Fit The Joint Moments

How to Attack any Masking Schemes

Valence Cristiani[1], Maxime Lecomte[1], Thomas Hiscock[1] and Philippe Maurine[2]

[1] CEA, Grenoble, France
[2] LIRMM, Montpellier, France

Abstract. Side-Channel Analysis (SCA) allows extracting secret keys manipulated by cryptographic primitives through leakages of their physical implementations. Supervised attacks, known to be optimal, can theoretically defeat any counter-measure, including masking, by learning the dependency between the leakage and the secret through the profiling phase. However, defeating masking is less trivial when it comes to unsupervised attacks. While classical strategies such as CPA or LRA have been extended to masked implementations, we show that these extensions only hold for Boolean and arithmetic schemes. Therefore, we propose a new unsupervised strategy, the Joint Moments Regression (JMR), able to defeat any masking schemes (multiplicative, affine, polynomial, inner product...), which are gaining popularity in real implementations. The main idea behind JMR is to directly regress the leakage model of the shares by fitting a system based on higher-order joint moments conditions. We show that this idea can be seen as part of a more general framework known as the Generalized Method of Moments (GMM). This offers strong mathematical foundations on which we can rely on to derive optimizations of JMR. Simulation experiments show that our proposal outperforms state-of-the-art attacks, even in the case of Boolean and arithmetic masking. Eventually, we apply this strategy to provide, to the best of our knowledge, the first unsupervised attack (successful in practice) on the protected AES implementation proposed by the ANSSI for SCA research, which embed an affine masking and shuffling counter-measures.

Keywords: Side-Channel · Masking · Joint Moments

1 Introduction

1.1 Context

Side-Channel Analysis (SCA) is defined as the process of gaining information on a device holding a secret through its physical leakage such as power consumption [KJJ99] or Electromagnetic (EM) emanations [QS01]. The underlying assumption is that the secret and the side-channel data are statistically dependent. This allows an adversary to extract sensitive information such as cryptographic keys by carefully exploiting these dependencies. Strategies are mainly divided into two categories: supervised and unsupervised SCA and their utilization depends on the considered threat model. In the first one, the adversary is supposed to be able to conduct a profiling step of the target, most likely on a clone device, in which she learns the leakage model of the intermediate variables and then adopts a maximum likelihood approach to recover the secret key. This includes strategies such as Gaussian template attack [CRR02] or deep learning profiled attacks [MDP19]. If the model is perfectly learned during the characterization phase, these attacks are known to be optimal from an information theory point of view.

If the profiling step is not possible, the adversary has to use an *a priori* on the leakage model to mount an unsupervised SCA. As shown in [WOS14] there does not exist a generic strategy that would work without requiring such an *a priori*. Different approaches have been developed allowing to exploit an amount of information corresponding to the quality of this *a priori*. For example, [CLM21] showed that Mutual Information Attacks (MIA) can exploit a large part of the information contained in the traces but require an explicit representation of the leakage model (recent deep learning based unsupervised SCA [Tim19, ZLG21] also fall into this category). The main alternatives to MIA are the stochastic attacks, such as Correlation Power Analysis (CPA) [BCO04] or Linear regression Analysis (LRA) [DPRS12], in which the adversary's *a priori* is reduced to a parameterized statistical model which parameters are regressed on the fly. A measure of fitness is then used as a distinguisher to discriminate key candidates.

To prevent instantaneous leakage of the sensitive variables, a classical strategy is to protect the implementation using masking techniques. It consists in splitting the internal state of the processing into multiple random shares following secret sharing ideas [BLA79]. SCA against masked implementations is still possible through the so called higher-order attacks which combine multiple leakage samples corresponding to each share. However, these attacks are harder to conduct since the impact of the noise is amplified exponentially with the masking order [PR13]. Among unsupervised attacks, the multivariate CPA described in [PRB09] has often proved to be an efficient strategy in practice. However, it relies on a Hamming weight leakage assumption (of the shares) that may not be correct especially when it comes to Electro-Magnetic (EM) measurements. Indeed, each bit of the intermediate variable can have very different leakage behavior and even sign inversions of their coefficients as shown in [CLH19]. To deal with such situations [DDP13] proposed a generalization of the LRA whose main strength is to offer flexibility on the *a priori* without constraining each bit to have the same impact on the leakage.

This method exploits information hidden in the covariance *i.e.* the second order joint moment of the distribution since the first order moments (the means) are leakage free thanks to masking. However, we argue that this method is not generic enough because it is based on the assumption that the covariance per class could be expressed as a low algebraical function, assumption that only holds for Boolean and low order arithmetic masking as shown in this paper. Indeed, the proposed attack fails even in theory (with zero noise) when dealing with other masking schemes such as the multiplicative or affine ones. These masking schemes along with the polynomial and inner product masking are getting more and more studied recently and begin to be used in modern implementations. This trend may continue in the future since these schemes seem to offer better resistance against side-channel attacks [BFGV12]. Mutual information-based attacks have also been extended to masked implementations but have not either proven to be valid strategies for any kind of masking and their applicability is mainly related to the open questions, raised in [CLM21], about the choice of the partitioning function. This leads us to the following observation:

To the best of our knowledge, no generic unsupervised strategy able to defeat any kind of masking outside of the Hamming weight leakage assumption emerges from the state-of-the-art.

We propose such a strategy in this paper: the Joint Moment Regression (JMR). The latter is built on the idea that the discriminative information if it exists, is necessarily hidden in higher-order joint moment since lower order leakages are prevented by masking. Intuitively, joint moments encapsulate information about the corresponding distribution. The idea is to make a leakage assumption on each share (for example a linear leakage) and try to directly regress the leakage model of each share, using joint moments conditions, instead of trying to regress the joint moment itself as it is done in [DDP13]. This comes at the cost of the loss of linearity since the joint moment conditions involve multiplication between the leakage parameters of the different shares which give rise to a non-linear system of equations. However, we show that numerical optimization algorithms can be used to find an estimation of the optimal solution that best fits the conditions. A measure of fitness is used as a distinguisher between key candidates. The joint moment conditions depend on the underlying masking scheme which allows to embed knowledge of the later into the system and therefore, makes the attack generic.

1.2 Contributions

- The first contribution of the paper is to present the state of the art on the stochastic higher-order attacks and especially focuses on the method proposed in [DDP13] to understand its strengths and limitations. This analysis can be found in Section 2.
- Then, we introduce a new attack strategy: the Joint Moment Regression (JMR) in Section 3. It is built to circumvent the issue found in the state of the art and proposes a method which is agnostic the underlying masking scheme.
- We then draw a parallel between the core of JMR and a more general framwork: the Generalized Method of Moment (GMM) which is a well-studied paradigm in statistics and economics. This allows to improve our attack in the case of biased masking schemes such as the multiplicative and affine ones. This analysis can be found in Section 4.
- Eventually, section 5 presents a real case application of JMR and provides at the same time the first unsupervised attack on the secured AES implementation of the ANSSI, protected by an affine masking scheme. Attacks that do and do not exploit the lower-order leakage are both presented.

2 Related Work and Limitations

2.1 Notations.

Random variables are represented as upper-case letters such as X. They take their values in the corresponding set $\mathcal{X}$ depicted with a calligraphic letter. Lower case letters such as x stand for elements of $\mathcal{X}$. Expectation of X is denoted $\mathbb{E}[X]$ and covariance between X_1 and X_2 is noted $\text{cov}(X_1, X_2)$. Eventually, $|\mathcal{X}|$ stands for the cardinal of $\mathcal{X}$.

2.2 General Attack Framework

In this paper, the attack framework is described considering that the adversary targets the manipulation of a sensitive variable $Z \in \mathcal{Z} = \mathbb{F}_2^n$, for a given $n \in \mathbb{N}$. This variable is supposed to functionally depends on a public variable $X \in \mathcal{X} = \mathbb{F}_2^m$, for a given $m \in \mathbb{N}$, and a secret key $k^* \in \mathcal{K} = \mathbb{F}_2^m$ through the relation: $Z = f(X, k^*)$ where $f : \mathcal{X} \times \mathcal{K} \to \mathcal{Z}$ is a known function depending on the underlying cryptographic algorithm. The adversary is supposed to own a set $\{(\ell_i, x_i), 1 \leq i \leq N\}$ of N side channels traces labelled with the corresponding public value of X. Traces correspond to realizations of a leakage variable $L \in \mathcal{L}$ coming from a stochastic process $\mathcal{S}$, $Z \xrightarrow{\mathcal{S}} L$ (often separable into a deterministic and a noise part). The leakage variable L is supposed to contain information about Z. The general idea of an unsupervised side-channel attack is to make a series of hypotheses k_i on the key, and to use the dependency between Z and L to build a distinguisher $\mathcal{D} : \mathcal{K} \to \mathbb{R}$

to rank the different key candidates. One of these distinguishers, the LRA, is described in the next section.

2.3 Linear Regression Analysis

The following recalls the steps required to perform an LRA such as suggested in [LPR13]. Traces are assumed to feature ones sample. In a real-life scenario, the same procedure would be repeated for each sample and the final distinguisher keeps the best value along all samples according to a chosen policy (often being the minimum/maximum value of the distinguisher).

1. **Partitioning.** Partition the traces into $|\mathcal{X}|$ classes: $\mathcal{L}_x = \{\ell_i, x_i = x\}$.
2. **Averaging.** Compute the average trace for each classes $\bar{L} = (\bar{\ell}_x)_{x\in\mathcal{X}}$ with
$$\bar{\ell}_x = \frac{1}{|\mathcal{L}_x|}\sum_{\ell\in\mathcal{L}_x} \ell$$
3. **Basis choice.** Choose a basis of functions $(b_i)_{1\le i\le r}$ such that $b_i : \mathcal{Z} \to \mathbb{R}$.
4. **Making hypotheses.** For $k \in \mathcal{K}$ compute the hypotheses matrix:
$$\mathcal{H}_k = \Big(b_i \circ f(x,k)\Big)_{\substack{x\in\mathcal{X},\\ 1\le i\le r}}$$
5. **Linear regression.** For $k \in \mathcal{K}$ find the parameter vector $\theta_k = (\theta_{k,1}, \dots, \theta_{k,r})^T$ minimizing the euclidean norm of the error vector:
$$\theta_k = \underset{\theta}{\operatorname{argmin}}\, ||\mathcal{H}_k \cdot \theta - \bar{L}||_2$$
6. **Ranking.** Rank the keys according to their distinguisher value (from low to high)[1]:
$$\mathcal{D}(k) = ||\mathcal{H}_k \cdot \theta_k - \bar{L}||_2$$

Since step 5 corresponds to a linear regression it has a closed-form solution:
$$\theta_k = ({\mathcal{H}_k}^T \cdot \mathcal{H}_k)^{-1} \cdot {\mathcal{H}_k}^T \cdot \bar{L}$$
However, to highlight similarities with JMR later in the paper, we decided to keep the generic formulation of the optimization problem.

