TUNING A GENETIC ALGORITHM FOR WAVELET SIGN PREDICTION

Antonio Martí Campoy, Francisco Rodríguez-Ballester
Universitat Politècnica de València
Camí de Vera s/n, València, Spain

Otoniel López, Manuel Pérez Malumbres
Universidad Miguel Hernández
Av. de la Universidad s/n, Elche, Spain

ABSTRACT

Discrete Wavelet Transform has proved to be a powerful tool for image compression because it is able to compact frequency and spatial localization of image energy into a small fraction of coefficients. In recent years coefficient sign coding has been used to improve image compression. The potential correlation between the sign of a coefficient and the signs of its neighbours opens the possibility to use a sign predictor to improve the image compression process. However, this relationship is not uniform and constant for any image. This work presents the tuning of a genetic algorithm in the search of an efficient predictor. This tuning process consists in the evaluation of several genetic algorithm variations created using different combinations of operators, input parameters and workload.
Results from this work allow promoting some design strategies in order to devise a genetic algorithm highly immune to input parameters.

KEYWORDS

Genetic algorithms, parameter tuning, wavelets, image coding, sign coding

1. INTRODUCTION

Genetic algorithms were first introduced by Holland in [Holland75] and they are nowadays well known and very popular to find optimal/suboptimal solutions in very large and complex problems [Goldberg89].

In a genetic algorithm, the evolution usually starts from a population of randomly generated individuals and runs in generations. In each generation the fitness of every individual in the population is evaluated, multiple individuals are stochastically selected from the current population (based on their fitness value), and modified (recombined and possibly randomly mutated) to form a new population. The new population is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached.

In this work we are looking for an optimal/suboptimal solution of our particular problem related with wavelet image compressors. This kind of image compressor is based in the use of a mathematical transform called Discrete Wavelet Transform (DWT). Wavelet transforms have proved to be very powerful tools for image compression, since many state-of-the-art image codecs, including the JPEG2000 standard [JPEG2000], employ DWT into their algorithms. One advantage of the wavelet transform is the provision of both frequency and spatial localization of image energy. The image energy is compacted into a small fraction of the transform coefficients and compression can be achieved by coding these coefficients. The energy of a wavelet transform coefficient is restricted to non-negative real numbers, but the coefficients themselves are not, and they are defined by both a magnitude and a sign. Shapiro stated in [Shapiro96] that a transform coefficient is equally likely to be positive or negative and thus one bit should be used to encode the sign. In recent years, several authors have begun to use context modelling for wavelet sign coding [Wu97, Taubman00, Deever00], showing that despite the equiprobability of wavelet sign values, some sign correlation can be found among wavelet coefficients, resulting in overall compression ratio improvements.

In a previous work we have observed that the sign of a wavelet coefficient may be strongly correlated with the sign of some neighbour coefficients. However, this relationship is not uniform and constant for any image, or even consistent within the same image. By increasing the number and kind of images under analysis, the relationship between the signs of the neighbour coefficients may be generalized or, on the other hand, it is possible that, when increasing the number of images, some relationships become contradictory.

In [Lopez11] a preliminary version of a genetic algorithm is developed to find the sign correlations between a wavelet coefficient and its neighbours in a particular image. The results obtained with this GA version suggest that wavelet sign prediction tools are feasible to achieve good compression performance when implemented in the Lower-Tree Wavelet (LTW) image encoder as in [Oliver06].

In order to improve the overall compression ratio, this work explores several design options of a genetic algorithm. We will propose several versions for initialization, selection and mutation operations, and will also evaluate the impact of several parameters like population size or mutation probability, on the GA performance. GA performance will be defined in terms of solution quality (measured as the number of sign prediction successes).

The paper is organized as follows: in section 2 we define the optimization problem. In section 3 we propose a GA algorithm that matches the problem definition and we perform a detailed tuning to find the best combination of GA operators and parameters that supply the best sign predictions in the shortest time. The paper then presents the main results in section 4 and finally general conclusions are highlighted.

