A Time Series Approach for Profiling Attack

Liran Lerman[1,2], Gianluca Bontempi[2], Souhaib Ben Taieb[2], and Olivier Markowitch[1]

[1] Quality and Security of Information Systems, Département d'informatique, Université Libre de Bruxelles, Belgium
[2] Machine Learning Group, Département d'informatique, Université Libre de Bruxelles, Belgium

Abstract. The goal of a profiling attack is to challenge the security of a cryptographic device in the worst case scenario. Though template attack is reputed as the strongest power analysis attack, they effectiveness is strongly dependent on the validity of the Gaussian assumption. This led recently to the appearance of nonparametric approaches, often based on machine learning strategies. Though these approaches outperform template attack, they tend to neglect the potential source of information available in the temporal dependencies between power values. In this paper, we propose an original multi-class profiling attack that takes into account the temporal dependence of power traces. The experimental study shows that the time series analysis approach is competitive and often better than static classification alternatives.

Keywords: side-channel attack, power analysis, machine learning, time series classification.

1 Introduction

Embedded devices such as smart cards, mobile phones, and RFID tags are widely used in our everyday lives. These devices implement cryptographic operations allowing to secure, for example, bank transfers, buildings and cars. A modern bank card embeds securely a secret information allowing *in fine* to transfer cash. This operation is allowed by the smart card when it receives the right PIN code. During the verification of the PIN code, a PIN-related information (associated to the right PIN) is processed by the device. This secret information could be retrieved by physical attacks that analyze the power consumption [24], the processing time [23], or the electromagnetic emanation [15] of the device. In this work, we focus on attacks based on power consumption analysis. These attacks aim to infer the key-related information (label) from a time series of power measurements called trace.

Differential Power Analysis (DPA) [24] is an example of physical attack which first models the theoretic power consumption for each secret information. Then the real and the predicted power consumption are compared by using metrics, also known as distinguishers, such as the correlation coefficient [12], the difference of means [24], the mutual information [16], or the Kolmogorov-Smirnov Test [40].

The rationale is that the likelihood of a secret information is related to the degree of similarity between the predicted and the real power consumption.

Profiling attack (PA), and more precisely Template Attack (TA) [11], makes another step forward in the use of statistical modelling of power consumption; it estimates the conditional density function of the time series for each key-related information by using a Gaussian parametric model. Thereafter, the time series are classified under a maximum likelihood approach. If the assumption of gaussianity holds, it can be considered as the strongest power analysis in an information theoretic sense [11]. TA is particularly suitable (1) to analyze the security of a cryptographic device in the worst case scenario and (2) when the attacker is only able to observe a single use of the key (e.g. in stream ciphers).

In recent years the cryptographic community explored new approaches based on machine learning models. These methods demonstrate that template attacks underestimate the security of embedded devices. Lerman *et al.* [26, 27] compared a template attack with a binary machine learning approach, based on non-parametric methods, against a cryptographic device (FPGA Xilinx Spartan XC3s5000) implementing 3DES. In this work the authors dealt with a limited number of traces (between 125 and 256 samples) and a very high number of dimensions (between 6,000 and 10,000 points per trace) by adopting a robust dimensionality reduction methods. Hospodar *et al.* [20, 21] analyzed a software implementation of a portion of the AES algorithm. Their experiments support the idea that non-parametric techniques can be competitive and sometimes better (i.e. less number of traces in the attack phase) than state-of-the-art approaches when simplistic assumptions do not hold. Heuser *et al.* [19] generalized this idea by analyzing multi-class classification models in several contexts (e.g. varying the signal-to-noise ratio by an additional Gaussian noise, and varying the number of required traces in the attack phase to achieve a fixed guessing entropy). In the same year Bartkewitz *et al.* [3] applied a multi-class machine learning model allowing to improve the attack success with respect to the binary approach.

However, all the attacks proposed so far tend to disregard the potential source of information available in the temporal dependencies between power values. We aim to fill this gap by proposing an original multi-class profiling attack based on the adoption of a time series approach. The idea is to adopt a time series prediction algorithm (notably the Lazy Learning algorithm [1, 6, 9]) i) to characterize the temporal dependencies in the traces associated to each target value (related to a secret key) and ii) to design a classifier based on the temporal likelihood of the new traces.

We make a detailed assessment of the proposed approach by considering 6 datasets with different signal-to-noise ratios. The experimental results confirm that the classical template attack is not optimal in several contexts [3, 19–21, 26, 27, 29]. At the same time we show that our time series profiling attack is competitive (or better) with state-of-the-art approaches. Our interpretation is that the proposed method allows a more compact way to address the issue of large dimensionality. So far classification techniques in side-channel attack focus on a set of values associated to relevant parts of the trace. Given the noise and the large number of collected values, this demands the adoption of feature selection techniques which have to deal with a large dimensionality issue. This is no more the case in our approach where the time series is no more seen as a very large set of independent values but rather as an auto-regressive stochastic process which can be described by a low dimensional mapping.

This paper is organized as follows. Section 2 reviews the state-of-the-art of profiling attacks including the well-known template attack and the profiling attack based on machine learning classification models. Section 3 introduces our original attack based on time series modeling. Section 4 illustrate the power of our proposal with several datasets. We conclude the paper in Section 5.

2 Profiling Attack

2.1 Preliminaries

Let e be an encryption algorithm (a block-cipher) that transforms plaintext $m \in \mathcal{M}$ into ciphertext $c \in \mathcal{C}$ under the control of a secret key $O_i \in \mathcal{O}$ where $\mathcal{O} = \{O_1, O_2, ..., O_K\}$. More formally

$$\begin{aligned} e : \mathcal{O} \times \mathcal{M} \rightarrow \mathcal{C} \\ c = e_{O_i}(m) \end{aligned}$$

Let d be the decryption algorithm such that

$$\begin{aligned} d : \mathcal{O} \times \mathcal{C} \rightarrow \mathcal{M} \\ m = d_{O_i}(c) \end{aligned}$$

Traditional cryptanalysis techniques search relations between plaintexts, ciphertexts and the corresponding used keys. On the other hand, profiling attacks analyze the implementation of cryptographic operations. In particular they perform the worst-case security evaluation of cryptographic devices with the most powerful adversary in the information theoretic sense, by analyzing the relation between the leaked information (i.e. the power consumption) and the secret key O_i.

During the encryption, a function $f_{O_i}(m)$ called a sensitive variable [37] (f in short) is executed. Examples of this function are:

$$f_{O_i}(m) = O_i \tag{1}$$

$$f_{O_i}(m) = m \oplus O_i \tag{2}$$

$$f_{O_i}(m) = \mathrm{SBox}(m \oplus O_i) \tag{3}$$

where $\oplus$ is the exclusive-or and SBox is a nonlinear function.

