Anais do XXXIV COBENGE. Passo Fundo: Ed. Universidade de Passo Fundo, Setembro de 2006.
ISBN 85-7515-371-4

A FIR FILTER DESIGN TOOL USING GENETIC ALGORITHMS

André Macário Barros – andre_barros@superig.com.br
Álvaro Luiz Stelle – stelle@cpgei.cefetpr.br
Heitor Silvério Lopes – hslopes@cpgei.cefetpr.br
Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial, CPGEI.
Universidade Tecnológica Federal do Paraná, UTFPR.
80230-901 – Curitiba – PR.

Abstract: *A simple FIR filter design tool was developed. This tool is based on a method using a genetic algorithm that was also developed. The main features of this tool that makes it different from others are: 1) it needs a minimum number of genetic algorithm parameters adjustment, that is, the user only needs to enter the FIR filter specifications and just need to specify one or two extra parameters, hiding all the evolutionary computation complexity; and 2) the quality of the frequency response generated by the tool revealed an interesting alternative for short transition bands when compared with the Parks-McClellan method. The tool works for symetric, anti-symetric, odd and even orders FIR filters and its main part was developed using the Galopps genetic algorithm tool. This free tool will be available for the community soon. This paper emphasises the description of the software developed.*

Keywords: *Genetic algorithm, FIR filter design, GALOPPS, MATLAB, Parks-McClellan.*

1. INTRODUCTION

The design of FIR (*Finite Impulse Response*) digital filters using techniques of DSP (*Digital Signal Processing*) is already an automatic procedure. There are computer programs like MATLAB and DSPLAY that offer this facility. Some of these techniques use methods like window, remez and frequency sampling (PROAKIS and MANOLAKIS, 1996). The only thing the user needs to concern is the the FIR filter specification which can involve some few additional parameters related to the method chosen.

FIR filter design using genetic algorithms has been studied at least for 16 years (Suckley, 1991). But choosing this way to get a digital filter, the user usually has to know a considerable number of additional parameters besides the filter specifications. They are the evolutionary parameters. Some of them need to be adjusted to each new filter specification. Another feature is the stochastic behavior of this kind of application: the user needs to know that it could be necessary to run more than one execution to get an interesting solution. And this solution could be different at each time the application is run. These two features of a FIR filter design tool based on genetic algorithms tend to restrict it for people that have some knowledge of evolutionary computation. This work is an attempt to break this wall: it offers sub-optimal FIR filter responses with a compromise of an average number of four executions. The sub-optimal pattern was created and the reference for this was the Parks-McClellan method (RABINER et. al., 1975) implemented in MATLAB through the remez command.

Hinding these evolutionary parameters, the tool can be used by undergraduate students that don´t have knowledge of Evolutionary Computation for some FIR filter configurations where the specialist methods can´t give satisfactory results.

2. FIR FILTER DESIGN

A FIR digital filter frequency response can be calculated from (PROAKIS and MANOLAKIS, 1996):

$$H(k) = \sum_{n=0}^{M-1} h(n) \times e^{\left(\frac{-j2\pi}{N}\right) \times kn} \tag{1}$$

Where:

- *H(k)* is the Discrete Fourier Transform complex vector. This is the FIR digital filter frequency response;
- *N* is the number of collected points during the sampling process;
- *k* is an index varying from zero to *N-1*;
- *h(n)* is the FIR filter response vector to the unit impulse. This vector corresponds to the FIR filter coefficients; and
- *M* is the number of the FIR filter coefficients.

A digital filter gives a realizable version of a desired frequency response that was specified as part of the filter specifications. This happens because an ideal digital filter response is unrealizable.

To express *H(k)* as a function of the normalized frequency, it can be used:

$$f = \frac{k}{N-1} \tag{2}$$

Where *f* is the normalized frequency ranging from 0,0 to 1,0 cicles/sample.
To get the real frequency it can be used:

$$F = \frac{f}{F_s} \tag{3}$$

Where:

- F is the frequency in Hertz; and
- F_s is the sample frequency in Hertz.