2. WAVELET SIGN PREDICTION: PROBLEM STATEMENT

To estimate wavelet sign correlation in a practical way we have applied a 6-level Dyadic Wavelet Transform decomposition of the source image and then a low quantization level to obtain the resulting wavelet coefficients. As a first approach, and taking into account that the sign neighbourhood correlation depends on the subband type (HL, LH, HH) as Deever assesses in [Deever00], we have used three different neighbours in each subband type. So, for the HL subband the selected neighbours are North (N), North-North (NN) and West (W). Taking into account symmetry, for the LH subband those neighbours are W, WW, and N. And finally the neighbours of the HH subband are N, W, and NW, exploiting the correlation along and across the diagonal edges in the image. So, for a particular subband type, we have defined three neighbours that can hold one of the three possible sign values: positive, negative or null (zero). This lead to a set of 3^3 different Neighbour Sign Patterns (NSP) for each subband type.

After running the genetic algorithm (GA) for each subband type, we obtain the sign prediction table that contains the sign predictions for every NSP[k], $k = 0 \ldots 3^3 - 1$. So, when coding the sign of a wavelet coefficient in a particular subband, first we will get the sign value of the corresponding neighbour set in order to form the actual NSP. Then we will look up this NSP in the table to find the sign prediction of the actual wavelet coefficient. Finally, what we are going to encode is the correctness of this prediction, i.e., a binary valued symbol resulting from expression $PSC_{i,j}[k] \cdot SC_{i,j}$, where $SC_{i,j}$ represents the sign value of actual coefficient C_{ij} (0: positive, 1: negative) and $PSC_{i,j}[k]$ represents the sign prediction of coefficient C_{ij} .

The performance of our binary entropy encoder will depend on the behaviour of our sign predictor, the higher the success prediction ratio the higher the compression rate.

As mentioned before, our goal is to find a prediction table with the best prediction performance for a set of representative images. As we need a sign prediction table for each subband type we will focus our study in one subband type only, taking into account that the prediction tables of the other subband types will be obtained running the same algorithm with their own neighbourhood definition.

3. GA DESIGN AND ALTERNATIVES

For this work 18 variations of the GA have been created. These variations come from the combination of different designs or policies of the GA operators. The adjustments of those parameters that influence most the algorithm behaviour have been also studied. In our case the parameters chosen are the population size and the mutation probability. We have selected these two parameters because they are, in the experience of the authors and the literature [Lobo07, Wessing11] on this subject, those which have a greater effect on the GA results and its execution time.

3.1 Solution representation

Each individual is represented by a binary vector where its elements represent a combination of signs from a predefined neighbourhood set of coefficients, and the stored values into these elements correspond to the sign prediction for the coefficient (binary value). The size of this vector depends on the number of neighbours that conforms the neighbourhood. The greater the number of neighbours considered, the greater the number of sign combinations, namely 3^n being n the number of neighbours, since the possible sign values of neighbour wavelet coefficients are ternary values (positive, negative or null).

As in [Lopez11], we have decided to use a configuration of three neighbours that are identified by their coordinates, North (N), North-North (NN) and West (W) with respect to the actual wavelet coefficient. For an individual i, the element i[j] will store (0: positive; 1: negative) the sign prediction based on the neighbour sign combination j, which it is represented by the index $J_2J_1J_0$ (in base 3) where the sign value of neighbour N is J_2 (0: positive; 1: negative; 2: zero), and the sign values of neighbours NN and W are J_1 and J_0 respectively ($j = J_2 \times 3^2 + J_1 \times 3^1 + J_0 \times 3^0$). For example, if neighbourhood's signs combination is (- , +, null), the corresponding sign prediction will be stored in i[11], where $11=1\times3^2 + 0\times3^1 + 2\times3^0$. This way, when calculating fitness function and processing a pixel x with the same sign neighbourhood (- , +, null), the sign of this pixel x is compared versus the value stored in i[11].

