fractal and fractional

MDPI

Article

Multi-Signal Multifractal Detrended Fluctuation Analysis for Uncertain Systems —Application to the Energy Consumption of Software Programs in Microcontrollers

Juan Carlos de la Torre [1], Pablo Pavón-Domínguez [2,*], Bernabé Dorronsoro [1,3], Pedro L. Galindo [1] and Patricia Ruiz [2,3]

[1] Department of Computer Science Engineering, University of Cadiz, 11003 Cadiz, Spain; juan.detorre@uca.es (J.C.d.l.T.); bernabe.dorronsoro@uca.es (B.D.); pedro.galindo@uca.es (P.L.G.)
[2] Department of Mechanical Engineering and Industrial Design, University of Cadiz, 11003 Cadiz, Spain; patricia.ruiz@uca.es
[3] Faculty of Engineering, The University of Sydney, Camperdown 2050, Australia
* Correspondence: pablo.pavon@uca.es

Abstract: Uncertain systems are those wherein some variability is observed, meaning that different observations of the system will produce different measurements. Studying such systems demands the use of statistical methods over multiple measurements, which allows overcoming the uncertainty, based on the premise that a single measurement is not representative of the system's behavior. In such cases, the current multifractal detrended fluctuation analysis (MFDFA) method cannot offer confident conclusions. This work presents multi-signal MFDFA (MS-MFDFA), a novel methodology for accurately characterizing uncertain systems using the MFDFA algorithm, which enables overcoming the uncertainty of the system by simultaneously considering a large set of signals. As a case study, we consider the problem of characterizing software (Sw) consumption. The difficulty of the problem mainly comes from the complexity of the interactions between Sw and hardware (Hw), as well as from the high uncertainty level of the consumption measurements, which are affected by concurrent Sw services, the Hw, and external factors such as ambient temperature. We apply MS-MFDFA to generate a signature of the Sw consumption profile, regardless of the execution time, the consumption levels, and uncertainty. Multiple consumption signals (or time series) are built from different Sw runs, obtaining a high frequency sampling of the instant input current for each of them while running the Sw. A benchmark of eight Sw programs for analysis is also proposed. Moreover, a fully functional application to automatically perform MS-MFDFA analysis has been made freely available. The results showed that the proposed methodology is a suitable approximation for the multifractal analysis of a large number of time series obtained from uncertain systems. Moreover, analysis of the multifractal properties showed that this approach was able to differentiate between the eight Sw programs studied, showing differences in the temporal scaling range where multifractal behavior is found.

Keywords: multifractal analysis; uncertainty; energy consumption; software characterization; green computing

check for updates

Citation: de la Torre, J.C.; Pavón-Domínguez, P.; Dorronsoro, B.; Galindo, P.L.; Ruiz, P. Multi-Signal Multifractal Detrended Fluctuation Analysis for Uncertain Systems—Application to the Energy Consumption of Software Programs in Microcontrollers. *Fractal Fract.* **2023**, *7*, 794. https://doi.org/10.3390/fractalfract7110794

Academic Editor: Lyudmyla Kirichenko

Received: 15 September 2023
Revised: 19 October 2023
Accepted: 23 October 2023
Published: 30 October 2023

1. Introduction

Fractal and multifractal analyses have been extensively applied in many different fields. Specifically, multifractal detrended fluctuation analysis (MFDFA) [1] is a powerful method for detecting long-range correlations and multifractal scaling properties in time series. It has been widely analyzed [2,3] and employed with success in different fields, such as financial markets [4], meteorology [5], traffic speed [6], biology [7], acoustics [8], neuroimaging [9,10], and seismicity [11,12], among others. However, the application of MFDFA for studying uncertain systems is not straightforward. The reason for this is that an

uncertain system produces different measurements for each observation, and therefore one single system measurement is not representative of its behavior. Therefore, the application of MFDFA on a single time series obtained from an uncertain system cannot provide meaningful results and, unlike what is commonly done in the literature (e.g., [13–16]), a large number of signals must be considered in the analysis.

Manually applying MFDFA to such a large number of time series is a cumbersome task, and extracting overall conclusions from all the individual multifractal analyses performed can be highly complicated. Consequently, there is a clear need for a methodology that allows simultaneously employing the MFDFA by considering multiple time series, to offer a comprehensible overall characterization of the system. Such a methodology would be highly suitable for characterizing uncertain systems.

In the framework of MFDFA, there are already different methods for jointly analyzing multiple time series. For example, ref. [17] proposed multifractal detrended cross-correlation analysis (MFDXA) for characterizing the long-range cross-correlations of two time series coexisting simultaneously. Additionally, ref. [18] developed multivariate multifractal detrended fluctuation analysis (MV-MFDFA), a method that directly studies the fractal dynamics of multichannel data in a complex system. Subsequently, ref. [19] extended the previous algorithm to the multiscale case through multivariate multifractal detrended fluctuation analysis (MMV-MFDFA), which is based on a moving fitting window that allows defining generalized dependent Hurst surfaces. An application of the above method can be found in [20]. However, the aforementioned methods are multivariate, due to their application to several variables measured at the same time. In this work, we propose a simple methodology called multi-signal multifractal detrended fluctuation analysis (or MS-MFDFA for short), which applies to univariate scenarios, i.e., a single variable is measured repeatedly at different moments in time. Therefore, multivariate MFDFA methods act on different, deterministic and dependent signals composed of multiple components, while MS-MFDFA acts on independent signals with uncertainty composed of a single component. Moreover, due to the high number of signals to be processed, this process is automated.

As a case study, MS-MFDFA was applied in this work to characterize the consumption of software (Sw) programs. This is the first time that an MFDFA analysis has been performed on this kind of signal, to the best of our knowledge. The main idea behind the green computing (GC) paradigm is reducing the energy consumption of Sw executions, while maintaining the same functionalities and performance [21]. The aim is to reduce the electricity demand of a device with computing capabilities, or alternatively to enlarge the autonomy of battery-based ones. Such computing devices are everywhere around us; from large servers in the Cloud, to the small equipment typically found in the Internet of things or sensor networks, which are widespread technologies in Industry 4.0.

To date, there have been significant achievements in the design of sustainable hardware (Hw), and most electronic devices include different technologies to reduce their consumption. In contrast, very few advances have been made in terms of green Sw, despite the fact that Sw drives the Hw and it ultimately determines its behavior [22]. Therefore, the development of novel techniques towards the generation of green Sw is of utmost importance, so that they can make efficient use of the underlying Hw [23].

In order to achieve energy-efficient Sw programs, accurate measurements of their consumption performance are necessary [24]. However, this is not a trivial task, given that there are high uncertainty levels [25], caused by other coexisting processes (e.g., those of the operating system), eventualities of the Sw execution itself (cache misses, memory allocation, or bottlenecks, among others), or the ambient temperature. Additionally, the energy efficiency of Sw strongly depends on the underlying Hw, meaning that Sw performs differently depending on the Hw platform used; the same Sw might produce low consumption levels on some Hw and high levels on another. This fact makes it difficult to define what green Sw is.

The reasons mentioned above show that the approach followed in the literature of just measuring the consumption when executing the Sw is definitely misleading. We argue that

there is a need to analyze the consumption profile of the Sw during the whole run, rather than the aggregated consumption value itself. In this work, we therefore propose analyzing the time series of the consumption of the Sw during the whole execution, with the aim of finding a consumption evolution profile that can help in characterizing green Sw.

The execution of a Sw program is preceded by a compilation process that creates the sequence of low level instructions to be executed on the Hw. This sequence is specific for every Sw program and directly depends on the Hw itself, the compiler, its version, and the flags used in the compilation process. Additionally, its performance can be highly influenced by other concurrent processes. These reasons, together with the fact that current processors typically execute several hundreds of thousands of millions of instructions per second, make it extremely difficult to determine the actual behavior of the system or the exact operations being executed at a specific time stamp.

This work proposes the characterization of Sw consumption by means of the MS-MFDFA approach, generating a signature of the Sw/Hw consumption profile, regardless of the execution time, the consumption values, and uncertainty. The consumption signal (time series) is a high-frequency sampling of the instant input current of the device when executing the Sw. The uncertainty in the consumption measurements of a Sw program [25,26] mean that a single experiment is not representative of the Sw performance, and multiple experiments must be considered to achieve meaningful conclusions. The proposed technique was applied here to analyze the performance of eight simple programs on the selected architecture.

Although fractal and multifractal analyses have been extensively applied in many different fields, to the best of our knowledge, there is a gap in the literature regarding the application of multifractal analysis for studying uncertain systems. The main reason behind this might be the need of analyzing and interpreting the results of a large number of signals, and the difficulty of doing this by hand. Although several implementations of the MFDFA algorithm are available in the literature [27,28], none of them tackle the automatic analysis of a batch of time series.

Moreover, the characterization of the multifractal (MF) properties of energy consumption has rarely been studied. Next, the most relevant works are briefly explained. In [29], a fractal analysis of the energy consumption datasets of a real test-bed assembly line was carried out. The Hurst exponent was computed for each of the datasets that corresponded to different assembly operations for several manufacturing cycles. The results showed an energy consumption pattern matching each operation. In addition, an MF study of the daily electric load time series of a power system was carried out in [30], concluding that the fluctuations in this time series exhibited a MF nature with a long-range correlation behavior. MF analysis has also been applied to study wind power consumption capacity [31]. The obtained MF parameters were used to describe the fluctuation characteristics of wind power and its correlation with consumption capacity.

Fluctuations in the electricity market and prices have also been tackled, where the MF analysis focused more on the time series of energy prices, rather than on the consumption fluctuations themselves. To mention a few, four electricity regions in Unites States were studied using MFDFA, using large daily data from 2001 to 2021 [32]. The results related the highest multifractality to the lowest efficiency region and vice versa. Moreover, MFDFA was used in [33] to numerically investigate the correlation, persistence, multifractal properties, and scaling behavior of time series data (the hourly spot prices) from the Spanish electricity market OMEL.

In terms of energy efficiency, there have been a number of works addressing Sw consumption, in most cases as a component of the development of sustainable Sw [34]. These works propose certain criteria that must be taken into account for the design of sustainable Sw from a life-cycle perspective [35]. Most works in this line proposed different criteria to enhance the sustainability of both the development of Sw products and the Sw design, taking into account the Hw used (storage, computing, network, use of remote/local resources, etc.), or required Sw services (databases, world wide web, web services, etc.) [36,37].

In contrast, in this work, we were interested in studying the energy-efficiency of the Sw execution and not of its development process. We believe this is worthy of study because of its major impact, given that the same Sw is, in many cases, executed in millions of computing devices 24 h a day. In this sense, Mancebo et al. [24] emphasized the difficulty of measuring Sw consumption and performed some experiments to measure the consumption of writing different tweets and Facebook posts. Their article proposed a more generic method for providing a characterization of the Sw consumption profile, analyzing how the available resources are used and not only what kind of resources or input data are considered, as previously done in the literature. Additionally, our methodology takes into account the high uncertainty levels present in the energy consumption of computing devices, unlike most existing works [25].

The contributions of this work are as follows: (i) the proposal of MS-MFDFA, a new simple methodology for applying MFDFA to a large number of signals, in order to mitigate the effects of uncertainty and obtain confident results; (ii) the development of a fully functional application to automatically accomplish the multifractal detrended fluctuation analysis for a large number of signals, which is freely available; (iii) the application, for the first time, of multifractal analysis to studying the performance of Sw programs; (iv) the characterization of the energy consumption induced by different Sw programs on a specific Hw architecture by means of MFDFA. Moreover, we propose a new Sw benchmark composed of eight simple programs to analyze the consumption of low-power devices.

