Experimental Assessment of the Reliability for Watermarking and Fingerprinting Schemes

Frédéric Cérou, Teddy Furon, Arnaud Guyader

Hindawi Publishing Corporation
EURASIP Journal on Information Security
Volume 2008, Article ID 414962, 12 pages
doi:10.1155/2008/414962

Research Article

Experimental Assessment of the Reliability for Watermarking and Fingerprinting Schemes

Frédéric Cérou, Teddy Furon, and Arnaud Guyader

TraitemEnt, Modélisation d'Images & CommunicationS (TEMICS), Institut Rennes - Bretagne Atlantique Research Center (INRIA), Campus de Beaulieu, 35042 Rennes Cedex, France

Correspondence should be addressed to Teddy Furon, teddy.furon@irisa.fr

Received 19 August 2008; Accepted 27 November 2008

Recommended by Deepa Kundur

We introduce the concept of reliability in watermarking as the ability of assessing that a probability of false alarm is very low and below a given significance level. We propose an iterative and self-adapting algorithm which estimates very low probabilities of error. It performs much quicker and more accurately than a classical Monte Carlo estimator. The article finishes with applications to zero-bit watermarking (probability of false alarm, error exponent), and to probabilistic fingerprinting codes (probability of wrongly accusing a given user, code length estimation).

1. INTRODUCTION

Watermark decoders are in essence stochastic processes. There are at least three sources of randomness: the unknown original content (for blind decoders), the unknown hidden message, and the unknown attack the watermarked content has undergone. The output of the decoder is thus a random variable and this leads to a very disturbing fact: there will be errors in some decoded messages. This also holds for watermark detectors which have to take the decision whether the content under scrutiny has been watermarked or not.

1.1. Watermarking reliability

In order to be used in an application, a watermarking technique must be reliable. We introduce here the concept of reliability as the guarantee that not only these inherent errors very rarely happen, but also that their frequency or their probability is assessed to be below a given level. Here are two application scenarii where a wrong estimation of the probability of error could lead to a disaster.

1.1.1. Copy protection

Assume commercial contents are encrypted and watermarked and that future consumer electronics storage devices have a watermark detector. These devices refuse to record a watermarked content. The probability of false alarm is the probability that the detector considers an original piece of content (which has not been watermarked) as protected. The movie that a user shot during his holidays could be rejected by his storage device. This absolutely nonuser-friendly behavior really scares consumer electronics manufacturers. In the past, the Copy Protection Working Group of the DVD forum evaluated that at most one false alarm should happen in 400 hours of video [1]. As the detection rate was one decision per ten seconds, this implies a probability of false alarm in the order of 10^{-5}. An accurate experimental assessment of such a low probability of false alarm would demand to feed a real-time watermarking detector with non-watermarked content during 40 000 hours, that is, more than 4 years! Proposals in response of the CPTWG's call were, at that time, never able to guarantee this level of reliability.

1.1.2. Fingerprinting

In this application, users' identifiers are embedded in purchased content. When content is found in an illegal place (e.g., a P2P network), the right holders decode the hidden message, find a serial number, and thus they can trace the traitor, that is, the customer who has illegally broadcasted his copy. However, the task is not that simple because dishonest

users might collude. For security reason, anticollusion codes have to be employed. Yet, these solutions (also called weak traceability codes [2]) have a nonzero probability of error (defined as the probability of accusing an innocent). This probability should be, of course, extremely low, but it is also a very sensitive parameter; anticollusion codes get longer (in terms of the number of bits to be hidden in content) as the probability of error decreases. Fingerprint designers have to strike a tradeoff, which is hard to conceive when only a rough estimation of the probability of error is known. The major issue for fingerprinting algorithms is the fact that embedding large sequences implies also assessing reliability on a huge amount of data which may be practically unachievable without using rare event analysis.

1.2. Prior works

Estimation of probabilities of false alarm or bit and message error rate have been a concern of watermark designers since the beginning. A first choice is to derive the formula of this probability. However, this is often impossible or with the restricting assumption that the real data fit the statistic model used in the derivation. The decoder's decision is often based on a value of score, which writes as a sum of many and more or less independent random extracted features. This explains the abusive resort to the central limit theorem to establish the probability that this score reaches a given value; the score is supposed to be Gaussian distributed and the probability is expressed with the erf function [3, 4]. However, even if the conditions are sometimes fulfilled to apply the CLT, the convergence rate to the Gaussian law is very crucial and depends on the third moment of the extracted features (in the most simple case) as stated by the Berry-Esséen bound [5]. Roughly speaking, a small probability of error amounts to integrate the tail of the cdf of the score, where the CLT approximation by a Gaussian law is very bad. A better way is to calculate upper bounds (Chernoff's bound, union bound, and nearest neighbor bound). The issue is then about the tightness of the bound, which is usually good only over a precise range of parameter values. Numerical approximations of the probability formula also exist like the Beaulieu method [6] or the DFT method [7] used when the score is the summation of i.i.d. random variables. However, they need the true pdf or the characteristic function of these variables.

When these approaches are not possible, then the last choice is the experimental estimation. However, we have seen in watermarking articles [3, 4, 8, 9] that experimental estimation were only based on the Monte Carlo (MC) method, which is very inefficient for a low probability of error. This naive approach consists in running n experiments and to count the number of times k that the decoder failed. Then, the probability of error p is estimated by the frequency that error happened on this set of experiments: $\widehat{p} = k/n$. Let X_i denote the result of the ith experiment. X_i equals 1 if there is a decoding error, 0 else. X_i is then a Bernoulli random variable, and $K = \sum_{i=1}^{n} X_i$ is a binomial r.v.: $K \sim B(n, p)$, whose expectation is $E[K] = np$ and variance $\mathrm{Var}[K] = np(1-p)$. The estimator $\widehat{p}$ is unbiased ($E[\widehat{p}] = p$) and its variance, $\mathrm{Var}[\widehat{p}] = p(1-p)/n$, asymptotically goes to zero. However, one needs at least around p^{-1} experiments to make it work, and even worse, its relative standard deviation is given by $\sqrt{\mathrm{Var}[\widehat{p}]}/E[\widehat{p}] \approx 1/\sqrt{pn}$. Hence, for a decent accuracy of the estimator reflected here by its relative standard deviation, n must be taken as several times bigger than p^{-1}. This method is thus inadequate for estimating a probability below 10^{-6} because it then needs millions of experiments.

As far as we know, almost nothing concerning experimental estimation of low probability of errors has been done by the watermarking community although there exist better methods than the simple MC simulations [10]. They have been successfully applied, for instance, to estimate frequencies of packet losses in digital communications [11]. We guess that, sadly, the watermarking community is more image processing oriented, so that people usually ignore these recent tools. Yet, the application of these methods to watermarking is not trivial, because they assume a proper statistical model which may not be suitable for watermarking. There is a need for very general or self-adaptive methods resorting to the fewest as possible assumptions.

This article proposes a new experimental method estimating low probabilities of error for watermarking applications. It is a strong adaptation of a complex method whose good properties have been theoretically proven in [12]. Section 2 presents as simply as possible this adaptation, and Section 3 experimentally validates the approach for a very simple scenario of watermarking detection (a.k.a. zero-bit watermarking). Section 4 applies the method to a more difficult watermarking problem: the experimental measurement of error exponents under attacks, the reliability of Tardos code in the fingerprinting application.

2. OUR ALGORITHM

Before explaining the method, let us first describe zero-bit watermarking within a few lines. A watermark detector receives two types of contents: original contents and watermarked (possibly attacked) contents. It decides the type of the piece of content under scrutiny based on the observation of L features extracted from the content, whose values are stacked in a vector $\mathbf{x}$. Then, it calculates the likelihood that this vector is watermarked thanks to a score function $d(\cdot) : \mathbb{R}^L \mapsto \mathbb{R}$. It decides that the content is watermarked if $d(\mathbf{x})$ is above a given threshold τ. The probability of false alarm p_{fa} is the probability that $d(\mathbf{x}) > \tau$ whereas the piece of content has not been watermarked. The probability of false negative p_{fn} is the probability that $d(\mathbf{x}) < \tau$ when the content is indeed watermarked. The reliability of the watermarking technique is assessed if p_{fa} is below a small level. From a geometrical point of view, let us define the acceptance region $\mathcal{A} \subset \mathbb{R}^L$ as the set of vectors $\mathbf{x}$ such that $d(\mathbf{x}) > \tau$.

