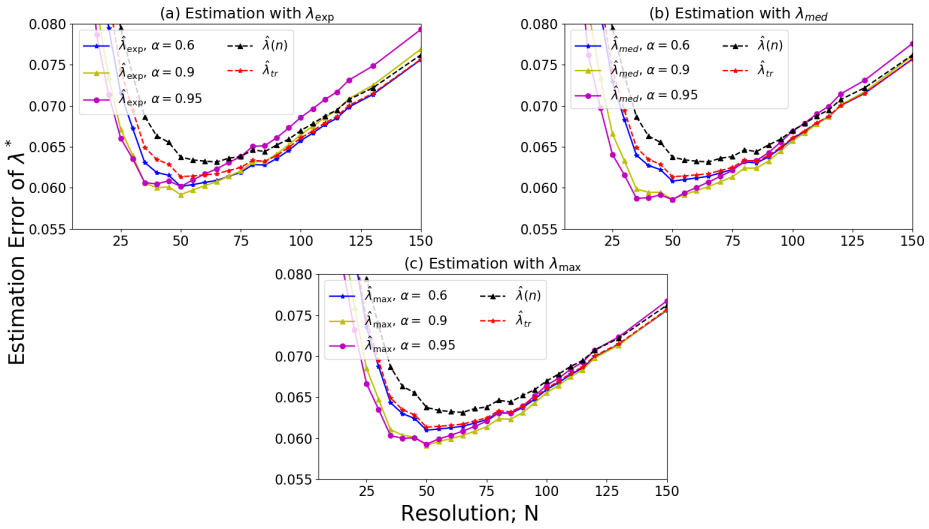

Fig. 1: **SWITCH-1000-1000.** In each of the three sub-figures, one of the max, med, and exp along with the Oommen's method $\lambda(n)$ and the transition λ_{tr} are depicted.

and there are many parameters in estimation of $p^*(n)$, we restrict the plots to the best choices of N in λ_{tr}, and (N and α) in $\lambda_{\max}$. However, we added minimum estimation error for $p^*(n)$ when $N = 5$ and $\alpha = 0.9$ to discuss the effect of resolution on estimation of $p^*(n)$.

To measure the estimation error in the estimation of $\lambda^*(n)$ and $p^*(n)$, the Mean Absolute Error (MAE) will be used. For $\lambda^*(n)$ this becomes

$$\text{MAE}_\lambda = \frac{1}{T}\sum_{n=1}^{T} \mid \hat{\lambda}(n) - \lambda^*(n) \mid \tag{18}$$

where T is the total number of time steps and $\hat{\lambda}(n)$ is the estimate at time step n. Similarly, for $p^*(n)$ this becomes

$$\text{MAE}_p = \frac{1}{T}\sum_{n=1}^{T} \mid \hat{p}(n) - p^*(n) \mid \tag{19}$$

where $\hat{p}(n)$ is the estimate at time step n.

Finally, to remove any Monte Carlo error in the results, we ran a total of 100 chains of length $T = 10^5$ for all cases.

4.1 Moderate changes of both $\lambda^*(n)$ and $p^*(n)$

In this section both $\lambda^*(n)$ and $p^*(n)$ change moderately. Fig. 1 shows the estimation error as a function of resolution for some choices of α. At any resolution, λ_{tr}

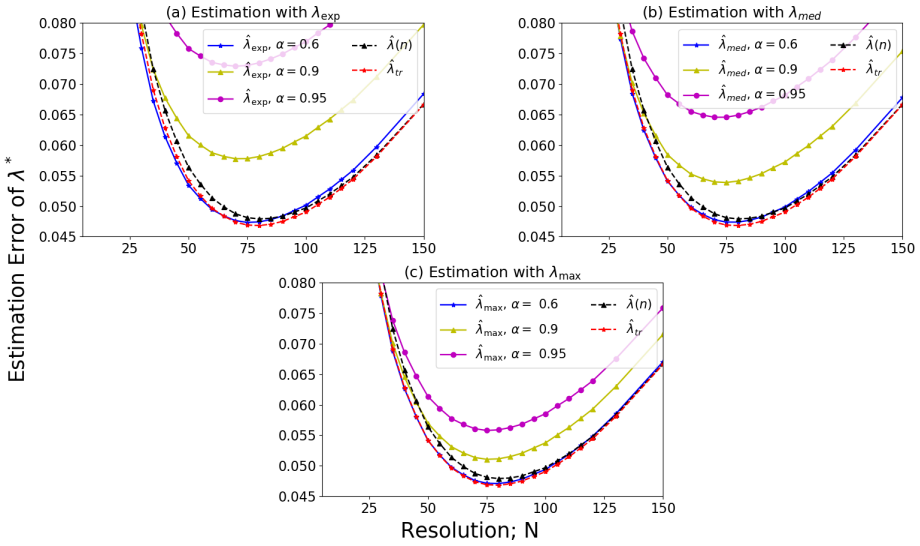

Fig. 2: **SINE-1080-1080.** In each of the three sub-figures, one of the max, med, and exp along with the Oommen's method $\lambda(n)$ and the transition λ_{tr} are depicted.

has lower estimation error than $\lambda(n)$ and indeed, all the cases $\lambda_{\max}$, λ_{med}, and λ_{exp} perform more efficiently than $\lambda(n)$ for, at least, a specific choice of α.

We also note that, the higher resolution will not result in a smaller error in all the cases. For instance, for $\lambda(n)$, the estimation error increases after resolution $N = 65$ in which there is a minimum of errors. As it is represented in Table 1, the minimum error for $\lambda(n)$ equals $e = 0.063149$ when $N = 65$. We reach error $e = 0.061353$ for λ_{tr} at resolution $N = 50$. The best error for $\lambda_{\max}$ is $e = 0.059138$ when $N = 50$ and $\alpha = 0.9$. The minimum error over all scenarios is $e = 0.058559$ which is achieved by λ_{med} estimator when $N = 50$ and $\alpha = 0.95$.

Fig. 2 shows the estimation error as a function of resolution for some choices of α for SINE-1080-1080. All the curves have an optimum resolution point in which any higher resolution cause higher estimation error. In the SINE-1080-1080 case, $\lambda_{\max}$ and λ_{tr} are best satisfying estimators. λ_{tr} with minimum error $e = 0.04682$ at $N = 80$, slightly outperforms $\lambda_{\max}$ with minimum error $e = 0.047095$ at ($N = 80$, and $\alpha = 0.6$).

4.2 Fast changes of both $\lambda^*(n)$ and $p^*(n)$

Here the effect of faster changes in $\lambda^*(n)$ and $p^*(n)$ are addressed through cases SWITCH-100-100 and SINE-360-360.

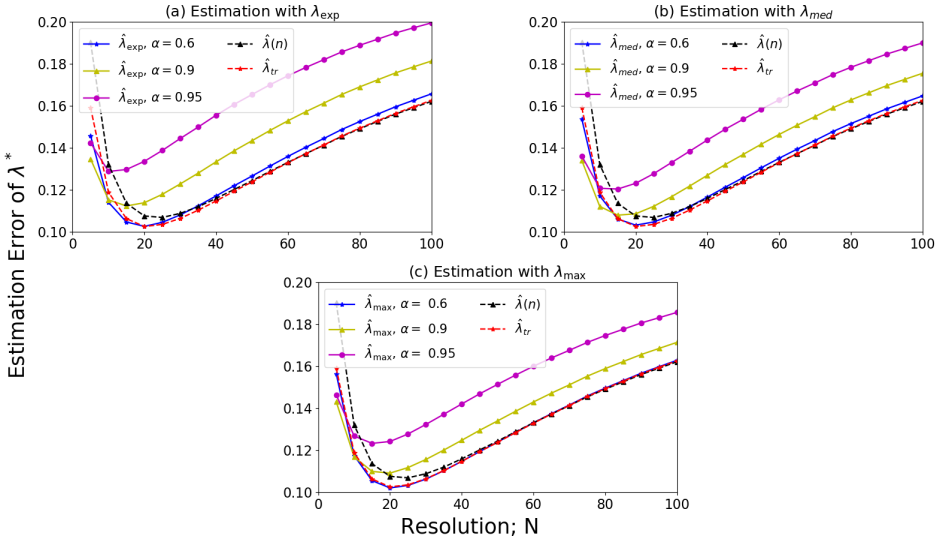

Fig. 3: **SWITCH-100-100.** In each of the three sub-figures, one of the max, med, and exp along with the Oommen's method $\lambda(n)$ and the transition λ_{tr} are depicted.

Fig. 3 is devoted to SWITCH-100-100 that both $\lambda^*(n)$ and $p^*(n)$ randomly switch to a new value in their domain. As expected, comparing the error with the SWITCH-1000-1000 case, the estimation errors are higher. From Table 1 we see that the minimum error for the estimators $\lambda(n)$, λ_{tr}, and $\lambda_{\max}$ are $e = 0.1069$, $e = 0.10258$, and $e = 0.1021$ respectively. The minimum error in case SWITCH-100-100 equals to $e = 0.1021$ and is achieved by $\lambda_{\max}$ when $N = 20$ and $\alpha = 0.6$

As expected, we see that faster changing Environment could be tracked more accurately with smaller values of resolution and α. For instance, compare resolution $N = 20$ in this case, for $\lambda_{\max}$, to $N = 50$ in case SWITCH-1000-1000. The same comparison between $\alpha = 0.9$ and 0.6 shows that to track faster changing Environment, we must rely less on memory.

In Fig. 3, we observe that the best choice of α is dependent on the resolution; for example, if $N = 5$, the $\lambda_{\max}$, λ_{med}, and λ_{exp} with $\alpha = 0.95$ and $\alpha = 0.9$ are superior to the choices with $\alpha = 0.6$. However, if $N = 15$, $\alpha = 0.6$ would be a more desirable option.