The remainder of this work is structured as follows: The next Section reviews the most relevant works of the state of the art. Section 2 presents the theoretical framework used, as well as the new methodology proposed. The new benchmark of Sw programs and the experimental settings are explained in Section 3. The results obtained are analyzed in Section 4, and they are used to validate the proposed MS-MFDFA methodology in Section 5. A brief description of the fully functional MS-MFDFA software is presented in Section 6. Finally, Section 7 concludes the work and proposes some future lines of research.

2. Multi-Signal Multifractal Detrended Fluctuation Analysis, MS-MFDFA

This section describes the proposed multi-signal multifractal detrended fluctuation analysis (MS-MFDFA hereinafter), used in this work for assessing the temporal scaling in energy consumption signals. For that, a description of MFDFA is first summarized in Section 2.1, with the proposed methodology in Section 2.2.

2.1. Multifractal Detrended Fluctuation Analysis

MFDFA [1] is a generalization of detrended fluctuation analysis (DFA) [38], a technique based on the detection of long-range correlations in time series. MFDFA allows a reliable multifractal characterization of time series by adding a q-dependent averaging procedure as an additional step to the DFA.

Initially, consider that x_k with $k = 1, 2, ..., N$ is a time series of a compact support, i.e., zero values represent an insignificant fraction of k (or simply zero values do not exist). The MFDFA is comprised of five steps, which are described as follows [1]:

- Step 1: Obtain the integrated time series $X(i)$ of x_k by means of subtracting the mean $\overline{x}$, to the accumulative sum of the time series. In this context, $X(i)$ is the so-called "profile":

$$X(i) = \sum_{k=1}^{i} [x_k - \overline{x}], \qquad i = 1, ..., N. \tag{1}$$

  Notice that subtraction of the mean $\overline{x}$ is optional, because it will be removed with the subsequent detrending in Step 3.
- Step 2: Divide the integrated time series $X(i)$ into $N_s = int(N/s)$ non-overlapping segments of identical length s. The set of s values correspond to the temporal scaling for the analysis. Since the length N of the time series is not usually a multiple of s, it is unavoidable that a short part at the end of the signal may be left out of the analysis.

To entirely consider the signal, a identical dividing procedure is carried out from the opposite end of the integrated time series. Thus, $2N_s$ divisions are obtained for each s value.

- Step 3: Compute the local trend for each of the $2N_s$ segments. Then, detrending in $X(i)$ is carried out by subtracting $x_v(i)$ from $X(i)$ in each segment v; $x_v(i)$ being the polynomial fitting through the least squares method in segment v. The variance of the residual, F^2, is calculated for each segment of length s, as follows:

$$F^2(s,v) = \frac{1}{s}\sum_{i=1}^{s}\{X[(v-1)s+i] - x_v(i)\}^2 \tag{2}$$

for each segment v, $v = 1, ..., N_s$ and

$$F^2(s,v) = \frac{1}{s}\sum_{i=1}^{s}\{X[N-(v-N_s)s+i] - x_v(i)\}^2 \tag{3}$$

for $v = N_s + 1, ..., 2N_s$. Since trends are removed according to the order of the polynomial fit m, the method is called MFDFA$_m$. In this work, MFDFA$_1$ was employed.

- Step 4: Assemble the local variances in the qth order fluctuation function $F_q(s)$ by averaging over all fluctuations for each time scale s. In this step, multifractal properties are assessed by adding to $F^2(s,v)$ different qth order moments, according to

$$F_q(s) = \left\{\frac{1}{2N_s}\sum_{v=1}^{2N_s}[F^2(s,v)]^{\frac{q}{2}}\right\}^{\frac{1}{q}} \tag{4}$$

As observed, q exponents magnify the local variances. Whereas positive q values emphasize large variances (large deviations from the polynomial fit), negative q values highlight small variances (small deviations from the corresponding fit). For the specific $q = 0$ case, the exponent in Equation (4) diverges and may be estimated by means of a logarithmic averaging procedure, resulting in

$$F_{q=0}(s) = \exp\left\{\frac{1}{4N_s}\sum_{v=1}^{2N_s}\ln F^2(s,v)\right\} \tag{5}$$

Steps 2–4 are then repeated for an sufficiently large set of values of s.

MFDFA focuses on how the generalized q-dependent fluctuation functions depend on the time scales for different values of q. For (multi)-fractal time series, it is expected that $Fq(s)$ increases for larger time scales s according to a power law. When this is trusted, the double logarithmic representation of the fluctuation functions $F_q(s)$ versus the scaling s exhibits a linear relationship in one or several scaling ranges;

- Step 5: Determine the (multi)-fractal properties of the fluctuation functions according to

$$F_q(s) \sim s^{h(q)} \tag{6}$$

where $h(q)$ is estimated from the slope of the linear regression between $ln\ F_q(s)$ and $ln\ s$ for a certain range of scales. Therefore, $h(q)$, which is known as the generalized Hurst exponent, represents the scaling behavior of the variance $F^2(s,v)$. When positive (negative) q moments are considered, $h(q)$ is governed by the segments with large (small) fluctuations, respectively. In the case that segments with large and small fluctuations scale similarly, $h(s)$ is independent of q, which is known as a monofractal case. This means that the scaling behavior of the variances $F^2(s,v)$ is identical for all segments v, and thus the averaging procedure in Equation (4) provides identical scaling behavior for all values of q. By contrast, a time series is multifractal when segments with large and small fluctuations scale differently. Multifractality is clearly evidenced when there is a significant dependence of $h(q)$ on q. The greater the

differences in large and small scaling fluctuations, the greater the multifractal strength, which is proportional to the width of the $h(q)$ function [$\Delta h(q)$].
On the other hand, the value of $h(q = 2)$ provides useful information about the time series. Whereas for stationary signals $h(q = 2)$ is identical to the Hurst exponent H [39], for non-stationary signals, the Hurst exponent is defined as $H = h(q = 2) - 1$ [40,41]. Additionally, $h(q = 2) > 0.5$ indicates the long memory or persistency in the signal and $h(q = 2) < 0.5$ short memory or anti-persistency. When $h(q = 2) = 0.5$, the signal is uncorrelated.

2.2. Multi-Signal Multifractal Detrended Fluctuation Analysis, MS-MFDFA

As has been previously discussed, the results of applying MFDFA on time series from one single experiment are not representative of the behavior of an uncertain system. Therefore, a novel methodology for simultaneously considering a large number of time series in the MFDFA is needed, to mitigate the effects of uncertainty. However, applying this type of MF analysis on a large number of signals is costly, because it is a manual process, and additionally it would be difficult to draw overall conclusions. Thus, this methodology allows analyzing multiple signals and offers overall conclusions on their performance as one single fluctuation function, which is easy to interpret. To the best of our knowledge, this approach has never been used before.

MS-MFDFA is a novel automatically driven methodology for applying MFDFA over a huge amount of signals that can provide significant results for the characterization of a system under high uncertainty levels. The proposed methodology first requires applying the MFDFA algorithm to a set of time series, obtained from different observations of the system. Once the fluctuation functions $F_q(s)$ of each time series are obtained, a representative $\langle F_q(s) \rangle$ is computed with the average value of all $F_q(s)$ functions (from negative to positive q values). This averaged fluctuation function is called *assembled* and this is the scaling function for the later estimation of the representative generalized Hurst dimension functions $\langle h(q) \rangle$. According to this, Equation (6) from Step 5 of MFDFA is replaced by

$$\langle F_q(s) \rangle \sim s^{\langle h(q) \rangle} \tag{7}$$

Because of the uncertain nature of the signals studied, we consider that averaging the fluctuation functions and obtaining a single generalized Hurst dimension function is more appropriate than calculating all generalized Hurst dimension functions and then averaging them. This second approach would require independently analyzing each of the 100 fluctuation functions and establishing crossover points common to all of them, so that the linear regions could be calculated to obtain the corresponding h(q). In order to validate the suitability of this approach, one hundred different synthetic signals with fractional Gaussian noise (fGn) and a controlled known Hurst exponent (H = 0.7 in this case) were generated using the MFDFA python package (https://mfdfa.readthedocs.io/en/latest/). The assembled function was obtained and the resulting Hurst exponent was the expected one.

Therefore, MS-MFDFA uses the *assembled* fluctuation function $\langle F_q(s) \rangle$ as representative of the 100 signals and determines the best fits for it. This representativeness is analyzed later in Section 5.

A diagram of the MS-MFDFA methodology is shown in Figure 1.

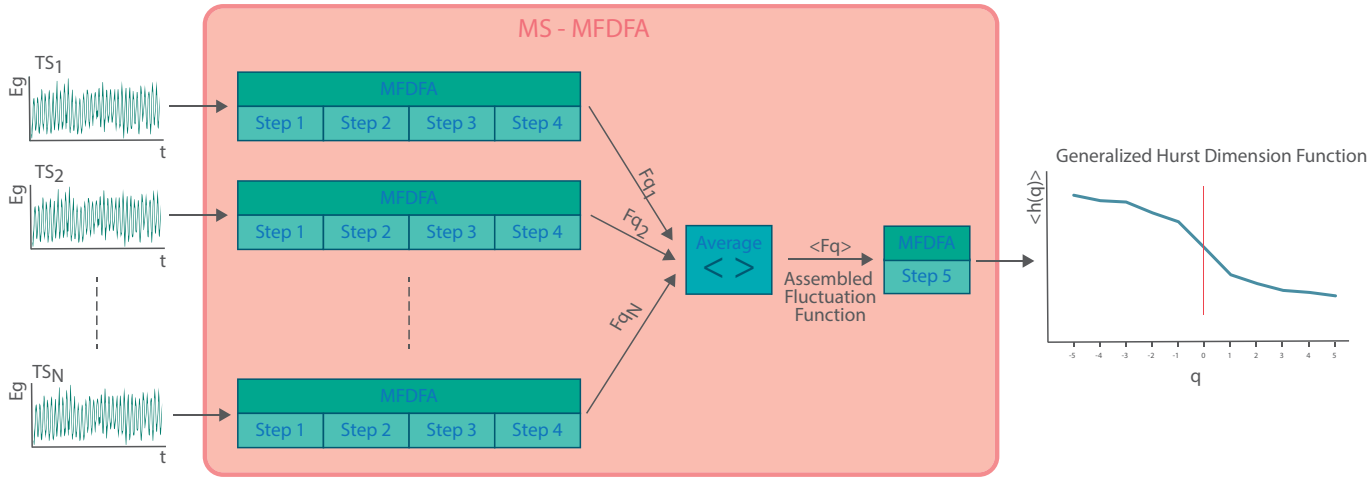

Figure 1. Illustration of the methodology followed by the proposed MS-MFDFA.

3. Experimental Settings

This section first describes the proposed benchmark of software programs employed in this work (Section 3.1). Second, it summarizes how the studied consumption time series were generated (Section 3.2). Finally, the procedure followed for pre-processing the signals is presented in Section 3.3.

