Cubic B–spline fuzzy transforms for an efficient and secure compression in wireless sensor networks

Matteo Gaeta[a], Vincenzo Loia[b,c], Stefania Tomasiello[a,c]

[a] *Dipartimento di Ingegneria dell'Informazione ed Elettrica e Matematica applicata (DIEM), Università degli Studi di Salerno, via Giovanni Paolo II, 132, Fisciano 84084, Italy*
[b] *Dipartimento di Scienze Aziendali - Management & Innovation Systems (DISA-MIS), Università degli Studi di Salerno, via Giovanni Paolo II, 132, Fisciano 84084, Italy*
[c] *Consorzio di Ricerca Sistemi ad Agenti (CORISA), Università degli Studi di Salerno, via Giovanni Paolo II, 132, Fisciano 84084, Italy*

Abstract

Joining data compression and encryption is a way to keep secure data, as discussed by the current literature. While data compression responds to the great demand on data storage and transmission techniques, the encryption allows to handle some important parameters in a secure way. In wireless sensor networks the usual transform–based compression is the Discrete Wavelet Transform. In a previous paper we showed the good perfomance of the fuzzy transform (or F–trasform for short) based compression with respect to it. In this work, we propose a cubic B–spline F–transform in order to have a higher accuracy, even when data are not correlated, and a lower computational cost. Besides, in order to show the efficiency of the proposed approach, we compare it with the most recent lossless compression scheme in the field. We discuss these issues formally and numerically by using publicly available real–world data sets. The parameters required to decompress data are encrypted by means of a suitable existing encryption algorithm. We show that even if an illegal user had access to one of these parameters, our scheme would be still secure.

Email addresses: mgaeta@unisa.it (Matteo Gaeta), loia@unisa.it (Vincenzo Loia), stomasiello@unisa.it (Stefania Tomasiello)

December 30, 2015

The published version of this manuscript is available at https://doi.org/10.1016/j.ins.2015.12.026

Keywords: Data compression, Wireless Sensor Networks, F–transform, Least–squares, encryption

1. Introduction

The aim of data compression is to reduce the memory space or the transmission time, especially for wireless sensor networks (WSNs) because of the energy constraints. In the past, data compression and cryptography were kept separated because any data can be compressed if necessary and then encrypted. Anyway because of the rapid progress in computing technology, the encrypted data could be secure no longer in a few years. A way to increase security is joining compression and encryption, using one of the existing cryptography techniques. This scheme has been adopted especially for images. Keat et al. used a wavelet based encoder with an RC4 encryption algorithm [1]: the authors encrypted some important parameters for recovering the image, such as initial threshold, scan order, size of the image.

In [2] a Quadtree image compression was used, by dividing the image into two parts, so that only the tree structure was encrypted by means of a public–key algorithm such as RSA.

In [3] the image is first compressed and then encrypted by rearranging the bits of the compressed image by means of a set of scanning paths; this set of scanning paths is kept secret and it is the encryption key.

In [4] the authors proposed to embed k–PCA into a compression–encryption scheme. After having compressed the input image, they encrypted the principal components and other three parameters, necessary for recovering the original image, by means of the RC4 symmetric cipher.

In the case of WSNs, since sensors have both limited memory and storage space and power limitations, the most part of the traditional techniques turns out to be not suitable, by requiring a certain amount of resources such as data memory, code space and energy. This is principally due to the fact that such techniques use asymmetric cryptography, where there is a public key to encrypt data and a private key to decrypt them. Asymmetric cryptography is computationally expensive for the individual nodes in a sensor network, even if some authors [5], [6], [7] showed that it is feasible by choosing the right algorithms. So symmetric cryptography is the typical choice when the computational complexity of asymmetric cryptography cannot be afforded. Symmetric schemes utilize a single shared key known only between the two

communicating hosts, which is used for both encrypting and decrypting data. Typical symmetric schemes are RC5 and AES [8].

With regard to the compression techniques avilable in WSNs, one can refer to [9] for a comprehensive survey. In general, nodes, which are able to collect, to process data and sharing them with neighboring nodes, are required to be relatively inexpensive, in terms of power supply, memory capacity, communication bandwidth, and processor performance [10]. Since mostly the energy consumption is due to radio communication [11], compression techniques allow lesser communication energy costs. A well–known approach in the WSN field is the discrete wavelet transform (DWT), which performs well for spatially– and temporally–correlated data, but this could not be true for outdoor environments [9]. In a previous work [12], we showed the good performance of an F–transform based approach when compared to the usual DWT approach, by finding out a high enough value of the compression rate with a lower distortion.

F–transform was proposed by Perfilieva [13] as a fuzzy approximation technique. It substantially expresses a functional dependency as a linear combination of basic functions and it can be used for the solution of direct and inverse problems or least–squares approximations [14]. The major applications of the F–transform are in image processing, e.g. [15]–[21].

In this paper, we propose cubic B–splines fuzzy transform in order to have a lower distortion in data compression, with a lower computational cost with respect to the existing transform–based compression approaches for WSNs. We discuss formally accuracy and computational cost, by also showing the compression performance numerically on publicly available real–world data sets.

It should be pointed out that recently some lossless compression schemes for WSNs was proposed [22] , [23]. In particular, in [22] an extension of the predictive coding–based scheme LEC, called S–LEC, is proposed to provide better results with respect to LEC and the dictionary–based scheme S–LZW. In [23], a lightweight block–based lossless adaptive compression scheme, called FELACS, is proposed with good performances with respect to LEC and S–LZW.

Unlike lossless compression schemes, transform–based approaches have the shortcoming of a certain distortion (i.e. approximation) in the reconstructed data. In this case, a loss of information may happen, but generally a higher compression ratio is achievable [9].

However, in order to show the good performance of our approach we

provide a comparison with the likely best lossless scheme between [22] and [23], discussing distortion and compression ratio.

Our approach turns out to be also suitable for data security, by integrating it with an existing encryption algorithm, such as RC4, which is fast and secure for WSNs under certain conditions [24]. We use such a algorithm to keep secure some parameters needed to decompress data. As it will be shown, even if one parameter were known, trying to reconstrut data would involve a considerable distortion.

The paper is structured as follows: Section 2 provides theoretical foundations, discussing formally accuracy and computational cost; in Section 3 the compression performance is assessed by means of numerical experiments; Section 4 introduces the compression–encryption scheme and finally Section 5 gives some conclusions.

2. Data compression based on F–transform

The F–transform changes a functional problem into a problem of linear algebra, by computing the approximate solution to the problem by means of the inverse F–transform. The same ideas hold on for the discrete problems via the discrete F–transform and the inverse discrete F–transform. Since F–transform was introduced [13], several papers on the topic appeared [25]–[30]. In particular, in [30] new types of F–transforms were presented, based on B–splines, Shepard kernels, Bernstein basis polynomials and Favard-Szasz-Mirakjan type operators for the univariate case.

