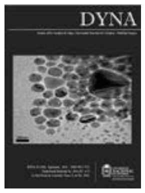
UNIVERSIDAD NACIONAL DE COLOMBIA
SEDE MEDELLÍN
FACULTAD DE MINAS



DYNA
http://dyna.medellin.unal.edu.co/

Algorithm for wideband spectrum sensing based on sparse Fourier transform

Alexander López-Parrado [ab] & Jaime Velasco-Medina [b]

[a] *GDSPROC Research Group, Universidad del Quindío, Armenia, Colombia. parrado@uniquindio.edu.co*
[b] *Bionanoelectronics Research Group, Universidad del Valle, Santiago de Cali, Colombia. jaime.velasco@correounivalle.edu.co*

Received: January 27th, 2015. Received in revised form: August 07th, 2015. Accepted: March 11th, 2016.

Abstract
In this paper we present a novel sub-Nyquist algorithm to perform Wideband Spectrum Sensing (WSS) for Cognitive Radios (CRs) by using the recently developed Sparse Fast Fourier Transform (sFFT) algorithms. In this case, we developed a noise-robust sub-Nyquist WSS algorithm with reduced sampling cost, by modifying the Nearly Optimal sFFT algorithm; this was accomplished by using Gaussian windows with small support. Simulation results show that the proposed algorithm is suitable for hardware implementation of WSS systems for sparse spectrums composed of highly-noisy multiband-signals.

Keywords: Cognitive Radio; Compressed Sensing; Sparse Fourier Transform; Spectrum Sensing.

Algoritmo para sensado de espectro de banda ancha basado en transformada dispersa de Fourier

Resumen
En este trabajo se presenta un nuevo algoritmo sub-Nyquist para realizar Sensado de Espectro de Banda Ancha (WSS) para Radios Cognitivos (CR) mediante el uso de los algoritmos de Transformada Dispersa de Fourier (sFFT) recientemente desarrollados. En este caso, hemos desarrollado un algoritmo sub-Nyquist robusto ante el ruido para WSS con reducción en el costo de muestreo, mediante la modificación del algoritmo sFFT casi óptimo; esto se logró mediante el uso de ventanas Gaussianas con soporte pequeño. Los resultados de simulación muestran que el algoritmo propuesto es adecuado para la implementación hardware de sistemas WSS sobre espectros dispersos compuestos por señales multibanda altamente ruidosas.

Palabras clave: Radio Cognitiva; Sensado Compresivo; Transformada Dispersa de Fourier; Sensado de Espectro.

1. Introduction

Cognitive Radio (CR) is becoming the new paradigm for developing the next generation of radio communication systems. CR addresses the issue of spectrum misuse of current radio communication systems by adding cognitive features to the radios such as: spectrum sensing (SS), power control and spectrum management [1,2]. These features are initially presented in the IEEE 802.22 [3] standard, which was developed in 2011 by the Institute of Electrical and Electronics Engineers (IEEE) and defines a Wireless Regional Area Network (WRAN) that uses the Very High Frequency (VHF) and Ultra High Frequency (UHF) television (TV) bands by considering cognitive radio capabilities.

One technical challenge in CR is the efficient implementation of the SS function for a higher bandwidth by minimizing the required sampling rate. The SS function detects Primary Users (PUs) or Secondary Users (SUs) in some regions of the spectrum, allowing an opportunistic usage of the available bands. The IEEE 802.22 standard has an informative annex that defines two categories of SS techniques: blind sensing and signal specific sensing. Blind sensing techniques use energy measures, and signal specific

DYNA 83 (198), pp. 79-86, Septiembre, 2016. Medellín. ISSN 0012-7353 Printed, ISSN 2346-2183 Online
DOI: http://dx.doi.org/10.15446/dyna.v83n198.48654

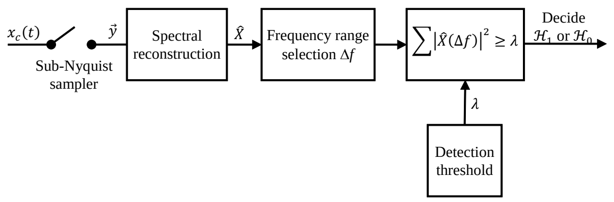

$x_c(t)$
Sub-Nyquist sampler
$\vec{y}$
Spectral reconstruction
$\hat{X}$
Frequency range selection Δf
$\sum|\hat{X}(\Delta f)|^2 \geq \lambda$
Decide $\mathcal{H}_1$ or $\mathcal{H}_0$
λ
Detection threshold

Figure 1. Block diagram of a Sub-Nyquist WSS system.
Source: [9].

sensing techniques use preambles and pilot signals. However, these sensing techniques are narrowband because they can only sense a single carrier frequency.

On the one hand, measurements carried out by the Microsoft Spectrum Observatory at Washington DC [4] show that around 3% of the band between 30 MHz and 3 GHz is used sparsely to host most of the worldwide radio services [4]. This sparse occupancy of the radio electric spectrum has motivated the theoretical research about Wideband Spectrum Sensing (WSS) techniques during the last five years [5-7]. In this case, the research results have shown that sub-Nyquist sampling techniques such as Analog to Information Conversion (AIC) [6],[8], Modulated Wideband Conversion (MWC) [6][10] and Multi Coset (MC) sampling [6],11,12] are promissory candidates for developing WSS systems as the shown in Fig. 1

The sub-Nyquist WSS systems are usually composed of a sub-Nyquist sampler, a spectral reconstruction block and a decision stage [5,6[9]. The spectral reconstruction block is usually constructed by using Compressive Sensing (CS) techniques [13,14].

On the other hand, spectral reconstruction considering the Sparse Fast Fourier Transform (sFFT) algorithms [15-19] has not been well developed as these algorithms either require a high sampling cost for typical spectrum occupancy [15,16,[19], they are very noise sensitive [17], or they are too complex [18].

In the context of WSS algorithms, the challenge is to achieve sub-Nyquist sampling rates using low-power and low-speed ADCs for highly-noisy signals. Thus, considering the above, the main contribution of this paper is the design of a new sub-Nyquist WSS algorithm that reduces the sampling cost by using a modified version of the sFFT algorithm with Gaussian small support windows. This proposed algorithm is very suitable for hardware implementation of WSS systems using ASICs or FPGAs, and to the best of our knowledge, it is the first that uses the recently developed Nearly Optimal Sparse Fourier Transform.

The rest of the paper is organized as follows: Section 2 presents some mathematical basics about the sub-Nyquist WSS algorithm we developed, Section 3 describes the proposed sub-Nyquist WSS algorithm, Section 4 presents simulation results performed on scenarios composed of highly-noisy multiband-signals, and, Section 5, presents our conclusions and suggestions for future work.

2. Mathematical background

In this section, we present some fundamental concepts about the sub-Nyquist sFFT algorithm we developed. First, we explain some basics about DFT and sparse signals, and second, we describe the mathematical tools pseudo-random spectral permutation, filtering window, and hashing function.

