Analytical Approach for Numerical Accuracy Estimation of Fixed-Point Systems Based on Smooth Operations
Romuald Rocher, Daniel Ménard, Olivier Sentieys, Pascal Scalart

To cite this version:

HAL Id: hal-00741741
https://inria.hal.science/hal-00741741
Submitted on 15 Oct 2012

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Analytical Approach for Numerical Accuracy Estimation of Fixed-Point Systems Based on Smooth Operations

Romuald Rocher, Daniel Menard, Pascal Scalart, and Olivier Sentieys Members, IEEE
University of Rennes 1-INRIA/IRISA
6 rue de Kerampont - 22305 Lannion, France
Email: name@irisa.fr

Abstract—In embedded systems using fixed-point arithmetic, converting applications into fixed-point representations requires a fast and efficient accuracy evaluation. This paper presents a new analytical approach to determine an estimation of the numerical accuracy of a fixed-point system, which is accurate and valid for all systems formulated with smooth operations (e.g., additions, subtractions, multiplications and divisions). The mathematical expression of the system output noise power is determined using matrices to obtain more compact expressions. The proposed approach is based on the determination of the time-varying impulse-response of the system. To speedup computation of the expressions, the impulse response is modelled using a linear prediction approach. The approach is illustrated in the general case of time-varying recursive systems by the Least Mean Square (LMS) algorithm example. Experiments on various and representative applications show the fixed-point accuracy estimation quality of the proposed approach. Moreover, the approach using the linear-prediction approximation is very fast even for recursive systems. A significant speed-up compared to the best known accuracy evaluation approaches is measured even for the most complex benchmarks.

Index Terms—Fixed-point arithmetic, quantization noises, adaptive filters, accuracy evaluation

I. INTRODUCTION

Digital image and signal processing applications are mostly implemented in embedded systems with fixed-point arithmetic to satisfy energy consumption and cost constraints [14], [16], [33], [15]. Fixed-point architectures manipulate data with relatively small word-lengths, which could offer the advantages of a significantly lower area, latency, and power consumption. Moreover, floating-point operators are more complex in their processing of exponent and mantissa and, therefore, chip area and latency are more important than for fixed-point operators.

Nevertheless, applications in their earliest design cycle are usually designed, described and simulated with floating-point data types. Consequently, application must be converted into a fixed-point specification where implementation is considered. To reduce application time-to-market, tools to automate fixed-point conversion are needed. Indeed, some experiments [7] have shown that the manual fixed-point conversion process can represent from 25% to 50% of the total development time. Likewise, in a survey conducted by a design tool provider [19], the fixed-point conversion was identified by a majority of respondents as one of the most difficult aspects during FPGA implementation of a DSP application. The aim of fixed-point conversion tools is to determine the optimized fixed-point specification that minimizes the implementation cost and provides a sufficient computation accuracy to maintain the application performance.

The fixed-point conversion process is divided into two steps to determine the number of bits for both integer and fractional parts. The first step is the determination of the binary-point position from the data dynamic range [36]. The integer part word-length of all data inside the application is determined in order to avoid overflow. The second step is the determination of the fractional part. The limitation of the number of bits for the fractional part leads to an unavoidable error resulting from the difference between the expected value and the real finite precision value. Obviously, numerical accuracy effects must be limited as much as possible to ensure algorithm integrity and application performance. Therefore, this stage of the fixed-point conversion flow corresponds to an optimization process in which the implementation cost is minimized under an accuracy constraint.

The fractional part word-length is optimized with an iterative process requiring to evaluated the numerical accuracy numerous times. Therefore precise and fast numerical accuracy estimator is a key element of any efficient fixed-point conversion method.

The numerical accuracy of an application can be obtained using two approaches: fixed-point simulations and analytical approaches. Accuracy evaluation based on fixed-point simulations [2], [20] is very time consuming because a new simulation is needed as soon as a fixed-point format changes in the system resulting in prohibitive optimization time for the fixed-point conversion process. Analytical approaches attempt to determine a mathematical expression of an estimation of a numerical accuracy metric, which leads to short evaluation times compared to fixed-point simulation based approaches. Consequently, the rest of the paper focuses on improving the quality and the application range of analytical approaches for numerical accuracy evaluation.

Numerical accuracy can be evaluated by determining one of the following metrics: quantization error bounds [13], [1], number of significant bits [6] or quantization noise power [27], [31], [5]. The error bound metric (or the number of significant bits) is used for critical systems where the quantization effects could have catastrophic consequences [21], [3]. The error has to be kept small enough to ensure the system behavior within acceptable bounds. The quantization error power is preferred when a slight degradation of application performance due to quantization effects is acceptable. This design approach concerns a great number of digital signal processing applications. Moreover, the quantization error can be linked to the degradation of the application performance using for example the technique presented in [26]. In this paper, the quantization noise power is used as the numerical accuracy metric.

This paper proposes an efficient and accurate analytical approach to estimate the numerical accuracy of all types of fixed-point systems based on smooth operations (an operation is considered to be smooth if the output is a continuous and differentiable function of its inputs, as in the case of arithmetic operations). From the signal flow graph of the application, the associated tool automatically generates the source code implementing the analytical expression of the output quantization noise power. In this expression, the gains between the different noise sources and the output are computed from the impulse response (IR), which can be time-varying. This expression
is unbiased but introduces infinite sums. To reduce the computation time of the gains, an approach based on linear prediction is also proposed. The approach is based on a matrix model which leads to a compact expression in the case of algorithms expressed with vectors and matrices, such as the Fast Fourier Transform (FFT) and the Discrete Cosine Transform (DCT). Compared to approaches based on fixed-point simulations, the time required to evaluate the fixed-point accuracy is also dramatically reduced. Existing analytical approaches are only valid for linear and time invariant (LTI) systems [22], [23], or for non-LTI and non recursive systems [24]. In [4], non-LTI systems with feedbacks are handled. But, as mentioned in [4], the execution time can be quite long for systems containing feedbacks. Indeed, to obtain an accurate estimation, this execution time depends on the length of the system impulse response. Compared to the last-mentioned approach, a significant speed-up is obtained with the proposed approach thanks to the proposed linear prediction approach.

This paper is organized as follows. In Section II, existing methods to evaluate the numerical accuracy of fixed-point applications are detailed. The comparison of simulation-based and analytical approaches highlights the benefits of analytical approaches. In Section III, the proposed approach for numerical accuracy evaluation is defined. The quantization noise propagation model is presented and the output noise power expression is expressed. A technique based on linear prediction is also presented to reduce the time to obtain the power expression. In Section IV, the approach is tested on several LTI and non LTI systems. The quality of the estimation and the time to obtain the power expression are evaluated on various applications and benchmarks. Finally, Section V concludes the paper.

II. EXISTING APPROACHES FOR NUMERICAL ACCURACY EVALUATION

In the fixed-point conversion process, the numerical accuracy evaluation is the most critical part, mainly because this evaluation is successively performed in the iterative word-length optimization process. The numerical accuracy can be evaluated using analytical or simulation-based approaches.

A. Accuracy Evaluation Based on Fixed-Point Simulations

In fixed-point simulation-based approaches, the application performance and the numerical accuracy are directly determined from fixed-point (bit-true) simulations. To obtain accurate results, the simulations must use a high number of samples $N_{samp}$. Let $N_{ops}$ denote the number of operations in the algorithm description. $N_I$—the number of times the accuracy is evaluated and $\tau$—the simulation time for one sample. Therefore, the total system simulation time is

$$T_{sim} = \tau N_{samp} N_{ops} N_I.$$  (1)

To reduce word-length optimization time, approaches based on fixed-point simulations aim at reducing the values of some parameters of (1). First, fixed-point simulators have to minimize the computing time $\tau$ by using efficient data types to simulate fixed-point arithmetic. Most of them use object oriented language and operator overloading [20] to implement fixed-point arithmetic. However, the fixed-point execution time of an algorithm is significantly longer than the one of a floating-point simulation [12]. To reduce this overhead, new data types have been introduced in [20]. They use more efficiently the floating-point units of general-purpose processors. With this new data type, the fixed-point simulation time is still 7.5 times longer than the one obtained with floating-point data types for a simple application like a fourth-order Infinite Impulse Response (IIR) filter [20]. However, the optimization time to accelerate the fixed-point simulations is not given in [20], which could actually annihilate the obtained gain for the fixed-point simulation. The approach proposed in [30] was developed for MATLAB code but its simulation time is still significantly higher. Finally, some methods [2] attempt to reduce $N_e$, the number of iteration of the optimization process by limiting the search space, and thus leading to sub-optimal solutions.