There are many applications of the F–transforms in image processing ([15]– [21]) and some others in time series analysis [31]–[35], also by integrating the F–transform and the fuzzy tendency modeling [32] or the F–transform and fuzzy natural logic [33]. In particular, the paper by Novak et al. [35] focuses on application of fuzzy transform (F-transform) to the analysis of time series under the assumption that the latter can be additively decomposed into trend-cycle, seasonal component and noise.

F–transforms were also used in data analysis [36],[37].

In [29] F–transforms combined with finite differences were used to numerically solve some classical partial differential equations (heat, wave, Poisson) in simple domains.

In [14] the relations between the least–squares approximation techniques and the F–transform for the univariate case were investigated.

2.1. Preliminaries

We briefly recall some definitions. Let $I = [a, b]$ be a closed interval and $x_1, x_2, \ldots, x_n$, with $n \geq 3$, be points of I, called nodes, such that $a = x_1 < x_2 < \ldots < x_n = b$. A fuzzy partition of I is defined as a sequence $A_1, A_2, \ldots, A_n$ of fuzzy sets $A_i : I \to [0, 1]$, with $i = 1, \ldots, n$ such that

– $A_i(x) \neq 0$ if $x \epsilon (x_{i-1}, x_{i+1})$ and $A_i(x_i) = 1$;

– A_i is continuous and has its unique maximum at x_i;

–$\sum_{i=1}^{n} A_i(x) = 1, \quad \forall x \epsilon I$.

The fuzzy sets $A_1, A_2, \ldots, A_n$ are called basic functions and they form an uniform fuzzy partition if the nodes are equidistant.

In general, $h = max|x_{i+1} - x_i|$ is the norm of the partition. For a uniform partition $h = (b - a)/n$ and $x_j = a + jh$.

A fuzzy partition with small support has the additional property that there exists $r \geq 1$ such that $suppA_i = cl\{x \epsilon I : Ai(x) > 0\} \subseteq [x_i, x_{i+r}]$, where cl stands for closure.

The fuzzy partition can be obtained by means of several basic functions:

- hat functions

$$A_j(x) = \begin{cases} (x_{j+1} - x)/(x_{j+1} - x_j), & x \epsilon [x_j, x_{j+1}] \\ (x - x_{j-1})/(x_j - x_{j-1}), & x \epsilon [x_{j-1}, x_j] \\ 0, & otherwise \end{cases} \quad (1)$$

- sinusoidal shaped basic functions

$$A_j(x) = \begin{cases} \frac{1}{2} \cos\left(\frac{\pi(x_j - x)}{(x_{j+1} - x)}\right) + 1, & x \epsilon [x_j, x_{j+1}] \\ \frac{1}{2} \cos\left(\frac{\pi(x - x_j)}{(x_j - x_{j-1}) + 1}\right), & x \epsilon [x_{j-1}, x_j] \\ 0, & otherwise \end{cases} \quad (2)$$

- cubic B–splines, for $j = 0, \ldots, n$ (here in explicit form as in [38],[39])

$$A_j(x) = \frac{1}{h^3} \begin{cases} (x - x_{j-2})^3, & x \epsilon [x_{j-2}, x_{j-1}) \\ (x - x_{j-2})^3 - 4(x - x_{j-1})^3, & x \epsilon [x_{j-1}, x_j) \\ (x_{j+2} - x)^3 - 4(x_{j+1} - x)^3, & x \epsilon [x_j, x_{j+1}) \\ (x_{j+2} - x)^3, & x \epsilon [x_{j+1}, x_{j+2}) \\ 0, & otherwise \end{cases} \quad (3)$$

It should be pointed out that in order to apply B–splines some auxiliary points are needed: for cubic B–splines two auxiliary points both on the left and on the right of the considered interval are required.

The fuzzy transform (F–transform) of a function $f(x)$ continuous on I with respect to $\{A_1, A_2, \ldots, A_n\}$ is the n–tuple $[F_1, F_2, \ldots, F_n]$ whose components are

$$F_i = \frac{\int_a^b f(x)A_i(x)dx}{\int_a^b A_i(x)dx} \tag{4}$$

The function

$$f_{F,n} = \sum_i^n F_i A_i(x), \qquad x\epsilon I \tag{5}$$

is called inverse F–transform of f with respect to $\{A_1, A_2, \ldots, A_n\}$ and it approximates a given continuous function f on I with arbitrary precision, as stated by Theorem 2 in [13].

In many real cases, where the function f is known only at a given set of points $\{p_1, p_2, \ldots, p_m\}$, the discrete F–transform can be used and Eq. (1) is replaced by

$$F_i = \frac{\sum_{j=1}^m f(p_j)A_i(p_j)}{\sum_{j=1}^m A_i(p_j)}, \qquad i = 1, \ldots, n \tag{6}$$

Similarly, Eq. (2) is replaced by

$$f_{F,n}(p_j) = \sum_i^n F_i A_i(p_j), \qquad j = 1, \ldots, m \tag{7}$$

giving the discrete inverse F–transform.

The above concepts can be extended to functions in two variables, as one will see in the next subsection.

2.2. Least–squares approximation, properties and theorems for the bivariate case

Let us consider the $N \times M$ data matrix $\mathbf{D}$ and the fuzzy partitions $\{A_1, \ldots, A_n\}$ and $\{B_1, \ldots, B_m\}$, with $n < N$ and $m < M$.

In the least squares approximation, the discrete F–transform of $\mathbf{D}$ with respect to $\{A_1, \ldots, A_n\}$ and $\{B_1, \ldots, B_m\}$ are unknowns λ_{ij} to be obtained by means of the error functional E

$$\mathbf{E} = \mathbf{D} - \mathbf{A}\mathbf{\Lambda}\mathbf{B}^T \tag{8}$$

i.e. by minimizing it with respect to the λ_{ij}, we get

$$\mathbf{\Lambda} = \mathbf{K}^{-1}\mathbf{G}\mathbf{H}^{-1} \tag{9}$$

where

$$\mathbf{G} = \mathbf{A}^T\mathbf{D}\mathbf{B} \tag{10}$$

$$\mathbf{K} = \mathbf{A}^T\mathbf{A}, \qquad \mathbf{H} = \mathbf{B}^T\mathbf{B} \tag{11}$$

The discrete inverse F–transform is given by:

$$\mathbf{D^F} = \mathbf{A}\mathbf{\Lambda}\mathbf{B}^T \tag{12}$$

It should be pointed out that, since $\mathbf{A}$ and $\mathbf{B}$ are the Gram matrices associated to given sets of basis functions, they have full rank and $\mathbf{K}$ and $\mathbf{H}$ turn out to be positive definite matrices.

With regard to the B–spline based case, we extend some results presented in [30].