3.1. The Proposed Benchmark of Software Programs

In order to explore the energy consumption of Sw programs in a microcontroller using MS-MFDFA, a benchmark constituted by eight simple programs was created. It is composed of six self-developed simple Sw programs, and two other more complex programs taken from the literature, which are introduced next. B1 just puts the device into sleep mode for the whole experiment. Programs B2 to B6 are specifically implemented to evaluate the impact of different types of matrix operations on energy consumption. Their pseudocodes are given in Table 1. Matrices A and B were randomly initialized and matrix C was initialized to zero, before the loop was invoked. Parameter N in all pseudocodes took a value of 80 in the experiments carried out.

As mentioned above, the first software program named IDLE-STATE (or B1) just put the device into sleep mode. Therefore, it would be right to consider the power signal produced by B1 as the baseline power consumption of the hardware. B2 performs a large number of sum operations, all conducted in an unrolled fashion, in order to avoid jumps in the program execution. In these sum operations, the program accessed all positions of the A and B matrices, and only position 0 of C in all cases. B4 is similar to B2, but performs multiplications instead of sums. B3 traverses the three matrices A, B, and C, making sum operations of the numbers they contain. B5 is similar to B3 but using multiplications. B6 is a triple nested loop, as B3 and B5, but with the important difference that the result of the multiplication operation is stored in C.

Additionally, two more sophisticated Sw, named B7 and B8, were taken from the literature. The former is *nettle SHA-256*, a widely used cryptographic algorithm designed by the United States National Security Agency (NSA), which transforms a file into a unique value of fixed length. This value is known as a hash number. The implementation used in this work was developed within the scope of the GNU Nettle project [42] (called *nettle-sha256*). This program is highly demanding for the micro-controller, as it is both CPU-intensive and memory-intensive, being a 256-bit cipher. The latter Sw taken from the literature is *picojpeg*, a scaled-down implementation of a JPEG compressor, optimized for minimal use of memory and resources, so that it is suitable for running on micro-controllers. The degree of compression can be adjusted, so that either the file size or image quality can be prioritized. The lightweight implementation of *picojpeg* used in this work was developed

under the Embench project [43]. Although it is also CPU- and memory-intensive, it differs from B7 in both the time and intensity of its resource usage.

Table 1. Pseudocode of the six simple Sw programs composing the proposed benchmark.

B1 IDLE-STATE	**B2 UNROLL-SUM**
loop: sleep()	loop: A[0] + B[0] + C[0] . . A[512,000] + B[512,000] + C[0]
B3 ROLL-SUM	**B4 UNROLL-PRODUCT**
loop: for i = 0 to N-1 for j = 0 to N-1 for k = 0 to N-1 A[i][k] + B[k][j] + C[i][j] end end end	loop: A[0] ∗ B[0] ∗ C[0] . . . A[512,000] ∗ B[512,000] ∗ C[0]
B5 ROLL-PRODUCT	**B6 SAVE-ROLL-PRODUCT**
loop: for i = 0 to N-1 for j = 0 to N-1 for k = 0 to N-1 A[i][k] ∗ B[k][j] ∗ C[i][j] end end end	loop: for i = 0 to N-1 for j = 0 to N-1 for k = 0 to N-1 C[i][j] <− A[i][k] ∗ B[k][j] end end end

3.2. Experimental Setup

The Hw where Sw is executed directly influences its behavior. In this work, an Adafruit Metro M4 mote was selected as the underlying infrastructure, because it is a commonly used device in the Internet of things. Additionally, the use of a low-computing-capacity device allowed performing low level analyses, involving the use of small programs that execute simple operations, along with others more complex. We believe this is the first step towards the characterization of green Sw.

The acquisition of accurate values for the energy consumption of Sw programs is not a trivial task. In our experiments, a tailored experimental meter designed and developed in the context of GENIUS research project [44] (currently under patent evaluation) was used. This experimental meter is able to synchronize the execution of a predefined Sw with a current sensor, so that the instant input current is sampled at high frequency and accuracy. The sampling rate is set to 1650 samples per second, and the accuracy of the intensity meter goes up to the μA. The length of the time series for the B1 to B6 programs was set to 90,000 samples (energy consumption in mA). This value was established according to the limitation of the storage capacity of the meter. For the B7 and B8 Sw programs, the time series lengths were defined by the program run time: 47,759 and 66,487 samples, respectively. These time series were obtained after around 55 s execution time for B1–B6, 29 s for B7, and 40 s for B8. This amount of samples was considered sufficient for the analyses performed.

The execution of a Sw program in a computing device is considered to be a highly uncertain system, due to a number of issues influencing its performance, such as the presence of coexisting processes, the state of software components (data and instructions cache memories, registers, etc.), and ambient and Hw temperature, among many others. Therefore, due to the nature of the system itself, different values will be measured in different observations. Thus, each benchmark Sw was executed 100 times, in order to

obtain confident results, providing 100 different time series with instant consumption values for every Sw program during its execution.

Figure 2 depicts the instant electric current measurements (in mA) for each of the samples considered for the B1–B8 Sw programs. Specifically, execution 84 of each program is depicted as a representative example of all the time series analyzed. As observed in Figure 2a, the behavior of B1 was completely different from the others. The consumption of the idle-state mode was close to 8 mA with an initial anomalous behavior during the first 20,000 samples. The functionality of B1 was to put the mote into sleep mode, in order to achieve its lowest possible power state. Therefore, the initial anomalous behavior observed during the first 20,000 samples was due to the process required to put the mote into sleep mode. Additionally, at around samples 32,000 and 64,000, the consumption of the mote suddenly reached a peak close to 35 mA for over 0.1 s. This fact suggests the periodic *watchdog* used by the Hw every 32,000 samples, probably to check whether it should leave the sleep mode or not. In Figure 2b–f, it is noted that the signals obtained from B2 to B6 exhibited more stable energy consumption values (37–40 mA) after an initial period of low consumption (justified by the time the mote needed to initialize all its components). In the case of B7 and B8 (see Figure 2g,h), after an initial unstable period, the consumption oscillated around values close to 30 mA on average, which was notably lower than the aforementioned consumption for B1–B6, although their variability in intensity was higher. We envision that the consumption of the B1–B6 programs was probably higher because the switching task caused idle periods in both the processor and memory.

3.3. Signal Pre-Processing

Since the interest of this work lies in the characterization of each of the Sw programs of the benchmark, the anomalies described above must be addressed. For the specific case of B1, both the initial samples and the *watchdogs* interfered in the analysis; thus, only the last 25,000 samples were chosen as representative (see the rectangle with dashed line in Figure 2a). For programs B2–B6, the first 5000 samples were removed, to avoid the initial instability in the signals (see scratched rectangles in Figure 2b–f). Thus, 85,000 samples remained for the analysis of programs B2 to B6. As observed in Figure 2g,h, after avoiding the first 4300 values located in the scratched rectangles, 43,466 and 62,169 samples were kept for programs B7 and B8, respectively. In this way, it was ensured that the signals only contained the real regime of each program.

As a last step, the signals were cropped, with the aim of excluding all possible outliers in the time series. For that, all values exceeding the mean value of the signal by three times its standard deviation were replaced by the mean value of the signal. An insignificant amount of outliers were found, ranging from 0.002% to 0.69% of the number of samples for B7 and B3, respectively. As required by MFDFA, it was ensured that no zero values were found in the time series.

Figure 3 shows the result of applying the explained pre-processing process to the signals displayed in Figure 2. Moreover, some descriptive data of the Sw benchmark are summarized in Table 2, i.e., the average values of the 100 executions, the standard deviation, as well as the max and min values. As seen in Figure 3 and Table 2, B2 and B4 (the unrolled versions of the sum and product, respectively) showed the highest and very similar energy consumption values. The lowest energy consumption corresponded to B1 (around 8.23 mA), as could be expected, followed by B7 and B8. In addition, the standard deviation of the B1 to B6 benchmarks was very low, ranging from 0.043 mA to 0.101 mA. By contrast, for both B7 and B8, it reached 2.855 mA and 2.678 mA, respectively. The maximum and minimum values around the mean, together with the computed deviation values, showed that no outliers remained in the signals.

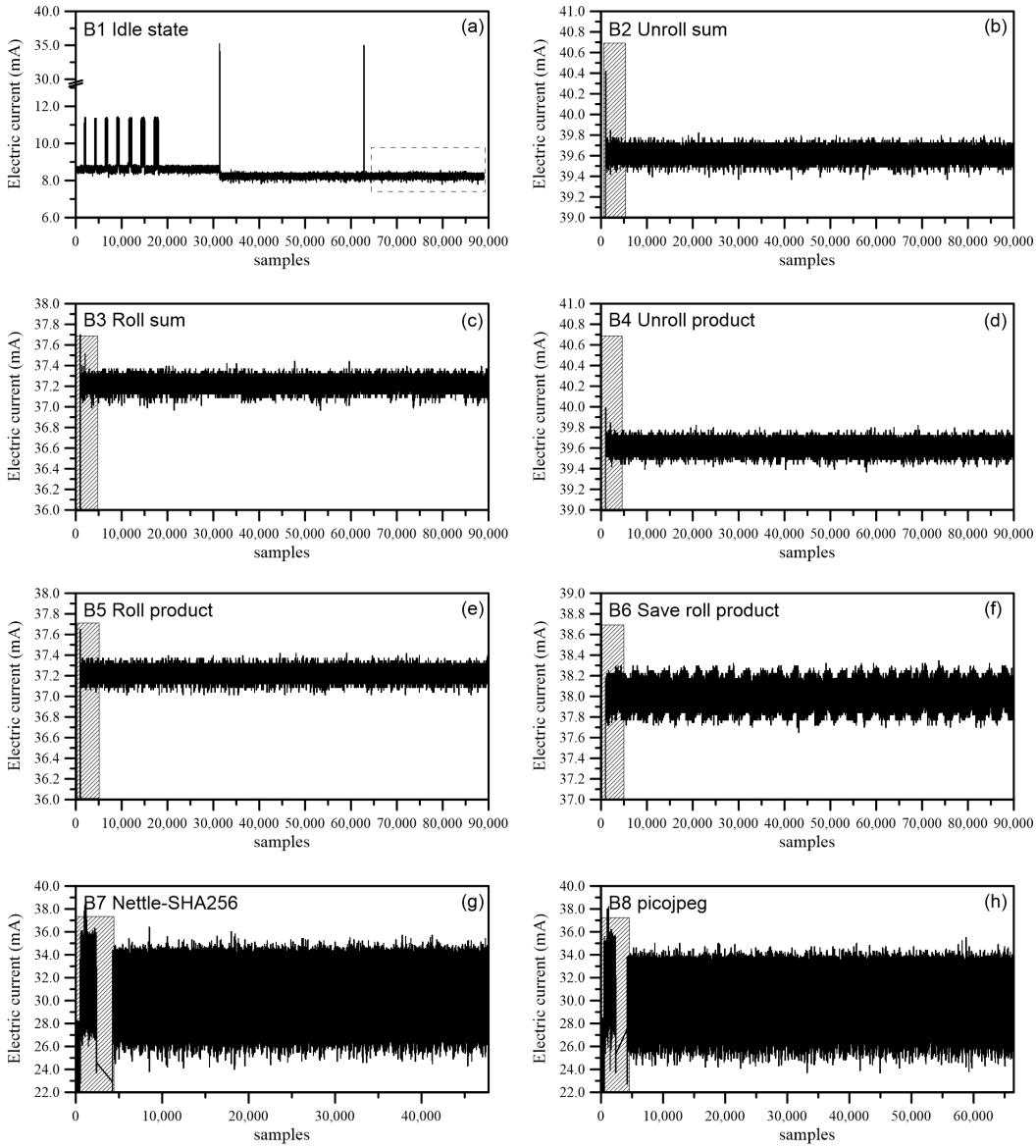

Figure 2. Representation of the obtained energy consumption time series of one sample execution (number 84 in this case) for every considered Sw program of the benchmark presented in Section 3.1, according to (**a**) B1, (**b**) B2, (**c**) B3, (**d**) B4, (**e**) B5, (**f**) B6, (**g**) B7, and (**h**) B8.

