17th Symposium IMEKO TC 4, 3rd Symposium IMEKO TC 19 and 15th IWADC Workshop
Instrumentation for the ICT Era
Sept. 8-10, 2010, Kosice, Slovakia

Sine-Fitting Algorithms Implemented in 32-bit Floating Point Systems

Pedro M. Ramos[1], Tomáš Radil,[2] Fernando M. Janeiro[3]

[1] Instituto de Telecomunicações, Department of Electrical and Computer Engineering, Instituto Superior Técnico, Technical University of Lisbon, Av. Rovisco Pais 1, 1049-001 Lisbon, Portugal
Phone: +351-218418485; Fax: +351-218418472, e-mail: pedro.ramos@lx.it.pt
[2] Instituto de Telecomunicações, Av. Rovisco Pais 1, 1049-001 Lisbon, Portugal
e-mail: tomas.radil@lx.it.pt
[3] Instituto de Telecomunicações, Universidade de Évora, Departamento de Física, Rua Romão Ramalho, nº. 59, 7000-671 Évora, Portugal, Phone: +351-266745300; e-mail: fmtj@uevora.pt

Abstract-The implementation issues of the iterative four-parameter sine-fitting algorithm in DSPs and other devices with floating point processors with 32 bits (called binary 32) are analyzed in this paper. Direct comparison with a PC based algorithm implementation, where floating point numbers are represented with 64 bits (called binary 64), is performed.

I. Introduction

The role of small and portable instruments in both industry applications and scientific research has an ever growing importance. Examples range from electric machines condition monitoring [1] to devices developed for space applications [2] where there are size and power limitations. Other applications include power quality measurements [3], impedance measurements [4] and some biomedical applications [5]. In this class of instruments, data processing is normally performed using dedicated digital signal processors (DSP), in which the floating-point representation is limited, by the hardware, to a fixed number of bits [6]. Although this is true for any digital system, calculations performed in a computer are usually done with binary 64 precision (formerly known as double precision), while portable processing devices mostly handle binary 32 precision numbers (or single). Format binary 32 has 23 bits to represent the fraction part of the significand, 8 bits for the exponent and a sign bit, in a total of 32 bits. Similarly, binary 64 format has 52 bits for the fraction, 11 bits for the exponent and a sign bit [7].

Signal processing algorithms used to process the acquired data are usually developed and tested in personal computers. However, these algorithms are sometimes implemented in systems with lower floating-point precision than used in personal computers. Examples include sine-fitting algorithms, defined in [8] for ADC testing, which are used to estimate the parameters of an acquired sinewave. The effects of implementing these algorithms in a platform with lower floating-point precision were addressed in [6]. However, further details regarding convergence and implementation issues still need to be analyzed.

In this paper, the implementation, in a binary 32 floating-point system of the four-parameter sine-fitting algorithm is studied. The theoretical Crámer-Rao Lower Bounds (CRLB) of the sinewave parameter estimation were determined in [9]. These bounds can be used to assess the different algorithm implementations. Simulations are performed to evaluate the algorithms performance and limitations.

II. Sine-Fitting Algorithms

An acquired sinewave with N samples y_n, should fit the model

$$y(t) = C + D\cos(\omega t + \phi) = C + A\cos(\omega t) + B\sin(\omega t) \tag{1}$$

where $\omega = 2\pi f$ is the angular frequency, D is the amplitude and C is the DC component. Parameters A and B are the in-phase and quadrature components, respectively. If the frequency is accurately known, then the 3 parameter sine-fitting algorithm can be used to estimate the remaining three parameters. However, most times this is not the case, and therefore the frequency must be estimated alongside the three parameters. This can be accomplished by the 4 parameter sine-fitting algorithm, which is an iterative method based on a least-squares minimization process. In each iteration the estimated parameters are

$$\mathbf{x}^{(i)} = \left[A^{(i)} \;\; B^{(i)} \;\; C^{(i)} \;\; \Delta f^{(i)} \right]^T \tag{2}$$

where $\Delta f^{(i)}$ is the frequency correction on the ith iteration. These parameters are obtained through