By following [40] and [41], we introduce the modulus of smoothness for a bivariate function $f : \mathbb{R}^2_+ \to \mathbb{R}$

$$\omega(f,\alpha,\beta) = \sup\{|f(u,v) - f(x,y)| : (u,v),(x,y)\epsilon\mathbb{R}^2_+, |u-x| \leq \alpha, |v-y|\beta\} \tag{13}$$

Let us recall the following properties for $\omega(f,\alpha,\beta)$ [42]:

(i) $\omega(f,0,0)$=0 and $\omega(f,\alpha,\beta)$ is nondecreasing with respect to α and β;

(ii) $\omega(f,\alpha_1+\alpha_2,\beta_1+\beta_2) = \omega(f,\alpha_1,\beta_1) + \omega(f,\alpha_2,\beta_2)$.

In particular, the last property reduces to $\omega(f,2\alpha,2\beta) = 2\omega(f,\alpha,\beta)$, if $\alpha_1 = \alpha_2$ and $\beta_1 = \beta_2$, so it is easy to generalize:

$$\omega(f,r\alpha,r\beta) = r\omega(f,\alpha,\beta) \tag{14}$$

with $r \geq 2$. Let $f^F_{nm}(x,y)$ be the composition of the inverse and direct F–transform. We prove the following theorem.

Theorem 1. *Let $f(x,y)$ be a function assigned over the set $[a,b]\times[c,d]$. Let $\{x_1,\ldots,x_n\}\epsilon[a,b]$ and $\{y_1,\ldots,y_m\}\epsilon[c,d]$ be the partitions, with norms α and β, of the intervals $[a,b]$ and $[c,d]$ respectively. If $A_1,\ldots,A_n$ and $B_1,\ldots,B_m$*

are fuzzy partitions with small support, with regard to the same integer r, then the following inequality holds:

$$|f_{nm}^F(x,y) - f(x,y)| \leq r\omega(f,\alpha,\beta) \tag{15}$$

Proof. The proof follows the one of Theorem 3.3 in [30], by considering the bivariate smoothness modulus and Eq. (14). ■

The theorem above shows that, if $A_1,\ldots,A_n$ and $B_1,\ldots,B_m$ are generated by means of B–splines with order $r-1$, we get a good approximation for $n,m >> r$.

Because of the local nature of B–splines, a related square matrix is a band matrix with only r nonzero elements in each row [43].

It is well–known that a band matrix has nonzero entries only through a band along the main diagonal and this is important with regard to the matrix inversion and in general for the computational complexity.

Remark 2. *If the matrices* $\mathbf{A}$ *and* $\mathbf{B}$ *are pseudo–banded matrices, i.e. non–square matrices which exhibit a band–like structure, then the matrices* $\mathbf{K}$ *and* $\mathbf{H}$ *are symmetric band matrices.*

For a symmetric banded matrix of order n the computational cost of the inversion can be reduced to $O(n)$ by using a simple algorithm as shown in [44].

Remark 3. *If the matrices* $\mathbf{A}$ *and* $\mathbf{B}$ *are band matrices, the computational complexity of the LS approach is* $O(nm)$.

As a comparison, we point out that the computational complexity of the (one level) DWT is bounded by $O(NM)$ [9], which is clearly higher.

2.3. A simple example

The proposed F–transform based compression is schematically depicted in Figure 1. In order to give a clear explaination on how the proposed scheme can be used for data compression, we present a simple example.

We generated an 11×5 data matrix $\mathbf{D}$ by the function $sinh(j)/(ij)$, with $i = 1, 1.5, \ldots, 6$ and $j = 1, 1.5, \ldots, 3$. In this example, we considered the F–transform compression rate $\rho = (nm)/(NM) = 0.58$. By means of cubic B-splines we obtain $MSE = 6.25E-05$, whereas by means of sinusoidal shaped and hat functions we get $MSE = 1.43E-04$ and $MSE = 1.03E-04$ respectively. The data matrix is

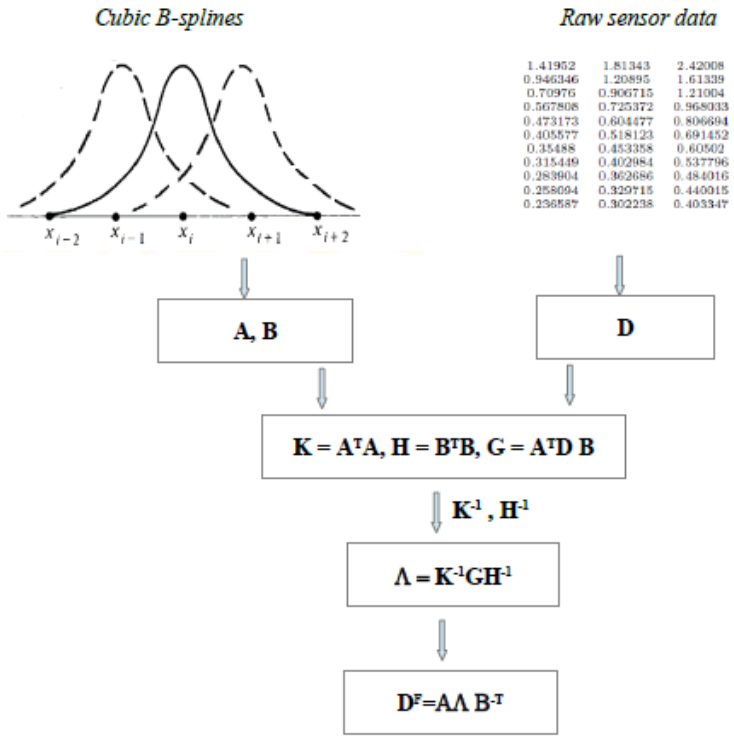

Figure 1: The proposed F–transform based scheme

$$\mathbf{D} = \begin{pmatrix} 1.1752 & 1.41952 & 1.81343 & 2.42008 & 3.33929 \\ 0.783467 & 0.946346 & 1.20895 & 1.61339 & 2.22619 \\ 0.587601 & 0.70976 & 0.906715 & 1.21004 & 1.66965 \\ 0.47008 & 0.567808 & 0.725372 & 0.968033 & 1.33572 \\ 0.391734 & 0.473173 & 0.604477 & 0.806694 & 1.1131 \\ 0.335772 & 0.405577 & 0.518123 & 0.691452 & 0.954083 \\ 0.2938 & 0.35488 & 0.453358 & 0.60502 & 0.834823 \\ 0.261156 & 0.315449 & 0.402984 & 0.537796 & 0.742065 \\ 0.23504 & 0.283904 & 0.362686 & 0.484016 & 0.667858 \\ 0.213673 & 0.258094 & 0.329715 & 0.440015 & 0.607144 \\ 0.195867 & 0.236587 & 0.302238 & 0.403347 & 0.556549 \end{pmatrix}$$

