FACTA UNIVERSITATIS
Series: **Electronics and Energetics** Vol. 32, N° 4, December 2019, pp. 539-554
https://doi.org/10.2298/FUEE1904539P

ON THE NODE ORDERING OF PROGRESSIVE POLYNOMIAL APPROXIMATION FOR THE SENSOR LINEARIZATION

Aneta Prijić, Aleksandar Ilić, Zoran Prijić, Emilija Živanović, Branislav Randjelović

University of Niš, Faculty of Electronic Engineering, Niš, Serbia

Abstract. *Many sensors exhibit nonlinear dependence between their input and output variables and specific techniques are often applied for the linearization of their transfer characteristics. Some of them include additional analog circuits, while the others are based on different numerical procedures. One commonly used software solution is Progressive Polynomial Approximation. This method for sensor transfer function linearization shows strong dependence on the order of selected nodes in the linearization vector. There are several modifications of this method which enhance its effectiveness but require extensive computational time. This paper proposes the methodology that shows improvement over Progressive Polynomial Approximation without additional increase of complexity. It concerns the order of linearization nodes in linearization vector. The optimal order of nodes is determined on the basis of sensor transfer function concavity. The proposed methodology is compared to the previously reported methods on a set of analytical functions. It is then implemented in the temperature measurement system using a set of thermistors with negative temperature coefficients. It is shown that its implementation in the low-cost microcontrollers integrated into the nodes of reconfigurable sensor networks is justified.*

Key words: *Sensor Linearization, Progressive Polynomial Approximation, Reconfigurable Sensor Networks, NTC Thermistor*

1. INTRODUCTION

Transfer functions of sensors used in measurement systems usually do not have linear dependence between input and output variables. In addition, transfer functions often change with time. For these reasons, measurement systems based on the sensors exhibit various errors such as offset, gain, hysteresis, cross-sensitivity, drift, and non-linearity [1], [2]. In order to achieve reliable measurement, these errors should be compensated. One approach is to use additional analog circuits to condition sensors output signal [3], [4], [5]. However, analog compensation is not always appropriate for sensors integrated

Received February 5, 2019; received in revised form July 1, 2019
Corresponding author: Aneta Prijić
Faculty of Electronic Engineering, Aleksandra Medvedeva 14, 18000 Niš, Serbia
(E-mail: aneta.prijic@elfak.ni.ac.rs)

into reconfigurable sensor networks [1], [6], [7]. A more flexible solution is to convert sensor output into the digital domain, where various numerical linearization methods may be applied in the form of compensation algorithms. These methods rely on a set of correction functions applied on a so-called linearization vector composed of linearization nodes. The effectiveness of the linearization method is evaluated on the basis of the number of nodes required to reduce non-linearity below a specific value, computation time, and implementation complexity.

The simplest linearization method is based on a look-up table (LUT) which is also the fastest one. However, to obtain a high accuracy of the estimated input value, a high number of linearization nodes should be implemented in the LUT, making it memory consuming. To reduce memory requirement, a sparse LUT can be combined with an interpolation method [8]. A simple method is piece-wise linear interpolation which connects each two adjacent LUT values with an appropriate linear function. This type of interpolation can be also used for linearization of the whole transfer function. For N linearization nodes, sensors inverse transfer function will be represented by $N-1$ first order polynomials [9]. The main disadvantage of this method is a high number of nodes required for the linearization of highly nonlinear functions. This implies either a large memory requirement or a slow response time of the system [8]. More advanced methods are Lagrange, Newton [10] and spline [11] interpolations. However, Lagrange interpolation suffers from overfitting effect for polynomials of higher degree and it is generally not applicable to highly nonlinear sensors [10], [12]. The Newton method is more flexible and efficient when additional linearization nodes are introduced, but it is primarily applied for equidistant nodes [13]. On the other hand, spline interpolation is effective, but it comes at high implementation costs [14].

More effective and more commonly used methods are Progressive Polynomial Approximation (PPA) [15], [16] and linearization methods based on the Artificial Neural Networks (ANN) [17], [18]. The effectiveness of the PPA method, besides the linearization nodes number, depends on the node ordering in the linearization vector. Results obtained using the same compensation algorithm, but with the different order of nodes (permutation) [19], may vary between almost perfect in some cases, to even increased non-linearity in the other. In the case of the ANN method, effectiveness depends on the neural network topology and the time needed for its training.

This paper proposes the methodology that improves the accuracy of the PPA while keeping its simplicity. Theoretical background, a summary of PPA, and an overview of its modifications are presented in Section 2. Section 3 contains an analysis of the PPA method effectiveness considering different permutations of the linearization vector for four different functions. Two of these functions are convex and two concave. This is done in order to elaborate on the idea that the optimal order of nodes in the linearization vector is dependent on the transfer function shape. In such a sense, an extensive computation time needed to accomplish the desired linearity by analysis of all permutations may be avoided. Experimental support of presented numerical results is given in Section 4, using negative temperature coefficient (NTC) thermistors as sensors.

