Advances in Electrical and Computer Engineering ***Volume 12, Number 2, 2012***

A Method for Designing FIR Filters with Arbitrary Magnitude Characteristic Used for Modeling Human Audiogram

Erwin SZOPOS, Marius NEAG, Ioana SĂRĂCUŢ, Horia HEDEŞIU, Lelia FEŞTILĂ
Technical University from Cluj-Napoca, 400114, Romania
Marius.Neag@bel.utcluj.ro

***Abstract*—This paper presents an iterative method for designing FIR filters that implement arbitrary magnitude characteristics, defined by the user through a set of frequency-magnitude points (frequency samples). The proposed method is based on the non-uniform frequency sampling algorithm. For each iteration a new set of frequency samples is generated, by processing the set used in the previous run; this implies changing the samples location around the previous frequency values and adjusting their magnitude through interpolation. If necessary, additional samples can be introduced, as well. After each iteration the magnitude characteristic of the resulting filter is determined by using the non-uniform DFT and compared with the required one; if the errors are larger than the acceptable levels (set by the user) a new iteration is run; the length of the resulting filter and the values of its coefficients are also taken into consideration when deciding a re-run.**

To demonstrate the efficiency of the proposed method a tool for designing FIR filters that match human audiograms was implemented in LabVIEW. It was shown that the resulting filters have smaller coefficients than the standard one, and can also have lower order, while the errors remain relatively small.

***Index Terms*—discrete Fourier transforms, error analysis, FIR filter, interpolation and non-uniform sampling.**

I. INTRODUCTION

An important subclass of filter synthesis problems is the design of a filter that matches a set of magnitude response specification without phase consideration. Several solutions to this problem have been proposed for digital filters but most of them introduce additional constraints to simplify the design, thereby restricting their application [1].

FIR filters are particularly well suited for this application as they can be easily constrained to have linear phase. But one needs to monitor the length of the filter, for using a large number of taps leads not only to implementation problems but also introduces delays that are not acceptable for some applications, such as audio signal processing.

This paper presents an iterative method for designing FIR filters that realize arbitrary magnitude characteristics, defined by the user through a set of magnitude-frequency points – usually referred to as frequency samples. The proposed method is based on the non-uniform frequency sampling algorithm [2], [3]. For each iteration a new set of frequency samples is created starting from the previous set, by changing the frequency position of the samples and adjusting their magnitude through interpolation; if necessary additional samples are introduced. After each run the magnitude characteristic of the resulting filter is compared against the target set by the user; if the errors are above the level set by the user a new design iteration starts.

The proposed method yields filters that are easier to implement than the standard designs: first, unpractical solutions, where filter parameters take unacceptable large values, are avoided by built-in checks; next, the filter order is decreased iteratively, until the optimum – defined by the user – trade-off between magnitude errors, the filter order and the maximum value of its coefficients is found. The method is based on FIR filters, which have an inherently linear phase characteristic and a symmetrical impulse response, so there is no need for group delay equalizers [4].

The theoretical background for the filter design and the analysis tools employed here is presented in Section II; the proposed method is described in Section III, including its implementation by using the LabVIEW environment. Section IV presents an important application of this method: the design of FIR filters able to match human audiograms as provided by audiologists [5], [6]. Such a filter provides an effective model of hearing impediments that can be used to develop tailored hearing aids [7], [8]. The difference between the target magnitude values and the ones corresponding to the same frequencies in the obtained magnitude response is computed, showing that the characteristics of the designed filter match well the audiogram. Section V presents the conclusions of the paper.

II. FIR FILTER DESIGN – THEORETICAL BACKGROUND

A. *A frequency sampling method for designing FIR filters*

In order to approximate a continuous frequency response, one can sample uniformly or non-uniformly the frequency response at N points along the normalized frequency range $(0,\ 2\pi)$, (measured in rad/sec); the approximate frequency response will then be obtained as an interpolation of the sampled frequency response [8], [9]. The approximation error would be exactly zero at the sampling frequencies and finite between them. The smoother the approximated frequency response is, the smaller the error of interpolation between the sample points.

There are several methods for designing digital filters able to approximate a desired frequency response defined by a set of frequency samples. A standard design method for FIR filters is described below [2]. There are four types to consider, each with its own design equation:

This work was supported by the Romanian University Research Council CNCSIS, within the PNII–IDEI framework, grant no. 2534 / 2008.