2.1. Discrete Fourier transform and sparsity

Given a discrete time signal $\boldsymbol{x} \in \mathbb{C}^N$ of length N, its N-point Discrete Fourier Transform (DFT) $\widehat{\boldsymbol{x}} \in \mathbb{C}^N$ is defined in Eq. (1).

$$\widehat{\boldsymbol{x}}_k = \frac{1}{N} \sum_{n \in [N]} \boldsymbol{x}_n \, \omega^{kn}, k \in [N] \tag{1}$$

Where N is a power of two, $[N]$ denotes the set of indexes $\{0,1,\ldots,N-1\}$, and $\omega = e^{-\mathrm{i}\,2\pi/N}$ is the N-th root of unity. In this case, the number of non-zero elements of the vector $\widehat{\boldsymbol{x}}$ is called the sparsity order K and is defined in Eq. (2).

$$K = |\mathrm{supp}(\widehat{\boldsymbol{x}})|_0 \tag{2}$$

Where $\mathrm{supp}(\widehat{\boldsymbol{x}})$ is the set of indexes of the non-zero elements of the vector $\widehat{\boldsymbol{x}}$, and $|\quad|_0$ represents the l_0-norm of the vector. Then, a time domain signal $\boldsymbol{x}$ is sparse in the DFT domain if $K \ll N$.

In this context, a set of algorithms called sFFT takes advantage of the signal sparsity in the DFT domain to speed up the runtime of the Fast Fourier Transform (FFT) algorithms used to calculate the DFT [15-18]. These sFFT algorithms, like the Nearly Optimal sFFT algorithm presented in [16], use the following mathematical tools: pseudo-random spectral permutation [15-19], filtering window [16] and hashing function [15,16].

2.2. Pseudo-random spectral permutation

This permutation isolates spectral components from each other [19] and is performed as described in Eq. (3).

$$\begin{gathered} \boldsymbol{xp}_n = \boldsymbol{x}_{\sigma(n-a) \bmod N}\,, \\ \widehat{\boldsymbol{xp}}_{\pi_p(k,\sigma,N)} = \widehat{\boldsymbol{x}}_k \omega^{\sigma k a} \end{gathered} \tag{3}$$

Where $\boldsymbol{xp}$ and $\widehat{\boldsymbol{xp}}$ are the permuted spectrum signals in the time domain and the DFT domain, respectively; $\pi_p(k,\sigma,N) = \sigma k \bmod N$ is the spectral permutation function; and $\sigma \in \{2c+1 | c \in [N/2]\}$ and $a \in [N]$ are the spectral permutation parameters. The spectral permutation function translates the frequency bin from the k-th location to the $\pi_p(k,\sigma,N)$-th location, in this case $\sigma^{-1} \bmod N$ exists for all odd σ if N is a power of two. The sFFT algorithm randomly chooses the spectral permutation parameters σ and a from a uniform distribution. Thus, the spectral permutation with these pseudo-random parameters is related to a pseudo-random sampling scheme [15-19].

2.3. Filtering window

The filtering window is a new mathematical tool that reduces the size of the FFT from points N to B. This is accomplished in the Nearly Optimal sFFT algorithm by extending a flat passband region of width N/B around each sparse component; this approach replaces the filter bank of previous sFFT algorithms [19,20] and avoids the use of non-equispaced data FFTs [21]. Nonetheless, this flat window has a support that is not small enough to achieve sub-Nyquist sampling rates.

Thus, in order to reduce the sampling cost of the sFFT algorithm, we designed a small support window $\boldsymbol{G}, \widehat{\boldsymbol{G}}'$ such that $|\text{supp}(\boldsymbol{G})|_0 = B$; nevertheless, this small support can reduce the accuracy, which is not a big issue in the case of WSS systems. In this case, the window in the time domain is designed with an ideal low-pass filter using a Gaussian window with finite duration to truncate the impulse response. The cutoff frequency of the low-pass filter is $2C$, where $C = 1/(2B)$, and the standard deviation σ_g of the Gaussian window is obtained from the 68-95-99.7 rule [22], as described in Eq. (4).

$$\sigma_g = \frac{B}{6} \tag{4}$$

Eq. (5)-(6) describe the Gaussian window in the time domain and the DFT domain, respectively.

$$\boldsymbol{G}_{n+\frac{N}{2}\text{mod}\,N} = 2\,C\,e^{-\frac{(n-N/2)^2}{2\sigma_g^2}}\,\text{sinc}(2C(n - N/2))/\widehat{\boldsymbol{G}}_{N/2}\ , n \in [N] \tag{5}$$

$$\widehat{\boldsymbol{G}}'_{k+\frac{N}{2}\text{mod}\,N} = \begin{cases} 0, \text{if } |k - N/2| \geq N(C + 1/\sigma_g) \\ \dfrac{\text{ncdf}\left(2\pi\sigma_g\left(\frac{k-\frac{N}{2}}{N} + C\right)\right) - \text{ncdf}\left(2\pi\sigma_g\left(\frac{k-\frac{N}{2}}{N} - C\right)\right)}{\widehat{\boldsymbol{G}}_{\frac{N}{2}}}, \text{otherwise} \end{cases}, k \in [N] \tag{6}$$

Where vector $\boldsymbol{G}'$ is the approximated window and $\text{ncdf}(x) = \text{erfc}(-x\,/\,\sqrt{2})/2$ is the Normal Cumulative Distribution Function [23].

The Gaussian window is normalized both in the time domain and the DFT domain in order to achieve unit DC gain, and its total bandwidth in DFT domain is given by Eq. (7).

$$BW_{\boldsymbol{G}'} = 13N/B \tag{7}$$

Finally, the windowing process is described in Eq. (8), and it is performed in the time domain after the pseudo-random spectral permutation is carried out.

$$y = xp \circ G \tag{8}$$

Where, $\boldsymbol{y}$ is the windowed signal in the time domain.