The choice of the basis is important since it should be large enough for the leakage to be decomposable linearly with the $b_i \circ f$ functions when $k = k^*$ but small enough so that it is not the case for wrong hypotheses. The adversary uses his *a priori* on the leakage model, often related to physical assumptions, to choose the basis.

A common example is to assumed that each bit of the sensitive variable contributes to the leakage independently from the others. If this assumptions holds there exists $\alpha = (\alpha_0, \dots, \alpha_n)$ such that $\ell_i = \sum \alpha_j bit_j(z_i) + \alpha_0 + \epsilon$ with bit_j denoting the projection on the j^{th} bit and ϵ being sampled from a noise distribution. In such a case, the basis would be $\{1, bit_1, \dots, bit_n\}$ and θ_{k^*} should be close to α.

Another example is to assume that the leakage is depending on the Hamming weight of the sensitive variable so that $\ell_i = \alpha_1 HW(z_i) + \alpha_0 + \epsilon$. The basis is then reduced to $\{1, HW\}$ and the attack corresponds to the classical CPA.

[1]Sometimes the coefficient of determination R^2 is used instead but the ranking is strictly equivalent except that one ranks from high to low values of the distinguisher.

2.4 Masking

To prevent instantaneous leakages and mitigate the first-order attacks presented above, one of the most widely used countermeasures is masking [CJRR99a]. The idea is to split each sensitive intermediate value Z, into d shares: $(Z_i)_{1\leq i\leq d}$. The $d-1$ shares $Z_2, ..., Z_d$ are randomly chosen and the last one, Z_1 is processed such that:

$$Z_1 = Z * Z_2 * \cdots * Z_d \tag{1}$$

for a group operation $*$ of $\mathcal{Z}$. This has the effect of complexifying the stochastic process $\mathcal{S}$ generating L from Z, rendering it no longer separable into a deterministic and a noise part. Assuming the masks are uniformly distributed, the knowledge of $d-1$ shares does not tell anything about Z (this is why such a masking is said to be of order $d-1$). Therefore, any sound SCA strategy has to combine leakage samples from the d shares to perform an attack (which corresponds to at least d samples if the leakages are disjoint). Such attacks are called d^{th} order attacks. One of them, the second-order LRA is presented in the next section.

The uniform assumption is sometimes not stricly realized in practice depending on the masking scheme being used. Four of the most common masking scheme that will be studied in this paper are listed in table 1. The $\oplus$ and $\otimes$ respectively stand for the addition and the multiplication operation in $\mathbb{F}_2^n$. Since the multiplication by 0 is not invertible the "multiplicative shares" have to be chosen in $\mathbb{F}_2^n \setminus \{0\}$. The multiplicative and affine schemes are then slightly biased, and therefore, do not guarantee in theory, SCA resilience to all the $(d-1)^{th}$ and lower order attacks. Such attacks will be discussed in section 4.

Table 1: Masking schemes studied in this paper

	Group operation	Masked Variable (Z_1)	Uniform	Reference
Boolean	$\oplus$	$Z \oplus Z_2 \oplus \cdots \oplus Z_d$	Yes	[CJRR99b]
Arithmetic	$+ \; mod \; 2^n$	$Z + Z_2 + \cdots + Z_d \; [2^n]$	Yes	[GPQ11]
Multiplicative	$\otimes$	$Z \otimes Z_2 \otimes \cdots \otimes Z_d$	No	[GPQ11]
Affine	$\oplus, \otimes$	$Z \otimes Z_2 \oplus Z_3$	No	[FMPR11]

2.5 Second-Order LRA

This section describes the generalization of the LRA introduced in [DDP13] which aims at defeating a first-order masked implementation ($d = 2$). Traces are considered to be composed of 2 samples: $L = (L_1, L_2)$ where L_1 and L_2 represent respectively the leakage of the first and second share. In a real-life scenario, the attack would be repeated with all the combinations of two samples from the raw traces. To perform a second-order LRA the adversary is supposed to own a set of N traces $\{(\ell_1^i, \ell_2^i), 1 \leq i \leq N\}$. The idea is to replace the estimated mean per class by the estimated covariance per class in the classical LRA which naturally combines information from the two samples. Indeed the covariance $Y = \text{cov}(L_1, L_2)$ involves the product of the centered variable $L_1 - \mu_1$ and $L_2 - \mu_2$, with $(\mu_1, \mu_2) = \mathbb{E}[L]$, which has been shown to be a good combining function for second-order SCA [PRB09]. The steps to perform a second-order LRA are depicted hereafter.

1. **Partitioning.** Partition the traces into $|\mathcal{X}|$ classes: $\mathcal{L}_x = \{(\ell_1^i, \ell_2^i), x_i = x\}$.
2. **Estimating covariances.** Compute the estimated covariance for each classes $\bar{Y} = (\bar{y}_x)_{x\in\mathcal{X}}$ with

$$\bar{y}_x = \frac{1}{|\mathcal{L}_x|} \sum_{\ell \in \mathcal{L}_x} (\ell_1 - \bar{\mu}_1)(\ell_2 - \bar{\mu}_2)$$

where $(\bar{\mu}_1, \bar{\mu}_2)$ stands for the estimated mean of L.

3. **Basis choice.** Choose a basis of functions $(b_i)_{1\leq i\leq r}$ such that $b_i : \mathcal{Z} \to \mathbb{R}$.

4. **Making hypotheses.** For $k \in \mathcal{K}$ compute the hypotheses matrix:

$$\mathcal{H}_k = \Big(b_i \circ f(x,k)\Big)_{\substack{x\in\mathcal{X},\\ 1\leq i\leq r}}$$

5. **Linear regression.** For $k \in \mathcal{K}$ find the parameter vector $\theta_k = (\theta_{k,1}, \ldots, \theta_{k,r})^T$ minimizing the euclidean norm of the error vector:

$$\theta_k = \operatorname*{argmin}_{\theta} ||\mathcal{H}_k \cdot \theta - \bar{Y}||_2$$

6. **Ranking.** Rank the keys according to their distinguisher value (from low to high):

$$\mathcal{D}(k) = ||\mathcal{H}_k \cdot \theta_k - \bar{Y}||_2$$

The attack may seem very similar to a first-order LRA except that it is performed on the covariance instead of the mean (the change happens in step 2). However, the choice of the basis is much more delicate. The link between the adversary *a priori* and a basis leading to a successful attack is not trivial anymore. Indeed, the hypotheses matrix is constructed using the unmasked variable $Z^{(k)} = f(X,k)$ while the leakage *a priori* concerns the shares. The choice of the basis proposed in [DDP13] is based on an assumption that is recalled hereafter.

Let first define the set of functions $(\varphi_k)_{k\in\mathcal{K}} : \mathcal{Z} = \mathbb{F}_2^n \to \mathbb{R}$ such that:

$$\varphi_k(z) = \text{cov}(L_1, L_2)|Z^{(k)} = z \tag{2}$$

Since all the Boolean functions in $\mathbb{F}_2^n$ can be represented by a multivariate polynomial in $\mathbb{R}[z_1,\ldots,z_n]/(z_1^2 - z_1,\ldots,z_n^2 - z_n)$ (*i.e.* the degree of every z_i in every monomial is at most 1) [Car07], there exists, for any k, a unique set of coefficients $(\alpha_{k,u})_{u\in\mathbb{F}_2^n}$ such that:

$$\varphi_k(z) = \sum_{u=(u_1,\ldots,u_n)\in\mathbb{F}_2^n} \alpha_{k,u} \cdot z^u \tag{3}$$

where each term z^u denotes the monomial (function) $z \to z_1^{u_1} z_2^{u_2} \ldots z_n^{u_n}$ with $z_i^{u_i} \in \mathbb{F}_2$. Let $deg(\varphi_k)$ stands for the degree of the polynomial representing φ_k. The assumption on which the attack from [DDP13] relies is the following:

Assumption 1. *$\forall k \neq k^*$, $deg(\varphi_{k^*}) < deg(\varphi_k)$.*

The intuition behind this assumption is that since $\varphi_k = \varphi_{k^*} \circ f_k \circ f_{k^*}^{-1}$ (where $f_k = f(\cdot,k)$), φ_k is expected to have a high degree (close to n) if $k \neq k^*$, due to cryptographic properties of f which often embed highly non-linear Sboxes to prevent algebraic attacks. Note that this reasoning only holds if φ_{k^*} itself has a low degree which is implicitly assumed in [DDP13]. This point will be discussed later.

If assumption 1 holds, the basis: $(b_i)_i = \{z^u, u \in \mathbb{F}_2^n, HW(u) \leq deg(\varphi_{k^*})\}$ is a valid basis for the second-order LRA. Indeed, it spans all the functions of degree less or equal $deg(\varphi_{k^*})$. Therefore there exists a decomposition of φ_{k^*} in this basis while it is not the case for other φ_k, by hypothesis, which guarantees the success of the attack (provided that the number of traces allows for a fair approximation of the covariances per class).

2.6 Limitations

The first observation is that even if assumption 1 holds, the attack may fail in practice if $\deg(\varphi_{k^*})$ is not low enough. Indeed, the cardinal of the basis, and therefore the number of parameters to estimate, increases quickly with $deg(\varphi_{k^*})$ offering a big capacity to the statistical model to fit the data whatever the considered value of k. If the noise is not negligible, this often means that the wrong hypotheses can reach similar scores than the correct one which reduces the distinguishability and therefore the effectiveness of the attack. For example, with $n = 8$ and $deg(\varphi_{k^*}) \in \{1, 2, 3\}$ the cardinal of the basis is respectively equal to 9, 37 and 93. In practice authors of [DDP13] run their attack with the following basis: $(b_i)_i = \{z^u, u \in \mathbb{F}_2^n, HW(u) \leq d_{max}\}$ where $d_{max} \in \{1, 2, 3\}$. Choosing $d_{max} = 3$ never led to the best attack even in cases where $deg(\varphi_{k^*})$ was strictly greater than 2 (due to the high model capacity and lack of distiguishability).

Then, one could ask if assumption 1 holds at all. Since $\varphi_{k^*}(z) = \mathrm{cov}(L_1, L_2)|Z^{(k^*)} = z$ it is obviously related to the nature of the leakage L_1 and L_2. These variables can be assumed to be separable into a deterministic and a noise part with respect to the shares:

$$L_i = l_i(Z_i) + \epsilon_i \tag{4}$$

with $l_i : \mathcal{Z} \to \mathbb{R}$ representing the leakage of share i and ϵ_i being an independent random noise variable. By bilinearity of the covariance and independence of ϵ_i:

$$\begin{aligned} \varphi_{k^*}(z) &= \mathrm{cov}\big(l_1(Z_1), l_2(Z_2)\big)|Z^{(k^*)} = z \\ &= \mathrm{cov}\big(l_1(z * Z_2), l_2(Z_2)\big) \end{aligned} \tag{5}$$

since $Z_1 = Z^{(k^*)} * Z_2$. Both l_1 and l_2 can be assumed of low degree (through a physical *a priori* on the leakage). For example, it is realistic to assume that both shares follow a linear leakage. But we argue that this is not enough to guarantee assumption 1 and that it is still depending on the underlying masking scheme, especially on the nature of the $*$ operation.