For the sake of simplicity, we explain the proposed method when applied on zero-bit watermarking. The key idea of our experimental method is to gradually encourage the occurrences of the rare event (here a false alarm) by generating a sequence of events which are rarer and rarer. In terms of probability, we factorize the very small probability

to be estimated in a product of bigger probabilities and thus easier to estimate. Our estimator can be written as $\widehat{p}_{\text{fa}} = \prod_{i=1}^{N} \widehat{p}_i$.

2.1. Key idea

To factorize a probability into a product of bigger probabilities, we use the following trick: let $A_N = A$ be the rare event, and A_{N-1} a related event such that when A_N occurs, A_{N-1} has also occurred. However, when A_{N-1} occurs, it does not imply that A_N is true. Hence, A_{N-1} is less rare an event than A_N. This justifies the first equality in the following equation, the second one being just the Bayes rule:

$$\begin{aligned} \text{Prob}[A_N] &= \text{Prob}[A_N, A_{N-1}] \\ &= \text{Prob}[A_N \mid A_{N-1}] \cdot \text{Prob}[A_{N-1}]. \end{aligned} \tag{1}$$

Repeating the process, we finally obtain:

$$\begin{aligned} p_{\text{fa}} = \text{Prob}[A_N] &= \text{Prob}[A_N \mid A_{N-1}]\text{Prob}[A_{N-1} \mid A_{N-2}] \\ &\cdots \text{Prob}[A_2 \mid A_1]\text{Prob}[A_1], \end{aligned} \tag{2}$$

provided that $\{A_j\}_{j=1}^{N}$ is a sequence of nested events. Knowing that estimation of a probability is easier when its value is bigger, we have succeeded in decomposing a hard problem into N much easier problems.

This decomposition is very general, but the construction of this sequence of nested events is usually not a simple task. An exception is when the rare event A_N admits a geometrical interpretation: A_N occurs when $\mathbf{x} \in \mathcal{A}_N$. A sequence of nested events translates then in a sequence of subsets $\mathcal{A}_1 \subset \cdots \subset \mathcal{A}_{N-1} \subset \mathcal{A}_N$. The task is even simpler in zero-bit watermarking because an indicator function of these events can be as follows: $\mathbf{x} \in \mathcal{A}_j$ if $d(\mathbf{x}) > \tau_j$. Nested events are created for a sequence of increasing thresholds: $\tau_1 < \tau_2 < \cdots < \tau_N = \tau$.

2.2. Generation of vectors

The first step of our algorithm estimates $p_1 = \text{Prob}[A_1]$. In practice, N is large enough so that this probability is not lower than 0.1. Then, a classical MC estimator is efficient. However, the variance of the estimator $\widehat{p}_1$ has a strong impact on the variance of $\widehat{p}_{\text{fa}}$. Therefore, the number of trials n_1 is several times bigger than p_1^{-1}, while being far less than p_{fa}^{-1}, the order of magnitude of the number of trials needed for a direct MC estimator of the probability of false alarm.

For this first step, we must generate n_1 vectors $\mathbf{x}$. Either we have a good statistical model, that is, the pdf $p_{\mathbf{X}}$ of these extracted features accurately captures the reality. Then, it is not difficult to generate synthetic data, that is, pseudo-random vectors that follow the statistical behavior of the extracted features. Or, we feed the watermarking detector with n_1 natural images. We count the number k_1 of vectors whose score is higher than τ_1.

As a byproduct, this first step leads not only to an estimator but also to a generator of the event A_1. It is not very efficient, as it produces vectors $\mathbf{x} \sim p_{\mathbf{X}}$ and then select those vectors belonging to the set $\mathcal{A}_1$. Hence, approximately only $p_1 n_1$ occurrences of the event A_1 are produced.

2.3. Replication of vectors

The issue of the second step is the estimation of the probability $p_2 = \text{Prob}[A_2 \mid A_1]$. We set the threshold τ_2 just above τ_1, so that this probability is large (typically not lower than 0.1). Once again, an MC estimator is $\widehat{p}_2 = k_2/n_2$, where k_2 is the number of vectors $\mathbf{x}$ of the set $\mathcal{A}_1$ which also belong to $\mathcal{A}_2$. We need to generate n_2 vectors $\mathbf{x}$ distributed according to $p_{\mathbf{X}}$ and in the set $\mathcal{A}_1$. We could the first step of the algorithm to generate these vectors, but it is not efficient enough as such.

We then resort to a so-called replication process, which almost multiplies by a factor ρ the size of a collection of vectors in a particular region of the space. For each vector of this collection, we make ρ copies of it, and then we slightly modify the copies in a random manner. This modification process must leave the distribution of the data invariant: if its inputs are distributed according to $p_{\mathbf{X}}$, its outputs follow the same distribution. A modified copy is likely to belong to the set if the modification is light. However, we check whether this is really the case, and go back to the original vector if not. The replication process is thus a modification (line 15 in Algorithm 1) followed by a filter (line 16). It leaves the distribution of the vector invariant.

Since we have run the first step, we have k_1 vectors in $\mathcal{A}_1$. We choose a replication factor $\rho_1 \approx \widehat{p}_1$ more or less conserving the same amount of vectors because the second probability to be estimated has the same order of magnitude than the first one and n_1 was enough for that purpose. We calculate the score of the $\rho_1 k_1$ modified copies. We conserve the copies whose score is bigger than τ_1, and replace the other ones by their original vector. This makes the $n_2 = \rho_1 k_1$ input vectors of the MC estimator. Again, these two first steps lead to an estimator $\widehat{p}_2$ of p_2, but also to a mean to generate occurrences of the event A_2.

The core of the algorithm is thus the following one. A selection process kills the vectors (named particles in statistics) whose score is lower than an intermediate threshold τ_i, these are branched on selected particles. A replication process proposes randomly modified particles and filters the ones that are still above the intermediate threshold. Selection and replication steps are iterated to estimate the remaining probabilities $\widehat{p}_3, \ldots, \widehat{p}_N$.

2.4. Adaptive threshold

The difficulty is now to give the appropriate values to the thresholds $\{\tau_i\}_1^{N-1}$, and also to the sizes of the sets $\{n_i\}_1^N$. The probabilities to be estimated must not be very weak in order to maintain reasonable set sizes. Moreover, it can be shown that the variance of $\widehat{p}_{\text{fa}}$ is minimized when the probabilities $\{p_i\}_i^N$ are equal. However, to set the correct value of the thresholds, we would need the map $\tau = F^{-1}(p)$ which we do not have . Otherwise, we would already know what the value of $p_{\text{fa}} = F(\tau)$ is.

Require: subroutines GENERATE, HIGHER_SCORE & MODIFY
for $i = 1$ to n **do**
  $\mathbf{x}_i \leftarrow$ GENERATE$(p_{\mathbf{X}})$; $dx_i \leftarrow d(\mathbf{x}_i)$;
end for
$N \leftarrow 1$;
$\tau_N \leftarrow$ HIGHER_SCORE$(\mathbf{dx}, k)$; $\tau' \leftarrow \tau_N$;
while $\tau' < \tau$ and $N < N_{\max}$ **do**
  $t \leftarrow 1$;
  for $i = 1$ to n **do**
    if $dx_i \geq \tau'$ **then**
      $\mathbf{y}_t \leftarrow \mathbf{x}_i$; $dy_t = dx_i$; $t \leftarrow t + 1$;
    end if
  end for
  for $i = 1$ to k **do**
    for $j = 1$ to n/k **do**
      $\mathbf{z} \leftarrow$ MODIFY$(\mathbf{y}_i)$;
      if $d(\mathbf{z}) > \tau'$ **then**
        $\mathbf{x}_{(i-1)n/k+j} \leftarrow \mathbf{z}$; $dx_{(i-1)n/k+j} \leftarrow d(\mathbf{z})$;
      else
        $\mathbf{x}_{(i-1)n/k+j} \leftarrow \mathbf{y}_i$; $dx_{(i-1)n/k+j} \leftarrow dy_i$;
      end if
    end for
  end for
  $N \leftarrow N + 1$
  $\tau_N \leftarrow$ HIGHER_SCORE$(\mathbf{dx}, k)$; $\tau' \leftarrow \tau_N$;
end while
$k' \leftarrow 0$;
for $i = 1$ to n **do**
  if $dx_i > \tau$ **then**
    $k' \leftarrow k' + 1$;
  end if
end for
return $\widehat{p}_{\text{fa}} = k' k^{N-1}/n^N$;

Algorithm 1: Estimation of the probability that $d(\mathbf{x}) > \tau$, when $\mathbf{x} \sim p_{\mathbf{X}}$.