$$\mathbf{x}^{(i)} = \left[\left[\mathbf{D}^{(i-1)} \right]^T \; \mathbf{D}^{(i-1)} \right]^{-1} \left[\left[\mathbf{D}^{(i-1)} \right]^T \; \mathbf{y} \right] \tag{3}$$

where $\mathbf{y} = \left[y_1 \; y_2 \; ... \; y_N \right]^T$ is a vector with the N acquired samples. $\mathbf{D}^{(i-1)}$ is an $N \times 4$ matrix

$$\mathbf{D}^{(i-1)} = \begin{bmatrix} \cos(\beta_1) & \sin(\beta_1) & 1 & \alpha_1 \\ \cos(\beta_2) & \sin(\beta_2) & 1 & \alpha_2 \\ \vdots & \vdots & \vdots & \vdots \\ \cos(\beta_N) & \sin(\beta_N) & 1 & \alpha_N \end{bmatrix} \tag{4}$$

with $\alpha_n = -2\pi A^{(i-1)} t_n \sin(\beta_n) + 2\pi B^{(i-1)} t_n \cos(\beta_n)$ and $\beta_n = \omega^{(i-1)} t_n$.
The initial estimate of the frequency parameter is obtained from the IpDFT [10] which is then used in the 3 parameter sine-fitting algorithm to obtain the initial estimate of the three amplitudes A, B and C. These estimates are then used in the next iteration to obtain new estimations of each parameter and a frequency correction Δf to the frequency from the previous iteration. Standard [8] defines that the algorithm has converged when the RMS fitting error between the acquired data and model (1) is below a predefined threshold. However, this threshold depends on the noise and distortion included in the signal, which is usually not known. An alternative approach consists in considering that the algorithm has converged when the relative frequency correction $\Delta f / f$ is below a predefined threshold (TH). This threshold depends on the desired frequency precision.
In the next subsections, two different approaches to the solution of (3) are described.

A. Implementation based on enhanced memory optimization

The memory requirements for the calculation of (4) can be very high when the number of samples is large. However, it is possible to reduce the amount of memory needed, as $\left[\mathbf{D}^{(i-1)} \right]^T \mathbf{D}^{(i-1)}$ can be reduced to a 4×4 matrix

$$\left[\mathbf{D}^{(i-1)} \right]^T \mathbf{D}^{(i-1)} = \begin{bmatrix} \sum_{n=1}^{N} \cos^2(\beta_n) & \sum_{n=1}^{N} \cos(\beta_n)\sin(\beta_n) & \sum_{n=1}^{N} \cos(\beta_n) & \sum_{n=1}^{N} \alpha_n \cos(\beta_n) \\ \sum_{n=1}^{N} \cos(\beta_n)\sin(\beta_n) & \sum_{n=1}^{N} \sin^2(\beta_n) & \sum_{n=1}^{N} \sin(\beta_n) & \sum_{n=1}^{N} \alpha_n \sin(\beta_n) \\ \sum_{n=1}^{N} \cos(\beta_n) & \sum_{n=1}^{N} \sin(\beta_n) & N & \sum_{n=1}^{N} \alpha_n \\ \sum_{n=1}^{N} \alpha_n \cos(\beta_n) & \sum_{n=1}^{N} \alpha_n \sin(\beta_n) & \sum_{n=1}^{N} \alpha_n & \sum_{n=1}^{N} \alpha_n^2 \end{bmatrix}, \tag{5}$$

while $\left[\mathbf{D}^{(i-1)} \right]^T \mathbf{y}$ is just a 4×1 matrix

$$\left[\mathbf{D}^{(i-1)} \right]^T \mathbf{y} = \left[\sum_{n=1}^{N} y_n \cos(\beta_n) \;\; \sum_{n=1}^{N} y_n \sin(\beta_n) \;\; \sum_{n=1}^{N} y_n \;\; \sum_{n=1}^{N} y_n \alpha_n \right]^T. \tag{6}$$

