EFFICIENT IMPLEMENTATION OF MULTIHARMONIC LEAST-SQUARES FITTING ALGORITHMS

Pedro E. Xavier [1], *Fernando M. Janeiro* [2], *Pedro M. Ramos* [3]

[1] Instituto de Telecomunicações, Instituto Superior Técnico, Lisbon, Portugal, peaxavier@gmail.com
[2] Instituto de Telecomunicações, Universidade de Évora, Évora, Portugal, fmtj@uevora.pt
[3] Instituto de Telecomunicações, Instituto Superior Técnico, UTL, Lisbon, Portugal, pedro.m.ramos@ist.utl.pt

Abstract: Multiharmonic least-squares fitting algorithms have been applied in many fields of instrumentation and measurement. However, as the number of harmonics increases, the size of the matrices needed to describe the relation between the sampled signal and the multiharmonic model increases substantially. In this paper an efficient memory-wise implementation of the algorithm is proposed, and numerical simulations are used to assess its performance. Measurement results are also presented.

Key words: least-squares, multiharmonic signals, memory efficient implementation.

1. INTRODUCTION

Least-squares sine-fitting algorithms were normalized for the test of analog-to-digital converters [1] and waveform records [2]. Since then, the application of these algorithms in many fields of instrumentation and measurement has been proposed. These include impedance measurements [3, 4], ultrasonic ranging [5], power quality measurements [6], radio receiver unbalance measurements [7] and phase estimation [8].

In [9] a generalized form of the fitting algorithms was published for multiharmonic signals. These algorithms can estimate the amplitudes and phases of the harmonics of a signal. Much like the original sine-fitting algorithms of [1], there are two versions for the multiharmonic situation. The first version estimates the DC component and the harmonic amplitudes and phases. It is called the 2*H*+1 algorithm for *H* frequency components (fundamental and harmonics) and is a simple multiple least-squares regression solved by determining the pseudo-inverse of the matrix that relates the time domain sampled signal and the model parameters.

The second version also estimates the signal frequency and is called the 2*H*+2 algorithm. However, the estimation of the frequency makes it a nonlinear regression since it is not possible to obtain a linear separable relation of the acquired signal and model parameters. This means that the algorithm is iterative, thus requiring fairly good initial estimates to improve convergence and even under these circumstances, convergence is very dependent on the actual harmonic signal composition as shown in [10].

One drawback of these algorithms consists on the amount of memory needed to construct the matrices used by them, since this can be a major limitation in standalone instrumentation. In fact, in DSP based standalone instruments, memory is usually very limited. In this paper, we have developed a procedure to greatly reduce the memory requirements for the 2*H*+2 algorithm. This memory efficient implementation is tested and compared with the original 2*H*+2 algorithm implementation.

2. ALGORITHM DESCRIPTION

A multiharmonic signal is modeled, in the time domain, by

$$u(t) = C + \sum_{h=1}^{H} A_h \cos(2\pi hft) + B_h \sin(2\pi hft) \qquad (1)$$

where A_h and B_h are the in-phase and quadrature amplitudes of harmonic h, f is the signal frequency, H is the total number of considered frequency components. Note that in (1) all the harmonics up to H are considered but this is not required by the algorithms. Some harmonics can be removed if a priori knowledge about the signal harmonic composition is available.

After acquiring N samples from the input signal, at a sampling rate f_S, the sample vector $\mathbf{y} = [y_1\, y_2 \ldots y_N]^T$ can be constructed where the timestamp of sample n is $t_n = (n-1)/f_S$.