B. Analytical Accuracy Evaluation

The general idea of analytical approaches is to determine a mathematical expression that provides estimations of the output quantization noise power. The computation of this analytical expression could be time-consuming, but it is executed only once. Then, the numerical accuracy is determined very quickly by evaluating the mathematical expression for a given word-length of each data. Existing analytical approaches for numerical accuracy evaluation are based on perturbation theory [8], [34]. The quantization of a signal $x$ generates a noise $b_c$ considered as a signal perturbation. Let $x_i$ and $y$ be respectively the $n$ inputs and the output of an operation $f$, and $b_{x_i}$ and $b_y$ their perturbations. The perturbation of the output value of a differentiable operation is given by

$$b_y = b_{x_1} \frac{df}{dx_1} + \ldots + b_{x_n} \frac{df}{dx_n}. \quad (2)$$

Then, the expression of the noise power $P_b$ corresponding to the second-order moment of the output quantization noise can be expressed as

$$P_b = \sum_{i=1}^{N_e} \sigma_{K_i}^2 + \sum_{i,j} m_{b_i} m_{b_j} L_{ij}, \quad (3)$$

where $N_e$ is the number of quantization noises, $m_{b_i}$ and $\sigma_{K_i}^2$ are respectively the mean value and the variance of the noise $b_i$ and the terms $K_i$ and $L_{ij}$ depend on system characteristics. The expressions of $K_i$ and $L_{ij}$ differ according to the considered approach and will be presented in the following subsection. Determining the terms $K_i$ and $L_{ij}$ is crucial for the analytical accuracy evaluation process.

1) Simulation-Based Computation: Approaches based on hybrid techniques [31], [10], [17] have been proposed to compute the coefficients $K_i$ and $L_{ij}$ of (3) from a set of simulations. In [31], $N_e \times (N_e + 1)$ coefficients are obtained by solving a linear system in which $K_i$ and $L_{ij}$ are the variables. The other elements of (3) are determined by carrying out fixed-point simulations. The word-lengths of the data are set to control each quantizer and to analyze its influence on the output. For each simulation, the output noise power is evaluated. The statistical parameters (mean and variance) of each noise source $b_i$ are extracted from the data word-length. At least $N_e \times (N_e + 1)$ fixed-point simulations are required to solve the system of linear equations. A similar approach to [31] is used in [17] to obtain the coefficients by simulation.

2) Affine Arithmetic Based Simulations: In approach based on affine arithmetic [4], the coefficients $K_i$ and $L_{ij}$ are computed with an affine arithmetic based simulation. This approach was proposed for LTI in [22], [23] and recently, for non-LTI systems in [4].

In affine arithmetic, a data $x$ is defined by a linear relation

$$x = x_0 + \epsilon_1 x_1 + \ldots + \epsilon_n x_n,$$  (4)

where $\epsilon_i$ is an uncertainty term included in the interval $[-1, 1]$ and such that $\epsilon_i$ is independent of $\epsilon_j \forall i \neq j$. Each noise source is defined by an affine form whose behavior is controlled by the mean and the variance of the noise source. Then, noise source expressions are propagated throughout the system thanks to an affine arithmetic based...
simulation. The values of $K_i$ and $L_{ij}$ are extracted from the affine form of the output noise. The approach proposed in [4] is general and can handle systems based on smooth operations. This estimation is accurate and the execution time is low for non-recursive systems. Nevertheless, in the case of recursive systems, several iterations for the affine arithmetic based simulation are required to converge to stable values. Thus, the execution time depends on the length of the impulse response and on the number of noise sources. As mentioned in [4], the execution time can be quite long for systems containing feedback loops.

3) Approaches Based on System Characteristics: The approaches presented in [11], [25] provide an approximation of the mathematical expression of the output quantization noise power in the case of Linear and Time Invariant (LTI) systems. The principle of [25] is based on the system transfer functions and the expressions of the coefficients $K_i$ and $L_{ij}$ are

$$K_i = \frac{1}{2\pi} \int_{-\pi}^{\pi} |H_i(e^{j\Omega})|^2 d\Omega, \quad (5)$$

$$L_{ij} = H_i(1) H_j(1), \quad (6)$$

where $H_i(z)$ is the transfer function between the white noise source $b_i$ and the system output. In [25], the authors proposed a technique to automatically obtain the transfer function from the application description.

For non-LTI and non-recursive systems, an extended approach was presented in [24]. Each noise source $b_i(n)$ is propagated through $M_i$ operations $v_{M_i}$, which leads to the calculation of the output noise $b'_i(n)$. The noise $b'_i(n)$ is the product of each noise source $b_i(n)$ and of different signals $\alpha_k$ associated to operations $v_{M_i}$ included in noise source $b_i(n)$ propagation. So, $K_i$ and $L_{ij}$ terms are equal to

$$K_i = E\left[\prod_{k=1}^{M_i} \alpha^2_k\right], \quad (7)$$

$$L_{ij} = E\left[\prod_{k=1}^{M_i} \prod_{l=1}^{M_j} \alpha_k \alpha_l\right]. \quad (8)$$

It is noteworthy that this approach is adequate for both non-LTI and non-recursive systems, but not for non-LTI and recursive systems such as the clan of adaptive filters.

C. Summary

While approaches based on fixed-point simulations lead to very long computing time and thus require to limit the search space during word-length optimization, analytical approaches are significantly less time-consuming. Unfortunately, they are restricted to a reduced class of systems and are not always efficient in terms of execution time. To cope with the lack of efficient and general approaches, an analytical approach for all types of systems based on smooth operations is presented in the next section.

III. PROPOSED ACCURACY EVALUATION APPROACH

In this section, a new approach for evaluating the quantization noise at the output of any system based on smooth operations (addition, subtraction, multiplication and division) is detailed. First, the models associated with the quantization process and the quantization noise propagation model are introduced. Then, the modelling of the system is presented and the output noise power expression is detailed.

A. Quantization Noise Models

1) Quantization Noise Source: The quantization process can be modelled by the sum of the original signal and of a uniformly distributed white noise [35], [32]. This quantization noise is uncorrelated with the signal and the other noise sources. Such a model is valid provided that the signal has a sufficiently large dynamic range compared to the quantization step. According to the quantization mode, the noise Probability Density Function (PDF) will differ. Three quantization modes are usually considered: truncation, conventional rounding, and convergent rounding. In the truncation mode, the Least Significant Bits (LSB) are directly eliminated. The resulting number is always smaller than or equal to the number obtained before quantization, and, therefore, the quantization noise is always positive. Consequently, the mean of the quantization noise is not equal to zero. To reduce the bias due to truncation, the rounding quantization mode is often used. In conventional rounding, the data are rounded to the nearest value representable in the reduced-accuracy format. For numbers located at the midpoint between two consecutive representable values, the data are rounded-up always to the larger output value. This technique leads to a (small) bias for the quantization noise. To eliminate this quantization noise bias, the convergent rounding can be used as well. In this case, the numbers located at the midpoint between two consecutive representable values are, with equal probability, rounded to the higher or lower output value.

Let $w_{FP}$ denote the number of bits for the fractional part after the quantization process and $k$—the number of bits eliminated during the quantization. The quantization step $q$ after the quantization is equal to $q = 2^{-w_{FP}}$. The quantization noise mean and variance are given in Table I for the three considered quantization modes [9], [28].

<table>
<thead>
<tr>
<th>Quantization mode</th>
<th>Truncation</th>
<th>Conventional rounding</th>
<th>Convergent rounding</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>$\frac{q}{2}(1 - 2^{-k})$</td>
<td>$\frac{q}{2}(2^{-k})$</td>
<td>0</td>
</tr>
<tr>
<td>Variance</td>
<td>$\frac{q^2}{12}(1 - 2^{-2k})$</td>
<td>$\frac{q^2}{12}(1 - 2^{-2k})$</td>
<td>$\frac{q^2}{12}(1 + 2^{-2k+1})$</td>
</tr>
</tbody>
</table>

TABLE I

FIRST- AND SECOND-ORDER MOMENTS FOR THE THREE CONSIDERED QUANTIZATION MODES.