3.2 Initial population

Three alternatives have been defined for the initial population:

Initialization type 1 (**I1**). The sign value for each combination is randomly assigned.

Initialization type 2 (**I2**). The population is created the same as I1 except for two individuals who have all its signs assigned to 0 and 1, respectively.

Initialization type 3 (**I3**). Each individual is initialized with all its signs assigned to 0 except for a continuous sequence of ones. The start point and length of this sequence are chosen randomly. The two special individual from I2 are also added to the population.

3.3 Selection

We have created three variations of the selection operator. All three share a binary tournament selection, where two individuals are randomly selected and the one with better fitness value is chosen as first parent. The second parent is chosen in the same way. Three elitist policies are applied before selection, crossover and mutation operators:

Selection type 1 (**S1**). The two best individuals survive to the next generation and they cannot be modified by the mutation operator. These two individuals are copied before selection and added to the new population after mutation.

Selection type 2 (**S2**). There are no elitist individuals. New generations are completely created through the crossover operator, and all individuals are exposed to mutation.

Selection type 3 (**S3**). The two best individuals are integrated into the next generation but are exposed to mutation. These two individuals are copied before selection and added to the new population before mutation, so they may be modified by mutation.

3.4 Mutation

Two mutation policies have been designed:

Mutation type 1 (**M1**). The value of a randomly chosen gene is inverted.

Mutation type 2 (**M2**). Two randomly chosen genes are swapped. The resulting individual may be the same as the original as both genes may have the same value.

The combination of the above elements produces 18 algorithm variations. The nomenclature for these variations is $I_xS_yM_z$, x = 1 . . . 3, y = 1 . . . 3, z = 1 . . . 2, meaning the initialization type I_x is combined with the selection policy S_x and the mutation policy M_z.

Table 1: Image sets used in experiments

Set	Image list	Set	Image list
Set.1-1	Lena	Set.3-3	09, 17, 19
Set.1-2	Barbara	Set.4-1	01, 05, 07, 17
Set.1-3	01	Set.4-2	03, 11, 16, 19
Set.2-1	Lena, Barbara	Set.5-1	02, 09, 13, 18, 20
Set.2-2	09, 10	Set.6-1	03, 06, 10, 15, 19, 21
Set.2-3	20, 21	Set.8-1	Lena, Barbara, 02, 06, 13, 14, 18, 20
Set.2-4	11, 22	Set.9-1	02, 05, 06, 09, 13, 18, 19, 20, 22, Barbara
Set.3-1	06, 11, 23	Set.10-1	Lena, Barbara, 01, 02, 03, 04, 05, 06, 07, 08
Set.3-2	Lena, 03, 12		

3.5 Input parameters

Using these GA variations we have also analyzed the effects of some input parameters: the population size, the mutation probability and the image or images the algorithm has to process.

To analyze the effect of the population size the algorithms has been run with the following number of individuals: 300, 280, 260, 250, 240, 220, 200, 180, 160, 150, 140, 120, 100, 80, 60, 50, 40, 20, and 10 and for the mutation probabilities (in percentage): 0.010, 0.025, 0.050, 0.075, 0.100, 0.250, 0.500, 0.750, and 1.000.

Finally, to evaluate the effect produced by the image being processed on the genetic algorithm operation the sets of images from Table 1 have been evaluated. These image sets come from [CIPR02] and are extensively used in the development of image processing algorithms.

The total number of different experiments to be carried out per GA variation is 2907 (19 population sizes, 9 mutation probabilities, 17 image sets). Each experiment requires 5000 generations to finish and it has been repeated 15 times to take into account the effect of the random seed.

