DIGITAL FILTER DESIGN OF IIR FILTERS USING REAL VALUED GENETIC ALGORITHM

MIKAEL NILSSON, MATTIAS DAHL AND INGVAR CLAESSON
Blekinge Institute of Technology
Department of Telecommunications and Signal Processing
Box 520, SE-372 25 Ronneby
SWEDEN

Abstract: - This paper presents a new paradigm for infinite impulse response (IIR) filter design using genetic algorithms (GA). By encode or transform the filter design problem into the z-plane the GA optimization procedure will be simplified. Additionally, given the z-plane encoding new mutation techniques are introduced, with the intention to locate promising regions in the search space. With proper design of the fitness function, the proposed algorithm can be used to evolve both full precision or quantized filter structures.

Key-Words: - Digital Filter Design, IIR, FIR, Quantization, Genetic Algorithm

1 Introduction

Classical design methods of infinite impulse response (IIR) filters are normally restricted to specific norms such as minimax or least square [1, 2]. Additionally, the quantization effects of the coefficients are normally not possible to consider during the design process. To design filters with special requirements such as a tradeoff in norms or concerning quantization effects, there is a need of more general optimization techniques.

A common approach to solve hard filter design problem is to use stochastically based search procedures. Some paper proposes variants of simulated annealing (SA) [3, 4], other different genetic algorithms (GAs) [5, 6].

Simulated annealing in particular is known to provide good results, but is very time consuming due to the stochastic search.

A genetic algorithm is a tool for optimization in complex multidimensional spaces. The inspiration for a genetic algorithm originates in Darwin’s ideas of evolution and survival of the fittest. The algorithm simulates the evolutionary process where the goal is to evolve solutions by means of selection, crossover and mutation [7, 8].

This paper proposes a filter design paradigm based on a real valued genetic algorithm. By including specialized filter mutation operators the stochastic search procedure will take part of known classical design properties.

2 The z-plane Encoding

In the general case an IIR filter can be described by:

$$y(n) + \sum_{i=1}^{q} a_i y(n-i) = \sum_{j=0}^{q} b_j x(n-j)$$  (1)

where $x(n)$ is the discrete input signal, $y(n)$ the output signal, $b_j$ are the coefficients for the Finite Impulse Response (FIR) part, $a_i$ are the coefficients for the recursive IIR part, $q$ is the number of FIR coefficients and $p$ is the number of recursive IIR coefficients.

Equation (1) can be transformed into the z-domain:

$$H(z) = \frac{\sum_{j=0}^{q} b_j z^{-j}}{1 + \sum_{i=1}^{p} a_i z^{-i}}$$  (2)

By finding the roots of the numerator and denominator polynomials, the transfer function can be expressed with poles and zeros

$$H(z) = b_0 \prod_{j=1}^{q} \left(1 - \beta_j z^{-1}\right) \prod_{i=1}^{p} \left(1 - \alpha_i z^{-1}\right)$$  (3)

where $\beta_j$ and $\alpha_i$ are the zeros and the poles respectively.

With the constraint that all of the filter coefficients are real, i.e. $b_j \in R$ and $a_i \in R$ the real zeros and poles (type 1) or complex conjugated pairs of zeros and zeros (type 2) can be divided into two different categories:

Type 1: polar representation of single zeros and/or poles with the angle 0 degrees, i.e. there is no imaginary part. The radius of the zeros will be denoted $r_{\beta j}$ and $r_{\alpha i}$ for the poles. The number of zeros in the z-plane will be denoted $l_{\beta j}$ and the number of poles $l_{\alpha i}$. Note that $r_{\beta j}$ and $r_{\alpha i}$ can be positive or negative radius.
Type 2: polar representation of complex conjugated zeros and/or poles i.e. two zeros or two poles are needed. Due to the symmetry property of complex conjugated pairs the optimization process only uses the zero or pole in the angular interval $0 - 180$ degrees ($0 - \pi$ rad). The radius, angles, the number of zeros and poles in $0 - 180$ will be denoted $r_{\beta 2}$ and $r_{\alpha 2}$, $\theta_{\beta 2}$ and $\theta_{\alpha 2}$, $l_{\beta 2}$ and $l_{\alpha 2}$, respectively. Note that $r_{\beta 2}$ and $r_{\alpha 2}$ is positive and half of total number of zeros and poles are taken into consideration.

The transfer function can now be described by (4).

The GA encoding of the filter is a string (chromosome, individual), denoted $x$, containing the type 1 and type 2 variables, see Fig. 2.

![Fig. 2. The z-plane encoding of the filter and the length for the different parameters.](image)

Fig. 2. The z-plane encoding of the filter and the length for the different parameters.