2.4. Hashing function

The hashing function obtains B points from the N-point spectrum of the signal $\boldsymbol{y}$, these points are separated by N/B bins, and they are obtained by calculating the B-point DFT of the sub-sampled signal obtained from $\boldsymbol{y}$. The vector that has the hashes of signal $\boldsymbol{y}$ is $\widehat{\boldsymbol{u}} \in \mathbb{C}^B$ and it is calculated using Eq. (9) [15,16].

$$\widehat{\boldsymbol{u}}_{\boldsymbol{j}} = DFT\left\{\sum_{i\in[N/B]} \boldsymbol{y}_{j+Bi}\right\}, j \in [B] \tag{9}$$

From Eq. (7)-(9), it is possible to note that for each sparse component of the signal $\boldsymbol{x}$ there are 14 non-zero hashes located in the offsets given by Eqs. (10) and (11).

$$o_{fk}(j,\sigma,N,B) = \pi_p(j,\sigma,N) - (h_f(j,\sigma,N,B) - k)N/B, k \in \{0,1,\dots,6\} \tag{10}$$

$$o_{ck}(j,\sigma,N,B) = \pi_p(j,\sigma,N) - (h_c(j,\sigma,N,B) + k)N/B, k \in \{0,1,\dots,6\} \tag{11}$$

Where $h_f(j,\sigma,N,B) - k \bmod B, k \in \{0,1,\dots,6\}$ and $h_c(j,\sigma,N,B) + k \bmod B, k \in \{0,1,\dots,6\}$ are the 14 indexes of the hashes $\widehat{\boldsymbol{u}}$ for each hashed single sparse component, and $h_f(j,\sigma,N,B) = \lfloor \pi_p(j,\sigma,N)B/N \rfloor$ is the floor-hash function and $h_c(j,\sigma,N,B) = \lceil \pi_p(j,\sigma,N)B/N \rceil$ is the ceil-hash function.

Finally, it has to be noted that the use of pseudo-random spectral permutation and small support windows leads to a sub-Nyquist random sampling scheme; where, under certain conditions, the average sampling rate is below Nyquist.

3. Sub-Nyquist wideband spectrum sensing algorithm

In this section, we describe the Sub-Nyquist WSS algorithm, called *SNSparseWSS*, which presents a reduced sampling rate compared to the sFFT algorithm described in [16]. This improvement is achieved by using the window described in the past section and by performing several modifications to the procedures presented in [16].

The *SNSparseWSS* algorithm, described in Alg. 1, calculates the spectrum occupancy $\boldsymbol{X}_{rfca}$ with constant False Alarm Probability (P_{FA}) [2],[11], and has the following input parameters: the sparse bandwidth BW in Hz, the total bandwidth W in Hz, the duration of the sensing window τ in seconds, the noise power σ_n, and the constants of the sFFT algorithm R_{est} and s [16].

The algorithm calculates the spectrum occupancy using four processing stages: the first one performs sub-Nyquist sampling on the wideband complex signal $x(t)$; the second one locates the sparse components by using the vector $\boldsymbol{x}_l$, the set of permutation parameters $\boldsymbol{P}_l$, and the modified *LocateSignal* procedure [16],[24]; the third one estimates the DFT values by using the vector $\boldsymbol{x}_e$, the set of permutation parameters $\boldsymbol{P}_e$, and the modified *EstimateValues* procedure [15]; and the fourth one detects the occupied bands with constant P_{FA} using the *ConstantPfaRecovery* procedure, which determines whether a channel is occupied or not by a PU. Therefore, it is necessary to test two spectrum sensing hypotheses, H_0 for vacant channel and H_1 for occupied channel [2], by using a detector with constant P_{FA} [2],[11] as described in Eq. (12) [11].

Algorithm 1. *SNSparseWSS* algorithm.

Input: Sparse bandwidth BW in Hz, total bandwidth W in Hz, sensing time τ in seconds, number R_{est} of estimation loops, threshold s for location, noise power σ_n
Output: $\boldsymbol{X}_{rcfa}$

```
SNSparseWSS procedure $(BW, W, \tau, R_{est}, s, \sigma_n)$
    //Window generation
    $N = 2^{\lfloor \log_2 (W \times \tau) \rfloor}$;
    $K = \lceil BW/W \times N \rceil$;
    $B = 2^{\lfloor \log_2 K \rfloor + 1}$;
    $R_{loc} = \lfloor \log_2 \log_2 N \rfloor$;
    Calculate Gaussian Window ($\boldsymbol{G}, \widehat{\boldsymbol{G}}'$) by using Eqs. (5) and (6).
    //Sub-Nyquist sampling set genration
    Pre-calculate all permutation parameters in $\boldsymbol{P}_l$ for 1 location loop [16];
    Pre-calculate all permutation parameters in $\boldsymbol{P}_e$ for $R_{est}$ estimation loops [16];
    $\boldsymbol{S}_l = \bigcup_{(\sigma,a,\beta) \in \boldsymbol{P}_l, n \in \mathrm{supp(G)}} \sigma(n - a + \beta) \bmod N$;
    $\boldsymbol{S}_e = \bigcup_{(\sigma,a) \in \boldsymbol{P}_e, n \in \mathrm{supp(G)}} \sigma(n - a) \bmod N$;
    //Parallel random sampling of signal
    $\boldsymbol{x}_{S_l} = x(\boldsymbol{S}_l/W)$;
    $\boldsymbol{x}_{S_e} = x(\boldsymbol{S}_e/W)$;
    //sFFT calculation
    //Location of Components
    $\boldsymbol{L} = \boldsymbol{LocateSignal}(\boldsymbol{x}_l, \boldsymbol{0}, B, \boldsymbol{P}_l, \boldsymbol{G}, \widehat{\boldsymbol{G}}', s, R_{loc})$;
        //Estimation of Components
    $\{\widehat{\boldsymbol{w}}, \boldsymbol{J}\} = \boldsymbol{EstimateValues}(\boldsymbol{x}_e, \boldsymbol{0}, B, \boldsymbol{P}_e, \boldsymbol{G}, \widehat{\boldsymbol{G}}', \boldsymbol{L}, 3K_r, R_{est})$;
    $\boldsymbol{X}_r = |\widehat{\boldsymbol{w}}_{\boldsymbol{J}}|_2^2$
    $\boldsymbol{X}_{rfca} = \boldsymbol{ConstantP_{fa}Recovery}(\boldsymbol{X}_r, \sigma_n, N, P_{FA})$;
    end
return $\boldsymbol{X}_{rfca}$
```

Source: Authors.

$$\boldsymbol{X}_{rcfa_j} = \begin{cases} H_1, & \boldsymbol{X}_{r_j} \geq \psi_j \\ H_0, & \text{otherwise} \end{cases} \tag{12}$$

Where, $\psi_j = \sigma_n^2 \left(\sqrt{\frac{2}{N^3}} Q^{-1}(P_{FA}) + 1/N \right)$ is the decision threshold.

3.1. Modified procedures

We modified the HashToBins, LocateSignal, LocateInner, and EstimateValues procedures described in [16] in order to use the proposed filtering window and to reduce the execution time.

3.1.1. HashToBins procedure

This procedure, presented in Alg. 2, calculates the hashes-error by subtracting the hashes of the instantaneous estimation $\hat{\boldsymbol{z}}$ from the hashes $\widehat{\boldsymbol{u}}$ [16], and it has the following input parameters: the time domain signal $\boldsymbol{x}$; the instantaneous estimation $\hat{\boldsymbol{z}}$ of $\hat{\boldsymbol{x}}$; the parameter B; the spectral permutation parameters σ, and a; and the vectors $\boldsymbol{G}, \widehat{\boldsymbol{G}}'$ of the filtering window in the time domain and the DFT domain respectively.