The FIR filter coefficients present a specific symmetry depending on the number of coefficients. Yet, this symmetry can be positive or negative related to the central sample.

Filters classified as Type I or II present the positive symmetry:

$$h(n) = h(M-1-n) \quad n = 0,1,2,...,M-1 \tag{4}$$

Filters classified as Type III or IV present the negative symmetry:

$$h(n) = -h(M-1-n) \quad n = 0,1,2,...,M-1 \tag{5}$$

Type I and III FIR filters have an odd number of coefficients and Type II and IV filters have an even number of coefficients. Types II to IV filters have restrictions related to frequency response specifications and a FIR filter Type III must have a central sample equal to zero.

The complex vector *H(k)* is more useful when viewed as separated in magnitude and phase frequency responses. The symmetry of the FIR filters presented in equations (4) and (5) guarantees a linear phase frequency response. Because of this, what is important in a FIR digital filter frequency response is its magnitude response.

Figure 1 shows some parameters used in a FIR filter desired frequency response specification.

These parameters are described in Table 1.

Bands one, two and three can be passband or stopband.

One parameter was not presented: the cutoff frequency, because it is more frequent to use in FIR filter design the parameters initial and final passband and stopband frequencies.

There is another specification parameter that is outside Figure 1: the FIR number of coefficients, *M*.

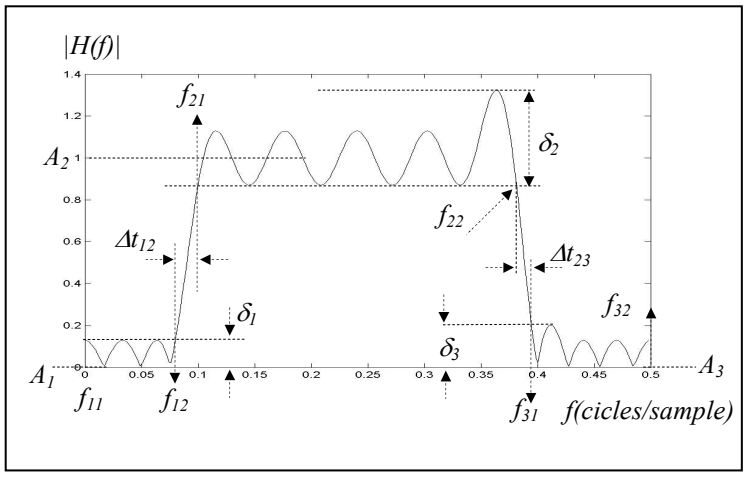

Figure 1: FIR filter specification parameters.

Table 1. FIR filter frequency response usual parameters.

Parameter	Description
A_1, A_2, A_3	Filter amplitude of bands one, two and three
f_{11}, f_{12}	Initial and final frequency of band one
f_{21}, f_{22}	Initial and final frequency of band two
f_{31}, f_{32}	Initial and final frequency of band three
δ_1	Maximum ripple allowed for band one
δ_2	Maximum ripple allowed for band two
δ_3	Maximum ripple allowed for band three
Δt_{12}	Transition band width between bands one and two
Δt_{23}	Transition band width between bands two and three

A FIR digital filter is a LTI (*Linear Time and Invariant*) system, that is, the filtering can be processed through the convolution formula:

$$y(n) = x(n) * h(n) = \sum_{k=0}^{U} x(k)h(n-k) \tag{6}$$

Where:

- *y(n)* is the filtered signal;
- *h(n)* is the FIR filter unit response;
- *x(n)* is the signal to be filtered;
- * is the convolution operator symbol.
- *U* is the sum of the number of samples founded in *h(n)* and in *x(n)*; and
- *n* varies from zero to *U-1*.

It is possible to get from from equation (6) the following conclusion: the larger number of *h(n)* coefficients (*M*), the more precise will be the filtering process. Larger values of *M* offer a better quality of filtering. Better here can be understood as a FIR filter frequency response with a minimum transition width and a minimum value of ripple.