For the multiharmonic model, when the frequency is known (the so-called 2*H*+1 algorithm), the estimated parameters are

$$\mathbf{x} = \begin{bmatrix} C \\ A_1 \\ B_1 \\ A_2 \\ B_2 \\ \vdots \\ A_H \\ B_H \end{bmatrix} \qquad (2)$$

with

$$\mathbf{y} = \mathbf{D}\mathbf{x} + \mathbf{r} \qquad (3)$$

where $\mathbf{r}$ is a vector with the residuals and matrix $\mathbf{D}$ has N rows and $2H+1$ columns, given by

$$\mathbf{D} = \begin{bmatrix} \mathbf{1} & \mathbf{c}_1 & \mathbf{s}_1 & \cdots & \mathbf{c}_H & \mathbf{s}_H \end{bmatrix} \tag{4}$$

where $\mathbf{c}_h = \cos(2\pi h f \mathbf{t})$, $\mathbf{s}_h = \sin(2\pi h f \mathbf{t})$, $\mathbf{1}$ is a vector with N rows with all elements equal to 1 and $\mathbf{t}$ is a vector with the timestamps. The column order of (4) matches the order of (2).

The parameters that minimize the least-squares error are obtained using

$$\hat{\mathbf{x}} = \left[\mathbf{D}^T \mathbf{D}\right]^{-1} \mathbf{D}^T \mathbf{y} . \tag{5}$$

The second version of the algorithm also estimates the signal frequency and thus the estimated parameters in each iteration are

$$\mathbf{x}^{(i)} = \begin{bmatrix} C \\ A_1 \\ B_1 \\ A_2 \\ B_2 \\ \vdots \\ A_H \\ B_H \\ \Delta\omega^{(i)} \end{bmatrix} \tag{6}$$

where $\Delta\omega^{(i)}$ is the angular frequency correction estimated in the current iteration. This frequency correction is then used to obtain the frequency of the next iteration according to

$$\omega^{(i+1)} = \omega^{(i)} + \Delta\omega^{(i)} . \tag{7}$$

In this algorithm, the matrix is

$$\mathbf{D}^{(i)} = \begin{bmatrix} \mathbf{1} & \mathbf{c}_1 & \mathbf{s}_1 & \cdots & \mathbf{c}_H & \mathbf{s}_H & \sum_{h=1}^{H} \boldsymbol{\alpha}_h \end{bmatrix} \tag{8}$$

where $\boldsymbol{\alpha}_h = -A_h^{(i-1)} h(\mathbf{t} \circ \mathbf{s}_h) + B_h^{(i-1)} h(\mathbf{t} \circ \mathbf{c}_h)$ (where $\circ$ is the Hadamard product or entrywise product).

Note that, in each iteration the last $2H+1$ columns of $\mathbf{D}^{(i)}$ must be recalculated due to changes in the estimated frequency and that, in the last column, the amplitudes of the previous iteration are used to determine $\boldsymbol{\alpha}_h$.

The criteria to stop the iterations, as defined in [1], is to stop when the RMS of the residuals reaches a certain pre-defined threshold. However, this threshold depends on the signal noise and non-harmonic disturbances. Another major drawback is that the residuals must be evaluated in each iteration and this requires a considerable amount of processing since the reconstructed signal must be determined. A different criteria was proposed in [10] in which the algorithm stops when the absolute relative frequency correction is below a threshold. With this method there is no need to reconstruct the signal thus reducing the processing time of each iteration.

3. EFFICIENT IMPLEMENTATION

The $2H+2$ algorithm, as described in the previous section, requires a considerable amount of memory to store matrix $\mathbf{D}^{(i)}$ with $N(2H+2)$ memory positions required. In addition to these, N memory positions are required for the sample vector, $(2H+2)^2$ are required for $\left[\mathbf{D}^{(i)}\right]^T \mathbf{D}^{(i)}$ and $2H+2$ for $\left[\mathbf{D}^{(i)}\right]^T \mathbf{y}$. The amount of memory per item and the total are presented in Table 1. For example, in [10], results with H=49 frequency components and N=10000 samples were presented. In this case, the amount of memory required is 1020100. For implementation in standalone instruments this is a prohibitive number and the algorithm cannot be used. External memory can be used to increase the available memory but always with a higher processing time.