Regarding fast changes, Fig. 4 is devoted to SINE-360-360 in which both $\lambda^*(n)$ and $p^*(n)$ change continuously as a sine function with period 360 degree. The minimum error in this case equals $e = 0.08425$ that is achieved by λ_{tr} at $N = 35$. Again, the simpler estimator λ_{tr} outperforms $\lambda_{\max}$ with minimum error $e = 0.085329$ when ($N = 35$ and $\alpha = 0.6$). Note that, the $\lambda_{\max}$ estimator is more efficient than $\lambda(n)$ that has minimum error $e = 0.08634$ when $N = 35$. In comparison with SINE-1080-1080, the estimated error is higher and the best resolution is much smaller in case SINE-360-360. Compare the best resolution $N = 35$ to the case

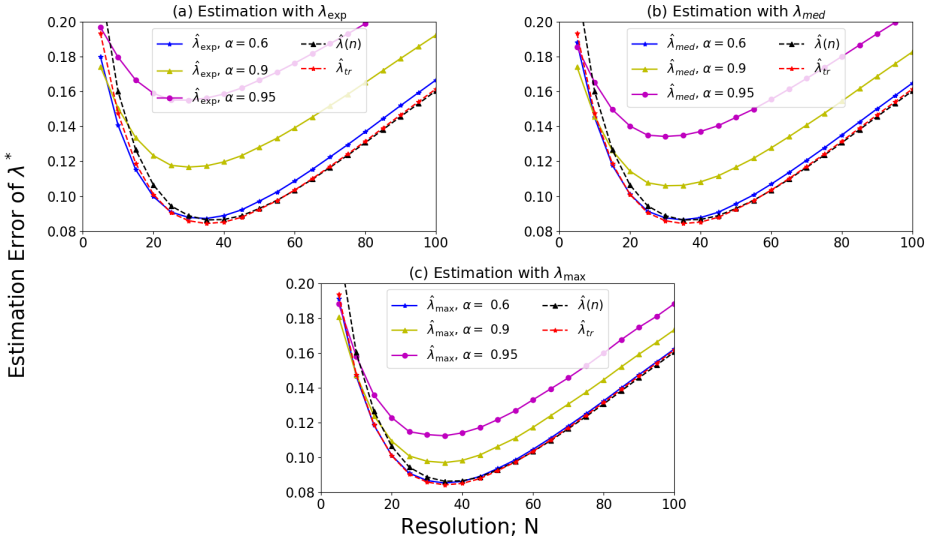

Fig. 4: **SINE-360-360.** In each of the three sub-figures, one of the max, med, and exp along with the Oommen's method $\lambda(n)$ and the transition λ_{tr} are depicted.

SINE-1080-1080 which equals to $N = 80$. Notice that α values closer to 1, produce weaker estimations.

4.3 Effect of changing rate of $p^*(n)$ on estimation of $\lambda^*(n)$ in SWITCH cases

In this section the effect of changing rate of $p^*(n)$ on estimating $\lambda^*(n)$ in SWITCH dynamic is examined. Two alternative cases SWITCH-1000-100 (Fig. 5) and SWITCH-1000-10000 (Fig. 6) are compared with SWITCH-1000-1000 (Fig.1). In Fig. 7 a trace plot for tracking $\lambda^*(n)$ through LTES (λ_{tr}) is presented and the behavior of the estimator is discussed. Additionally, Table 1 summarizes the choices of tuning parameters resulting into the minimum error for λ_{tr} and $\lambda_{\max}$ to the SWITCH cases.

From Fig. 5 and Table 1, we observe that estimators perform better in SWITCH-1000-100 in comparison with SWITCH-1000-1000. For instance, compare minimum error of estimator λ_{tr} in case SWITCH-1000-100 which is $e = 0.03671$ for $N = 80$ with $e = 0.06135$ for $N = 50$ in case SWITCH-1000-1000.

The minimum error in various settings is $e = 0.03538$ which is achieved by λ_{med} estimator when $N = 75$ and $\alpha = 0.9$.

Fig. 6 presents the case SWITCH-1000-10000 where $p^*(n)$ changes ten times slower than SWITCH-1000-1000. It is observable that estimators show a better performance in case SWITCH-1000-10000 compared with SWITCH-1000-1000. As presented in Table 1 we see that the minimum error for the estimators $\lambda(n)$, λ_{tr}, and $\lambda_{\max}$ are $e = 0.05027$, $e = 0.048795$, and $e = 0.04521$ respectively. The best estimator is $\lambda_{\max}$ when $\alpha = 0.95$ and $N = 50$.

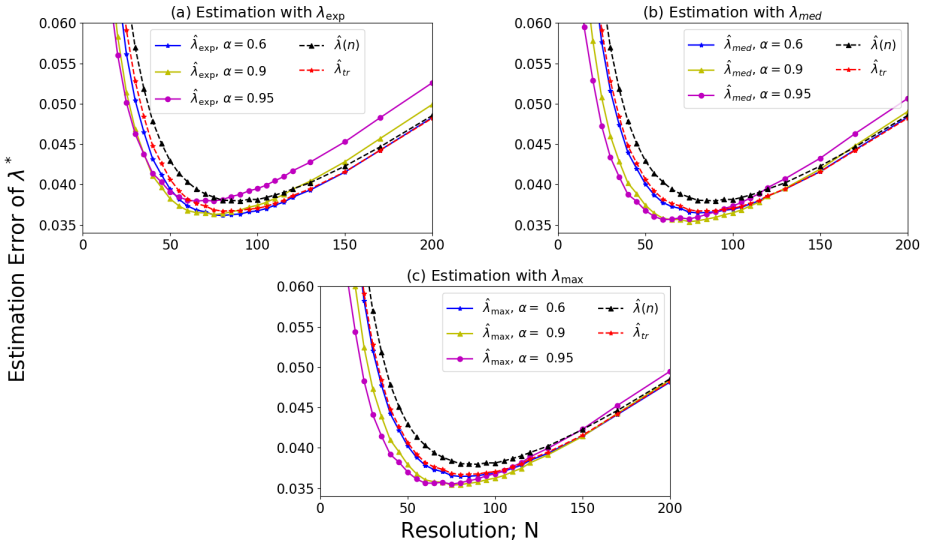

Fig. 5: **SWITCH-1000-100.** In each of the three sub-figures, one of the max, med, and exp along with the Oommen's method $\lambda(n)$ and the transition λ_{tr} are depicted.

In summary, the results, as shown in Fig 5, 6, and Table 1, indicate that when the Environment effectiveness changes fast, the minimum estimation error will be smaller. Comparing minimum errors $e = 0.03539$ to $e = 0.05914$ and $e = 0.04521$ for cases SWITCH-1000-100, SWITCH-1000-1000, and SWITCH-1000-10000 respectively. However, the error in SWITCH-1000-10000 when $p^*(n)$ changes very slow is better than moderate changes in SWITCH-1000-1000. This result is somewhat counterintuitive. In order to understand it, we compare the trace plots of SWITCH-1000-100, SWITCH-1000-1000, and SWITCH-1000-10000 together in Fig. 7.

Fig. 7 shows tracking $\lambda^*(n)$ under optimal choices of parameters for λ_{tr} in order to study the impact of Environment effectiveness on estimation of $\lambda^*(n)$. For the sake of simplicity, suppose there is a same chain $\lambda^*(n)$ in all the cases.

Consider $\lambda^*(n)$ along with three Environment effectiveness chains $p_f^*(n)$, $p_m^*(n)$, and $p_s^*(n)$, for SWITCH-1000-100 (fast changes), SWITCH-1000-1000 (moderate changes), and SWITCH-1000-10000 (slow changes) respectively, in which their average value are approximately the same i.e.

$$\frac{1}{T}\sum_{t=1}^{T} p_f^*(n) \approx \frac{1}{T}\sum_{t=1}^{T} p_m^*(n) \approx \frac{1}{T}\sum_{t=1}^{T} p_s^*(n).$$

Consider SWITCH-1000-10000 with $p_s^*(n)$ and let the estimation of $\lambda^*(n)$ be $\hat{\lambda}(n)$; suppose the following three scenarios:

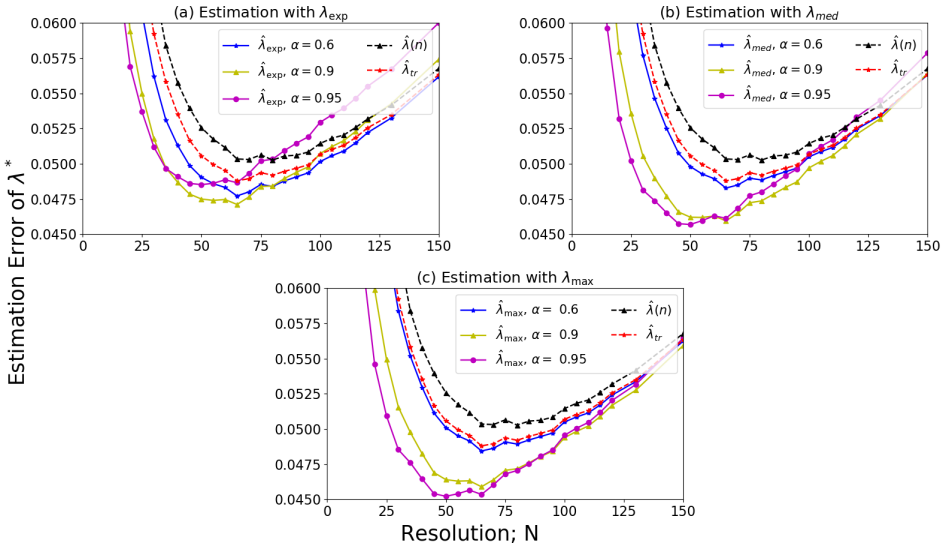

Fig. 6: **SWITCH-1000-10000.** In each of the three sub-figures, one of the max, med, and exp along with the Oommen's method $\lambda(n)$ and the transition λ_{tr} are depicted.

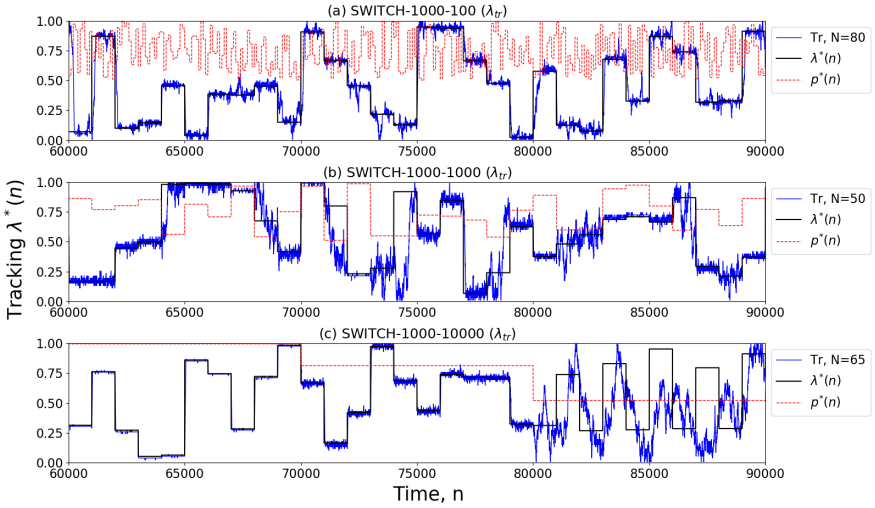

Fig. 7: (a) shows how λ_{tr} tracks $\lambda^*(n)$ in case SWITCH-1000-100. (b) and (c) show the same for SWITCH-1000-1000 and SWITCH-1000-10000 respectively. In all cases, a slice of Environment from $n = 60000$ to $n = 90000$ are represented.