But also through equation (6), it is possible to conclude that for a same signal *x(n)* to be filtered, the number of products between *x(n)* and *h(n)* is regulated by the number of FIR filter coefficients, *M*. And, because of this, larger values of *M* cause larger delays during the filtering process. This can be unacceptable in some real-time applications.

The fetch for an acceptable FIR filter response in terms of quality with a minimum value of FIR filter coefficients is the area of study called FIR filter design.

There are several FIR filter design techniques and the most known are: windows, frequency sampling and Chebyshev approximation techniques. The technique that is considered one of the most accurate is the Parks-McClellan (RABINER et. al., 1975) method, that is one kind of Chebyshev approximation technique. This method uses internally the Second Remez Exchange Algorithm (REMEZ, 1957) and offer over windows and frequency sampling some advantages like:

a) take control over the transition bands; and
b) offer a way to preview the recommended *M* to satisfy the desired specifications.

To get the recommended *M* that will satisfy a determined specification in the Parks-McClellan method, it can be used the HERRMANN et. al. (1973) formula:

$$
\begin{aligned}
M &\cong \frac{D_\infty(\delta_1,\delta_2) - f(\delta_1,\delta_2)\times(\Delta f_t)^2}{\Delta f_t} + 1 \\
D_\infty(\delta_1,\delta_2) &= [0{,}005309\times(\log\delta_1)^2 + 0{,}07114\times(\log\delta_1) - 0{,}4761]\times(\log\delta_2) - \\
&\quad -[0{,}00266\times(\log\delta_1)^2 + 0{,}5491\times(\log\delta_1) + 0{,}4278] \\
f(\delta_1,\delta_2) &= 11{,}012 + 0{,}51244\times(\log\delta_1 - \log\delta_2)
\end{aligned}
\tag{7}
$$

Where Δf_t is $f_{21} - f_{12}$. These values are changed if the smallest transition band is the one between bands two and three. So, for the Parks-McClellan method, it is important the ripple values and the transition bands to preview the recommended number of FIR filter coefficients.

Using the Parks-McClellan method with a value of *M* below the recommended by equation (7), it is not guaranteed that the obtained filter will offer the desired specifications.

This work used this region, the one below the recommended M, to demonstrate that it is possible to get useful filters using GAs (*Genetic Algorithms*) as an alternative tool.

3. GENETIC ALGORITHMS AND THE MODELING CREATED

A GA is based on a sequence of actions that can be represented by Figure 2 (GOLDBERG, 1989).

A brief description of these actions with the model created by this work is:

a) a possible numerical solution of the problem is codified as an individual. Such representation usually adopts symbols to codify the numerical solution. The vector of the individual corresponds to the called chromosome. In the present work, an individual is the *h(n)* vector represented by the binary alphabet in Gray codification;

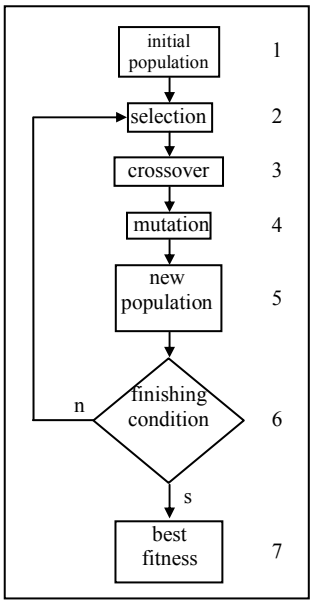

Figure 2: A GA basic flow.

b) a set of individuals is generated at random. This set is called population and corresponds to the step 1 in Figure 2. It is called the search space, *S*, that corresponds to the all possible solutions that can be formed with the chosen alphabet. In the present work, the search space is variable with the number of FIR filter coefficients according to:

$$S = 2^{l} \tag{8}$$

Where:

- *S* is the search space; and
- *l* is the size in bits of the chromosome: 77 for *M*=13 or 14, 88 for *M*=15 or 16, 99 for *M*=17 or 18 and so on.

c) this population is submitted to an evaluation. Each individual is tested according to how good it is as an optimal or suboptimal numerical solution for the proposed problem. In this way, each individual receives a score.