2. LINEARIZATION METHODS

2.1. Underlying theory

The transfer function of sensors is usually expressed as $y = f(x)$, where x is sensor input and y is sensor output [20]. The linearization method calculates the desired output value y_{lin}, using the workflow depicted in Fig. 1. Sensor output is first digitized, using an analog-to-digital converter (ADC), and then the linearization algorithm is applied. Obtained output value y_{lin} should vary linearly with the sensor input, i.e. $y_{lin} = kx + n$, where k is the gain and n (usually 0) is the offset of the desired transfer function [1]. All operations are performed by a microcontroller, which is a part of the reconfigurable sensor node.

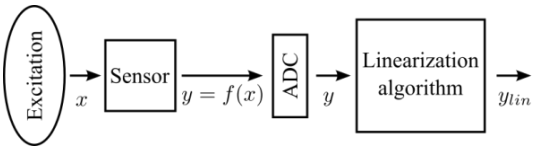


Fig. 1 Workflow of a sensor linearization process

There are two types of linearization methods, as illustrated in Fig. 2. The first involves an estimation of the sensor transfer function $y = f_{est}(x)$ and subsequent numerical determination of sensor inverse transfer function which is used to obtain the estimated input value $x_{est} = f_{est}^{-1}(y)$. The linearized output value is $y_{lin} = kx_{est}$.

Methods of the second type modify sensors output y using a correction function g, so the linearized output is calculated as $y_{lin} = g(y)$. Estimated input value, in this case, is calculated as $x_{est} = y_{lin}/k$.

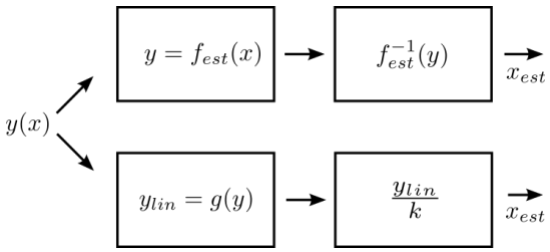


Fig. 2 Linearization of a sensor output for two distinct methods.

The linearization node is a pair of two values: input x and corresponding output y. Values are determined experimentally by applying a known stimulus at the sensor input and measuring the value of its output. Input values are usually chosen equidistantly, starting at the minimal input that a sensor can detect, and ending at the full-scale. Nodes are then ordered to form the linearization vector. Linearization coefficients, implemented into the correction function $g(y)$, are determined using the linearization vector, and then stored in the memory of a microcontroller. On each measurement, these coefficients are used to calculate the linear output value of the sensor.

The effectiveness of linearization is evaluated using relative non-linearity [21]:

$$Nonlinearity = \frac{D_{max}}{x_{max}} \cdot 100\%, \tag{1}$$

where x_{max} is full-scale input and D_{max} is the maximum deviation of the real input from the ideal transfer characteristic. Due to the nature of the linearization algorithm, non-linearity will be equal to zero for any of the linearization nodes if a quantization error introduced by AD conversion is neglected. Therefore, additional measurements need to be performed to form a set of N_e evaluation nodes and calculate the maximum deviation as:

$$D_{max} = max\left(\left|x_i - \frac{y_i}{k}\right|\right) \quad \text{for } i=1,\ldots N_e, \tag{2}$$

where x_i is the applied input value and y_i is the corresponding output value.

2.2. Progressive Polynomial Approximation (PPA)

Calculation of the correction function in PPA is a successive process [15]. For each linearization node, new correction function, denoted as $g_i(y)$, is defined. Therefore, *i*-th function corrects the non-linearity around the *i*-th node, while keeping the corrections introduced by the previously defined functions. The final correction function is the one defined for the last node.

In order to calculate correction function, linear output value t_i at each node should be defined as:

$$t_i = kx_i + n \quad \text{for } i = 1, \ldots, N. \tag{3}$$

These values are then used to calculate linearization coefficients for correction functions.

The first correction function $g_1(y)$ is defined as:

$$g_1(y) = y + c_1 \tag{4}$$

where c_1 is the linearization coefficient calculated for the first linearization node:

$$c_1(y) = t_1 - y_1. \tag{5}$$

Thus, $g_1(y)$ adds the value c_1 to the sensor output, eliminating the offset (if present). The correction function at *i*-th node is defined as:

$$g_i(y) = g_{i-1}(y) + c_i \prod_{j=1}^{i-1}(g_j(y) - t_j) \quad \text{for } i = 2, \ldots, N \tag{6}$$

and the linearization coefficient is calculated as:

$$c_i = \frac{t_i - g_{i-1}(y_i)}{\prod_{j=1}^{i-1}(g_j(y_j) - t_j)} \quad \text{for } i = 2, \ldots, N. \tag{7}$$

These correction functions eliminate the gain error and successively minimize the transfer function non-linearity. Since the final correction function includes all the linearization nodes, it will output the desired value for any of these nodes while between them linearized output will deviate from the ideal one to some extent.

2.3. Modifications of the PPA method

In PPA, the first two nodes in the linearization vector are chosen from the ends of a sensor range, thus eliminating the offset and gain errors [15]. Then, each new node is added halfway between the previous two. When the linearization vector ordered in that way is used, achieved results are not optimal, but large non-linearity is avoided. Note that nodes are not necessarily equidistant. The main advantage of PPA lies in its simplicity, so it does not require intensive time-consuming operations. This makes it particularly suitable for the implementation in reconfigurable sensor networks.