by means of cubic B–splines the matrices $\mathbf{K}$ and $\mathbf{H}$ turn out to be

$$\mathbf{K} = \begin{pmatrix} 0.0394004 & 0.630292 & 0.506921 & 0.0143588 & 0 & 0 & 0 & 0 & 0 \\ 0.630292 & 11.0881 & 12.294 & 1.4594 & 0.0143588 & 0 & 0 & 0 & 0 \\ 0.506921 & 12.294 & 29.3791 & 14.5499 & 1.47254 & 0.00927098 & 0 & 0 & 0 \\ 0.0143588 & 1.4594 & 14.5499 & 29.5514 & 14.6317 & 1.47615 & 0.0114197 & 0 & 0 \\ 0 & 0.0143588 & 1.47254 & 14.6317 & 29.4757 & 14.5957 & 1.46448 & 0.015625 & 0 \\ 0 & 0 & 0.00927098 & 1.47615 & 14.5957 & 29.6587 & 14.5308 & 1.46448 & 0.0114197 \\ 0 & 0 & 0 & 0.0114197 & 1.46448 & 14.5308 & 29.6587 & 14.5885 & 1.45834 \\ 0 & 0 & 0 & 0 & 0.015625 & 1.46448 & 14.5885 & 26.9731 & 7.96345 \\ 0 & 0 & 0 & 0 & 0 & 0.0114197 & 1.45834 & 7.96345 & 3.66225 \end{pmatrix}$$

$$\mathbf{H} = \begin{pmatrix} 0.015625 & 0.359375 & 0.359375 & 0.015625 & 0 \\ 0.359375 & 9.28125 & 12.625 & 1.71875 & 0.015625 \\ 0.359375 & 12.625 & 33.5469 & 16.9844 & 1.71875 \\ 0.015625 & 1.71875 & 16.9844 & 33.5469 & 12.625 \\ 0 & 0.015625 & 1.71875 & 12.625 & 9.28125 \end{pmatrix}$$

3. Compression performance: numerical experiments

In order to show the better approximation obtainable by means of cubic B–splines, firstly we compare the results so obtained with the ones presented in our previous work [12] relatively to two SensorScope deployments: Patrouille des glacier (PDG) and Plaine Morte glacier (PM) [45]. We recall that the PDG deployment had 10 locations whereas PM deployment had 13 locations. Both data sets contain data from several sensors, namely, ambient temperature (^oC), surface temperature (^oC), solar radiation (W/m^2), relative humidity (%), wind speed (m/s), wind direction (deg). For not available data we adopted zero value. In order to evaluate distortion, we computed the Mean Absolute Error MAE_C, MAE_S for the LS approach based on cubic B–splines and sinusoidal shaped basic functions respectively and we tabled the ratio $r_{MAE} = MAE_C/MAE_S$ for two different values of the F–transform compression ratio ρ. The ratio r_{MAE} is useful to exploit differences between the results obtained by the two different basic functions above. Besides, in order to give a more complete vision, we computed the ratio $r_{MSE} = MSE_C/MSE_S$ between the Mean Squared Error MSE_C, MSE_S for the LS approach based on cubic B–splines and sinusoidal shaped basic

Table 1: PDG deployment: r_{MAE} rate for node 1

ρ	AT	ST	SR	RH	WS	WD
0.45	0.0531102	0.00908386	0.0501134	0.011492	0.0383367	0.997299
0.33	0.0638878	0.0119798	0.0663918	0.016865	0.0479208	1.15912

Table 2: PDG deployment: r_{MAE} rate for node 9

ρ	AT	ST	SR	RH	WS	WD
0.45	0.0209357	-	0.0825336	0.0126524	0.0233794	0.775867
0.33	0.0286697	-	0.10183	0.017353	0.0373243	0.966843

functions. The error values are referred to ambient temperature (AT), surface temperature (ST), solar radiation (SR), relative humidity (RH), wind speed (WS), wind direction (WD).

We recall that the compression ratio (CR) is usually defined as

$$CR = (1 - \frac{d'}{d}) \tag{16}$$

where d and d' are the original raw data size and the compressed data size in bits, respectively. With regard to the F–transform, clearly it is $CR = 1-\rho$.

As one can see from Tables 1–6, where the r_{MAE} ratio for some nodes in the two deployments mentioned above is tabled, cubic B–splines provide a better approximation with respect to sinusoidal shaped basic functions even for a small F–transform compression rate ρ (the lower ρ the higher data compression ratio). This behaviour is confirmed by the r_{MSE} ratio (see Figure 2) for both the deployments.

Table 3: PDG deployment: r_{MAE} rate for node 16

ρ	AT	ST	SR	RH	WS	WD
0.45	0.0220832	0.00651324	0.0236523	0.0106423	0.0273434	0.976772
0.33	0.0525902	0.00842405	0.0365564	0.0136515	0.0340729	1.02484

Table 4: PM deployment: r_{MAE} rate for node 20

ρ	AT	ST	SR	RH	WS	WD
0.45	0.0467689	0.00391376	0.00506575	0.00582795	0.00194302	0.289915
0.33	0.0620435	0.00475539	0.00590066	0.00738769	0.00247275	0.364533

Table 5: PM deployment: r_{MAE} rate for node 78

ρ	AT	ST	SR	RH	WS	WD
0.45	0.0746895	0.0135576	0.00806175	0.00796555	0.00240307	0.289915
0.33	0.093278	0.0160717	0.0103972	0.00995694	0.00301168	0.480915

Table 6: PM deployment: r_{MAE} rate for node 83

ρ	AT	ST	SR	RH	WS	WD
0.45	0.0417603	0.00127867	0.0053775	0.00598658	0.0011765	0.293997
0.33	0.0537003	0.00167265	0.00685941	0.00751448	0.00152556	0.367496