Digital Object Identifier 10.4316/AECE.2012.02009

Type I:

$$A_r(\omega_k) = h\left(\frac{N-1}{2}\right) + 2\sum_{n=0}^{\frac{N-3}{2}} h(n)\cdot cos\left[\omega_k\left(\frac{N-1}{2}-n\right)\right] \quad (1)$$

Type II:

$$A_r(\omega_k) = 2\sum_{n=0}^{\frac{N}{2}-1} h(n)\cdot cos\left[\omega_k\left(\frac{N-1}{2}-n\right)\right] \quad (2)$$

Type III:

$$A_r(\omega_k) = 2\sum_{n=0}^{\frac{N-3}{2}} h(n)\cdot sin\left[\omega_k\left(\frac{N-1}{2}-n\right)\right] \quad (3)$$

Type IV:

$$A_r(\omega_k) = 2\sum_{n=0}^{\frac{N}{2}-1} h(n)\cdot sin\left[\omega_k\left(\frac{N-1}{2}-n\right)\right] \quad (4)$$

where $A_r(\omega_k)$ is the desired real-valued frequency response, $h(n)$ represents the filter parameters and N is the total number of samples. Each of (1)-(4) represents a set of linear equations for determining the parameters of the FIR filter. In general the values of the frequencies ω_k and $A(j\omega_k)$ can be chosen arbitrarily. An effective way of dealing with (1)-(4) is to use the matrix form of these equations:

$$\mathbf{V}\cdot\mathbf{h} = \mathbf{A}_r \quad (5)$$

where **V** is the Vandermonde matrix computed as follows:

Type I:

$$V_{ij} = \begin{cases} 2cos\left[\omega_i\left(\frac{N-1}{2}-j\right)\right], j \neq \frac{N-1}{2} \\ 1 \quad , j = \frac{N-1}{2} \end{cases} \quad (6)$$

Type II:

$$V_{ij} = 2cos\left[\omega_i\left(\frac{N-1}{2}-j\right)\right] \quad (7)$$

Type III, IV:

$$V_{ij} = 2sin\left[\omega_i\left(\frac{N-1}{2}-j\right)\right] \quad (8)$$

where i and j are the row and the column indexes defined as:

$$\begin{cases} i = \overline{0,\frac{N-1}{2}};\ j = \overline{0,\frac{N-3}{2}};\ Type\ I \\ i = j = \overline{0,\frac{N-3}{2}};\ Type\ III \\ i = j = \overline{0,\frac{N}{2}-1};\ Type\ II, IV \end{cases} \quad (9)$$

Terms **h** and $\mathbf{A}_r$ in (5) are the impulse response (here giving the filter parameters/coefficients, as well) and the magnitude vectors, respectively, expressed as follows (N odd):

$$\mathbf{h} = \begin{pmatrix} h(0) \\ h(1) \\ \vdots \\ h\left(\frac{N-1}{2}\right) \end{pmatrix}, \ \mathbf{A}_\mathrm{r} = \begin{pmatrix} A_r(\omega_0) \\ A_r(\omega_1) \\ \vdots \\ A_r\left(\omega_{(N-1)/2}\right) \end{pmatrix} \quad (10)$$

One can obtain the filter coefficients, **h**, by solving the linear matrix equation (5); this implies determining the inverse of the **V** matrix, for which the determinant of the Vandermonde matrix, *det(***V***)*, is required. Several effective ways of calculating *det(V)* have been proposed [10], [11].

The frequency sampling design methods, based on uniform and non-uniform sampling, can be used for any given magnitude response; they are particularly useful for the design of non-prototype filters where the desired magnitude response can take an irregular shape, beyond the four types considered in (1)-(4).

B. FIR filter analysis by using non-uniform DFT

A key stage in filter design is the analysis of the synthesized filter and the evaluation/computation of the errors (differences with respect to the target characteristic). For generality, the case of non-equispaced arrangement of the magnitude samples is considered here and the non-uniform DFT (NDFT) is used [12], [13]. This is a general form of DFT that can be employed to evaluate the frequency samples at N arbitrary but distinct points in the z-plane. If the samples can be taken at non-equispaced intervals both in time (t_n) and frequency (f_k), the NDFT is defined as follows:

$$A(\omega_k) = \sum_{n=0}^{N-1} h(t_n)\cdot e^{-j\omega_k t_n} \quad (11)$$

For the case envisaged here the samples are taken non-uniformly in the frequency domain but uniformly in the time domain. Considering t_n defined only for discrete values nT_s, where T_s is the sampling period, eq. (11) becomes:

$$A(\omega_k) = \sum_{n=0}^{N-1} h(n)\cdot e^{-j\omega_k n} \quad (12)$$

Two differences between DFT and NDFT should be noted. First, the samples in normalized frequency are taken at intervals $2\pi/T$ in the non-uniform case instead of $2\pi/N$ in the uniform case, where T is the duration of the signal $h(t)$, $t \in [0,T]$ and N is the number of samples. The second difference is the integer index n in the uniform case instead of t_n in the non-uniform case.