Algorithm 2. Modified Hash to bins function.

Input: $\boldsymbol{x} \in \mathbb{C}^N$, $\hat{\boldsymbol{z}} \in \mathbb{C}^N$, $B \in \{2^c | c \in [\log_2 N]\}$, $\sigma \in \left\{2c + 1 \middle| c \in \left[\frac{N}{2}\right]\right\}$, $a \in [N]$, $\boldsymbol{G} \in \mathbb{R}^{\boldsymbol{B}}$, $\widehat{\boldsymbol{G}}' \in \mathbb{R}^{\boldsymbol{13N/B}}$
Output: $\widehat{\boldsymbol{u}} \in \mathbb{C}^B$

```
    HashToBins procedure $(\boldsymbol{x}, \hat{\boldsymbol{z}}, B, \sigma, a, \boldsymbol{G}, \widehat{\boldsymbol{G}}')$
    $\boldsymbol{u}_j = 0 \;\forall\, j \in [B]$;
    //Spectral permutation and sub-sampling
    for $j \in \{N - |\mathrm{supp}(\boldsymbol{G})|_0/2,\ N + |\mathrm{supp}(\boldsymbol{G})|_0/2 - 1\}$ do
        $\boldsymbol{u}_{j \bmod B} = \boldsymbol{u}_{j \bmod B} + \boldsymbol{x}_{\sigma(j-a) \bmod N} \boldsymbol{G}_{j - N + |\mathrm{supp}(\boldsymbol{G})|_0/2}$;
    end
    //Sub-sampled DFT
    $\widehat{\boldsymbol{u}} = \mathrm{FFT}_B(\boldsymbol{u})$;
    //Efficient convolution calculation
    for $j \in supp(\hat{\boldsymbol{z}})$ do
        for $k \in [6]$ do
            $\widehat{\boldsymbol{u}}_{h_f(j,\sigma,N,B) - k \bmod B} = \widehat{\boldsymbol{u}}_{h_f(j,\sigma,N,B) - k \bmod B}$
                                    $- \widehat{\boldsymbol{G}}'_{|o_{fk}(j,\sigma,N,B)|} \hat{\boldsymbol{z}}_j \omega^{\sigma a j}$;
            $\widehat{\boldsymbol{u}}_{h_c(j,\sigma,N,B) - k \bmod B} = \widehat{\boldsymbol{u}}_{h_c(j,\sigma,N,B) - k \bmod B}$
                                    $- \widehat{\boldsymbol{G}}'_{|o_{ck}(j,\sigma,N,B)|} \hat{\boldsymbol{z}}_j \omega^{\sigma a j}$;
        end
    end
return $\widehat{\boldsymbol{u}}$
```

Source: Adapted from [16].

This procedure calculates the hashes-error using three processing stages: the first one simultaneously calculates in the time domain the pseudo-random spectral permutation, the windowing, and the hashing process; the second calculates the DFT domain hashes $\widehat{\boldsymbol{u}}$ by performing the B-point FFT of the time domain hashes $\boldsymbol{u}$; and the third one calculates the hashes-error by subtracting the DFT domain hashes of $\hat{\boldsymbol{z}}$ from the hashes $\widehat{\boldsymbol{u}}$, in this case there are 14 hashes for each sparse component in $\hat{\boldsymbol{z}}$.

3.1.2. LocateSignal procedure

This procedure, presented in Alg. 3, calculates the set $\boldsymbol{L} \in \mathbb{N}^{O(B)}$ of frequency bins corresponding to $O(B)$ sparse components found in $\boldsymbol{x}$; and it has the following input parameters: the time domain signal $\boldsymbol{x}$; the instantaneous estimation $\hat{\boldsymbol{z}}$ of $\hat{\boldsymbol{x}}$; the parameter B; the spectral permutation parameters $\boldsymbol{P}_l$; the vectors $\boldsymbol{G}, \widehat{\boldsymbol{G}}'$ of the filtering window in the time domain and the DFT domain respectively; the threshold constant for location s [16]; and the number of location iterations R_{loc} [16].

This procedure locates the sparse components of $\boldsymbol{x}$ using four processing stages: the first one sets the initial conditions, the second one adjusts the frequency locations, the third one reduces the search region, and the fourth one inverts the spectral permutation. The setting of initial conditions is performed by first calculating a reference hashes vector $\widehat{\boldsymbol{u}}$; second pre-calculating an initial guess of value $\boldsymbol{l}_j^{(0)} = jN/B \;\forall \sim j \in [B]$ of frequency locations in the permuted spectrum; and third by pre-calculating the initial values of w, t, and D_{max}, where w is the width of the region for searching the frequency adjustment, t is the number of candidate adjustments in w, D_{max} is the number of

Algorithm 3. Modified locate signal function.

```
Input: $\boldsymbol{x} \in \mathbb{C}^{O(K)},\ \hat{\boldsymbol{z}} \in \mathbb{C}^{N},\ B \in \{2^c | c \in [\log_2 N]\}, \boldsymbol{P}_l \in \mathbb{N}^{3\times O(K)}, \boldsymbol{G} \in \mathbb{R}^{\boldsymbol{B}}, \hat{\boldsymbol{G}}' \in \mathbb{R}^{\frac{13N}{B}},\ s \in \mathbb{R},\ R_{loc} \in \mathbb{N}$
Output: $\boldsymbol{L} \in \mathbb{N}^{O(B)}$
LocateSignal procedure $(\boldsymbol{x}, \hat{\boldsymbol{z}}, B, \boldsymbol{P}_l, \boldsymbol{G}, \hat{\boldsymbol{G}}', s, R_{loc})$
    Choose $a$ and $\sigma$ from $\boldsymbol{P}_l$;
    $\hat{\boldsymbol{u}}$ = HashToBins$(\boldsymbol{x}, \hat{\boldsymbol{z}}, B, \sigma, a, \boldsymbol{G}, \hat{\boldsymbol{G}}')$
    $\boldsymbol{l}_j^{(0)} = jN/B\ \forall \sim j \in [B]$;
    $w_0 = N/B$;
    $t = \log_2 N$;
    $t' = t/4$;
    $D_{max} = \lceil \log_{t'} (w_0 + 1) \rceil$;
    //Main loop
    for $D \in [D_{max}]$ do
        $\boldsymbol{l}^{(D+1)} = \boldsymbol{LocateInner}(\boldsymbol{x}, \hat{\boldsymbol{z}}, B, \sigma, a, \boldsymbol{P}_l, \boldsymbol{G}, \hat{\boldsymbol{G}}', s, R_{loc}, w_0 /(t')^D, t, \hat{\boldsymbol{u}}, \boldsymbol{l}^{(D)})$;
    end
    $\boldsymbol{L} = \left\{\sigma^{-1} \left\lfloor \boldsymbol{l}_j^{(D_{max})} + 0.5 \right\rfloor \middle| j \in [B] \right\}$;
return $\boldsymbol{L}$
```

Source: Adapted from [16].

adjustments, and R_{loc} is the number of location iterations for each adjustment. The adjustment of frequency locations is performed by using D_{max} times the procedure *LocateInner*. The reduction of the search region is performed by dividing w by a factor $1/(t')^D$ at the D-th adjustment, this reduction allows a systematic refining of the frequency location. Finally, the spectral permutation inversion is performed by calculating the function $\pi_p^{-1}(k, \sigma, N)$ on each located frequency.