Improved Progressive Polynomial Approximation (IMPPA) is the method based on permutations of nodes in the initial linearization vector. Each permutation is tested in order to find the best one [19]. To reduce the number of arithmetic operations, IMPPA does not test all possible permutations of the initial vector. Rather, it fixes the first node from the beginning of the sensor range as the first, the node from the end of the range as the second, and then permutes the remaining ones. The effectiveness of each permutation is determined by the non-linearity obtained at nodes which are inserted between the nodes of the linearization vector. A major drawback of this method is increased implementation costs in terms of the complexity and computation time. It should be noted that this method finds the optimal permutation of the given linearization vector for equidistant nodes. The method can be further improved if linearization vector with non-equidistant nodes is used.

A probability density function is used to improve PPA, as presented in [22], [23]. This approach proposes an accumulation of linearization nodes in the part of a transfer function that will be used most commonly during a sensor lifetime. This can significantly improve measurement system accuracy in some cases, but the problem of the further ordering of the selected nodes still remains unsolved.

A different linearization method inspired by PPA, called Modified Progressive Polynomial Approximation (MPPA), is addressed in [24]. Methodology for selection of the nodes which does not consider their order in the linearization vector is introduced. The larger set of equidistant nodes is formed first. The linearization vector is not predefined, but it is populated at each step by the node from the original set at which current linearized function deviates most from the linear one. Consequently, selected nodes in the linearization vector are not equidistant.

3. PROPOSED METHODOLOGY

Proposed modification of the PPA method concerns the order of nodes in the linearization vector to obtain the desired transfer function linearity without increasing the algorithm complexity. Several analytic expressions commonly used to model sensors transfer functions are analyzed. In order to make a comparison of the results, transfer functions are normalized before the linearization methodology was applied. Both, input (argument) and output (function value) are normalized to range [0, 1]. If x^m is sensor input and y^m is sensor output, normalized input and output values are calculated using the following equations:

$$x = \frac{x^m - x^m_{min}}{x^m_{max} - x^m_{min}}, \tag{8}$$

$$y = \frac{y^m - y^m_{min}}{y^m_{max} - y^m_{min}}, \tag{9}$$

where x_{min}^{m} and x_{max}^{m} are minimum and full range input values, while $y_{min}^{m}$and $y_{max}^{m}$are the corresponding output values.

The initial linearization vector T_I is formed as a set of N equidistant nodes starting from the beginning of the sensors range. It is expressed as:

$$T_I = [(x_1, y_1),\ (x_2, y_2), \ldots, (x_N, y_N)], \tag{10}$$

or, using shorthand notation: $T_I = [1,2, \ldots, N]$. Permutations of the linearization vector are denoted as T_i, i = 1, 2, . . . , N − 1!.

3.1. Convex functions

The PPA linearization methodology is applied to an exponential function:

$$f(x) = 1 - e^{-x/p} \tag{11}$$

where p is parameter used to adjust its non-linearity ($0<p<1$). The optimal permutation of the linearization vector depends on this parameter value [19].

For the basic permutation analysis, linearization vector with N=5 nodes is selected. Two permutations of the initial vector are specially considered:

$$\mathrm{T}_1 = [1, 5, 2, 3, 4], \tag{12}$$
$$T_2 = [1, 5, 4, 3, 2]. \tag{13}$$

Following the IMPPA method, in both permutations, the first node is taken from the beginning and the second from the end of the input range. The remaining nodes in permutation T_1 are ordered starting from the beginning, while in permutation T_2 starting from the end of the range. The linearization method is applied and output non-linearity is calculated considering different input function non-linearities. Obtained results are shown in Fig. 3. It can be seen that permutation T_1 is more effective since it can reduce the output non-linearity below 1% for the input non-linearity up to 24%, while T_2 can achieve similar results for the input non-linearity up to 19%. The analysis is extended for other permutations and distribution shown in Fig. 4 is obtained. It is evident that output non-linearity is strongly dependent on the order of nodes in the linearization vector. For example, when the input function non-linearity is 24%, the output non-linearities achieved using all permutations vary between satisfactory 0.83%, and quite high 3.12%.

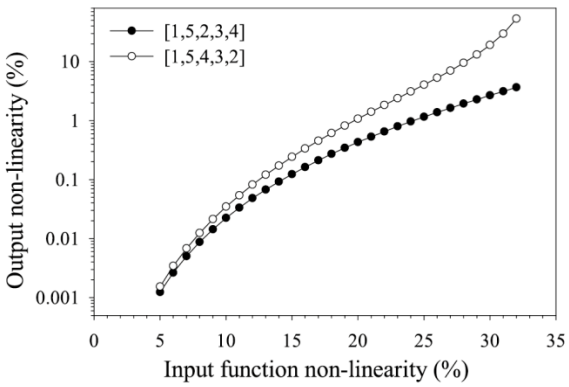


Fig. 3 Output non-linearity vs. input non-linearity for convex exponential function (11).