4. ANALYSIS OF RESULTS

It is very important to understand that each image has, in principle at least, a varying degree of predictability. That is, the percentage of success predicting the wavelet coefficient signs will be different from image to image. This means that the same algorithm, using the same parameters and operators, may produce different results depending on the image it works on. Although it is possible, and it should be studied in a future work, to find a prediction scheme to get good results for a large number of different images, the purpose of this work is not to determine whether it is possible to find an acceptable prediction scheme for any image, even if the success ratio of prediction is the same for different image sets used in this work. The work presented here looks for a combination of operators and parameters of a genetic algorithm that, regardless of the images on which the algorithm works, maximize the probability of finding the best possible result. That is, that there is no other combination of operators and parameters that would obtain a better result.

The fitness function of the GA may be used to compare the results between algorithm variations. These results, however, have to be normalized somehow due to the fact the numeric values are different depending on the algorithm's workload (the image set being analyzed).

A simple and useful method to carry out the normalization is to use as a reference the maximum value of the fitness function obtained by any GA variation for a given workload. Then the normalized value is calculated as the number of times (number of executions) the GA finishes with a fitness value lower than the reference, with respect to the same workload.

The maximum fitness value (henceforth simply referred as the maximum) is different for each image set, and may be obtained by more than one algorithm variation. That is, different operators or the same operators working with different parameters (populations of different sizes, for example) are able to produce the same result. The goal of the analysis presented here is to determine the combination of operators, population size and mutation probability that makes the genetic algorithm produce the best possible result (the maximum) for each image set, so we can observe the effect of these parameters and design variations.

4.1 Analysis of GA operators

The effect of the genetic algorithm operators have been analyzed first. Figure 1 shows the frequency histogram for percentage of cases where results are below the maximum. This percentage of cases in which the genetic algorithm has not reached the maximum is represented on the Y axis while the X axis is used to represent the different GA versions (combinations of operators).

The first thing that can be observed, and this will mark the following analysis, is that algorithms that use the M2 mutation policy offer much worse results than those using the M1 policy. Furthermore, the M2 mutation policy exhibits a large variability in its results.

This can be explained by the fact this mutation policy exchanges genes without checking if these are different or not (the mutated individual will be the same as the original in this case), so the net result is that the effect of the mutation gets lost.

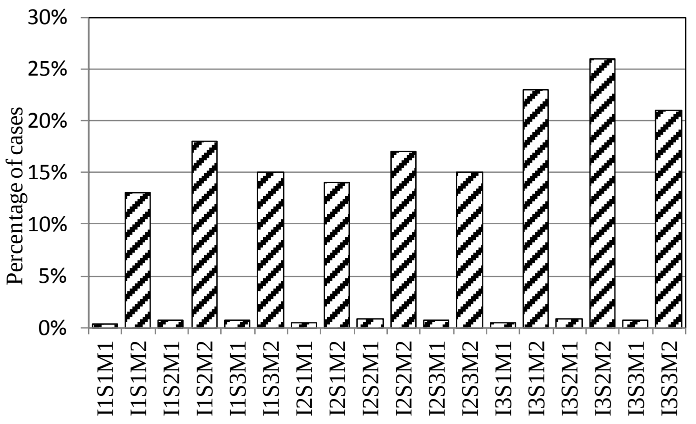

Figure 1. % of cases where result is below maximum for all operators combination

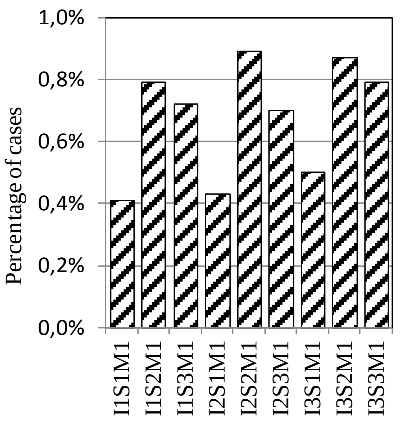

Figure 2. % of cases where result is below maximum for M1 operator only

The same histogram is shown in Figure 2 when only algorithms with mutation policy M1 are considered. In this figure it can be observed that the combination of population sizes and mutation probabilities produce a result lower than the maximum in less than 1% of the cases. This means all these combinations of operators make the algorithm evolve in the correct direction independently of the input parameters (when these are within a range described below).