3.1.3. LocateInner procedure

This procedure, presented in Alg. 4, calculates the adjustment $\boldsymbol{l}' \in \mathbb{N}^{O(B)}$ of the frequency locations in the permuted spectrum, and it has the following input parameters: the time domain signal $\boldsymbol{x}$; the instantaneous estimation $\hat{\boldsymbol{z}}$ of $\hat{\boldsymbol{x}}$; the parameter B; the spectral permutation parameters σ, a, and $\boldsymbol{P}_l$; the vectors $\boldsymbol{G}, \hat{\boldsymbol{G}}'$ of the filtering window in the time domain and the DFT domain respectively; the threshold constant for location s; the number of location iterations R_{loc}; the width of search region w; the number of candidate adjustments t; the reference hashes vector $\hat{\boldsymbol{u}}$; and the current estimation of frequency locations $\boldsymbol{l}$.

This procedure performs the adjustment of the frequency location using five processing stages: the first one sets the initial conditions, the second calculates the hashes-error, the third calculates the angles of the hashes-error and candidate frequency bins, the fourth performs the voting stage, and the fifth locates the frequency bins. The setting of initial conditions clears the vote counters of the t candidate adjustments for the B candidate frequency bins. The hashes calculation stage obtains R_{loc} hashes vectors $\hat{\boldsymbol{u}}'$, which are calculated from the signal $\boldsymbol{x}$ by using the *HashToBins* procedure with pseudo-random permutation parameters of the form $(\sigma, a + \beta)$. The angle calculation stage obtains the vector $\boldsymbol{c}$, which represents the angle differences between the reference hashes $\hat{\boldsymbol{u}}$ and the hashes $\hat{\boldsymbol{u}}'$. The voting stage increments the vote counter $\boldsymbol{v}_{j,q}$ corresponding

Algorithm 4. Modified locate signal inner function.

```
Input: $\boldsymbol{x} \in \mathbb{C}^{N},\ \hat{\boldsymbol{z}} \in \mathbb{C}^{N},\ B \in \{2^c | c \in [\log_2 N]\}, \sigma \in \left\{2c+1 \middle| c \in \left[\frac{N}{2}\right]\right\}, a \in [N],\ \boldsymbol{G} \in \mathbb{R}^{\boldsymbol{B}}, \hat{\boldsymbol{G}}' \in \mathbb{R}^{\frac{13N}{B}}, s \in \mathbb{R},\ R_{loc} \in \mathbb{N}^+, w \in \mathbb{R}, t \in \mathbb{N}^+,\ \hat{\boldsymbol{u}} \in \mathbb{C}^B, \boldsymbol{l} \in \mathbb{N}^{O(B)}$
Output: $\boldsymbol{l}' \in \mathbb{N}^{O(B)}$
LocateInner procedure $(\boldsymbol{x}, \hat{\boldsymbol{z}}, B, \sigma, a, \boldsymbol{P}_l, \boldsymbol{G}, \hat{\boldsymbol{G}}', s, R_{loc} w, t, \hat{\boldsymbol{u}}, \boldsymbol{l})$
    $\boldsymbol{v}_{j,q} = 0\ \forall\ (j,q) \in [B] \times [t]$;
    //Main loop
    for $i \in [R_{loc}]$ do
        Choose $\beta$ from $\boldsymbol{P}_l$;
        $\hat{\boldsymbol{u}}'$ = HashToBins$(\boldsymbol{x}, \hat{\boldsymbol{z}}, B, \sigma, a + \beta, \boldsymbol{G}, \hat{\boldsymbol{G}}')$;
        for $j \in [B]$ do
            if $\boldsymbol{l}_j \neq \perp$ then
                $r_j = \hat{\boldsymbol{u}}_j / \hat{\boldsymbol{u}}_j'$;
                $c_j = \text{arctan2}(\Im\{r_j\}, \Re\{r_j\})$;
                if $c_j < 0$ then
                    $c_j = c_j + 2\pi$;
                end
                for $q \in [t]$ do
                    $m_{j,q} = \boldsymbol{l}_j + \frac{q + 1/2}{t} w$;
                    $\theta_{j,q} = \frac{2\pi\beta m_{j,q}}{N} \bmod 2\pi$;
                    if $\min\{|\theta_{j,q} - c_j|, 2\pi - |\theta_{j,q} - c_j|\} < s\pi$ then
                        $\boldsymbol{v}_{j,q} = \boldsymbol{v}_{j,q} + 1$;
                    end
                end
            end
        end
    end

    for $j \in [B]$ do
        $\boldsymbol{Q} = \{q \in [t]\ |\boldsymbol{v}_{j,q} > R_{loc}/2\}$;
        if $\boldsymbol{Q} \neq \emptyset$ then
            $\boldsymbol{l}_j' = \boldsymbol{l}_j + \min_{q \in \boldsymbol{Q}} qw/t$;
        else
            $\boldsymbol{l}_j' = \perp$;
        end
    end
return $\boldsymbol{l}'$
```

Source: Adapted from [16].

to the j-th frequency bin and q-th adjustment, if Eq. (13) is satisfied.

$$\min\{|\theta_{j,q} - c_j|, 2\pi - |\theta_{j,q} - c_j|\} < s\pi \tag{13}$$

Where $\theta_{j,q}$ is the angle of the j-th candidate frequency bin for the q-th adjustment, that is, $\theta_{j,q}$ is related to the q-th candidate frequency adjustment qw/t. Additionally, the above calculation converges if β is chosen at random from the set $\{\lfloor sNt/4w \rfloor, \dots, \lfloor sNt/2w \rfloor\}$ with enough small threshold s. Finally, the frequency location stage selects the minimum q from the set $\boldsymbol{Q} = \{q \in [t]\ |v_{j,q} > R_{loc}/2\}$; thus the estimated j-th permuted-frequency bin is refined using Eq. (14).

$$\boldsymbol{l}_j' = \boldsymbol{l}_j + \min_{q \in \boldsymbol{Q}} qw/t \tag{14}$$