1. The Environment effectiveness is close to 1, see $n = 60000$ to $n = 70000$ in Fig. 7 (c). $\lambda^*(n)$ is easily tracked in this segment and the estimation $\hat{\lambda}(n)$ is satisfactory.
2. The Environment effectiveness is slightly distant from 1, but it is informative, see $n = 70000$ to $n = 80000$ in Fig. 7 (c) where $p_s^*(n)$ value is close to 0.8. Because the information from Environment is somewhat faulty, tracking the point location in this segment is more difficult, but still satisfactory.
3. The Environment effectiveness has a value close to 0.5, see $n = 80000$ to $n = 90000$ in Fig. 7 (c). The estimation $\hat{\lambda}(n)$ is unsatisfactory and it is almost a random chain with a lot of fluctuations. The reason is that estimator does not receive new information from Environment and after a short time $\hat{\lambda}(n)$ will deviate from $\lambda^*(n)$.

In summary, we keep the well estimation in the first segment, the estimation performance is reduced in the second segment, but still satisfactory. Within the third segment, possibility of error is rather high, and $\hat{\lambda}(n)$ fluctuates at a distant point from $\lambda^*(n)$. For $p_s^*(n)$ segments like the last one is discouraging, since we remain in an unsatisfactory situation for a long period of time.

Alternatively, consider the Environment effectiveness $p_f^*(n)$, Fig. 7 (a). It is possible to detect segments like the above three segments but with a much shorter length. So, the behavior of $\hat{\lambda}(n)$ in each of them is not long lasting. Faster changes make the behavior of estimators more like the second segment, with fluctuations around $\lambda^*(n)$.

In Fig. 7 (b); i.e. SWITCH-1000-1000 case, the error is the highest among the three cases. In this case, $p_m^*(n)$ and $\lambda^*(n)$ are changing at the same time. So, the changes of $p_m^*(n)$ has no positive effect on estimation of $\lambda^*(n)$. The best resolution in this case equals $N = 50$, which suggests more changes than SWITCH-1000-10000 with $N = 65$ and SWITCH-1000-100 with $N = 80$. This smaller resolution, produces a higher error. To investigate the negative effect of the simultaneous changes more, we run the SWITCH-1000-1000 case where there is a 500 steps delay between $\lambda^*(n)$ and $p_m^*(n)$ changes. That reduces the minimum error to $e = 0.05144$ for $N = 65$, and approves the negative effect of simultaneous changes in SWITCH cases.

The main observations in Fig. 7 are:

- Tracking $\lambda^*(n)$ is heavily affected by Environment effectiveness. In Fig. 7 (c), there are no fluctuations when $p_s^*(n) \approx 1$, however, when $p_s^*(n) \approx 0.8$, the estimator fluctuates more around the optimal $\lambda^*(n)$, and then when $p_s^*(n)$ is slightly larger than 0.5 the fluctuations are much more bigger.
- Faster changes in Environment effectiveness leads to better estimations of $\lambda^*(n)$. Note that, the rate of changes must be regulated in a way that $\hat{\lambda}(n)$ can converge to $\lambda^*(n)$ when Environment effectiveness is close to 1.
- When $\lambda^*(n)$ and $p^*(n)$ changes together, it is much harder to track $\lambda^*(n)$.
- Since the average value of Environment effectiveness in three cases are supposed to be the same, and the most estimation error is produced in the third segment, there is a better performance in case $p_f^*(n)$ in total.

From Table 1, we observe that the best estimations belong to the case SWITCH-1000-100 and estimator $\lambda_{\max}$.

Table 1: Summary of the choices of tuning parameters resulting into minimum error for λ_{tr} and $\lambda_{\max}$ in SWITCH experiments. The smallest error value in each experiment is represented in bold font.

Estimator \ SWITCH-	1000-1000		100-100		1000-100		1000-10000	
	Error	N	Error	N	Error	N	Error	N
Oommen($\lambda(n)$)	0.06315	65	0.1069	25	0.03797	90	0.05027	80
LTES(λ_{tr})	0.06135	50	0.10258	20	0.03671	80	0.04879	65
FES($\lambda_{\max}$) $\alpha = 0.6$	0.06098	50	**0.1021**	20	0.03646	80	0.04843	65
FES($\lambda_{\max}$) $\alpha = 0.7$	0.06069	50	0.10253	20	0.03628	80	0.04806	65
FES($\lambda_{\max}$) $\alpha = 0.8$	0.06	50	0.10349	20	0.03586	80	0.047312	65
FES($\lambda_{\max}$) $\alpha = 0.9$	**0.05914**	50	0.10916	20	**0.03539**	75	0.04589	65
FES($\lambda_{\max}$) $\alpha = 0.95$	0.05928	50	0.1233	15	0.0355	75	**0.04521**	50
FES($\lambda_{\max}$) $\alpha = 0.99$	0.07094	35	0.20845	10	0.0406	40	0.06035	45

4.4 Effect of changing rate of $p^*(n)$ on estimation of $\lambda^*(n)$ in SINE cases

This section focuses on the SINE dynamic and presents the results for the effect of changing rate of $p^*(n)$ on estimating $\lambda^*(n)$. Two alternative cases SINE-1080-360 (Fig. 8) and SINE-1080-10080 (Fig. 9) are compared with SINE-1080-1080 (Fig. 2), and Table 2 summarizes the same data as Table 1 for SINE cases.

As reported in Table 2, the best estimation error for case SINE-1080-360 (Fig. 8) is achieved through λ_{tr} at $N = 60$ which equals to $e = 0.05363$; compare to the best estimation error for SINE-1080-1080 that equals $e = 0.04682$. In contrast to the SWITCH case, we see that faster changes of probability does not result in smaller estimation errors. We later explain that along with the rate of changes, another factor which plays a role is the phase of changes. In SINE-1080-1080 both $\lambda^*(n)$ and $p^*(n)$ are in phase but in SINE-1080-360 they have different periods and can not be in phase. The effect of this will be addressed in section 4.5 in details. The final case, SINE-1080-10080 in Fig. 9, provides a more clear insight.

In SINE-1080-10080, the changes are asymmetric and the Environment effectiveness varies slower. The minimum estimation error equals $e = 0.08935$ and occurs for $\lambda_{\max}$ at $(N = 40, \alpha = 0.7)$. In this case, we observe that $\lambda_{\max}$ estimator slightly outperforms λ_{tr}. The minimum error for λ_{tr} is $e = 0.08952$ at $N = 40$. Moreover, λ_{exp} results the best minimum error, $e = 0.08835$ when $N = 35$ and $\alpha = 0.6$

By comparing SINE-1080-10080 with SINE-1080-1080 and SINE-1080-360, we observe that its estimation error is the weakest.

If only SINE-1080-360 and SINE-1080-10080 are compared with together, we detect a better estimation at faster changing Environment effectiveness. While SINE-1080-1080 is not following this hypothesis. In contrast to SWITCH cases, where moderate changes of $p^*(n)$ in SWITCH-1000-1000 show the weakest performance, moderate changes of $p^*(n)$ in SINE-1080-1080 show the best results. This

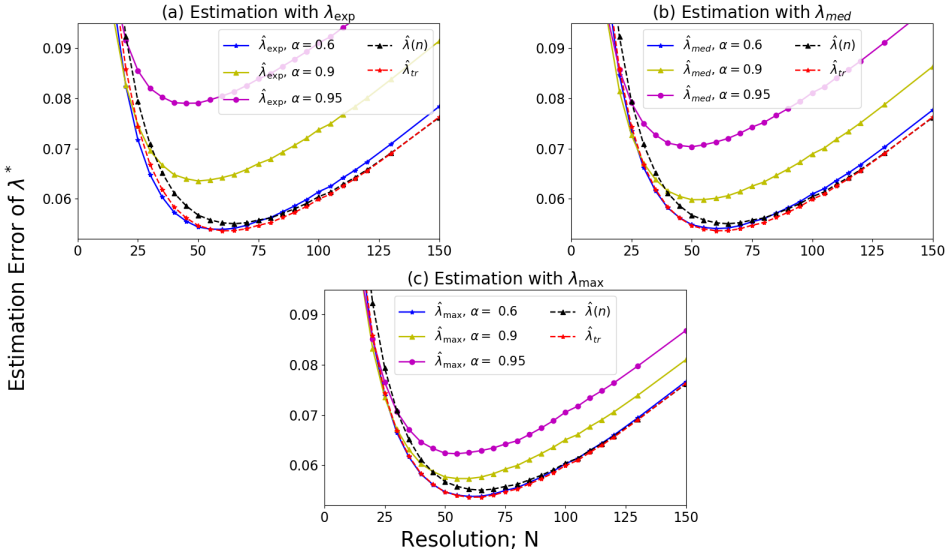

Fig. 8: **SINE-1080-360.** In each of the three sub-figures, one of the max, med, and exp along with the Oommen's method $\lambda(n)$ and the transition λ_{tr} are depicted.

suggests that another factor affects the estimation. Later in section 4.5, Fig. 10, we explain it through the assessment of two different trace plots for SINE-1080-1080 case.

Table 2: Summary of the choices of tuning parameters resulting in minimum error for λ_{tr} and $\lambda_{\max}$ in SINE Experiments. The smallest error value in each experiment is represented in bold font.