The idea is to set the thresholds adaptively. The number of vectors is constant in all the experimental rounds: $n_i = n$ for all $i \in \{1 \cdots N\}$. The threshold τ_i has the value such that $k_i = k$. k and n are thus the parameters of the algorithm. The estimated probabilities are all equal to $\widehat{p}_i = p = k/n$ for all $i \in \{1 \cdots N-1\}$. It means that the selection process sorts the scores in a decreasing order, and adaptively sets τ_i as the value of the kth higher score. Vectors whose score are below this threshold are removed from the stack, and replaced by copies of vectors above. This means that the size of the stack is constant and that the replication factors $\{\rho_i\}$ are all equal to n/k (k divides n). All the vectors in the stack are independently modified. The modification of a vector is accepted if its new score is still above the threshold.

The last step is reached when $\tau_i > \tau$. Then, we set $N = i$, $\tau_N = \tau$, and k_N is the number of scores above τ, so that, for this last iteration, $\widehat{p}_N = k_N/n$. At the end, the probability of false alarm is estimated by

$$\widehat{p}_{\text{fa}} = \frac{k_N}{n} p^{N-1}. \tag{3}$$

We stress the fact that, formally, N, k_N, and $\{\tau_i\}_1^N$ are indeed random variables and their estimations in the algorithm should be denoted by $\widehat{N}$, $\widehat{k}_{\widehat{N}}$, and $\{\widehat{\tau}_i\}_{i=1}^N$. For the sake of clarity, we did not use different notations from the deterministic case of the preceding section. The number of iterations is expected to be as follows:

$$E[N] = \left\lfloor \frac{\log p_{\text{fa}}^{-1}}{\log p^{-1}} \right\rfloor + 1. \tag{4}$$

The total number of detector trials is nN, which has a logarithmic scale with respect to p_{fa}^{-1}, whereas a classical MC estimator would need at least p_{fa}^{-1} trials.

The method is given in pseudocode in Algorithm 1. Note that the thresholds $\{\tau_i\}$ are stored in memory. This is not useful when estimating p_{fa}, but this gives a nice byproduct for ROC curves: the map $p = f(\tau)$ is estimated through the following points: $\{(p^j, \tau_j)\}_{j=1}^{N-1}$. From [12], the method inherits the asymptotic properties of consistency and normality. With equations

$$\begin{aligned} \widehat{p}_{\text{fa}} &\xrightarrow[n \to +\infty]{\text{a.s.}} p_{\text{fa}}, \\ \sqrt{n}(\widehat{p}_{\text{fa}} - p_{\text{fa}}) &\xrightarrow[n \to +\infty]{\mathcal{L}} \mathcal{N}(0, \sigma^2), \end{aligned} \tag{5}$$

Require: subroutine GENERATE_RAND_PERM
  π = GENERATE_RAND_PERM (k);
  for $i = 1$ to n **do**
    $Index(i) = \pi(\mathrm{mod}(i, k) + 1)$;
  end for
  return vector $Index$;

Algorithm 2: SELECT: Random selection of n objects among a set of size k.

with

$$\sigma^2 \gtrsim p_{\mathrm{fa}}^2\left((N-1)\frac{1-p}{p} + \frac{1-p_N}{p_N}\right). \tag{6}$$

We can also show that, in the asymptotic regime, the bias decreases inversely proportional with n:

$$E\left[\frac{\hat{p}_{\mathrm{fa}} - p_{\mathrm{fa}}}{p_{\mathrm{fa}}}\right] = \frac{1}{n}\frac{N(1-p)}{p} + o(n^{-1}), \tag{7}$$

which means that $E[(\hat{p}_{\mathrm{fa}} - p_{\mathrm{fa}})/p_{\mathrm{fa}}] \gtrsim \alpha n^{-1}$, where α is always a positive number. A remarkable fact is that the bias is positive, so that estimations tend to overestimate the probability of rare event. In concrete situations, the rare event often corresponds to a catastrophic scenario to be prevented, and overestimating is then a nice property.

2.5. Some improvements

This subsection proposes some improvements of Algorithm 1 . If n is huge, the subroutine HIGHER_SCORE must be carefully implemented. In general, an efficient way is to use the quick-sort algorithm whose complexity is on average $O(n \log n)$. If $n = 2k$, a median algorithm is recommended because its complexity is on average $O(n)$.

The most restrictive part is that, so far in Algorithm 1, k must divide n; the k selected vectors $\{\mathbf{y}_i\}_1^k$ are each replicated n/k times (line 14). To take over this restriction, we create a subroutine SELECT which randomly picks up n vectors from the set $\{\mathbf{y}_i\}_1^k$. Each $\mathbf{y}_i$ is thus replicated a random number of times, whose expectation is n/k. Algorithm 2 shows an implementation, where each vector is at least selected once, and where it is selected a constant number of times if k divides n.

3. APPLICATION TO ZERO-BIT WATERMARKING

This part first applies the method to a well-known watermarking detector for which there exist bounds and a numerical method to derive the probability of false alarm. This allows to benchmark the method and to fine-tune its parameters.

We have selected the absolute value of the normalized correlation [13] as the score function, so that $\mathbf{x}$ is deemed watermarked if

$$d(\mathbf{x}) = \frac{|\mathbf{x}^T\mathbf{u}|}{\|\mathbf{x}\|} > \tau, \tag{8}$$

where $\mathbf{u}$ is a secret vector whose norm equals one. A geometrical interpretation shows that the acceptance region is a two-sheet hypercone whose axis is given by $\mathbf{u}$ and whose angle is $\theta = \cos^{-1}(\tau)$ (with $0 < \theta < \pi/2$). Hence, for an isotropic distribution whose probability density function only depends on the norm of $\mathbf{x}$, the probability of false alarm is the proportion of the solid angle of the hypercone compared to the solid angle of the full space $p_{\mathrm{fa}} = I_L(\theta)/I_L(\pi/2)$, with $I_L(\theta) = \int_0^\theta \sin^{L-2}(u)du$ ($L \geq 2$). The authors of [9] derived a simple program numerically calculating the probability of false alarm based on iterative integration by part. Yet, when implemented in MATLAB or standard C, this program fails calculating very weak probabilities, due to computer precision limit. For this reason, they proposed an upper and lower bound, respectively, based on a Gaussian and a Fisher Z-statistic approximations:

$$2\,\mathrm{erfc}\left(\frac{1}{2}\log\frac{1+\tau}{1-\tau}\sqrt{L-2}\right) \leq p_{\mathrm{fa}} \leq 2\,\mathrm{erfc}(\tau\sqrt{L}). \tag{9}$$

Indeed, a much better way is to resort to the Fisher-Snedecor F-distribution. If $\mathbf{x}$ belongs to the acceptance region, then the angle $\langle \mathbf{x}, \mathbf{u}\rangle$ is smaller than θ or bigger than $\pi - \theta$. Instead of a cosine, we translate this in term of tangent: $\tan^2(\langle \mathbf{x}, \mathbf{u}\rangle) < \tan^2\theta$, with $\tan^2(\langle \mathbf{x}, \mathbf{u}\rangle) = \|(\mathbf{I}-\mathbf{P})\mathbf{x}\|^2/\|\mathbf{P}\mathbf{x}\|^2$. Here, $\mathbf{P}$ is the projector matrix: $\mathbf{P}\mathbf{x} = (\mathbf{x}^T\mathbf{u})\mathbf{u}$, and $\mathbf{I}$ the $L \times L$ identity matrix. We suppose that $\mathbf{x}$ is a white Gaussian noise, as an example of isotropic distribution: $p_{\mathbf{X}} = \mathcal{N}(\mathbf{0}, \mathbf{I}_L)$. Therefore, its projections on complementary subspaces $\mathbf{P}\mathbf{x}$ and $(\mathbf{I} - \mathbf{P})\mathbf{x}$ are independent and Gaussian distributed. This implies that $\|\mathbf{P}\mathbf{x}\|^2$ and $\|(\mathbf{I} - \mathbf{P})\mathbf{x}\|^2$ are chi-square with one and $L - 1$ degrees of freedom, respectively, and that the random variable $F = \|(\mathbf{I} - \mathbf{P})\mathbf{x}\|^2/(L-1)\|\mathbf{P}\mathbf{x}\|^2$ has a Fisher-Snedecor F-distribution. Consequently, $p_{\mathrm{fa}} = \mathrm{Prob}[F < (L-1)^{-1}\tan^2\theta]$. This is by definition the cumulative distribution function whose expression relies on incomplete beta function. After simplifications, we have

$$p_{\mathrm{fa}} = I\left(1 - \tau^2; \frac{L-1}{2}, \frac{1}{2}\right), \tag{10}$$

with $I(x; a, b)$ the regularized incomplete beta function. MATLAB or Mathematica provides far more accurate approximations than those proposed in [9].