Algorithm 5. Modified estimate values function.

```
Input: $\boldsymbol{x} \in \mathbb{C}^N,\ \hat{\boldsymbol{z}} \in \mathbb{C}^N,\ B \in \{2^c | c \in [\log_2 N]\}, \boldsymbol{P}_e \in \mathbb{N}^{2\times O(K)}, \boldsymbol{G} \in \mathbb{R}^{\boldsymbol{B}}, \widehat{\boldsymbol{G}}' \in \mathbb{R}^{\frac{13N}{B}}, \boldsymbol{L} \in \mathbb{N}^{O(B)}, K' \in \mathbb{N}^+, R_{est} \in \mathbb{N}^+$
Output: $\boldsymbol{L} \in \mathbb{N}^{O(B)}$
EstimateValues procedure $(\boldsymbol{x}, \hat{\boldsymbol{z}}, B, \boldsymbol{G}, \widehat{\boldsymbol{G}}', \boldsymbol{L}, K', R_{est})$
    $\widehat{\boldsymbol{w}}_j = 0\ \forall\, j \in [|\boldsymbol{L}|_0]$;
    //Main loop
    for $i \in [R_{est}]$ do
        Choose $\sigma_i$ and $a_i$ from $\boldsymbol{P}_e$;
        $\widehat{\boldsymbol{u}}$ = HashToBins$(\boldsymbol{x}, \hat{\boldsymbol{z}}, B, \sigma_i, a_i, \boldsymbol{G}, \widehat{\boldsymbol{G}}')$;
        for $j \in [|\boldsymbol{L}|_0]$ do
            $\widehat{\boldsymbol{u}}'_{i,j} = \widehat{\boldsymbol{u}}_{h(\boldsymbol{L}_j,\sigma_i,N,B) \bmod \boldsymbol{B}}\, \omega^{-\sigma_i a_i \boldsymbol{L}_j} / \widehat{\boldsymbol{G}}'_{|o(\boldsymbol{L}_j,\sigma_i,N,B)|}$;
        end
    end
    for $j \in [|\boldsymbol{L}|_0]$ do
        // Median for real and imaginary parts separately across the i dimension
        $\widehat{\boldsymbol{w}}_j = \mathrm{median}_i\, \widehat{\boldsymbol{u}}'_{i,j}$
    end
    $\boldsymbol{J} = \arg\max_{|\boldsymbol{J}|_0 = \min\{|\boldsymbol{L}|_0, K'\}} |\widehat{\boldsymbol{w}}_{\boldsymbol{J}}|_2$

return $\{\widehat{\boldsymbol{w}}_{\boldsymbol{J}}, \boldsymbol{J}\}$
```

Source: Adapted from [16].

3.1.4. EstimateValues procedure

This procedure, presented in Alg. 5, calculates the DFT estimation adjustment $\widehat{\boldsymbol{w}}_{\boldsymbol{J}}$, and it has the following input parameters: the time domain signal $\boldsymbol{x}$; the instantaneous estimation $\hat{\boldsymbol{z}}$ of $\hat{\boldsymbol{x}}$; the parameter B; the spectral permutation parameters $\boldsymbol{P}_e$; the vectors $\boldsymbol{G}, \widehat{\boldsymbol{G}}'$ of the filtering window in the time domain and the DFT domain respectively; the set of located sparse components $\boldsymbol{L}$; the number of sparse components to estimate K'; and the number of estimation iterations R_{est} [16]. This procedure calculates the DFT estimation adjustment $\widehat{\boldsymbol{w}}_{\boldsymbol{J}}$ using three processing stages: the first one calculates the R_{est} sets of hashes-error from the signal $\boldsymbol{x}$ by using the HashToBins procedure and considering different pseudo-random permutation parameters; the second one separately calculates the median of the real and imaginary parts of the calculated hashes-error by only considering the set of located sparse frequency bins in $\boldsymbol{L}$, and by cancelling the pseudo-random spectral permutation and the effect of the windowing in the DFT domain; and the third one saves the K' most energetic components $\widehat{\boldsymbol{w}}_{\boldsymbol{J}}$.

4. Simulation results of SNSparseWSS algorithm

This section presents the simulation results of the *SNSparseWSS* algorithm for sub-Nyquist sampling and verification results for Wideband Spectrum Sensing.

4.1. Sub-Nyquist capabilities

In order to verify the sampling cost, we need to know all the spectral permutation parameters σ_i and a_i that are pseudo-randomly generated by the SNSparseWSS algorithm; thus, the set of sampling points can be calculated using Eq. (15).

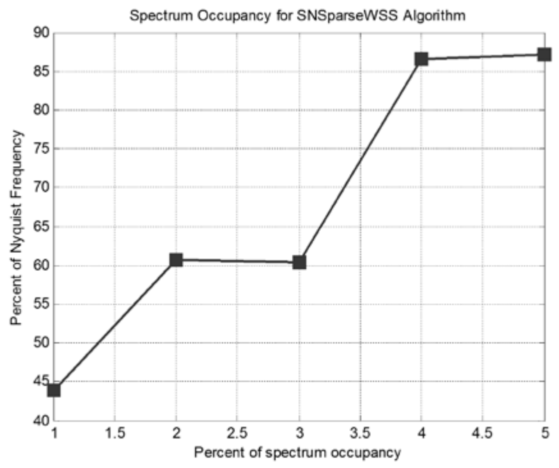


Figure 2. Percent of Nyquist Sampling Rate versus Spectrum occupancy percent for *SNSparseWSS* algorithm.
Source: Authors.

$$\boldsymbol{S} = \bigcup_{n \,\in\, \mathrm{supp}(\boldsymbol{G})} \{(\sigma_i\,(n - a_i)) \bmod N\} \tag{15}$$

Considering the set of sampling points $\boldsymbol{S}$, it is possible to calculate the average sampling rate f_{sa} of the *SNSparseWSS* algorithm using Eq. (16).

$$f_{sa} = f_{Nyq} \frac{|\boldsymbol{S}|_0}{\sum_{k \in \mathrm{supp}(\boldsymbol{S})} \nabla \boldsymbol{S}_k} \tag{16}$$

Where, f_{Nyq} is the Nyquist frequency, and $\nabla \boldsymbol{S}_k = \boldsymbol{S}_k - \boldsymbol{S}_{k-1}$ is a finite-backward difference to estimate the average separation between samples. Fig. 2 shows the percentage of Nyquist frequency versus the percentage of spectrum occupancy for the *SNSparseWSS* algorithm.

From Fig. 2, we can see that the algorithm reaches a sampling rate of close to $0.6 \times f_{Nyq}$ for a spectrum occupancy between 2% and 3%; thus, for typical scenarios where the spectrum occupancy is close to 2% the *SNSparseWSS* algorithm is very suitable for implementing sub-Nyquist WSS systems and its performance is comparable to the WSS systems based on MC sampling [11].

4.2. Verification of the SNSparseWSS algorithm

In order to verify the WSS capabilities of the SNSparseWSS algorithm, we considered as test vehicle a highly-noisy multiband-signal scenario composed of 5 time domain signals located in the center frequencies 317-576-1300-984-163 MHz, and a total bandwidth W of 1.5 GHz. Each signal has a bandwidth of 5 MHz, is composed of random 4-QAM symbols that are filtered using a raised cosine filter with roll-off factor r = 0.5, and has SNR values from -5 dB to 5 dB. The test signal with SNR=-5 dB is shown in Fig. 3.