Estimator \ SINE-		1080-1080		360-360		1080-360		1080-10080	
		Error	N	Error	N	Error	N	Error	N
Oommen($\lambda(n)$)		0.04791	80	0.08634	35	0.05502	65	0.09244	40
LTES(λ_{tr})		**0.04682**	80	**0.08425**	35	**0.05363**	60	0.08952	40
FES($\lambda_{\max}$)	$\alpha = 0.6$	0.0471	80	0.08533	35	0.05377	60	0.08937	40
	$\alpha = 0.7$	0.04749	80	0.08659	35	0.05398	60	**0.08935**	40
	$\alpha = 0.8$	0.0485	75	0.08965	35	0.0548	60	0.08959	35
	$\alpha = 0.9$	0.05106	75	0.09706	35	0.05733	60	0.09085	35
	$\alpha = 0.95$	0.05578	75	0.11246	35	0.06228	55	0.09474	30
	$\alpha = 0.99$	0.111	65	0.31157	20	0.12002	30	0.14673	25

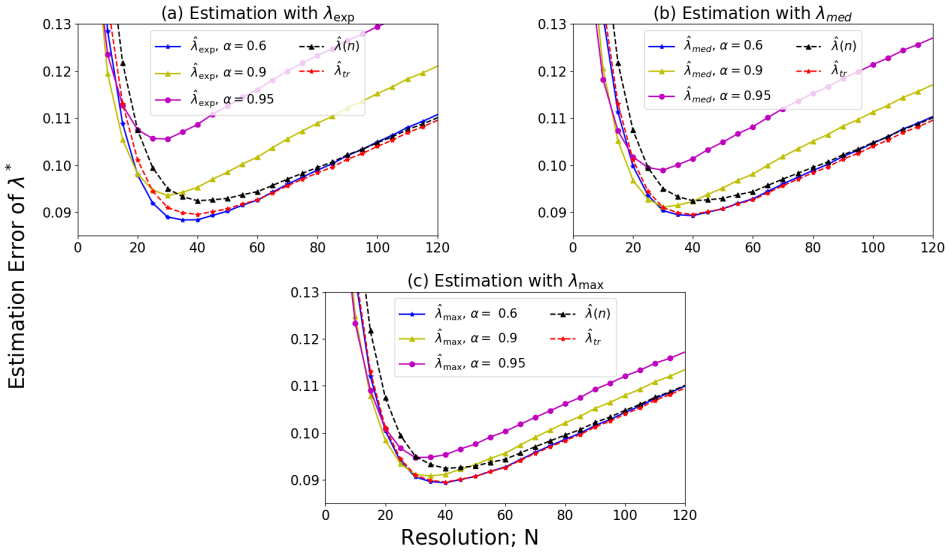

Fig. 9: **SINE-1080-10080.** In each of the three sub-figures, one of the max, med, and exp along with the Oommen's method $\lambda(n)$ and the transition λ_{tr} are depicted.

We have collect the best parameter values and resulted minimum errors of $\lambda(n)$, λ_{tr}, and λ_{max} in Table 2. The best estimations belong to the case SINE-1080-1080. Here, LTES estimator (λ_{tr}) is the best estimator.

4.5 The relation between $\lambda^*(n)$ and $p^*(n)$ changes and the estimation performance

To study the effect of relation between $\lambda^*(n)$ and $p^*(n)$ dynamic on the estimators, we consider the case SINE-1080-1080 that both $\lambda^*(n)$ and $p^*(n)$ are changing according a sine curve. So, we re-run the SINE-1080-1080 case when the argument of sine functions for $\lambda^*(n)$ and $p^*(n)$ differ by $\pi/2$. More formally, what we have reported on Fig. 2 and on the top of Fig. 10 is $\lambda^*(n) = 0.5 + 0.5\sin(\frac{\pi}{3 \cdot 180})$ and $p^*(n) = 0.75 + 0.25\sin(\frac{\pi}{3 \cdot 180})$. In the second run, $\lambda^*(n)$ argument is added by $\pi/2$, so $\lambda^*(n) = 0.5 + 0.5\sin(\frac{\pi}{3 \cdot 180} + \pi/2)$ (Fig. 10 (c)-(d)). Hereafter, we call it SINE-1080-1080-Shift. We observe significant differences between tracking $\lambda^*(n)$ throughout the two scenarios SINE-1080-1080 (Fig. 10 (a)-(b)) and SINE-1080-1080-Shift (Fig. 10 (c)-(d)). The estimators λ_{tr} and λ_{max} track $\lambda^*(n)$ more accurately in SINE-1080-1080 than SINE-1080-1080-Shift. Moreover, the minimum estimation error for λ_{tr} in SINE-1080-1080-Shift is $e = 0.10504$, while it equals to $e = 0.04682$ for λ_{tr} in SINE-1080-1080. Similarly, the minimum estimation error for λ_{max} in SINE-1080-1080-Shift is $e = 0.10382$. Compare it to $e = 0.0471$ for λ_{max} in SINE-1080-1080. We explain it by analyzing Fig. 10. In general, when $\lambda^*(n)$ value is close to 0 or 1, the effect of wrong guidance from Environment is reduced. The reason is that $\lambda(n)$ cannot pass the boundaries.

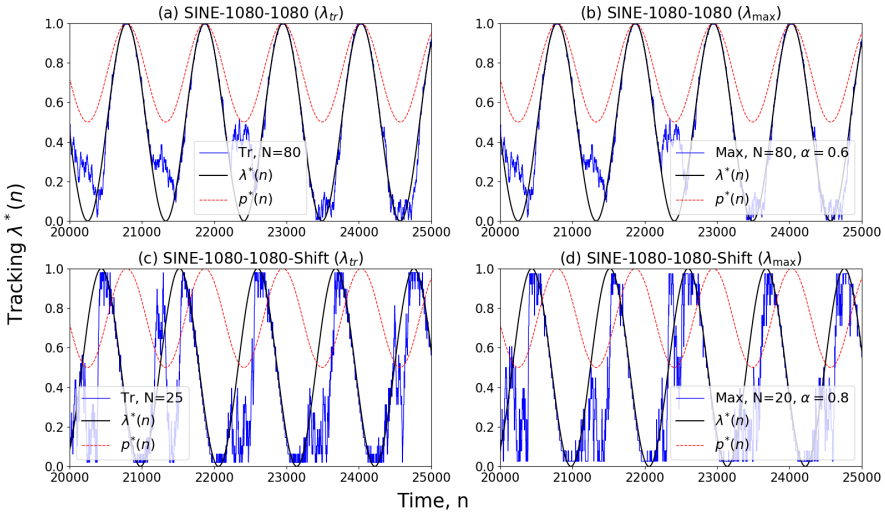

Fig. 10: **SINE-1080-1080.** (a) and (b) subfigures show how the estimation tracks $\lambda^*(n)$ when the argument of sine function is the same by λ_{tr} and λ_{max}, respectively. The (c) and (d) sub-figures show the same when the argument of sine function differ by $\pi/2$.

Let $\lambda^*(n) = 0.5 + 0.5\sin(\theta)$, so we have $p_1^*(n) = 0.75 + 0.25\sin(\theta)$ for SINE-1080-1080 and $p_2^*(n) = 0.75 + 0.25\sin(\theta - \pi/2)$ for SINE-1080-1080-Shift.

Let us take the case SINE-1080-1080 where $p^*(n)$ and $\lambda^*(n)$ are in phase, i.e. $p^*(n) = p_1^*(n)$. Interestingly, the valley of $p_1^*(n)$ corresponds to the valley of $\lambda^*(n)$. Since $p_1^*(n)$ has a valley around 0.5, then the tracking of $\lambda^*(n)$ will be handicapped during that period but this will not affect much the accuracy as $\lambda^*(n)$ is also experiencing a valley and the changes are slow over that valley. However, in case SINE-1080-1080-Shift where $p_2^*(n)$ and $\lambda^*(n)$ are out of phase, a valley of $p_2^*(n)$ coincides with a change of $\lambda^*(n)$ from its lowest value to its biggest value. Then, during that valley of $p_2^*(n)$, $\lambda^*(n)$ tracking gets handicapped and the error is big due to the scheme not being able to track the true underlying $\lambda^*(n)$ that changes dramatically from its min to its max. To be more precise, let

$\theta_1 = 2k\pi + \pi/4$, $\theta_2 = 2k\pi + 3\pi/4$, $\theta_3 = 2k\pi + 5\pi/4$, $\theta_4 = 2k\pi + 7\pi/4$, and $\theta_5 = 2(k+1)\pi + \pi/4$ where k is a positive integer. This way we divide a period of 1080 steps to four equal parts each with 270 steps. For the above values, $\lambda^*(n_1)$ up to $\lambda^*(n_2)$ is situated in the range $[0.85, 1]$; i.e. in 270 successive steps the point location is placed within this range. However, for the next 270 steps, i.e. from $\lambda^*(n_2)$ to $\lambda^*(n_3)$, the values locate in range $[0.15, 0.85]$. We observe that the rate of changes is not uniform. Similarly, $\lambda^*(n_3)$ to $\lambda^*(n_4)$ is placed in the range $[0, 0.15]$ and $\lambda^*(n_4)$ to $\lambda^*(n_5)$ is situated in the range $[0.15, 0.85]$.

Similar to the discussions we had about SWITCH cases, we have

$$\frac{1}{T}\sum_{t=1}^{T} p_1^*(n) \approx \frac{1}{T}\sum_{t=1}^{T} p_2^*(n).$$

This time $p_1^*(n)$ and $p_2^*(n)$ are exactly the same, but their relation to $\lambda^*(n)$ makes them different.

In range n_1 to n_2, where $\lambda^*(n)$ is located in $[0.85, 1]$, $p_1^*(n_1)$ to $p_1^*(n_2)$ is situated in the range $[0.93, 1]$. In range n_2 to n_3, where $\lambda^*(n)$ is located in $[0.15, 0.85]$, $p_1^*(n_2)$ to $p_1^*(n_3)$ is situated in the range $[0.57, 0.93]$. Moreover, in the range n_3 to n_4, where $\lambda^*(n)$ is located in $[0, 0.15]$, $p_1^*(n_3)$ to $p_1^*(n_4)$ is placed in the range $[0.5, 0.57]$. Finally, for range n_4 to n_5, where $\lambda^*(n)$ is located in $[0.15, 0.85]$, $p_1^*(n_4)$ to $p_1^*(n_5)$ is situated in the range $[0.57, 0.93]$.