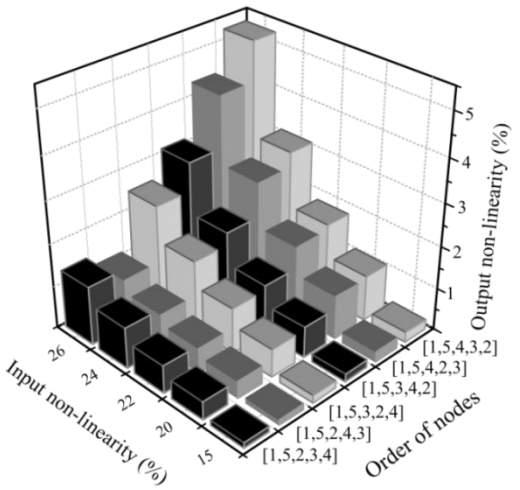


Fig. 4 Output non-linearity distribution vs. input function non-linearity and linearization vector permutations for convex exponential function (11).

Dependence of the output non-linearity on permutations of the linearization vector is also analyzed using a quadratic function:

$$f(x) = px - qx^2 \tag{14}$$

where p and q are parameters which determine the input non-linearity ($0<p,q<1$). Obtained results are shown in Fig. 5. A similar dependence of the output non-linearity on the order of nodes in the linearization vector is observed, as it was the case with function (11). It is important to note that permutation $[1, 5, 2, 4, 3]$ gives the best results for both analyzed functions for all input non-linearity values.

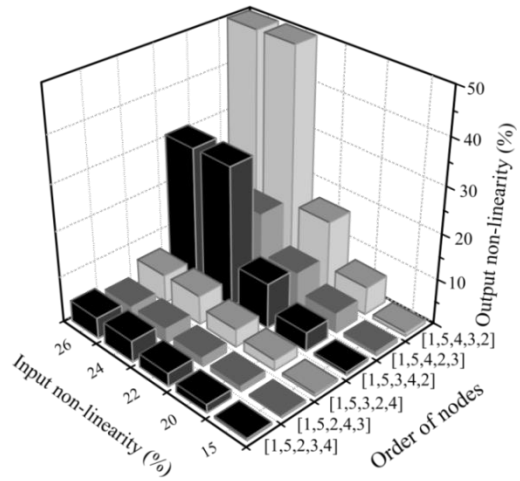


Fig. 5 Output non-linearity distribution vs. input function non-linearity and linearization vector permutations for convex quadratic function (14).

Permutations that will give the best results for both functions and all input non-linearity values can be identified for other numbers of nodes in the linearization vector as well. Those permutations are listed below:

$$\begin{aligned} N = 4 \quad & T_{cx} = [1, 4, 3, 2] \\ N = 5 \quad & T_{cx} = [1, 5, 2, 4, 3] \\ N = 6 \quad & T_{cx} = [1, 6, 2, 4, 3, 5] \\ N = 7 \quad & T_{cx} = [1, 7, 2, 3, 5, 4, 6] \\ N = 8 \quad & T_{cx} = [1, 8, 3, 2, 4, 6, 5, 7] \end{aligned}$$

The proposed methodology is compared with the PPA and IMPPA methods. Maximum input non-linearity that can be compensated (reduced below 1%) is shown in Fig. 6 for function (11), and in Fig. 7 for function (14). These figures show that higher input non-linearity can be compensated by the proposed methodology than with PPA. The methodology is somewhat less efficient than IMPPA for the higher number of nodes since it is related to the transfer function shape and does not necessarily distinguish the best one from the set of all permutations. On the other hand, its implementation is much simpler. This is evident comparing the number of arithmetical operations required to calculate the linearization coefficients of the considered methods, as presented in Tab. 1. It is evident that improvement is achieved at a negligible cost regarding the number of required arithmetical operations. The linearization vectors with the maximum of eight nodes are considered, due to the fact that the IMPPA algorithm requires a large number of arithmetical operations that should be executed within the microcontroller. This makes it unsuitable for applications based on the low-cost microcontrollers.

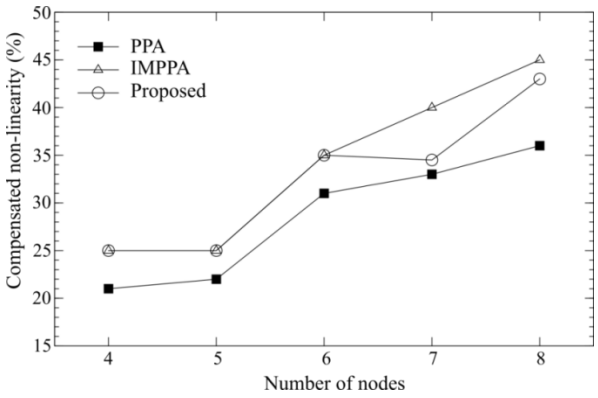


Fig. 6 Compensated non-linearity of the convex exponential function (11) vs. the number of nodes in the linearization vector.