Table 2. Average values for the different statistics of the pre-processed signals for the considered Sw programs over 100 executions (mA).

Benchmark	Samples	Mean	std	Max	Min
B1 idle state	25,000	8.230	0.068	8.625	7.722
B2 unroll sum	85,000	39,606	0.046	39.845	39.320
B3 roll sum	85,000	37.229	0.043	37.490	36.917
B4 unroll product	85,000	39.612	0.046	39.845	39.320
B5 roll product	85,000	37.233	0.043	37.466	36.941
B6 save roll product	85,000	38.000	0.101	38.417	37.588
B7 nettle SHA256	43,466	29.818	2.856	37.400	21.262
B8 picojpeg	62,169	29.084	2.678	36.100	21.054

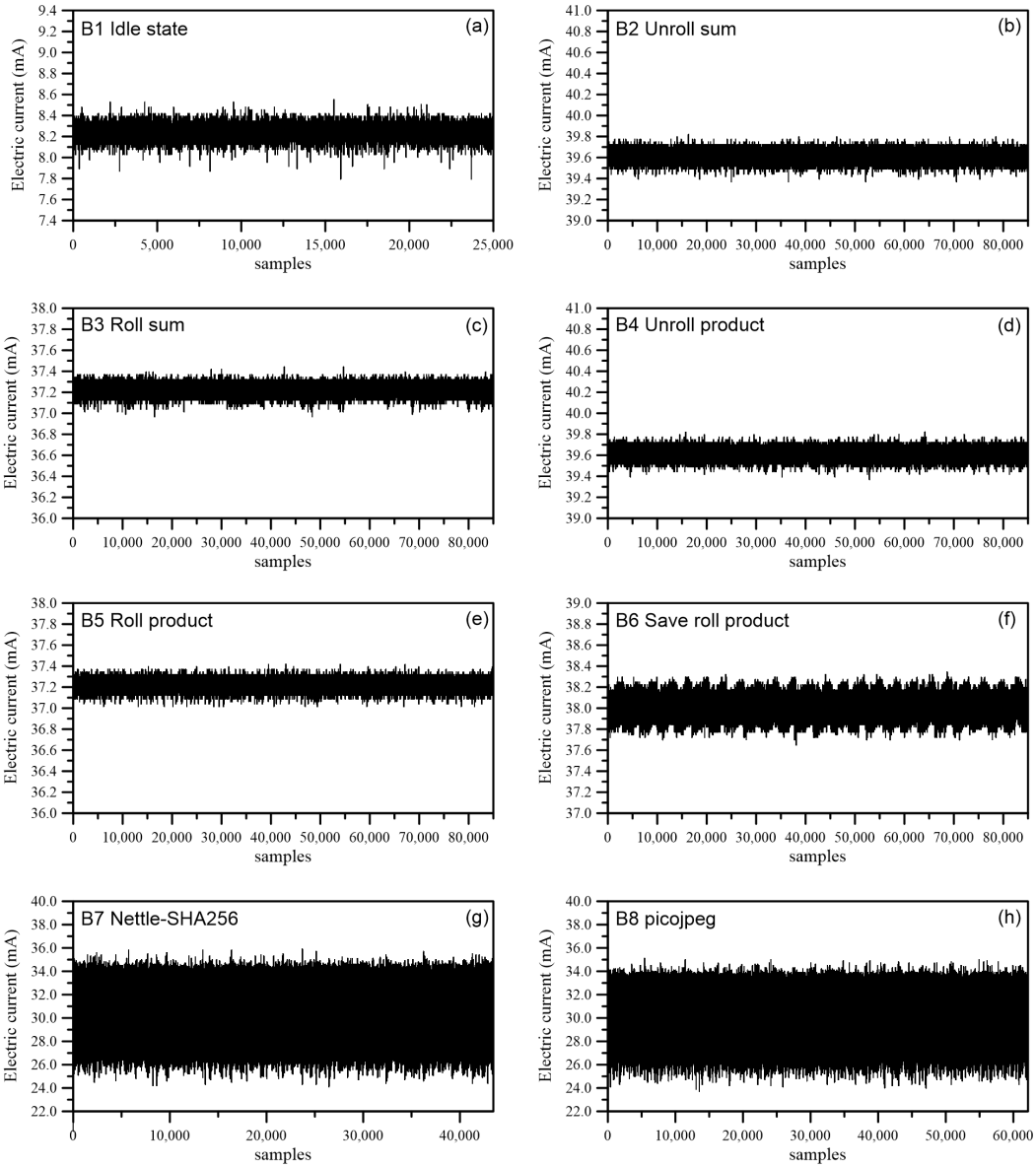

Figure 3. Representation of the pre-processed energy consumption time series of one execution (84 in this case) of each Sw program of the benchmark presented in Section 3.1, according to (**a**) B1, (**b**) B2, (**c**) B3, (**d**) B4, (**e**) B5, (**f**) B6, (**g**) B7, and (**h**) B8.

4. Results

Log–log plots of the fluctuation functions of the energy consumption for the eight Sw programs were calculated from the MFDFA for $q = -5$ to $q = 5$ with increments of $\Delta q = 1$. The *assembled* fluctuation function of each Sw benchmark, computed as the average over the corresponding 100 fluctuation functions, is depicted in Figure 4. As seen, all *assembled* fluctuation functions $\langle F_q(s) \rangle$ increased with higher values of s. In general, the q-curves seem to be linear and parallel to each other, except for B6, which exhibited a divergent area depending on the q values (see Figure 4f).

The treatment of the fluctuation functions was as follows: In the initial region, which corresponds to very small segment s, some anomalous points are observed in Figure 4a–e. In this initial region, the segment sizes are very sensitive to signal noise, so a minimum size s was usually set, from which to perform the subsequent analysis. In this case, we chose a minimum segment value s between 19 and 21, so that all initial anomalies present for negative q moments were excluded from the further analysis. This value is depicted in Figure 4 by a dashed vertical line labeled s_{min}. Likewise, s scales larger than $N/4$ were

also excluded, due to the statistical unreliability derived from the scarce amount of large segments averaging in Step 4 of MFDFA [45]. Thus, an upper limit was established in the fluctuation functions, depicted as a dashed vertical line with the label s_{max}.

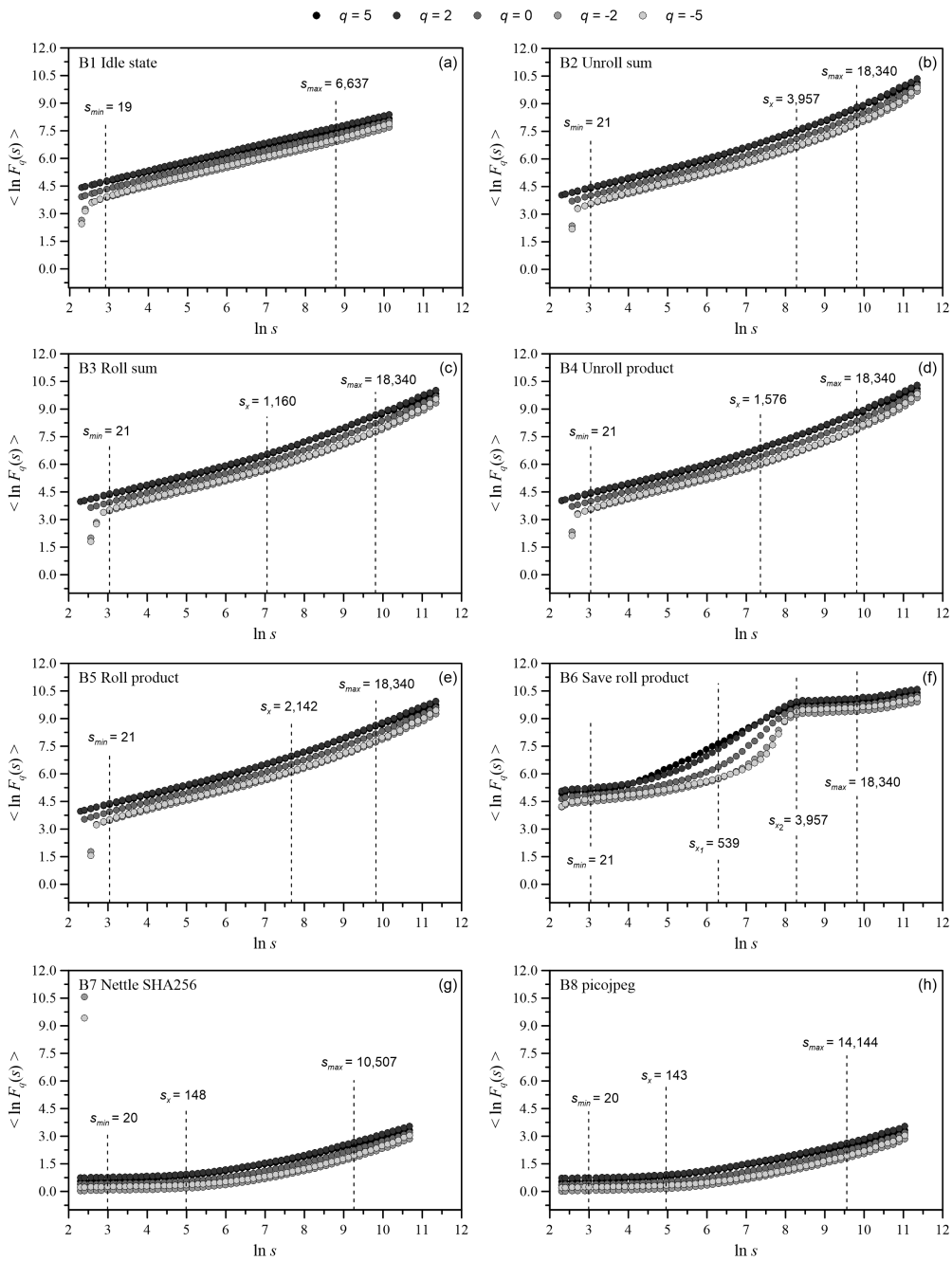

Figure 4. Assembled fluctuation functions for each program, of the benchmark presented in Section 3.1, according to (**a**) B1, (**b**) B2, (**c**) B3, (**d**) B4, (**e**) B5, (**f**) B6, (**g**) B7, and (**h**) B8. It also shows for each program, the lower and upper limits, and their crossover points.