Once the magnitude response of the designed filter is obtained by using NDFT it is compared against the required characteristic; the main error metric used by the iterative optimization process proposed in the next Section is the *sample-by-sample error* (SSE): the difference between the desired magnitude values, A_r, and the ones corresponding to the same frequencies in the obtained magnitude response, A:

$$SSE(k) = A_r^{dB}(\omega_k) - A^{dB}(\omega_k) \quad (13)$$

III. PROPOSED METHOD FOR DESIGNING FIR FILTERS WITH ARBITRARY MAGNITUDE CHARACTERISTIC

A. Theoretical description of the proposed method

A major drawback of the standard design method presented in Section II.A is that if the distance between samples varies over a large range the determinant *det(***V***)* required to solve (5) will have a small value; therefore the coefficients of the resulting filter will take high values that are not practical to implement. To avoid this a minimum value for *det(***V***)* should be imposed during the design.

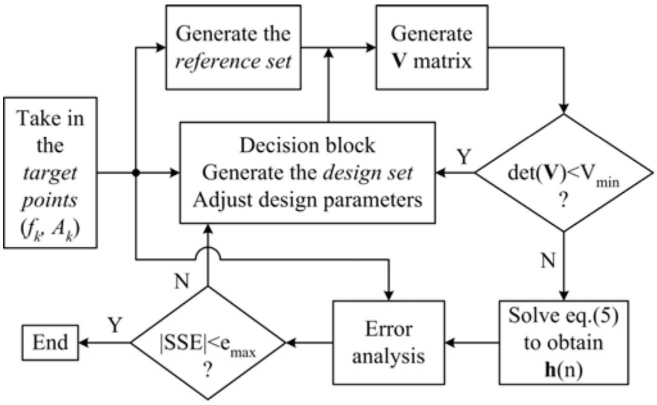

Figure 1. Block diagram of the proposed method for designing a FIR filter that matches the required magnitude characteristic A_k - f_k.

Figure 1 shows the block diagram of the proposed method for designing arbitrary magnitude response FIR filters. The first step is to collect the magnitude samples at given frequencies and normalize the frequency values; this way the *reference set* of samples is obtained.

The second step is to design the matching FIR filter, with a magnitude response that fits the reference set, by using the non-uniform frequency sampling algorithm (5)-(10): the process starts with the generation of the matrix **V** defined by (6)-(9), followed by the computation of its determinant, then of the inverse of matrix **V**; this allows the solving of (5) for obtaining the filter parameters, *h(n)*.

The determinant of the **V** matrix is used as a first measure for assessing the suitability of the designed filter: if the value of *det(**V**)* is smaller than the threshold V_{min} (set by the user) then the filter parameters will take unacceptably high values. In this case the filter has to be re-designed, starting from a new set of frequency samples; these samples – called hereafter the *design set* – are obtained by processing the reference frequency samples (or the design set of samples used in the previous iteration) by repositioning them and/or increasing their number. This step requires two operations:

- changing the sample position on the frequency axis, so that the distance between samples is reduced for all or some frequency intervals.
- computing the magnitude corresponding to the new frequency location of each sample in the design set, by using an interpolation method based on the reference set of samples.

Sample repositioning aims at decreasing the unevenness of the frequency intervals between samples, which results in a higher value of *det(**V**)*. If *det(**V**)* is still lower than V_{min} the process is repeated for a new arrangement of samples, driven by the decision block, which can imply the introduction of additional frequency samples. Increasing the number of samples in the design set is a last-resort option, as it leads to filters with a larger number of taps.

The new position on the frequency axis of the samples within the design set can be decided by using various algorithms; the simplest is the uniform placement given by:

$$\Delta f = \frac{f_{max} - f_{min}}{M - 1} \tag{14}$$

where Δf is the distance between the frequency samples, f_{min} and f_{max} are the frequency interval limits, and M is the number of samples in the $(0, \pi)$ normalized frequency range, derived from the number of samples in the $(0, 2\pi)$ range, N:

$$M = \frac{N+1}{2} \text{ for } N \text{ odd} \; ; \; M = \frac{N}{2} \text{ for } N \text{ even} \tag{15}$$

Various strategies for non-uniform positioning of samples can be devised, from arbitrary insertion to optimization procedures based on adaptive and genetic algorithms. In this work the decision block inserts a few additional samples, targeting the frequency intervals where the SSE value or the distance between the reference samples is above the average. Numerous interpolation algorithms for deriving the magnitude of the samples in the design set are available; here the linear and the cosine interpolation methods were used, defined by the following expressions, respectively:

$$A_{k\_lin} = (1 - m) \cdot A_1 + m \cdot A_2 \tag{16}$$

$$A_{k\_cos} = \left(1 - \frac{1 - \cos(\pi \cdot m)}{2}\right) \cdot A_1 + \frac{1 - \cos(\pi \cdot m)}{2} \cdot A_2 \tag{17}$$

where A_k is the magnitude interpolated between the given values A_1 and A_2; m is defined as:

$$m = \frac{f_k - f_1}{f_2 - f_1} \tag{18}$$

with the condition for the frequencies $f_1 < f_k < f_2$. The decision block determines which of the available algorithms is used.