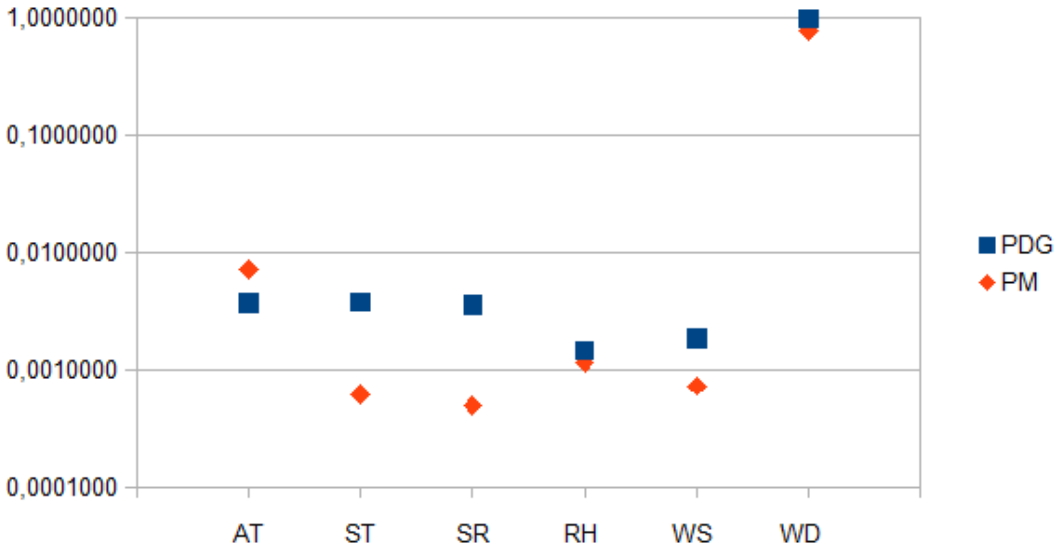

Figure 2: r_{MSE} rate for the two deployments with $\rho = 0.33$

In order to show the robustness of our approach, we compare the results so obtained with the ones in [22], because S–LEC [22] seems to be in many cases better than FELACS [23].

To the scope, we considered, as in [22], temperature and relative humidity measurements from two other SensorScope deployments: LUCE and Le Genepi [45]. In particular, for the LUCE deployment, node 20 with 21,523 samples in the range September 4th, 2007–October 3th, 2007 were considered, whereas for Le Genepi deployment, node 84 with 64,913 samples in the range November 23, 2006–December 17, 2006. In Table 7 the characteristics of the considered data for the node 84 from the LUCE deployment (here denoted as LU84) and for the node 20 from Le Genepi deployment (here denoted as GE20) are tabled.

The obtained compression ratios in [22] are referred to temperature (likely ambient temperature) and relative humidity measurements. In Table 8 there are our results, with a better CR with respect to S–LEC and the corresponding MAE.

The value of CR achievable by our approach is the same for AT, ST and RH, since unlike other approaches such as the lossless ones, the compression is

Table 7: Absolute maximum and minimum values of the data from GE20 and LU84

	variable	*AT*	*ST*	*RH*
min	*GE*20	0	0.025	11.078
max	*GE*20	13.13	23.663	93.877
min	*LU*84	0	0.025	50.981
max	*LU*84	17.36	16.788	98.315

Table 8: CR and MAE for LU84 and GE20 measurements

variable	data set	CR (S-LEC [22])	CR (present scheme)	MAE (S-LEC [22])	MAE (present scheme)
AT	*GE*20	0.548	0.65	0	8.3E-02
ST	*GE*20	-	0.65	-	6.5E-02
RH	*GE*20	0.514	0.65	0	5.9E-03
AT	*LU*84	0.7207	0.82	0	1.86E-02
ST	*LU*84	-	0.82	-	5.9E-02
RH	*LU*84	0.6371	0.82	0	2.03E-03

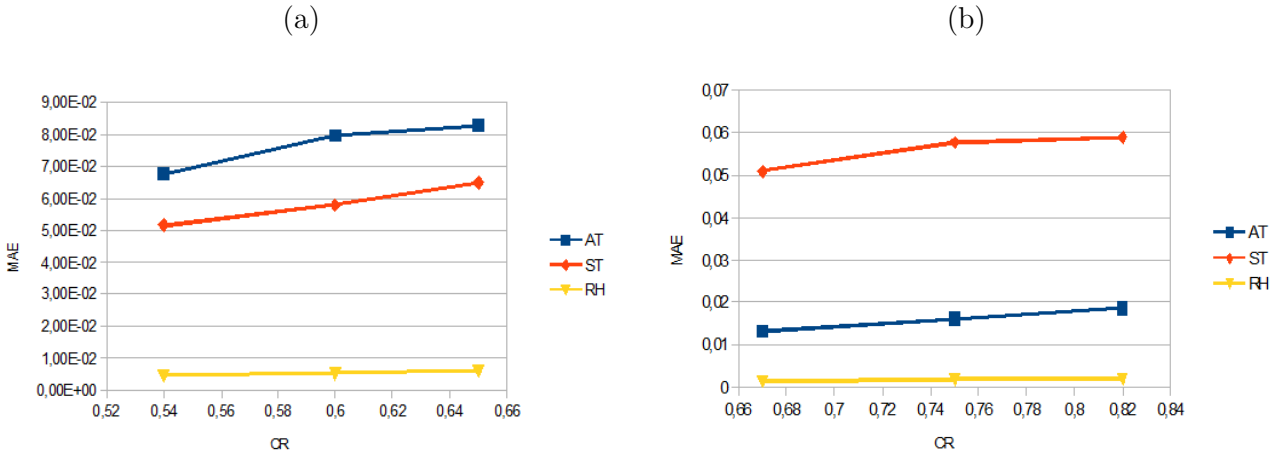

Figure 3: MAE vs CR: (a) GE20 case, (b) LU84 case

Table 9: The r_{MAE} rate

data set	CR	AT	ST	RH
GE20	0.65	0.0205477	0.00216003	0.0113007
LU84	0.82	0.52677	0.71232	0.582145

executed on the data matrix and not on its single columns. Besides, it should be pointed out that the values of MAE are compatible with the minimum and maximum values of the original data: a mean variation in ambient or surface temperature less than $1/10\ ^oC$ is meaningless. A similar remark holds on for the relative humidity.

Figure 3 shows as MAE varies with CR for the different measurements relatively to GE20 and LU84. In Figure 3, the minimum value of CR is the one referred to S–LEC [22].

Figures 4–6 show a sample of the reconstructed data for the GE20 case, with $CR = 0.65$.

For the sake of completeness, in Table 9 we tabled the values of the r_{MAE} rate relatively to the GE20 and the LU84 cases, for the values of CR fixed in Table 8. These results show again the higher performance of cubic B–splines basic functions with respect to sinusoidal shaped basic functions.