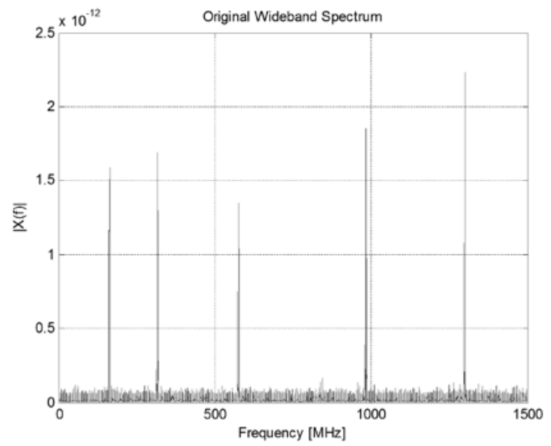


Figure 3. Wideband spectrum of 1.5 GHz containing 5 signals each with bandwidth $\boldsymbol{BW}$ = 5 MHz and SNR = -5 dB.
Source: Authors.

In this case, the duration of the spectrum sensing window is $\tau = 160\ \mu s$ which implies that an FFT with $N = 2^{17}$ must be used, and the total sparse bandwidth is 25 MHz (5 signals $\times$ 5 MHz) which leads to a sub-sampled FFT with $B = 8192$. The algorithm was parameterized with $R_{est} = 10$ estimation iterations and a location threshold of $s = 0.1$, with these settings the *SNSparseWSS* algorithm achieves an average sampling rate of $f_{sa} = 0.599 \times f_{Nyq}$.

Fig. 4 shows the recovered spectrum using the *SNSparseWSS* algorithm. In this figure, we can see that the Gaussian small support window increases the total error of estimated DFT value; this issue can be mitigated using appropriate settings for the constant P_{FA} detector, which implies that the occupied channels can be detected with constant P_{FA} [11].

Fig. 5 shows the WSS simulation results using the constant P_{FA} detector with $P_{FA} = 0.01$. In this figure, we can see that the algorithm can perform WSS by detecting the occupied channels by the Pus. If additional information about the multiband-signal is available, such as the minimum channel separation $\varDelta_f$ [11], the detection can be improved by reducing the P_{FA}.

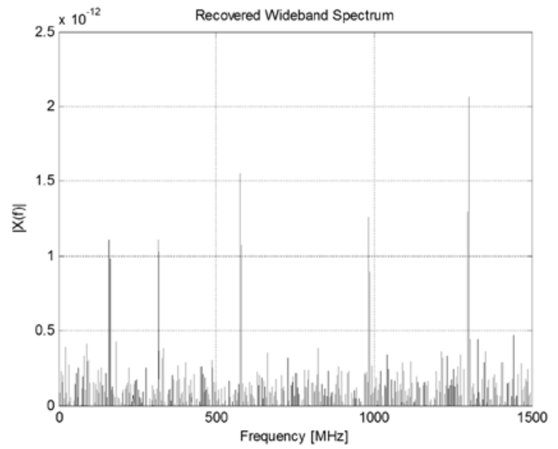


Figure 4. Recovered spectrum for $\tau = 160\ \mu s\ (N = 2^{17})\ B = 8192$, , 1 location iteration, 10 estimation iterations, and Gaussian window with $|\text{supp}(\boldsymbol{G})|_0 = B$.
Source: Authors.

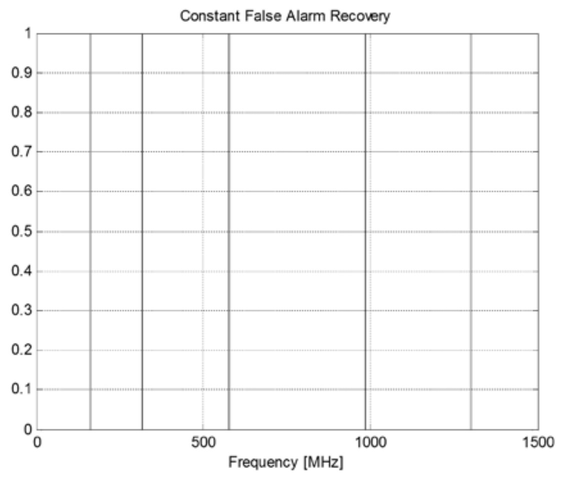


Figure 5. Wideband spectrum sensing result by using the constant false alarm probability detector.
Source: Authors.

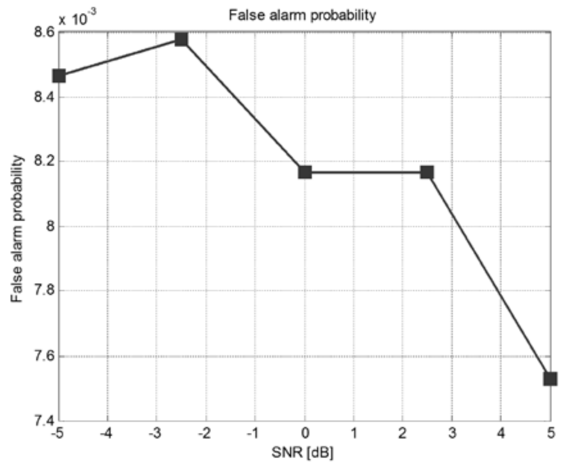


Figure 6. False alarm probability performance for different SNRs.
Source: Authors.

Fig. 6 shows simulation results for the false alarm probability with SNR values of -5 dB, -2.5 dB, 0 dB, 2.5 dB, and 5 dB. In this figure we can see that the P_{FA} is approximately constant regardless the SNR of the multiband-signals.

5. Conclusions and future work

In this paper, we present the design of a novel algorithm for sub-Nyquist Wideband Spectrum Sensing based on a modified Nearly Optimal sFFT algorithm. This WSS algorithm was verified using several tests. From the verification results we can conclude that the proposed algorithm is suitable for implementing the spectrum sensing function of wideband cognitive radios in highly-noisy environments. To the best of our knowledge, the proposed WSS algorithm is the first that uses the new Nearly Optimal Sparse Fourier Transform algorithm, and it has a reduced sampling cost by using flat Gaussian small support windows and modified procedures.

Future work will addressed efficient hardware implementation of the modified Nearly Optimal sFFT algorithm using an FPGA.

Acknowledgements

Alexander López Parrado thanks Colciencias for the scholarship, and he also thanks *Universidad del Quindío* for the study commission.