The intervals for values of $p_2^*(n)$ are achieved through shifting $p_1^*(n)$ values. When $\lambda^*(n)$ is in range $[0.85, 1]$ it takes values in $[0.57, 0.93]$, and when $\lambda^*(n)$ is in range $[0.15, 0.85]$ it takes values in $[0.93, 1]$. When $\lambda^*(n)$ is placed in range $[0, 0.15]$ it takes values in $[0.57, 0.93]$, and when $\lambda^*(n)$ is placed in range $[0.15, 0.85]$ it takes values in $[0.5, 0.57]$. See Table 3 for a summary:

Table 3: Summary of SINE-1080-1080 alternatives

n range	θ range	$\lambda^*(n)$ range	$p_1^*(n)$ range	$p_2^*(n)$ range
$n_1 - n_2$	$(2k\pi + \pi/4) - (2k\pi + 3\pi/4)$	$[0.85, 1]$	$[0.93, 1]$	$[0.57, 0.93]$
$n_2 - n_3$	$(2k\pi + 3\pi/4) - (2k\pi + 5\pi/4)$	$[0.15, 0.85]$	$[0.57, 0.93]$	$[0.93, 1]$
$n_3 - n_4$	$(2k\pi + 5\pi/4) - (2k\pi + 7\pi/4)$	$[0, 0.15]$	$[0.5, 0.57]$	$[0.57, 0.93]$
$n_4 - n_5$	$(2k\pi + 7\pi/4) - (2(k+1)\pi + \pi/4)$	$[0.15, 0.85]$	$[0.57, 0.93]$	$[0.5, 0.57]$

A comparison of the two Environments reveals why estimation of SINE-1080-1080 ($p_1^*(n)$) outperforms SINE-1080-1080-Shift ($p_2^*(n)$):

- In range $n_1 - n_2$, since $p_1^*(n)$ values are higher, we will have more promising estimations. Note that $p_2^*(n)$ is in range $[0.57, 0.93]$, $\lambda^*(n)$ is close to 1, when it reaches its peak, hence its value changes slowly. That is to say in this period, estimations in SINE-1080-1080-Shift are satisfactory. See, for instance, around $n = 24500$ in Fig. 10 (c)-(d).
- In range $n_2 - n_3$, the changes in $\lambda^*(n)$ are fast. Estimation in SINE-1080-1080 case is more difficult than SINE-1080-1080-Shift, because $p_1^*(n)$ is in range $[0.57, 0.93]$ and $p_2^*(n)$ is in range $[0.93, 1]$.
- In range $n_3 - n_4$, the changes in $\lambda^*(n)$ are not fast and the value is close to the boundary. $p_1^*(n)$ is in range $[0.5, 0.57]$ while the information from Environment is almost random. However, since $\lambda^*(n)$ is in a peak, its value is close to boundary and does not change fast, as we explained before, the most fluctuations will be nearby the true $\lambda^*(n)$; see around $n = 21000$ in Fig. 10 (a)-(b). Tracking $\lambda^*(n)$ changes in SINE-1080-1080-Shift case is more accurate than in SINE-1080-1080 case.
- In range $n_4 - n_5$, the changes in $\lambda^*(n)$ are fast. Tracking $\lambda^*(n)$ in SINE-1080-1080 Environment, similar to the range $n_2 - n_3$, is acceptable to some extent. However, tracking the point location in SINE-1080-1080-Shift Environment is almost impossible. As can be seen in Fig. 10 (c)-(d) around $n = 22500$, the combination of fast changes of $\lambda^*(n)$ and distance from boundaries, cause huge

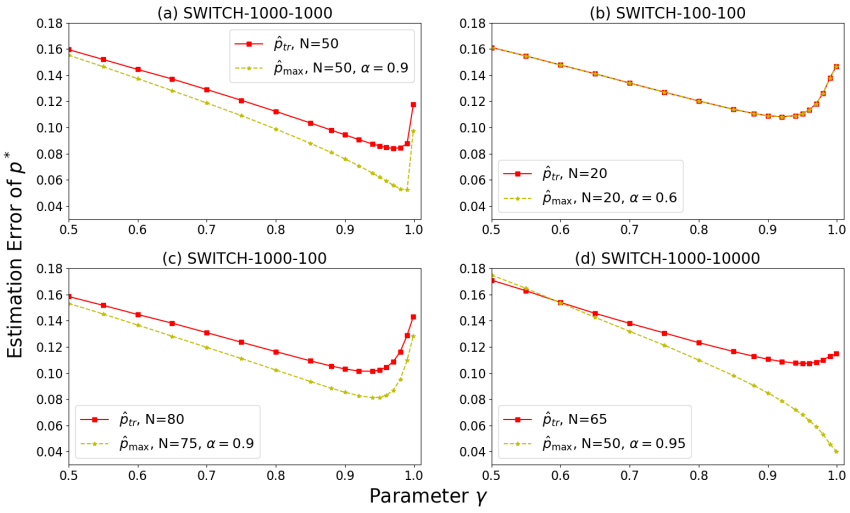

Fig. 11: Estimation of $p^*(n)$ with binomial weak estimation via two alternative choices of $\lambda^*(n)$ estimations ($\lambda_{\max}$ and λ_{tr}) for SWITCH-1000-1000 (a), SWITCH-100-100 (b), SWITCH-1000-100 (c), and SWITCH-1000-10000 (d).

deviation. Such periods result in higher estimation error in SINE-1080-1080-Shift Environment than SINE-1080-1080.

Therefore, both the rate of changes in Environment effectiveness and its relationship to the point location might affect the estimations.

4.6 Estimation of Environment effectiveness: $p^*(n)$

In Fig. 11 and Fig. 12 the estimation error for various SWITCH and SINE cases are presented respectively. Moreover, Table 4 summarizes the choices of tuning parameters resulting into the minimum error while Environment effectiveness is estimated.

In order to depict the estimation performance of $p^*(n)$, we restrict the results to estimation based on two cases $\lambda_{\max}$ and λ_{tr}, for these are the main contribution in this paper which perform the best. Moreover, we consider the best parameters of these two estimators based on results of previous error plots. The tuning parameters resulted to the the best minimum error reported in Table 4. We will consider and discuss the results for $N = 5$ and $\alpha = 0.9$ in Table 4 later. In the following we will compare the results from best $\lambda^*(n)$ estimations. The best minimum error in case SWITCH-1000-1000 is achieved for $\hat{p}_{\max}$ when ($N = 50, \alpha = 0.9$ and $\gamma = 0.99$) equals to $e = 0.0524$. The error for alternative method, i.e. $\hat{p}_{\text{tr}}$ equals to $e = 0.08406$ when ($N = 50,$ and $\gamma = 0.97$), see Table 4.

The best minimum error in case SWITCH-100-100 is simultaneously achieved for $\hat{p}_{\max}$ and $\hat{p}_{\text{tr}}$ at value $e = 0.10819$ with parameters ($N = 20, \alpha = 0.6,$ and $\gamma = 0.92$). In comparison with SWITCH-1000-1000, it is weaker than the previous case in which Environment changes more slowly.

In case SWITCH-1000-100, the best minimum error is obtained through $\hat{p}_{\max}$ when ($N = 75, \alpha = 0.9$ and $\gamma = 0.94$), that equals to $e = 0.08126$. Error for $\hat{p}_{\text{tr}}$ equals to $e = 0.10135$ in the case ($N = 80,$ and $\gamma = 0.94$).

Finally, the case SWITCH-1000-10000 where the minimum error for $\hat{p}_{\max}$ when ($N = 50, \alpha = 0.95$ and $\gamma = 0.999$) equals to $e = 0.04036$. The error for alternative method $\hat{p}_{\text{tr}}$ equals to $e = 0.10736$ in case that ($N = 65,$ and $\gamma = 0.95$).

Overall, it seems like $\lambda_{\max}$ performs a little better than λ_{tr}. However, a significant disadvantage of $\lambda_{\max}$ compared with λ_{tr} is that the tracking of $\lambda^*(n)$ requires tuning of two parameters compared to only one for λ_{tr}. For dynamically changing environments it is usually hard enough to tune one parameter.

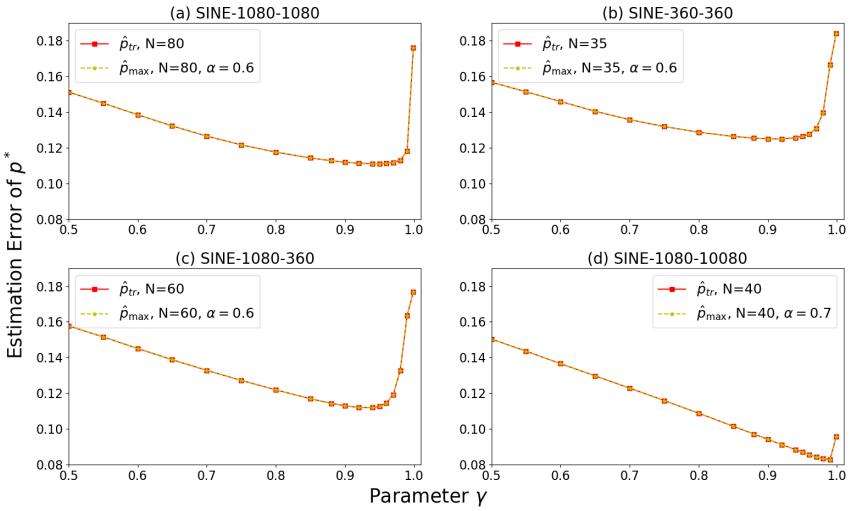

Fig. 12: Estimation of $p^*(n)$ with binomial weak estimation via two alternative choices of $\lambda^*(n)$ estimations ($\lambda_{\max}$ and λ_{tr}) for SINE-1080-1080 (a), SINE-360-360 (b), SINE-1080-360 (c), and SINE-1080-10080 (d) Environments.

In case SINE-1080-1080 illustrated in Fig. 12 (a), we choose $N = 80$ for λ_{tr} and ($N = 80, \alpha = 0.6$) for $\lambda_{\max}$ as the best parameters. As reported in Table 4, the best estimation error for both $\hat{p}_{\max}$ and $\hat{p}_{\text{tr}}$ occurs at $\gamma = 0.94$ and equals $e = 0.11103$.

Similarly, We observe that both estimators are equally well for cases SINE-360-360, SINE-1080-360, and SINE-1080-10080; where the best estimation error equals to $e = 0.125$, $e = 0.11201$, and $e = 0.08299$ respectively, see Table 4 for more details.

Even though the estimation error for point location in case SINE-1080-10080 is weaker than all the cases SINE-1080-1080, SINE-360-360, and SINE-1080-360, its estimated probability is preferred; because slower changes can be tracked more easily.