The assessment of the generalized Hurst dimension functions, $\langle h(q) \rangle$, was carried out using least squares linear regression as the slope of the *assembled* fluctuation function. Linear regions are delimited by changes in the slope of the fluctuation function, which are known as crossovers in the MFDFA literature. Crossovers separate scaling regions with different multifractal behaviors. Thus, the criterion followed in this work was to define an adequately low and high cut-off for each linear region, to ensure proper goodness fits, i.e., the coefficients of determination (R^2) from linear fits had to be, at least, greater than 0.90 for

every q moment. According to the shape of the *assembled* fluctuation function obtained from the programs analyzed, three typologies can be differentiated, and each will be discussed separately. The crossover location and lower and upper limits for linear regressions are summarized in Table 3.

Table 3. High and low cut-off for each linear region of the different assembled fluctuation functions for each Sw program.

Benchmark	**Fit 1**		**Fit 2**		**Fit 3**	
	s_{min}	s_{max}				
B1 idle state	19	6637				
	s_{min}	s_x	s_x	s_{max}		
B2 unroll sum	21	3957	3957	18,340		
B3 roll sum	21	1160	1160	18,340		
B4 unroll product	21	1576	1576	18,340		
B5 roll product	21	2142	2142	18,340		
B7 nettle SHA256	20	148	148	10,507		
B8 picojpeg	20	143	143	14,144		
	s_{min}	s_{x_1}	s_{x_1}	s_{x_2}	s_{x_2}	s_{max}
B6 save roll product	21	539	539	3957	3957	18,340

As can be seen in Figure 4a, B1 lacked of any crossover point, so the linear fits were set between the aforementioned limits, i.e., from $s_{min} = 19$ to s_{max} = 6637, which approximately represented 0.01 s and 4 s, respectively. The R^2 values were greater than 0.999 for each q-moment, meeting the goodness-of-fit criterion. This implies that the scaling behavior of B1 was identical across the complete time range of the analysis. This is designated as Fit 1 in Table 3 for the B1 idle state program. Figure 5a depicts the values of the generalized Hurst dimension function of the B1 program, along with the standard error as vertical bars, which are almost imperceptible due to the quality of the linear fit. The estimation of $\langle h(q) \rangle$ was carried out by applying Equation (7). As seen, $\langle h(q) \rangle$ was practically horizontal, with values very close to 0.500. This implies that the consumption time series from an idle state program running in the microcontroller were monofractal signals.The value of $\langle h(2) \rangle = 0.499$ obtained represents the uncorrelated nature of the device going into sleep mode.

The most generalized typology for the studied benchmark is explained next. A single crossover was found in each fluctuation function for B2, B3, B4, B5, B7, and B8, which allowed differentiating two scaling regimes, one on each side of the crossovers. These crossovers are depicted in Figure 4 as s_x, and as can be seen their locations were different for each Sw program. Simple sum and product programs (B2, B3, B4, and B5) exhibited a crossover at medium and high scales, ranging from s_x = 1160 to 3957, which represented time scales between 0.7 s and 2.4 s. By contrast, as observed in Figure 4g,h, the *nettle-SHA256* and *picojpeg* programs presented a crossover at smaller scales, s_x = 148 and 143, respectively, which represented approximately 0.09 s. In either case, this allowed linear adjustments to be made on both sides of the crossover, as detailed in Table 3 as Fit 1 and Fit 2 (left and right sides, respectively).

The generalized Hurst exponent functions were estimated as the slopes of the linear regressions of the fluctuation functions and are depicted in Figure 5b,c. Figure 5b shows the values of the $\langle h(q) \rangle$ for fits to the left of the crossover, which can be considered small scales. Furthermore, Figure 5c represents the values of $\langle h(q) \rangle$ for large scales. The R^2 values were grater than 0.95 for all programs and every q-moment considered. Error estimations are also depicted as vertical bars. The main feature is that all $\langle h(q) \rangle$ are almost horizontal, so this independence of q evidences that the energy consumption behavior of these programs tended to be monofractal, both for short and long time scales. Monofractallity indicates that the scaling behavior of the variances $(F^2(s, v))$, computed in Step 3 in the MFDFA, was identical for all segments v [1].

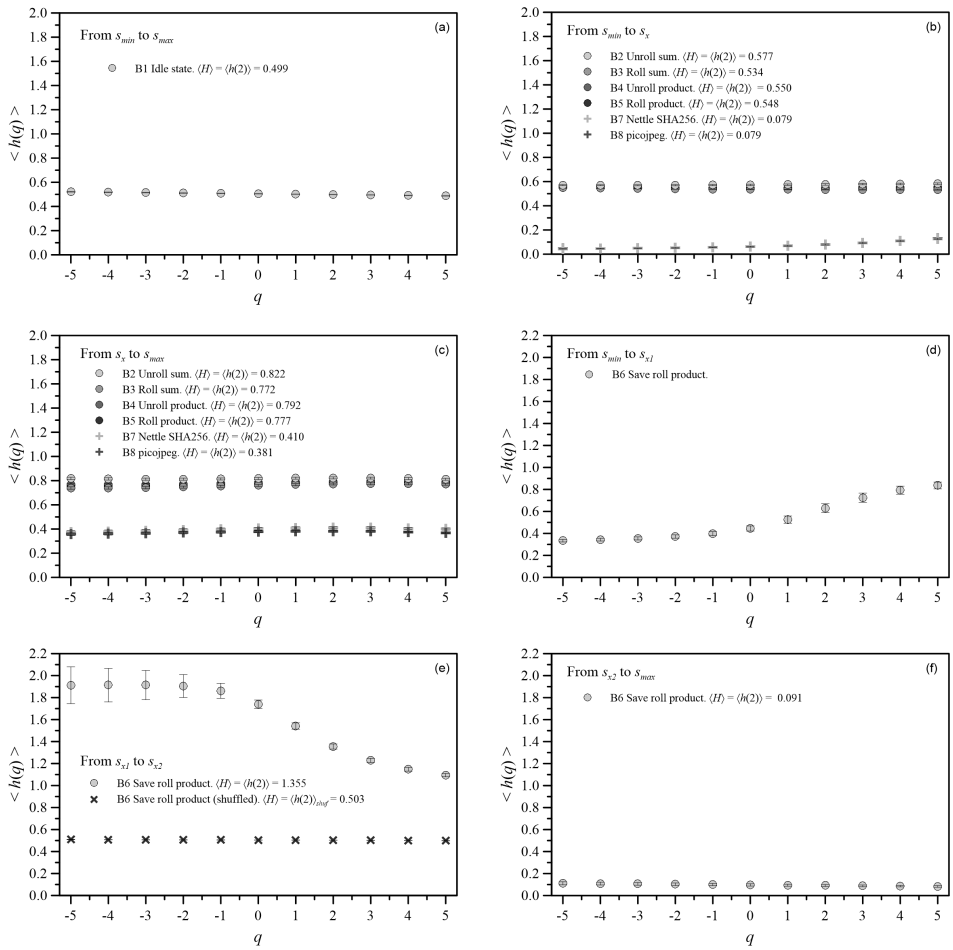

Figure 5. Generalized Hurst exponent $h(q)$ obtained from the multifractal fluctuation functions for (**a**) benchmark B1, (**b**) fit 1 of benchmarks B2, B3, B4, B5, B7 and B8, (**c**) fit 2 of benchmarks B2, B3, B4, B5, B7 and B8, (**d**) fit 1 of benchmark B6, (**e**) fit 2 of benchmark B6, and (**f**) fit 3 of benchmark B6.

However, differences exist between the behavior of the simple programs (B2, B3, B4, and B5) compared to the more complex ones (B7 and B8). On the one hand, if we focus on small scales (Figure 5b), the simple programs tended to show almost overlapping values of $\langle h(q) \rangle$ (gray circles). They not only presented a monofractal behavior but their $\langle h(2) \rangle$ values varied from 0.534 to 0.577, which allowed defining them as stationary and slightly persistent signals. Regarding the complex programs, they presented very similar $\langle h(q) \rangle$ values (see gray crosses), which tended to slightly increased with values of $\langle h(2) \rangle$ close to zero, meaning there was no scaling behavior for small time scales for the *nettle-SHA256* and *picojpeg* programs. On the other hand, linear fits to the right of the crossover allowed estimating $\langle h(q) \rangle$ for large scales. As can be seen in Figure 5c, all the programs were monofractal given the independence of $\langle h(q) \rangle$ versus q. Additionally, the functions tended to overlap according to the type of program, so that the time series of the simple programs B2, B3, B4, and B5 had a value of $\langle h(2) \rangle$ that oscillated between 0.772 and 0.822 (gray circles), which enabled defining them as stationary and persistent signals (i.e., high (low) fluctuations are likely to be followed by high (low) fluctuations). On the other hand, for large scales, the behavior of the *nettle-SHA256* and *picojpeg* signals was similar (gray crosses) with a value of $\langle h(2) \rangle$ close to 0.4, which shows that these signals were stationary and anti-persistent (and the opposite, high–low–fluctuations were more often followed by low–high–fluctuations).

The last typology was exclusively shown by the B6 Sw program, which was the only one that included a large number of memory storage operations. As seen in Figure 4f, the fluctuation function of B6 was completely different to the others. On the one hand, the functions for each q were no longer parallel along all the scales, diverging in interme-

diate scales. On the other hand, its shape allowed establishing two crossovers (s_{x1} and s_{x2}), which defined three regions: (i) small scales from $s_{min} = 21$ to $s_{x1} = 539$, (ii) medium scales from $s_{x1} = 539$ to s_{x2} = 3957, and (iii) large scales from s_{x2} = 3957 to s_{max} = 18,340. These are summarized in Table 3 as Fit 1, Fit 2, and Fit 3, respectively. These two crossovers corresponded to a time stamp of approximately 0.33 s (s_{x1}) and 2.4 s (s_{x2}). Figure 5d–f show the values of $\langle h(q) \rangle$ obtained for the linear fits in the three previously delimited regions, along with their estimation errors. Figure 5d represents an increasing function $\langle h(q) \rangle$, which suggests that the B6 program signal did not show scaling behavior for small scales. Figure 5e shows the dependency of $\langle h(q) \rangle$ on q. This decreasing relationship shows that there was a multifractal behavior for intermediate scales in the consumption of the B6 program. The width of $\langle h(q) \rangle$ measured the strength of the multifractality, in this case being $\Delta h(q)$ = 0.818. Although this fact indicates that the scaling behavior of the large and small fluctuations scaled differently, being strict, it can be observed that $\langle h(q) \rangle$ was insensitive to negative q-moments and was exclusively decreasing for positive ones. This phenomenon, known as leveling of the q-order Hurst exponent [27], indicates that large fluctuations were responsible for the multifractal behavior of these signals. The $\langle h(2) \rangle$ greater than 1 indicates that the signal was non-stationary with a Hurst exponent $H = 1.355 - 1 = 0.335$, which indicates anti-persistency. Finally, Figure 5f depicts the behavior for large time scales. As can be seen, it became monofractal, with values of $\langle h(2) \rangle$ close to zero.