2) Quantization Noise Propagation: The aim of this part is to define the model for the propagation of the quantization noise for smooth operations. We specifically concentrate on an operation $f$ with two inputs $x$ and $y$ and on the calculation of its output $z = f(x, y)$. The finite precision values $\hat{x}$, $\hat{y}$ and $\hat{z}$ are respectively the sum of infinite precision values $x$, $y$ and $z$ with their associated quantization noises $b_x$, $b_y$ and $b_z$. The propagation model is then defined such as $b_z$ is a linear combination of $b_x$ and $b_y$:

$$b_z = \alpha_1 b_x + \alpha_2 b_y. \quad (9)$$

$\alpha_1$ and $\alpha_2$ are obtained from a first order Taylor development at the output of the differentiable operator $f$ defined as

$$z = f(\hat{x}, \hat{y}) = f(x, y) + (\hat{x} - x) \frac{\partial f}{\partial x}(x, y) + (\hat{y} - y) \frac{\partial f}{\partial y}(x, y). \quad (10)$$

Thus, the expressions of $\alpha_1$ and $\alpha_2$ are

$$\alpha_1 = \frac{\partial f}{\partial x}(x, y) \quad \text{and} \quad \alpha_2 = \frac{\partial f}{\partial y}(x, y). \quad (11)$$
The values of $\alpha_1$ and $\alpha_2$ are summarized in Table II for the arithmetic operations.

<table>
<thead>
<tr>
<th>Operation</th>
<th>$\alpha_1$</th>
<th>$\alpha_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$z = x \pm y$</td>
<td>1</td>
<td>$\pm 1$</td>
</tr>
<tr>
<td>$z = x \times y$</td>
<td>$y$</td>
<td>$x$</td>
</tr>
<tr>
<td>$z = \frac{x}{y}$</td>
<td>$\frac{1}{y}$</td>
<td>$-\frac{x}{y^2}$</td>
</tr>
</tbody>
</table>

**Table II**

Values $\alpha_1$ and $\alpha_2$ of equation (9) for arithmetic operations

In the rest of the paper, upper-case letters in bold will represent matrices. However, when applied to non-scalar multiplications and since we seek to obtain a general model, each noise source on the operation input is multiplied by a signal term on the left or on the right. Therefore, in the general case, the noise source propagation is expressed as the multiplication of each noise input by two matrices ($A$ on the left and $D$ on the right) as

$$B_s = A_x B_x D_x + A_y B_y D_y.$$

To illustrate the need of the two matrices $A$ and $D$, the adaptive filter based on the Affine Projection Algorithm (APA) is then considered. APA has been proposed in [29] to have a faster convergence compared to the Least Mean Square (LMS) algorithm and to reduce complexity compared to Recursive Least Square (RLS) algorithm. Let $X(n)$ denote an $N \times K$ matrix made-up of the $K$ last observation vectors of the input, $w(n)$—the coefficient vector and $e(n)$ the error vector. The updated coefficient equation is given by

$$w(n+1) = w(n) + \mu X(n)(X^T(n)X(n))^{-1}e(n).$$

The noise matrix $B_x(n)$ associated to the inversion propagation matrix $(X^T(n)X(n))^{-1}$ is then multiplied by two terms

$$A = \mu X(n),$$

$$d = e(n).$$

**B. System Model**

Each noise source propagates throughout the system and contributes to the global output noise. In this section, the system characterizing each noise propagation is defined to compute afterward the expression of the system output noise power. Let $N_e$ denote the number of noise sources inside the system. As shown in Table II, the noise expression does not contain crossed noise terms. So, each noise source $b_i(n)$ at time $n$ contributes to the output noise by a noise $b_i(n)$ that is independent of all other noise terms. The system output noise $b_y(n)$ is the sum of all contributions as presented in Figure 1 and expressed by

$$b_y(n) = \sum_{i=1}^{N_e} b_i'(n).$$

Each noise contribution $b_i'(n)$ is obtained from the propagation of noise sources $b_i(n)$ through the system $S_i$. Thus, to determine the complete expression of the output noise $b_y(n)$, each subsystem $S_i$ must be characterized analytically.

1) **System Characterization:** The contribution $b_i'(n)$ of the noise source $b_i(n)$ depends on the previous samples $b_i(n-k)$ of the noise source with $k \in \{1,2,..Q_i\}$. If the system $S_i$ is recursive, the noise $b_i'(n)$ depends on its previous samples $b_i(n-m)$ with $m \in \{1,2,..P_i\}$. Thus, the general expression of the quantization noise $b_i(n)$ is

$$b_i'(n) = \sum_{k=0}^{Q_i} g_i(k)b_i(n-k) + \sum_{m=1}^{P_i} f_i(m)b_i(n-m),$$

where $g_i(k)$ represents the contribution of noise source $b_i$ at time $(n-k)$ to the output noise and $f_i(m)$—the contribution of noise $b_i$ at time $(n-m)$. The terms $f_i$ and $g_i$ may be time-varying and

**Fig. 1.** System noise model for $N_e$ input noises $b_i(n)$ and output noise $b_y(n)$
depend on the system implementation. For LTI systems, the terms \( f_i \) and \( g_i \) are constant and correspond to filter coefficients.

Nevertheless, to compute the output noise power expression from the statistical parameters of the noise \( b_i(n) \), (20) must be developed to express contribution \( b'_i(n) \) with only input noise terms \( b_i \). This development introduces the time-varying impulse response associated with the system \( S_i \).

2) Time-Varying Impulse Response: In this part, the time-varying impulse response \( h_i \) of the system \( S_i \) is determined. By developing the recurrence in (20), the noise term becomes

\[
b'_i(n) = \sum_{k=0}^{n} h_i^{(n)}(k) b_i(n-k)
\]

where \( h_i^{(n)}(k) \) is the time-varying impulse response of the system \( S_i \) which formulates the noise \( b'_i(n) \) only from the samples of the noise source \( b_i(k) \). The term \( h_i^{(n)}(k) \) represents the contribution of the noise source \( b_i(n-k) \) at time \( n-k \) to generate \( b'_i(n) \) at time \( n \). For the sake of clarity, \( h_i^{(n)}(k) \) is noted by \( h_i(k) \) since output time is always considered at time \( n \). This impulse response is obtained from the terms \( f_i \) and \( g_i \) with

\[
h_i(k) = \sum_{j=1}^{p_i} f_i(j) h_i(k-j) + g_i(k)
\]

was obtained in the case of scalar noise sources. The model can be extended to vector (or matrix) noise sources by using vector (or matrix) terms for the time-varying impulse response. In this case, the time-varying impulse response is equivalent to the multiplication of the noise source \( B_i(k) \) (to be more general, noises are considered as matrices in the following) by two terms \( A_i(k) \) and \( D_i(k) \). In case of non-scalar noise source, (21) can be rewritten as

\[
B'_i(n) = \sum_{k=0}^{n} A_i(k) B_i(n-k) D_i(k).
\]

The output noise \( B'_i(n) \) is finally the sum of all noise source contributions and can be expressed as

\[
B'_i(n) = \sum_{i=1}^{N_e} \sum_{k=0}^{n} A_i(k) B_i(n-k) D_i(k).
\]

The output noise \( B'_i(n) \) and input noises \( B_i(n) \) have formal dimensions of \( M_i \times N_g \) and \( M_i \times N_i \). So, \( A_i(k) \) and \( D_i(k) \) are \( M_i \times M_i \) and \( N_i \times N_i \) matrices. Their numerical values depend on the system characteristics and can be computed from a analysis of the considered application.

C. Output Noise Power Expression

1) Noise Power: The output noise power \( P_b \) is defined as the second order moment of (24) and can be expressed as

\[
P_b = E[Tr(B'_i(n)B'_i(n))],
\]

which is equivalent to

\[
P_b = \sum_{i=1}^{N_e} \sum_{j=1}^{N_e} \sum_{k=0}^{n} \sum_{m=0}^{n} E[Tr(A_i(k)B_i(n-k)D_i(k)D_i^T(m)B'_i(n-m)A'_i(m))].
\]

Equation (26) can be split into three parts by separating the noise \( B_i \) from the other noises \( B_j \) when \( j \neq i \) and the terms \( B_i(n-k) \) from \( B_i(n-m) \) when \( m \neq k \).