The third step of the proposed method involves the analysis of the designed filter with the aim of deriving the errors with respect to the reference set of samples. Its core is the error analysis block in Fig. 1, which takes in the current filter parameters, *h(n)*, and the reference frequency samples, $A_r(f_k)$ and outputs the SSE. The magnitude response of the filter is obtained by using NDFT as shown in (12); then the difference between the reference frequency samples and the corresponding magnitude values from the obtained characteristic is computed, yielding the SSE values (13); finally, the peak value of the SSE is computed and compared against a threshold set by the user, e_{max}.

If the SSE values are within the range set by the user the design process ends successfully and the current *h(n)* vector gives the final filter parameters; if this condition is not met then a new iteration is necessary. Based on the SSE values and considering the previous iterations the decision block generates a new set of design samples – by introducing additional samples and/or by using another interpolation algorithm – and/or it can adjust some of the design parameters, from V_{min} to the width of the range around the initial frequency each sample can be repositioned within.

The process can be repeated after reducing the number of frequency samples, in order to obtain a lower order filter.

B. LabVIEW implementation of the building blocks

The LabVIEW implementation of the main sections of the proposed design method shown in Fig. 1 is described in the followings. LabVIEW was chosen because it provides some of the required blocks – such as the *DFT, det(V), solve linear equations* – as well as an easy way to realize user-friendly graphic interfaces.

To generate the reference set of frequency samples one need to perform frequency normalization on the samples given by the user. The virtual instrument (*vi*) that performs this task is shown in Fig. 2; the reference frequency for normalization – called hereafter the *sampling frequency* – is computed by using the following expression:

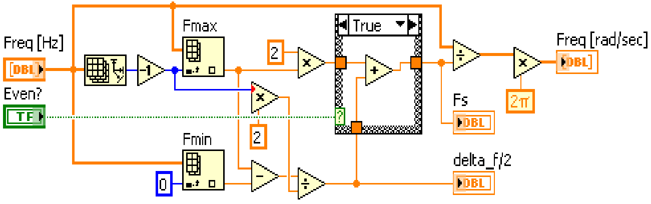

Figure 2. The frequency axis normalization (N even).

$$f_s = \begin{cases} 2 \cdot f_{max}, \; if \;\; N \; is \; odd \\ 2 \cdot f_{max} + \Delta f, \; if \;\; N \; is \; even \end{cases} \tag{19}$$

where f_{max} is the last frequency from the first half-band and Δf is defined in (14). The expression for f_s when N is even is determined such that $f_s > 2 \cdot f_{max}$. One need to differentiate between the N odd/even cases in order to avoid superposition, hence the two branches in (19); Δf is the minimum quantity necessary for this; its value was determined first experimentally, and then formalized in (14). Larger Δf results in higher values for the filter parameters.

The reference set of frequency samples, in *rad/s* units, is obtained by dividing the initial samples (defined by the user) by the sampling frequency (19) then multiply them by 2π.

The *vi* used to generate the Vandermonde matrix (6)-(9) is presented in Fig. 3. It takes in the normalized frequency samples, the reference set for the first iteration, and then the design set. The *vi* consists of two *for loops*, four *case structure* and the associated control logic. The internal *for loop* computes iteratively the elements for a given row while the row number is indexed by the external *for loop*; the *case structures* are controlled according to the number of samples and the desired type of FIR filter (9). The value of *det(***V***)* is obtained by using the LabVIEW *determinant* function.

Figure 4 shows the *vi* that finalizes the generation of the design set of frequency samples, after the samples used in the previous iteration have been repositioned in frequency and/or new frequency positions have been allocated for additional samples. In every iteration of the *while loop*, three checks are performed: (a) the comparison of the current frequency f_{k_re} from the repositioned vector with the next value f_{k+1} from the original vector; (b) the search (*Search 1D Array*) of f_{k_re} in the original frequency vector; and (c) determination of the reset condition for the *while loop*.