The name of the mathematical function that does this job is the fitness function. In the present work, the fitness used is:

$$fitness = \sum_{i=0}^{N-1} \frac{1}{\left(\left|H_d(f_i)\right| - \left|H_i(f_i)\right|\right)^2} \tag{9}$$

Where:

- *fitness* is the fitness function;
- $|H_d(f)|$ is the desired (given by user specification) magnitude frequency response; and
- $|H_i(f)|$ is the magnitude frequency response from each individual;

$|H_i(f)|$ is calculated through equation (1), because an individual corresponds to one vector *h(n)*. This fitness function is one kind of a MSE (*Mean Square Error*) function that

numerically indicates how close an answer is from the desired answer. This happens in step one of Figure 2 in the first time;

d) it is applied in this evaluated population some mechanism of selection. This is an attempt to exclude from the next steps the individuals with low values of fitness. In the present work, it was used the Stochastic Universal Sampling (GOODMAN, 1996) selection process. This happens in step two of Figure 2;

e) the next step, with the selected population, is a recombination mechanism called crossover. A crossover consists basically of changes in portions of the chromosome between two individuals. The crossover is applied in this selected population with a probability *pcross* between 0,0 to 1,0. In the present work, it was used the two point crossover operation. This happens in step three of Figure 2;

f) after this, another operation over the chromosomes is applied. It is the mutation, that consists basically of a change in one or more bits in an individual at random. This operation is applied with a probability *pmutation* also ranging from 0,0 to 1,0. In the present work, it was used the single bit mutation operation. This is step four in Figure 2;

g) this new population is evaluated as the same way as made in the step described in item *c*. This happens in step five of Figure 2;

h) it is called *generation* to the steps covered by the items *d* to *g* inclusive. Every time a generation is concluded, a finishing condition is tested to finish the GA run. In the present work the finish condition is a max generation number. This is step six in Figure 2;

i) if the finish condition is not true, there is a return to the step that corresponds to item *d* and a new generation cycle is executed. This happens in the decision structure 6 in Figure 2;

j) if the finish condition is true, the last population is obtained and the most fit individual in this population is the best solution the GA can give.

4. FIR FILTER DESIGN WITH GENETIC ALGORITHMS

Beginning with SUCLEY (1991), more than one hundred FIR Filter Design methods using GAs are already published, e.g., CEMES and AIT-BOUDAOUD (1993), LEE et. al. (1998), and KARABOĞA and ÇETINKAYA (2003). In these works, it was not found a software based on GAs with the explicit prior goal of give FIR filter coefficients without the need to adjust evolutionary parameters at each new specification and with a predetermined number of mean trials to get an acceptable answer.

Two related works were found.

The first is a MATLAB toolbox (AHMED, 2004). Some differences between that tool and the proposed work are:

a) The GA parameters specifications. In that tool the GA parameters are configurable, that is, the user must know GAs. And here, the proposed work, this kind of knowledge is minimal;

b) The platform: to run that application, it is necessary to have MATLAB. And here, the proposed work, the final version of the tool runs over Windows directly.

The second work, also a MATLAB toolbox, is a tool called CSDFIR (HASAN et.al., 2001). The final version of this tool is automatic, it is not clear how many times it is necessary to run it to get a satisfactory result and nowadays this tool is owned by a private company.

5. METHODOLOGY

It was developed a specific methodology to get a tool operating in an automatic basis hiding the GA parameters and giving suboptimal results in at least four attempts.

The developed methodology basically consisted of a three-phase set of tests, changing the GA parameters looking for an stable GA parameter configuration that could give a suboptimal result for a bank of filters that was also created.

The suboptimal pattern was also created and the source of comparison was the results given by the Parks-McClellan method.

The parameters presented in the modeling created were obtained by following these concepts of the methodology and the results of its tests.

The limitations of the proposed method and GA modeling was:

-do not cover more than three amplitude levels between 0,0 and 0,5 cycles/sample;

-do not cover any frequency range smaller than 0,01 cycles/sample;

-do not cover all the possibilities of arbitrary level response frequencies; and

-depending of the number of coefficients (which determine the search space), the execution cannot be processed in usual machines;

Details about the methodology and its results can be obtained in BARROS (2006).