The search space for the encoding in Fig. 2 is in general found as:

$$\begin{align*}
\mathbf{r}_{\beta 2} & \geq 0 \\
0 & \leq \frac{l_{\beta 2}}{2} \\
0 & \leq \frac{l_{\alpha 2}}{2} \\
0 & \leq b_0 \\
\theta_{\beta 2} & \leq 180 \\
\theta_{\alpha 2} & \leq 180
\end{align*}$$

(6)

where $r_{\text{max}} \leq 1$ to ensure stability of an IIR filter.

3 Evolving the Population

Given the encoding and the search space defined by equation (6), the initial population of size $P$ corresponding to the filter candidates is created. The initial population (filters) can for example be randomized initialized.

Given the population, genetic operators are applied to evolve the population and the evolution steps made can be found in Fig. 3.

![Fig. 3. The evolving steps](image)

3.1 Evaluate population

Evaluation is done by assigning a fitness measure to each individual in the population by using a fitness function. The fitness function maps an individual from the population to a single judging value (the cost). The fitness function is based on the design goals for the filter (e.g. the amplitude deviations or constraints, the cost of non-zero bits in quantized filters, etc.). The fitness function should avoid to be extremely rugged which will lead to slow or poor convergence of the GA [9].
The encoding scheme in this paper employing a fitness function applied to (4). The goal for the GA is to find a $H(z)$ which fulfill certain constraints on this function. This is probably the most convenient and straightforward solution to create a proper fitness function.

Extension to fitness functions for quantized filters can for example be created by using the following procedure:

1. Convert the string (chromosome) to $b_j$ and $a_i$
2. Transform $b_j$ and $a_i$ to a desired structure
3. Quantize the coefficients in the desired structure
4. Transform back to $b_j$ and $a_i$
5. Check stability of the filter
6. Check the constraints on $H(z)$ by using (2) and calculate the fitness value
7. Penalize unstable filters

Note that the quantization operations will require a check of stability. By adjusting some settings, e.g. $r_{max}$, the amount of unstable filters due to quantization can be reduced.

The penalty for unstable filters can be achieved by:

$$\text{eval}(x) = \begin{cases} f(x_{\text{quant}}) & \text{if } x_{\text{quant}} \text{ is stable} \\ E_{\text{worst}} + \text{stability measure} & \text{otherwise} \end{cases}$$

(7)

where $x_{\text{quant}}$ is the quantized coefficients in some chosen structure, $f(x_{\text{quant}})$ is the fitness criteria of the filter given the quantized coefficients in the chosen structure, $E_{\text{worst}}$ the worst found individual value found so far (from all previous generations until now) and the stability measure is a measure of how far from stability the filter is (e.g. the maximum radius for the poles found from $a_j$ in step 4).

### 3.2 The Selection

Several selection techniques using GA has been proposed: steady state, tournament, proportional, rank, etc. In this paper the rank selection is used in combination with k-elitism and a kind of immigrants scheme [7, 8, 9].

In k-elitism scheme the $k$ most fitted individuals in the population will be directly selected to the next generation. A number $\lambda$ of randomly created immigrants will be inserted into the next generation as well. The remaining $P - k - \lambda$ individuals, which are used for crossover and mutation, are selected according to the probability:

$$p_{\text{rank}} = \frac{\text{rank}(x_i)}{\sum_{i=1}^{P} \text{rank}(x_i)}$$

(8)

where $\text{rank}($ will rank the individuals from $[P, P - 1, \ldots, 1]$ according to the fitness.

### 3.3 Crossover

Given two selected parents according to (8), crossover are used to create new offsprings. Crossover will be performed if a uniform distributed random number will be less than the crossover probability $p_c$. Four standard techniques for crossover are considered single point crossover, two point crossover, uniform crossover and arithmetic crossover [7, 8, 9]. The choice of the crossover operation applied depends on the probability vector:

$$p_{c, \text{vec}} = \begin{bmatrix} P(\text{single point crossover}) \\ P(\text{two point crossover}) \\ P(\text{uniform crossover}) \\ P(\text{arithmetic crossover}) \end{bmatrix}$$

(9)

### 3.4 Mutation

After a crossover operation the resulting individuals will undergo mutation. The probability of mutation $p_m$ is used to decide if mutation should take place. Five different mutation operations are considered: the Gaussian mutation is commonly used in real coded GAs, the other four are new mutation paradigms introduced by considering classical digital filter and their pole and zeros patterns in the z-plane. The selection of actual mutation operation is performed according to the probability vector:

$$p_{m, \text{vec}} = \begin{bmatrix} P(\text{Gaussian mutation}) \\ P(\text{minimum phase mutation}) \\ P(\text{unit circle mutation}) \\ P(\text{linear phase mutation}) \\ P(\text{magnitude mutation}) \end{bmatrix}$$