The attacker focuses on a single (or combined [14]) function f in order to recover the key. In order to be close to the power consumption, the value of f is mapped with a leakage model to another value $Q \in \mathcal{Q}$ where $\mathcal{Q} = \{Q_1, Q_2, ..., Q_K\}$. Examples of leakage models are the identity, the Hamming weight (HW), and the Hamming distance (HD) [31].

For each value of this function let us observe N times the power consumption of a device (identically to the one that the attacker wants to target) over a time interval of length n and denote by *trace* the series of observations. Let ${}^{j}T_i = \left\{ {}^{j}_{t}T_i \in \mathbb{R} \mid t \in [1;n] \right\}$ be the j-th trace associated to the target value Q_i and $\mathcal{T}$ be a set of traces.

Profiling Attack approaches model the stochastic dependency between the value of Q_i and a trace ${}^{j}T_i$. More precisely, they estimate the probability distribution $P(Q_i|{}^{j}T_i;\theta_i)$ (where θ_i is the parameter of the distribution) on the basis of a set of traces (training set) associated to each target value (also known as class).

Two criteria are typically used to assess the quality of a profiling attack: (1) the number of required traces (the lower the better) (2) the success rate (the higher the better) of the model. We chose the second but both methods allow to know which attack is faster to recover the key. Note that these two criteria are related since a high success rate allows the adoption of a low number of traces during the attacking phase. In the following, the accuracy of an attack represents its success rate. More precisely, the accuracy relates to the probability that the correct target value is returned by the attack.

2.2 Template Attack

In order to classify a trace, the Template Attack strategy estimates a template $P(Q_i|{}^{j}T_i;\theta_i)$ for each target value Q_i. By making the assumption of normality, each template's estimation demands the estimation of the means μ_i and the covariance matrix Σ_i from traces associated to the i-th target value. This set of traces is measured on a controlled device similar to the target chip.

Once a template is estimated for each target value, the attacker classifies a new trace T (measured on the target device) by computing the value $\hat{Q} \in \mathcal{Q}$ which maximizes the *a posteriori* probability

$$\begin{aligned}\hat{Q} &= \arg\max_{Q} P(Q|T) && (4)\\ &= \arg\max_{Q} \frac{P(T|Q) \times P(Q)}{P(T)} && (5)\\ &= \arg\max_{Q} \hat{P}(T|Q;\hat{\mu}_i,\hat{\Sigma}_i) \times \hat{P}(Q) && (6)\end{aligned}$$

where the *apriori* probabilities $\hat{P}(Q)$ are estimated by the user accordingly.

If a set $\mathcal{T}$ of traces for a constant secret key are available, the attacker uses the equation (or the log-likehood rule):

$$\hat{P}(Q|\mathcal{T}) = \frac{(\prod_{T\in\mathcal{T}} \hat{P}(T|Q)) \times \hat{P}(Q)}{\sum_{q\in\mathcal{Q}}(\prod_{T\in\mathcal{T}} \hat{P}(T|q)) \times \hat{P}(q))} \qquad (7)$$

2.3 Profiling Attack Based on Machine Learning

In recent years we have assisted to a growing use of machine learning for profiling attack. These techniques do not require the adoption of any parametric or normal assumption and are more suitable to deal with very large dimensional noisy datasets. A conventional machine learning approach to classification relies on two main steps: the dimensionality reduction and the model building.

Dimensionality Reduction. Dimensionality reduction (also known as feature selection or points of interest) aims to extract a subset of p informative variables out of the original n variables [2, 35, 36]. There are plenty of advantages in dimensionality reduction: speed up of the learning process, enhancement of model interpretability, reduction of the amount of storage and improvement of the quality of models by mitigating the curse of dimensionality [4].

The curse of dimensionality is a well known problem in machine learning which states that by increasing dimensionality, the sparsity of data increases at an exponential rate, too. This is a problem when considering classifiers which have to group traces associated to the same target value.

There are several feature selection methods in the literature but we restrict ourselves to three methods. The MAX method selects a set of highest values in a trace.

Another feature selection is the minimum Redundancy Maximum Relevance (mRMR). It was first proposed in the bioinformatics literature [34] in order to deal efficiently with configurations where the number of points in each trace is much larger than the number of traces in the learning set. Its purpose is to rank variables by prioritizing the ones which have a low mutual dependence (i.e., low redundancy) while still providing a large amount of information about the output (i.e., large relevance). The method starts by selecting the variable $r = \left\{ {}_t^jT_i \mid i \in [1;K]\,; j \in [1;N] \right\}$ having the highest mutual information about the target variable $Q = \{Q_i \mid i \in [1;K]\}$. Given a set of selected variables R, the criterion updates this set by adding the variable $t = \left\{ {}_t^jT_i \mid i \in [1;K]\,; j \in [1;N]\,; t \notin R \right\}$ that maximizes the mutual information with the target variable and that minimizes the mutual information with the already selected variables.

Another feature selection method is the Sum of Squared Pairwise t-differences (SOST) [17] based on the T-Test. The T-Test assesses whether the weighted means of traces associated to two different classes are significantly different from each other at time t. More precisely it is expressed by:

$$\frac{{}_t\mu_i - {}_t\mu_j}{\frac{\sqrt{{}_t\sigma_i{}^2}}{N_i} + \frac{\sqrt{{}_t\sigma_j{}^2}}{N_j}} \tag{8}$$

where ${}_t\mu_i$, ${}_t\sigma_i$ and N_i are respectively the means, the standard deviation and the number of traces at time t that are associated to the class Q_i.

The SOST method selects the most relevant components t that have the highest values according to:

$$\sum_{j>i=0}^{K} \left(\frac{{}_t\mu_i - {}_t\mu_j}{\sqrt{\frac{{}_t\sigma_i{}^2}{N_i}} + \sqrt{\frac{{}_t\sigma_j{}^2}{N_j}}} \right)^2 \tag{9}$$

Model Building. Machine learning literature proposes plenty of nonparametric algorithms to estimate $P(Q_i|{}^jT_i;\theta_i)$ on the basis of data. Two well-known examples are Random Forest (RF) [10] and a Support Vector Machine (SVM) [13]. These two techniques allow to remove the Gaussian assumption and to infer in a data driven manner the model which best fits the stochastic dependency between target value and power consumption.

SVM. In a binary classification setting (e.g. $Q_1 = -1$ and $Q_2 = 1$), if the two classes are separable, the SVM algorithm is able to compute from a set of traces the separating hyperplane with the maximal margin, where the margin is the sum of the distances from the hyperplane to the closest traces of each of the two classes.