In check (a) if $f_{k\_re} < f_{k+1}$, than the case with interpolation is selected and the interpolated magnitude value will be inserted at the same index k_re as of the frequency being tested f_{k_re}. When $f_{k\_re} > f_{k+1}$, the magnitude vector remains unchanged and will be reused in the next iteration by a *shift register* created on the *for loop*. If $f_{k\_re} = f_{k+1}$ (check (b)) then the corresponding magnitude value at index $k+1$ is copied from the original magnitude vector in the interpolated one at the same index as f_{k_re}. In check (c) the *while loop* will be reset whenever one of the following conditions becomes true: $f_{k\_re} < f_{k+1}$, $f_{k\_re} = f_{k+1}$ and $f_{k\_re} = f_{max}$, where f_{max} has the same significance as in (14).

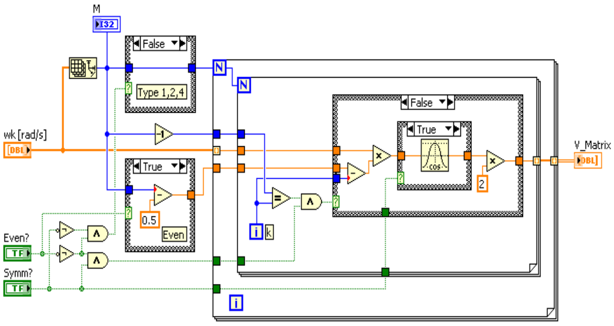

Figure 3. The *vi* that generates the **V** matrix, according to (6)-(9).

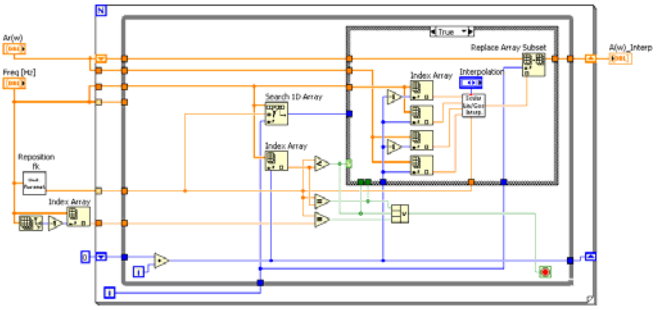

Figure 4. The $f_{k\_re} < f_{k+1}$ case of the *vi* that calculates the magnitudes of the frequency samples in the design set.

The filter parameters are obtained by using the LabVIEW function *solve linear equations* after substituting in (5) the **V** matrix created in Fig. 3 in the design set obtained in Fig. 4.

The core of the error analysis section is the *vi* that calculates the NDFT, presented in Fig. 5. It is a fairly straightforward implementation of (12) by using two *for loops* and their associated logic. Despite it being a general-purpose block, with numerous applications, the NDFT is not available as a ready-to-use function in LabVIEW, nor in other popular languages such as Matlab and Mathematica.

In order to reduce the computational requirements all the iterative operations described in Fig. 1 are performed considering only the *(0, π)* normalized frequency range. After the final iteration, the obtained set of filter parameters is extended to the full *(0, 2π)* range, by taking advantage of the FIR filter symmetry. First, the frequency positions for the *(π, 2π)* range are introduced in the frequency vector by using the *vi* presented in Fig. 6: it computes the new frequency positions based on the known ones, corresponding to the *(0, π)* interval, by using the fact that the intervals between samples when moving from *0* to *π* counterclockwise on the unit circle in the *z*-plane are equal with the differences between samples when moving clockwise from *0* to *π*.

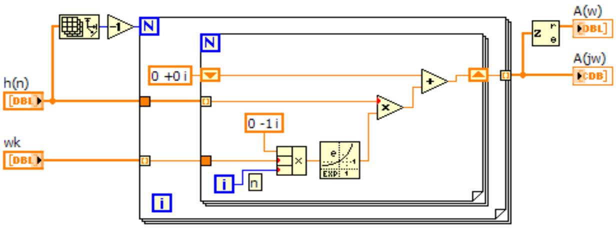

Figure 5. The *vi* that performs NDFT according to (12).

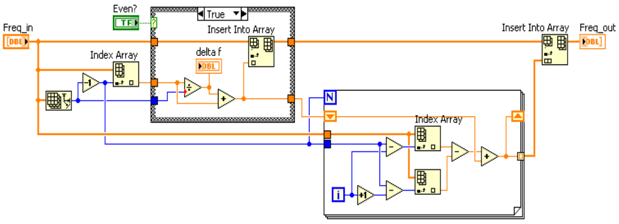

Figure 6. The full band frequency vector generation (N even).

To extend the magnitude vector accordingly one can simply concatenate the existing vector (corresponding to the $0\div\pi$ positions in the frequency vector) with a reversed version of itself; this way, the last position in the extended magnitude vector holds the same value as the first, the last-but-one position will hold the same value as the second, etc.

IV. DESIGN EXAMPLE: FIR FILTER THAT MATCHES A GIVEN HUMAN AUDIOGRAM

This section describes the design of several FIR filters that match a given (arbitrary) human audiogram, using both the standard and the proposed methods. These filters model the frequency response of the human hearing defined by the audiogram and allow the design of hearing aids [8].