(10)

#### 3.4.1 Gaussian mutation

The Gaussian mutation modifies the string by adding a randomly generated vector from the gaussian distribution [8].

$$x' = x + N(0, \sigma)$$

(11)

where $x$ is the individual to to mutated, $x'$ the mutated individual and $\sigma$ the vector which controls the shaped of the gaussian distribution. Here $\sigma$ will contain $\sigma_{\tau_1}, \sigma_{\tau_2}, \sigma_{\theta_1}, \sigma_{\tau_2}, \sigma_{\theta_2}$ and $\sigma_{b_0}$, in this way the movements of the different parts in $x$ can be controlled.

#### 3.4.2 minimum phase mutation

In minimum phase mutation the zeros outside the unit circle are wrapped into the unit circle by inverting the radius, see Fig. 4. This operation combined with a scaling of $b_0$ yields
a filter with the same magnitude with minimum phase property [1].

3.4.3 unit circle mutation

Zeros on the unit circle in the $z$-plane commonly appear then filters are designed by classical approaches [1]. With this in mind the unit circle mutation change the radius to one for all zeros (i.e. keep the angles but move all zeros in the $z$-plane to the unit circle), see Fig. 5.

3.4.4 linear phase mutation

A linear phase FIR filter can be achieved by moving all zeros to the unit circle as described in unit circle mutation or mirror the zero in the unit circle (“mirrored” positions). Assume that there is a zero with radius $r$ when it exists another zero with radius $\frac{1}{r}$. Note that this mutation will only give exact linear phase if the filter is strictly FIR.

The linear phase mutation is applied by replacing the upper half of $r_{\beta 1}$ and $r_{\beta 2}$ with the “mirrored” zero position of the lower half. If $l_{\beta 1}$ or $l_{\beta 2}$ is odd, will the remaining radius be set to one, see Fig. 6.

3.4.5 magnitude mutation

The filter magnitude in digital filter design is commonly close to one in the passband regions or some frequency regions. The magnitude mutation selects a single angle between 0 and 180 degrees randomly with uniform distribution and adjusts $b_0$ so that the magnitude of the filter will be one at this angle.

3.5 Check population

After the crossover and mutation, some individuals can be outside the desired search space (e.g. considering the Gaussian mutation on $r_{|alpha|}$). This will be avoided by the check population step after the genetic operations. If the parameters are outside the desired search space they will be transformed back into the desired search space according to:

$$
\begin{align*}
    r_{\beta 2} &= |r_{\beta 2}| \\
    \theta_{\beta 2} &= \begin{cases} 
        360 - \theta_{\beta 2} \mod 360 & \text{if } \theta_{\beta 2} \mod 360 > 180 \\
        \theta_{\beta 2} \mod 360 & \text{else}
    \end{cases} \\
    r_{\alpha 1} &= \begin{cases} 
        \frac{r_{\alpha 1}}{|r_{\alpha 1}|} & \text{if } |r_{\alpha 1}| > r_{\text{max}} \\
        |r_{\alpha 1}| & \text{else}
    \end{cases} \\
    r_{\alpha 2} &= \begin{cases} 
        |r_{\alpha 2}| & \text{if } |r_{\alpha 2}| > r_{\text{max}} \\
        \frac{r_{\alpha 2}}{|r_{\alpha 2}|} & \text{else}
    \end{cases} \\
    \theta_{\alpha 2} &= \begin{cases} 
        360 - \theta_{\alpha 2} \mod 360 & \text{if } \theta_{\alpha 2} \mod 360 > 180 \\
        \theta_{\alpha 2} \mod 360 & \text{else}
    \end{cases} \\
    b_0 &= |b_0|
\end{align*}
$$

where $a \mod b$ are the modulus operator working on real numbers (e.g. $-10.1 \mod 360 = 349.9$ and $372.1 \mod 360 = 12.1$).

4 Design Example

As a design example using the proposed algorithm, a filter with both magnitude\(^1\) and group delay constraints is considered, see Fig. 7.

The complete fitness function used for optimization is a sum of three special max-norms. Consider the upper limit function for the magnitude in decibels, $M_u(f)$ where $f$ is normalized frequency. A norm for this constraint is defined as:

$$
E_1 = \max_f \left[ (M(f) - M_u(f)) (M(f) > M_u(f)) \right] \quad 0 \leq f \leq 0.5
$$

where $M(f) = 20 \log_{10} |H(f)|$ is the magnitude of the filter in decibels. This norm will give the maximum deviation in decibels from the upper limit in the magnitude. If $M(f)$ is strictly below $M_u(f)$ the norm will yield in $E_1 = 0$.