The SVM classification computes the parameters b (the bias) and w (the weight vector) of the separating hyperplane $[w^\top T + b]$ by solving the following convex optimisation problem:

$$\min \frac{1}{2}(w^\top w) \tag{10}$$

subject to

$$Q_i(w^{\top j}T_i + b) \geq 1 \; \forall i \in [1;2], j \in [1;N] \tag{11}$$

A trace T is assigned to class Q_1 if $w^\top T + b < 0$ and to Q_2 otherwise.

In a setting where the two classes cannot be linearly separated the formulation is changed by introducing a set of slack variables $\xi_j^i \geq 0$ with $i \in [1;2], j \in [1;N]$ then leading to the problem

$$\min_w \frac{1}{2}(w^\top w) + C \sum_{i=1}^{2} \sum_{j=1}^{N} \xi_j^i \tag{12}$$

subject to

$$Q_i(w^{\top j}T_i + b) \geq 1 - \xi_j^i \; \forall i \in [1;2], j \in [1;N] \qquad C \geq 0 \qquad \xi_j^i \geq 0 \tag{13}$$

A larger C means that a higher penalty to classification errors is assigned.

SVM is also modifiable to nonlinear classification tasks by performing a nonlinear transformation κ of traces. This function is named kernel function and can have several forms (e.g. linear, polynomial, radial basis function, sigmoid).

Its purpose is to find a linear separation in a higher dimension if there are no linear separation in the initial dimension. We refer to [19] for a discussion about the role of the kernel function in Equation 13. We used the kernel Radial Basis Function in our experiments.

Several extensions for constructing a multi-class SVM are possible such as one-against-one and one-against-all [22]. We used the one-against-one strategy in our experiments since all methods perform similarly [25].

RF. The Random Forest algorithm was introduced by Breiman in 2001 to address the problem of instability in large decision trees, where by instability we denote the sensitivity of a decision tree structure to small changes in the training set. In other words, large decision trees prone to high variance resulting in high prediction errors.

Let DT_i be a decision tree. In order to reduce the variance, this method relies on the principle of model averaging by building a set of B $(B > 1)$ approximately unbiased decision trees $(\{\mathrm{DT}_1, \mathrm{DT}_2, ..., \mathrm{DT}_B\})$ and returning the most consensual prediction. This means that the target value $\hat{Q}$ of an unlabeled observation T is calculated through a majority vote of the set of trees. More formally,

$$\hat{Q} = f_{\text{majority}}\left(\mathrm{DT}_1\left(T\right), \mathrm{DT}_2\left(T\right), ..., \mathrm{DT}_B\left(T\right)\right) \tag{14}$$

where f_{majority} is the majority vote function and DT_i is the i-th classification tree which returns its prediction for T.

RF is based on two aspects. First each tree is constructed with a different set of traces through the boostrapping method. This method builds a bootstrap sample for each decision tree by resampling (with replacement) the original data set. Observations in the original data set that do not occur in a bootstrap sample are called out-of-bag observations and are used as a validation set. Secondly, each tree is built by adopting a random partitioning criterion. This idea allows to obtain decorrelated trees, thus improving the accuracy of the resulting RF model. The number of trees (B) in the random forest has to be large enough to create diversity among the predictions. In our experiment we use 500 trees which as a rule of thumb is considered as a sufficient number for obtaining accurate prediction.

In conventional decision trees each node is split using the best split among all variables. In the case of a random forest, each node is split using the best among a subset of variables randomly chosen at that node. Also, unlike conventional decision trees, the trees of the random forest are fully grown and are not pruned. In other words, each node contains traces associated to a value of the key. This implies null training error but large variance and consequently a large test error for each single tree. The averaging of the single trees represents a solution to the variance issue without increasing the bias, and allows the design of an overall accurate predictor. Hence the improvements in prediction obtained by random forests are solely a result of variance reduction.

3 A Time Series Approach for Profiling Attacks

Power analysis deals with time series representing the power consumption of a cryptographic device over time. A time series represents a sequence of data points. The most distinctive aspect of a time series is the existence of a stochastic dependency between past and future values. This section introduces our original approach to take into account such a dependency during a profiling attack.

State-of-the-art approaches assume that this dependency is negligible [3, 19–21, 26, 27]. They start with a feature selection step (before a classification step) by projecting traces in new dimensions where (1) the new dimensions correlate highly with the target value and, optionally, (2) the new dimensions correlate weakly between them. We intend to show that in fact such temporal dependence is relevant and can be used in order to improve the success rate of the attack.

Let $y = \{y_1, y_2, ..., y_n\}$ be a time series made of n observations. In the time series literature, the autoregressive formalism is the conventional way to represent the stochastic dependency in a time series [7, 30]. According to this formalism there exists a dependency between the future value y_{t+1} and a set of p past values $\{y_t, y_{t-1}, ..., y_{t-p+1}\}$. More formally it is assumed that each time series has been generated by an autoregressive process of the form

$$y_{t+1} = f_\theta \left(y_t, y_{t-1}, ..., y_{t-p+1}\right) + \varepsilon_{t+1} \tag{15}$$

where ε_t is the additive noise at time t and f is the autoregressive function parametrized by θ. For instance f can be a linear function defined as

$$f_\theta \left(y_t, y_{t-1}, ..., y_{t-p+1}\right) = c + \sum_{i=0}^{p-1} a_i y_{t-i} \tag{16}$$

where $\theta = [c, a_0, a_1, ..., a_{p-1}]$ is the parameter of the model.

The fitting error of $\hat{f}$ represents the difference between the observed values y and the fitted values $\hat{y}$ provided by the estimated model $\hat{f}_{\hat{\theta}}$, i.e.

$$r(\hat{f}_{\hat{\theta}}, y) = \frac{1}{n-p} \sum_{t=p}^{n-1} \left(\hat{f}_{\hat{\theta}} \left(y_t, y_{t-1}, ..., y_{t-p+1}\right) - y_{t+1}\right)^2 \tag{17}$$

In our case the time series y is a trace ${}^jT_i$. A trace contains a succession of consumptions peaks related to the rising and falling edge of the clock (see Figure 5). The data processed inside the cryptographic device are supposed to influence the amplitude of these peaks. As a result, it is expected that the parameter θ describing these peaks is related to the secret (or target) information.

In order to take into account the temporal dependence we fit a time series model $\hat{f}_{\hat{\theta}_i}$ for each class i. The embedding step allows to estimate the parameter θ_i by transforming each time series (related to the i-th class) in a set of $n-p$ time series. More precisely this step transforms each time series $\{y_1, y_2, ..., y_n\}$ into

$$\begin{bmatrix} y_{p+1} \\ y_{p+2} \\ \cdots \\ y_n \end{bmatrix} \begin{bmatrix} y_p & \cdots & y_2 & y_1 \\ y_{p+1} & \cdots & y_3 & y_2 \\ \cdots & \cdots & \cdots & \cdots \\ y_{n-1} & \cdots & y_{n-p+1} & y_{n-p} \end{bmatrix} \tag{18}$$

where the matrix $(n-p)\times 1$ represents the expected output value of the regression model f_θ given the other matrix $(n-p)\times p$. Then the learning step selects the value of θ_i that minimizes the fitting error on the set of transformed time series (by the embedding step) related to the i-th class.