Then, a natural question arises: why does the attack presented in [DDP13] work? We argue that it is related to the studied masking schemes in their paper. Indeed, the latter one focuses on the Boolean and arithmetic masking schemes which are both exceptions as far as assumption 1 is concerned. This claim is justified by the two following propositions.

Proposition 1. *(Boolean masking) Let $* = \oplus$. Let $l_1 : \mathcal{Z} \to \mathbb{R}$ and $l_2 : \mathcal{Z} \to \mathbb{R}$ be two leakage functions of degre 1. Let $\varphi_{Bool}(z) = \mathrm{cov}\big(l_1(z \oplus Z_2), l_2(Z_2)\big)$. Then,*

$$deg(\varphi_{Bool}) \leq 1 \tag{6}$$

Proof. Proof can be found in appendix A. □

Proposition 2. *(Arithmetic masking) Let $* = + \bmod 2^n$. Let $l_1 : \mathcal{Z} \to \mathbb{R}$ and $l_2 : \mathcal{Z} \to \mathbb{R}$ be two leakage functions of degre 1. Let $\varphi_{Arith}(z) = \mathrm{cov}\big(l_1(z + Z_2\,[2^n]), l_2(Z_2)\big)$. Then,*

$$deg(\varphi_{Arith}) \leq 2 \tag{7}$$

Proof. Proof can be found in appendix A. □

These two propositions explain the success of the attacks presented in [DDP13]. However, we could not find equivalent propositions for other masking schemes, suggesting that Boolean and arithmetic masking are, in fact, exceptions. This will be empirically confirmed in section 3.4 where it is shown that even without noise, the higher-order LRA fails against multiplicative or affine masking with a linear leakage of the shares. Therefore, to the best of our knowledge, there is no strategy in the literature able to defeat a generic masking scheme in an unsupervised context, with a simple linear leakage assumption of the shares. We introduce such a strategy in the next section.

3 Joint Moments Regression

We first introduce the concept of Joint Moment (JM) that generalizes to any masking order the idea of the covariance, found in the previous section.

3.1 Joint Moments

Moments of probability distributions are quantitative measures related to the shape of the distribution. The moment of order d, denoted μ_d, of the variable X is defined as:

$$\mu_X^{(d)} = \mathbb{E}[X^d] \tag{8}$$

For second and higher orders, the centered moment $\check{\mu}_d$ of order d are often used instead and are defined as:

$$\check{\mu}_X^{(d)} = \mathbb{E}[(X - \mu_X^{(1)})^d] \tag{9}$$

Joint moments are the generalization of moments to multivariate variables. Let $X = (X_1, \ldots, X_k) \in \mathbb{R}^k$ be a multivariate random variable. Let $u = (u_1, \ldots, u_k) \in \mathbb{N}^k$ be a vector of positive integers such that $\sum u_i = d$. The JM of order d with respect to vector u, denoted jm_u, is defined as:

$$jm_X^{(u)} = \mathbb{E}[X_1^{u_1} \ldots X_k^{u_k}] \tag{10}$$

Centered JM are also defined as:

$$\check{jm}_X^{(u)} = \mathbb{E}[(X_1 - \mu_{X_1}^{(1)})^{u_1} \ldots (X_k - \mu_{X_k}^{(1)})^{u_k}] \tag{11}$$

One important property of JM (and of simple moments) is that, for distributions defined on a compact set of $\mathbb{R}^n$, the distribution is fully defined by the list (maybe infinite) of all its JM. This is also true for the centered JM provided that the first order JM are also given.

The effect of a d-order masking is that no information related to the sensitive variable can be found in the $d-1$ and lower JM. That is why the second-order LRA performed a regression on the second-order centered JM with $u = (1, 1)$ which happens to be the covariance. Indeed it is the lowest order JM bringing information on the sensitive variable. Information could also be found in higher-order JM but they are harder to estimate. Indeed, more terms are involved in the product and the noise in each one of them is amplified through the multiplication. One typically wants to take the JM with the lowest standard error (the standard deviation of its estimator). This also explains why centered JM are preferred: as shown in [PRB09], they have a lower standard error than their uncentered counterpart.

3.2 Attack description

Let an adversary owns a set of N traces $\{\ell^i, 1 \leq i \leq N\}$ of a d order masked implementation. Traces are considered to be composed of d samples: $\ell^i = \{(\ell_1^i, \ldots, \ell_d^i)$. In a real-life scenario, the attack would be repeated on combinations of d samples from the raw traces depending of the attacker a priori on the points of interest. To defeat this implementation a naive solution would be to extend the attack proposed in [DDP13] using centered JM instead of covariance but as stated in section 2.6: there is no obvious link between the physical *a priori*, which happens to be on the shares, and the basis that has to be chosen and applied to the unmasked sensitive variable.

That is why we propose a new strategy where the adversary chooses d basis, one for each share (in practice they will often be the same basis), and directly regress the leakage of each share using information from the estimated d order centered JM. The steps of what we call the Joint Moment Regression (JMR) are depicted hereafter.

JMR Procedure

1. **Partitioning.** Partition the traces into $|\mathcal{X}|$ classes: $\mathcal{L}_x = \{(\ell_1^i, \dots, \ell_d^i), x_i = x\}$.

2. **Estimating JM.** Compute the estimated centered d order JM for each classes $\overline{JM} = (\overline{jm}_x)_{x \in \mathcal{X}}$ with

$$\overline{jm}_x = \frac{1}{|\mathcal{L}_x|} \sum_{\ell \in \mathcal{L}_x} (\ell_1 - \bar{\mu}_1) \dots (\ell_d - \bar{\mu}_d)$$

   where $(\bar{\mu}_1, \dots, \bar{\mu}_d)$ stands for the estimated mean of L.

3. **Basis choice.** For $j \in [1, d]$, choose a basis of functions $(b_i^{(j)})_{1 \le i \le r}$ such that $b_i^{(j)} : \mathcal{Z} \to \mathbb{R}$. Intuitively, if l_j corresponds to the leakage of share j, the adversary wants to choose basis such that $l_j(z_j) = \sum_{i=1}^r \theta_{j,i} \cdot b_i^{(j)}(z_j) + \epsilon_j$ for some coefficient $\theta_j \in \mathbb{R}^r$, with ϵ_j representing an independent random noise variable.

4. **Making hypotheses.** For $k \in \mathcal{K}$, define the theoretical JM vector $JM_k(\theta)$ with respect to $\theta \in \mathbb{R}^{d \times r}$, that traduces the leakage assumption of step 3 into $|\mathcal{X}| = 2^m$ JM per class expressions:

$$JM_k(\theta) = \begin{pmatrix} a_0 \sum\limits_{(z_1,\dots,z_d) \in \mathcal{A}_0} \left(\sum\limits_{i=1}^r \theta_{1,i} \cdot b_i^{(1)}(z_1) - \mu_{\theta_1} \right) \dots \left(\sum\limits_{i=1}^r \theta_{d,i} \cdot b_i^{(d)}(z_d) - \mu_{\theta_d} \right) \\ \vdots \\ a_{2^m-1} \sum\limits_{(z_1,\dots,z_d) \in \mathcal{A}_{2^m-1}} \left(\sum\limits_{i=1}^r \theta_{1,i} \cdot b_i^{(1)}(z_1) - \mu_{\theta_1} \right) \dots \left(\sum\limits_{i=1}^r \theta_{d,i} \cdot b_i^{(d)}(z_d) - \mu_{\theta_d} \right) \end{pmatrix}$$

   with $\mathcal{A}_x = \{(z_1, \dots, z_d) | Z = f(x, k)\}$ and $a_x = \frac{1}{|\mathcal{A}_x|}$. Here, μ_{θ_j} stands for the theoretical mean of the leakage of share j under the assumption of θ_j:

$$\mu_{\theta_j} = \mathbb{E}_{Z_j} \left[\sum_{i=1}^r \theta_{j,i} \cdot b_i^{(1)}(Z_j) \right]$$

5. **Non-linear regression.** For $k \in \mathcal{K}$, find through numerical optimization techniques (see Subsection 3.3), the parameter vector $\theta^{(k)} \in \mathbb{R}^d \times \mathbb{R}^r$ minimizing the euclidean norm of the error vector:

$$\theta^{(k)} = \underset{\theta}{\operatorname{argmin}} \, ||JM_k(\theta) - \overline{JM}||_2$$

6. **Ranking.** Rank the keys according to their distinguisher value (from low to high):

$$\mathcal{D}(k) = ||JM_k(\theta^{(k)}) - \overline{JM}||_2$$

3.3 Attack Soundness

The general attack structure of JMR is very similar to the LRA and second-order LRA. The main difference with the latter one is that the assumption is done on the leakage of the shares and is therefore directly related to the physical *a priori*. These assumptions are then combined to build a parameterized system of unknown $\theta \in \mathbb{R}^{d \times r}$:

$$JM_k(\theta) - \overline{JM} = 0 \tag{12}$$

where each line represents a condition on the JM knowing that $X = x$. Note that by the independence assumption, the noise terms ϵ_j are canceled from the theoretical equations of the JM per class, listed in the $JM_k(\theta)$ vectors. The goal is then to find the solution $\theta^{(k)}$ that fits the most the system and to use a measure of fitness as distinguisher. Note that the knowledge of the underlying masking scheme is embedded in the system through the $\mathcal{A}_x$ sets which describe the possible value $(z_1, \dots, z_d)$ of the shares given the value of Z. This is what ensures the genericity of JMR regarding the masking scheme.

When the number of traces N tends towards infinity, the estimated JM per class $\overline{JM}$ tends towards the true JM per class. If the leakage assumptions are correct there exists $\theta^{(k^*)} \in \mathbb{R}^{d \times r}$ such that $JM_{k^*}(\theta^{(k^*)})$ is equal to the true JM per class. Therefore:

$$\lim_{N \to \infty} \mathcal{D}(k^*) = 0 \tag{13}$$

while it is unlikely to be the case for $k \neq k^*$ due to cryptographic property of f, which assures the soundness of the attack.

However, this multi-shares assumption comes at the cost of linearity. Indeed, even if all the shares are assumed to leak linearly, the system that JMR regresses is not linear anymore: it is of degree d. Therefore there is no closed-form solution and one has to use numerical optimization tools to find an approximation of the solution. Numerical optimization is a research field in itself and is out of the scope of this paper. There exists multiple ready-to-use implementations in different programming languages which is enough for our concern. Note that since the system is of degree d, the uniqueness of the solution of step 5 is not guaranteed. This is not a problem for the attack: as long as one solution can be found and that equation 13 holds only for the correct key hypothesis, the attack will succeed for a sufficient number of traces.