**Theorem 1:** Each term of \( D_i(k)D_i^T(k) \) is less or equal to its trace value.

**Proof:** The matrix \( D_i(k)D_i^T(k) \) is real, symmetric and positive and therefore diagonalizable. Thus, eigenvalues of \( D_i(k)D_i^T(k) \) are positive, and each of them is less than the trace value. Indeed, this matrix \( D_i(k)D_i^T(k) \) can be approximated by its trace \( Tr(D_i(k)D_i^T(k)) \). In the case where the noises are vectors, then \( D_i(k)D_i^T(k) \) is a scalar and is equal to its trace.

Moreover, by applying Theorem 1, (26) can be reformulated and leads to (27) given at page 6.

Following the method described in [5], the expected value of the noise and signal terms are computed separately. For the two matrices \( A \) and \( D \) made-up of signal terms and for a noise \( B \) non-correlated with these two signal terms, the following relation is used: \( E[ABD] = E[A]E[B]E[D] \). Moreover, the different quantization noises \( B_i \) are assumed to be white and independent so that the intercorrelation between two noises \( B_i \) and \( B_j \) is equal to

\[
\phi_{B_iB_j}(k-m) = E[B_i(n-k)B_j(n-m)] = \sigma_i^2 \delta(i-j) \delta(k-m) I_N + m_{ij} m_{ij} \mathbf{1}_N
\]

where \( m_{ij} \) and \( \sigma_i^2 \) represent mean and variance of each element of input noises \( B_i(n) \), \( I_N \) is the identity matrix and \( \mathbf{1}_N \) is the \( N \)-size matrix composed of 1. Thus, the output noise power \( P_b \) can be summarized as (29) given at page 6.

By introducing the terms \( K_i \) and \( L_{ij} \), (29) is reformulated as

\[
P_b = \sum_{i=1}^{N_e} \sigma_i^2 K_i + \sum_{i=1}^{N_e} \sum_{j=1}^{N_e} m_{ij} m_{ij} L_{ij}.
\]

\( K_i \) represents the gain on the variance for the noise source \( b_i \) and is equal to

\[
K_i = \sum_{k=0}^{n} E[Tr((D_i(k)D_i^T(k))D_i(k))].
\]

\( L_{ij} \) represents the gain on the mean for the couple of noise sources \( b_i \) and \( b_j \) and is equal to

\[
L_{ij} = \sum_{k=0}^{n} \sum_{m=0}^{n} E[Tr(A_i(k)1_N D_i(k)D_i^T(m)1_N A'_i(m))].
\]

The expressions of \( K_i \) and \( L_{ij} \) are obtained by a single floating-point simulation since they only involve signal terms. All the terms included in these expressions are stored from this single floating-point simulation and the terms \( K_i \) and \( L_{ij} \) are then computed. These terms are independent from noise sources and lead to constants in the output noise power expression. Noise statistics \( m_{ij} \) and \( \sigma_i^2 \) depend on fixed-point formats and are the variables describing the output noise power expression. They are computed from the data word-length of each data and operation in the application graph.

It should be noted that (29) is unbiased since no restrictive assumption was made about the system. \( K_i \) and \( L_{ij} \) are defined by infinite sums. However, different cases must be taken into account. If the system is LTI, the matrices \( A \) and \( D \) are constant, (31) and (32) can be simplified, and it leads to the results obtained in equation (5). If the system is not recursive, the length of the time-varying impulse responses is finite and the sums in (29) are finite and can be computed directly.

For the recursive case, these infinite sums are approximated by computing the first \( p \) elements of the sum, and therefore (31) and (32) are computed from \( k = 0 \) to \( p \) with \( p \) depending on the signal correlation inside the terms \( K \) and \( L \). Nevertheless, according to the different experimentations that have been carried out, a number \( p \) equal to or less than 500 leads to very realistic and accurate results. Moreover, to compute (29), the ergodic assumption is used so that
The complexity of the proposed method has been analyzed and leads to accurate modelling properties, as it will be shown in Section IV.  

For the sake of clarity on notation, let denote by \( \lambda_j \), \( \forall j \in [1, P] \), the coefficients of the impulse response \( h(k) \). Therefore, the rest of the section will focus on non-LTI recursive systems.  

The derivative of the mean square error, with respect to the coefficients \( \lambda_j \), \( \forall j \in [1, P] \), is  

\[
\frac{\partial E[\varepsilon^2]}{\partial \lambda_j} = -2E [h(k)h(k-j) - \sum_m \lambda_m h(k-m)h(k-j)].
\]

(38)

These \( P \) prediction coefficients \( \lambda_j \) can be determined by setting (38) to zero, leading to the solution \( \lambda_{opt} \) defined as  

\[
\lambda_{opt} = U^{-1}J,
\]

(39)

with

\[
U = \begin{bmatrix}
E[h^2(k-1)] & \ldots & E[h(k-1)h(k-P)] \\
E[h(k-1)h(k-2)] & \ldots & E[h(k-2)h(k-P)] \\
\vdots & \ddots & \vdots \\
E[h(k-1)h(k-P+1)] & \ldots & E[h(k-P+1)h(k-P)] \\
E[h(k-1)h(k-P)] & \ldots & E[h^2(k-P)]
\end{bmatrix}
\]

(40)

and

\[
J = \begin{bmatrix}
E[h(k)h(k-1)] \\
E[h(k)h(k-2)] \\
\vdots \\
E[h(k)h(k-P+1)] \\
E[h(k)h(k-P)]
\end{bmatrix}
\]

(41)

To determine \( J \) and \( U \), the first values of the impulse response must be known. Therefore, the approach to determine the prediction coefficients is as follows:

1) The first \( P+Q-1 \) terms of the time-varying impulse response are determined using (22), with \( Q \) and \( P \) representing respectively the maximal number of delays for the non-recursive and the recursive part of the system \( S \).

2) The two matrices \( U \) et \( J \) are determined.

3) The prediction coefficients \( \lambda_{opt} = [\lambda_1,...,\lambda_P] \) are determined by \( \lambda_{opt} = U^{-1}J \).

The coefficients \( \lambda_{opt} \) are determined according to Appendices A and B and the infinite sums can therefore be written as

\[
\sum_{m=0}^{\infty} h(m) = \frac{1}{P} \sum_{j=1}^{P} \left( \sum_{m=0}^{Q-2} \omega_m + \left( I_P - A \right)^{-1} \omega_{Q-1} + \theta \right)_{(j)}
\]

(42)

\[
\sum_{m=0}^{\infty} h(m)^2 = \frac{1}{P} \text{Tr} \left( \sum_{m=0}^{Q-2} \omega_m \omega_m^T + S + \Phi \right)
\]

(43)

where

\[
P_h = \sum_{i=1}^{N_e} \sum_{k=0}^{n} E \left[ \text{Tr}(D_i(k)D_i^T(k)) \text{Tr}(A_i(k)B_i^T(n-k)A_i^T(k)) \right]
\]

\[
+ \sum_{i=1}^{N_e} \sum_{k=0}^{n} \sum_{m} E \left[ \text{Tr}(A_i(k)B_i(n-k)D_i^T(k)B_i^T(m)A_i^T(m)) \right]
\]

\[
+ \sum_{i=1}^{N_e} \sum_{k=0}^{n} \sum_{j} E \left[ \text{Tr}(A_i(k)B_i(n-k)D_i^T(m)B_i^T(n-m)A_i^T(m)) \right]
\]

(27)

\[
P_h = \sum_{i=1}^{N_e} \sum_{k=0}^{n} \sum_{m} \sigma_i^2 E \left[ \text{Tr}(D_i(k)D_i^T(k)) \text{Tr}(A_i(k)A_i^T(k)) \right]
\]

\[
+ \sum_{i=1}^{N_e} \sum_{j} \sum m mb_i m_j \sum_{k=0}^{n} \sum_{m} E \left[ \text{Tr}(A_i(k)1_N D_i^T(m)1_N A_i^T(m)) \right]
\]

(29)

\[
N_{mult} = N_1 P \sum_{i} (2M_y M_i N_i + M_y N_y (M_i + N_i))
\]

(33)

\[
N_{add} = N_1 P \sum_{i} (2M_y M_i N_i + M_y N_y (M_i + N_i) + M_y N_y)
\]

(34)

where \( M_y \times N_y \) corresponds to the output noise matrix \( B_y(n) \) size and \( M_i \times N_i \) those of input noise \( B_i(n) \). In the next paragraph, an approach to reduce the complexity of the infinite sum computation without a significant loss of accuracy is presented.