\(^1\)The filter magnitude is found from [3] and complemented with a group delay constraint
The norm for the lower limit function for the magnitude in decibels is defined as:

$$E_2 = \max_f [(M(f) - M_\alpha(f)) (M(f) < M_\alpha(f))] \quad 0 \leq f \leq 0.5$$

where $M_\alpha(f)$ is the lower limit for the magnitude. This norm will give the maximum deviation in decibels from the lower limit. If $M(f) > M_\alpha(f) \forall f$ the norm will yield in $E_2 = 0$.

The norm for the upper limit function for the group delay in samples is defined as:

$$E_3 = \max_f [(G(f) - G_u(f)) (G(f) > G_u(f))] \quad 0 \leq f \leq 0.5$$

where $G(f)$ is the group delay in samples for the filter and $G_u(f)$ is the upper limit for the group delay. This norm will give the maximum deviation in samples from the upper limit of the group delay. If $G(f) < G_u(f) \forall f$ the norm will yield in $E_3 = 0$.

In the practical implementation of the three norms the normalized frequency is sampled yielding in $f_k \in [0, 0.5]$ for $K = 1, 2, \ldots, K$, for the simulations made here is $K = 256$ used.

Given these three norms the sum of these will yield in the complete norm for the filter design:

$$f(x) = E_1 + E_2 + E_3$$

Note that 1 dB deviation in the magnitude will punish a filter with the same order as 1 sample deviation in the group delay.

Two filters are considered for full precision optimization, one FIR filter with filter length $q = 35$ ($l_{\beta 1} = 0, l_{\beta 2} = 17$), and one IIR with $q = 20$ ($l_{\beta 1} = 1, l_{\beta 2} = 9$) and $p = 8$ ($l_{\alpha 1} = 1, l_{\alpha 2} = 3$). The settings used for both runs can be found in Tab. 1 and Tab. 2.

### Table 1. Settings for GA

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
<th>Value(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N$</td>
<td>Number of generations</td>
<td>5000</td>
</tr>
<tr>
<td>$P$</td>
<td>Population size</td>
<td>100</td>
</tr>
<tr>
<td>$k$</td>
<td>Number of elitists</td>
<td>2</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>Number of immigrants</td>
<td>10</td>
</tr>
<tr>
<td>$r_{\max}$</td>
<td>Max radius for poles</td>
<td>0.99</td>
</tr>
<tr>
<td>$p_c$</td>
<td>Probability of crossover</td>
<td>0.7</td>
</tr>
<tr>
<td>$p_m$</td>
<td>Probability of mutation</td>
<td>0.6</td>
</tr>
<tr>
<td>$p_{c,vec}$</td>
<td>See (9)</td>
<td>$[0.25, 0.25, 0.25, 0.25]$</td>
</tr>
<tr>
<td>$p_{m,vec}$</td>
<td>See (10)</td>
<td>$[0.8, 0.05, 0.05, 0.05, 0.05]$</td>
</tr>
</tbody>
</table>

### Table 2. Settings for Gaussian mutation in GA

The result for the FIR optimization can be found in Fig. 8 and the IIR result is found in Fig. 9.

### Fig. 7. Filter specification. Upper plot: a zoom of the passband limits for the magnitude. Middle plot: the upper and lower limits for the magnitude on the full frequency range. Lower plot: the group delay constraints in the passband.

### Fig. 8. Result for FIR filter of length 35, optimized with proposed method

The full precision filters are found to fulfill the specification. However no quantization is taken into consideration. As a final optimization the design problem of an IIR filter is considered to be implemented in direct form with quantization of the coefficients in two-complement using 8 bits. A direct quantization of the full precision filter, yield a filter which violates the desired specification. By using the
full precision filter as a population initiation and use the fitness calculation according to (7) a result which fulfills the specification is reached, see Fig. 10.

Fig. 10. Direct form quantized, two-complement with 8 bits, IIR filter of length 20 (FIR) and 8 (IIR), optimized with proposed method

5 Conclusions

A new genetic genetic algorithm with encoding in the z-plane and special mutation operations is proposed. The proposed algorithm takes into account properties which has been found interesting in classical design of filters by means of the new mutation operations. Design examples illustrating the ability of designing special FIR and IIR filters are presented. The examples takes into account both magnitude and group delay in the optimization procedure.

References: