Chapter 5
Evolutionary Techniques for Speech Enhancement

Abstract Genetic Algorithms have become increasingly appreciated as an easy-to-use general method for a wide range of optimization problems. Their principle consists of maintaining and manipulating a population of solutions and implementing a 'survival of the fittest' strategy in their search for better solutions. In this chapter, GAs are combined with a signal subspace decomposition technique to enhance speech that is severely degraded by noise. To evaluate the effectiveness of this hybrid approach, a set of continuous speech recognition experiments is carried out by using the NTIMIT telephone speech database.

Keywords Genetic Algorithms • KLT • Mel-frequency cepstral coefficients • Telephone speech • Channel degradations • NTIMIT database

5.1 Principle of the Method

Genetic algorithms (GAs) are a subset of evolutionary computation [37] that mimic the process of natural evolution. To perform such process, GAs implement mechanisms inspired by biological evolution such as selection, recombination and mutation, applied in a pool of individuals belonging to a same population. The fittest individuals, that represent parameters to optimize, are encouraged to reproduce and survive to the next generation, thus improving successive generations. A small proportion of inferior individuals can also be selected to survive and also reproduce. Recombination and mutation create the necessary diversity and therefore facilitate novelty, while selection is used to increase quality. Many aspects of such an evolutionary process are stochastic. In this chapter, GAs are used to overcome the limit of estimating the noise variance in subspace methods. The idea is to exploit the power of GAs to investigate beyond the classical space of solutions by exploring a wide range of promising areas [131, 123]. The approach consists of combining subspace decomposition methods and GAs as a means to determine robust solutions.

S.-A. Selouani, *Speech Processing and Soft Computing*, SpringerBriefs in Electrical and Computer Engineering, DOI 10.1007/978-1-4419-9685-5_5,

5.2 Global Framework of Evolutionary Subspace Filtering Method

The principle of subspace decomposition methods consists of constructing an orthonormal set of axes that point in the directions of maximum variance and the enhancement is performed by estimating the noise variance. As described in the previous chapters, the enhancement is performed by assuming that the clean speech is concentrated in an $r < N$ dimensional subspace (signal subspace) whereas the noise occupies the $N - r$ dimensional observation space. In their pioneering work, Ephraim and Van Trees [41], the noise reduction is obtained through an optimal estimator that would minimize the speech distortion considering the fact that the residual noise fell below a preset threshold. The determination of such a threshold requires a noise variance estimation.

Mathematically, the subspace filtering consists of finding a linear estimate of $\mathbf{s}$ (the clean signal) given by $\hat{\mathbf{s}} = \mathbf{Hx}$, which can be written $\hat{\mathbf{s}} = \mathbf{Hs} + \mathbf{Hn}$ where $\mathbf{H}$ is the enhancement filter and $\mathbf{x}$ the noisy signal. The filter matrix $\mathbf{H}$ can be written as: $\mathbf{H} = \mathbf{QGQ^T}$ in which the diagonal matrix $\mathbf{G}$ contains the weighting factors g_i for the eigenvalues of the noisy speech. In the evolutionary eigendecomposition, the $\mathbf{H}$ matrix becomes $\mathbf{H_{gen}}$ and is given by the following: $\mathbf{H_{gen}} = \mathbf{QG_{gen}Q^T}$ in which the diagonal matrix $\mathbf{G_{gen}}$ contains weighting factors that are optimized using genetic operators. Optimization is reached when the Euclidean distance between $\mathbf{C_{gen}}$ and $\mathbf{C}$, the genetically enhanced and original parameters respectively, is minimized. The space of feature representation is reconstructed by using the eigenvectors weighted by the optimal factors of the $\mathbf{G_{gen}}$ matrix.

By using GAs, no empirical or *a priori* knowledge is needed. The problem of determining optimal order of reconstruction r is avoided since the GA implicitly discovers this optimal order. The complete space dimension N is considered at the beginning of the evolution process. As illustrated in Figure 5.1, the space of feature representation is reconstructed by using the eigenvectors weighed by the optimal factors of $\mathbf{G_{gen}}$ matrix.

5.3 Hybrid KLT-GA Enhancement

The evolution process starts with the creation of a population of the weight factors, g_i, which constitute the individuals. The individuals evolve through many generations in a pool where genetic operators are applied [49]. Some of these individuals are selected to reproduce according to their performance. The individuals' evaluation is performed through the use of an objective function. When the fittest individual (best set of weights) is obtained, it is used, in the test phase, to project the noisy data. Genetically modified MFCCs, their first and second derivatives, are finally used as enhanced features.