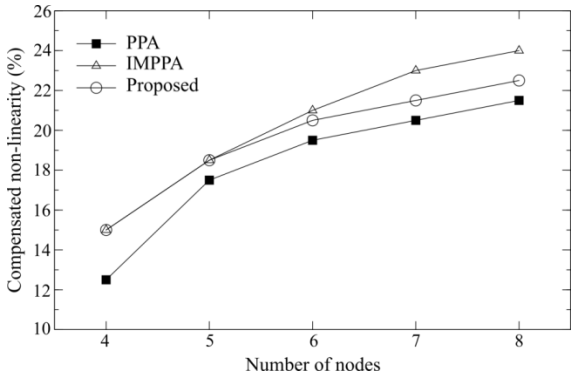


Fig. 7 Compensated non-linearity of the convex quadratic function (14) vs. the number of nodes in the linearization vector.

Table 1 Number of arithmetical operations required to calculate the linearization coefficients for the different number of nodes.

N	PPA	Proposed	IMPPA
5	121	146	726
6	320	350	8400
7	841	871	100920
8	2205	2245	1587600

3.2. Concave functions

Permutations (12) and (13) are also used for the linearization of the concave function:

$$f(x) = e^{px} \tag{15}$$

with different values of the non-linearity parameter p and results are shown in Fig 8. In this case permutation T_2 is more effective (opposite to the results shown in Fig. 3).

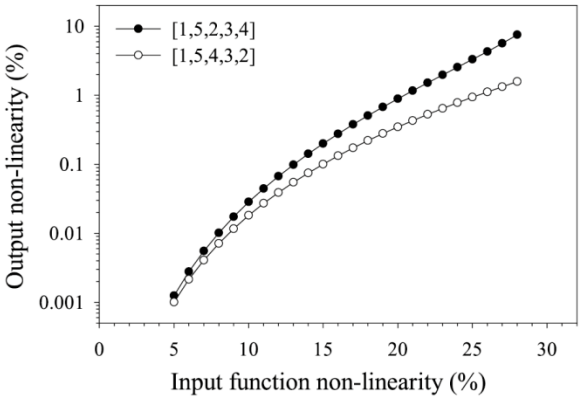


Fig. 8 Output non-linearity vs. input non-linearity for concave exponential function (15)**.**

Analysis of the quadratic function:

$$f(x) = px + qx^2 \tag{16}$$

also gives permutation T_2 as better for the linearization.

Following the same procedure as for the convex functions, it is possible to identify the best permutations of the linearization vector for the concave functions:

$$\begin{aligned} N &= 4 \quad T_{cv} = [1, 4, 2, 3] \\ N &= 5 \quad T_{cv} = [1, 5, 4, 2, 3] \\ N &= 6 \quad T_{cv} = [1, 6, 5, 3, 4, 2] \\ N &= 7 \quad T_{cv} = [1, 7, 6, 5, 3, 4, 2] \\ N &= 8 \quad T_{cv} = [1, 8, 7, 6, 5, 3, 4, 2] \end{aligned}$$

The proposed methodology is again compared with PPA and IMPPA. In order to compare these methods, the maximum value of the input non-linearity that can be reduced below 1% using different lengths of the linearization vector is calculated. This value is plotted as a function of the number of linearization nodes in Fig. 9 for the function (15) and in Fig. 10 for the function (16). It can be seen that the improvement over the PPA is in this case even greater than it was in the case of convex functions. Also, the improvement offered by IMPPA over the proposed methodology remains small.

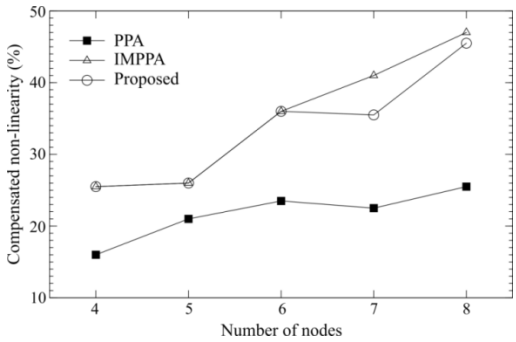


Fig. 9 Compensated non-linearity of the concave exponential function (15) vs. the number of nodes in the linearization vector.

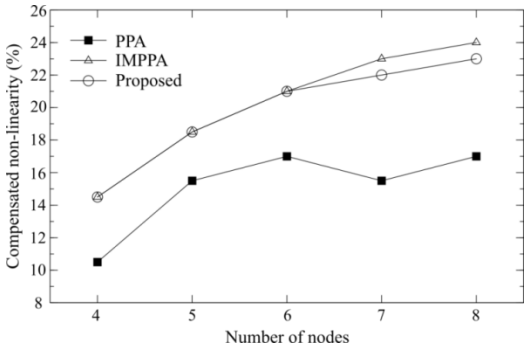


Fig. 10 Compensated non-linearity of the concave quadratic function (16) vs. the number of nodes in the linearization vector.

The simplified flowchart of the calibration procedure by the proposed method is presented in Fig. 11. The sign of the second derivative of the transfer function is used to distinguish between the concave or convex property. It is easily determined by numerical differentiation at the set of normalized measured values x_i and y_i. The sensor transfer functions with the inflection point can be treated as a union of segments with the specific convex or concave property. Particular segments of the transfer function can be independently linearized by the proposed method, which encounters the increased number of linearization nodes. Therefore, it is cost effective to employ the PPA method for such functions, even though it gives less linearized output. However, in real measurements, a very few sensor transfer characteristics have the inflection point in their specific application range.

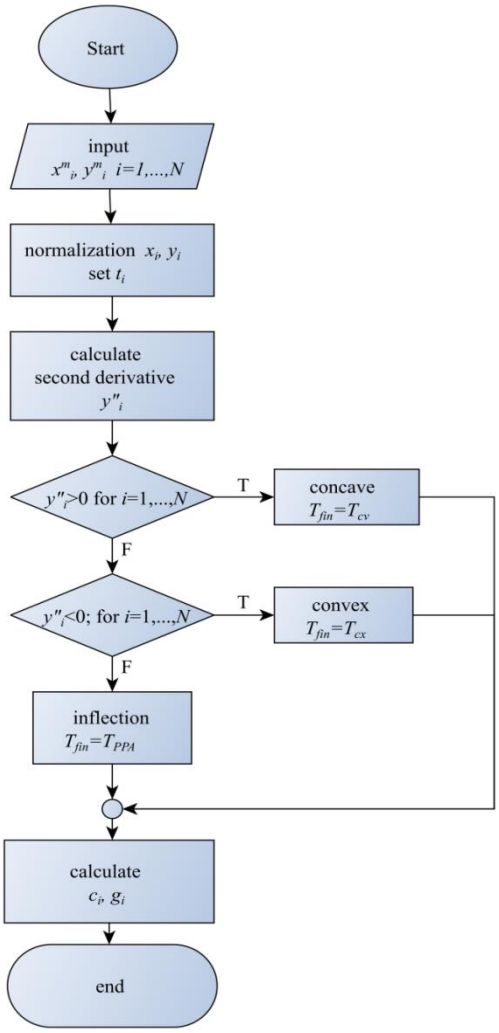


Fig. 11 The simplified flowchart of the calibration procedure by the proposed method

4. CASE STUDY

A temperature measurement system using an NTC thermistor is chosen for testing of the proposed method. NTC thermistor has highly nonlinear dependence of the resistance on temperature:

$$R = R_0 e^{\beta\left(\frac{1}{T}-\frac{1}{T_0}\right)}, \tag{17}$$

where R_0 is resistance at the reference temperature T_0 and β is material–specific constant.

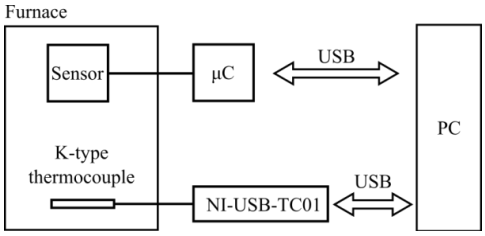
Block diagram and a photo of the experimental setup are shown in Fig. 12. The sensing part of the system is constructed using NTC thermistor in the voltage divider configuration with referent resistance R_{ref}. It gives a convex shape of the transfer function [25]:

$$y(x) = \frac{1}{1+Ae^{Bx}}, \tag{18}$$

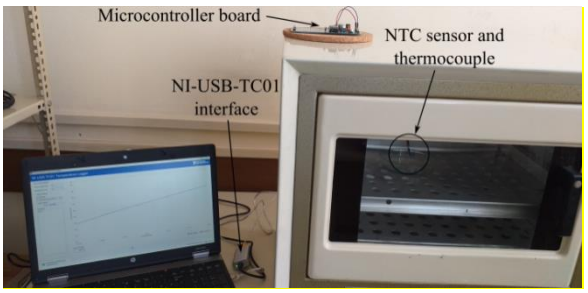
where y is the normalized output voltage of the divider, $x = 1/T$, $B = \beta$ and A is a constant dependent on the voltage divider configuration:

$$A = \frac{R_0 e^{-\frac{\beta}{T_0}}}{R_{ref}}. \tag{19}$$

The microcontroller is used to convert the output voltage of the sensing part into a digital domain and then to apply the linearization algorithm to the transfer function. In order to calculate non-linearity after the linearization, K–type thermocouple with ±0.75% accuracy connected through NI-USB-TC01 interface is used as a reference [26]. The temperature in the furnace is varied and measured using both, the NTC sensor and the referent thermocouple. Results are then transferred to a PC, where non-linearity is calculated by the software.



a)



b)

Fig. 12 Experimental setup: a) Block diagram, b) Photograph.

Four different NTC thermistors are employed within the sensing part to achieve different non-linearities of the transfer function. Thermistors parameters are listed in Table 2, while Fig. 13 shows their transfer functions non-linearities as a function of temperature for $R_{ref}=R_0$. The proposed methodology of linearization is applied for the different number of linearization nodes. The results for the thermistor R_1 calculated using five linearization nodes are shown in Fig. 14. Relative non-linearity of the linearized output as a function of the normalized input for PPA and the proposed methodology are presented. It can be seen that using the proposed methodology the non-linearity was reduced to a value of 0.49%, while PPA reduced it to a value of 1.39%. This is a significant improvement, achieved without increased complexity.