3.4 Simulation Experiments

This section provides simulations experiments in order to assess the feasibility of JMR in practice against the masking schemes presented in table 1. Its efficiency is compared with state of the art attacks at second and third order.

Implementation. We implemented the core of the JMR attack using the `least_squares` function from the python `scipy.optimize` package [VGO+20]. It solves a non-linear least-squares fitting problem using the Levenberg-Marquardt (LM) algorithm [Lev44, Mar63] which is itself based on the Gauss-Newton algorithm and the method of gradient descent. The attack time or complexity is mostly constant regarding the number of traces because the latter does not affect the number of parameters nor equations in the system. The only part that scales with the number of traces is the estimation of the JM per class which is just a product and a sum and that can be handled with `numpy` [HMG+20] arrays manipulations.

Since the least-squares problems related to the different key hypotheses are independent, the implementation is highly parallelizable. We exploited this using a 48 cores Xeon Platinium 8168 processor which speeded up the attack by a factor of 48 since the implementation of the `least_squares` function is not parallelized in itself. Other implementation optimization could be explored such as using the fast GPU version of the LM algorithm proposed in [PTK+17] but this is not in the scope of this paper. To give an order of magnitude, with our setup, running the full JMR procedure as describe in Subsection 3.2 for a $d-$tuple of time samples, requires around 10 and 15 seconds for respectively a second and third-order attack (assuming one trace for each possible value of the shares: 2^{16} and 2^{24} respectively).

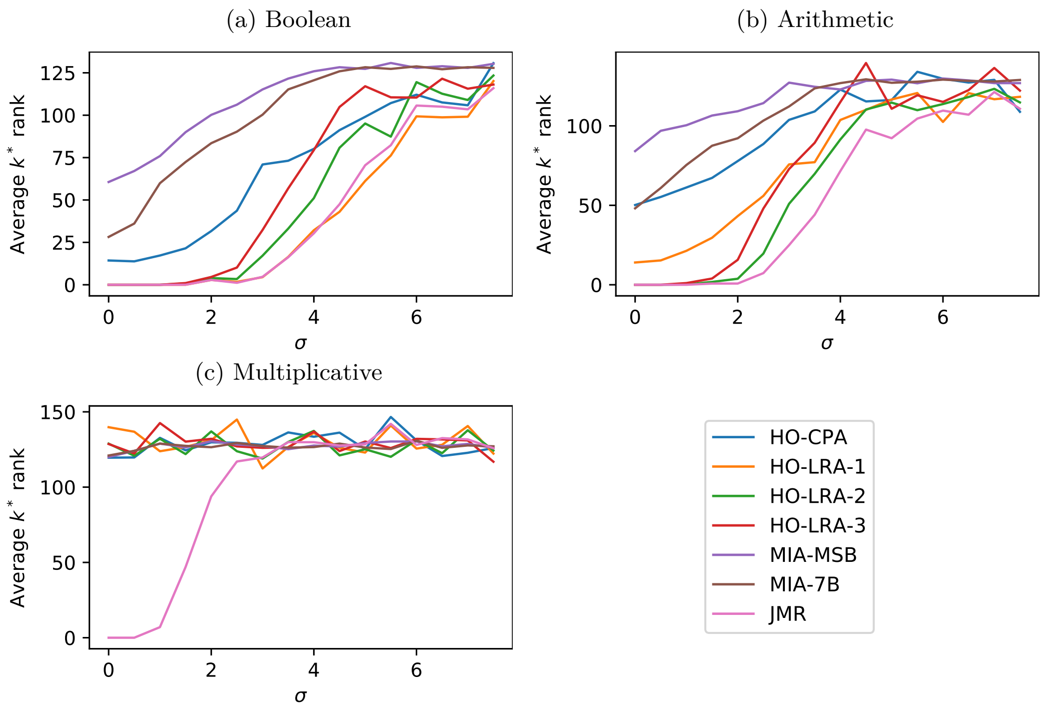

Figure 1: Guessing entropies in terms of standard deviation of the noise for the considered second order attacks

Generating Datasets. To assess the JMR method and to compare it with state of the art attacks, synthetic traces datasets with linear leakage of the shares have been generated for first and second-order masking ($d \in \{2, 3\}$). Boolean, arithmetic, multiplicative and affine (only with $d = 3$) schemes are used to mask the classical sensitive variable of an AES: $Z = Sbox[k^* \oplus P]$ (k^* and P are both supposed to be 8 bits long). To avoid any kind of estimation error (the error coming from sampling), each dataset contains one trace for each of the possible value of the shares $(z_1, \ldots, z_d) \in \mathcal{Z}^d$ (for multiplicative and affine schemes the multiplicative shares can not be 0 so we take them from $[\![1, 255]\!]$ instead. The trace $\ell_{(z_1,\ldots,z_d)}$ corresponding to the d-tuple $(z_1, \ldots, z_d)$ is generated as follows:

$$\ell_{(z_1,\ldots,z_d)} = \left[\alpha_{1,0} + \sum_{i=1}^{8} \alpha_{1,i} \cdot z_1[i] + \epsilon_1(\sigma), \cdots, \alpha_{d,0} + \sum_{i=1}^{8} \alpha_{d,i} \cdot z_d[i] + \epsilon_d(\sigma)\right] \tag{14}$$

where $z_j[i]$ corresponds to the i^{th} bit of z_j, each $\alpha_{j,i}$ is a coefficient uniformly drawn from $[-1, 1]$ and ϵ_j is drawn from a normal distribution $\mathcal{N}(0, \sigma^2)$. To be able to average the result of different attacks, we have generated 100 sets of coefficients $(A_i)_{1 \leq i \leq 100}$ where $A_i \in [-1, 1]^{9 \times d}$. So each trace dataset depends on four parameters: the masking scheme, the order, the set of coefficients representing the leakage model, and a value of σ.

Results. Second-order attacks results are presented in Figure 1. Each point represents the average rank of k^* over the 100 datasets for a given value of σ. We compare JMR with

- A higher-order CPA, denoted HO-CPA, computed with a Hamming weight leakage model and using the JM of order d as combining function which happens to be the same as the centered product described in [PRB09].
- Higher order LRA, denoted HO-LRA-d_{max} where d_{max} is the assumed degree of φ_{k^*} as defined in Equation 5. Therefore the basis used in HO-LRA-d_{max} is $(b_i)_i =$

$\{z^u, u \in \mathbb{F}_2^n, HW(u) \leq d_{max}\}$. The combining function is also the JM of order d which for $d = 3$ is a straightforward extension of the second order attack described in [DDP13].

- Mutual Information Analysis, denoted MIA-f, where the distinguisher used is $MI(f(Z_k), L)$. MIA requires the use of a non-injective function f to create distinguishability for the correct hypothesis. Since the leakage model is unknown we used very generic model: $f = MSB$ and $f = 7B$, where MSB stands for the most significant bit of Z_k and $7B$ stands for the 7 most significant bits of Z_k. The MI has been estimated using the histogram method described in [PR09].

(a) For the Boolean case, JMR and HO-LRA-1 performs approximately the same which is not surprising since, by Proposition 1, assumption 1 holds for HO-LRA1. It also holds for HO-LRA-2/3 but HO-LRA-1 perfectly explains the data with fewer parameters which explains why it performs better. One can notice that even without noise the HO-CPA is not converging towards 0 which confirms that it relies on the Hamming weight leakage assumption. Also, MIA strategies do not perform well which is not surprising since the underlying leakage model is unknown and it is therefore hard to select a good non-injective function.

(b) For the arithmetic scheme, JMR outperforms all the other attacks even, HO-LRA-2 in which assumption 1 holds by Proposition 2. Again this is explained by the fact that JMR only needs (9×2) parameters to predict the data while HO-LRA-2 needs 37 parameters. Even without noise, the data can not be perfectly explained in an HO-CPA or HO-LRA-1 model since their curves do not converge towards 0.

(c) For the multiplicative scheme, as predicted, none of the state of the art attacks perform better than random even without noise which confirms that assumption 1 does not hold at all for such a masking scheme. JMR is the only sound attack in this case.

Results for third-order attacks are presented in Figure 2. In this case HO-LRA-d_{max} represents the generalisation of the second-order LRA replacing the covariance by the third order joint moment. Conclusions are the same as for second-order attacks. As, for the multiplicative case, JMR is the only sound option to attack affine masking under a linear leakage of the shares. However, both multiplicative and affine schemes are a slightly biased which can induce lower-order leakage. This could be exploited with simpler attacks such as a CPA with a zero value based power model as shown in the next section. However, as the attack presented in this section does not take advantage of this, JMR should be expected to work even for non-biased scheme with a high algebraical degree where other attacks would not. More advanced attacks combining leakage at multiple orders are discussed in the next section where we introduce the generalized method of moment paradigm.

4 Generalized Method of Moments Paradigm

Looking from a broader perspective, it appears that the core of the JMR attack can be seen as part of a more general framework known as the Generalized Method of Moments (GMM) [Han82]. This method comes from the field of statistics and economy and its main purpose is to estimate parameters in a statistical model. Embracing this paradigm requires to gain a level of abstraction but it allows to use the powerful mathematical foundations behind it. In particular, it will tell us how to optimally combines information from different orders which is useful when the masking scheme is biased.