In SINE cases, apart from the case SINE-1080-10080 and SINE-1080-1080-Shift, the estimation error is poor, i.e. optimal error is greater than 0.1 in both

$\hat{p}_{\rm tr}$ and $\hat{p}_{\rm max}$. In Fig. 11, Fig. 12 and Table 4, we see that in some cases $\hat{p}_{\rm tr}$ and $\hat{p}_{\rm max}$ perform equally well, even though the estimators $\lambda_{\rm tr}$ and $\lambda_{\rm max}$ results are different. The reason is in estimation of Environment effectiveness, the important data is whether the suggested direction by Environment agrees with the estimation or not. In other words the distance between estimation and the point location is not important, and the crucial issue is that both point location and estimation of it, are at the same side- left or right- of the query location. So we can have exactly the same results even if the estimated point is not the same in two estimators.

A natural question that might arise is that whether the best parameters for estimation of $\lambda^*(n)$ are the best for estimating $p^*(n)$ or not. A simple simulation, where we set $N = 5$ and $\alpha = 0.9$, provides a negative answer to this question. These parameters results into a smaller estimation error for all the cases compare to the best parameters. Moreover, the $\hat{p}_{\rm max}$ estimations are all better than $\hat{p}_{\rm tr}$. For instance, in case SWITCH-1000-1000 the estimation error for $\hat{p}_{\rm tr}$ drops from $e = 0.08406$ for $N = 50$, to $e = 0.06812$ for $N = 5$. For $\hat{p}_{\rm max}$, error drops from $e = 0.0524$ with $e = 0.04199$ for the same resolutions. Similarly, for SINE-1080-1080, please compare the error $e = 0.11103$ for $N = 80$ and $\alpha = 0.6$ to $e = 0.06308$ and $e = 0.05056$ for $\hat{p}_{\rm tr}$ and $\hat{p}_{\rm max}$ respectively, when $N = 5$ and $\alpha = 0.9$; see Table 4. We try to justify the reason behind this in the following.

Recall equations (4), (5), and (6) where we have:

$$\begin{array}{ll} \pi_i = e.\pi_{i-1} & \text{whenever } \ i \leq Z, \\ \pi_{Z+1} = \pi_Z, & \text{and} \\ \pi_i = \frac{\pi_{i-1}}{e} & \text{whenever } \ i > Z+1, \end{array}$$

where $e = \frac{p}{q}$. To find a relation between resolution and π_Z^+ we have:

$$\begin{aligned} 1 &= \sum_{i=0}^{N-1} \pi_i^+ \\ &= \sum_{i=0}^{Z-1} \pi_i^+ + \sum_{i=Z+1}^{N-1} \pi_i^+ + \pi_Z^+ \end{aligned} \tag{20}$$

By substituting the relations (7), (8), and (9):

$$\begin{aligned} &= 2q(\sum_{i=1}^{Z-1} \pi_i + \sum_{i=Z+1}^{N-1} \pi_i) + 2p(\pi_Z) \\ &= 2q\pi_Z(\sum_{i=1}^{Z-1} (\frac{1}{e})^i + \sum_{i=Z+1}^{N-1} (\frac{1}{e})^{i-Z}) + 2p(\pi_Z) \end{aligned} \tag{21}$$

By removing q and simplification:

$$\begin{aligned} &= 2p\pi_Z[1 + \frac{1}{e^2}(\sum_{i=0}^{Z-2} (\frac{1}{e})^i + \sum_{i=Z}^{N-2} (\frac{1}{e})^{i-Z})]. \text{ So} \\ \pi_Z &= \frac{1}{2p[1 + \frac{1}{e^2}(\sum_{i=0}^{Z-2} (\frac{1}{e})^i + \sum_{i=Z}^{N-2} (\frac{1}{e})^{i-Z})]} \end{aligned} \tag{22}$$

The above equation implies that for a static environment the larger N, the smaller π_Z. Since for the estimation of $p^*(n)$ the accuracy of point location is not important, a smaller resolution will increase the probability to be at the correct pair, i.e.

$Z/N \leq \lambda^*(n) < (Z+1)/N$. Based on this argument, we can formally prove that a smaller resolution gives a better estimation of $p^*(n)$ while a larger resolution yields a better estimation of $\lambda^*(n)$. Based on the above theoretical result that is in accordance with our experimental results, we therefore suggest to run the SPL in parallel using two different resolutions: a smaller resolution for better estimation of $p^*(n)$ and a larger resolution for better estimation of $\lambda^*(n)$.

Table 4: Summary of tuning parameters resulting into minimum error, along with the parameters $N = 5$ and $\alpha = 0.9$ for p_{tr} and $p_{\max}$. The lowest error value among p_{tr} and $p_{\max}$ for each case is represented in bold font.

Case \ Estimator	λ_{tr}			$\lambda_{\max}$			
	Error	N	γ	Error	N	α	γ
SWITCH-1000-1000	0.08406	50	0.97	**0.0524**	50	0.9	0.99
SWITCH-1000-1000	0.06812	5	0.98	**0.04199**	5	0.9	0.99
SWITCH-100-100	**0.10819**	20	0.92	**0.10819**	20	0.6	0.92
SWITCH-100-100	0.08698	5	0.94	**0.07445**	5	0.9	0.95
SWITCH-1000-100	0.10135	80	0.94	**0.08126**	75	0.9	0.94
SWITCH-1000-100	0.08711	5	0.94	**0.07086**	5	0.9	0.95
SWITCH-1000-10000	0.10736	65	0.95	**0.04036**	50	0.95	0.999
SWITCH-1000-10000	0.08444	5	0.999	**0.0399**	5	0.9	0.999
SINE-1080-1080	**0.11103**	80	0.94	**0.11103**	80	0.6	0.94
SINE-1080-1080	0.06308	5	0.97	**0.05056**	5	0.9	0.97
SINE-360-360	**0.125**	35	0.92	**0.125**	35	0.6	0.92
SINE-360-360	0.077	5	0.94	**0.07433**	5	0.9	0.94
SINE-1080-360	**0.11201**	60	0.94	**0.11201**	60	0.6	0.94
SINE-1080-360	0.07282	5	0.94	**0.06873**	5	0.9	0.94
SINE-1080-10080	**0.08299**	40	0.99	**0.08299**	40	0.7	0.99
SINE-1080-10080	0.04871	5	0.99	**0.0345**	5	0.9	0.99
SINE-1080-1080-Shift	**0.07681**	25	0.97	**0.07681**	20	0.8	0.97
SINE-1080-1080-Shift	0.0586	5	0.97	**0.05437**	5	0.9	0.97

4.7 Environment effectiveness tracking

The results of estimation of Environment effectiveness through tracking curves, which are depicted in Fig. 13 and Fig. 14, are analyzed in this section.

Fig. 13 compares tracking $p^*(n)$ for two cases SWITCH-1000-100 (a)-(b) and SWITCH-1000-10000 (c)-(d) based on estimations $\lambda_{\max}$ and λ_{tr}. We observe that $\hat{p}_{\text{tr}}$ fluctuation is higher than $\hat{p}_{\max}$. Indeed, $\hat{p}_{\max}$ documents a little better peak performance (as the trace plots show), but with the price of requiring tuning of an additional parameter in $\lambda_{\max}$.

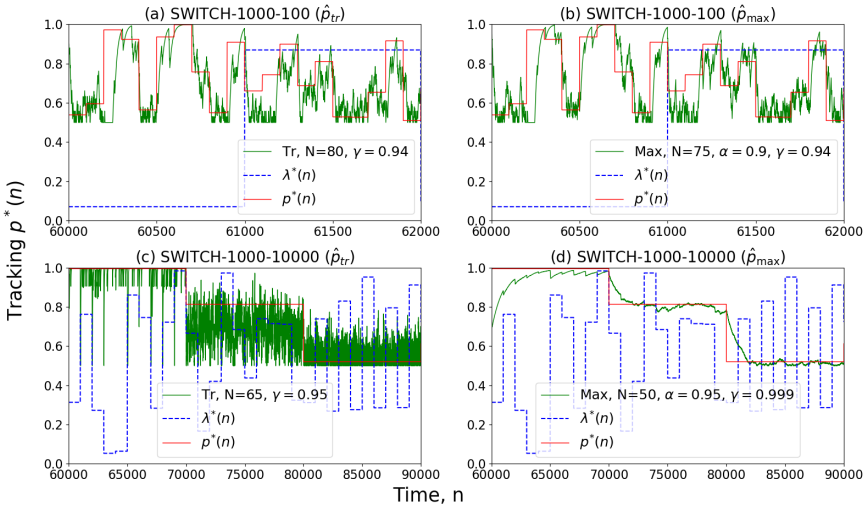

Fig. 13: Comparison of how the estimators track $p^*(n)$ with different dynamics. The sub-figures (a) and (b) show how $\lambda^*(n)$ is tracked by λ_{tr} and λ_{max} respectively in case SWITCH-1000-100. Sub-figures (c) and (d) track $\lambda^*(n)$ by λ_{tr} and λ_{max} respectively, in case SWITCH-1000-10000. In SWITCH-1000-100 cases a slice of Environment from $n = 60000$ to $n = 62000$ are represented. The represented slice for SWITCH-1000-100 is $n = 60000$ to $n = 90000$.

Moreover, as we see more clear at SWITCH-1000-10000 case, whenever probability is closer to 1, any change in $\lambda^*(n)$ intensely affects the $p^*(n)$ estimators. So we detect sharper changes in $p^*(n)$ estimators within the range of $n = 60000$ to $n = 70000$. Then, there is a middle range probability around 0.8 from $n = 70000$ to $n = 80000$. In this range estimators fluctuate more but change less shapely. Interestingly, within the range of $n = 80000$ to $n = 90000$ there are fewer fluctuations. The reason is that the Environment provides almost random directions and the changes of $\lambda^*(n)$ are not followed by the estimators efficiently. Since $\lambda^*(n)$ estimation is the basis for $p^*(n)$ estimation, the changes in $\lambda^*(n)$ could not affect $p^*(n)$ estimations. Therefore, there are no sharp changes when $p^*(n)$ is close to 0.5.

Even though the estimation error for point location in case SWITCH-1000-10000 is weaker than SWITCH-1000-100, its estimated probability is preferred; because slower changes can be tracked more easily.