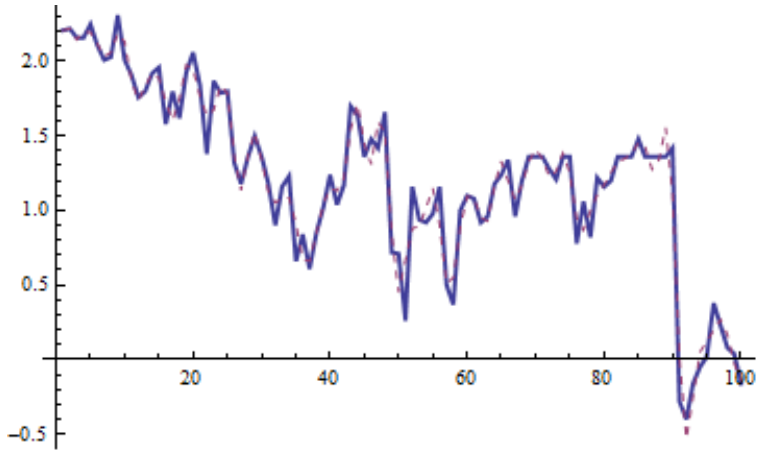

Figure 4: GE20, ambient temperature: original data (thick line), reconstructed data (dashed line)

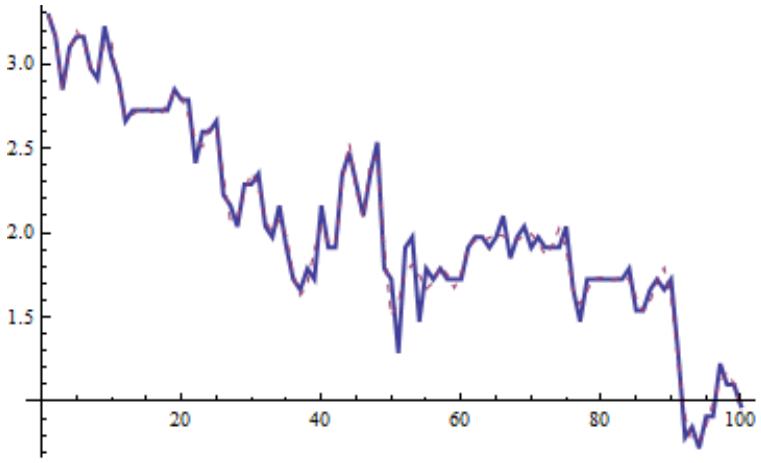

Figure 5: GE20, surface temperature: original data (thick line), reconstructed data (dashed line)

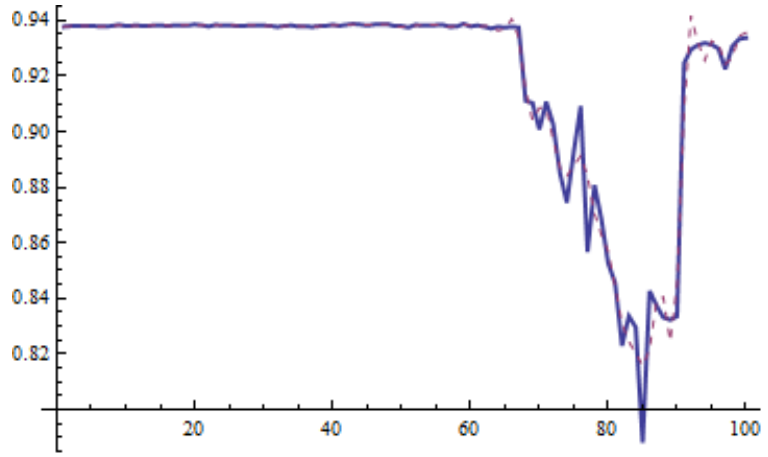

Figure 6: GE20, relative humidity: original data (thick line), reconstructed data (dashed line)

Finally, we wish to compare the computational cost of our approach with that of the S–LEC algorithm, which has substantially the same structure of the LEC.

In [46], the computational cost of LEC is expressed in number of instructions (NI), namely for the LU84 case, LEC requires 44,784 NI for the temperature and 62,817 NI for the relative humidity.

In Section 2, we showed that the computational cost of our approach is $O(nm)$. This means for the LU84 case, as tabled in Table 8, getting $O(17,550 \times 2) = O(35,100)$ for AT, ST and RH jointly considered.

4. A secure compression scheme

In the previous sections we have discussed the efficiency of the cubic B–splines F–transform based compression, due to its low distortion and low computational cost. Now, we wish to show how to achieve a secure compression. We propose a compression–encryption scheme as shown in Figure 7. The procedure is as follows:

1. the matrix Λ Eq. (9) is generated by using cubic B–splines;

2. the size (i.e. N and M) of the data matrix $\mathbf{D}$ is kept secret, since without N and M the inverse F–transform cannot be computed;

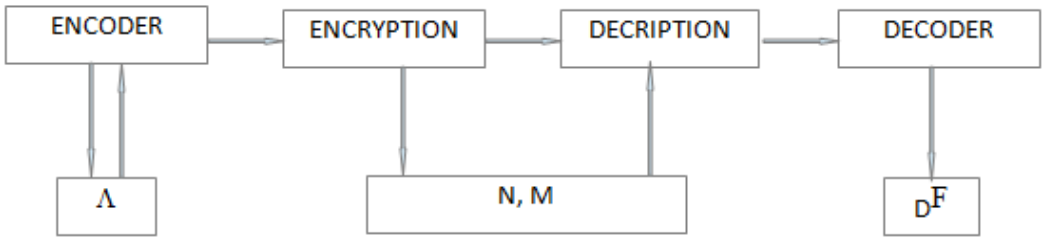

Figure 7: The compression–encryption scheme

3. once N and M are retrieved the inverse F–transform is computed by Eq. (12).

In symmetric key cryptosystems, the same key is used for both encryption and decryption, by resulting in a much faster scheme than public key cryptosystems. The two major types of symmetric key systems are block ciphers and stream ciphers. Block ciphers in general process the plaintext in relatively large blocks at a time with the same key. Stream ciphers encrypt bits individually, by adding a bit from a key stream to a plaintext bit [47].

Even if under certain modes of operation, block ciphers can be used in WSNs [48], stream ciphers are faster and seem the most suitable to WSNs [24].

A very popular stream cipher is RC4. It was introduced in '90s by Rivest [49]. It is essentially a pseudo–random number generator initialized by a secret key. The RC4 algorithm turns out to be really a secure cipher under certain conditions, i.e. by pre–processing the base key, whose length should be at least 128 bits, and any counter or initialization vector by means of a hash function such as MD5 or by discarding the first 256 output bytes of the pseudo–random generator before beginning encryption [24].

Unlike [4], here we consider the RC4 algorithm with a 128 bits–base key, as suggested in [24]. In this way, in a brute force attack, one should try 2^{128} guesses to find the key and recover the encrypted parameters, i.e. if a 1000 MIPS computer were used, this would mean $2^{128}/(1000 \times 10^6 \times 3600 \times 24 \times 365) > 10^{22}$ years.