Once a time series model is fitted for each class, we use them to perform a classification of a new trace T. Our technique returns the class by mapping the fitting errors of each regression model to the real class with a classifier h. Examples of classifier h are the random forest, the support vector machine and the arg min function. The arg min function is formalized by

$$\hat{Q} = \arg\min_i r(\hat{f}_{\hat{\theta}_i}, T) \tag{19}$$

The resulting method presents two main advantages with respect to the state-of-the-art: (1) it allows to take into account the natural temporal ordering of data, and (2) it reduces the variance of the feature selection step. This is expected to reduce the complexity of the resulting method with respect to static classifiers since a time series model considers a few number of points ($p-1$ points instead of n) at a time. As a consequence the new approach has a lower variance and leads to a more robust method against noise.

During the attacking phase of a target device, the main expensive step is the computation of the fitting error of each regression model on a trace T. Its complexity is $O(K \times (n-p))$ where K is the number of regression models. In practice the proposed approache can be easily made parallel: each time series model can be executed on different processors allowing to speed up the attacking phase.

4 Experiments

In order to assess the quality of the time series approach, we benchmarked it against template attacks and profiling attacks based on static classification models and three feature selection strategies. We used several datasets with different signal-to-noise ratios. The parameters of each classification and time series models are estimated with a learning set made of 80% of the original dataset. For each model we search the best number of inputs per clock cycle of the crypto device (between 2 and 5 since additional points in the same clock cycle do not provide additional information [35]) by using a validation set. Finally a test set (independent of the learning and the validation set) is used to compare the accuracy of each approach.

4.1 Target Implementation

In order to easily reproduce the results, the experiments were carried out on power leakages that are freely available on the DPAContest V1 website [38] where the cryptographic device (a SecmatV1 SoC in ASIC) implements the unprotected block-cipher DES. Note that the proposed profiling attack is generalizable to other crypto algorithms.

The target value represents the Hamming distance between the left input block and left output block of the last round. Since we have no control on the crypto device (the keys/plaintexts was randomly chosen) we focused on the target values of 7 to 25 in order to have at least several traces per class. However since all the attacks are assessed on the same datasets, this should not have any impact on the results. We choose this target value because (1) it is highly correlated with the traces, (2) it allows to recover 48 bits of the secret key[1] when a plaintext-ciphertext pair is known, and (3) it is good enough in order to compare several approaches.

In the worst case we need $\binom{32}{16} \times 8$ (less than 2^{38}) tests to find all the secret key when the target value is known (in the best case we need only 8 tests to find the secret key). In order to reduce the number of tests an attacker can target the Hamming weight (or the Hamming distance) of a byte (which involves a worse success rate due to a decreasing of the signal-to-noise ratio). Another solution is to use this Hamming distance as an input to other attacks such as an Algebraic Side-Channel Attack [32] or a classical DPA combined with a template attack approach [33]. But the purpose of this section is essentially to compare several approaches in several contexts with real datasets and, therefore, to validate the theoretical presentation performed in previous sections.

4.2 Measurement Setup

In order to generate the traces of the DPAContest V1 an oscilloscope collected 81,089 traces (see a trace in Figure 5), each composed of 20,000 points. A more detailed description of the attacked implementation and the measurement setup can be found in [38]. A fairly standard way of applying power analysis is to focus on one round of the crypto algorithm (where the trace and the target value are dependent) [31]. As a result, we reduced the size of each trace by zooming on the time interval when the target value is manipulated. For this we computed the Pearson correlation between each time of 500 traces and their relatively target values (see Figure 6). We selected a time interval of 100 where the first significant correlation is obtained (a trace in this time interval is plotted in Figure 7).

As stated in the previous section, we focused on the target values (i.e. de HD) between 7 and 25. We reduced the size of the dataset to 8095 by computing the average of 10 traces (associated to the same target value) in order to reduce the

[1] The last 8 bits of the key are searched with a brute-force strategy that requires plaintext-ciphertext pairs.

noise[2]. Table 1 shows the number of traces per target value. It is worth to note that the number of traces per class is strongly imbalanced.

We added a Gaussian noise in order to analyze the signal-to-noise impact on the prediction of each approach[3]. The Gaussian noise follows a Gaussian distribution with a mean of 0 and a standard deviation varying by 0.001 between 0 and 0.005. It allows to confront several models against 6 different contexts. Another approach would be to reduce the number of traces involved in the average. We decided to use the first approach in order to have the same number of traces in each dataset while controlling the noise level with a good precision. The limit of 0.005 for the noise level was mainly based on the result of Figure 8 which shows the impact of the added noise to two traces associated to two different target values. In other words, the noise level was selected according to the difficulty to distinguish traces associated to different target values. A higher noise level does not allow to distinguish classes which include more than 70% of traces of the dataset. This is corroborated in our experiments where several profiling attacks behave as a random model with a noise level of 0.005.

Table 1. Number of traces per target value

Target value	7	8	9	10	11	12	13	14	15	16
Number of traces	6	21	51	122	245	435	666	880	1064	1128
Target value	17	18	19	20	21	22	23	24	25	
Number of traces	1067	873	675	423	239	127	51	16	6	

4.3 Model Selection

We tried several autoregressive models, classification models and feature selection algorithms. For the sake of conciseness we report only the most successful models. We tested the RF, the SVM and the TA as static classification models with the feature selection algorithms MAX, SOST and mRMR. For the time series approach, we considered the Lazy Learning autoregressive model [1, 6, 9] (LL) with the classifiers $\arg\min$ (see Equation 19), RF and SVM.

The LL model proved to be a very effective technique in a number of academic and industrial case studies [8] as it can be applied for a nonlinear regression case. Each lazy model keeps in memory the learning set in order to predict the output value by interpolating the samples in the neighborhood of the input value. The rationale is that it is the neighbors of the input value that are the more relevant for the interpolation problem and, more precisely, to build a regression model.

In order to search the neighbors of the input value we used the euclidean metric. The final performance is extremely sensitive to the choice of the number

[2] We used 6482 traces in the profiling phase and 1613 traces in the test set during the attacking phase.

[3] The Shapiro-Wilk test (with a significance level of 5%) corroborates that the noise on the collected traces follows a univariate Gaussian distribution.

of neighbors: a too small value allows to fit an eventual nonlinearity but at the cost of a high variance; a too high value leads to a large modeling bias. The local regression model tunes automatically the best number of neighbors on the basis of a fast leave-one-out procedure.