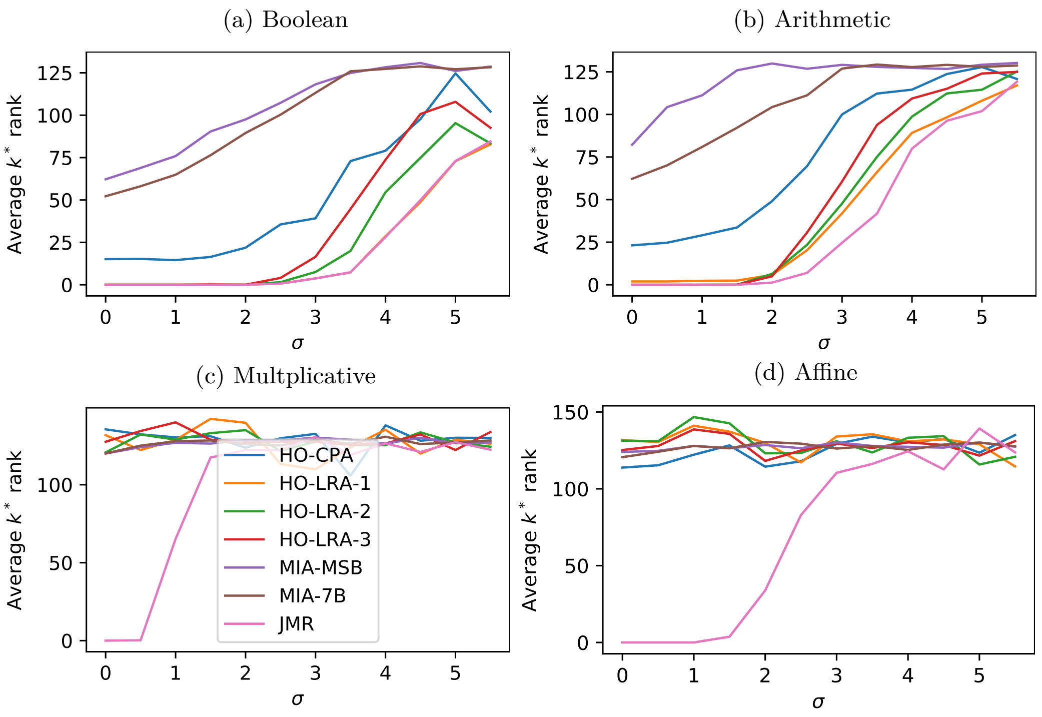

Figure 2: Guessing entropies in terms of standard deviation of the noise for the considered third order attacks

4.1 Background on GMM

Let suppose that the available data consist of N observations $(L_i)_{1\leq i\leq N}$ of a random variable $L \in \mathbb{R}^n$. This data is assumed to come from a stochastic process defined up to an unknown parameter vector $\theta \in \mathbb{R}^p$. The goal is to find the true value θ_0 of this parameter or at least a reasonably close estimate.

In order to apply GMM the data must come from a weakly stationary ergodic stochastic process (independent and identically distributed (iid) variables are a special case of these conditions). Then one needs to have c “moment conditions” defined as a function $g(\ell, \theta) : \mathbb{R}^n \times \mathbb{R}^p \rightarrow \mathbb{R}^c$ such that:

$$\mathbb{E}[g(L, \theta_0)] = 0 \tag{15}$$

The idea is then to replace the theoretical expectations with its empirical analog:

$$m(\theta) = \frac{1}{N}\sum_{i=1}^{N} g(\ell_i, \theta) \tag{16}$$

and to minimize the norm of $m(\theta)$ with respect to θ. The properties of the GMM estimator depend on the chosen norm and therefore the theory considers the entire family of norms defined up to a positive-definite weighting matrix $W \in M_c(\mathbb{R})$:

$$||m(\theta)||_W = \sqrt{m(\theta)^T W m(\theta)} \tag{17}$$

The GMM estimator is then defined as:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}}\, ||m(\theta)||_W \tag{18}$$

The way of solving this optimization problem is not specified in the GMM theory. It is left to the numerical optimization field.

The purpose of W is to weight the different conditions. Choosing $W = \mathrm{Id}_c$ gives rise to the classical euclidean norm and is equivalent to consider that all conditions should weight the same. The intuition behind the fact that one may prefer another norm is that some conditions may be less informative, redundant, or more volatile in their empirical estimation. One typically wants to use the norm minimizing the asymptotic variance of the resulting estimator. This problem has a closed-form solution with the following theorem:

Theorem 1. *(Hansen 1982) Let $\hat{\theta}_N$ be the random variable representing the output of the GMM estimator with N data observations. Let also define Ω as the covariance matrix of the conditions function g evaluated at θ_0: $\Omega = \text{cov-mat}\big(g(L,\theta_0)\big)$. Then,*

$$\underset{W}{\operatorname{argmin}} \lim_{N\to\infty} \operatorname{var}(\hat{\theta}_N) = \Omega^{-1} \tag{19}$$

In the particular case where conditions are independent the matrix Ω^{-1} is diagonal and choosing $W = \Omega^{-1}$ simply means that the moments condition should be weighted inversely proportionally to their underlying variance. This is in line with the intuition that conditions with high variance are less informative.

4.2 Parallel with the JMR attack

This section exhibits the similarities between the GMM and JMR. The core of the JMR attack relies on an estimation of the true parameters $\theta_0 \in \Theta = \mathbb{R}^d \times \mathbb{R}^r$ of a chosen statistical model (encoded in the choice of the basis) in order to explain the leakage of each share. Let $L = (L_1, \ldots, L_d)$ represents the observed leakage variable and L_θ the predicted leakage variable under the assumption of θ so that, under the assumption that the chosen model is correct, $L = L_{\theta_0}$.

Since the moment conditions in JMR depends on the value of another public variable X, let define, for each key hypothesis k, a condition function $g_k : \mathbb{R}^d \times \mathcal{X} \times \Theta \to \mathbb{R}^{|\mathcal{X}|}$ as:

$$g_k(\ell, x, \theta) = e_x \cdot \left(\check{jm}^{(\mathbf{1}_d)}_{L_\theta|Z=f(x,k)} - \prod_{i=1}^{d} (\ell_i - \bar{\mu}_i) \right) \tag{20}$$

where $e_x = (0, \ldots, 1, \ldots, 0) \in \mathbb{R}^{|\mathcal{X}|}$ stands for a vector of 0 with one 1 at position[2] x, $\mathbf{1}_d$ stands for a vector of d ones : $\mathbf{1}_d = (1, \ldots, 1)$ and $\check{jm}^{(u)}_L$ is defined as in Equation 11. This definition of g_k may seem very artificial but it is designed so that Equation 15 holds for the correct hypothesis $k = k^*$ (under the assumption that $L = L_{\theta_0}$):

$$\begin{aligned} \mathbb{E}[g_{k^*}(L, X, \theta_0)] &= \frac{1}{|\mathcal{X}|} \left(\check{jm}^{(\mathbf{1}_d)}_{L_{\theta_0}|Z=f(x,k^*)} - \check{jm}^{(\mathbf{1}_d)}_{L|X=x} \right)_{x\in\mathcal{X}} \\ &= 0 \end{aligned} \tag{21}$$

Therefore applying GMM with g_{k^*} as condition function is sound while it is not for wrong key hypotheses. In fact this property is the one exploited by JMR since step 1 to 5 of JMR are equivalent to apply $|\mathcal{K}|$ GMM estimations, one for each of the g_k condition functions, with $W = \mathrm{Id}_{|\mathcal{X}|}$.

[2]Here $x \in \mathcal{X} = \mathbb{F}_2^m$ is seen as an element of $\mathbb{Z}/2^m\mathbb{Z}$.

4.3 Improving JMR using GMM theory

This section describes two ways of improving JMR using the GMM theory. The first one is generic and the second one focuses on the unbalanced masking schemes.

Using the Optimal Weighting Matrix. Since the GMM theory recommends to use Ω^{-1} as weighting matrix, one could ask if using the identity matrix was optimal. Indeed, the adversary typically wants to minimize the variance of the GMM estimator for the correct key $k = k^*$. Therefore it would be natural to replace the identity matrix with Ω^{-1} where $\Omega = \text{cov-mat}\big(g_{k^*}(L, X, \theta_0)\big)$. The problem is that Ω is hard to estimate with data since θ_0 is unknown. The solution to this problem is usually to apply the so-called two-step estimator where an estimation of θ_0 is first computed with JMR with a sub-optimal weighting matrix (for example the identity) which allows estimating Ω and eventually apply GMM with the latter estimation as weighting matrix. However, in our case, Ω does not depend on θ_0 which makes the process easier. Indeed, the variance of the components of g_{k^*} (and therefore the covariance matrix) only comes from the right term of Equation 20 which does not depend on θ. Therefore the equation of Ω can be re-written as:

$$\Omega = \text{cov-mat}\left[e_X\left(\prod_{i=1}^{d}(L_i - \bar{\mu}_i)\right)\right] \tag{22}$$

In addition, since for a fixed x, only one component of $g_{k^*}(\ell, x, \theta)$ is non-zero, Ω is diagonal. Then, one can estimate the diagonal terms of Ω using the observed data and then apply GMM. We denote by JMR_{++} the JMR attack but with $W = \bar{\Omega}^{-1}$ where $\bar{\Omega}$ is an estimation of the optimal weighting matrix.

To confirm the soundness of this approach, we performed the same experiments as described in Subsection 3.4 to compare JMR and JMR_{++}. Figures 3a and 3b show the results for the second-order Boolean and arithmetic masking and, according to the theory, JMR_{++} performs a little better than JMR. It can be noticed that in the case of Boolean masking JMR_{++} also outperforms HO-LRA-1, which had approximately the same performance as JMR, despite having more parameters to estimate.

The Case of Unbalanced Schemes. Some masking schemes, such as the multiplicative or the affine one, violate the assumption of shares uniformity. Therefore the resilience to $(d-1)^{th}$-order attack is not guaranteed anymore. For example, $Z = 0$ implies $Z_1 = 0$ in a multiplicative scheme inducing a first-order leakage. As well, when $Z = 0$, the affine scheme becomes a Boolean scheme of order 2 inducing second-order leakages. Since lower-order JM are informative in these cases, a first idea to exploit this weakness is to apply JMR but at a lower-order. This means that all the conditions concern first-order moments for a multiplicative scheme and second-order JM for an affine scheme. Such an attack is denoted $\text{JMR}_{\text{Lower}}$. Since this would only exploit the difference between the class $Z = 0$ and $Z \neq 0$ this attack would be very close to a CPA computed with a zero valued model computed considering only two classes: $Z = 0$ and $Z \neq 0$, denoted CPA-0 afterward.

Figures 3c and 3d confirm this intuition by showing that both CPA-0 and $\text{JMR}_{\text{Lower}}$ behave very similarly and have better results than JMR for high noise values but worse results for low noise values. Indeed, since the main advantage of masking is to amplify the impact of the noise exponentially with the order of the mask [PR13] or more accurately, with the order of the attack required to defeat it. For low values of σ the JM conditions used in $\text{JMR}_{\text{Lower}}$ are less informative than the one used by JMR (they only exploit a difference between the class $Z = 0$ and the other classes) but the impact of the noise is amplified by a lower order which explains the better results of $\text{JMR}_{\text{Lower}}$ for high σ.