6. THE SOFTWARE AND THE RESULTS

At this moment, the software is divided in three tools:

a) The FIR filter specificator program;

b) The GA FIR filter generator tool; and

c) The answer checker program, called *RespFreq*.

6.1 The FIR filter specificator program

This program has a very simple algorithm:

-Begin;

-Read FIR filter specifications (Table 1);

-Check Type filter restrictions;

-Return to get FIR filter specifications if restrictions are nok;

-Create N-point FIR filter ideal magnitude frequency response;

-Create N-point Parks-McClellan FIR filter magnitude frequency response;

-Save in files the frequency responses and the remaining FIR filter specifications;

-Calculate, if asked by user, the equation (7) recommended M;

-End.

The implementation of the method was written in the C language on Borland C++ version 5. To create the Parks-McClellan FIR filter magnitude frequency response it was used a free code (JANOVETZ, 1998), that is the converted version of the Parks-McClellan code (MCCLELLAN et. al., 1973) from FORTRAN to the C language.

One FIR filter Specificator screen can be viewed in Figure 3:

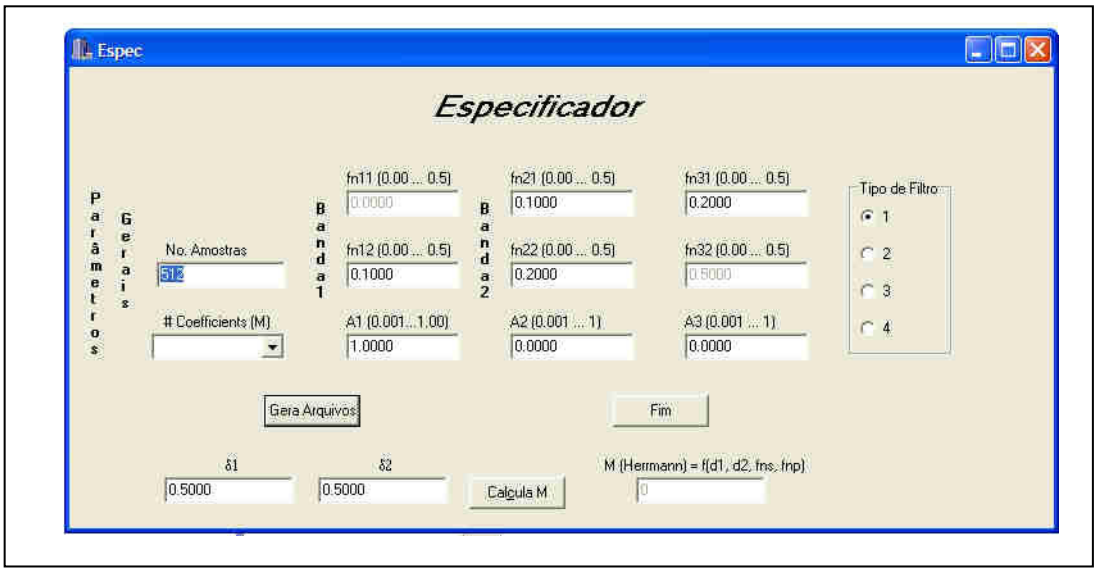

Figure 3: The FIR filter specificator program.

6.2 The GA FIR filter generator tool

To develop this program it was used the GALOPPS GA tool (GOODMAN and PUNCH, 2002). This is a powerful tool that contais the flow of actions of Figure 2 plus a lot of other GA techniques like elitism. What the developer has to do is to apply his our her developed model in one set of specific programs. The language of the tool and the specific programs is C, and it is totally opened and documented.

The executable form is in Dos/Console. The user needs only to do two actions:

a) edit a text-file called *app.in* and change two parameters depending on the size of the FIR filter *h(n)*, i.e., its number of coefficients. The parameters that need a light adjust are *popsize* (the size of the population) and *maxgen* (maximum number of generations);

b) after those adjustments, the user needs only to type *onepop –i app.in* and expect the end of the execution or monitor the answer being generated by other tool.