2) Linear Prediction Approach: To reduce complexity, a linear prediction approach is proposed to replace the infinite sums. (22) has shown that impulse response terms \( h_i(k) \) are defined by a recurrent equation where the terms \( P \) corresponds to the maximal delay for the recursive part. For LTI systems, terms \( f \) and \( g \) are directly equal to transfer function coefficients. Therefore, the rest of the section will focus on non-LTI recursive systems.

As the impulse response is time-varying, the relation (22) is non linear. The aim is then to linearize this expression with prediction coefficients \( \lambda_i \) of the impulse response \( h_i(k) \) equal to

\[
\hat{h}_i(k) = \sum_{m=1}^{P_i} \lambda_i \hat{h}_i(k-m).
\]

(35)

Computing the coefficient vector \( \lambda_i = [\lambda_{i1}...\lambda_{iP_i}] \) gives a model of the impulse response term \( h_i(P_i+Q_i-1) \) with the aim to minimize the mean square error \( e_i \) between \( h_i(P_i + Q_i - 1) \) and its estimate \( \hat{h}_i(P_i + Q_i - 1) \) defined as

\[
E[e_i^2] = E[(h_i(P_i + Q_i - 1) - \hat{h}_i(P_i + Q_i - 1))^2].
\]

(36)

For the sake of clarity on \( e_i \), a term \( k = P_i + Q_i - 1 \) is introduced and the index \( i \) is removed. Thus, the mean square error is defined as

\[
E[e^2] = E[(h(k) - \hat{h}(k))^2].
\]

(37)

the statistical mean is replaced by a time average over \( N_t \) samples to get a realistic result. In practice, a number \( N_t \) equal to 100 leads to accurate modelling properties, as it will be shown in Section IV. The complexity of the proposed method has been analyzed and leads to a number of multiplications \( N_{mult} \) and additions \( N_{add} \) equal to

\[
N_{mult} = N_1 P \sum_{i} (2M_y M_i N_i + M_y N_y (M_i + N_i))
\]

\[
N_{add} = N_1 P \sum_{i} (2M_y M_i N_i + M_y N_y (M_i + N_i) + M_y N_y)
\]

(33)

(34)
By using the noise model propagation defined in Section III-A, the final expression of the power is given by:

\[ \sigma^2_n = \sigma^2_{n,b} + \sigma^2_{n,g} + 4 \mu_1 \mu_2 \sum_{i=0}^{r-2} h(k)^2 \]

and with matrix \( S \) being the solution of

\[ S - AS^{t}A = \omega_{Q-1}^{2} \omega_{Q-1}^{2}. \]

In practice, this model is applied to matrices \( A \) and \( D \) for each noise source. The complexity of this method is computed and leads to a number of multiplications \( N_{\text{mult}} \) and additions \( N_{\text{add}} \) equal to

\[ N_{\text{mult}} = \sum_i N_i \left[ (P_i + Q_i - 1) (2 M_i N_i + M_i N_i + N_i) \right] + P_i (M_i + N_i) + P_i^2 \]

\[ N_{\text{add}} = \sum_i N_i \left[ (P_i + Q_i - 1) (2 M_i N_i + M_i N_i + N_i) \right] + 4 P_i \]

D. Accuracy Evaluation Tool

The proposed approach has been implemented in a software tool to automate this process, its synoptic is given in Figure 2. Our numerical accuracy evaluation tool generates the analytical expression of the output quantization noise from the signal flow graph (SFG) of the application. This analytical expression is implemented through a C function having the fractional part \( w_{\text{FP}} \) of all data word-lengths as input parameters. This C code can be compiled and dynamically linked to the fixed-point conversion tool for the optimization process. The accuracy evaluation tool uses a technique similar to the one proposed in [25]. The first step transforms the application SFG \( G_s \) into an SFG \( G_{as} \) representing the application at the quantization noise level. All the potential noise sources are included in the graph. The expression of the variance \( \sigma^2_{n,b} \) and the mean \( m_{b_i} \) of each noise source \( b_i \) according to \( w_{\text{FP}} \) are generated in order to be included in the output C function. The expression of the number of bits eliminated between two operations can be found in [25]. Then, each operation is replaced by its propagation noise model, as presented in Section III-A2. The aim of the second step is to determine the expression of the system impulse response between the output and the different noise sources. To handle recursive systems, cycles in the graph are detected and then dismantled. The third step computes the value of the gains \( K_i \) and \( L_{ij} \) given in (31) and (32). The linear prediction approach presented in Section III-C2 is used to approximate the infinite sums. The values of the different terms of the matrices \( A \) and \( D \) are collected during a single floating-point simulation. The reference simulation and the third step of the tool are performed with the Matlab tool.

A. A non-LTI non-recursive system: Vector Normalization

The first example considers an \( N \)-size vector \( x(n) \) normalized according to the relation

\[ y(n) = \frac{x(n)}{x^2(n)x(n)}. \]

Here to simplify the final expression of the power, only two noises are considered: the noise \( b_x(n) \) due to the quantization of \( x(n) \) and the noise \( b_g(n) \) generated by the normalization computation. By using the noise model propagation defined in Section III-A, the output noise vector \( b_y(n) \) is equal to

\[ b_y(n) = \frac{b_x(n)}{x^2(n)x(n)} + 2 \frac{x(n)x^2(n)}{x^2(n)x(n)} b_x(n) + b_g(n) \]

which can be simplified as

\[ b_y(n) = \frac{I_N}{x^2(n)x(n)} + 2 \frac{x(n)x^2(n)}{x^2(n)x(n)} b_x(n) + b_g(n) \]

with \( I_N \) the \( N \)-size identity matrix. This example has no delay in the graph so only one impulse term \( A_x(n) \) has to be determined. The other terms \( A_x(k) \) are then equal to zero for \( k \neq n \) and \( d_x(n) \) is equal to one.

By applying (29) to the output noise expression, its power \( P_b \) is

\[ P_b = E[Tr(A_x^2(n)A_x(n))]\sigma^2_x + N\sigma^2_x + m^2_x E[Tr(A_x^2(n)A_x(n))] \]

\[ = Nm^2_x + 2m_x E[Tr(A_x^2(n)A_x(n))] \]

Fig. 2. Synoptic of the Accuracy Evaluation Tool

IV. RESULTS

In this section, experimental results are carried out to validate the proposed approach on LTI and non-LTI systems. Results are given on various applications: vector normalization illustrates non-LTI non-recursive systems, adaptive filters such as LMS, NLMS and APA illustrate non-LTI recursive systems, and various benchmarks illustrate all kinds of systems. For these applications the Signal-to-Quantization Noise Ratio (SQNR) values are chosen from 30 dB to 90 dB to ensure that the assumptions of the Widrow model are satisfied (the noise variance is significantly smaller than the signal power). Moreover, the optimization time of the proposed approach is compared with approaches based on fixed-point simulations.
where $m_x$ and $m_y$ are the means of $b_x(n)$ and $b_y(n)$ and $\sigma^2$ their variances.

For this example, $N$ is set to 4, and the truncation mode is considered. The fractional part word-length of the input $x$ and for the output $y$ are respectively $w_{in}$ and $w_{out}$. So, using (30) and by introducing values of mean and variance of the two considered noises, (54) can be written as:

$$P_b = K_1 \frac{2^{-2w_{in}}}{12} + K_2 \frac{2^{-2w_{out}}}{12}(1 - 2^{-2(k_{out})}) + L_{11} \frac{2^{-2w_{in}}}{4} + L_{22} \frac{2^{-2w_{out}}}{4}(1 - 2^{-2(k_{out})})^2 + L_{12} \frac{2^{-w_{out}}}{2} + L_{21} \frac{2^{-w_{in}}}{2}(1 - 2^{-2(k_{out})})^2 2^{-w_{in}}$$

(55)

where $k_{out}$ represents the number of bits eliminated at the output of the normalization. The expression of the gain $K$ and $L$ are given in Table IV.