B. Implementation based on QR decomposition

Another alternative for the solution of (3) is the QR decomposition of matrix (4)

$$\mathbf{D}^{(i-1)} = \mathbf{Q}^{(i-1)} \begin{bmatrix} \mathbf{R}^{(i-1)} \\ \mathbf{0} \end{bmatrix} \tag{7}$$

where $\mathbf{R}^{(i-1)}$ is an upper triangular matrix with 4 rows and 4 columns and $\mathbf{Q}^{(i-1)}$ is an orthogonal matrix with N rows and N columns. As stated in [11], the use of QR decomposition is an appropriate method to avoid round-off errors of computing $\left[\mathbf{D}^{(i-1)}\right]^T \mathbf{D}^{(i-1)}$ which is also frequently ill-conditioned. From the QR decomposition, the estimated parameters are obtained by

$$\mathbf{x}^{(i)} = \left[\mathbf{R}^{(i-1)}\right]^{-1} \left[\mathbf{Q}^{(i-1)}\right]^T \mathbf{y}\,. \tag{8}$$

III. Numerical Results

In this section the simulated numerical results are presented. In order to assess the performance of the four-parameter sine-fit algorithm in binary 32 and binary 64 systems and with the two different implementations (enhanced memory and QR decomposition), four different versions of the algorithm were implemented in Matlab. In the binary 64 version, all variables are Matlab type double with 8 bytes. For the binary 32, all variables are converted using the *single* function of Matlab. Furthermore, all math operations are cast to binary 32 using the same function.
The first analysis deals with assessing the influence of the threshold parameter and the signal amplitude for constant number of samples and additive white Gaussian noise. For each pair of *TH* and *D*, 10000 repetitions were simulated with different noise vectors and initial phase. The average number of iterations, the experimental standard deviation of the estimated frequency error and phase error were registered for 1920 samples, signal frequency of 1 kHz, sampling rate of 96 kS/s and noise (n_{RMS}) of 10 mV. The results for the binary 64 enhanced memory algorithm implementation are shown in Figure 1. It can be seen that the average number of iterations does not exceed 4 and that the worst case scenario occurs for the lowest amplitude ($D = 0.1$ V) and strictest convergence condition ($TH = 10^{-10}$). Note that the experimental standard deviations do not depend on the *TH* value and that the best results are achieved for the highest amplitude (better signal to noise ratio).

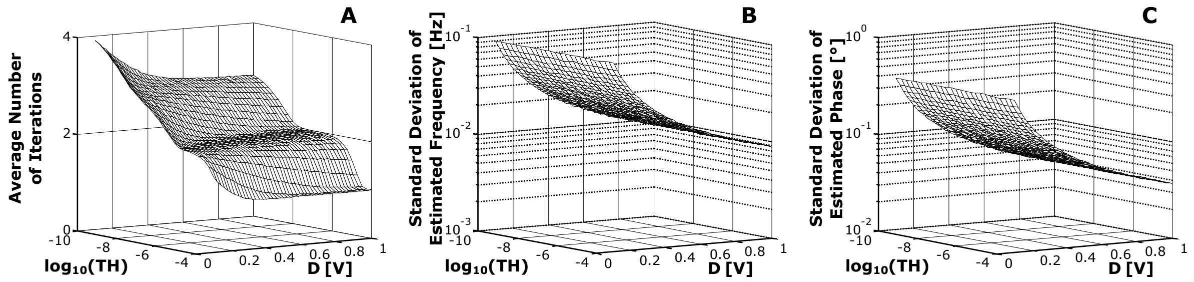

Figure 1. Simulation results obtained for binary 64 enhanced memory algorithm implementation with 1920 samples, $n_{RMS} = 10$ mV. In A, the average number of iterations is presented, while in B the standard deviation of the estimated frequency is shown and in C, the standard deviation of the estimated phase is represented. The three plots are shown as a function of the signal amplitude and the threshold criteria (*TH*).