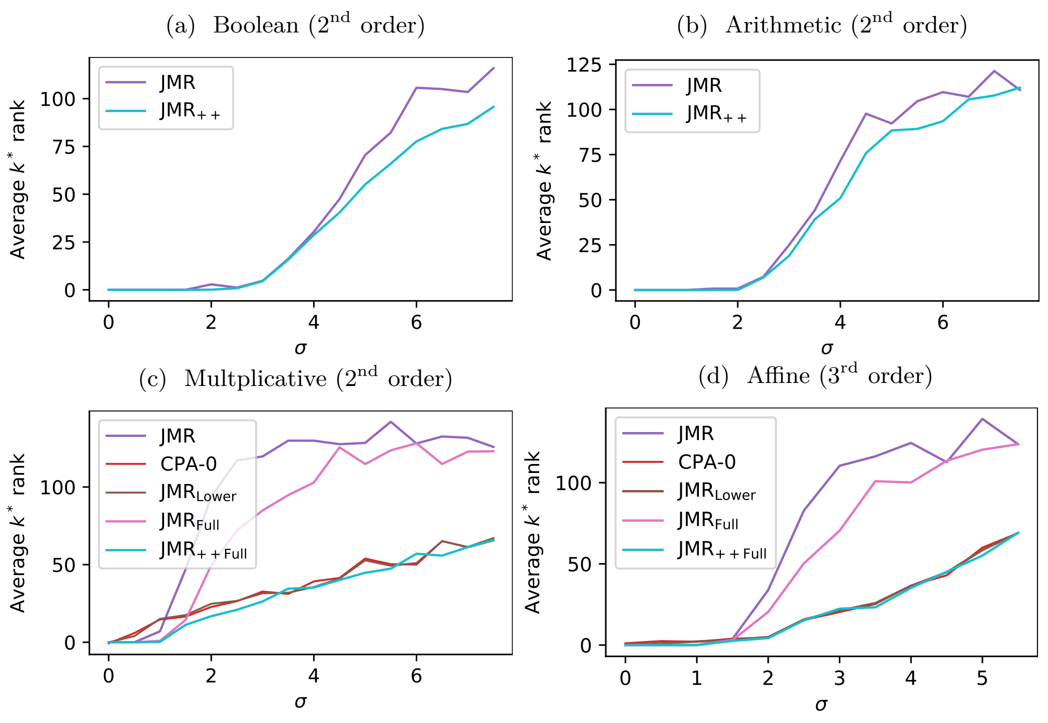

Figure 3: Guessing entropies for the improved JMR attacks, using the GMM theory, and for CPA-0

A natural challenge is to design an attack benefiting from the best of both worlds: JMR and JMR$_{\text{Lower}}$. To this aim, we propose to use the flexibility of the GMM paradigm to develop an attack with conditions from both informative orders at the same time. This corresponds to building a system with 512 conditions instead of 256 when attacking a key byte. In this case, the weighting matrix is very important since each half of the system concerns conditions with very different variances (estimating joint moments is exponentially hard with the order). To highlight this fact we denote by JMR$_{\text{Full}}$ and JMR$_{++\text{Full}}$ the version of JMR with both order conditions respectively with $W = \text{Id}_{512}$ and $W = \bar{\Omega}^{-1}$.

Results are presented in figures 3c and 3d. As excepted, JMR$_{\text{Full}}$ outperforms JMR but is impacted by the variance of the d-order conditions and therefore performs worse than JMR$_{\text{Lower}}$ for high values of σ. However, JMR$_{++\text{Full}}$ benefits from the advantage of exploiting the d-order conditions for low values of σ but still converges towards JMR$_{\text{Lower}}$ for high noise values thanks to the well-chosen weighting of these conditions. Indeed, it is proven in [Han82] that adding more moments conditions can only improve the performance of the GMM estimator (by lowering its variance) when using the optimal weighting matrix Ω^{-1}.

5 Attack of an Open Source secured AES implementation

To assess the performance of JMR on real traces, we decided to attack the protected AES implementation proposed by the *Agence Nationale de la Sécurité des Systèmes d'Information* (ANSSI) [BKPT19]. They published a library implementing an AES-128 on an ARM Cortex-M4 architecture using state-of-the-art counter-measures. Indeed, this implementation uses an affine masking as well as random shuffling of independent operations [VCMKS]. It is accompanied by a publicly available dataset called ASCADv2 providing 800,000 traces acquired on a STM32F303 microcontroller running this protected AES. A detailed leakage analysis of this dataset has been published in [MS21]. Following their terminology we tried to attack the unmasked variable $Z = Sbox[k^* \oplus P]$ using the leakage of the three shares:

$$\begin{aligned} Z_1 &= Z \otimes r_{\text{mul}} \oplus r_{\text{out}} \\ Z_2 &= r_{\text{mul}} \\ Z_3 &= r_{\text{out}} \end{aligned} \tag{23}$$

Unfortunately, the number of traces turned-out to be too low to analyze the unsupervised attacks discussed in this paper. Thus, we reproduced a similar experimental setup, described in the next section, in order to collect significantly more traces.

5.1 Acquisition Setup

Our setup has the following features:

- The acquisitions have been performed on a NUCLEO-F303RE board, which embeds the same STM32F303 micro-controller as used in ASCADv2.
- The device is clocked at 8MHz, while ASCADv2 device is clocked a 4 MHz. This allows faster acquisitions without altering the execution behavior (e.g., introducing FLASH wait cycles). Being in an evaluation setup, we had the labels of the shares and validated that it did not affect the signal to noise ratio of these intermediate variables.
- We measured the magnetic field produced by the circuit with a Langer H-field probe (RF-U 5-2). This differs from ASCADv2 setup, which measures the current of the device through a ChipWhisperer [OC14]. However, we observed better signal to noise ratios on the EM field. The probe covers a large portion of the CPU and no specific tuning of the probe placement was performed.
- The scope was configured at 3.125 GS/s and acquired a window of $8\mu s$, which represents 25,000 time samples.
- The masked AES implementation was taken “as-this” from the SecAESSTM32 repository [BKPT19]. We only made the following changes to the assembly code:
  - a GPIO is raised in the `Load_random` function, which manipulates r_{mul} and r_{out}.
  - a GPIO is raised in the first round of the AES, just after the `Xor_Word` operation.

To further speed up the acquisitions, we do not transfer the plaintext and masking inputs through the serial port. Indeed, this represents 54 bytes $(16 + 19 \times 2)$ per encryption, which quickly becomes a bottleneck for acquisitions. Instead, the device runs a PCG32 Pseudo-Random Number Generator (PRNG) [O’N14] to generate those data on the fly. This PRNG is re-seeded randomly (by sending 8 bytes on the serial port) every 250 acquisitions. This allows to regenerate (from the stored seeds) the plaintexts and random masks offline, to label the dataset.

For each encryption, the scope triggers twice and acquires 50,000 samples. For a dataset of 100M traces, this represents 5 TB of storage. To reduce the size of the dataset, we performed a preliminary leakage analysis and selected only 1334 samples corresponding to significant signal to noise of the different shares. To this aim, with the knowledge of the shares, a simple SNR has been computed for each share keeping only times samples above a threshold with a value equal to 1 (for at least one of the shares). The final dataset contains 100M traces (121GB) and took 14 days to acquire.

In summary, we used the same AES implementation and micro-controller than the ASCADv2 setup. We only made some changes in the instrumentation and measurement chain to reduce the number of traces needed and improve the speed of acquisition.

5.2 Simulating an Unshuffled Version

The implementation uses random permutation of the 16 `Sboxes` applications. However, using the same idea as developed in technical analysis of the ANSSI repository [BKPT19], one can simulate (through the knowledge of the key and the permutation Sh being used) an attack on an unshuffled version even if the acquired traces are shuffled. Instead of targeting the first byte $Z = Sbox[k^*[0] \oplus P[0]]$ one may target:

$$Z = Sbox[k^*[Sh^{-1}(0)] \oplus P[Sh^{-1}(0)]] \tag{24}$$

where $Sh^{-1}(0)$ denotes the index of the byte that is computed first through the permutation Sh. Then such an attack would uses $Z^{(\bar{k})} = Sbox[\bar{k} \oplus k^*[Sh^{-1}(0)] \oplus P[Sh^{-1}(0)]]$ as hypothesis intermediate variable, the attack being successful if the best hypothesis is 0.

5.3 Results

Using the dataset described in Subsection 5.1, we performed the different attacks discussed in this paper both on the shuffled and unshuffled versions. The attacks on the shuffled version only use the leakage of the first `Sbox` computation. Shuffling adds a lot of noise since even for the correct key hypothesis the predicted value of Z is only correct once out of 16 in average. More efficient attacks could use the leakage of the 16 computations but this leakage has not been kept in traces for storage capacity reasons.

Results are presented in Figure 4. Each point represents the mean ranking of the correct key over 100 attacks performed with the corresponding number of traces. For each attack, traces are randomly drawn among the 100M dataset. Results are presented using the 3 times samples (one for each share) giving the best results. Both $\text{JMR}_{++\text{Full}}$ and JMR_{++} converge towards a guessing entropy of 0 which provides by the same token, the first unsupervised attack on the secured ANSSI's AES implementation.

Not surprisingly, $\text{JMR}_{++\text{Full}}$, which exploits the bias in the masking scheme, gives the best results. It requires 30k and 15M traces to converge towards 0 for the unshuffled and shuffled version respectively. This confirms that for high noise value, a lower-order leakage induces attacks with at least one order of magnitude smaller data complexity. Thus, it confirms that even though d shares are used to mask the sensitive value, a biased d-order masking should not be considered of order d as far as security is concerned.

Even when this lower-order leakage is not considered in the attack, JMR_{++} seems to outperform the other state-of-the-art attacks. One may notice that these attacks are making better than random predictions towards the correct key which was not the case in our simulated experiments. This is explained by the fact that in this case, the leakage model is much closer to a Hamming weight model. Even in such a case, these results confirm the soundness of the JMR approach for unsupervised attacks/analysis when dealing with masked implementations.

It should be noted that even if some of the presented attacks require less that 800k traces, they have not been successful on the original ASCADv2 dataset. We have confirmed that our traces have a better SNR on the leakage of each of the shares which explains this difference.

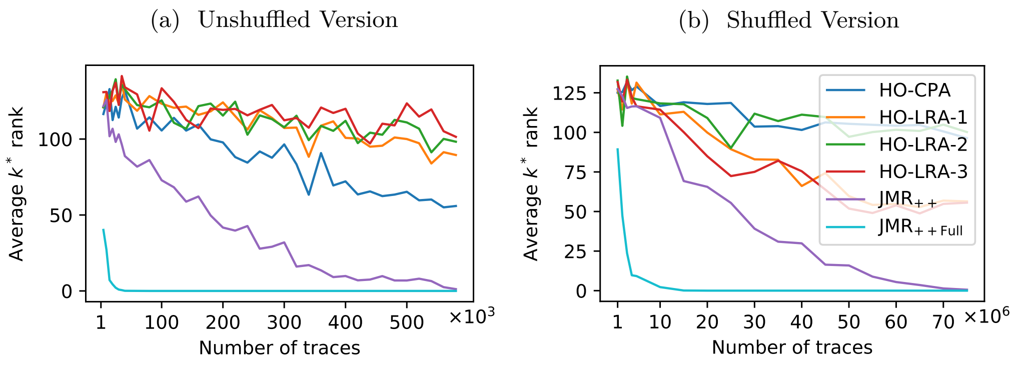

Figure 4: Comparison of different attacks' guessing entropies on the secured ANSSI's AES

6 Conclusion

This paper introduced a new unsupervised strategy, JMR, able to defeat any masking schemes. It is based on a non-linear system regression which allows to derive the leakage model of each shares by carefully exploiting higher-order joint moments conditions. JMR outperforms state-of-the-art attacks which are limited to Boolean and arithmetic masking, especially when the Hamming weight leakage assumption does not hold. We reduced the core of JMR into a more general framework: the generalized method of moments and derived optimizations of JMR from it. Experiments performed on synthetic data confirmed the effectiveness of the proposed attack especially against multiplicative and affine masking schemes. Eventually, this new strategy has been confirmed on real traces, allowing a fully unsupervised attack of the ANSSI's protected AES implementation which embed an affine masking and shuffling counter-measures.