References

[1] Mitola, J. and Maguire, G.Q., Cognitive radio: Making software radios more personal. IEEE Personal Communications, 6(4), pp. 13-18, 1999. DOI: 10.1109/98.788210.
[2] Arslan, H., Cognitive radio, software defined radio, and adaptive wireless systems. Dordrecht: Springer, 2007. DOI: 10.1007/978-1-4020-5542-3.
[3] IEEE. IEEE 802.22-2011, wireless regional area networks (wran) – specific requirements part 22: Cognitive wireless ran medium access control (mac) and physical layer (phy) specifications: Policies and procedures for operation in the tv bands. 2011. DOI: 10.1109/IEEESTD.2011.5951707.
[4] M.T.P. Group, Microsoft spectrum observatory, [online], Seattle, [Date of reference, Nov. 2013.]. Available at: http://spectrum-observatory.cloudapp.net/.
[5] Vito, L.D., A review of wideband spectrum sensing methods for cognitive radios, Proceedings of Instrumentation and Measurement Technology Conference (I2MTC), pp. 2257-2262, 2012. DOI: 10.1109/I2MTC.2012.6229530.
[6] Hongjian, S., Nallanathan, A., Wang, C.X. and Chen, Y., Wideband spectrum sensing for cognitive radio networks: a survey. IEEE Wireless Communications, 20(2), pp. 74-81, 2013. DOI: 10.1109/MWC.2013.6507397.
[7] Axell, E., Leus, G., Larsson, E.G. and Poor, H.V., Spectrum sensing for cognitive radio: State-of-the-art and recent advances. IEEE Signal Processing Magazine, 29(3), pp. 101-116, 2012. DOI: 10.1109/MSP.2012.2183771.
[8] Laska, J., Kirolos, S., Duarte, M., Ragheb, T., Baraniuk, R. and Massoud, Y., Theory and implementation of an analog-to-information converter using random demodulation, Proceedings of IEEE International Symposium on Circuits and Systems, pp. 1959-1962, 2007. DOI: 10.1109/ISCAS.2007.378360.
[9] Sun, H., Chiu, W.Y., Jiang, J., Nallanathan, A. and Poor, H.V., Wideband spectrum sensing with Sub-Nyquist sampling in cognitive radios. IEEE Transactions on Signal Processing, 60(11), pp. 6068-6073, 2012. DOI: 10.1109/TSP.2012.2212892.
[10] Mishali, M. and Eldar, Y.C., From theory to practice: Sub-nyquist sampling of sparse wideband analog signals. IEEE Journal of Selected Topics in Signal Processing, 4(2), pp. 375-391, 2010. DOI: 10.1109/JSTSP.2010.2042414.
[11] Yen, C.-P., Tsai, Y. and Wang, X., Wideband spectrum sensing based on Sub-Nyquist sampling. IEEE Transactions on Signal Processing, 61(12), pp. 3028-3040, 2013. DOI: 10.1109/TSP.2013.2251342.
[12] Tsui, J.B., Digital techniques for wideband receivers, 2nd Edition. Raleigh: Scitech, 2004. DOI: 10.1049/SBRA005E.
[13] Donoho, D.L., Compressed sensing. IEEE Transactions on Information Theory, 52(4), pp. 1289-1306, 2006. DOI: 10.1109/TIT.2006.871582.
[14] Lobato-Polo, A.P., Ruiz-Coral, R.H., Quiroga-Sepúlveda, J.A. and Recio-Vélez, A.L., Sparse signal recovery using orthogonal matching pursuit (OMP). Ingeniería e Investigación, 29(2), pp. 112-118, 2009.
[15] Hassanieh, H., Indyk, P., Katabi, D. and Price, E., Simple and practical algorithm for sparse fourier transform, Proceedings of ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 1183-1194, 2012. DOI: 10.1137/1.9781611973099.93.
[16] Hassanieh, H., Indyk, P., Katabi, D. and Price, E., Nearly optimal sparse fourier transform, Proceedings of the 44th symposium on Theory of Computing (STOC), pp. 563-578, 2012. DOI: 10.1145/2213977.2214029.
[17] Hassanieh, H., Shi, L., Abari, O., Hamed, E. and Katabi, D., Ghz-wide sensing and decoding using the sparse fourier transform, Proceedings of IEEE INFOCOM, pp. 2256-2264, 2014. DOI: 10.1109/INFOCOM.2014.6848169.
[18] Indyk , P., Kapralov, M. and Price, E., (Nearly) Sample-optimal sparse fourier transform, Proceedings of ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 480-499, 2014. DOI: 10.1137/1.9781611973402.36.
[19] Gilbert, A.C., Strauss, M.J. and Tropp, J.A., A tutorial on fast fourier sampling. IEEE Signal Processing Magazine, 25(2), pp. 57-66, 2008. DOI: 10.1109/MSP.2007.915000.
[20] Gilbert, A.C., Muthukrishnan, S. and Strauss, M.J., Improved time bounds for near-optimal sparse fourier representations, Proceedings of SPIE Wavelets XI, pp. 1-15, 2005. DOI: 10.1117/12.615931.
[21] Dutt, A. and Rokhlin, V., Fast fourier transforms for nonequispaced data, ii. Applied and Computational Harmonic Analysis, 2(1), pp. 85-100, 1995. DOI: 10.1006/acha.1995.1007.
[22] Tanton, J., Encyclopedia of Mathematics. New York: Facts on File, 2005.
[23] Abramowitz, M. and Stegun, I., Handbook of mathematical functions with formulas, graphs, and mathematical tables, 10th printing. Washington, D.C.: Dover, 1972. DOI: 10.1063/1.3047921.
[24] Gilbert, A.C., Li, Y., Porat, E. and Strauss, M.J., Approximate sparse recovery: Optimizing time and measurements, Proceedings of the 42nd ACM symposium on Theory of computing, pp. 475-484, 2010. DOI: 10.1145/1806689.1806755.

A. López-Parrado, completed his BSc. Eng. in Electronics Engineering in 2002 at Universidad del Quindío, Armenia, Colombia, and his MSc. degree in Electronics in 2009 at Universidad del Valle, Cali, Colombia. He is currently a PhD. candidate in Electrical and Electronics Engineering at Universidad del Valle, Cali, Colombia. His research interests are the FPGA design of complex digital systems, design of baseband processors, DSP complex functions, embedded systems, and compressive sensing. He is an assistant professor in the Electronics Engineering Program at Universidad del Quindío, Colombia.
ORCID: 0000-0002-0274-6901

J. Velasco-Medina, completed his BSc. Eng. in Electrical Engineering in 1985 at Universidad del Valle, Cali, Colombia, his MSc. degree in Microelectronics 1995 at Universite De Grenoble I (Scientifique Et Medicale - Joseph Fourier), and his PhD. degree in Microelectronics in 1999 at Institut National Polytechnique De Grenoble, France. His research interests are the FPGA design of complex digital systems, design of baseband processors, Cryptosystems, DSP complex functions, DNA processor, bionanomachines, bionanosensors, biological systems modeling, and Citocomputation. He is a titular professor in the Electrical and Electronics Engineering School of Universidad del Valle, Cali, Colombia.
ORCID: 0000-0003-4091-1055