Figures 3 and Figure 4 show the number of cases where the maximum is not reached considering the effect of the initialization and selection for the mutation policy M1 only.

The conclusion that can be extracted analyzing the results from Figure 3 is that any of the three initialization alternatives produce almost the same results in the behaviour of the genetic algorithm. Analyzing Figure 4 in detail the question arises if the selection policy S1 offers a better behaviour compared to the other selection policies. Although the absolute value of the percentage of cases is low (less than 1%), in terms of relative values the S1 selection policy presents noticeable better results (around half of the cases below the maximum) than the other two.

To determine if there is a statistically significant difference in the genetic algorithm as a function of the selection policy the scatter plot of the fitness function in its maximum value has been created and depicted in Figure 5, that is, the best individual of each genetic algorithm version. As this value exhibits a large deviation for each experiment set (deviations may reach an order of magnitude) the results have been normalized dividing the fitness function value by the maximum (best of all) value for each image set. This way the normalized values are in the range [0,1], the higher the normalized value the better the algorithm behaviour.

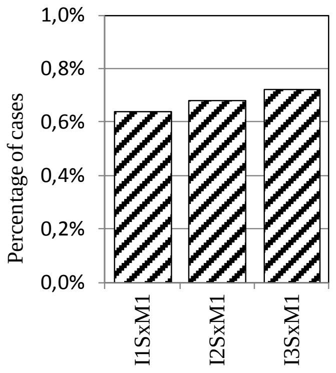

Figure 3. % of cases where result is below maximum as function of initialization policy and M1 mutation.

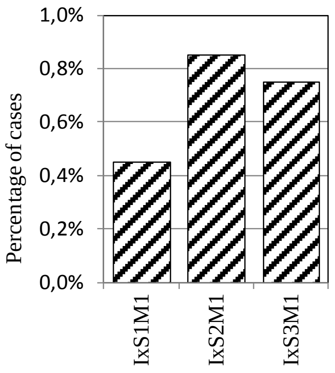

Figure 4. % of cases where result is below maximum as function of selection policy and M1 mutation.

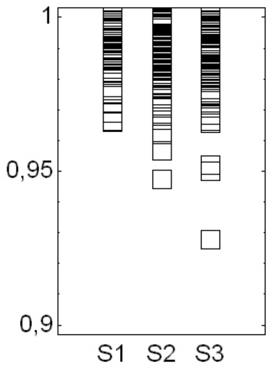

Figure 5. Scatter plot of fitness function value as function of selection policy and M1 mutation

In Figure 5 it can be seen that when the selection policy S1 is used, those cases which do not reach the best result still provide a result closer to the maximum (there is less data dispersion than when another selection policy is in use). Conclusions from Figure 4 and Figure 5 recommend the use of the selection policy S1.

4.2 Analysis of GA input parameters

The effect of some input parameters has been also studied: the population size, the mutation probability and the image set.

Table 2 shows the number of cases the algorithms have not reached the best result as a function of the population size and mutation policy. Due to the clear tendency shown by the data, the table does not show all population sizes analyzed, just those with interesting data.

Table 2: Number of cases below maximum: Population size versus mutation policy

Population size	10	20	40	50	100	150	200	250	300
Number of cases below max. for M1	143	30	0	0	0	0	0	0	0
Number of cases below max. for M2	969	805	631	478	235	68	68	56	50

Results presented in Table 2 show again the mutation policy M1 makes the behaviour of the algorithm highly insensitive to other parameters, in this case the population size. With a population as low as 40 individuals all executions, regardless of other factors (except for the mutation), converge to the maximum value. However, when the M2 mutation policy is in use this independence from the other factors cannot be achieved even with a population size of 300 individuals.

Table 3 presents the number of cases in which the algorithm has not reached the best result as a function of the mutation probability and policy. This table again demonstrates the large effect of the mutation policy on the behaviour of the algorithm.