The chosen synthetic input signal $x(n)$ is a first-order autoregressive signal defined by

$$x(n + 1) = c_x x(n) + u(n),$$

(56)

with $c \in [0, 1]$ and $u(n)$ a zero-mean white noise of variance 0.01.

The numerical values of the terms $K$ and $L$ are given in Table IV. As shown by the results, the terms $K$ and $L$ depend on signal characteristics and correlation between them.

**B. A non-LTI recursive system: the Least Mean Square (LMS) adaptive filter**

1) Fixed-Point LMS Algorithm: To illustrate the proposed model and to verify its accuracy on a non-LTI and recursive system, the Least Mean Square (LMS) example is detailed. The LMS adaptive algorithm [18] addresses the problem of estimating a sequence of scalars $y(n)$ from an $N$-length vector $x(n) = [x(n), x(n-1), \ldots x(n-N+1)]^T$. The linear estimate of $y(n)$ is $w^T(n)x(n)$ where $w(n)$ is an $N$-length vector which converges to the optimal vector $w_{opt}$ in the mean-square error (MSE) sense. This optimal vector is equal to $w_{opt} = R^{-1}v$ where $R$ corresponds to the correlation matrix of the input signal equal to $E[x(n)x^T(n)]$ and $v$ represents the intercorrelation vector between $x(n)$ and $y(n)$ equal to $E[x(n)y(n)]$. The vector $w(n)$ is updated according to

$$w(n+1) = w(n) + \mu_{LMS} x(n)(y(n) - w^T(n)x(n))$$

(57)

where $\mu_{LMS}$ is a positive constant representing the adaptation step.

In a fixed-point implementation of the LMS, four equivalent noise sources have to be considered as presented in Figure 3. $b_x(n)$ and $b_y(n)$ are noises associated respectively with the input data $x(n)$ quantization and the desired signal $y(n)$ quantization. The vector $b_{w}(n)$ is the noise vector due to the quantization of the products between $\mu_x(n)$ and the error $e(n) = y(n) - w^T(n)x(n)$. Finally, $b_i(n)$ groups all noises generated inside the filtering part during the computation of the inner product $w^T(n)x(n)$. The terms $\mu_i$ and $\sigma_i^2$ represent the mean and variance of each noise source $b_i$.

---

**TABLE IV**

<table>
<thead>
<tr>
<th>Expression and Values of the Gains $K$ and $L$ in the Case of the Vector Normalization Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gains</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>$K_1$</td>
</tr>
<tr>
<td>$K_2$</td>
</tr>
<tr>
<td>$L_{11}$</td>
</tr>
<tr>
<td>$L_{12}$</td>
</tr>
<tr>
<td>$L_{21}$</td>
</tr>
<tr>
<td>$L_{22}$</td>
</tr>
</tbody>
</table>

---

Figure 3. Fixed-point LMS algorithm and noise model for input data $x^T(n)$, estimate data $\hat{y}(n)$ and coefficients $w(n)$.
The time-varying impulse response $h_w$ is defined through two equivalent matrices as defined in Section III. For this noise, the matrix $A_w$ corresponds to an $N$-size vector $a_w$. This vector is introduced in (61) and is defined in (62).

\[
b_w(n) = \sum_{k=0}^{n-1} a_w(k)b_w(k) = x^t(n)F(n,k)
\]  
\[
a_w(k) = x^t(n)F(n,k)
\]

with

\[
F(n,k) = \prod_{m=k+1}^{n-1} \left( I_N - \mu_{LMS}x(m)x^t(m) \right).
\]

The expression of $a_w$ can be modelled using linear prediction coefficient as presented in Section III-C2. The first impulse response terms of $a_w(k)$ are

\[
a_w(n) = 0,
a_w(n - 1) = x^t(n),
a_w(n - 2) = x^t(n)(I_N - \mu_{LMS}x(n)x^t(n-1)).
\]

The LMS algorithm is a first-order recursive system. Thus, only one prediction coefficient has to be determined using the approach defined before. The matrix $U_w$, defined by (40), is a scalar in that case and is equal to

\[
U_w = E(a_w(n-1)a_w^t(n-1)) = E(x^t(n)x(n)).
\]

The term $J_w$, defined by (41), is equal to

\[
J_w = E(a_w(n-2)a_w^t(n-1)) = E(x^t(n)(I_N - \mu_{LMS}x(n)x^t(n-1))x(n)).
\]

So, from (39), the prediction coefficient $\lambda_w$ is expressed as

\[
\lambda_w = \frac{J_w}{U_w} = \frac{E[a_w(n-2)a_w^t(n-1)]}{E[a_w(n-1)a_w^t(n-1)]} = \frac{E[x^t(n)(I_N - x(n-1)x^t(n-1))x(n)]}{E[x^t(n)x(n)]}.
\]

Thus, with this coefficient, the impulse response term $a_w$ can be expressed as

\[
a_w(k) = \lambda_w a_w(k + 1),
\]

for $k < n - 1$ since $a_w(n) = 0$. The infinite sum is obtained using (42) as

\[
\sum_{k=0}^{n-\infty} a_w(k) = \frac{1}{1 - \lambda_w} a_w(n - 1).
\]
\[ E[b_p^2(n)] = \sum_{k=0}^{n} \sigma^2_x E\left[ Tr\left( a_x(k) a_x^T(k) \right) \right] + \sum_{k=0}^{n} \sigma^2_w E\left[ Tr\left( a_w(k) a_w^T(k) \right) \right] \]

\[ + \sum_{k=0}^{n} \sigma^2 \left( \sum_{l=0}^{n} E\left[ Tr\left( M(k) M^T(l) \right) \right] \right) \]

(71)

\[ E[b_p^2(n)] = \frac{\sigma^2_x E[Tr(a_x(n)a_x^T(n))]}{1 - \lambda_x^2} + \frac{\sigma^2_w E[Tr(a_w(n)a_w^T(n))]}{1 - \lambda_w^2} + \frac{\sigma^2 \left( \sum_{l=0}^{n} E[Tr(M(n)M^T(l))] \right)}{1 - \lambda^2_0} + Tr\left( TT^T \right) \]

(73)

\[ T = \frac{m_x E[a_x(n)]}{1 - \lambda_x} + \frac{m_r E[a_r(n)]}{1 - \lambda_r} + \frac{m_f E[a_f(n)]}{1 - \lambda_f} + \frac{m_w E[a_w(n)]}{1 - \lambda_w} \]

(74)

c of the input signal \( x(n) \). The input signal can be a white noise \((c = 0)\), a fairly correlated noise \((c = 0.5)\) or a very correlated noise \((c = 0.95)\). As \( p \) increases, the relative error decreases. Indeed, when \( p \) increases, more terms are included in the computation of the sums, which leads to a better estimation quality. Moreover, the relative error convergence speed depends on input data correlation. For non-correlated input data, the relative error convergence is slower than the one for very-correlated input data. In fact, the relative error is less than 20% after 300 samples for very correlated input data, after 350 samples for fairly correlated data and after 550 samples for uncorrelated input data.

![Fig. 5. Relative error for the 32-tap LMS according to the number of samples \( p \) in the computation of the infinite sums.](image)

To explain this result, we recall that the terms \( A \) representing the noise propagation through the system, include the terms \( F(n,k) \) defined as

\[ F(n,k) = \prod_{m=k+1}^{n-1} \left( I_N - \mu_{LMS} x(m)x^T(m) \right) \]

(77)

For very correlated input data, the term \( I_N - \mu_{LMS} x(m)x^T(m) \) decreases faster than for uncorrelated data. Thus, the number of elements \( p \) required to compute the infinite sums depends on the input data correlation. Nevertheless, with experimentation presented for \( p > 500 \), the relative error is less than 25% in all cases, which represents a difference equivalent to less than 1 bit between the noise power obtained with the proposed approach and the real noise power. For the linear prediction approach, the obtained relative error is equal to 21%. For other sizes \( N \) of the LMS algorithm, same kinds of results are obtained, thus indicating that the proposed approach is accurate for the LMS algorithm.