Table 1. Amount of memory required for each item in the original $2H+2$ algorithm.

Item	Memory Positions
$\mathbf{D}^{(i)}$	$N(2H+2)$
$\mathbf{y}$	N
$\left[\mathbf{D}^{(i)}\right]^T \mathbf{D}^{(i)}$	$(2H+2)^2$
$\left[\mathbf{D}^{(i)}\right]^T \mathbf{y}$	$2H+2$
Total	$2NH+3N+4H^2+10H+6$

As shown in [11], it is possible to reduce the amount of memory used for processing the traditional sine-fitting algorithms by directly building $\left[\mathbf{D}^{(i)}\right]^T \mathbf{D}^{(i)}$ and $\left[\mathbf{D}^{(i)}\right]^T \mathbf{y}$ thus totally bypassing the need to build $\mathbf{D}^{(i)}$. In this case, the sizes of these two elements are independent on the number of samples N which can reduce significantly the total amount of memory needed.

In the case of the multiharmonic $2H+2$ algorithm, $\left[\mathbf{D}^{(i)}\right]^T \mathbf{D}^{(i)}$ can be written as

$$\begin{bmatrix} E_{1,1} & E_{1,2} & E_{1,3} & \cdots & E_{1,2H} & E_{1,2H+1} & E_{1,2H+2} \\ E_{1,2} & E_{2,2} & E_{2,3} & \cdots & E_{2,H} & E_{2,2H+1} & E_{2,2H+2} \\ E_{1,3} & E_{2,3} & E_{3,3} & \cdots & E_{3,H} & E_{3,2H+1} & E_{3,2H+2} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ E_{1,2H} & E_{2,H} & E_{3,H} & \cdots & E_{2H,2H} & E_{2H,2H+1} & E_{2H,2H+2} \\ E_{1,2H+1} & E_{2,2H+1} & E_{3,2H+1} & \cdots & E_{2H,2H+1} & E_{2H+1,2H+1} & E_{2H+1,2H+2} \\ E_{1,2H+2} & E_{2,2H+2} & E_{3,2H+2} & \cdots & E_{2H,2H+2} & E_{2H+1,2H+2} & E_{2H+2,2H+2} \end{bmatrix} \tag{9}$$

with

$$E_{r,c} = \left[\mathbf{vector}(r)\right]^T \mathbf{vector}(c) . \tag{10}$$

In (10), $\mathbf{vector}(v)$ is a vector with N rows and 1 column

$$\mathbf{vector}(v)=\begin{cases}\sum_{h=1}^{H}\boldsymbol{\alpha}_h & \text{if } v=2H+2\\ \mathbf{1} & \text{if } v=1\\ \mathbf{c}_{\text{harm}(v)} & \text{otherwise if } v \text{ is even}\\ \mathbf{s}_{\text{harm}(v)} & \text{otherwise if } v \text{ is odd}\end{cases} \quad (11)$$

where $\text{harm}(v)=\text{floor}(v/2)=\lfloor v/2 \rfloor$.

$\left[\mathbf{D}^{(i)}\right]^T\mathbf{y}$ is given by

$$\left[\mathbf{D}^{(i)}\right]^T\mathbf{y}=\begin{bmatrix}P_1\\P_2\\\vdots\\P_{2H+2}\end{bmatrix} \quad (12)$$

with

$$P_r=\left[\mathbf{vector}(r)\right]^T\mathbf{y}. \quad (13)$$

With this implementation, N memory positions are required for the sample vector, $(2H+2)^2$ are required for $\left[\mathbf{D}^{(i)}\right]^T\mathbf{D}^{(i)}$ and $2H+2$ for $\left[\mathbf{D}^{(i)}\right]^T\mathbf{y}$. The amount of memory per item and the total are presented in Table 2. For the previously presented example, the total amount of memory required is 20100.

Table 2. Amount of memory required for each item in the efficient implementation of the $2H+2$ algorithm.

Item	Memory Positions
$\mathbf{y}$	N
$\left[\mathbf{D}^{(i)}\right]^T\mathbf{D}^{(i)}$	$(2H+2)^2$
$\left[\mathbf{D}^{(i)}\right]^T\mathbf{y}$	$2H+2$
Total	$N+4H^2+10H+6$

In order to minimize the number of evaluations of trigonometric functions, each sample is processed and the elements corresponding to that sample are added to each element of $\left[\mathbf{D}^{(i)}\right]^T\mathbf{D}^{(i)}$ and $\left[\mathbf{D}^{(i)}\right]^T\mathbf{y}$. Since $\left[\mathbf{D}^{(i)}\right]^T\mathbf{D}^{(i)}$ is symmetric, only the upper triangular matrix elements are computed. Once all the samples are processed, the remaining elements are copied from their symmetrical.

In Figures 1 and 2, a direct comparison of the amount of memory needed for both implementations is presented. In Fig. 1 the analysis is based on the influence of the number of samples in the acquired record for a fixed number of harmonics (H=49). This figure clearly shows the advantage, in terms of memory required, of the efficient implementation presented in this paper. For large N, the ratio of both implementations is about 100 (from the totals presented in Table 1 and Table 2, the actual ratio is $2H$+3 which for H=49 is a little above 100).

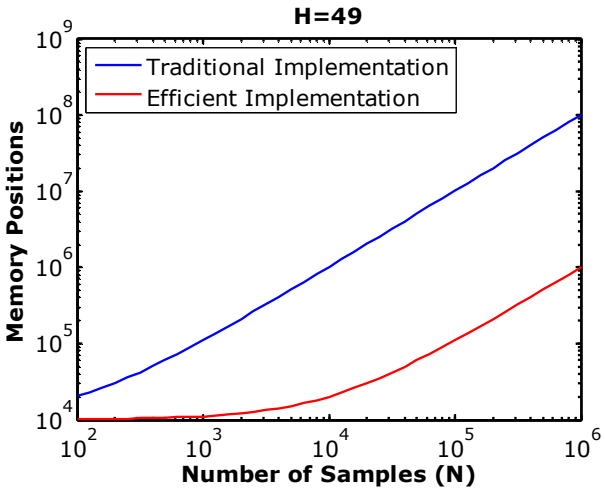

Fig. 1. Comparison of the total amount of memory for both implementations as a function of the number of samples for 49 harmonics.

In Fig. 2, the analysis is focused on the influence of the number of harmonics for a fixed number of samples. It can be seen that the efficient implementation is always better than the traditional implementation as expected. However, as the number of harmonics increases, this difference is reduced because the elements that are dominant are the ones related with $\left[\mathbf{D}^{(i)}\right]^T\mathbf{D}^{(i)}$ and eventually the ratio would become 1. Nevertheless, it should be pointed out that this is not a realistic situation as the number of harmonics rarely increases that much. On the other hand, the use of a higher number of samples (as analyzed in Fig. 1) is quite relevant due to the fact that the variance of the estimated parameters can be reduced with the use of more samples.

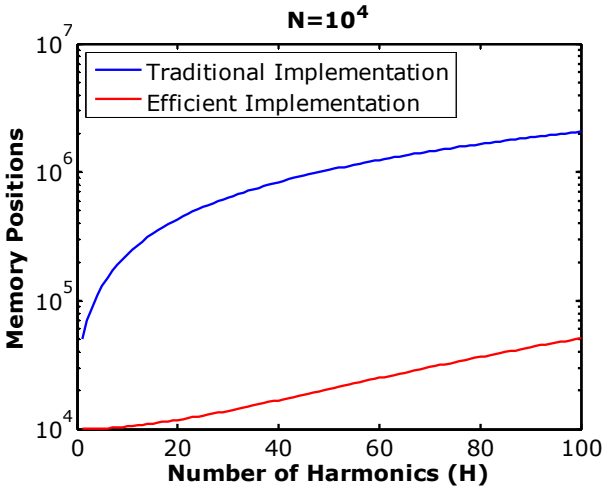

Fig. 2. Comparison of the total amount of memory for both implementations as a function of the number of harmonics for 10 000 samples.

Although the proposed implementation is memory-wise more efficient than the traditional implementation, it is, as expected, more time consuming due to the way the elements are calculated. Since the overall execution time of each implementation depends on the number of iterations, the

comparison analysis is performed by measuring the time needed for each iteration in both implementations.

The comparison of the simulation time per iteration, as a function of the number of samples in the acquired record for a fixed number of harmonics (H=49), is presented in Fig. 3. In Fig. 4, the simulation results as a function of the number of harmonics is shown. The variations observed in the execution time for the traditional implementation are caused solely by the reduced times measured and the fact that the results were obtained for one single execution (*i.e.,* no repetitions). As expected, the execution time per iteration of the traditional implementation is much smaller than the time per iteration of the efficient implementation. It should be noted that the algorithms were implemented in Matlab and that for the traditional implementation Matlab is very efficient in matrix manipulations and operations. As for the efficient implementation, it is done as described above and does not take advantage of the efficient matrix manipulations of Matlab.

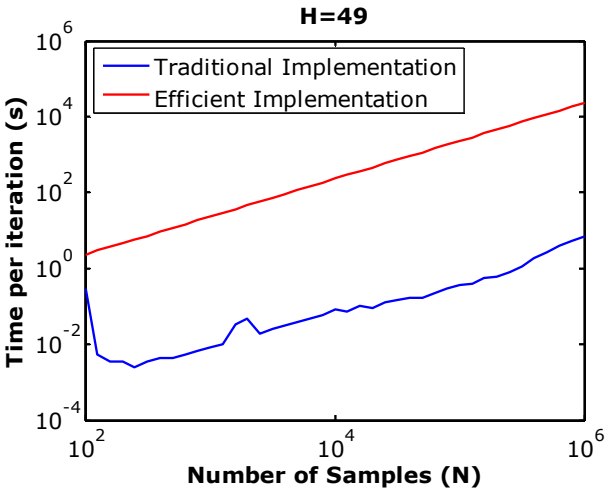

Fig. 3. Comparison of the simulation time per iteration for both implementations as a function of the number of samples for 49 harmonics.

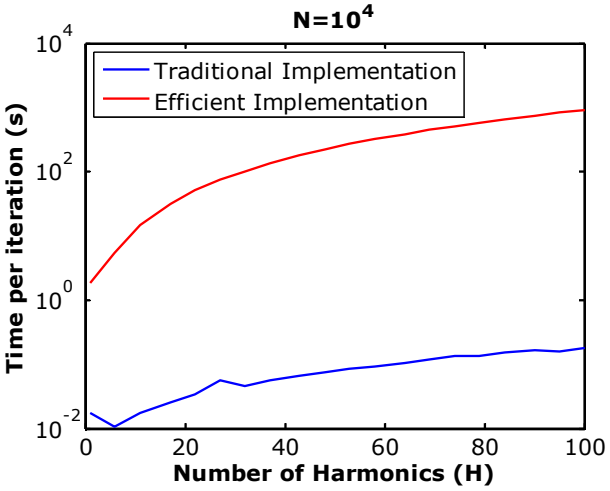

Fig. 4. Comparison of the simulation time per iteration for both implementations as a function of the number of harmonics for 10 000 samples.