Under a known–data attack, let us suppose that an illegal user obtained some information, e.g. the exact value of M. The encryption scheme turns out to be still secure, because even a small change in N would cause a

Table 10: PDG16: MAE for incorrect values $\overline{N}$

$\overline{N}$	AT	ST	SR	RH	WS	WD
$1.5n = 1890$	3.981746	5.005864	129.978459	47.333548	3.645330	149.764020
$2n = 2520$	2.985519	3.753404	97.458035	35.490762	2.733273	112.293276
$2.4n = 3024$	0.744714	1.08467	32.2789	5.20535	2.04482	100.037
$3n = 3780$	5.132178	8.486034	302.041375	51.470216	5.577987	213.155838

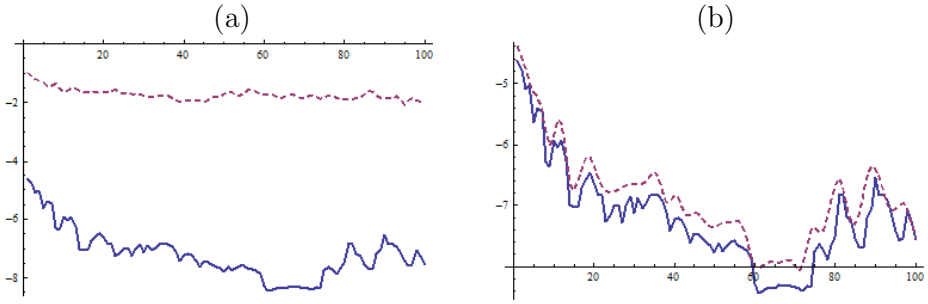

Figure 8: PDG16: reconstructed ambient temperature with (a) $\overline{N} = 1890$, (b) $\overline{N} = 3024$ (dashed line: reconstructed data; thick line: original data)

substantial distortion in the reconstructed data.

As an example, we consider again the data from the node 16 in the PDG deployment (for short PDG16 in what follows) with a compression ratio $CR = 0.67$, i.e. $N = 3072$, $M = 6$, $n = 1260$, $m = 5$.

An illegal user may try some incorrect values $\overline{N} > n$ to reconstruct the data, even just partially if $\overline{N} < N$.

In Table 10, the distortion (i.e. the MAE) of the reconstructed data for different values of $\overline{N}$ is tabled. As one can see, the distortion is noticeable, especially for $\overline{N} = 3n > N$. When the value of $\overline{N}$ is close to N, the distortion is lesser but not meaningless.

This behaviour is graphically shown in Figures 8–9, where some samples of the reconstructed data obtained by means of the incorrect values $\overline{N}$ are depicted.

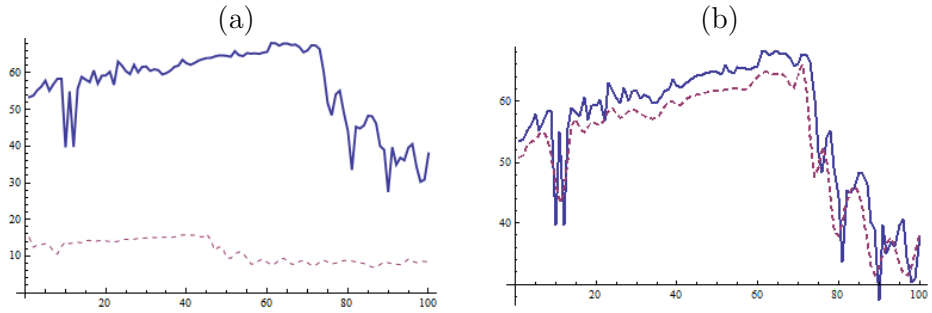

Figure 9: PDG16: reconstructed relative humidity with (a) $\overline{N} = 1890$, (b) $\overline{N} = 3024$ (dashed line: reconstructed data; thick line: original data)

5. Conclusions

In this paper we firstly investigated the use of cubic B–splines to improve the performance of an F–transform based data compression, namely the LS approach, and afterwards we showed how to keep the compression scheme secure. In spite of the fact that cubic B–splines require two auxiliary points both on the left and on the right of the considered interval, this choice has the following advantages:

– high accuracy, as also formally proved by means of Theorem 1;

– low computational cost of the final LS approach, when compared to the usual DWT or even to a state–of–the–art lossless compression scheme;

– reliability of the integrated compression-encryption scheme.

In particular, with respect to the last point above, we proposed to integrate the compression technique with an existing encryption algorithm suitable to WSNs, such as the RC4 cipher, in order to keep secret two parameters necessary to recover the source data. Even if an illegal user had access to one of these parameters, the scheme would be still secure, because a small change in the remaining parameter would cause a not meaningless distortion in the reconstructed data.

References

References

[1] G. H. Keat, A. Samsudin, Z. Zainol, Enhance Performance of Secure Image Using Wavelet Compression, Int. J. Computer, Control, Quantum

and Inform. Eng. 1(1) (2007) 165–168.

[2] X. Li, J. Knipe, H. Cheng, Image compression and encryption using tree structures, Pattern Recognition Letters 18 (1997) 12531259.

[3] S.S. Maniccam, N.G. Bourbakis, Lossless image compression and encryption using SCAN, Pattern Recognition 34 (2001) 1229–1245.

[4] C. Lv, Q. Zhao, Integration of Data Compression and Cryptography: Another Way to Increase the Information Security, in: Advanced Information Networking and Applications Workshops (AINAW'07), 2007, pp. 543–547.

[5] N. Gura, A. Patel, A. Wander, H. Eberle, S. Shantz, Comparing elliptic curve cryptography and rsa on 8-bit cpus, in: Cryptographic Hardware and Embedded Systems, Lecture Notes in Computer Science, vol. 3156, Springer, Berlin, DE, 2004, pp 119–132.

[6] G. Gaubatz, J.P. Kaps, B. Sunar, Public key cryptography in sensor networks - revisited, in: Security in Ad-hoc and Sensor Networks, Lecture Notes in Computer Science, vol. 3313, Springer, Berlin, DE, 2005, pp 2–18.

[7] Q. Huang, J. Cukier, H. Kobayashi, B. Liu, J. Zhang, Fast authenticated key establishment protocols for self-organizing sensor networks, in: Proc. 2nd ACM Int. Conf. Wireless sensor networks and applications, ACM Press, 2003, pp. 141150.

[8] B. Schneier, Applied Cryptography (2nd Ed.), John Wiley & Sons, London, 1996.

[9] M. A. Razzaque, C. Bleakley, S. Dobson, Compression in Wireless Sensor Networks: A Survey and Comparative Evaluation, ACM Trans. Sensor Networks 10(1) (2013) 5:1–43.

[10] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, Wireless sensor networks: A survey, Computer Networks 38(4) (2002) 393-422.

[11] G.J. Pottie, W.J. Kaiser, Wireless integrated network sensors, Communications of ACM, 43 (2000) 51-58.

[12] M. Gaeta, V. Loia, S. Tomasiello, Multisignal 1-D compression by F-transform for wireless sensor networks applications, Appl. Soft Comput. 30 (2015) 329–340

[13] I. Perfilieva, Fuzzy transforms: theory and applications, Fuzzy Sets Syst. 157 (2006) 993-1023.

[14] G. Patane', Fuzzy Transform and least-squares approximation: analogies, differences, and generalizations, Fuzzy Sets Syst. 180(1) (2011) 41–54.