4.4 Experimental Results

We first check the quality of the time series fit. Figure 1 shows the fitting returned by three LL models[4] associated to the 7-th, the 16-th and the 25-th HD and with a p equal to 2 (see Equation 15). As it can be seen, each time series model predicts values close to the actual data.

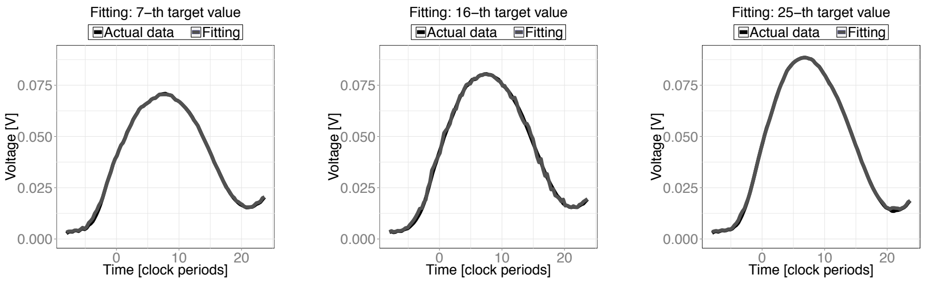

Fig. 1. Fitting of three traces (associated with classes 7, 16 and 25) by three lazy models (with a p equal to 2)

The second experiment compares the template attack against the time series approach. Clearly, from Figure 2, it can be observed that the success rates are similar when the traces contain a high signal-to-noise ratio. Moreover, as expected, the higher is the level of noise, the lower is the performances of both approaches due to the fact that there are less information leakage available. However the time series approach outperforms the state-of-the-art approach when the noise increases. Another important advantage of the time series approach over TA is that the higher the noise the higher the difference between their success rates. It confirms the robustness of the new approach against noise and therefore the model parameters are expected to be more reliably estimated.

The third experiment assesses the time series approach vs. a static classification strategy based on random forest. The results are shown in Figure 3. Random forest allows a higher success rate than template attack in high noise level while their results are similar in the high signal-to-noise ratio context. It confirms the results of previous research [3, 19–21, 26, 27, 29] that template attack is not optimal on several contexts. Nevertheless the success rates of the random forest is lower than the time series approach when the level of noise increases.

[4] We used the implementation available on CRAN [5].

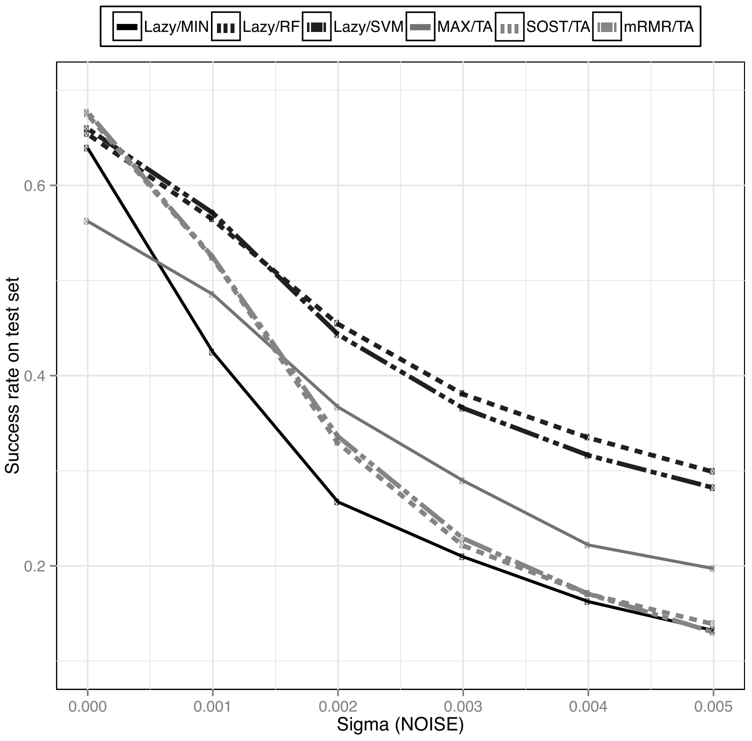

Fig. 2. Success rate per noise level on test set using time series approaches vs template attacks. A/B symbolizes the use of the preprocessing method A with the classifier B.

The last experiment compares the time series approach to a classifier based on support vector machine. The result is plotted in Figure 4. It highlights a similar performance between both approaches when we pick out the best feature selection method for each noise level. However our approach allows a higher success rate (in a noisy context) without the drawback of selection of the best feature selection for each noise level. Note that the selection step of the classification model (both in the regression and in the classification approach) influences the success rate.

4.5 Discussion and Open Questions

The experimental results of the previous sections suggest some considerations. The first one concerns accuracy. We show that for several datasets our approach improves the accuracy of the power analysis attack with respect to conventional template attack as well as to static classification model approach in low signal-to-noise ratio settings.

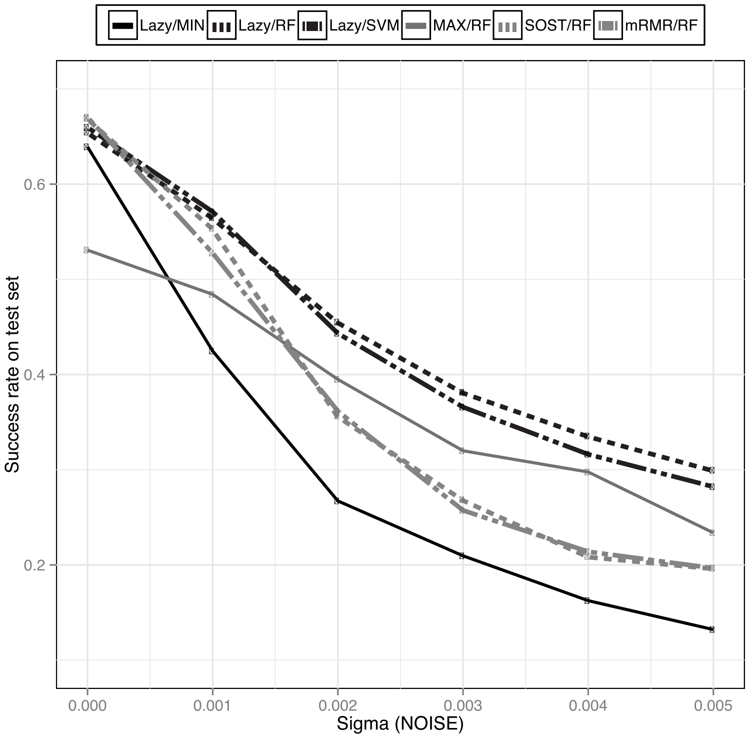

Fig. 3. Success rate per noise level on test set using time series approaches vs random forest. A/B symbolizes the use of the preprocessing method A with the classifier B.