The random vector $\mathbf{x}$ being a white Gaussian noise, the replication process modifies $\mathbf{x}$ into $\mathbf{z} = (\mathbf{x}+\mu\mathbf{n})/\sqrt{1+\mu^2}$, with $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_L)$. Hence, $\mathbf{z}$ is still a white Gaussian noise with unit variance. This defines the GENERATE and MODIFY processes in Algorithm 1.

3.1. Experimental investigation number 1: strength of the replication

The main shortcoming of our algorithm is that the parameter μ needs a manual fine-tuning. The algorithm as described above works fine for the problem studied in this section when $\mu = 0.2$. This value sets the strength of the replication process, which can be expressed as the expected square distance between the modified vector $\mathbf{z}$ and its initial value $\mathbf{x}$. Here, this square distance is equal to $2L(1-(1+\mu^2)^{-1/2})$. The strength of the replication fixes the dynamic of the system.

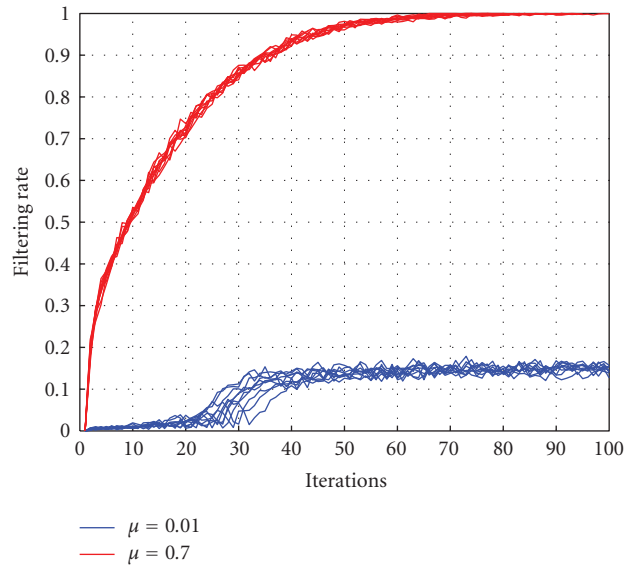

Figure 1: Filtering rate for 10 estimator runs with $\mu = 0.7$, and 10 estimator runs with $\mu = 0.01$.

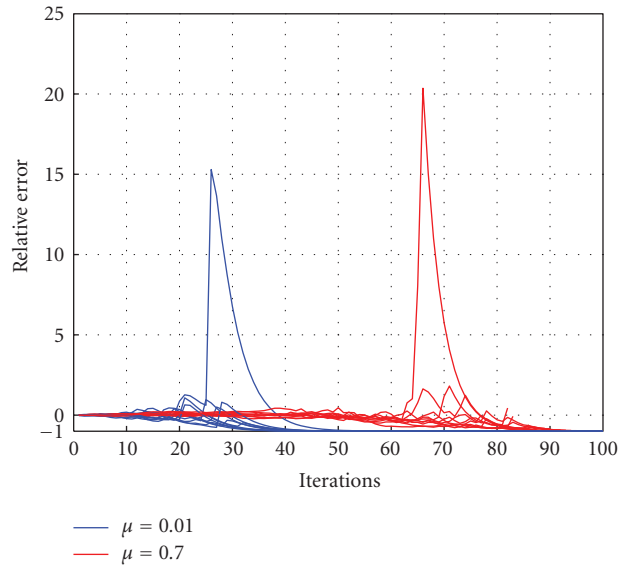

Figure 2: Relative errors for the same estimator runs as used in Figure 1.

There is a tradeoff to be found between two undesirable effects. The goal of this subsection is to experimentally show and explain these two effects and to find a trick to circumvent this manual fine-tuning shortcoming. The others parameters are set as follows: $L = 20$, $\tau = 0.95$, and $n = 6400$. This gives $p_{\text{fa}} = 4.704 * 10^{-11}$. We have found that $\mu = 0.2$ is a correct value because the algorithm behaves as expected. However, a greater or a lower value have negative impacts as we will see.

As the estimator goes, the set $\mathcal{A}_i$ is smaller and smaller, and the modification process is more and more likely to move vectors out of this set when the strength is too big. Let us define the filtering rate of the replication process as the number of times a modification is refused divided by n. Figure 1 shows this filtering rate along the iteration number. Typically, a factor μ greater than 0.5 (red curves) yields a filtering rate of 100% for the last iterations. This implies that the particules and their scores are not renewed any longer. Thus, threshold τ_j saturates and the algorithm does not converge. It stops thanks to the constraint on the maximum number of iterations $N_{\max} = 100$. We seize the opportunity of this case study where the true map $p = F(\tau)$ is known to plot the relative error along the ROC curve $(p^j - F(\tau_j))/F(\tau_j)$ in Figure 2. Red curves were simulated for a strong replication strength: $\mu = 0.7$. We observe that, when the filtering rate is too high, the relative error has a peak followed by an exponential decay towards -1. The peak is explained by the fact that the vectors and their scores are no longer renewed, so that the thresholds quickly converge towards the supremum of these scores. Once the thresholds saturate to this supremum, $F(\tau_j)$ became fixed, and the relative error has an exponential decay due to the term p^j. When this latter becomes negligible compared to $F(\tau_j)$, the relative error tends to -1.

The impact of a small μ is not noticeable in the filtering rate which is far below the saturation phenomenon (see Figure 1). Yet, Figure 2 shows very strong relative errors

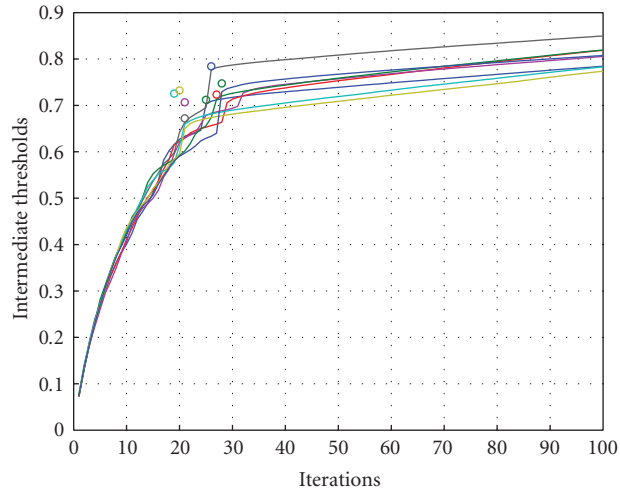

Figure 3: Typical evolution of the intermediate thresholds when μ is too small. Intermediate thresholds are taken from the same estimator runs as used in Figure 1 with $\mu = 0.01$.

(blue curves) in the first iterations. Factor μ is so weak that replicated particules are almost located at the same place as the previous ones. This prevents us from exploring the space due to a low dynamic and from moving the vectors towards the acceptance region. Hence, the scores of the replicated particules are almost the same scores than the previous ones. This is almost as if $\mu = 0$, that is, classical Monte Carlo. The behavior of the relative error is then strongly dependent on the initialization process which yields the first set of particules. The selection process keeps a thinner and thinner portion $(k/n)^j$ of this initial cloud of particules and the intermediate threshold converges to the maximum of the initial scores. Once this is achieved, the intermediate thresholds saturate to this maximum value, and we again

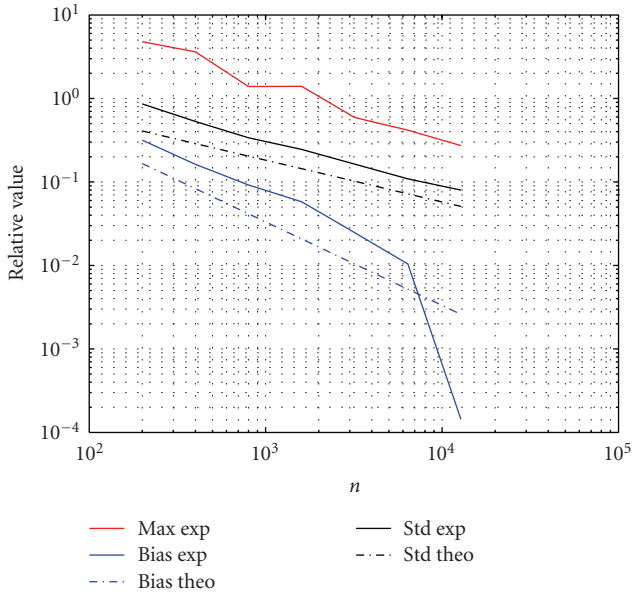

Figure 4: Statistics measured for 1000 estimation runs with $k/n = 1/2$.