There are eleven standard frequency values where audiologists test the hearing of the patients: *125, 250, 500, 750, 1k, 1.5k, 2k, 3k, 4k, 6k* and *8k Hz* [7], [14]. Note that these values are non-uniformly spaced, being an octave apart. The second column of Table I gives the corresponding magnitudes from a real-life audiogram; with respect to the design method described by Fig. 1, the first two columns of Table I give the magnitude-frequency points set by the user.

Let us first use the standard method described in Section II.A to design the matching FIR filter. The first row of Table II gives the $det(\mathbf{V})$ values for the four filter types (1)-(4) obtained with this method. The continuous-line plot in Fig. 7 shows the magnitude response of the type-I filter; as expected, the SSE values are practically nil for all the *11* frequency points of the audiogram. The impulse response of this filter is given by the continuous line plot in Fig. 8. As expected due to the low value of $det(\mathbf{V})$, the spread of filter coefficients values is very large: seven order of magnitude. Their extreme values are detailed in the 2nd row of Table II.

Next, the method described in Section III was used to design a FIR filter based on the same number of samples, *11*; the design parameters were set as follows: $V_{min}=1E+3$, $e_{max} = 0.5dB$. A higher precision is not usually required, as the audiogram comes from subjective measurements, with a resolution of *1-2dB*. The samples were first placed at equal intervals (14) then repositioned iteratively by the decision block. The 3rd column of Table I presents the SSE values of the obtained type-I filter; the values of $det(\mathbf{V})$ are given by the 3rd row of Table II, while the magnitude and impulse responses are depicted by the dotted-line plots in Fig. 7 and Fig. 8, respectively. Clearly, this filter provides a better trade-off between precision ($SSE_{max} = 0.5dB$) and the range of the filter coefficients ($|h|_{max}/|h|_{min}=7.46$) than the filter obtained by using the standard method ($|h|_{max}/|h|_{min}=1E+7$).

To simplify the implementation even further the design procedure was repeated targeting filters with lower orders. The number of frequency samples was reduced gradually while the design parameters V_{min} and e_{max} were relaxed accordingly. The 4th and 5th columns of Table I give the SSE values obtained for filters obtained by using *8* and *6* samples, respectively. The values of $det(\mathbf{V})$ and the extreme values of the filter coefficients for these cases are detailed in the 7th to 9th and10th to 12th rows of Table II. The magnitude and impulse responses of the FIR filter obtained for the 8-samples case are depicted by the continuous-line plots in Fig. 9 and Fig. 10, respectively. The corresponding plots for the filter based on 6 samples are depicted with dotted lines.

TABLE I. THE TARGET AUDIOGRAM AND THE SSE OF THE MATCHING FILTERS DESIGNED WITH THE PROPOSED METHOD

Audiogram		**FIR filters designed with the proposed method SSE [dB]**		
f [Hz]	**A [dB]**	**11 samples**	**8 samples**	**6 samples**
125	20	0	0	0
250	30	0	1.5	6
500	32	-0.33	-0.31	0
750	35	0.34	0.34	1.67
1k	37	0	0	2.34
1.5k	38	-0.5	-0.5	0.67
2k	40	0	0	0
3k	36	-0.44	-1	-1
4k	34	0.02	0	0
6k	25	0.37	0	0
8k	12	0	0	0

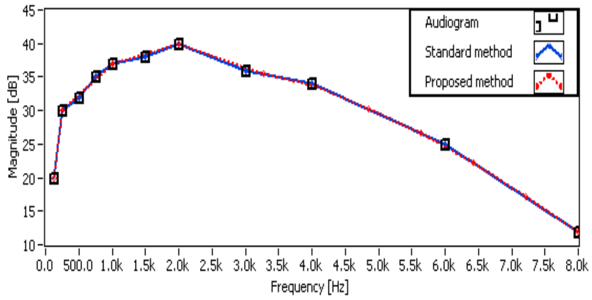

Figure 7. The magnitude responses of the *11*-samples FIR filters designed with the standard (solid line) and the proposed (dotted line) methods.

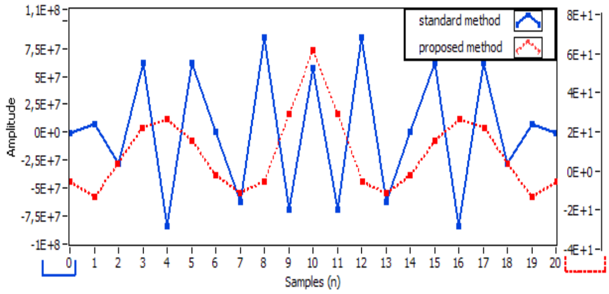

Figure 8. The impulse responses of the *11*-samples FIR filters designed with the standard (solid line) and the proposed (dotted line) methods.