The JMR method is not highly multi-dimensional in the sense that it only exploits d times sample when applied on a d^{th}-order masking. It is well known that sensitive variables can leak several time in a single trace. Strategies able to extend JMR approach to use more informative time samples simultaneously (*i.e.* exploit more of the available information) would be of great interest for further research.

References

[BCO04] Eric Brier, Christophe Clavier, and Francis Olivier. Correlation power analysis with a leakage model. In Marc Joye and Jean-Jacques Quisquater, editors, *Cryptographic Hardware and Embedded Systems - CHES 2004*, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[BFGV12] Josep Balasch, Sebastian Faust, Benedikt Gierlichs, and Ingrid Verbauwhede. Theory and practice of a leakage resilient masking scheme. In *ASIACRYPT*, volume 7658, pages 758–775. Springer, 2012.

[BKPT19] Ryad Benadjila, Louiza Khati, Emmanuel Prouff, and Adrian Thillard. Hardened library for aes-128 encryption/decryption on arm cortex m4 achitecture., 2019.

[BLA79] G. R. BLAKLEY. Safeguarding cryptographic keys. In *1979 International Workshop on Managing Requirements Knowledge (MARK)*, pages 313–318, 1979.

[Car07] Claude Carlet. Boolean functions for cryptography and error correcting codes. 11 2007.

[CJRR99a] Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. Towards sound approaches to counteract power-analysis attacks. In Michael Wiener, editor, *Advances in Cryptology — CRYPTO' 99*, pages 398–412, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg.

[CJRR99b] Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. Towards sound approaches to counteract power-analysis attacks. In Michael Wiener, editor, *Advances in Cryptology — CRYPTO' 99*, pages 398–412, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg.

[CLH19] Valence Cristiani, Maxime Lecomte, and Thomas Hiscock. A Bit-Level Approach to Side Channel Based Disassembling. In *CARDIS 2019*, Prague, Czech Republic, November 2019.

[CLM21] Valence Cristiani, Maxime Lecomte, and Philippe Maurine. Revisiting mutual information analysis: Multidimensionality, neural estimation and optimality proofs. Cryptology ePrint Archive, Report 2021/1518, 2021.

[CRR02] Suresh Chari, Josyula R Rao, and Pankaj Rohatgi. Template attacks. In *International Workshop on Cryptographic Hardware and Embedded Systems*, 2002.

[DDP13] Guillaume Dabosville, Julien Doget, and Emmanuel Prouff. A new second-order side channel attack based on linear regression. *IEEE Transactions on Computers*, 62(8):1629–1640, 2013.

[DPRS12] Julien Doget, Emmanuel Prouff, Matthieu Rivain, and François-Xavier Standaert. Univariate side channel attacks and leakage modeling. *Journal of Cryptographic Engineering*, 1:123–144, 04 2012.

[FMPR11] Guillaume Fumaroli, Ange Martinelli, Emmanuel Prouff, and Matthieu Rivain. Affine masking against higher-order side channel analysis. In Alex Biryukov, Guang Gong, and Douglas R. Stinson, editors, *Selected Areas in Cryptography*, pages 262–280, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[GPQ11] Laurie Genelle, Emmanuel Prouff, and Michaël Quisquater. Thwarting higher-order side channel analysis with additive and multiplicative maskings. In Bart Preneel and Tsuyoshi Takagi, editors, *Cryptographic Hardware and Embedded Systems – CHES 2011*, pages 240–255, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[Han82] Lars Peter Hansen. Large sample properties of generalized method of moments estimators. *Econometrica*, 50(4):1029–1054, 1982.

[HMG$^+$20] Charles R. Harris, K. Jarrod Millman, Ralf Gommers, Pauli Virtanen, David Cournapeau, and Eric Wieser et al. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[KJJ99] Paul Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In *Annual International Cryptology Conference*, 1999.

[Lev44] Kenneth Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly of Applied Mathematics*, 2(2):164–168, 1944.

[LPR13] Victor Lomné, Emmanuel Prouff, and Thomas Roche. Behind the scene of side channel attacks. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology - ASIACRYPT 2013*, pages 506–525, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[Mar63] Donald W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial and Applied Mathematics*, 11(2):431–441, 1963.

[MDP19] Loïc Masure, Cécile Dumas, and Emmanuel Prouff. A comprehensive study of deep learning for side-channel analysis. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2020, 2019.

[MS21] Loïc Masure and Rémi Strullu. Side channel analysis against the anssi's protected aes implementation on arm. Cryptology ePrint Archive, Report 2021/592, 2021.

[OC14] Colin O'Flynn and Zhizhang (David) Chen. Chipwhisperer: An open-source platform for hardware embedded security research. In Emmanuel Prouff, editor, *Constructive Side-Channel Analysis and Secure Design*, pages 243–260, Cham, 2014. Springer International Publishing.

[O'N14] Melissa E. O'Neill. Pcg: A family of simple fast space-efficient statistically good algorithms for random number generation. Technical Report HMC-CS-2014-0905, Harvey Mudd College, Claremont, CA, September 2014.

[PR09] Emmanuel Prouff and Matthieu Rivain. Theoretical and practical aspects of mutual information based side channel analysis. pages 499–518, 01 2009.

[PR13] Emmanuel Prouff and Matthieu Rivain. Masking against side-channel attacks: A formal security proof. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology – EUROCRYPT 2013*, pages 142–159, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[PRB09] E. Prouff, M. Rivain, and R. Bevan. Statistical analysis of second order differential power analysis. *IEEE Transactions on Computers*, 58(6):799–811, 2009.

[PTK+17] Adrian Przybylski, Björn Thiel, Jan Keller, Bernd Stock, and Mark Bates. Gpufit: An open-source toolkit for gpu-accelerated curve fitting, 10 2017.

[QS01] Jean-Jacques Quisquater and David Samyde. Electromagnetic analysis: Measures and counter-measures for smart cards. 2001.

[Tim19] Benjamin Timon. Non-profiled deep learning-based side-channel attacks with sensitivity analysis. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2019(2):107–131, Feb. 2019.

[VCMKS] Nicolas Veyrat-Charvillon, Marcel Medwed, Stéphanie Kerckhof, and François-Xavier Standaert. Shuffling against side-channel attacks: A comprehensive study with cautionary note. In Xiaoyun Wang and Kazue Sako, editors, *Advances in Cryptology – ASIACRYPT 2012*.

[VGO+20] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, and Pearu et al. Peterson. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[WOS14] Carolyn Whitnall, Elisabeth Oswald, and François-Xavier Standaert. The myth of generic dpa. . . and the magic of learning. In Josh Benaloh, editor, *Topics in Cryptology – CT-RSA 2014*, pages 183–205, Cham, 2014. Springer International Publishing.

[ZLG21] Chi Zhang, Xiangjun Lu, and Dawu Gu. Binary classification-based side-channel analysis. In *2021 Asian Hardware Oriented Security and Trust Symposium (AsianHOST)*, pages 1–6, 2021.

A Proofs

Proposition 1. *(Boolean masking) Let $* = \oplus$. Let $l_1 : \mathcal{Z} \to \mathbb{R}$ and $l_2 : \mathcal{Z} \to \mathbb{R}$ be two leakage functions of degree 1. Let $\varphi_{Bool}(z) = \operatorname{cov}\big(l_1(z \oplus Z_2), l_2(Z_2)\big)$. Then,*

$$deg(\varphi_{Bool}) \leq 1 \tag{25}$$

Proof. Since both l_1 and l_2 are of degree 1, there exists two unique sets of coefficients $(\alpha_i^{(1)})_{0 \leq i \leq n} \in \mathbb{R}$ and $(\alpha_i^{(2)})_{1 \leq i \leq n} \in \mathbb{R}$ such that:

$$l_j(z) = \alpha_0^{(j)} + \sum_{i=1}^{n} \alpha_i^{(j)} \cdot z[i] \tag{26}$$

where $z[i]$ stands for the i^{th} bit of z. Since the covariance involves a centered product, one can suppose without loss of generality that $\alpha_0^{(j)} = 0$ (we removed $\alpha_0^{(j)}$ for readability reasons but it does not change anything to the proof). Injecting Equation 26 into the expression of φ_{Bool}:

$$\begin{aligned}
\varphi_{Bool}(z) &= \frac{1}{|\mathcal{Z}|} \sum_{z_2 \in \mathcal{Z}} \big(\sum_{i=1}^{n} \alpha_i^{(1)} \cdot (z \oplus z_2)[i] - \mu_1\big) \cdot \big(\sum_{i=1}^{n} \alpha_i^{(2)} \cdot z_2[i] - \mu_2\big) \\
&= \frac{1}{|\mathcal{Z}|} \sum_{z_2 \in \mathcal{Z}} \big(\sum_{i=1}^{n} \alpha_i^{(1)} \cdot (z[i] \oplus z_2[i]) - \mu_1\big) \cdot \big(\sum_{i=1}^{n} \alpha_i^{(2)} \cdot z_2[i] - \mu_2\big)
\end{aligned} \tag{27}$$