In order to determine the source of the multifractality found in the medium range scales of B6 (Figure 5e), another study was accomplished. It is known that multifractality can be due to both a broad probability density function and different long-range correlations of small and large fluctuations [1]. Owing to the fact that multifractality originating from correlations of small and large fluctuations can be destroyed through a random shuffling of the signal, this process is a simple way to determine the origin of this multifractality. Conversely, it should be noted that the multifractality due to the probability density function cannot be removed with a shuffling procedure, because the order of the signal data does not influence this function. Therefore, a random shuffling of the 100 signals taken from the B6 program was carried out. Subsequently, all of them were introduced in the MFDFA, so that an *assembled* fluctuation function ($\langle F_q(s)_{shuf} \rangle$) was obtained, which is represented in Figure 6. As can be seen, the new *assembled* fluctuation function changed notably after shuffling: the crossovers disappeared and the functions became parallel. Linear fits were made to obtain the $\langle h(q)_{shuf} \rangle$, which is shown in Figure 5e through crosses. It is noticeable that the shuffled signal lost its multifracality, and its Hurst exponent H_{shuf} was 0.503, very close to non-correlation.Therefore, since the multifractality completely disappeared after shuffling, it can be stated that this was due to different long-range correlations of small and large fluctuations. This correlation was due to the specific order in which the operations of the Sw was performed, i.e., due to the Sw itself. Therefore, the multifractal features obtained were descriptive for that specific Sw program.

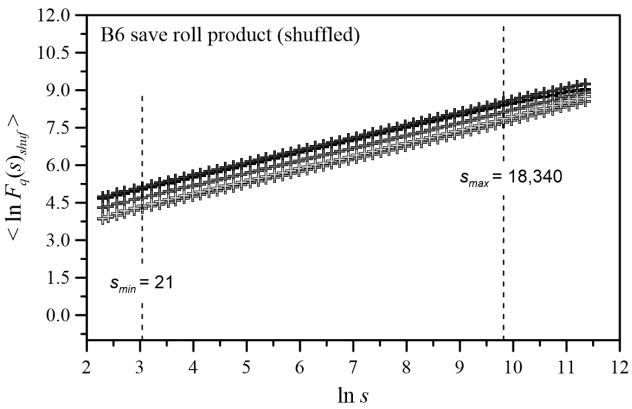

Figure 6. Assembled fluctuation function for the shuffled B6 signal.

5. Validation of the Proposed MS-MFDFA Methodology

In order to deal with uncertainty signals, in this work, we propose a novel methodology that automatically applies the MFDFA algorithm to multiple signals. MS-MFDFA proposes to execute the four first steps of the MFDFA algorithm for each of the signals individually, calculate the *assembled* fluctuation function ($\langle F_q(s) \rangle$) as the average value of the $F_q(s)$ functions, and then apply the fifth step of the MFDFA algorithm to it.

In order to validate MS-MFDFA, the representativeness of the *assembled* fluctuation function as the overall behavior of each of the 100 signals was evaluated for each of the eight Sw programs studied. For this, five q-moments were selected: -5, -2, -0, 2, and 5, and the coefficient of determination (R^2) and root mean squared error (RMSE) metrics were used. Table 4 presents the values of R^2 and RMSE obtained from the linear fits of the *assembled* fluctuation functions in rows *Assem Bi*. Each Sw program presents one row per linear regression performed considering the crossover calculated for the *assembled* function. As observed, the goodness-of-fit criterion established in this work was met for most of the linear fits performed, which were higher than 0.90, with only one exception found for Fit 3 and q = -5 in program B6. It can be observed that the goodness-of-fit was above 0.99 for all monofractal benchmarks of simple programs (B1–B5), presenting lower values for those that were more complex (B7 and B8) or for multifractal (B6) software programs.

Table 4. Analysis of the quality of the linear regressions by means of the coefficient of determination (R^2) and root mean squared error (RMSE) for each Sw program for a set of q values: $q = \{-5, -2, 0, 2, 5\}$. The rows Assem Bi show the R^2 and RMSE of the *assembled* function i with respect to its linear regression. Rows Fit j show the average value on top, and in the bottom row are the average standard deviation of the R^2 and RMSE of the 100 executions in terms of the linear regression of the *assembled* function.

Bi	R^2 $q=-5$	RMSE $q=-5$	R^2 $q=-2$	RMSE $q=-2$	R^2 $q=0$	RMSE $q=0$	R^2 $q=2$	RMSE $q=2$	R^2 $q=5$	RMSE $q=5$
Assem B1	0.9993	0.0230	0.9996	0.0158	0.9997	0.0122	0.9998	0.0100	0.9998	0.0113
Fit 1	0.9938	0.0653	0.9950	0.0567	0.9951	0.0555	0.9944	0.0590	0.9917	0.0713
	0.0042	*0.0206*	*0.0037*	*0.0202*	*0.0036*	*0.0199*	*0.0039*	*0.0203*	*0.0050*	*0.0220*
Assem B2	0.9987	0.03096	0.9980	0.0388	0.9972	0.0465	0.9963	0.0540	0.9954	0.0613
Fit 1	0.9970	0.0464	0.9969	0.0477	0.9961	0.0542	0.9949	0.0627	0.9927	0.0760
	0.0015	*0.0119*	*0.0014*	*0.0115*	*0.0016*	*0.0122*	*0.00215*	*0.0134*	*0.0030*	*0.0164*
Assem B2	0.9986	0.0144	0.9985	0.0150	0.9985	0.0153	0.9985	0.0154	0.9982	0.0165
Fit 2	0.8761	0.1268	0.9058	0.1084	0.9134	0.1038	0.9008	0.1120	0.8502	0.1385
	0.1432	*0.0498*	*0.1155*	*0.0490*	*0.0999*	*0.0514*	*0.1073*	*0.0547*	*0.1438*	*0.0591*
Assem B3	0.9992	0.0176	0.9997	0.0100	0.9996	0.0118	0.9993	0.0167	0.9986	0.0235
Fit 1	0.9983	0.0262	0.9990	0.0187	0.9989	0.0193	0.9984	0.0239	0.9968	0.0349
	0.0005	*0.0047*	*0.0005*	*0.0056*	*0.0007*	*0.0068*	*0.0009*	*0.0074*	*0.0016*	*0.0092*
Assem B3	0.9967	0.0355	0.9976	0.0306	0.9979	0.0289	0.9983	0.0267	0.9987	0.0227
Fit 2	0.9688	0.1050	0.9778	0.0881	0.9788	0.0869	0.9749	0.0965	0.9613	0.1199
	0.0206	*0.0329*	*0.0166*	*0.0327*	*0.0168*	*0.0366*	*0.0197*	*0.0409*	*0.0297*	*0.0481*
Assem B4	0.9993	0.0185	0.9994	0.0164	0.9989	0.0224	0.9982	0.0299	0.9970	0.0388
Fit 1	0.9981	0.02972	0.9986	0.0248	0.9982	0.0287	0.9973	0.0358	0.9953	0.0473
	0.0009	*0.0066*	*0.0008*	*0.0073*	*0.0010*	*0.0080*	*0.0013*	*0.0088*	*0.0020*	*0.0111*
Assem B4	0.9974	0.02917	0.9982	0.0240	0.9985	0.0227	0.9985	0.0226	0.9984	0.02316
Fit 2	0.9452	0.1228	0.9637	0.1006	0.9690	0.0947	0.9655	0.1010	0.9496	0.1230
	0.0585	*0.04881*	*0.0367*	*0.0418*	*0.0314*	*0.0388*	*0.0356*	*0.0412*	*0.0477*	*0.0490*
Assem B5	0.9993	0.0197	0.9992	0.0207	0.9987	0.0268	0.9979	0.0339	0.9968	0.0425
Fit 1	0.9980	0.0324	0.9983	0.0291	0.9978	0.0337	0.9969	0.0409	0.9949	0.0525
	0.0011	*0.0086*	*0.0010*	*0.0088*	*0.0011*	*0.0089*	*0.0015*	*0.0099*	*0.0025*	*0.0130*

Table 4. *Cont.*

Bi	R^2 $q=-5$	RMSE $q=-5$	R^2 $q=-2$	RMSE $q=-2$	R^2 $q=0$	RMSE $q=0$	R^2 $q=2$	RMSE $q=2$	R^2 $q=5$	RMSE $q=5$
Assem B5	0.9976	0.02442	0.9983	0.0205	0.9984	0.0198	0.9985	0.0197	0.9985	0.0192
Fit 2	0.9383	0.1181	0.9529	0.1004	0.9552	0.0983	0.9462	0.1078	0.9154	0.1328
	0.0540	*0.0447*	*0.04527*	*0.0439*	*0.0414*	*0.0429*	*0.0582*	*0.0435*	*0.1070*	*0.0525*
Assem B6	0.9766	0.0504	0.9670	0.0669	0.9469	0.1028	0.9254	0.1746	0.9733	0.1351
Fit 1	0.9753	0.0518	0.9661	0.0679	0.9463	0.1034	0.9252	0.1749	0.9727	0.1365
	0.0025	*0.0032*	*0.0024*	*0.0031*	*0.0023*	*0.0028*	*0.0021*	*0.0026*	*0.0021*	*0.0054*
Assem B6	0.9159	0.3582	0.96549	0.2227	0.9941	0.0826	0.9965	0.0496	0.9977	0.0322
Fit 2	0.9109	0.3678	0.9635	0.2283	0.9937	0.0852	0.9962	0.0516	0.9971	0.0359
	0.0171	*0.0348*	*0.0060*	*0.0189*	*0.0005*	*0.0038*	*0.0004*	*0.0030*	*0.0007*	*0.0043*
Assem B6	0.8833	0.0196	0.9014	0.0164	0.9149	0.0143	0.9270	0.0123	0.9355	0.0105
Fit 3	0.7798	0.0262	0.7755	0.0238	0.7664	0.0225	0.7472	0.0217	0.6886	0.0218
	0.1225	*0.0061*	*0.1455*	*0.0064*	*0.1680*	*0.0068*	*0.1972*	*0.0073*	*0.2557*	*0.0081*
Assem B7	0.9691	0.0052	0.9532	0.0074	0.9450	0.0096	0.9435	0.0124	0.9690	0.0147
Fit 1	0.9509	0.0064	0.9320	0.0088	0.9179	0.0114	0.8985	0.0159	0.8054	0.0355
	0.0203	*0.0012*	*0.0263*	*0.0015*	*0.0366*	*0.0021*	*0.0710*	*0.0055*	*0.3002*	*0.0643*
Assem B7	0.9626	0.0935	0.9721	0.0841	0.9775	0.0775	0.9822	0.0699	0.9855	0.0614
Fit 2	0.9372	0.1173	0.9522	0.1065	0.9580	0.1020	0.9579	0.1024	0.9314	0.1183
	0.03867	*0.0275*	*0.0285*	*0.0268*	*0.0276*	*0.0281*	*0.0320*	*0.0332*	*0.1131*	*0.05658*
Assem B8	0.9789	0.0043	0.9655	0.0064	0.9567	0.0084	0.9520	0.0111	0.9716	0.0140
Fit 1	0.9671	0.0053	0.9519	0.0074	0.9402	0.0097	0.9266	0.0134	0.8577	0.0302
	0.0167	*0.0011*	*0.0196*	*0.0013*	*0.0247*	*0.0017*	*0.0413*	*0.0032*	*0.2356*	*0.0457*
Assem B8	0.9691	0.0871	0.9788	0.0743	0.9848	0.0642	0.9899	0.0527	0.9940	0.0390
Fit 2	0.9525	0.1063	0.9648	0.0941	0.9701	0.0879	0.9712	0.0859	0.9523	0.1026
	0.0206	*0.0222*	*0.0163*	*0.0224*	*0.0154*	*0.0237*	*0.0177*	*0.0275*	*0.0467*	*0.0385*

Moreover, the linear goodness-of-fit of each of the 100 signals was quantified with respect to the linear fit of its corresponding *assembled* function by means of the parameters R^2 and RMSE. The average values of R^2 and RMSE, along with the standard deviation in italics, are shown in Table 4 (row *Fit j*). In general, the values of R^2 and RMSE are similar if compared with the linear fits of the *assembled* itself and the average values of the 100 samples with respect to the *assembled*. Only ten out of the eighty (12.5%) possible combinations presented a goodness-of-fit criterion lower than the 0.90 established. However, the RSME presented low values in these cases.