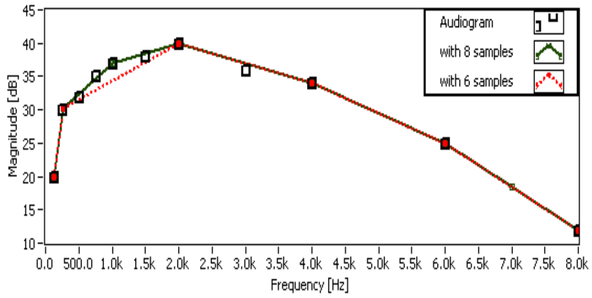

Figure 9. The magnitude responses of the filters designed with the proposed method, based on *8* samples (solid line) and *6* samples (dotted line).

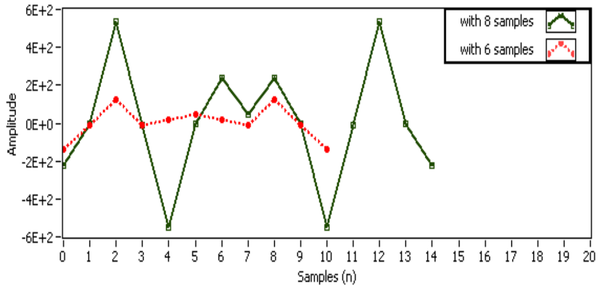

Figure 10. The impulse responses of the filters designed with the proposed method, based on *8* samples (solid line) and *6* samples (dotted line).

TABLE II. THE VALUES OF IMPLEMENTATION-RELATED PARAMETERS FOR THE DESIGNED FILTERS

FIR Type		**I**	**II**	**III**	**IV**
Standard method	***det(V)***	1.58E-7	5.33E-7	1.1E-13	7.3E-12
	|h|min	8.3E+0	7.1E+5	0	1.5E+0
	|h|max	8.5E+7	5.8E+7	8.5E+9	6.1E+1
Proposed *11 samples*	***det(V)***	1.45E+6	9.75E+5	6.23E+3	6.1E+4
	|h|min	7.9E+0	1.1E-1	0	5.2E+0
	|h|max	5.9E+1	4.7E+1	2.9E+2	5.2E+1
Proposed *8 samples*	***det(V)***	2.37E+2	1.86E+2	3.14E-1	2.74E0
	|h|min	4.6E+0	1E+2	1E+2	9.5E+2
	|h|max	5.5E+2	3.9E+2	3.9E+2	4.7E+3
Proposed *6 samples*	***det(V)***	5.15E+1	4.91E+1	6.83E-2	6.09E-1
	|h|min	7.9E+0	8.2E+0	0	2.2E+2
	|h|max	1.4E+2	9.5E+1	2.7E+3	1.4E+3

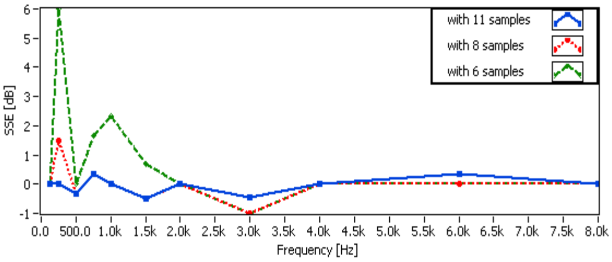

Figure 11. The SSE values with respect to the target audiogram for the filters designed with the proposed method, based on *11* samples (solid line), *8* samples (dotted line) and *6* samples (interrupted line).

Figure 11 presents the SSE corresponding to the *11* points of the audiogram for the three FIR filters designed by using the proposed method, based on *11* samples (solid line), *8* (dotted line) and *6* samples (interrupted line). The SSE values peak at *250Hz*, a point deemed of less importance, while for the key *1.5kHz-6kHz* range all three magnitude responses stay within *1dB* of the target audiogram.

Once the FIR filter that matches the given audiogram was obtained one can derive the appropriate equalizer for the hearing aid tailored for that patient; an effective way for this is to use LabVIEW FPGA as shown in [8], [15].

V. CONCLUSIONS

This paper presents an iterative method for designing FIR digital filters that realize arbitrary magnitude characteristics, defined by the user through a set of frequency-magnitude points. The proposed method is based on the non-uniform frequency sampling algorithm, that is applied here within an optimization-like design loop: the main idea is to generate a new set of frequency samples for each design iteration by manipulating the samples used in the previous run; this implies changing the frequency position of the samples and adjusting their magnitude through interpolation; if necessary additional frequency samples are introduced. After each iteration the magnitude characteristic of the resulting filter is determined by using the non-uniform DFT and the difference between the target magnitude values and the ones corresponding to the same frequencies in the obtained magnitude response is computed. If the errors are larger than the acceptable levels set by the user a new iteration is run. Other triggers for re-runs are the spread of filter coefficient values and the length of the resulting filter.