Table 2 Parameters of the considered NTC thermistors

Parameter	R_1	R_2	R_3	R_4
R_0 (kΩ) at T_0=298K	1	10	100	470
β (K)	3730	4300	4600	5000

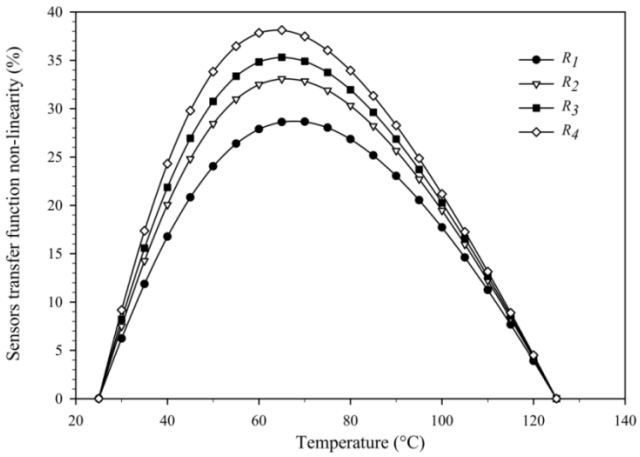


Fig. 13 Sensor transfer function non-linearity vs. temperature**.**

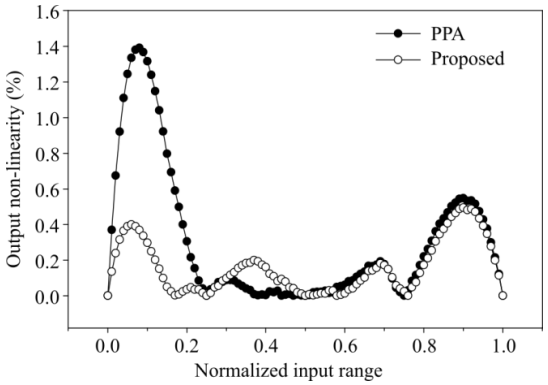


Fig. 14 Output non-linearity of the transfer function vs. the normalized input range for N = 5 linearization nodes and thermistor R_1.

Summary of the linearization results is shown in Tabs. 3 and 4. Maximal output non-linearities of the transfer function obtained by three methods for the different numbers of linearization nodes are listed. It is evident that the proposed methodology performs better than PPA in almost all considered cases. Also, an interesting observation is that in the case of thermistor R_3 and seven linearization nodes, PPA gives even better results than IMPPA. This is possible since PPA does not use equidistant nodes. Therefore, the combination of nodes used by PPA can, in some cases, give better results than the optimal permutation of the linearization vector with equidistant nodes, used by IMPPA.

Table 3 Linearization results for NTC thermistors R_1 and R_2

N	R_1			R_2		
	PPA	Proposed	IMPPA	PPA	Proposed	IMPPA
5	1.39%	0.49%	0.49%	12.5%	1.93%	1.93%
6	0.34%	0.32%	0.31%	1.58%	1.53%	0.88%
7	1.04%	0.91%	0.30%	1.15%	1.05%	0.20%
8	0.31%	0.26%	0.08%	0.69%	0.37%	0.12%

Table 4 Linearization results for NTC thermistors R_3 and R_4

N	R_3			R_4		
	PPA	Proposed	IMPPA	PPA	Proposed	IMPPA
5	6.25%	1.13%	1.13%	5.23%	1.54%	1.54%
6	1.14%	0.77%	0.56%	0.90%	0.48%	0.44%
7	0.66%	0.84%	0.73%	0.59%	0.43%	0.37%
8	0.42%	0.19%	0.15%	0.89%	0.40%	0.28%

5. CONCLUSION

Progressive Polynomial Approximation is analyzed as a method for linearization of sensors. Methods based on PPA could achieve low output non-linearity only by using a large number of permutations of the linearization vector. It is found that the number of permutations of the linearization vector, required to accomplish the desired output non-linearity, may be significantly reduced by taking into account the shape of the sensors transfer function. By using numerical experiment, the optimal order of nodes in the linearization vector is determined for convex and concave exponential and quadratic functions. It is shown that non-linearity could be reduced to lower values than obtained using PPA, with the negligible increase of complexity in the linearization algorithm. Comparing to the IMPPA, the proposed methodology is somewhat less efficient for the higher number of nodes, but it requires much lower computational resources. Therefore, the proposed methodology is suitable for implementation in low-cost microcontrollers integrated into the nodes of reconfigurable sensor networks.

Acknowledgement: *This work was supported in part by the Serbian Ministry of Education, Science and Technological Development under Grant TR32026 and in part by Ei PCB Factory, Niš, Serbia.*

REFERENCES

[1] J. Rivera-Mejia, M. Carrillo-Romero, and G. Herrera-Ruiz, "Self-compensation to build reconfigurable measurement systems", *IEEE Instrumentation & Measurement Magazine*, vol. 16, pp. 10–19, 2013.

[2] D. Vasiljević, Č. Žlebič, G. Stojanović, M. Simić, L. Manjakkal, Z. Stamenković, "Cost-effective sensors and sensor nodes for monitoring environmental parameters", *Facta Universitatis, Series: Electronics and Energetics*, vol. 31, no. 1, pp. 11-23, 2018.