In Figure 2 the results obtained with the binary 32 algorithm are presented for the same situation. The results of the experimental standard deviations are very similar to the ones obtained with the binary 64 version of the algorithm. The main difference is in the average number of iterations. Since the algorithm stops after 20 iterations (a predefined value to stop the algorithm in case of non-convergence) this value is reached for the strictest values of *TH*. Convergence problems occur for $TH < 10^{-7}$. However, as can be seen from the experimental standard deviations and the average error values (not shown but clearly well below the standard deviation values), the algorithm has reached the minimum region but failed to reach the threshold value. From our experience in the DSP implementation [4], what happens in this situation is that the frequency correction value is not enough to actually change the value of the frequency because of the limited number of bits for the frequency significand and also because the frequency correction is low (e.g., in binary 32, for $f = 1000$ Hz, $\Delta f = 10^{-5}$ Hz, $f_{new}=f+\Delta f$, $f_{new}-f = 0$ Hz while for binary 64 the Δf value that does not change the frequency is 10^{-14}).

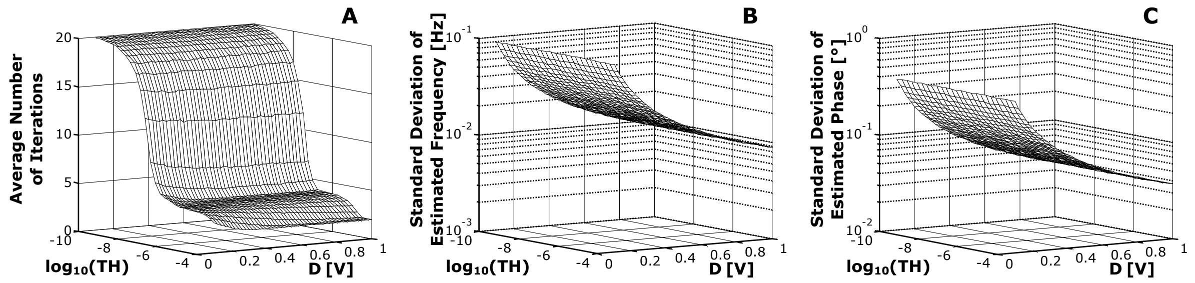

Figure 2. Simulation results obtained for binary 32 enhanced memory algorithm implementation. All parameters correspond to the ones used in Figure 1. The maximum number of iterations (20) is reached for the strictest *TH* situations.

In Figure 3, the results obtained for the binary 32 QR decomposition implementation are shown for the same situations. Note that the results are identical to the ones obtained with the enhanced memory implementation (Figure 2). The results for the binary 64 QR decomposition implementation (not shown) are also identical to the ones shown in Figure 1. However, the algorithm execution times are very different. For these results, the QR decomposition implementation took roughly 100 times the time needed for the enhanced memory algorithm implementation.

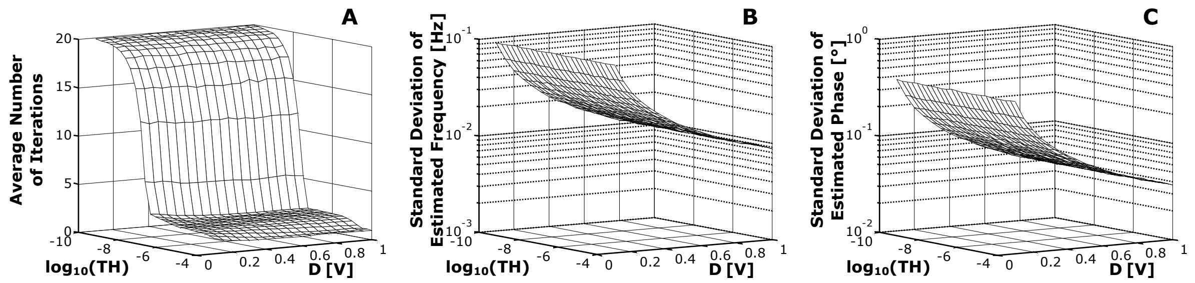

Figure 3. Simulation results obtained for binary 32 QR decomposition algorithm implementation. All parameters correspond to the ones used in Figure 1 and Figure 2.