The time series models and the feature selection methods can be seen as a pre-processing step where traces are projected in new dimensions. Their influences in the success rate can be described in terms of the Bias-Variance trade-off [18]. An increase in the complexity of the model leads to an increase of its variance which in turns induces a high sensitivity to noise. On the other hand, in a low noise context where the variance of models does not influence its success rate a more complex model is advantageous due to the fact that its low bias improves its success rate. As a result, a low (resp. high) complex model is favorable in the case of a low (resp. high) signal-to noise context. This reasoning is supported by our experiments: the pre-processing model with the lowest complex outperforms the others in the noisy case. More precisely, the MAX function as well as the time series models lead to higher success rates when we use a RF or a SVM for the classification step. In contrast the SOST and the mRMR seem to outperform the MAX function when the noise is low. As a result the choice of a feature selection should be related to the level of noise on the collected traces. Surprisingly the lazy models combined with the RF or the SVM have a high success rate compared to the presented methods in low and high signal-to-noise case. This is motivated by its low bias rate and its low variance rate.

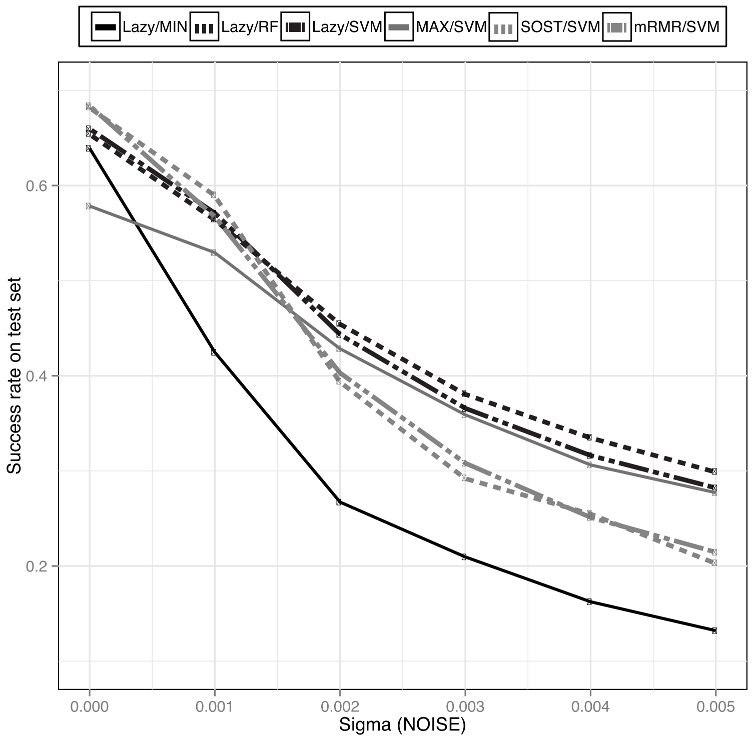

Fig. 4. Success rate per noise level on test set using time series approaches vs support vector machine. A/B symbolizes the use of the preprocessing method A with the classifier B.

An interesting open problem concerns the selection of the classification model in the time series approach. Our results suggest that a random forest or a support vector machine allow to improve the accuracy compared to a MIN function. We could guess that the reason is related to the estimation's accuracy of the error of fitting of each time series model. This estimation is linked to the number of traces used in the learning set of each time series model. A higher number of trace leads to a better estimation of each parameter [39]. As the number of traces in each class is not uniformly distributed (i.e. the number of traces in each class is imbalanced), some time series models estimate better their error of fitting compared to other. As a result, the error of fitting of each time series model should be weighted with the accuracy of their model. We speculate that the RF and the SVM estimate these weight values which allow them to outperform the MIN function. Another issue is that there are some classes that are more difficult to fit than other. Indeed Figure 9 shows that the distribution of the fitting's errors is different for each class. As a result an important question for further research is to determine whether we can improve the success rate by varying the p value for each time series model or at least to rebalance the learning set.

5 Conclusion

Profiling attacks are useful tools in the evolution of leaking cryptographic devices in a worst case scenario. In this paper, we first proposed a new and efficient profiling attack in a multi-class problem. More precisely we introduced a transformation of traces to new dimensions by taking into account the temporal dependence of traces. This new approach offers several starting points for further work with other time series models in the profiling attacks.

We showed that the choice of a feature selection should be related to the level of noise in the collected traces. It led to discuss the advantage of our new proposed technique from a theoretical point of view based on the bias-variance trade-off [18]. We put forward that such profiling attack is less sensitive to noise thanks to its lower variance compared to the presented attacks. Therefore, our method can be carried out in all scenarios where the previously profiling attacks are relevant.

The theoretical point of view is confirmed with several experiments where the new approach allows to improve (significantly) the success rate in several contexts (with several levels of noise). Eventually we discussed the results that lead to interesting open questions such that the impact of differences between the distributions, for each class, of fitting errors in the time series approach. Another interesting question concerns the effect of the number of traces in the learning set for each approach. A more robust model against noise needs less traces in the learning set. As a result, future works will verify whether our proposal outperforms the previous models in a high dimensionality context where the number of traces is less than the number of components in each trace.

In summary this paper confirms that template attack can be improved with machine learning models by designing automatically models from data. More precisely a more powerful adversary is obtained by taking into account the temporal dependence of traces. Hence, practically secure crypto implementations would clearly require to be analyzed with the time series approach. In order to make the time series approach easier to reproduce, an open-source program has been made publicly available [28].

Acknowledgements. Gianluca Bontempi acknowledges the support of the Communauté Francaise de Belgique (ARC project).