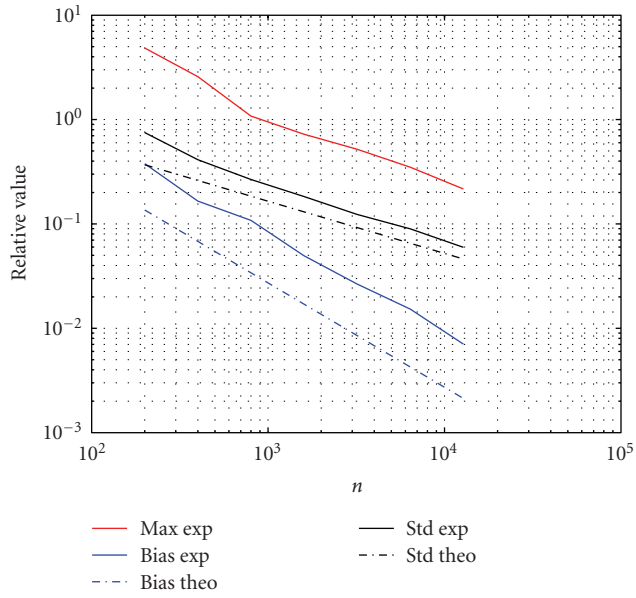

Figure 5: Statistics measured for 1000 estimation runs with $k/n = 3/4$.

observe an exponential decay toward -1 (Figure 2—blue curves). To check our explanations, we plotted the values of the intermediate thresholds in Figure 3. For the iteration number giving a maximum of relative error in Figure 2, we plot a circle centered on the maximum score of the initial particles. This illustrates the dependence between this maximum initial score and the saturated intermediate threshold.

The best tradeoff can be stated in the following terms: find the maximum value of μ such that the filtering rate is below a given level. We modify Algorithm 1 as follows: μ is set to one at the beginning. For each iteration, we measure the filtering rate. If this latter is bigger than the level, we reduce the value of μ and repeat the iteration until the filtering rate is below the level. The value of μ is thus now found adaptively. However, the number of detection trials is no longer fixed. Experimentally, we decrease μ by a factor 1.1 anytime the filtering rate is above 0.7.

3.2. *Experimental investigation # 2:* $p = k/n$

Parameter p strikes a tradeoff between the speed and the accuracy of the estimator. Equation (4) tells us that the lower p is, the faster is the estimation of p_{fa}. However, (6) and (7) show that the relative variance and the bias are decreasing functions of p.

The experimental set up is the following: $L = 20$, $\tau = 0.95$, $\mu = 0.2$, and $p_{\text{fa}} = 4.704 * 10^{-11}$. We try two values for p: 3/4 and 1/2. We run 1000 estimations $\{\widehat{p}_{\text{fa}}^{(i)}\}$ to measure the relative bias as $(\text{Mean}(\{\widehat{p}_{\text{fa}}^{(i)}\}) - p_{\text{fa}})/p_{\text{fa}}$, the relative standard deviation $\text{Std}(\{\widehat{p}_{\text{fa}}^{(i)}\})/p_{\text{fa}}$, and the relative maximum deviation $(\text{Max}(\{\widehat{p}_{\text{fa}}^{(i)}\}) - p_{\text{fa}})/p_{\text{fa}}$. Figures 4 and 5 plots these values against the number of particles n.

Observe first the excellence of the estimator. $n = 12\,800$ particles (last point on curves) represent around 1000 000 detection trials for $p = 3/4$ or around 430 000 for $p = 1/2$. Any estimation yielded a result between $4.0 * 10^{-11}$ and $5.7 * 10^{-11}$ with $p = 3/4$, or between $3.6 * 10^{-11}$ and $6.0 * 10^{-11}$ with $p = 1/2$. The relative standard deviation represents less than 10%. A classical MC estimator would need more than $2 \cdot 10^{12}$ detection trials to achieve such a precision.

Table 1: Anderson-Darling test on 1000 estimation runs for $p = 1/2$ and $p = 3/4$. The hypothesis of Gaussian distribution is accepted when the score is lower than 0.752 with a significance level of 0.05.

n	200	400	800	1600	3200	6400	12 800
$p = 1/2$	32.40	12.46	5.35	4.42	2.87	1.82	0.38
$p = 3/4$	23.53	9.16	2.96	1.76	1.46	0.50	0.53

Surprisingly enough, the measured variance and bias follow the laws (6) and (7) known for the asymptotic regime even for a low number of particles. The bias is not measured with enough precision with only 1000 trials for $n = 12\,800$ because its order of magnitude is 0.001 times the value of p_{fa}. Yet, the asymptotic regime is only achieved if the estimations are Gaussian distributed. An Anderson-Darling test [14] reveals that this is the case only for the biggest values of n. This happens quicker for p closer to one according to the scores of Table 1: estimations $\{\widehat{p}_{\text{fa}}^{(i)}\}$ are deemed Gaussian distributed when n equals 6400 for $p = 3/4$ whereas this hypothesis is clearly rejected for $p = 1/2$.

Our conclusions of this experiment are the following. There are two typical use cases of our algorithm. If the user looks for the order of magnitude of the probability to be estimated, then the choice $p = 1/2$ with around $n = 2000$ particules gives a fast estimation (around 68 000 detection trials). This is especially true since the variance (6) and the bias (7) are not drastically bigger than the ones for $p = 3/4$.

If the issue is to assess an estimation with a given accuracy and confidence range, then the estimator must be in the asymptotic regime where the pdf of the estimation error is known. This experiment shows that a ratio 3/4 (i.e., closer to one) is advised. Each estimation lasts longer but, in the end, this is the quickest way to achieve the asymptotic regime.

This experimental work also stresses what we still ignore; the standard deviation and the bias are in practice bigger than the theoretical expressions. This is normal as these latter are theoretical lower bounds. However, we ignore the scaling factor. Other experiments showed that it does not depend on L. We suppose however that there is a strong relationship with the detection score function. This prevents us from establishing confidence ranges supported with the Gaussian distribution in the asymptotic regime. This strategy implies a heavy experimental work, where a hundred of estimations are needed in order to first confirm the asymptotic regime, and second, to estimate the standard deviation. Then, the probability of false alarm is lower than $\text{Mean}(\{\widehat{p}_{\text{fa}}^{(i)}\}) + 2*\text{Std}(\{\widehat{p}_{\text{fa}}^{(i)}\})$ with a confidence of 97.7%.

A faster way to yield a confidence interval is to observe the number of iterations of several independent estimations. For $p = 1/2$ and $n \geq 800$, more than two thirds of the estimations end at $N = 34$ iterations (see Figure 6), which gives a confidence interval of $[p^N, p^{N+1}] = [2.91, 5.82]*10^{-11}$. For $p = 3/4$ and $n \geq 1600$, more than two thirds of the estimations end at $N = 82$ iterations (see Figure 7), which gives a confidence interval of $[p^N, p^{N+1}] = [4.26, 5.69]*10^{-11}$. Once again, a bigger p provides more accurate results but at the cost of slower estimations.