In the second analysis, the influence of the signal to noise ratio is analyzed for values between 0 dB and 100 dB with constant signal amplitude $D = 1$ V. Again, 10000 repetitions were simulated for each SNR, *TH* value set. The results obtained for the binary 64 version of the algorithm are shown in Figure 4, while the results from the binary 32 algorithm are represented in Figure 5. The effect of the limited resolution of the binary 32 algorithm is also visible for the strictest values of *TH*. In addition, there is also a difference noticeable for the higher values of SNR. For SNR > 90 dB the values of the experimental standard deviations tend to stabilize for all values of *TH*. For the frequency, the experimental standard deviation does not fall below 4×10^{-5} which is near the resolution for the 1000 Hz frequency value in binary 32 representation. The results obtained with the binary 32 QR decomposition algorithm are presented in Figure 6 and the results show only a slight improvement for high SNRs when compared with the results from the enhanced memory algorithm implementation (Figure 5).

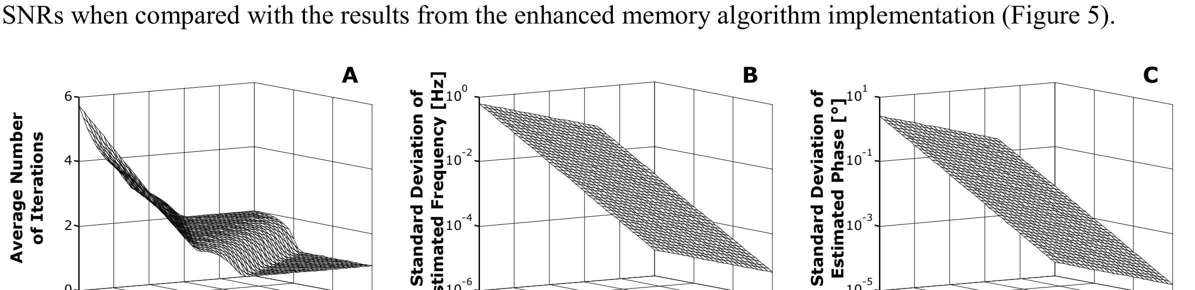

Figure 4. Simulation results obtained for binary 64 enhanced memory algorithm implementation with $D = 1$ V, 1920 samples. In A, the average number of iterations is presented, while in B the standard deviation of the estimated frequency is shown and in C, the standard deviation of the estimated phase is represented.

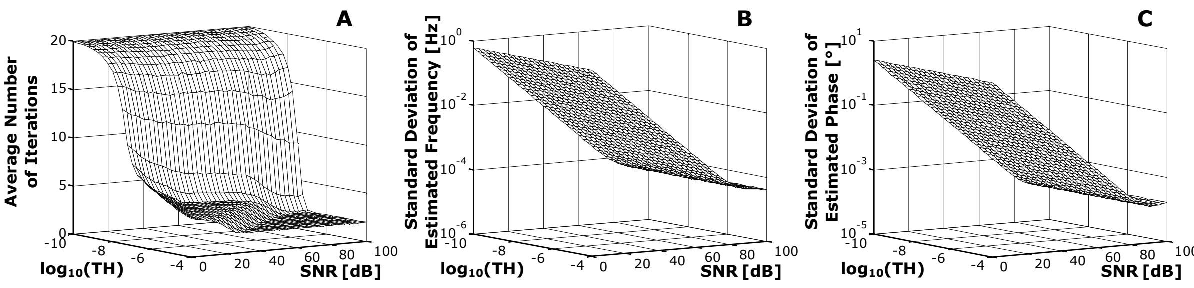

Figure 5. Simulation results obtained for binary 32 enhanced memory algorithm implementation.

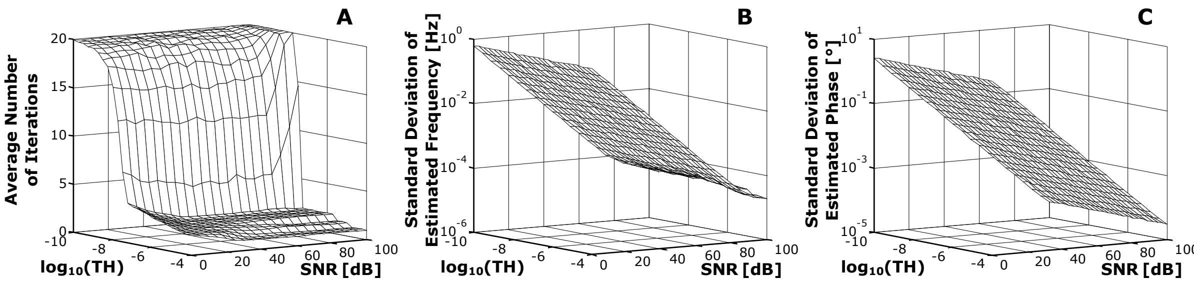

Figure 6. Simulation results obtained for binary 32 QR decomposition algorithm implementation.