Table 3: Number of cases below maximum: Mutation probability versus mutation policy

Mutation probability	0.010%	0.025%	0.050%	0.075%	0.100%	0.250%	0.500%	0.750%	1.000%
Number of cases below max. for M1	148	18	2	0	0	0	0	3	5
Number of cases below max. for M2	1540	1132	800	573	426	202	54	6	6

Using probabilities between 0.075% and 0.5% for the mutation policy M1 ensure that the other parameters do not affect the convergence of the algorithm. A mutation probability too low causes the algorithm to improperly explore the solution space, and a mutation too high produce an erratic behaviour of the algorithm, the higher the mutation probability the worse the result.

With regard to mutation policy M2 it can be observed there is the need for a much higher mutation probability to reduce the number of cases producing results below the maximum indicating that the effect of this mutation policy on the population is too weak.

In order to determine if there is any GA variation able to produce an effective wavelet coefficient sign predictor (in terms of absolute number of sign prediction successes) the prediction hit ratio has been defined as the ratio of the number of successes in predicting a coefficient sign and the number of coefficients whose value is nonzero (that is, those coefficients whose sign should be predicted).

Instead of showing all the data obtained, we are going to use one of the GA variations with better behaviour as an example of the predictions that can be obtained. To select the best GA variation we have used the conclusions above, so only those GA variations with mutation policy M1 and selection S1 have been considered, and regarding the initialization type, anyone would be similarly adequate.

We have chosen I1S1M1 as the best performing algorithm and its ability to predict the wavelet coefficient signs processing the different sets of images is depicted in Figure 6.

Taking into account that images are not all equally predictable, differences in the hit ratio are expected. Figure 6 shows that the maximum value is 68% for one of the image sets (Set.1-2), and the lower value is 55% for various image sets. However a strong relationship between the hit ratio and the image set used cannot be established.

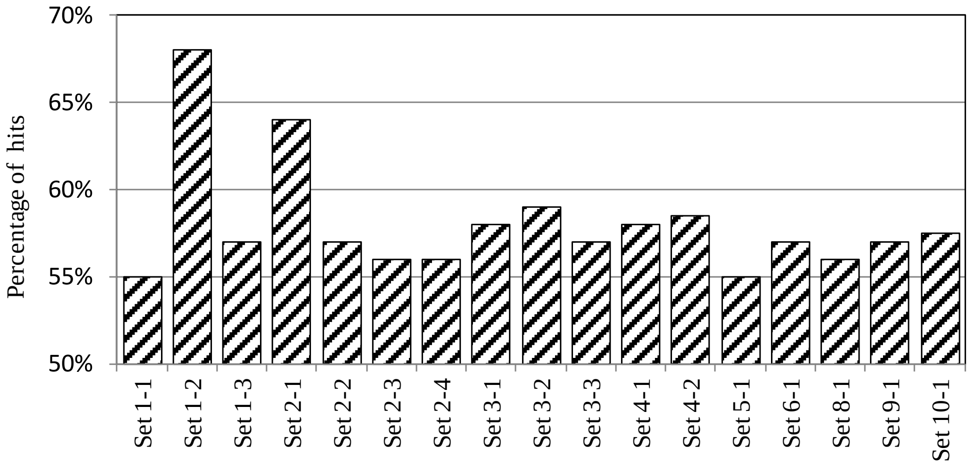

Figure 6. I1S1M1 hit ratio versus image set

5. CONCLUSIONS

In this work 18 variations of a genetic algorithm developed to produce a wavelet sign predictor have been designed and evaluated using different input parameters and workloads.

These design variations come from the combination of three different initialization alternatives, two mutation policies and three selection policies. Multiple sets of images have been processed by each of these variations, and the results have been tuned by the use of different values of input parameters like the population size and the mutation probability.

From the results obtained it can be concluded the mutation policy M1 —which consists in the inversion of a randomly chosen gene— clearly beats the second one —M2, the swap of two randomly chosen genes—. Regarding the analysis of each mutation policy, the M1 variations are mostly insensitive to other operators and input parameters, while the results from M2 variations exhibit a large variability exposing the fact that this mutation policy is too weak and that the influence of other operators and input parameters on the behaviour of the algorithm is very important.