These results denote that the *assembled* fluctuation function used was representative of the scaling behavior of the 100 signals that generated it and that the propose methodology was effective for working with an unmanageable number of signals that could not be analyzed otherwise.

6. MS-MFDFA Software

In order to automatically perform a multifractal detrended fluctuation analysis on a large set of signals, a software tool called MS-MFDFA was designed and implemented. MS-MFDFA software was applied in this work for the first time, to study the energy consumption of software programs, but it can be applied to any other type of uncertain system, noisy signals, or to the analysis of different batches of signals.

This tool has great applicability and it is freely available for the research community. A brief description of the application is provided next, but a more detailed user manual and the software itself can be found at https://efracware.uca.es/software/ once the work is published.

The proposed MS-MFDFA software is fully developed and allows the automatic MFDFA analysis of a large set of temporal signals or time series. It is composed of three

main modules, which enable not only the automatic multifractal analysis but also signal processing, in case it is needed. These three main modules are **Preprocessing**, **Multianalysis**, and **Multianalysis Quality**. The software also offers two different graph viewers: one for Matlab in *fig* format, and the other for images in *jpeg* format. The main screen of the software is shown in Figure 7 and the three main modules included in the software are outlined next.

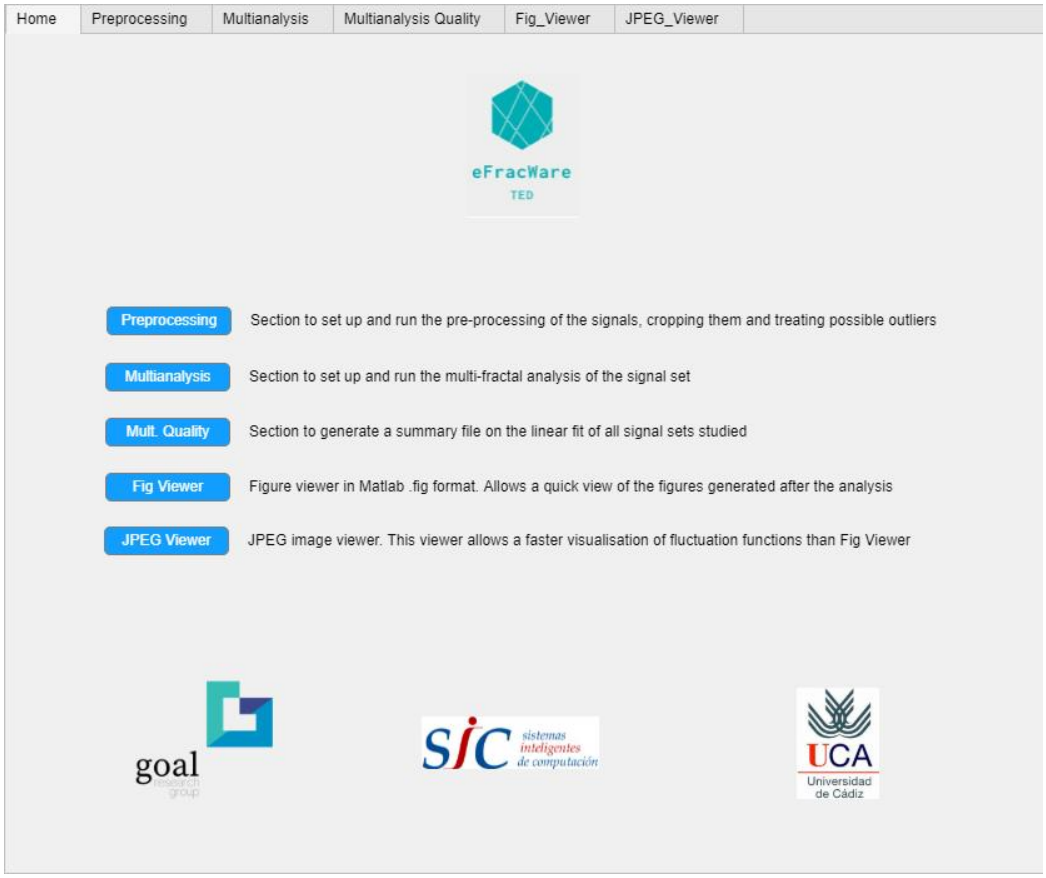

Figure 7. MS-MFDFA software home page.

The **Preprocessing** module allows the user to trim a signal, in case the user wants to obtain a sub-sample of the original signal. Moreover, it is also possible to set the order of magnitude of the standard deviation a sample must exceed to be considered an outlier.

The **Multianalysis** module generates the *assembled* fluctuation function (as shown in Figure 4). In order to keep track of all the steps in the analysis, the software not only stores the *assembled* fluctuation function, but also the fluctuation functions of each of the signals under study. Two different folders can be generated to store these graphs, in *fig* or *jpeg* formats, respectively, and viewers are also provided. A tool for establishing the crossover points for the *assembled* fluctuation function is available, and the linear regressions for each specific section are calculated.

Finally, the software stores in an *xlsx* file the linear regressions performed on each section of the *assembled* fluctuation function and their quality indices (R^2 and RMSE) according to the fluctuation functions of the batch of signals studied.

Finally, the **Multianalysis Quality** module generates a table in *xlsx* and LaTeX *tex* formats, containing the quality indices of the study carried out, similar to the one shown in Table 4. In addition, it generates a table in *xlsx* format, containing the generalized Hurt exponent $h(q)$ obtained from the multifractal fluctuation functions, which is required to generate charts like the one shown in Figure 5.

For more information about the software capabilities, please refer to the user manual in https://efracware.uca.es/software/.

7. Conclusions and Future Work

In this work, a novel automatically-driven methodology called multi-signal multifractal detrended fluctuation analysis (MS-MFDFA) for applying MFDFA over a huge amount of signals was presented. This methodology allows, for the first time, the characterization of uncertain systems comprised of a large amount of measurements. To the best of our knowledge, the application of MF techniques to such a large amount of time series has not been addressed before. MS-MFDFA is based on computing an averaged fluctuation function, called *assembled* and denoted as $\langle F_q(s) \rangle$, which scales with s according to an exponent $\langle h(q) \rangle$, which is estimated by means of linear fits of $\langle F_q(s) \rangle$.

For the first time in the literature, multifractal techniques have been applied to analyze the energy consumption of software programs, where high uncertainty is present. In order to do so, a software benchmark composed of eight programs of different complexity was defined and proposed, to analyze the consumption of a microcontroller device, specifically an Adafruit Metro M4 platform. Because the consumption profile of a given Sw presents high uncertainty levels and requires the analysis of a large amount of signals, the MS-MFDFA methodology was developed, applied, and assessed.

According to the results, the energy consumption of program B1 was unique. The analysis suggests that these signals were stationary and showed a monofractal behavior characterized by quasi-uncorrelated fluctuations for all temporal scales. Therefore, this was a scaling feature of the baseline power consumption of the hardware. Moreover, the results evidenced that the energy consumption of the studied software programs that performed sums and/or multiplications without memory storage operations (programs B2 to B5) exhibited two scaling regimes, separated by a crossover. The crossover depended on the program, being located at 2.4 s for B2, 0.7 s for B3, 0.96 s for B4, and 1.3 for B5. These were always monofractal signals, but uncorrelated and persistent for scales lower and lager than the crossover. *nettle-SHA256* and *picojpeg*, which are more complex programs, also exhibited two scaling regimes, separated by a crossover. Unlike the previous programs, their crossovers were located with smaller temporal scaling (around 0.9 s). In this way, for very small time scales, the energy consumption of these signals did not follow a scaling behavior. Although, as seen from the said crossover, they were stationary and monofractal signals, showing anti-persistence. Therefore, the main difference between simple and complex programs is that the former are persistent, so high (low) fluctuations in consumption are likely to be followed by high (low) fluctuations, whereas in the latter, high (low) fluctuations are often followed by small (high) fluctuations, as they are anti-persistent.

We detected that the program performing memory storage (B6) showed a multifractal behavior. First of all, this signal was the only one that showed two crossovers, which allowed establishing three zones with different scaling behaviors. The scaling region for small scales did not show fractal properties, so the analysis had to be extended to intermediate scaling regions, to locate a multifractal behavior, which occurred between 0.33 and 2.4 s. In this second region, the signal was non-stationary and persistent. Its multifractality was mainly determined by large fluctuations, since small fluctuations remained independent of q-moments. The third region, the one with large scales, showed monofractality. Finally, with the objective of determining the source of multifractality, the same procedure was carried out after subjecting the signals of the B6 program to a random process of shuffling their values. The analysis of the shuffled series showed that the origin of the multifractality was due to the different long-range correlations of small and large fluctuations. That is, it depended on the order in which operations were executed, i.e., on the software features.

A study was carried out to validate the proposed MS-MFDFA methodology, showing that the *assembled* fluctuation function was representative of the scaling behavior of all single functions used to build it (highly suitable values of R^2 and RMSE metrics were obtained in all cases).

Additionally, software was developed implementing the novel proposed methodology. Its name is MS-MFDFA and it is freely available. MS-MFDFA software allows applying

MFDFA to large batches of signals, which is especially interesting for characterizing uncertain systems and noisy signals.

This work forms a basis for the full characterization of the profiles of the energy consumption of software programs, and this might lead to the categorization of the energy efficiency of software. This is a novel work, not only for the object of study, but for the treatment of large amounts of signals. The results demonstrate that the analysis of multifractal properties together with other characteristics of the software programs might lead to a differentiating pattern. This opens an interesting line of research, to better understand the behavior of the energy consumption of different software programs on different hardware. As future work, we plan to completely automatize the multifractal analysis, to allow the performance of more exhaustive experimentation. Additionally, it would be highly interesting to obtain the multifractal properties of software programs executed in more advanced architectures that exhibit higher uncertainties in their consumption, e.g., mobile phones or computers with operating systems running on them.

Author Contributions: Conceptualization, J.C.d.l.T. and P.P.-D.; Methodology, J.C.d.l.T.; Software, J.C.d.l.T. and P.R.; Validation, B.D.; Formal Analysis, P.P.-D.; Investigation, P.R.; Resources, B.D. and P.L.G.; Data Curation, J.C.d.l.T.; Writing—Original Draft Preparation, J.C.d.l.T. and P.P.-D.; Writing—Review and Editing, B.D. and P.R.; Visualization, P.P.-D. and P.L.G.; Supervision, P.R.; Project Administration, B.D.; Funding Acquisition, B.D., P.L.G. and P.R. All authors have read and agreed to the published version of the manuscript.

Funding: This publication was supported by the projects eFracWare [TED2021-131880B-I00] funded by the Spanish MCIN and the European Union "NextGenerationEU"/PRTR on MCIN/AEI/10.13039/501100011033, and the project eMob [PID2022-137858OB-I00] funded by the Spanish MCIN, and the Agency and the European Regional Development on MCIN/AEI/10.13039/501100011033/FEDER, UE. This work was also supported by Junta de Andalucía and ERDF under contract P18-2399 (GENIUS), ERDF [OPTIMALE – FEDER-UCA18-108393], and the Spanish Ministerio de Ciencia, Innovación y Universidades and the ERDF [NEMOVISION PID2019-109465RB-I00].