The third and final analysis included in this paper concerns the influence of the number of samples in the record. The number of samples is changed from 1000 up to 1000000 while the threshold range is similar to the previous analysis. In Figure 7, the results obtained in the binary 64 enhanced memory algorithm implementation are presented. Notice that the average number of iterations does not exceed 3 while the experimental standard deviations show, as expected, a monotonic decrease as the number of samples in the record increases.

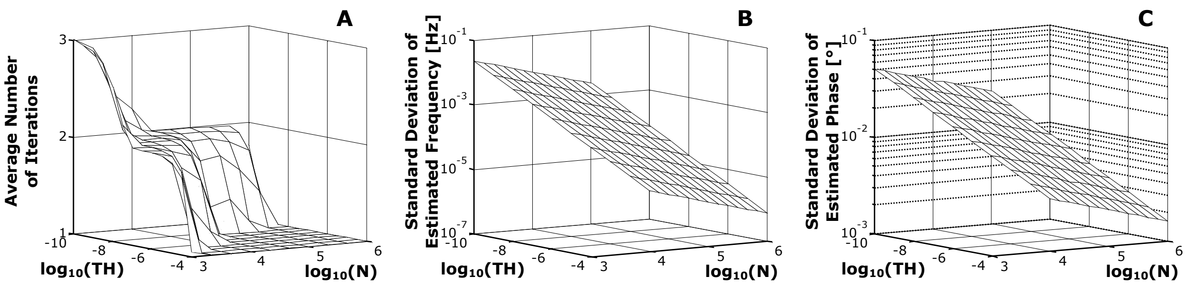

Figure 7. Simulation results obtained for binary 64 enhanced memory algorithm implementation with $D = 1$ V, $n_{RMS} = 10$ mV. In A the average number of iterations is presented, while in B the standard deviation of the estimated frequency is shown and in C, the standard deviation of the estimated phase is represented. The three plots are shown as a function of the number of samples and the threshold criteria (*TH*).

The results obtained with the binary 32 enhanced memory algorithm implementation are shown in Figure 8. As in the previous analysis, the average number of iterations reaches the maximum (currently set to 20) for the lowest values of *TH*. The experimental standard deviation of the estimated frequency has a monotonic behavior with the increase in the number of samples but only until about $N = 10^5$. For a higher number of samples, the experimental standard deviation remains practically constant. The explanation for this behavior is directly connected with the lowest value of the standard deviation that can be reached for f=1000 Hz (which is near 4×10^{-5}) with binary 32 resolution.

In the case of the experimental standard deviation of the phase error, the behavior is even more different from the behavior observed with the binary 64 algorithm (Figure 7 C). In fact, the initial monotonic decrease is only valid until about $N = 10^5$. Afterwards, what can be observed is a steep increase as the number of samples increases. This effect is still not explained, however, it is believed to be caused by the rapid increase in the fourth row, fourth column element of (5).

Unfortunately, it is not possible to compare these results with the results from the QR decomposition method since the increase in the number of samples makes this algorithm extremely slower due to the increasing memory requirements (the QR decomposition is applied to a N×4 matrix). In fact, in the Matlab PC based simulation the number of samples cannot exceed 100 000 (even with this value, the time it takes to simulate one single situation is near 7 hours).