4. MEASUREMENT RESULTS

In this section, measurement results that demonstrate the validity of the proposed method are presented. A LEM voltage sensor is used to measure the 50 Hz power grid voltage. Samples are acquired using a National Instruments PCI-6122 with a 16-bit ADC in the ±10 V voltage range. The sampling rate was set at f_s≈1111 kS/s and 1000 samples were acquired. In Fig. 5, the time domain samples are shown for the first five periods. Some distortion of the power grid voltage, namely in the peaks of the signal, is clearly visible.

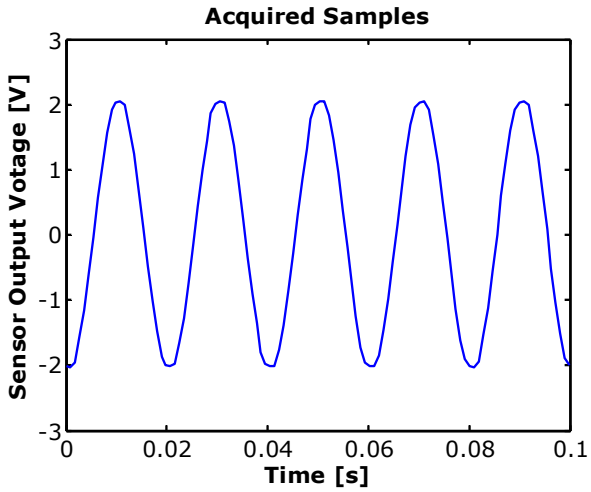

Fig. 5. Output voltage of power grid sensor, acquired at f_s≈1111 kS/s. Only the first five periods are represented.

In Fig. 6, the results obtained from the FFT of the acquired record are shown (red line). Also plotted are the amplitudes of the estimated harmonics obtained with the new efficient implementation proposed in this paper (limited to 11 harmonics due to the low value of the sampling frequency). The results show a good agreement between both methods.

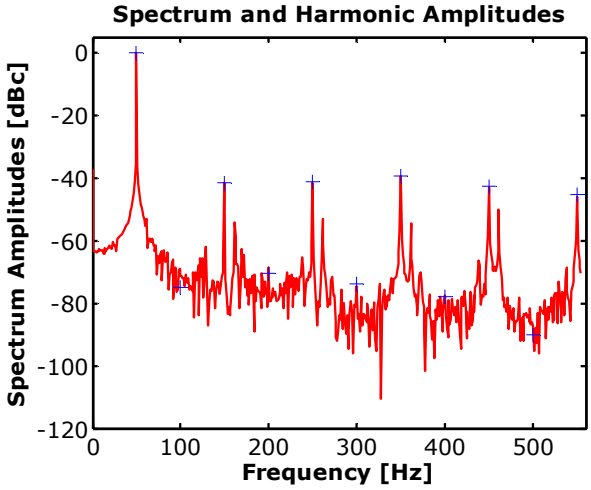

Fig. 6. Spectrum and harmonic amplitudes of the acquired signal. Red line represents the FFT results obtained for 1 000 samples while the blue crosses represent the harmonic amplitudes estimated with the efficient implementation algorithm for 1 000 samples and 11 harmonics (those that fit up to f_s/2)

The situation depicted in Fig. 6 corresponds to practically no spectral leakage and therefore the FFT results are quite good when compared with the results from the multiharmonic efficient implementation. In Fig. 7, the results for a situation with heavy spectral leakage are presented. In this situation, the multiharmonic fit has a very good agreement with a maximum difference of 14 μV occurring for the sixth harmonic. On the other hand, for the FFT the maximum difference can reach up to 29 mV for the 2nd harmonic.