3.3. Error exponents for zero-rate watermarking scheme

A watermarking scheme is deemed as sound if its probability of false alarm and its probability of false negative decrease exponentially with the dimension L of the signals under an embedding power constraint. Within this class, the comparison of two watermarking schemes can be based on their exponential decreasing rates, that is, their error exponents defined as follows:

$$E_{\text{fa}}(\tau) = -\lim_{L\to+\infty} \frac{1}{L} \log\, p_{\text{fa}},$$
$$E_{\text{fn}}(\tau) = -\lim_{L\to+\infty} \frac{1}{L} \log\, p_{\text{fn}}. \tag{11}$$

There are very few watermarking schemes where error exponents have closed form expressions [13]; for instance, the additive spread spectrum with a single nappe hypercone detection region, the improved sign embedder with a dual nappe hypercone detection region. Furthermore, these theoretical expressions do not foresee a noisy channel (i.e., attack) to calculate $E_{\text{fn}}(\tau)$. In practice, it is extremely hard to estimate these error exponents because huge values of L should imply very very low probabilities of errors if the watermarking scheme is good. This is no more a problem with our algorithm, and we simply estimate the error exponents by

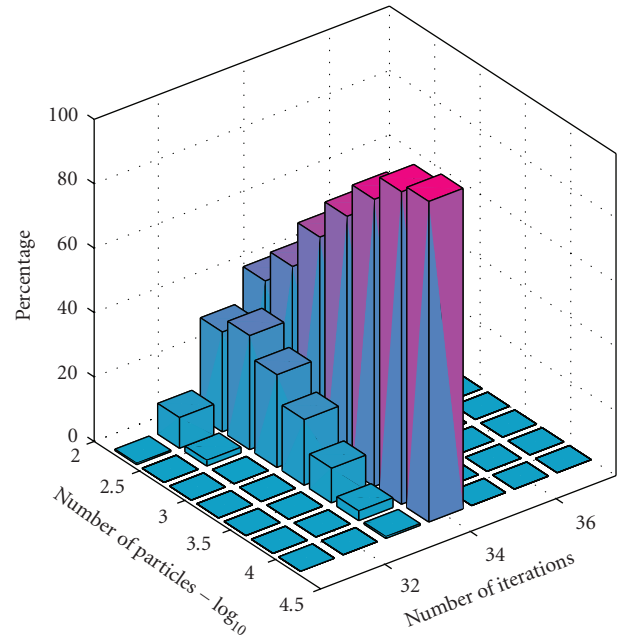

Figure 6: Confidence intervals are smaller as number of particles increases. Percentage of estimations over 1000 runs for $k/n = 1/2$.

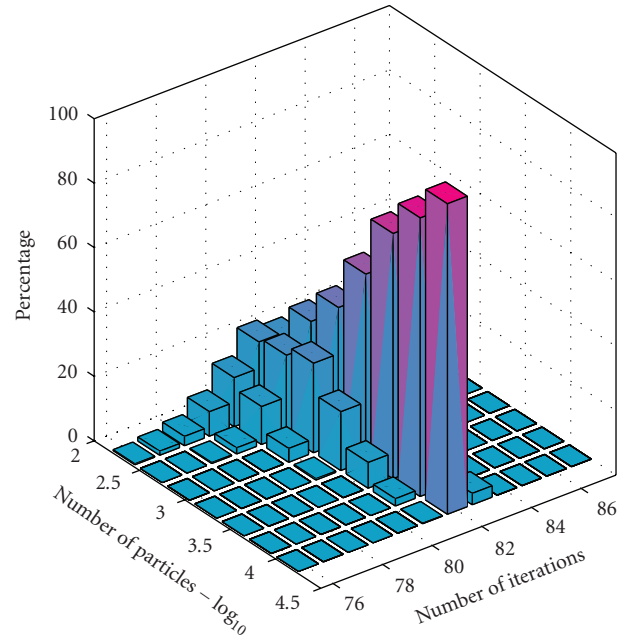

Figure 7: Confidence intervals are smaller as number of particles increases. Percentage of estimations over 1000 runs for $k/n = 3/4$.

$\widehat{E}_{\text{fa}}(\tau) = -\log \widehat{p}_{\text{fa}}/L$ and $\widehat{E}_{\text{fn}}(\tau) = -\log \widehat{p}_{\text{fn}}/L$ with a given big enough L.

For the false negative, the rare event is that a watermarked (and possibly attacked) vector has a score below a small threshold. At each step, the estimator sets τ_j as the kth highest scores. Hence, the intermediate thresholds are indeed decreasing. We can also study the impact of an attack on E_{fn}

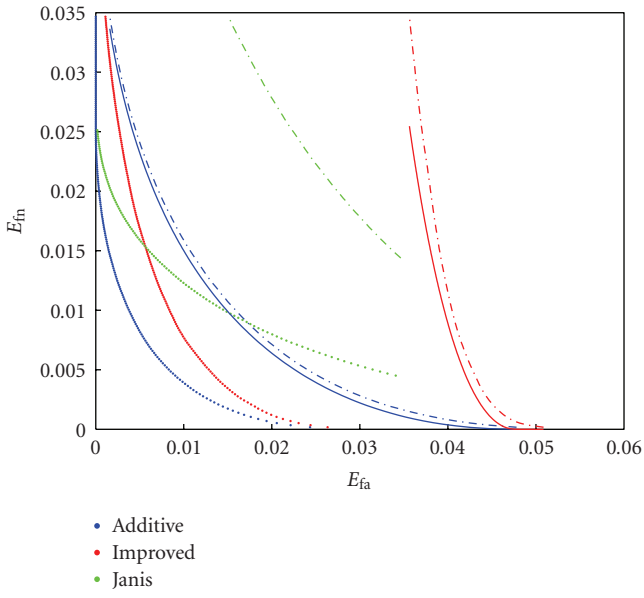

FIGURE 8: Error exponent experimental measurement. E_{fn} against E_{fa}. Solid line: theoretical curves (without noise). Dash-dot line: experimental curve (without noise). Dotted line: experimental curve (with AGWN $\sigma_N^2 = 10 \cdot P_e$).

as soon as the attack vector $\mathbf{n}$ has a statistical model with the two following properties:

(i) we are able to generate vectors distributed as $p_{\mathbf{N}}$,

(ii) there exists a modification process with a controllable strength that lets this distribution invariant.

Then, a particle is now a couple of vectors $\{\mathbf{x}, \mathbf{n}\}$, and its score is the detection function applied to the attacked and watermarked vector: $d(\mathbf{z}) = d(\mathbf{w}(\mathbf{x}) + \mathbf{n})$, where $\mathbf{w}(\cdot) : \mathbb{R}^L \mapsto \mathbb{R}^L$ is the watermark embedding function. The replication process changes both vectors of the particle, each one with its law invariant modification process. Another technical detail is that our algorithm is run only once, storing the intermediate thresholds in order to estimate the mapping $\{\widehat{E}_{\text{fn}}(\tau_j), \tau_j\}$. The same holds for the false alarm error exponents $\{\widehat{E}_{\text{fa}}(\tau'_j), \tau_{j'}\}$. An interpolation finally gives $\{\widehat{E}_{\text{fa}}(\tau_j), \widehat{E}_{\text{fn}}(\tau_j)\}$.

The experimental setup is the following: $L = 4000$, host vectors are Gaussian distributed with variance $\sigma_X = 1$. The embedding power equals $P_e = 0.1$. We test three watermarking schemes: additive spread spectrum scheme with $d(\mathbf{x}) = \mathbf{x}^T\mathbf{u}/\|\mathbf{x}\|$, "improved" sign embedder with $d(\mathbf{x}) = |\mathbf{x}^T\mathbf{u}|/\|\mathbf{x}\|$ as detailed in [13], and the JANIS scheme with order 2 [8]. For the first two schemes, the relationship between $E_{\text{fa}}(\tau)$ and the threshold τ is perfectly known [13]. However, there is no expression for $E_{\text{fn}}(\tau)$ under an attack (here a Gaussian white noise with variance $\sigma_N^2 = 0.1$). For the JANIS scheme, we have to estimate both $E_{\text{fa}}(\tau)$ and $E_{\text{fn}}(\tau)$. Figure 8 shows the results.

From an experimental point of view, the measurements are good with only a small inaccuracy. We blame two shortcomings. L is not big enough and the ratio $L^{-1} \log p_{\text{fa}}$ (*idem* with the false negative exponent) does not reflect the rate of the exponential decay. A better way would be to estimate $\log(p_{\text{fa}})$ for several values of L and to estimate the exponent with a linear regression. Second, these plots were obtained very rapidly with our algorithm working with only a $n = 3200$ vectors stack, and $k = n/2$. Therefore, the accuracy of the estimation of p_{fa} itself is not at best, but, we are indeed interested in showing that error exponents can be measured very rapidly; the experimental curves for the additive and improved watermarking scheme have the right shape (in particular for the improved scheme, E_{fn} goes to infinity when E_{fa} goes to zero). In the same way, the range of the measurements is limited by the maximum number of iterations allowed, which is here set to 200.

From a watermarking point of view, it is quite difficult to announce which scheme performs better. All of them share the same detection complexity. The improved scheme has the advantage of an infinite E_{fn} when there is no attack. JANIS performances curve seems to be better only at high E_{fa}. Yet, performances of course collapse with the presence of an attack, but JANIS seems to be more robust of the three compared schemes.