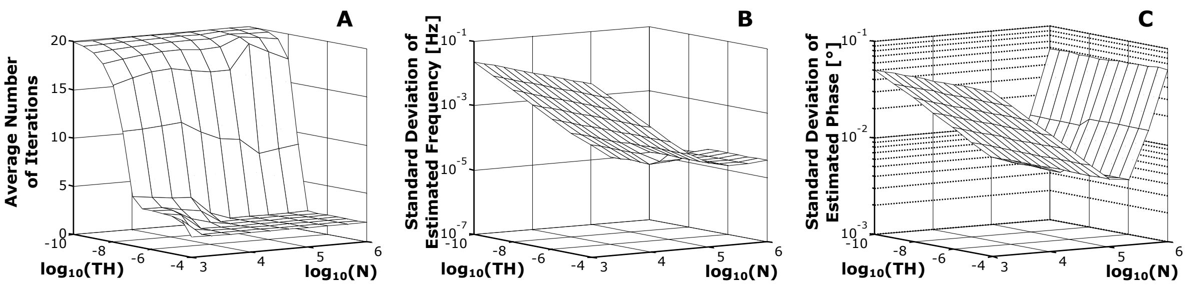

Figure 8. Simulation results obtained for binary 32 enhanced memory algorithm implementation.

IV. Conclusions

In this paper an analysis of the performance of the iterative four-parameter sine-fitting algorithms in binary 64 and binary 32 systems for two different implementations has been presented. Specific emphasis on the average number iterations was given as a direct measure of convergence issues that arise in binary 32 systems. The standard deviation of the estimated frequency and phase were also used to assess the performance of the various implementations. For the binary 32 enhanced memory implementation, the results are almost identical to the binary 64 version, except when the standard deviations falls below the binary 32 resolution.
Although the binary 32 QR decomposition method does not suffer from these shortcomings, the processing time and memory requirements are extremely high when compared to the enhanced memory method. As a direct result of these substantially different memory requirements, the QR decomposition cannot be applied to records above 10000 samples even on a PC, while the enhanced memory implementation can be applied easily.

Acknowledgements

Work sponsored by the Portuguese national research project reference PTDC/EEA-ELC/72875/2006.

References

[1] J. Rangel-Magdaleno, R. Romero-Troncoso, R. A. Osornio-Rios, E. Cabal-Yepez, L. M. Contreras-Medina, ”Novel Methodology for Online Half-Broken-Bar Detection on Induction Motors”, *IEEE Trans. Instrum. Meas.*, Vol, 58, No. 5, pp. 1690-1698, May 2009.
[2] R. Hourani, R. Jenkal, W. Rhett Davis, W. Alexander, “Automated Design Space Exploration for DSP Applications”, *J. Sign. Process. Syst.*, vol. 56, pp. 199-216, 2009.
[3] T. Radil, P. M. Ramos, F. M. Janeiro, A. C. Serra "PQ Monitoring System for Real-Time Detection and Classification of Disturbances in a Single-Phase Power System", *IEEE Trans. Instrum. Meas.*, Vol. 57, No. 8, pp. 1725 - 1733, August 2008.
[4] 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, No. 9, pp. 1370 - 1379, November 2009.
[5] P. Arpaia, F. Clemente and C. Romanucci, “An instrument for prosthesis osseointegration assessment by electrochemical impedance spectrum measurement”, *Measurement*, vol. 41, n° 9, pp. 1040-1044, Nov. 2008.
[6] Bernard Widrow, István Kollár, *Quantization Noise*, Cambridge University Press 2008.
[7] IEEE Std. 754-2008, *IEEE Standard for Floating-Point Arithmetic*, The Institute of Electrical and Electronic Engineers, New York, August 2008.
[8] IEEE Std. 1057-2007, *IEEE Standard for Digitizing Waveform Records*, The Institute of Electrical and Electronic Engineers, New York, December 2007.
[9] P. Händel, “Properties of the IEEE-STD-1057 four-parameter sine wave fit algorithm,” *IEEE Trans. Instrum. Meas.*, vol. 49, no. 6, pp. 1189–1193, Dec. 2000.
[10] H. Renders, J. Schoukens and G. Vilain, “High-accuracy spectrum analysis of sampled discrete frequency signals by analytical leakage compensation” *IEEE Trans. Instrum. Meas.*, vol. 33, no. 4, pp. 287-292, Dec. 1984.
[11] Gene H. Golub and Charles F. van Loan, *Matrix Computations*, 3rd ed., The John Hopkins University Press, 1996.