Experiments have also been conducted on other adaptive algorithms. The results are presented in Table V for the previous LMS, a 32-tap Normalized LMS (NLMS) and an Affine Projection Algorithm (APA) of size 32 × 12. The expression of the coefficient update for the APA algorithm is given in (16). The APA application is suitable to show the case where the general model presented in Section III include the two matrices \( A \) and \( D \). Both NLMS and APA algorithms moreover include division operations for the normalization module. The floating-point simulation is executed on \( N_t + p \) samples where \( N_t \) is the number of samples to compute the statistical mean and \( p \) is the number of terms to compute the sums. In these experiments, \( N_t \) is set to 100 and different values of \( p \) are tested.

The approach based on linear prediction leads to a relative error of around 20% (equal to 1 dB). However, the method using the computation of the sums can be really accurate since the relative error is less than 17.65% in mean with \( p = 100 \) and less than 6.24% with \( p = 1000 \). The accuracy of the estimation can be controlled by adjusting the number of elements \( p \) used to compute these sums. We can therefore see the term \( p \) as a way to trade-off between the accuracy of the estimation and the time required to compute this estimation.

### C. Experiments on LTI and Non-LTI Non-Recursive Systems

In this section, the proposed approach is first evaluated on LTI systems such as FIR and IIR filters, an MP3 coder/decoder and an FFT. Then, the approach is applied to non-LTI and non-recursive systems (square, Volterra filter, correlator). The average and maximum relative errors between the noise power estimated with the proposed approach and the noise power obtained by fixed-point simulations are provided.
1) **LTI Systems:** First, non-recursive systems, such as FIR filter, MP3 coder/decoder and FFT, are evaluated for different fixed-point specifications. The results are presented in Table VI. For the 32-tap FIR filter, rounding and truncation have been tested. The average relative error is 2.49% for rounding and 1.14% for truncating. Moreover, the maximum relative errors are very low (about 8%).

For the MP3 coder/decoder, the average error is equal to 5.24% for the polyphase filter and to 8.42% for the DCT. The maximum errors are 20.57% for the polyphase filter and 24.8% for the DCT. These values represent a difference of 2dB (less than 1 bit) between the real value of the noise power and the one estimated by the proposed approach. For the FFT the maximal relative error is less than 7.89%, which also illustrates the accuracy of the proposed approach on relevant applications such as the FFT. To test the method on a recursive LTI system, we have considered an 8-order IIR filter divided into four 2-order cells. The relative error is less than 3.3% with an average value of 1.68%.

2) **Non-LTI and Non-Recursive systems:** For the case of non-LTI systems we have tested three applications: a 2\textsuperscript{nd}-order polynomial evaluation, a 2\textsuperscript{nd}-order Volterra filter and a correlator. The results are presented in Table VII. For the 2\textsuperscript{nd}-order polynomial, the estimation is highly accurate, since the error is lower than 0.8%. For the Volterra filter and the correlator, the average relative error is equal to 1.79% and 1.35% with the maximum values of 3.22% and 5.78% respectively. For these three applications, the error is less than 5.78% , thus confirming the accuracy of the proposed approach.

In summary, for all types of systems based on arithmetic operations, the presented results tend to indicate that the method has an overall relative error that is less than 25% even in the worst cases, which can be translated to less than 1 bit of precision.

### TABLE VI

<table>
<thead>
<tr>
<th>Application</th>
<th>Average relative error</th>
<th>Maximum relative error</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIR32(R)</td>
<td>2.49%</td>
<td>7.9%</td>
</tr>
<tr>
<td>FIR32(T)</td>
<td>1.14%</td>
<td>2.41%</td>
</tr>
<tr>
<td>Polyphase filter</td>
<td>5.24%</td>
<td>20.57%</td>
</tr>
<tr>
<td>DCT</td>
<td>8.42%</td>
<td>24.8%</td>
</tr>
<tr>
<td>FFT128</td>
<td>3.68%</td>
<td>7.32%</td>
</tr>
<tr>
<td>FFT256</td>
<td>4.53%</td>
<td>7.89%</td>
</tr>
<tr>
<td>IIR8</td>
<td>1.68%</td>
<td>3.3%</td>
</tr>
</tbody>
</table>

**TABLE VI**

<table>
<thead>
<tr>
<th>Application</th>
<th>Average relative error</th>
<th>Maximum relative error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Polynomial eval.</td>
<td>0.65%</td>
<td>0.8%</td>
</tr>
<tr>
<td>Volterra filter</td>
<td>1.79%</td>
<td>3.22%</td>
</tr>
<tr>
<td>Correlator</td>
<td>1.35%</td>
<td>5.78%</td>
</tr>
</tbody>
</table>

**TABLE VII**

**Relative error for non-LTI and non-recursive systems**

The optimization times are given in Figure 6 for the 32-tap LMS algorithm. A computer with Intel Core 2 Duo T8300 at 2.4 GHz and 2GB of RAM was used for the experiments. For the approach based on fixed-point simulation, a new simulation is required as long as a fixed-point format is modified. So, the optimization time depends linearly on the number of optimization iterations. For approaches based on fixed-point simulations, experiments were conducted on Matlab to evaluate their computing time. For the proposed approach, the analytical expression of $P_e$ needs to be computed first. The determination of this analytical expression is the most time-consuming part and its computation takes 46 seconds for the approach based on infinite sum approximation and 4 seconds for the linear prediction approach. Then, each iteration of this optimization process corresponds only to the noise power expression evaluation, whose computing time is negligible. For the LMS algorithm, the proposed approach leads to a gain after only 100 iterations, which represents an execution time equal to 46 seconds. As an example, for an optimization process with about 30 variables, between $10^3$ and $10^5$ iterations is required. In that case the proposed approach would require between 10 and $10^3$ less time to perform the optimization process. The approach based on linear prediction leads to time gain after only 10 iterations compared to approaches based on fixed-point simulations, thus leading to an execution time equal to 4 seconds. These results clearly show that the proposed approach is faster than the best simulation-based techniques for the process of fixed-point optimization.

A comparison with methods from [4] and [31] has also been carried-out. As they are also analytical, only the time to determine the coefficients $K_i$ and $L_{ij}$ is given. For the approach developed in [31], no result on the quality and the execution time were given. However, for an application with $N_e$ noise sources, $N_e^2$ fixed-point simulations are required. Thus, the execution time grows quadratically with the complexity of the application. For the 32-tap LMS algorithm presented above, 66 noise sources are considered and thus the approaches proposed in [31] would require 4356 fixed-point simulations. Therefore, the proposed method leads to significantly lower optimization time, especially when the application becomes complex.

In the approach based on affine arithmetic simulations proposed in [4], the relative error $E_r$ for the estimation and the time $T_{exe}$ required to determine the coefficients $K_i$ and $L_{ij}$ are given in Table VIII for a second-order IIR filter (IIR2), an 8-order IDCT, and for an LMS filter with 2 and 5 taps. For the case of the IIR2 application, the time to execute the third step corresponding to the determination of the noise power expression (computation of $K_i$ and $L_{ij}$) corresponds to 70 % of the global execution time for the approach based on infinite sums and to 50 % for the approach based on
linear-prediction. The time to execute the second step corresponding to the determination of system expression (computation of \( A_i \) and \( D_i \)) represents to 26% of the global execution time for the approach based on infinite sums. For this approach, the rest of the time (4%) is dedicated to the other steps (front-end, determination of system noise model and expression generation). For the proposed method based on infinite sums with \( p = 1000 \), the quality of the estimation is close to the one obtained with the technique of [4]. For our linear prediction based approach, the quality is slightly lower, but the execution time is reduced. For our two approaches, the execution time is significantly reduced compared to the approach proposed in [4] and a reduction of several order of magnitude is observed.

The results show the benefit of the proposed approaches to reduce the development time of fixed-point systems. We propose two different approaches with different characteristics, as shown in Table VIII. It is seen that the infinite sums approach is really accurate and fast, whereas, the approach based on linear prediction is faster but a little less accurate. So, the proposed framework allows the user to choose between two different approaches with different trade-offs between speed and accuracy.