4. APPLICATION TO PROBABILISTIC FINGERPRINTING CODE

Fingerprinting is the application where a content server gives personal copies of the same content to n different buyers. c of them are dishonest users, called colluders, who mix their copies to yield a pirated content. A binary fingerprinting code is a set of N different m bit sequences $\{\mathbf{x}_i\}_{i\in[N]}$. Each sequence identifying a user has to be hidden in the personal copy with a watermarking technique. When a pirated copy is found, the server retrieves an m bit sequence and accuses some users or nobody. There are two kinds of errors: accusing an innocent (i.e., a false alarm), and accusing none of the colluders (i.e., a false negative). The designers of the fingerprinting code must assess the minimum length of the code so that the probabilities of error are below some significance levels: $p_{\text{fa}} < \epsilon_1$ and $p_{\text{fn}} < \epsilon_2$. One of the best fingerprinting codes is a probabilistic code proposed by Tardos [15], where $m = O(c^2 \log(1/\epsilon_1))$. Before Tardos' work, the existence of such a short code was only theoretically proven. Tardos is the first to exhibit a construction which is, moreover, surprisingly simple. The main point of interest for us is that the accusation is based on the calculus of scores and their comparison to a threshold. Consequently, this fingerprinting code is very well suited with respect to our algorithm.

4.1. New accusation strategy

In Tardos probabilistic fingerprinting code, the accusation is focused; the detection decides whether a given user is guilty or not. It calculates his score from the code sequence of the user and the sequence recovered in the pirated copy. The user is deemed guilty when his score is higher than a threshold. The size of the collusion c, the probabilities ϵ_1 and ϵ_2 are the

inputs of the code. The outputs are the code length m and the value of the threshold T.

We think that this approach is not adapted in practice. We believe that the length of the code sequence to be embedded in content is not tunable but fixed by the payload of the watermarking technique and the length of the content. It is clear that the longer the sequence is, the better the accusation process works. But, in practice, there is certainly a wide range in the length of the sequences to be embedded due to a wide diversity of contents. In the same way, it might be complicated to derive the right value of the threshold for different sequence lengths. We propose a different approach. Once we have recovered the sequence $\mathbf{y}$ in the pirated copy, we calculate all the scores of the users to which the content has been delivered and accuse the most likely guilty users, that is, the ones with the highest scores. In the sequel, consider that user j is accused because $j = \arg\max_{i\in[N]} S_i$. There is no longer need of a threshold. However, we cannot guaranty a probability ϵ_1. To be fair, the output of our accusation process is the index j of the most likely guilty user associated with the probability of making an error, that is, the probability that an innocent gets a score bigger or equal to S_j knowing the pirate sequence $\mathbf{y}$: $\text{Prob}(\texttt{SCORE}(\mathbf{x}, \mathbf{y}) > S_j | \mathbf{y})$.

We use our algorithm to estimate this probability where $\mathbf{x}$ is distributed as a code sequence. Particules in this framework are now binary sequences of length m. The `GENERATE` and the `SCORE` functions are given by the construction of the code and its accusation method [15]. One very important fact is that the symbols in a code sequence are statistically independent and distributed according to their own law. The `MODIFY` function is thus very simple: we randomly select a fraction μ of them (parameter μ sets the replication strength), and regenerate them according to their own law. These nondeterministic changes leave the distribution of the code sequence invariant.

4.2. Code length

We now come back to the original Tardos focused accusation process. Typical papers about Tardos code [16–18] aim to find the tightest lower bound of the length, that is, to find the lower constant A so that $m > Ac^2 \log(1/\epsilon_1)$ implies that the constraints on the probabilities of error are fulfilled. In the original Tardos' paper, ϵ_2 is set to $\epsilon_1^{c/4}$, and $A = 100$ for his asymmetric code. In our experiments, we use the symmetric version of Tardos code as proposed by Škorić et al. [17] where $A = 50$ thanks to the symmetry. The goal of this subsection is to experimentally find this constant A, which is a priori challenging because ϵ_1 is very low.

The experiment is twofold. First, we need to estimate the plot mapping ϵ_1 and the threshold T, for different couples (c, m). We generate c code sequences of length m. The collusion strategy is to randomly pick up the symbols of pirated copy $\mathbf{y}$ among the c colluders' sequences. Then, we estimate the curve $T = F^{-1}(\text{Prob}(\texttt{SCORE}(\mathbf{x}, \mathbf{y}) > T | \mathbf{y}))$ with our algorithm. Indeed, the target threshold is fixed to a very high value so that the algorithm stops after $N_{\max}$ iterations. The obtained mapping is indeed $N_{\max}$ couples $(T_j, (k/n)^j)$. However, this holds for a special occurrence of $\mathbf{y}$. Therefore, we need to integrate this conditional probability by integrating it over K different sequences $\{\mathbf{y}_i\}_{i\in[K]}$. Each time, c code sequences are drawn independently and uniformly to forge a pirated sequence. The jth threshold is averaged over the K estimates.

The second part of the experiment measures the false negative probability ϵ_2. A particle is now a set of c Tardos sequences $\{\mathbf{x}_1, \ldots, \mathbf{x}_c\}$ plus an allocation sequence $\mathbf{a}$ which dictates the way to produce $\mathbf{y}$ from the c sequences: $y(k) = x_{a(k)}(k)$ for all $k \in [m]$. The indices stored in $\mathbf{a}$ are independent and identically distributed over $[c]$. The `MODIFY` function is twofold. It independently modifies the c Tardos sequences as described above, and it modifies the allocation sequence by randomly selecting a fraction μ of indices and draw them against the uniform law over $[c]$.

The `SCORE` function is more complicated. From a particle, it generates the pirated sequence $\mathbf{y}$ thanks to its allocation sequences, and calculates the c accusation sums. The score of the particle is then their mean or their maximum. Tardos and his followers based their analysis on the mean of the scores of the colluders because this leads to tractable equations. The rationale is that if the mean is above the threshold, then there is at least one colluder whose score is higher than the threshold. However, the probability that the mean is below the threshold T is a very rough estimate of the probability of false negative ϵ_2. We choose to follow Tardos' choice of the mean to appreciate the refinement about the constant A given by our experimental investigation compared to the constants found by Tardos and his followers via Chernoff bounds. However, we can also set the score of a particle as the maximum of the c accusation sums in order to really estimate ϵ_2 as the probability that none of the c colluders gets caught.

The rest of the second part works like the first part. We are interested by estimating the mapping $T = F^{-1}(\text{Prob}(\max_{i\in[c]} \texttt{SCORE}(\mathbf{x}_i, \mathbf{y}) < T))$ (max or mean) using our algorithm. The experiment is run K times, and the intermediate thresholds are averaged for a better precision. Then, we plot in the same figure (see Figure 9) the false positive mapping and the false negative mapping, except that for this latter one, the probability is taken to the power $4/c$ to provide a fair comparison to previous evaluations of constant A where ϵ_2 was set to $\epsilon_1^{c/4}$. The intersection of the two mappings at a point $(T_0(m, c), \epsilon_0(m, c))$ implies that it is possible to find a threshold such that $\epsilon_1 = \epsilon_0(m, c)$ and $\epsilon_2 = \epsilon_0(m, c)^{c/4}$. This value indeed reflects the best we can do given m and c. We cannot achieve a lower significance level while preserving the relationship $\epsilon_2 = \epsilon_1^{c/4}$ because it does not exist any threshold fulfilling this constrain at a lower significance level than $\epsilon_0(m, c)$. The only way to get lower significance levels is to increase the code length for a given collusion size.

Several experimentations have been carried on with $m \in \{100, 150, 200, 300, 400, 600\}$ and $c \in \{2, 3, 4\}$ to obtain different values of $\epsilon_0(m, c)$. The final plot draws m against the function $c^2 \log \epsilon_0(m, c)^{-1}$, see Figure 10. Some comments are in order. The curves are surprisingly straight so that the length of the code is really asymptotically equal to $Ac^2 \log \epsilon_1^{-1}$. The constraints on the significance levels are

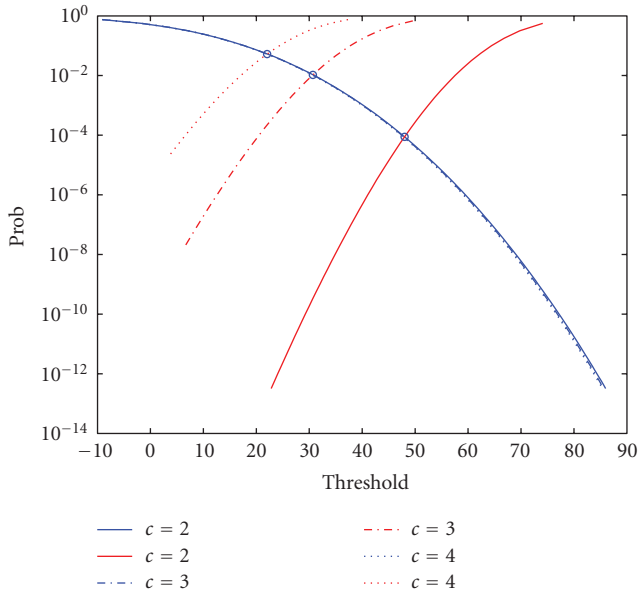

FIGURE 9: Mappings of the false positive probability (blue) and false negative probability (red) against the threshold. $m = 200$, $c \in \{2, 3, 4\}$. The score of a particle is the mean of the c colluders scores.