Fig. 14 compares tracking $p_1^*(n)$ with $p_2^*(n)$ for SINE-1080-1080 and SINE-1080-1080-Shift, based on two estimations λ_{max} and λ_{tr}. An interesting observation regarding the Fig. 14 is the different behavior of estimations near value 1. The estimation for $p_1^*(n)$ is more accurate comparing to $p_2^*(n)$, which can be explained due to the value of $\lambda^*(n)$. The tracking is promoted by the fact that in SINE-1080-1080 (Fig. 14 (a)-(b)), $\lambda^*(n)$ is both close to 1 and changes more slowly. However, the tracking is weakened in SINE-1080-1080-Shift due to $\lambda^*(n)$ changes faster in the middle ranges. Through comparing the two results, it can be seen that although

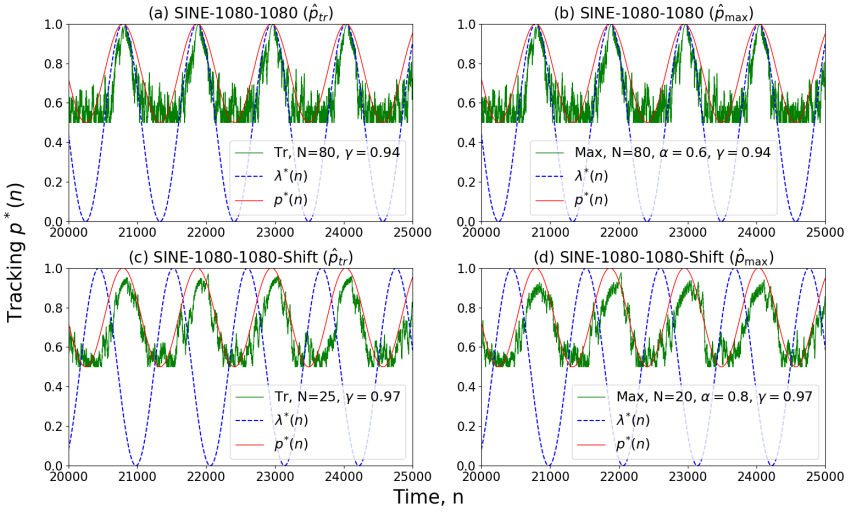

Fig. 14: **SINE-1080-1080.** Comparison of how the estimations track $p^*(n)$, when $\lambda^*(n)$ and $p^*(n)$ are either in-phase (a)-(b), or out of phase (c)-(d).

the estimation of $\lambda^*(n)$ is weaker in SINE-1080-1080-Shift, the proposed estimators for $p^*(n)$ in SINE-1080-1080-Shift are more precise. Compare the minimum error $e = 0.07681$ in SINE-1080-1080-Shift to $e = 0.11103$ in SINE-1080-1080.

5 Real-Life Experiment

In this Section, we show how our proposed algorithms can be used for topic tracking in a stream of text by enhancing an existing estimator proposed in the literature [32]. Online tracking of topics in a stream of text, such as news/social media feeds, has been addressed in several research [4, 25, 9].

Consider *News* and *Entertainment* (including sports) as the two topics of interest. The aim is to model this problem such that the point location would be the probability of current topic being News. This quantity has the characteristics of point location ($\lambda^*(n)$). Additionally, we need some guidance from Environment to be able to run the proposed algorithms. As we will explain in the following, we can consider $x(n) \in \{0, 1\}$ to be a stream of zero and ones, where zero stands for Entertainment and one stands for News. So, $x(n)$ is a Binomial variable and $\lambda^*(n)$- i.e. probability that the current topic is News- is the Bernoulli parameter for each trial.

In [32], the *Stochastic Search on the Line-based Discretized weak Estimator (SSLDE)* is used to estimate the parameters of a distribution, when these parameters changes with time. Note that in distribution parameter estimation problem, the Environment is rather artificial and is constructed to suggest whether increase or decrease the current estimate. We follow the same method as the SSLDE for the online tracking problem and create an artificial Environment that guides us to the point location.

Recall that for resolution N, we have $\lambda(n) \in \{0, 1/N, 2/N, \cdots, i/N, \cdots, (N-1)/N, 1\}$. The estimator is assigned initially the value $\lambda(0) = \frac{\lfloor N/2 \rfloor}{N}$. The updating rules for SSLDE [32] depends on whether the current estimate is greater or less than $N/2$. Suppose that $\lambda(n) = \frac{i}{N}$ and, as mentioned above, let $x(n)$ be the Binomial variable that takes zero or one at time n.

1. Case 1: $[i \geq (N/2)]$:
   - If $x(n) = 1$ and $rand() \leq \frac{N}{2 \cdot i}$:
   $$E(n,i) = 1 \ \rightarrow \ \lambda(n+1) = \frac{\min((i+1), N)}{N}$$
   - Else:
   $$E(n,i) = 0 \ \rightarrow \ \lambda(n+1) = \frac{i-1}{N}$$
2. Case 2: $[i < (N/2)]$:
   - If $x(n) = 0$ and $rand() \leq \frac{N}{2(N-i)}$:
   $$E(n,i) = 0 \ \rightarrow \ \lambda(n+1) = \frac{\max((i-1), 0)}{N}$$
   - Else:
   $$E(n,i) = 1 \ \rightarrow \ \lambda(n+1) = \frac{i+1}{N}$$

where $0 \leq rand() \leq 1$ is a uniform random number generator. Now, we are able to track the probability of the current topic be News by using the above suggestions by Environment.

5.1 Tracking Problem

As mentioned above, News and Entertainment are the two topics we consider in this experiment. To generate the text feed, a large set of related articles are collected from the popular Norwegian newspaper site *vg.no*. The articles are shuffled randomly with the assumption that the algorithm is unaware of when transitions between News and Entertainment take place. In the same line as in [9], based on the stream of text, two methods are used for generating binary observations namely the keyword-list approach and the Machine learning approach.

- **Keyword lists.** A keyword list is a set of words for each topic, here News and Entertainment. For generation of the keyword lists, the popular Pointwise Mutual Information criterion [17] is used. We assume that one word at time is received from the News feed and the task is to track the probability of the current topic of the text stream is News. The best possible estimate based on the keyword list approach is to compute the portion of keywords in each article that are News keywords. This approach is called offline approach. The performance of our algorithm can be compared to this offline approach and see how close our online estimates are to the optimal offline approach.
  Fig. 15 shows the tracking of the probability that the current topic is News for FES (Max) and LTES (Tr) for the first 15000 words. The total number of keywords in the experiment was 800400, while there was not a fixed period for changing between topics. We see that our algorithms are able to track changes

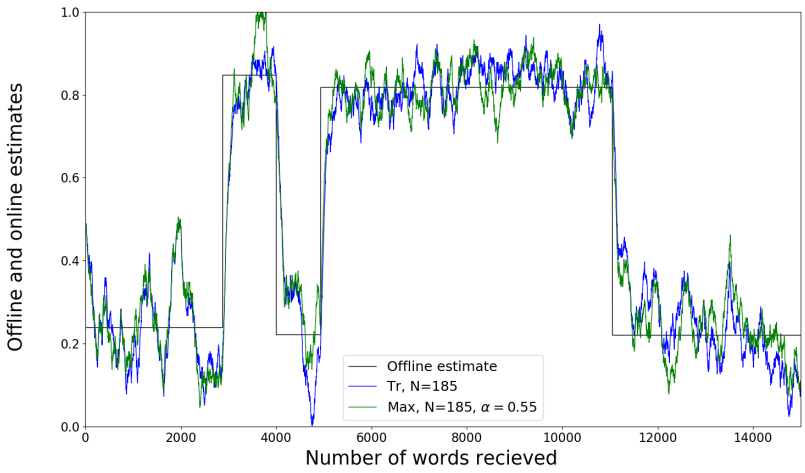

Fig. 15: Trace plot for FES (Max) and LTES (Tr) for tracking the probability of the current topic be News in topic tracking experiment with keyword list approach. The black curves show the offline estimate.

in the News stream well. For instance, look at period $n = 5000$ to $n = 11000$. A difference between these real data from the simulations in section 4 is that here the rate of changes is not fixed. As we see in Fig. 15 the offline estimate changes rapidly in some periods (for $n = 2000$ to $n = 5000$) and does not change for a long period ($n = 5000$ to $n = 11000$). Since in average the data has long fixed periods, the best achieved resolution is $N = 185$, which is better for the long periods.

It is worth mentioning that this tracking data can be used as a classifier. Consider what we really want to understand from the data is that if the current feed belongs to the News or Entertainment. Indeed, the required answer is if the probability of the current topic be News is greater than 0.5 or not. Interestingly, for classification application, the best resolution is much smaller, i.e $N = 45$. The reason is that for classification, the flexibility is much more important compared to accuracy.

– **Machine Learning.** The most used approach to automatically classify text into different classes like topics or sentiment is to train a machine learner. The process starts by dividing the training text stream in batches of 20 words, each within one of the News or Entertainment topics. In the machine learning approach, the documents (batches) were represented by word frequencies in a bag of word matrix. These batches are used to train a machine learning model. For this experiment multinomial ridge regression [5] is used through the glmnet package in R [27]. For the testing part, the single words of the text stream were collected into batches of 20 words. Each batch in this phase were classified into one of the News or Entertainment topics using the trained multinomial

regression model. The probabilities of the current topic were updated in the same manner as for the keyword list approach.

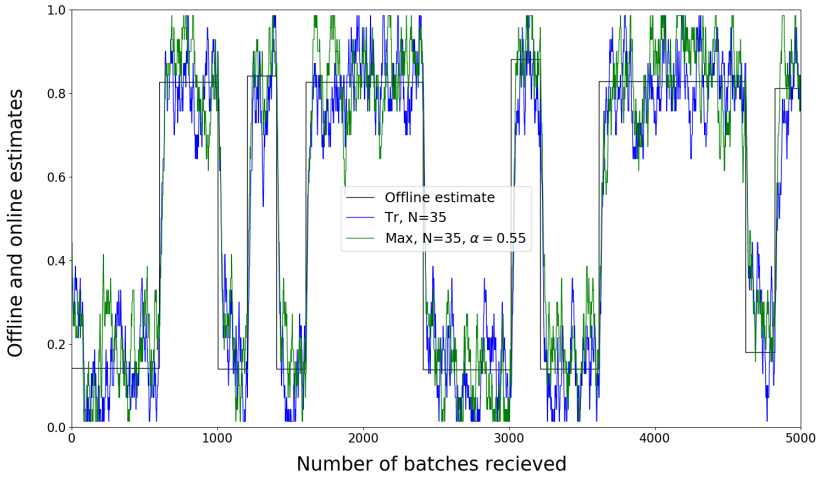

Fig. 16: Evaluation of FES (Max) and LTES (Tr) for tracking the News in feed experiment in machine learning approach. The black curves show the offline estimate.