References

1. Aha, D.W.: Editorial. Artificial Intelligence Review 11, 7–10 (1997)
2. Archambeau, C., Peeters, E., Standaert, F.-X., Quisquater, J.-J.: Template attacks in principal subspaces. In: Goubin, L., Matsui, M. (eds.) CHES 2006. LNCS, vol. 4249, pp. 1–14. Springer, Heidelberg (2006)
3. Bartkewitz, T., Lemke-Rust, K.: Efficient template attacks based on probabilistic multi-class support vector machines. In: Mangard, S. (ed.) CARDIS 2012. LNCS, vol. 7771, pp. 263–276. Springer, Heidelberg (2013)

4. Bellman, R.: Dynamic Programming, 1st edn. Princeton University Press, Princeton (1957)
5. Birattari, M., Bontempi, G.: Lazy: Lazy Learning for Local Regression, R package version 1.2-14 (2003)
6. Birattari, M., Bontempi, G., Bersini, H.: Lazy learning meets the recursive least squares algorithm. In: Proceedings of the 1998 Conference on Advances in Neural Information Processing Systems II, pp. 375–381. MIT Press, Cambridge (1999)
7. Bisgaard, S., Kulahci, M.: Time Series Analysis and Forecasting by Example. Wiley Series in Probability and Statistics. John Wiley Sons (2011)
8. Bontempi, G., Birattari, M., Bersini, H.: Lazy learners at work: The lazy learning toolbox. In: EUFIT 1999: The 7th European Congress on Intelligent Techniques and Soft Computing, Abstract Booklet with CD Rom, Aachen, Germany. ELITE Foundation (1999)
9. Bontempi, G., Birattari, M., Bersini, H.: Lazy Learning: A local method for supervised learning. In: Jain, L.C., Kacprzyk, J. (eds.) New Learning Paradigms in Soft Computing, pp. 97–137. Springer, Heidelberg (2001)
10. Breiman, L.: Random forests. Machine Learning 45, 5–32 (2001)
11. Chari, S., Rao, J., Rohatgi, P.: Template attacks. In: Kaliski Jr., B.S., Koç, Ç.K., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 13–28. Springer, Heidelberg (2003)
12. Coron, J.-S., Naccache, D., Kocher, P.: Statistics and secret leakage. ACM Trans. Embed. Comput. Syst. 3, 492–508 (2004)
13. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning, 273–297 (1995)
14. Elaabid, M.A., Meynard, O., Guilley, S., Danger, J.-L.: Combined side-channel attacks. In: Chung, Y., Yung, M. (eds.) WISA 2010. LNCS, vol. 6513, pp. 175–190. Springer, Heidelberg (2011)
15. Gandolfi, K., Mourtel, C., Olivier, F.: Electromagnetic Analysis: Concrete Results. In: Koç, Ç.K., Naccache, D., Paar, C. (eds.) CHES 2001. LNCS, vol. 2162, pp. 251–261. Springer, Heidelberg (2001)
16. Gierlichs, B., Batina, L., Tuyls, P., Preneel, B.: Mutual Information Analysis - A Generic Side-Channel Distinguisher. In: Oswald, E., Rohatgi, P. (eds.) CHES 2008. LNCS, vol. 5154, pp. 426–442. Springer, Heidelberg (2008)
17. Gierlichs, B., Lemke-Rust, K., Paar, C.: Templates vs. stochastic methods. In: Goubin, L., Matsui, M. (eds.) CHES 2006. LNCS, vol. 4249, pp. 15–29. Springer, Heidelberg (2006)
18. Hastie, T., Tibshirani, R., Friedman, J.: The elements of statistical learning: data mining, inference and prediction, 2nd edn. Springer (2009)
19. Heuser, A., Zohner, M.: Intelligent machine homicide - Breaking cryptographic devices using support vector machines. In: Schindler, W., Huss, S.A. (eds.) COSADE 2012. LNCS, vol. 7275, pp. 249–264. Springer, Heidelberg (2012)
20. Hospodar, G., Gierlichs, B., Mulder, E.D., Verbauwhede, I., Vandewalle, J.: Machine learning in side-channel analysis: a first study. J. Cryptographic Engineering 1(4), 293–302 (2011)
21. Hospodar, G., Mulder, E.D., Gierlichs, B., Vandewalle, J., Verbauwhede, I.: Least Squares Support Vector Machines for Side-Channel Analysis, pp. 99–104. Center for Advanced Security Research Darmstadt (2011)
22. Hsu, C.-W., Lin, C.-J.: A comparison of methods for multiclass support vector machines. Trans. Neur. Netw. 13(2), 415–425 (2002)

23. Kocher, P.C.: Timing attacks on implementations of diffie-hellman, RSA, DSS, and other systems. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 104–113. Springer, Heidelberg (1996)
24. Kocher, P.C., Jaffe, J., Jun, B.: Differential power analysis. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 388–397. Springer, Heidelberg (1999)
25. KreBel, U.H.-G.: Pairwise classification and support vector machines. In: Advances in Kernel Methods, pp. 255–268. MIT Press, Cambridge (1999)
26. Lerman, L., Bontempi, G., Markowitch, O.: Side Channel Attack: an Approach Based on Machine Learning, pp. 29–41. Center for Advanced Security Research Darmstadt (2011)
27. Lerman, L., Bontempi, G., Markowitch, O.: Power analysis attack: an approach based on machine learning. International Journal of Applied Cryptography (to appear, 2013)
28. Lerman, L., Bontempi, G., Markowitch, O.: sideChannelAttack: Side Channel Attack, R package version 1.0-7 (2013)
29. Lerman, L., Fernandes Medeiros, S., Veshchikov, N., Meuter, C., Bontempi, G., Markowitch, O.: Semi-supervised template attack. In: Prouff, E. (ed.) COSADE 2013. LNCS, vol. 7864, pp. 184–199. Springer, Heidelberg (2013)
30. Makridakis, S., Wheelwright, S., Hyndman, R.J.: Forecasting: Methods and Applications. Wiley series in management. Wiley (1998)
31. Mangard, S., Oswald, E., Popp, T.: Power analysis attacks - revealing the secrets of smart cards. Springer (2007)
32. Oren, Y., Renauld, M., Standaert, F.-X., Wool, A.: Algebraic side-channel attacks beyond the hamming weight leakage model. In: Prouff, E., Schaumont, P. (eds.) CHES 2012. LNCS, vol. 7428, pp. 140–154. Springer, Heidelberg (2012)
33. Oswald, E., Mangard, S.: Template Attacks on Masking-Resistance Is Futile. In: Abe, M. (ed.) CT-RSA 2007. LNCS, vol. 4377, pp. 243–256. Springer, Heidelberg (2006)
34. Peng, H., Long, F., Ding, C.: Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. IEEE Transactions on Pattern Analysis and Machine Intelligence 27(8), 1226–1238 (2005)
35. Rechberger, C., Oswald, E.: Practical template attacks. In: Lim, C.H., Yung, M. (eds.) WISA 2004. LNCS, vol. 3325, pp. 440–456. Springer, Heidelberg (2005)
36. Reparaz, O., Gierlichs, B., Verbauwhede, I.: Selecting time samples for multivariate DPA attacks. In: Prouff, E., Schaumont, P. (eds.) CHES 2012. LNCS, vol. 7428, pp. 155–174. Springer, Heidelberg (2012)
37. Rivain, M., Dottax, E., Prouff, E.: Block ciphers implementations provably secure against second order side channel analysis. In: Nyberg, K. (ed.) FSE 2008. LNCS, vol. 5086, pp. 127–143. Springer, Heidelberg (2008)
38. DPAContest V1 (February 2013), `http://www.dpacontest.org/home/`
39. Wallace, B.C., Dahabreh, I.J.: Class probability estimates are unreliable for imbalanced data (and how to fix them). In: Zaki, M.J., Siebes, A., Yu, J.X., Goethals, B., Webb, G.I., Wu, X. (eds.) ICDM, pp. 695–704. IEEE Computer Society (2012)
40. Whitnall, C., Oswald, E., Mather, L.: An exploration of the kolmogorov-smirnov test as a competitor to mutual information analysis. In: Prouff, E. (ed.) CARDIS 2011. LNCS, vol. 7079, pp. 234–251. Springer, Heidelberg (2011)