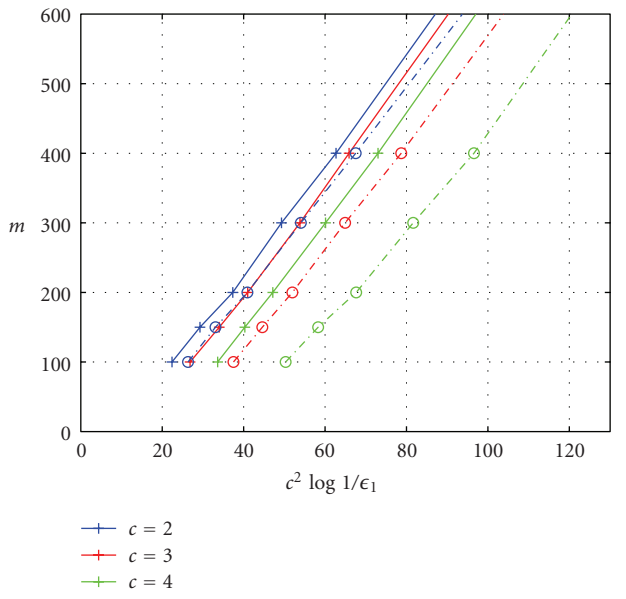

FIGURE 10: Code length needed to obtain $\epsilon_1 = \epsilon_2^{c/4}$ for $c = 2$, 3, or 4 colluders, against function $c^2 \log \epsilon_1^{-1}$. ϵ_2 has been estimated with the mean of the colluders' scores (solid lines) or their maximum (dotted lines).

fulfilled if $A \geq 8.0$ (resp., $A \geq 7.6$), when we based our estimation of the false negative on the mean (resp., maximum) of the colluders score. The code we used being symmetric, Tardos equivalent was $A = 50$ and Škorić et al. found $A = 2\pi^2 \approx 19.7$ with a Chernoff bound, or $A = \pi^2 \approx 9.9$ with a Gaussian distribution approximation of the accusation sums. The differences between evaluations based on mean or maximum are bigger when the size of collusion increases, which is not a surprise. The mean based evaluations are overestimating the power of the colluders. For a given threshold T, it is much more difficult to forge a pirated copy such that the maximum of the accusation sums is below T, than to forge a pirated copy such that the mean of the accusation is below T. This has two consequences. The coefficient A is lower when its estimation is based on the maximum. In the nonasymptotical regime, the code length is estimated by $m = Ac^2 \log \epsilon_1^{-1} + B$. The offset B is lower when working with the maximum. For instance, estimations based on maximum give codes shorter of around 100 symbols for $c = 3$. This is quite substantial in practice.

5. CONCLUSION

We have presented an algorithm estimating probabilities of rare events. It works for a kind of rare events defined via a scalar score being above a threshold. The algorithm is far more powerful than the classical Monte Carlo estimator because it provides much more accurate estimations while needing much less runs of the rare event detector (or less calculus of score function). Several runs of this estimator allow to rapidly give small confidence intervals.

This algorithm is very well suited for watermarking issues because errors (false positive or false negative) belong to this kind of rare event. Hence, this algorithm has been shown very useful to evaluate probability of false or error exponent in zero-bit watermarking, probability of accusing an innocent user and probability of missing a colluder in a fingerprinting scenario. This helps setting a fingerprinting code length to ensure that these probabilities are below a significance level. This is also very useful to propose a global accusation process which outputs the most likely dishonest user while estimating the probability that this user is indeed innocent.

ACKNOWLEDGMENT

This work is supported in part by the French national program "Securité ET INformatique" under project NEBBIANO, ANR-06-SETIN-009. This work has been patented.

REFERENCES

[1] "Copy protection technical working group," http://www.cptwg.org/.

[2] A. Barg, G. R. Blakley, and G. A. Kabatiansky, "Digital fingerprinting codes: problem statements, constructions, identification of traitors," *IEEE Transactions on Information Theory*, vol. 49, no. 4, pp. 852–865, 2003.

[3] T. Furon, B. Macq, N. Hurley, and G. Silvestre, "JANIS: just another n-order side-informed watermarking scheme," in *Proceedings of IEEE International Conference on Image Processing (ICIP '02)*, vol. 2, pp. 153–156, Rochester, NY, USA, September 2002.

[4] M. L. Miller and J. A. Bloom, "Computing the probability of false watermark detection," in *Proceedings of the 3rd International Workshop on Information Hiding (HI '99)*, A. Pfitzmann, Ed., pp. 146–158, Springer, Dresden, Germany, September 1999.

[5] J. Galambos, *Advanced Probability Theory*, vol. 10 of *Probability: Pure and Applied*, Marcel Dekker, New York, NY, USA, 2nd edition, 1995.

[6] F. Pérez-González, F. Balado, and J. R. Hernández Martin, "Performance analysis of existing and new methods for data hiding with known-host information in additive channels," *IEEE Transactions on Signal Processing*, vol. 51, no. 4, pp. 960–980, 2003.

[7] P. Comesaña, *Side-informed data hiding: robustness and security analysis*, Ph.D. thesis, Universidade de Vigo, Vigo, Spain, 2006.

[8] J.-P. Linnartz, T. Kalker, and G. Depovere, "Modelling the false alarm and missed detection rate for electronic watermarks," in *Proceedings of the 2nd International Workshop on Information Hiding (IH '98)*, vol. 1525 of *Lecture Notes in Computer Science*, pp. 329–343, Springer, Portland, Ore, USA, April 1998.

[9] V. Solachidis and I. Pitas, "Optimal detector for multiplicative watermarks embedded in the DFT domain of non-white signals," *EURASIP Journal on Applied Signal Processing*, vol. 2004, no. 16, pp. 2522–2532, 2004.

[10] G. Fishman, *Monte Carlo: Concepts, Algorithms and Applications*, Springer, New York, NY, USA, 1996.

[11] M. Villen-Altamirano and J. Villen-Altamirano, "RESTART: a method for accelerating rare event simulations," in *Proceedings of the 13th International Teletraffic Congress (ITC '91)*, pp. 71–76, Copenhagen, Denmark, June 1991.

[12] F. Cérou and A. Guyader, "Adaptive multilevel splitting for rare event analysis," *Stochastic Analysis and Applications*, vol. 25, no. 2, pp. 417–443, 2007.

[13] N. Merhav and E. Sabbag, "Optimal watermark embedding and detection strategies under limited detection resources," *IEEE Transactions on Information Theory*, vol. 54, no. 1, pp. 255–274, 2008.

[14] H. C. Thode Jr., *Testing for Normality*, vol. 164 of *Statistics: Textbooks and Monographs*, Marcel Dekker, New York, NY, USA, 2002.

[15] G. Tardos, "Optimal probabilistic fingerprint codes," in *Proceedings of the 35th Annual ACM Symposium on Theory of Computing (STOC '03)*, pp. 116–125, ACM, San Diego, Calif, USA, June 2003.

[16] B. Škorić, T. U. Vladimirova, M. Celik, and J. C. Talstra, "Tardos fingerprinting is better than we thought," *IEEE Transactions on Information Theory*, vol. 54, no. 8, pp. 3663–3676, 2008.

[17] B. Škorić, S. Katzenbeisser, and M. U. Celik, "Symmetric Tardos fingerprinting codes for arbitrary alphabet sizes," *Designs, Codes, and Cryptography*, vol. 46, no. 2, pp. 137–166, 2008.

[18] Koji Nuida, Satoshi Fujitsu, Manabu Hagiwara, Takashi Kitagawa, Hajime Watanabe, Kazuto Ogawa, and Hideki Imai, "An Improvement of Tardos's Collusion-Secure Fingerprinting Codes with Very Short Lengths," in *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, Lecture Notes in Computer Science, pp. 80–89, Springer, Berlin, Germany, 2007.