Fig. 16 shows the tracking of the probabilities for the different topics for the machine learning approach. The total number of batches was 189141 which we depict the tracking in the first 5000 batches. Data changes faster in the ML approach and therefore the best resolution is much smaller; $N = 35$. We see that the fluctuations are greater than Fig. 15 because of this smaller resolution, but in turn, it is more adaptable with fast changes. Similar to the keyword list, if we use the algorithms for classification, the resolution will be even smaller; i.e. $N = 11$. So, being aware of that there might be different best parameters in the estimation for different applications is an important point.

6 Conclusion

A wide range of real-life problems can be modeled as a SPL problem, especially when the Environment is considered to be non-stationary. The random walk method, that Oommen presented for solving the SPL problem, is known to converge into a value arbitrarily close to the point location, when both resolution and time tend to infinity. Oommen's method simply discretizes the interval and performs a controlled random walk on it. This paper is an extention of the preliminary work presented [18] where we propose a new method to estimate the point location in the SPL problem domain. In the current paper, we have introduced

the mutual probability flux concept and have proved that Flux-based Estimation Solution (FES) and Last Transition-based Estimation Solution (LTES), as a special case, always outperform Oommen's method. Moreover, we present a method to estimate Environment effectiveness, $p^*(n)$. This simple method could track the probability of receiving correct response from the Environment in tandem with the unknown location estimation.

Apart from theoretical proofs several experiments are presented in order to understand the characteristics of each method. We argued that λ_{tr}, proposed in this paper, is equally simple but with better estimation performance than Oommen's method. $\lambda_{\max}$, $\lambda_{\exp}$ and λ_{med} show better estimation performance than λ_{tr} in low resolutions, but this comes with the price of tuning one additional parameter. This suggests that if we have no constraint on N, i.e. $\lambda^*(n)$ represents a continuous quantity, we can tune just with N and estimate with LTES. But in the case where freely tuning over N is not possible, tuning with α and using one of $\lambda_{\max}$, $\lambda_{\exp}$, and λ_{med} could provide more accurate estimations.

As experiments show, the tracking of $\lambda^*(n)$ performs better when $p^*(n)$ value is close to 1. The estimation performance of $\lambda^*(n)$ drops drastically when $p^*(n)$ is close to 0.5. This is as expected, since in case $p^*(n) = 1$, our estimation procedure will be correct, i.e. $\hat{\lambda}(n)$ switches back and forth around the true $\lambda^*(n)$. In contradiction, in the case $p^*(n)$ is close to 0.5, we have more faulty feedback, and so an unsatisfactory estimation of $\lambda^*(n)$.

Based on the results, we have also discussed when $\lambda^*(n)$ value is close to 0 or 1, the effect of faulty guidance from Environment will be reduced to some extent and the estimation is slightly better. Moreover, if $p^*(n)$ takes the same value in average, faster changes of $p^*(n)$ are preferable; with the condition that changes in $p^*(n)$ are slow enough that estimator could converge into $\lambda^*(n)$ when $p^*(n)$ is reaching to 1. In this case, faster changes interrupt a long lasting weak estimation and bring the estimator back into a more accurate value. However, if $p^*(n)$ and $\lambda^*(n)$ changes simultaneously, the positive effect of faster changes of $p^*(n)$ is lost. We have also discussed that, not only the rate of changes, but also the relation between $\lambda^*(n)$ and $p^*(n)$ affects the estimation error where $p^*(n)$ represents reliability of the feedback from Environment. A satisfactory estimation of $p^*(n)$ informs us to what extent we can trust the feedback and subsequently the estimations we have built upon that.

References

1. Erik Arntzen and Hanna Steinunn Steingrimsdottir. On the use of variations in a delayed matching-to-sample procedure in a patient with neurocognitive disorder. In Braunstein Swahn, Palmier, editor, *Mental Disorder*. iConcept Press, 2014.
2. Cameron J Camp, Jean W Foss, Ann M O'Hanlon, and Alan B Stevens. Memory interventions for persons with dementia. *Applied Cognitive Psychology*, 10(3):193–210, 1996.
3. Cameron J Camp, G Gilmore, and P Whitehouse. Facilitation of new learning in alzheimer's disease. *Memory, aging, and dementia: Theory, assessment, and treatment*, pages 212–225, 1989.
4. Massimo De Santo, Gennaro Percannella, Carlo Sansone, and Mario Vento. A multi-expert approach for shot classification in news videos. In *International Conference Image Analysis and Recognition*, pages 564–571. Springer, 2004.
5. Jerome Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1, 2010.

6. Ole-Christoffer Granmo and B John Oommen. Optimal sampling for estimation with constrained resources using a learning automaton-based solution for the nonlinear fractional knapsack problem. *Applied Intelligence*, 33(1):3–20, 2010.
7. Ole-Christoffer Granmo and B John Oommen. Solving stochastic nonlinear resource allocation problems using a hierarchy of twofold resource allocation automata. *IEEE Transactions on Computers*, 59(4):545–560, 2010.
8. Ying Guo, Hao Ge, Jinchao Huang, and Shenghong Li. A general strategy for solving the stochastic point location problem by utilizing the correlation of three adjacent nodes. In *Data Science in Cyberspace (DSC), IEEE International Conference on*, pages 215–221. IEEE, 2016.
9. Hugo Lewi Hammer and Anis Yazidi. Parameter estimation in abruptly changing dynamic environments using stochastic learning weak estimator. *Applied Intelligence*, pages 1–17, 2018.
10. Jessica Havelock, B John Oommen, and Ole-Christoffer Granmo. Novel distance estimation methods using ?stochastic learning on the line? strategies. *IEEE Access*, 6:48438–48454, 2018.
11. M Anwar Hossain, Jorge Parra, Pradeep K Atrey, and Abdulmotaleb El Saddik. A framework for human-centered provisioning of ambient media services. *Multimedia Tools and Applications*, 44(3):407–431, 2009.
12. De-Shuang Huang and Wen Jiang. A general cpl-ads methodology for fixing dynamic parameters in dual environments. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 42(5):1489–1500, 2012.
13. Yeong Min Jang. Estimation and prediction-based connection admission control in broadband satellite systems. *ETRI journal*, 22(4):40–50, 2000.
14. Wen Jiang, De-Shuang Huang, and Shenghong Li. Random walk-based solution to triple level stochastic point location problem. *IEEE transactions on cybernetics*, 46(6):1438–1451, 2016.
15. Frank P Kelly. *Reversibility and stochastic networks*. Cambridge University Press, 2011.
16. EE Kpamegan and N Flournoy. Up-and-down designs for selecting the dose with maximum success probability. *Sequential Analysis*, 27(1):78–96, 2008.
17. Christopher D Manning, Christopher D Manning, and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT press, 1999.
18. Asieh Abolpour Mofrad, Anis Yazidi, and Hugo Lewi Hammer. Solving stochastic point location problem in a dynamic environment with weak estimation. In *Proceedings of the International Conference on Research in Adaptive and Convergent Systems*, pages 30–35. ACM, 2017.
19. Randolph Nelson. *Probability, stochastic processes, and queueing theory: the mathematics of computer performance modeling*. Springer Science & Business Media, 2013.
20. B John Oommen. Stochastic searching on the line and its applications to parameter learning in nonlinear optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 27(4):733–739, 1997.
21. B John Oommen and Dragos Calitoiu. Modeling and simulating a disease outbreak by learning a contagion parameter-based model. In *Proceedings of the 2008 Spring simulation multiconference*, pages 547–555. Society for Computer Simulation International, 2008.
22. B John Oommen, Sang-Woon Kim, Mathew T Samuel, and Ole-Christoffer Granmo. A solution to the stochastic point location problem in metalevel nonstationary environments. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 38(2):466–476, 2008.
23. B John Oommen and Govindachari Raghunath. Automata learning and intelligent tertiary searching for stochastic point location. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 28(6):947–954, 1998.
24. B John Oommen, Govindachari Raghunath, and Benjamin Kuipers. Parameter learning from stochastic teachers and stochastic compulsive liars. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 36(4):820–834, 2006.
25. B John Oommen and Luis Rueda. Stochastic learning-based weak estimation of multinomial random variables and its applications to pattern recognition in non-stationary environments. *Pattern Recognition*, 39(3):328–341, 2006.
26. Tongtong Tao, Hao Ge, Guixian Cai, and Shenghong Li. Adaptive step searching for solving stochastic point location problem. In *International Conference on Intelligent Computing*, pages 192–198. Springer, 2013.
27. R Core Team. R: A language and environment for statistical computing, 2017.

28. Anis Yazidi, Ole-Christoffer Granmo, B John Oommen, and Morten Goodwin. A novel strategy for solving the stochastic point location problem using a hierarchical searching scheme. *IEEE transactions on cybernetics*, 44(11):2202–2220, 2014.
29. Anis Yazidi, Hugo Hammer, and B John Oommen. Higher-fidelity frugal and accurate quantile estimation using a novel incremental discretized paradigm. *IEEE Access*, 6:24362–24374, 2018.
30. Anis Yazidi and B John Oommen. A novel technique for stochastic root-finding: Enhancing the search with adaptive d-ary search. *Information Sciences*, 393:108–129, 2017.
31. Anis Yazidi and B John Oommen. The theory and applications of the stochastic point location problem. In *New Trends in Computing Sciences (ICTCS), 2017 International Conference on*, pages 333–341. IEEE, 2017.
32. Anis Yazidi and Basantkumar John Oommen. Novel discretized weak estimators based on the principles of the stochastic search on the line problem. *IEEE transactions on cybernetics*, 46(12):2732–2744, 2016.
33. JunQi Zhang, SiYu Lu, Di Zang, and MengChu Zhou. Integrating particle swarm optimization with stochastic point location method in noisy environment. In *Systems, Man, and Cybernetics (SMC), 2016 IEEE International Conference on*, pages 002067–002072. IEEE, 2016.
34. Junqi Zhang, Yuheng Wang, Cheng Wang, and MengChu Zhou. Symmetrical hierarchical stochastic searching on the line in informative and deceptive environments. *IEEE transactions on cybernetics*, 47(3):626–635, 2017.
35. Junqi Zhang, Liang Zhang, and Mengchu Zhou. Solving stationary and stochastic point location problem with optimal computing budget allocation. In *Systems, Man, and Cybernetics (SMC), 2015 IEEE International Conference on*, pages 145–150. IEEE, 2015.