The program takes the files created by the FIR filter program specificator and runs the GA according to Figure 2 and the modeling presented previously. The results of the execution are two files: one which contains the *h(n)* vector, that is the best GA FIR filter coefficients, and other that contains its magnitude frequency response. These two files are generated only at the end of the execution. During the execution, another file is generated. In this file the best of the generaton intermediate answers are being stored.

One of the steps that is relevant is the calculation of the equation (9), the fitness function. The time necessary to run the pure Discrete Fourier Transform presented in equation (1) is impractical. To process this function to each individual in a population for each generation, is was chosen a faster function, the FFT (*Fast Fourier Transform*) (PROAKIS and MANOLAKIS, 1996). The FFT piece of C-code used was from NIELSEN (2000).

Another part interesting to note was the binary positional to binary Gray codification. It is just a matter of a *XOR* manipulation that can be obtained through the shift operator.

6.3 The answer checker *RespFreq*

During of after the GA execution it is possible to check at each generation the best frequency response generated by the GA. It is done through the *RespFreq* program. This is also a C language program developed in Borland C++ version 5, and its basic algorithm is:

-Begin;
    -if *Best Answer* bottom is pressed, run function best_answer;
    -if Best Answer in dB bottom is pressed, run function best_answer_dB;
    -if Capture Answer bottom is pressed, run function capture_answer;
-End.

Function best_answer:
-Begin;
    -Read from file best GA FIR filter frequency response;
    -Read from file Parks-McClellan FIR filter frequency response;
    -present on screen both frequency responses;
-Return;

Function best_answer_dB:
-Begin;
    -Read from file best GA FIR filter frequency response, $|H_{AG}(f)|$;
    -Calculate $|H_{AG}(f)|_{dB} = 20 \times log[|H_{AG}(f)|]$;
    -Read from file Parks-McClellan FIR filter frequency response $|H_R(f)|$;
    -Calculate $|H_R(f)|_{dB} = 20 \times log[|H_R(f)|]$;
    -present on screen both calculated frequency responses;
-Return;

Function capture_answer:
-Begin;
    -Read from file intermediate best generation GA FIR filter frequency response;
    -Read from file Parks-McClellan FIR filter frequency response;
    -repeat
        -present on screen both frequency responses;
        -read next record of intermediate best GA FIR filter frequency response;
    -until record equal to last
-Return;

Being responsible for present the results, this program is important because it reveals an ocasional need to rerun the GA tool to get another answer.

Figure 4 presents one of the results where the recommended M from equation (7) is not respected. The FIR filter specifications were: $f_{11}=0,0;f_{12}=0,35/A_1=1;\ f_{21}=0,37;f_{22}=0,45/A_2=0;\ f_{31}=0,45;f_{32}=0,5/A_3=0$, *M=15, Filter Type=I, popsize=160, maxgen=1000*.

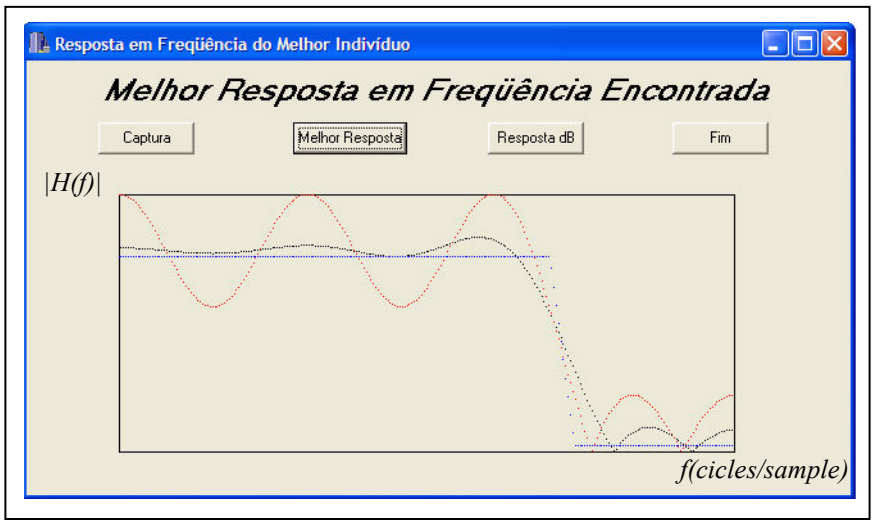

Figure 4: Low Pass Filter, ideal in blue, GA in black, and Parks-McClellan in red.