Data Availability Statement: The tool developed in this work and a detailed user manual will be freely available for the research community at https://efracware.uca.es/software/ once the work has been published.

Acknowledgments: J.C. de la Torre would like to acknowledge the Spanish Ministerio de Ciencia, Innovación y Universidades for the support through FPU grant [FPU17/00563]. B. Dorronsoro and P. Ruiz acknowledge "ayuda de recualificación" funding by Ministerio de Universidades and the European Union-NextGenerationEU.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Kantelhardt, J.; Zschiegner, S.A.; Koscielny-Bunde, E.; Havlin, S.; Bunde, A.; Stanley, H. Multifractal detrended fluctuation analysis of nonstationary time series. *Phys. A Stat. Mech. Its Appl.* **2002**, *316*, 87–114. [CrossRef]
2. Wang, J.; Shang, P.; Dong, K. Effect of linear and nonlinear filters on multifractal analysis. *Appl. Math. Comput.* **2013**, *224*, 337–345. [CrossRef]
3. Lin, J.; Dou, C.; Liu, Y. Multifractal detrended fluctuation analysis based on optimized empirical mode decomposition for complex signal analysis. *Nonlinear Dyn.* **2021**, *103*, 2461–2474. [CrossRef]
4. Saâdaoui, F. Skewed multifractal scaling of stock markets during the COVID-19 pandemic. *Chaos Solitons Fractals* **2023**, *170*, 113372. [CrossRef] [PubMed]
5. Plocoste, T.; Pavón-Domínguez, P. Temporal scaling study of particulate matter (PM10) and solar radiation influences on air temperature in the Caribbean basin using a (3D) joint multifractal analysis. *Atmos. Environ.* **2020**, *222*, 117115. [CrossRef]
6. Shang, P.; Lu, Y.; Kamae, S. Detecting long-range correlations of traffic time series with multifractal detrended fluctuation analysis. *Chaos Solitions Fractals* **2008**, *36*, 82–90. [CrossRef]
7. Pavlov, A.; Abdurashitov, A.; Koronovskii, A.; Pavlova, O.; Semyachkina-Glushkovskaya, O.; Kurths, J. Detrended fluctuation analysis of cerebrovascular responses to abrupt changes in peripheral arterial pressure in rats. *Commun. Nonlinear Sci. Numer. Simul.* **2020**, *85*, 105232. [CrossRef]
8. Weng, Y.; Unni, V.R.; Sujith, R.; Saha, A. Synchronization-based model for turbulent thermoacoustic systems. *Nonlinear Dyn.* **2023**, *111*, 12113–12126. [CrossRef]

9. Zorick, T.; Mandelkern, M.A. Multifractal detrended fluctuation analysis of human EEG: Preliminary investigation and comparison with the wavelet transform modulus maxima technique. *PLoS ONE* **2013**, *8*, e68360. [CrossRef]
10. Kubota, M.; George, Z. Differentiation of task complexity in long-term memory retrieval using multifractal detrended fluctuation analysis of fNIRS recordings. *Exp. Brain Res.* **2022**, *240*, 1701–1711. [CrossRef]
11. Telesca, L.; Lapenna, V.; Macchiato, M. Multifractal fluctuations in earthquake-related geoelectrical signals. *New J. Phys.* **2005**, *7*, 214–214. [CrossRef]
12. Flores-Márquez, E.; Ramírez-Rojas, A.; Telesca, L. Multifractal detrended fluctuation analysis of earthquake magnitude series of Mexican South Pacific Region. *Appl. Math. Comput.* **2015**, *265*, 1106–1114. [CrossRef]
13. David, S.; Machado, J.; Inácio, C.; Valentim, C. A combined measure to differentiate EEG signals using fractal dimension and MFDFA-Hurst. *Commun. Nonlinear Sci. Numer. Simul.* **2020**, *84*, 105170. [CrossRef]
14. Sarker, A.; Mali, P. Detrended multifractal characterization of Indian rainfall records. *Chaos Solitons Fractals* **2021**, *151*, 111297. [CrossRef]
15. Nashat, D.; Hussain, F.A. Multifractal detrended fluctuation analysis based detection for SYN flooding attack. *Comput. Secur.* **2021**, *107*, 102315. [CrossRef]
16. Schadner, W. U.S. Politics from a multifractal perspective. *Chaos Solitons Fractals* **2022**, *155*, 111677. [CrossRef]
17. Zhou, W.X. Multifractal detrended cross-correlation analysis for two nonstationary signals. *Phys. Rev. E-Stat. Nonlinear Soft Matter Phys.* **2008**, *77*, 2–5. [CrossRef]
18. Zhang, X.; Zeng, M.; Meng, Q. Multivariate multifractal detrended fluctuation analysis of 3D wind field signals. *Phys. A Stat. Mech. Its Appl.* **2018**, *490*, 513–523. [CrossRef]
19. Fan, Q.; Liu, S.; Wang, K. Multiscale multifractal detrended fluctuation analysis of multivariate time series. *Phys. A Stat. Mech. Its Appl.* **2019**, *532*, 121864. [CrossRef]
20. Li, S.; Li, J.; Lu, X.; Sun, Y. Exploring the dynamic nonlinear relationship between crude oil price and implied volatility indices: A new perspective from MMV-MFDFA. *Phys. A Stat. Mech. Its Appl.* **2022**, *603*, 127684. [CrossRef]
21. Calero, C.; Piattini, M. Puzzling out Software Sustainability. *Sustain. Comput. Inform. Syst.* **2017**, *16*, 117–124. [CrossRef]
22. Ghaleb, T.A. Software Energy Measurement at Different Levels of Granularity. In Proceedings of the 2019 International Conference on Computer and Information Sciences (ICCIS), Sakaka, Saudi Arabia, 3–4 April 2019; pp. 1–6. [CrossRef]
23. Chantem, T.; Guan, N.; Liu, D. Sustainable embedded software and systems. *Sustain. Comput. Inform. Syst.* **2019**, *22*, 152–154. [CrossRef]
24. Mancebo, J.; Calero, C.; Garcia, F.; Moraga, M.A.; Garcia-Rodriguez de Guzman, I. FEETINGS: Framework for Energy Efficiency Testing to Improve Environmental Goal of the Software. *Sustain. Comput. Inform. Syst.* **2021**, *30*, 100558. [CrossRef]
25. Mancebo, J.; García, F.; Calero, C. A process for analysing the energy efficiency of software. *Inf. Softw. Technol.* **2021**, *134*, 1–15. [CrossRef]
26. Aragón, J.M.; de la Torre, J.C.; Dorronsoro, B. Técnicas de Optimización para Problemas con Incertidumbre y su Aplicación a la Optimización Robusta de Programas Informáticos. Master's Thesis, Universidad de Cádiz, Cádiz, Spain, 2021.
27. Ihlen, E. Introduction to multifractal detrended fluctuation analysis in Matlab. *Front. Physiol.* **2012**, *3*, 141. [CrossRef] [PubMed]
28. Rydin Gorjão, L.; Hassan, G.; Kurths, J.; Witthaut, D. MFDFA: Efficient multifractal detrended fluctuation analysis in python. *Comput. Phys. Commun.* **2022**, *273*, 108254. [CrossRef]
29. Mihai, V.; Dragana, C.; Popescu, D.; Ichim, L. Condition Monitoring of Manufacturing Production Lines Using Fractal Analysis of Energy Consumption Datasets. In Proceedings of the 23rd International Conference on Control Systems and Computer Science (CSCS), Bucharest, Romania, 26–28 May 2021; pp. 249–253. [CrossRef]
30. Yuan, X.; Ji, B.; Yuan, Y.; Huang, Y.; Li, X.; Li, W. Multifractal detrended fluctuation analysis of electric load series. *Fractals* **2015**, *23*, 1550010. [CrossRef]
31. Li, H.; Wang, Y.; Zhang, X.; Fu, G. Evaluation Method of Wind Power Consumption Capacity Based on Multi-Fractal Theory. *Front. Energy Res.* **2021**, *9*, 634551. [CrossRef]
32. Ali, H.; Aslam, F.; Ferreira, P. Modeling Dynamic Multifractal Efficiency of US Electricity Market. *Energies* **2021**, *14*, 6145. [CrossRef]
33. Norouzzadeh, P.; Dullaert, W.; Rahmani, B. Anti-correlation and multifractal features of Spain electricity spot market. *Phys. A Stat. Mech. Its Appl.* **2007**, *380*, 333–342. [CrossRef]
34. Calero, C.; Ángeles Moraga, M.; Piattini, M. *Software Sustainability*; Springer: Berlin/Heidelberg, Germany, 2021.
35. Ibrahim, S.R.A.; Yahaya, J.; Salehudin, H.; Bakar, N.H. Towards Green Software Process: A Review on Integration of Sustainability Dimensions and Waste Management. In Proceedings of the 2019 International Conference on Electrical Engineering and Informatics (ICEEI), Bandung, Indonesia, 9–10 July 2019; pp. 128–133. [CrossRef]
36. Kern, E.; Hilty, L.M.; Guldner, A.; Maksimov, Y.V.; Filler, A.; Gröger, J.; Naumann, S. Sustainable software products—Towards assessment criteria for resource and energy efficiency. *Future Gener. Comput. Syst.* **2018**, *86*, 199–210. [CrossRef]
37. Naumann, S.; Dick, M.; Kern, E.; Johann, T. The GREENSOFT Model: A reference model for green and sustainable software and its engineering. *Sustain. Comput. Inform. Syst.* **2011**, *1*, 294–304. [CrossRef]
38. Peng, C.K.; Buldyrev, S.V.; Havlin, S.; Simons, M.; Stanley, H.E.; Goldberger, A.L. Mosaic organization of DNA nucleotides. *Phys. Rev. E* **1994**, *49*, 1685–1689. [CrossRef]
39. Feder, J. *Fractals*; Plenum Press: New York, NY, USA, 1988.

40. Movahed, M.S.; Jafari, G.R.; Ghasemi, F.; Rahvar, S.; Tabar, M.R.R. Multifractal detrended fluctuation analysis of sunspot time series. *J. Stat. Mech. Theory Exp.* **2006**, *2006*, P02003. [CrossRef]
41. Zhang, Q.; Xu, C.Y.; Chen, Y.D.; Yu, Z. Multifractal detrended fluctuation analysis of streamflow series of the Yangtze River basin, China. *Hydrol. Process.* **2008**, *22*, 4997–5003. [CrossRef]
42. Möller, N. Nettle: A Low-Level Cryptographic Library. 2009. Available online: https://github.com/breadwallet/nettle (accessed on April 2023).
43. Patterson, D. Open Benchmarks for Embedded Platforms. 2018. Available online: https://github.com/embench/embench-iot (accessed on April 2023).
44. Dorronsoro, B.; Ruiz, P. Intelligent Generation of Sustainable Software (GENIUS) research project, 2019–2022. reference PY18-2399.
45. Kantelhardt, J.W., Fractal and Multifractal Time Series. In *Mathematics of Complexity and Dynamical Systems*; Springer: New York, NY, USA, 2011; pp. 463–487. [CrossRef]