Focusing on the analysis of the M1 mutation policy, the differences of the results from the diverse initialization alternatives are negligible, and while there is a selection policy with better results than the others —called S1, where the two best individuals survive to the next generation— its effect on the results is very small.

This mutation policy is also almost immune to the population size and the mutation probability. With a population size as low as 40 individuals and mutation probabilities ranging from 0.075% and 0.5%, any genetic algorithm variation using mutation M1 is able to correctly explore the solution space and find an almost equally good sign predictor for the wavelet coefficients.

Regarding the effectiveness of this sign prediction, results show the best GA design variation —I1S1M1— is able to predict between 55% and 68% of the coefficient signs (what we have nominated the prediction success ratio).

Finally, the prediction success ratio of the genetic algorithm presented here have been validated comparing its results with those obtained from the algorithm in [Lopez11], both processing the same 17 sets of images used in this work. The improvement media obtained by this new genetic algorithm is 9% with a maximum value of 15% in one of the sets of images tested. In only 2 cases this new algorithm cannot beat the previous one, but in these cases the difference in the prediction success ratio is below 1%.

ACKNOWLEDGEMENT

This work has been supported partially by project PAID-06-11/2055 of Universitat Politècnica de València and projects TIN2011-28435-C03-01, TIN2011-27543-C03-03 and TEC2010-11776-E of Ministerio de Ciencia e Innovación.

REFERENCES

[Deever00] Aaron Deever and Sheila S. Hemami. What's your sign? Efficient sign coding for embedded wavelet image coding. In *Proceedings of the IEEE Data Compression Conference*, pages 273–282, 2000.

[Lobo07] Lobo, F.J.; Lima, Cláudio F.; Michalewicz, Zbigniew (Eds.). *Parameter Setting in Evolutionary Algorithms*. Springer-Verlag, Berlin Heidelberg, 2007.

[CIPR02] C. for Image Processing Research (CIPR). CIPR Still Images. http://www.cipr.rpi.edu/resource/stills/kodak.html, 2002. [Online; last accessed 25-January-2012].

[Goldberg89] D. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989.

[Holland75] J. Holland. *Adaption in Natural and Artificial Systems*. University of Michigan Press, 1975.

[JPEG2000] ISO/IEC. *ISO/IEC 15444-1: JPEG2000 image coding system (2000) Genetic Algorithm for Wavelet Sign Coding 9*. ISO, 2000.

[Oliver06] Jose Oliver and Manuel P. Malumbres. Low-complexity multiresolution image compression using wavelet lower trees. *IEEE Transactions on Circuits and Systems for Video Technology*, 16(11):1437–1444, 2006.

[Lopez11] Otoniel Lopez Granado, Ricardo Garcia, Manuel Perez Malumbres and Antonio Martí. On the use of genetic algorithms to improve wavelet sign coding performance. In *Proceedings of the 11th International Work-Conference on Artificial Neural Networks, IWANN 2011*, pages 505–512, June 2011.

[Shapiro96] J. Shapiro. A fast technique for identifying zerotrees in the ezw algorithm. In *Proceedings of the IEEE Int. Conf. Acoust., Speech, Signal Processing 3*, pages 1455–1458, 1996.

[Taubman00] D. Taubman. High performance scalable image compression with ebcot. *IEEE Transactions on Image Processing*, 9(7):1158–1170, July 2000.

[Weesing11] S. Wessing, M. Preuss, and G. Rudolph. When parameter tuning actually is parameter control. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, GECCO '11, pages 821–828, New York, NY, USA, 2011. ACM.

[Wu97] X. Wu. High-order context modeling and embedded conditional entropy coding of wavelet coefficients for image compression. In *Proceedings of the 31st Asilomar Conf. on Signals, Systems, and Computers*, pages 1378–1382, 1997.