The proposed method yields filters that are easier to implement than the standard designs: first, unpractical solutions – for which the values of the filter coefficients are too widely spread – are avoided by built-in checks; next, the filter order is decreased iteratively, until the optimum – as defined by the user – trade-off between magnitude errors, the filter order and the maximum value of its coefficients is found. It is also possible to increase the number of samples, in order to decrease both the magnitude errors and the spread of filter coefficients, but this has less practical value.

A tool for designing FIR filters based on this method was implemented in LabVIEW. It was demonstrated on an important real-life application: the synthesis of filters that match a given (arbitrary) human audiogram. Three filters have been designed, with lengths equal or smaller than the filter obtained by using a standard method; for all of them the coefficients vary within a far narrower range than the standard filter – from seven decades to one – while the errors remain relatively small, less than *1dB* above *1kHz*.

ACKNOWLEDGMENTS

The authors deeply thank National Instruments Romania for their generous support.

REFERENCES

[1] A. Antoniou, “*Digital Signal Processing*”, McGraw Hill, pp. 425-458, 2005.

[2] T. W. Parks, C. S. Burrus, “*Digital filter design*”, John Wiley & Sons Inc., pp. 33-44, 1987.

[3] L. R. Rabiner, R. W. Schafer, “Recursive and nonrecursive realizations of digital filters designed by frequency sampling techniques”, *IEEE Trans. on Audio and Electroacoustics*, 1971. [Online]. Available: http://dx.doi.org/10.1109/TAU.1971.1162185.

[4] H. Samueli, “On the design of optimal equiripple FIR digital filters for data transmission application”, *IEEE Trans. on Circuits and Systems*, 1988. Available: http://dx.doi.org/10.1109/31.9919.

[5] G. J. Dolecek “Demo Program for Frequency Sampling FIR Filter Design Method”, *Frontiers in Education Conference*, 2010. [Online]. Available: http://dx.doi.org/10.1109/FIE.2010.5673496.

[6] J. Huopaniemi, M. Karjalainen, “HRTF filter design based on auditory criteria”, *Proc. Nordic Acoustical Meeting*, 1996.

[7] Y. Lian, Y. Wei, “A computationally efficient nonuniform FIR digital filter bank for hearing aids”, *IEEE Trans. on Circuits and Systems*, 2005. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=& arnumber=1556782&isnumber=33114.

[8] E. Szopos, M. Topa, L. Festila, H. Hedesiu, “FIR synthesis of the human hearing mechanism response”, *Acta Technica Napocensis, Electronics and Telecommunications*, pp. 41-44, 2010.

[9] L. R. Rabiner, “Techniques for designing finite-duration impulse-response digital filters”, *IEEE Trans. on Audio and Electroacoustics*, 1971. Available: http://dx.doi.org/10.1109/TCOM.1971.1090625.

[10] I. Gohberg, V. Olshevsky, “The fast generalized Parker-Traub algorithm for inversion of Vandermonde and related matrices”, *Journal of Complexity*, 1997. [Online]. Available: http://dx.doi.org/10.1006/jcom.1997.0442.

[11] A. Eisinberg, G. Fedele, “On the inversion of the Vandermonde matrix”, *Applied Mathematics and Computation, Elsevier*, 2006. [Online]. Available: http://dx.doi.org/10.1016/j.amc.2005.06.014.

[12] M. A. De Jesus, M. Teixeira, L. Vicente, Y. Rodriguez, “Nonuniform discrete short-time Fourier transform a Goertzel filter bank versus a FIR filtering approach”, *IEEE International Midwest Symposium on Circuits and Systems*, 2007. [Online]. Available: http://dx.doi.org/10.1109/MWSCAS.2006.382241.

[13] S. Bagchi, S. K. Mitra, “The nonuniform discrete Fourier transform and its applications in filter design: Part I-1-D”, *IEEE Trans. On Circuits and Systems-II: Analog and Digital Signal Processing*, 1996. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp= &arnumber=502315&isnumber=10979.

[14] O. O. Khalifa, M. H. Makhtar, M. S. Baharom, “Hearing aids system for impaired people”, *International Journal of Computing & Information Sciences*, 2004.

[15] E. Szopos, H. Hedesiu, “LabVIEW FPGA based noise cancelling using the LMS adaptive algorithm”, *Acta Technica Napocensis, Electronics and Telecommunications*, pp. 5-8, 2009.