[15] I. Perfilieva, B. De Baets, Fuzzy transforms of monotone functions with application to image compression, Inform. Sci. 180 (2010) 3304-3315.

[16] F. Di Martino, V. Loia, I. Perfilieva, S. Sessa, An image coding/decoding method based on direct and inverse fuzzy transforms, Int. J. Approx. Reas. 48 (2008) 110-131.

[17] F. Di Martino, V. Loia, I. Perfilieva, S. Sessa, Fuzzy transform for coding/decoding images: A short description of methods and techniques, Studies in Fuzziness and Soft Comput. 298 (2013) 139–146.

[18] P. Vlasanek, I. Perfilieva, Influence of various types of basic functions on image reconstruction using F-transform, in: 8th Conference of the European Society for Fuzzy Logic and Technology (EUSFLAT 2013) - Advances in Intelligent Systems Research 32 (2013) 497–502.

[19] P. Hurtik, I. Perfilieva, Image compression methodology based on fuzzy transform using block similarity, in 8th Conference of the European Society for Fuzzy Logic and Technology, EUSFLAT 2013 - Advances in Intelligent Systems Research 32 (2013) 521–526.

[20] F. Di Martino, V. Loia, S. Sessa, Fuzzy transforms for compression and decompression of color videos, Inform. Sci. 180 (2010) 3914-3931.

[21] F. Di Martino, P. Hurtik, I. Perfilieva, S. Sessa, A color image reduction based on fuzzy transforms, Inform. Sci. 266 (2014) 101-111

[22] Y. Liang, Y. Li, An Efficient and Robust Data Compression Algorithm in Wireless Sensor Networks, IEEE Commun. Letters 18(3) (2014) 439–442.

[23] J. Gana Kolo, S. A. Shanmugam, D. W. Gin Lim, L. Ang, Fast and efficient lossless adaptive compression scheme for wireless sensor networks, Computers and Electrical Eng. 41 (2015) 275-287.

[24] N. Fournel, M. Minier, S. Ubeda, Survey and Benchmark of Stream Ciphers for Wireless Sensor Networks, in: Information Security Theory and Practices. Smart Cards, Mobile and Ubiquitous Computing Systems, Lecture Notes in Computer Science, vol. 4462, Springer, Berlin, DE, 2007, pp. 202–214.

[25] I. Perfilieva, Fuzzy transforms, in: J.F. Peters, et al. (Eds.), Transactions on Rough Sets II, Lecture Notes in Computer Science, vol. 3135, Springer, Berlin, DE, 2004, pp. 63-81.

[26] I. Perfilieva, M. Dankova, Towards F-transforms of a higher degree, in: IFSA-EUSFLAT Conference, 2009, pp. 585-588.

[27] M.Dankova, M.Stepnicka, Fuzzy transform as an additive normal form, Fuzzy Sets Syst. 157 (2006) 1024-1035.

[28] M. Stepnicka, O. Polakovic, A neural network approach to the fuzzy transform, Fuzzy Sets Syst. 160 (2009) 1037-1047.

[29] M. Stepnicka, R. Valasek, Numerical Solution of Partial Differential Equations with Help of Fuzzy Transform, in: Proc. IEEE Int. Conf. Fuzzy Systems, 2005, pp. 1104–1109.

[30] B. Bede, I.J. Rudas, Approximation properties of fuzzy transforms, Fuzzy Sets Syst. 180 (2011) 20-40.

[31] I. Perfilieva, V. Novak, V. Pavliska, A. Dvorak, M. Stepnicka, Analysis and prediction of time series using fuzzy transform, in: IEEE World Congress on Computational Intelligence, 2008, pp. 3875-3879.

[32] I. Perfilieva, N. Yarushkina, T. Afanasieva, A. Romanov, Time series analysis using soft computing methods, Int. J. General Syst. 42(6) (2013) 687–705.

[33] V. Novk, V. Pavliska, I. Perfilieva, M. Stepnicka, F-transform and Fuzzy natural logic in time series analysis, in: 8th Conference of the European Society for Fuzzy Logic and Technology, EUSFLAT 2013 - Advances in Intelligent Systems Research 32 (2013) 40–47.

[34] M.L. Guerra, L. Stefanini, Expectile smoothing of time series using F-transform, in: 8th Conference of the European Society for Fuzzy Logic and Technology, EUSFLAT 2013 - Advances in Intelligent Systems Research 32 (2013) 559–564.

[35] V. Novak, I. Perfilieva, M. Holcapek, V. Kreinovich, Filtering out high frequencies in time series using F-transform, Inform. Sci. 274 (2014) 192-209.

[36] F. Di Martino, V. Loia, S. Sessa, Fuzzy transforms method and attribute dependency in data analysis, Inform. Sci. 180 (2010) 493-505.

[37] J.K.I. Tomanova, Hidden functional dependencies found by the technique of F-transform, in: 8th Conference of the European Society for Fuzzy Logic and Technology, EUSFLAT 2013 - Advances in Intelligent Systems Research 32 (2013) 662–668.

[38] R.C. Mittal, A. Tripathi, Numerical solutions of generalized Burgers-Fisher and generalized Burgers-Huxley equations using collocation of cubic B-splines, Int. J. Computer Mathematics, 92(5) 2015 745–758.

[39] R.C. Mittal, R. Bhatia, A numerical study of two dimensional hyperbolic telegraph equation by modified B-spline differential quadrature method, Appl. Math. Computation 244 (2014) 976-997.

[40] G.G. Lorentz, Approximation of Functions, Holt, Rinehart and Winston, New York, 1966.

[41] G.A. Anastassiou, S.G. Gal, Approximation Theory: Moduli of Continuity and Global Smoothness Preservation, Birkhauser, Boston, 2000.

[42] S. G. Gal, G. A. Anastassiou, Shape-Preserving Approximation by Real and Complex Polynomials, Birkhauser, Boston,2008.

[43] H. M. Antia, Numerical methods for scientists and engineers, vol. 1, Birkhauser Verlag, Basel, Switzerland, 2002.

[44] T. Yamamoto, Y. Ikebe, Inversion of band matrices, Linear Algebra and its Applications 24 (1979) 105–111.

[45] Sensorscope dataset. http://sensorscope.epfl.ch/index.php/EnvironmentalData

[46] F. Marcelloni, M. Vecchio, Enabling Compression in Tiny Wireless Sensor Nodes, in: S. Tarannum (Eds.), Wireless Sensor Networks, InTech, Vienna, 2011, pp. 257–276.

[47] C. Paar, J. Pelzl, Understanding Cryptography, Springer-Verlag, Berlin, 2010.

[48] Y. W. Law, J. Doumen, P. Hartel, Survey and benchmark of block ciphers for wireless sensor networks, ACM Trans. Sen. Netw. 2(1) (2006) 65–93.

[49] R. Rivest, The RC4 encryption algorithm, RSA Data Security Inc., 1992.