Another suboptimal result can be viewed in Figure 5. The FIR filter specifications were: $f_{11}=0,0;f_{12}=0,08/A_1=0;\quad f_{21}=0,12;f_{22}=0,32/A_2=1;\quad f_{31}=0,40;f_{32}=0,5/A_3=0,\quad M=17$, *Filter Type=III, popsize=160, maxgen=1000.*

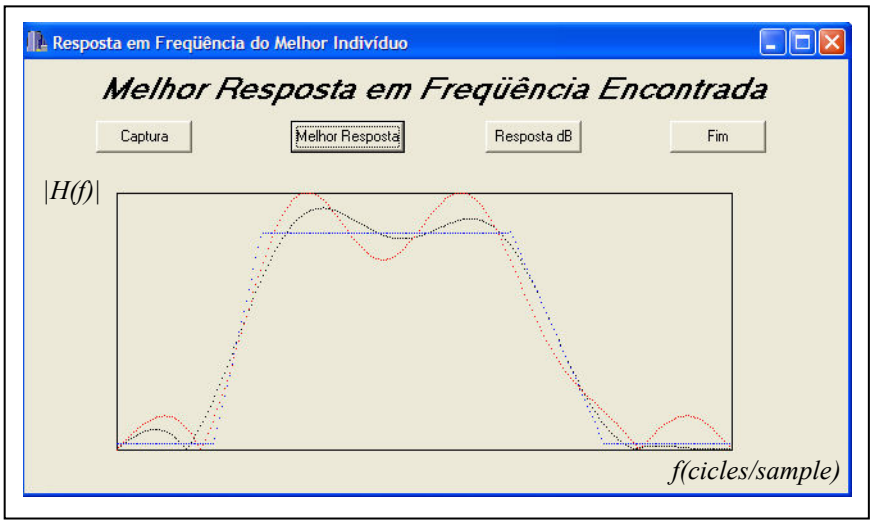

Figure 5: Band Pass Filter, ideal in blue, GA in black, and Parks-McClellan in red.

7. CONCLUSIONS

A FIR filter design software tool based on Genetic Algorithms requiring a minimum number of GA parameters adjustments was developed. Only population size (*popsize*) and maximum number of generations (*maxgen*) are the Genetic Algorithms parameters the user needs to interact with the software.

The quality of the obtained FIR frequency responses are better then the Parks-McClellan method can give when the number of coefficients (M) is far bellow the recommended by equation (7). As close as M is from the recommended, the quality of the GA tends to offer only suboptimal results. For both cases it is still useful for some applications that tolerate the suboptimal pattern proposed.

At the moment the tool is in a segmented form that is not the best to be used with students in a Digital Signal Processing Laboratory. But it is already usable and it will be available as a free tool for the community soon. An unified version, totally graphic is being developed as well as differents forms of solve the FIR filter design problem also using Genetic Algorithms.

According to what was presented in the modeling section, dedicating time to study and learn Genetic Algorithms is an important educational investment not only for FIR filter design. By just changing the modeling, this structure can be used for very different scenarios like IIR filter design, employee scheduling, and best route between two points.

Acknowledgements

The first author thanks to Professor Álvaro Luiz Stelle and to Professor Heitor Silvério Lopes for their strong support for the development of the work.

REFERENCES

AHMED, S.M. Design of FIR filters with arbitrary amplitude and phase specifications using genetic algorithm. In: 46th IEEE International Midwest Symposium on Circuits and Systems MWSCAS 2003, Cayro. **Proceedings**. Cayro: IEEE, 2003. v.2, p. 648-651.