Using the identity : $z[i] \oplus z_2[i] = z[i] + z_2[i] - 2 \cdot (z[i] \wedge z_2[i])$ where $\wedge$ stands for the Boolean AND:

$$\begin{aligned}
\varphi_{Bool}(z) &= \frac{1}{|\mathcal{Z}|} \sum_{z_2 \in \mathcal{Z}} \big(\sum_{i=1}^{n} \alpha_i^{(1)} \cdot (z[i] + z_2[i] - 2(z[i] \wedge z_2[i])) - \mu_1\big) \cdot \big(\sum_{i=1}^{n} \alpha_i^{(2)} \cdot z_2[i] - \mu_2\big) \\
&= \frac{1}{|\mathcal{Z}|} \sum_{z_2 \in \mathcal{Z}} \sum_{i=1}^{n} \big(\alpha_i^{(1)} \cdot (z[i] + z_2[i] - 2(z[i] \wedge z_2[i])) - \mu_1\big) \cdot \big(\sum_{i=1}^{n} \alpha_i^{(2)} \cdot z_2[i] - \mu_2\big) \\
&= \frac{1}{|\mathcal{Z}|} \sum_{i=1}^{n} \sum_{z_2 \in \mathcal{Z}} \big(\alpha_i^{(1)} \cdot (z[i] + z_2[i] - 2(z[i] \wedge z_2[i])) - \mu_1\big) \cdot \big(\sum_{i=1}^{n} \alpha_i^{(2)} \cdot z_2[i] - \mu_2\big) \\
&= \frac{1}{|\mathcal{Z}|} \sum_{i=1}^{n} \sum_{\substack{z_2 \in \mathcal{Z} \\ z_2[i]=0}} \big(\alpha_i^{(1)} \cdot (z[i] + z_2[i]) - \mu_1\big) \cdot \big(\sum_{i=1}^{n} \alpha_i^{(2)} \cdot z_2[i] - \mu_2\big) + \\
&\qquad \sum_{i=1}^{n} \sum_{\substack{z_2 \in \mathcal{Z} \\ z_2[i]=1}} \big(\alpha_i^{(1)} \cdot (-z[i] + z_2[i]) - \mu_1\big) \cdot \big(\sum_{i=1}^{n} \alpha_i^{(2)} \cdot z_2[i] - \mu_2\big) \\
&= \frac{1}{|\mathcal{Z}|} \sum_{i=1}^{n} z[i] \cdot \Big[\sum_{\substack{z_2 \in \mathcal{Z} \\ z_2[i]=1}} \big(\alpha_i^{(1)} \cdot z_2[i] - \mu_1\big) \cdot \big(\sum_{i=1}^{n} \alpha_i^{(2)} \cdot z_2[i] - \mu_2\big) - \\
&\qquad \sum_{i=1}^{n} \sum_{\substack{z_2 \in \mathcal{Z} \\ z_2[i]=1}} \big(\alpha_i^{(1)} \cdot z_2[i] - \mu_1\big) \cdot \big(\sum_{i=1}^{n} \alpha_i^{(2)} \cdot z_2[i] - \mu_2\big) \Big]
\end{aligned} \tag{28}$$

which is of degree at most 1 since the $z[i]$ terms are not mixed. □

Proposition 2. *(Arithmetic masking) Let $* = + \, mod \, 2^n$. Let $l_1 : \mathcal{Z} \to \mathbb{R}$ and $l_2 : \mathcal{Z} \to \mathbb{R}$ be two leakage functions of degre 1. Let $\varphi_{Arith}(z) = \operatorname{cov}\big(l_1(z + Z_2 \, [2^n]), l_2(Z_2)\big)$. Then,*

$$deg(\varphi_{Arith}) \leq 2 \tag{29}$$

Proof. We give a proof by induction. Let define the property $\mathcal{P}_n$:

$\mathcal{P}_n$ **:** For any l_1 and l_2 of degree 1, $deg(\varphi_n) \leq 2$, where for $z \in \mathcal{Z} = \mathbb{F}_2^n$:

$$\varphi_n(z) = \operatorname{cov}\big(l_1(z + Z_2 \, [2^n]), l_2(Z_2)\big)$$

Initialisation. The case $n = 1$ is trivial since $deg(\varphi_{Arith})$ is at most 1 in this case.

Induction. Let suppose that $\mathcal{P}_n$ holds. We are going to prove that $\mathcal{P}_{n+1}$ also holds. Since both l_1 and l_2 are of degree 1, there exists two unique sets of coefficients $(\alpha_i^{(1)})_{0 \leq i \leq n+1} \in \mathbb{R}$ and $(\alpha_i^{(2)})_{0 \leq i \leq n+1} \in \mathbb{R}$ such that:

$$l_j(z) = \alpha_0^{(j)} + \sum_{i=1}^{n+1} \alpha_i^{(j)} \cdot z[i] \tag{30}$$

where $z[i]$ stands for the i^{th} bit of z. Since the covariance involves a centered product, one can suppose without loss of generality that $\alpha_0^{(j)} = 0$ (we removed $\alpha_0^{(j)}$ for readability reasons but it does not change anything to the proof). Injecting this into the expression of φ_{n+1} one has:

$$\varphi_{n+1}(z) = \sum_{z_2=0}^{2^{n+1}-1} \big(\sum_{i=1}^{n+1} \alpha_i^{(1)} \cdot (z + z_2 \, [2^{n+1}])[i] - \mu_1\big) \cdot \big(\sum_{i=1}^{n+1} \alpha_i^{(2)} \cdot z_2[i] - \mu_2\big) \tag{31}$$

for $i \in [\![1, n+1]\!]$, the following identity holds: $(z + z_2 \, [2^{n+1}])[i] = (z + z_2)[i]$. Indeed, the modulo corresponds to either doing nothing or subtracting 2^{n+1} when $z + z_2 \geq 2^{n+1}$. Then:

$$\begin{aligned}
\varphi_{n+1}(z) &= \sum_{z_2=0}^{2^{n+1}-1} \big(\sum_{i=1}^{n+1} \alpha_i^{(1)} \cdot (z + z_2)[i] - \mu_1\big) \cdot \big(\sum_{i=1}^{n+1} \alpha_i^{(2)} \cdot z_2[i] - \mu_2\big) \\
&= \sum_{z_2=0}^{2^n-1} \big(\sum_{i=1}^{n+1} \alpha_i^{(1)} \cdot (z + z_2)[i] - \mu_1\big) \cdot \big(\sum_{i=1}^{n+1} \alpha_i^{(2)} \cdot z_2[i] - \mu_2\big) + \\
&\quad \sum_{z_2=2^n}^{2^{n+1}-1} \big(\sum_{i=1}^{n+1} \alpha_i^{(1)} \cdot (z + z_2)[i] - \mu_1\big) \cdot \big(\sum_{i=1}^{n+1} \alpha_i^{(2)} \cdot z_2[i] - \mu_2\big) \\
&= \sum_{z_2=0}^{2^n-1} \big(\sum_{i=1}^{n} \alpha_i^{(1)} \cdot (z + z_2)[i] + \alpha_{n+1}^{(1)} \cdot (z + z_2)[n+1] - \mu_1\big) \cdot \\
&\quad \big(\sum_{i=1}^{n} \alpha_i^{(2)} \cdot z_2[i] + \alpha_{n+1}^{(2)} \cdot z_2[n+1] - \mu_2\big) + \\
&\quad \sum_{z_2=2^n}^{2^{n+1}-1} \big(\sum_{i=1}^{n} \alpha_i^{(1)} \cdot (z + z_2)[i] + \alpha_{n+1}^{(1)} \cdot (z + z_2)[n+1] - \mu_1\big) \cdot \\
&\quad \big(\sum_{i=1}^{n} \alpha_i^{(2)} \cdot z_2[i] + \alpha_{n+1}^{(2)} \cdot z_2[n+1] - \mu_2\big)
\end{aligned} \tag{32}$$

Again, one can add a $[2^n]$ in the $(z+z_2)[i]$ terms since it does not change anything for $i \in [\![1, n]\!]$. Then:

$$\begin{aligned}\varphi_{n+1}(z) = &\sum_{z_2=0}^{2^n-1} \Big(\sum_{i=1}^{n} \alpha_i^{(1)} \cdot (z + z_2\ [2^n])[i] - \mu_1\Big) \cdot \Big(\sum_{i=1}^{n} \alpha_i^{(2)} \cdot z_2[i]\ - \mu_2\Big) + \\ &\sum_{z_2=2^n}^{2^{n+1}-1} \Big(\sum_{i=1}^{n} \alpha_i^{(1)} \cdot (z + z_2\ [2^n])[i] - \mu_1\Big) \cdot \Big(\sum_{i=1}^{n} \alpha_i^{(2)} \cdot z_2[i] - \mu_2\Big) + \\ &\sum_{z_2=0}^{2^{n+1}-1} \big(\alpha_{n+1}^{(1)} \cdot (z+z_2)[n+1]\big) \cdot \big(\alpha_{n+1}^{(2)} \cdot z_2[n+1]\big)\end{aligned} \tag{33}$$

The second line of Equation 33 can be re-indexed summing from 0 to $2^n - 1$. Then, by $\mathcal{P}_n$, the first two line of Equation 33 are of degree at most 2. So let us focus on the last term denoted A and prove that it is also of degree at most 2:

$$\begin{aligned} A &= \sum_{z_2=0}^{2^{n+1}-1} \big(\alpha_{n+1}^{(1)} \cdot (z+z_2)[n+1]\big) \cdot \big(\alpha_{n+1}^{(2)} \cdot z_2[n+1]\big) \\ &= \alpha_{n+1}^{(1)} \cdot \alpha_{n+1}^{(2)} \cdot \sum_{z_2=2^n}^{2^{n+1}-1} (z+z_2)[n+1] \end{aligned} \tag{34}$$

since $z_2[n+1] = 0$ implies that all the term in the sum are equal to 0.
One can notice that the latter sum has two expression depending on the $(n+1)^{th}$ bit of z:

$$\sum_{z_2=2^n}^{2^{n+1}-1} (z+z_2)[n+1] = \begin{cases} 2^n - z & \text{if } z[n+1] = 0 \\ z - 2^n & \text{if } z[n+1] = 1 \end{cases} \tag{35}$$

Therefore:

$$\begin{aligned} A &= \alpha_{n+1}^{(1)} \cdot \alpha_{n+1}^{(2)} \cdot (z - 2^n) \cdot (2 \cdot z[n+1] - 1) \\ &= \alpha_{n+1}^{(1)} \cdot \alpha_{n+1}^{(2)} \cdot \Big(\sum_{k=1}^{n+1} 2^{k-1} \cdot z[k] - 2^n\Big) \cdot (2 \cdot z[n+1] - 1) \end{aligned} \tag{36}$$

which is of degree at most 2 since developing the latter sum involves product of at most 2 bits of z together.

Injecting this into Equation 33 show that $deg(\varphi_{n+1}) \leq 2$ and therefore that $\mathcal{P}_{n+1}$ holds. This concludes the induction and therefore the proof of Proposition 2.

For the interested reader, we give as a bonus the coefficients of φ_{Arith} in terms of $\alpha_i^{(j)}$:

$$\varphi_{Arith} = \alpha_0 + \sum_{i=1}^{n} \alpha_i \cdot z[i] + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \alpha_{i,j} \cdot z[i]z[j] \tag{37}$$

With:

$$\begin{aligned} \alpha_0 &= \frac{1}{4} \cdot \sum_{k=1}^{n} \alpha_k^{(1)} \alpha_k^{(2)} \\ \alpha_i &= -\sum_{k=1}^{i} \frac{\alpha_k^{(1)} \alpha_k^{(2)}}{2^{i-k}}, \text{ for } 1 \leq i \leq n \\ \alpha_{i,j} &= \frac{\alpha_i^{(1)} \alpha_i^{(2)}}{2^{j-i}}, \text{ for } 1 \leq i < j \leq n \end{aligned} \tag{38}$$

□