Appendix

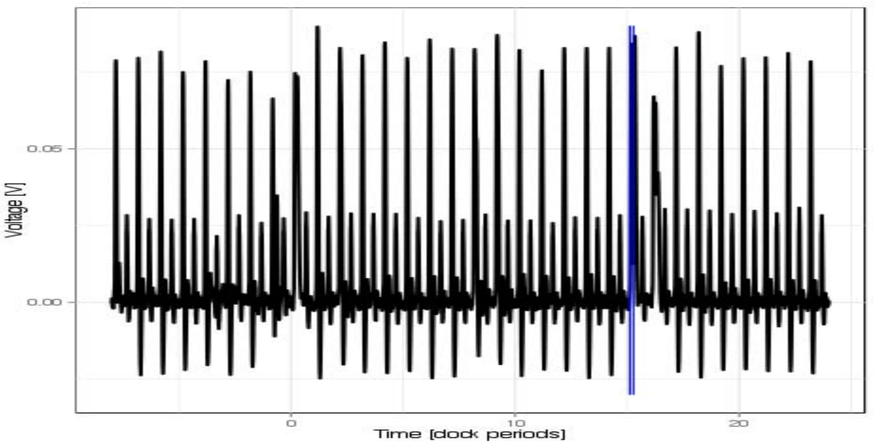

Fig. 5. A trace from the DPAContest V1 where the blue lines represent the time interval where the target value is manipulated

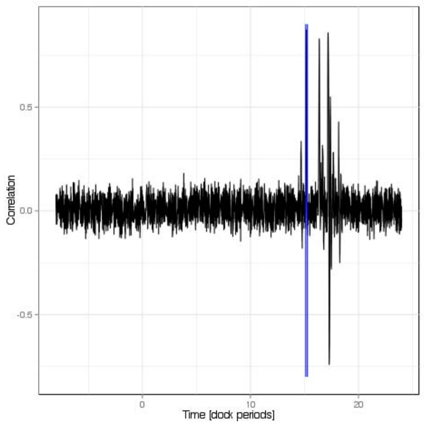

Fig. 6. Correlation between traces and the target value for each time

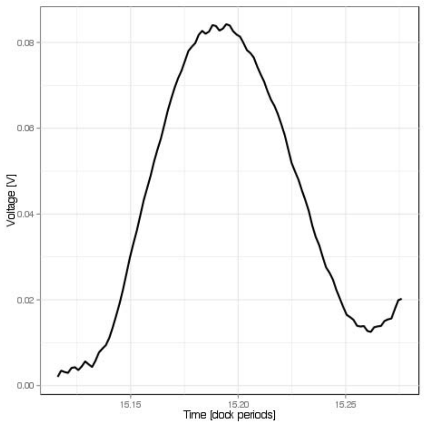

Fig. 7. A zoom on a trace when the target value is manipulated

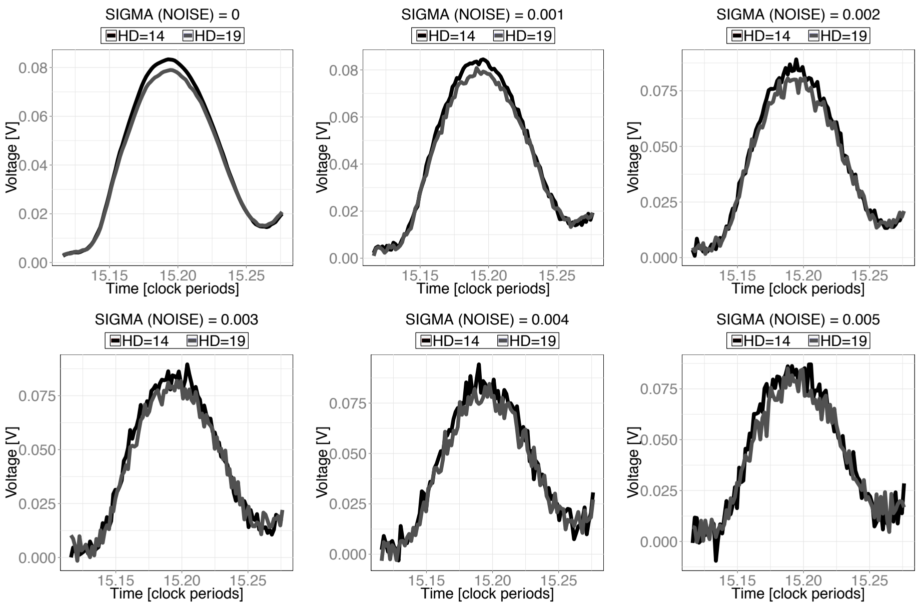

Fig. 8. Each figure shows two traces associated to two different target values (i.e. 14-th and 19-th HD) with a different noise level

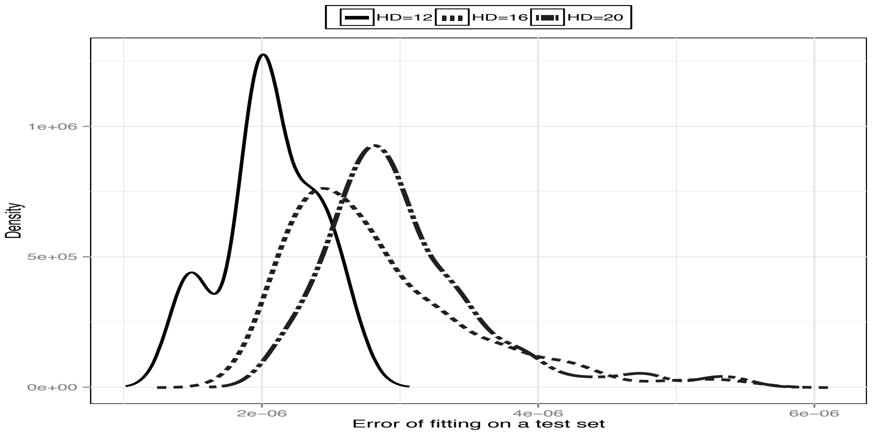

Fig. 9. Three errors densities of fitting on a testing set by different lazy models (where p equal to 2) for classes 9, 16 and 20