<table>
<thead>
<tr>
<th>Benchmarks</th>
<th>Infinite sums</th>
<th>Linear prediction</th>
<th>Approach from [4]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( T_{exe} )</td>
<td>( E_r )</td>
<td>( T_{exe} )</td>
</tr>
<tr>
<td>IIR 2</td>
<td>0.15</td>
<td>1.68%</td>
<td>0.08</td>
</tr>
<tr>
<td>IDCT 8</td>
<td>0.04</td>
<td>0.62%</td>
<td>0.04</td>
</tr>
<tr>
<td>LMS 2</td>
<td>0.0587</td>
<td>1.09%</td>
<td>0.0485</td>
</tr>
<tr>
<td>LMS 5</td>
<td>0.2142</td>
<td>2.87%</td>
<td>0.1256</td>
</tr>
</tbody>
</table>

**TABLE VIII**

Comparison of the proposed approaches with the one proposed in [4] in terms of execution time \( T_{exe} \) (seconds) and relative error \( E_r \) for the estimation.

In this section, infinite sums are modelled for impulse response terms \( h(k) \). The vector \( \omega_m \) is defined as

\[
\omega_m = \begin{pmatrix}
  h(m) \\
  h(m+1) \\
  \vdots \\
  h(m+P-1)
\end{pmatrix},
\]

The first \( P + Q - 1 \) terms of impulse response are directly determined. Thus, terms \( h(0) \) to \( h(P+Q-2) \) are computed with (22). The term \( P + Q - 1 \) is present in the matrix \( \Omega_Q \) and in the following \( P - 1 \) matrices. This term can also be computed with prediction coefficient \( \lambda_i \). The \( P \)-size matrix \( \Lambda \) is defined by

\[
\Lambda = \begin{pmatrix}
  0 & 1 & 0 & \ldots & 0 \\
  0 & 0 & 1 & \ldots & 0 \\
  \vdots & \vdots & \vdots & \ddots & \vdots \\
  0 & \ldots & 0 & 1 & \lambda_P \\
  \lambda_P & \lambda_{P-1} & \ldots & \ldots & \lambda_1
\end{pmatrix}
\]

and the following recurrence relation is obtained:

\[
\omega_{m+1} = \Lambda \omega_m = \Lambda^{m+2-Q} \omega_{Q-1}.
\]

**Theorem 2:** The matrix \( \Lambda^n \) has a limit equal to 0 when \( n \to \infty \).

**Proof:**

The characteristic polynomial \( \psi_p(X) = \det(XI_p - \Lambda) \) of the matrix \( \Lambda \) is equal to

\[
\psi_p(X) = \begin{vmatrix}
  X & -1 & 0 & \ldots & 0 \\
  0 & X & -1 & \ldots & 0 \\
  \vdots & \vdots & \ddots & \ddots & \vdots \\
  0 & \ldots & \ldots & X & -1 \\
  -\lambda_P & -\lambda_{P-1} & \ldots & \ldots & X - \lambda_1
\end{vmatrix}
\]
To compute the determinant, we develop according to the first column, which leads to the recurrence expression

$$\psi_P(X) = X\psi_{P-1}(X) - \lambda_P(-1^{P-1})(-1^{P-1})$$

$$= X\psi_{P-1}(X) - \lambda_P$$

As for $P = 1$, $\Lambda = \lambda_1$, $\psi_1(X) = X - \lambda_1$ and by using the previous recursion, $\psi_P(X)$ is equal to

$$\psi_P(X) = X^P - \sum_{k=1}^{P} \lambda_k X^{P-k}. \quad (82)$$

The roots of the previous equation are the eigenvalues of $\Lambda$. Moreover, these prediction coefficients define a transfer function. Indeed, they linearize the time-varying transfer function. The poles $\rho_m$ of the transfer function satisfy the equality

$$1 - \sum_{k=1}^{P} \rho_k = \prod_{n=1}^{P} (1 - \rho_m). \quad (83)$$

Thus, comparing solutions of (82) with (83), eigenvalues of $\Lambda$ are the poles of the transfer function, and $\Lambda$ can be written as

$$\Lambda = T \begin{pmatrix} \rho_1 & 0 & 0 & \cdots \\ 0 & \rho_2 & 0 & \cdots \\ & \ddots & \ddots & \cdots \\ 0 & \cdots & \rho_{P-1} & 0 \\ 0 & \cdots & \cdots & \rho_P \end{pmatrix} T^{-1}. \quad (84)$$

where $T$ is an invertible matrix and $\rho_m$ can be real or complex. However, since the system is stable, the modulus of $\rho_m$ is less than one so $\rho_m^n$ converges to 0 when $n \to \infty$. So, $\Lambda^n$ is equal to

$$\Lambda^n = T \begin{pmatrix} \rho_1^n & 0 & 0 & \cdots \\ 0 & \rho_2^n & 0 & \cdots \\ & \ddots & \ddots & \cdots \\ 0 & \cdots & \rho_{P-1}^n & 0 \\ 0 & \cdots & \cdots & \rho_P^n \end{pmatrix} T^{-1}. \quad (85)$$

(85) proves that $\Lambda^n$ tends to zero when $n \to \infty$.

Therefore:

$$\sum_{m=0}^{\infty} \omega_m = \sum_{m=0}^{Q-2} \omega_m + \sum_{k=0}^{Q-1} \Lambda^k \omega_{Q-1} = \sum_{m=0}^{Q-2} \omega_m + (\text{I}_P - \Lambda)^{-1} \omega_{Q-1} \quad \text{with I}_P \text{ the P-size identity matrix.} \quad (86)$$

The vector $\sum_{m=0}^{\infty} \omega_m$ is also equal to

$$\sum_{m=0}^{\infty} \omega_m = \begin{pmatrix} \sum_{m=0}^{\infty} h(m) \\ \sum_{m=1}^{\infty} h(m) \\ \vdots \\ \sum_{m=P-1}^{\infty} h(m) \end{pmatrix}. \quad (87)$$

So, the vector $\theta$ is introduced where each element $\theta_m$ is defined by

$$\theta_m = \sum_{k=0}^{m-2} h(k) \quad (88)$$

$\theta$ contains the $P - 1$ first impulse response terms which are computed manually. Thus, each line of $\sum_{m=0}^{\infty} \omega_m + \text{boldsymbol}{\theta}$ is equal to $\sum_{m=0}^{\infty} h(m)$. The sum of impulse response terms is then equal to

$$\sum_{m=0}^{\infty} h(m) = \frac{1}{P} \sum_{j=1}^{P} \left( \sum_{m=0}^{Q-2} \omega_m + (\text{I}_P - \Lambda)^{-1} \omega_{Q-1} + \theta \right)_{(j)}. \quad (89)$$

This expression is developed for impulse terms $h$. In practice, the impulse terms are defined by two matrices $A$ and $D$ as shown in (Section III). The previous method is then applied to $A1D$ where $1$ is a matrix containing ones (and replacing the noise $B$).

**APPENDIX B**

In this appendix, the method modelling the second order infinite sums is presented. Using the same approach as in Appendix A, the following recurrence relation is obtained:

$$\omega_{m+1} \omega_{m+1}^T = \Lambda \omega_m \omega_m^T A^T = \Lambda^{m+2} Q \omega_{Q-1} \omega_{Q-1}^T A^{m+2} \quad (90)$$

Matrix $\Lambda^n$ has a limit equal to zero when $m \to \infty$ as demonstrated previously, which gives

$$\sum_{m=0}^{\infty} \omega_m \omega_m^T = \sum_{m=0}^{Q-2} \omega_m \omega_m^T + \sum_{k=0}^{\infty} \Lambda^k \omega_{Q-1} \omega_{Q-1}^T A^k. \quad (91)$$

Yet, the next equality can be determined:

$$S - \Lambda S A^T = \omega_{Q-1} \omega_{Q-1}^T. \quad (92)$$

This equality is a Lyapunov equation. So, $S$ is the solution of the Lyapunov equation and can be obtained using a classical solver. As in Appendix A, we introduce the diagonal matrix $\Phi$ where each diagonal element is equal to

$$\Phi_{r,r} = \sum_{k=0}^{r-2} h(k)^2 \quad (93)$$

Each element of the diagonal of $\sum_{m=0}^{\infty} \omega_m \omega_m^T + \Phi$ is equal to the sum of square impulse terms $\sum_{m=0}^{\infty} h(m)^2$. So, it leads to

$$\sum_{m=0}^{\infty} h(m)^2 = \frac{1}{P} \text{Tr} \left( \sum_{m=0}^{Q-2} \omega_m \omega_m^T + S + \Phi \right). \quad (94)$$