[3] C. Renneberg and T. Lehmann, "Analog circuits for thermistor linearization with Chebyshev optimal linearity error", In Proc. of 18th European Conference on Circuit Theory Design, Seville, Spain, August 2007, pp. 910–913.

[4] J. Lukić, D. Živanović and D. Denić, "A compact and cost-effective linearization circuit used for angular position sensors", *Facta Universitatis, Series: Automatic Control and Robotics*, vol. 14, no. 2, pp. 123–134, 2015.

[5] N. Khachab and M. Ismail, "Linearization techniques for nth-order sensor models in MOS VLSI technology", *IEEE Transactions on Circuits and Systems*, vol. 38, no. 12, pp. 1439–1450, December 1991.

[6] S. Jasko and G. Simon, "CSP-based sensor network architecture for reconfigurable measurement systems", *IEEE Transactions on Instrumentation and Measurement*, vol. 60, no. 6, pp. 2104–2117, June 2011.

[7] A. Žorić, D. Martinović and S. Obradović, "A simple 2D digital calibration routine for transducers", *Facta Universitatis, Series: Electronic and Energetics*, vol. 19, no. 2, pp. 197-207, 2006.

[8] L. Bengtsson, "Lookup table optimization for sensor linearization in small embedded systems", *Journal of Sensor Technology*, vol. 2, pp. 177–184, 2012.

[9] J. E. Brignell, "Software techniques for sensor compensation", *Sensors and Actuators, A*, vol. 25, pp. 29–35, 1991.

[10] A. Pašić and J. Dowling, "Linearising calibration methods for a generic embedded sensor interface (GESI)", In Proc. of 1st International Conference on Sensing Technology, November 2005, pp. 185–190.

[11] F. Shangchun, Z. Qiuli, Q. Jie, and Z. Pengcheng, "The temperature compensation of high precision pressure sensor based on the cubic spline interpolation", *MAPAN Journal of Metrology Society of India*, vol. 21, no. 3, pp. 167–170, 2006.

[12] R. Burden and J. Faires, *Numerical Analysis*, 8-th ed. Thomson Higher Education, 2005.

[13] R. B. Srivastava and P. K. Srivastava "Comparison of Lagrange's and Newton's interpolating polynomials", *Journal of Experimental Science*, vol. 3, no. 1, pp. 1-4, 2012.

[14] C. Bluemm, R. Weiss, R. Weigel, and D. Brenk, "Correcting non-linearity and temperature influence of sensors through B-spline modeling", In Proc. of IEEE International Symposium on Industrial Electronics, July 2010, pp. 3356–3361.

[15] G. van der Horn and J. L. Huijsing, *Integrated Smart Sensors Design and Calibration*. Springer, 1998.

[16] K. F. Lyahou, G. van der Horn, and J. H. Huijsing, "A noniterative polynomial 2-D calibration method implemented in a microcontroller", *IEEE Transactions on Instrumentation and Measurement*, vol. 46, no. 4, pp. 752–757, August 1997.

[17] J. Rivera, M. Carrillo, M. Chacon, G. Herrera, and G. Bojorquez, "Self-calibration and optimal response in intelligent sensors design based on artificial neural networks", *Sensors*, vol. 7, pp. 1509–1529, 2007.

[18] T. Nenov and S. Ivanov, "Linearization of characteristics of relative humidity sensor and compensation of temperature impact", *Sensors and Materials*, vol. 19, no. 2, pp. 95–106, 2007.

[19] J. Rivera, G. Herrera, M. Chacon, P. Acosta, and M. Carrillo, "Improved progressive polynomial algorithm for self-adjustment and optimal response in intelligent sensors", *Sensors*, vol. 8, pp. 7410–7427, 2008.

[20] J. Fraden, *Handbook of Modern Sensors Physics, Designs, and Applications*, 4th ed., New York: Springer, 2010.

[21] J. Carr, *Sensors and Circuits: Sensors, Transducers, and Supporting Circuits for Electronic Instrumentation, Measurement, and Control*, Prentice Hall, 1993.

[22] J. M. Dias Pereira, O. Postolache, and P. S. Girao, "PDF-based progressive polynomial calibration method for smart sensors linearization", *IEEE Transactions on Instrumentation and Measurement*, vol. 58, no. 9, pp. 3245–3252, September 2009.

[23] J. M. Dias Pereira, O. Postolache, and P. S. Girao, "Adaptive self-calibration algorithm for smart sensors", In Proc. of IMTC 2005 - Instrumentation and Measurement Technology Conference, May 2005, pp. 648–652.

[24] S. Rahili, J. Ghaisari, and A. Golfar, "Intelligent selection of calibration points using a modified progressive polynomial method", *IEEE Transactions on Instrumentation and Measurement*, vol. 61, no. 9, pp. 2519–2523, September 2012.

[25] B. Baker, "Thermistors in single supply temperature sensing circuits", Microchip Technology Inc., Tech. Rep. AN685, 2004. Available at: http://ww1.microchip.com/downloads/en/AppNotes/00685b.pdf

[26] *NI USB-TC01 Thermocouple Measurement Device with NI Instant DAQ Technology*, National Instruments, 2014, Data Sheet. Available at: http://www.ni.com/datasheet/pdf/en/ds-215.