BARROS, A.M. **Projeto de filtros FIR através de algoritmos genéticos.** 2006. Thesis (Master of Science) – Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial (CPGEI), Universidade Tecnológica Federal do Paraná (UTFPR), Curitiba.

CEMES, R., AIT-BOUDAOUD, D. Multiplier-less FIR filter design with power-of-two coefficients. In: Digital and Analogue Filters and Filtering Systems (Digest No. 1993/199), London. **Colloquium on**. Stevenage: IEE, p. 6/1-6/4.

GOLDBERG, D. **Genetic Algorithms in Search, Optimization, and Machine Learning**. Reading: Addison-Wesley, 1989.

GOODMAN, E. **An Introduction to GALOPPS, the "Genetic Algoritm Optimized for portability and parallelism".** East Lansing: Department of Computer Science, and Case Center for Computer-Aided Engineering and Manufacturing, Michingan State University, 1996. 84 pages. Technical Report n. 96-07-01. Available at: http://garage.cps.msu.edu/software/galopps. Accessed in 05/24/2006.

GOODMAN, E., PUNCH, B. **Galopps – the "Genetic Algoritm Optimized for portability and parallelism".** East Lansing, 2002. 601 kbytes. Genetic Algorithms Research and Applications Group (GARAGe). Auxiliary software tool to develop genetic algorithms. Available at: http://garage.cps.msu.edu/software/galopps. Accessed in 05/24/2006.

HASAN, Y.M., KARAM, L.J., FALKINBURG, M., HELWIG, A. Canonic signed digit Chebyshev FIR filter design. **IEEE Signal Processing Letters**. Hamilton, v. 8, n. 6, p. 167-169, 2001.

HERRMANN, O., RABINER, L.R., CHAN, D.S.K., Practical design rules for optimum finite impulse response lowpass digital filters. **Bell Systems Technical Journal**. Hoboken: v. 52, p. 769-799, 1973.

JANOVETZ, J. **McClellan-Parks Program.** Illinois, 1998. 23,9 kbytes. Source code in C language used to generate FIR filters through the Parks-McClellan method. Available at: http://www.janovetz.com/jake. Acessed in 05/24/2006.

KARABOĞA, N., ÇETINKAYA, B. Performance comparison of genetic algorithm based methods of digital filters with optimal magnitude response and minimum phase. In: 46th IEEE International Midwest Symposium on Circuits and Systems MWSCAS 2003, Cayro. **Proceedings**. New York: IEEE, 2003. v.2, p. 644-647.

LEE, A., AHMADI, M., JULIEN, G.A., LASHKARI, R.S., MILLER, W.C. Design of 1-D FIR filters with genetic algorithms. In: 5th International Symposium on Signal Processing and its Applications ISSPA´99, Brisbane. **Proceedings**. New York: IEEE, 1998. v.2, p. 955-958.

MCCLELLAN, J.H., PARKS, T.W., RABINER, L.R. A computer program for designing optimum FIR linear phase digital filters. **IEEE Transactions on Audio and Electroacustics**. New York, v. AU-21, n. 6, p. 506-526, 1973.

NIELSEN, J.J. **MIXFFT.** Hoersholm, 2000. 40,7 kbytes. Source code in C language used to calculate the Fast Fourier Transform. Available at: http://hjem.get2net.dk/jjn/fft.htm. Acessed in 05/24/2006.

PROAKIS, J., MANOLAKIS, D. **Digital Signal Processing: principles, algorithms and applications**. 3 ed. Upper Saddle River: Prentice-Hall, 1996.

RABINER, L.R., MCCLELLAN, J.H., PARKS, T.W. FIR digital filter design techniques using weighted Chebyshev approximation. **Proceedings of the IEEE**. New York, v. 63, n. 4, p. 595-610, 1975.

REMEZ, E.Y. General computational methods of Tchebycheff approximation. **Atomic Energy Comission**. Kiev: translation n. 4491, p. 1-85, 1957.

SUCKLEY, D. Genetic algorithm in the design of FIR filters. **IEE Proceedings G: Circuits, Devices and Systems**. Stevenage: v. 138, n. 2, p. 234-238, 1991.