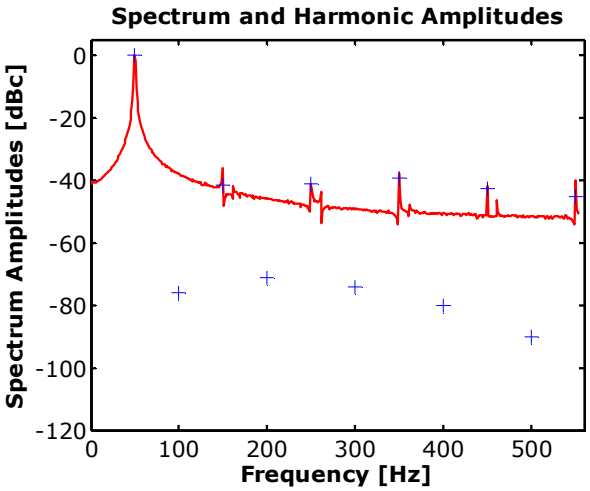

Fig. 7. Spectrum and harmonic amplitudes of the acquired signal. Red line represents the FFT results obtained for 988 samples while the blue crosses represent the harmonic amplitudes estimated with the efficient implementation algorithm for 988 samples and 11 harmonics (those that fit up to $f_s/2$)

5. CONCLUSIONS

A memory-wise efficient method of applying the multiharmonic fitting algorithm has been presented. It has been shown that the memory requirements of the proposed solution are significantly smaller than the requirements using the traditional approach.

A strategy to reduce the number of trigonometric function calculations was presented. Numerical simulations were used to analyze the performance of the presented memory efficient implementation.

REFERENCES

[1] IEEE Std. 1241-2011, *IEEE Standard for Terminology and Test Methods for Analog-to-Digital Converters*, New York, January 2011.

[2] IEEE Std. 1057-2007, *IEEE Standard for Digitizing Waveform Recorders*, New York, April 2008.

[3] Fernando M. Janeiro, Pedro M. Ramos, "Impedance measurements using genetic algorithms and multiharmonic signals", *IEEE Transactions on Instrumentation and Measurement*, vol. 58, n.º 2, pp. 383-388, February 2009.

[4] Pedro M. Ramos, A. Cruz Serra, "Impedance measurement using multiharmonic least-squares waveform fitting algorithm", *Computer Standards and Interfaces, Elsevier*, vol. 30, n.º 5, pp. 323-328, Julho 2008.

[5] R. Queirós, P. S. Girão, A. Cruz Serra, "Cross-correlation and sine-fitting techniques for high resolution ultrasonic ranging", *IEEE Instrumentation and Measurement Technology Conference (IMTC)*, pp. 552-556, April 2006.

[6] Tomáš Radil, Pedro M. Ramos, "Power quality detection and classification method for IEC 61000-4-30 Class A instruments", *IEEE I2MTC 2010 – International Instrumentation and Measurement Technology Conference,* Austin, EUA, pp. 691-696, May 2010.

[7] P. Handel, P. Zetterberg, "Receiver I/Q imbalance: Tone test, sensitivity analysis, and the universal software radio peripheral", *IEEE Transactions on Instrumentation and Measurement*, vol. 59, no. 3, pp. 704-714, March 2010.

[8] N. M. Vucijak, L. V. Saranovac, "A simple algorithm for the estimation of phase difference between two sinusoidal voltages", *IEEE Transactions on Instrumentation and Measurement*, vol. 59, no. 12, pp. 3152-3158, December 2010.

[9] P. M. Ramos, M. F. Silva, R. C. Martins, A. C. Serra, "Simulation and experimental results of multiharmonic least-squares fitting algorithms applied to periodic signals", *IEEE Transactions on Instrumentation and Measurement*, vol. 55, no. 2, pp. 646-651, April 2006.

[10] P. M. Ramos, A. C. Serra, "Least Squares Multiharmonic fitting: Convergence improvements", *IEEE Transactions on Instrumentation and Measurement*, vol. 56, no. 4, pp. 1412-1418, August 2007

[11] P. M. Ramos, F. M. Janeiro, T. Radil, "Comparison of impedance measurements in a DSP using ellipse-fit and seven-parameter sine-fit algorithms", *Measurement*, vol. 42, pp. 1370-1379, November 2009.