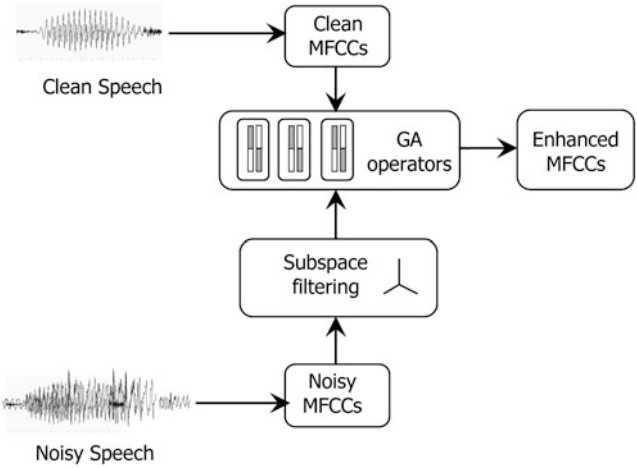

Fig. 5.1 General overview of the KLT-GA-based system.

5.3.1 Solution Representation

A solution representation is needed to describe each individual g_i in the population. A useful representation of individuals involves genes or variables from an alphabet of floating point numbers with values varying within upper and lower bounds a_i, b_i respectively. Concerning the initialization of the pool, the ideal zero-knowledge assumption is to start with a population of completely random values of weights. These values follow a uniform distribution within the upper and lower boundaries.

5.3.2 Selection Function

Selection is the process of determining the number of trials a particular individual is chosen for reproduction. The selection method used here, is the Stochastic Universal Sampling (SUS) introduced by Baker [8]. It consists of transforming raw fitness values into a real-valued expectation of an individual's probability to reproduce, and then to perform the selection based on the relative fitness of individuals. To do that, the individuals g_i are mapped to contiguous segments of a line such that the length of each individual's segment is equal to the value of its fitness. Equal space pointers are then placed over the line as many as the predetermined number (N_s) of individuals to select. The complete SUS procedure is given by Algorithm 5.1.

```
Data: population of g_k, and N_s
Result: index k of individuals selected to reproduce
order population by fitness;
Calculate F_t the total fitness of the population ;
Determine a random number Rand between 0 and F_t/N_s;
for i ← 0 to N_s − 1 do
    calculate f = rand + i ∗ F_t/N_s;
    ptr=0;
    while not at end of population do
        if ptr < f and fitness of g_k + ptr > f then
            return k;
        end
        ptr=ptr+fitness of fitness of g_k;
    end
end
```

Algorithm 5.1: Stochastic universal sampling algorithm for individual selection

5.3.3 Crossover and Mutation

To avoid the extension of the exploration domain in order to reach the best solution, a simple crossover operator can be used [65]. It generates a random number l from a uniform distribution and undergoes an exchange of the genes of the parents (X and Y) on the offspring genes (X' and Y'). It can be expressed by the following equations:

$$\begin{cases} X' = lX + (1-l)Y \\ Y' = (1-l)X + lY. \end{cases} \tag{5.1}$$

In addition to the crossover operator, a mutation is performed. Mutation consists of altering one or more gene values of the individual. This can result in entirely new individual. Through this manipulation, the genetic algorithm prevents the population from stagnating at a given non optimal solution. Usually the mutation rate is low (as in the biological world) and it is fixed according to a user-definable value. If this value is set very high, the search will become a random search. Most mutation methods in canonical GAs are randomly driven. Some methods such as that proposed by Temby *et al.* [132] suggest the use of directed mutation based on the concept of the momentum commonly used in the training of neural networks.

The principle of the mutation-with-momentum algorithm used here, requires that a gene's value has both the standard Gaussian mutation and a proportion of the current momentum term added to it. The update of the momentum term is performed to reflect the combined mutation value. The following equations summarizes the process of Gaussian mutation with momentum. Some individuals are selected and then their original genes, x, produce mutant genes, x_m,

$$\begin{cases} x_m = x + \mathcal{N}(0,1) + \eta M_0 \\ M_m = x_m - x \end{cases} \tag{5.2}$$

where $\mathcal{N}(0, 1)$ is a random variable of normal distribution with zero mean and standard deviation 1 which is to be sampled for each component individually, and η is the parameter controlling the amount of momentum ($0 < \eta < 1$). M_0 is the value of the momentum term for the gene and M_m is the value of the momentum term after mutation. The momentum is updated at each iteration by substituting M_0 by M_m. To prevent the momentum term from becoming large, the difference of Equation 5.2 is limited to a maximum value of M_m.

5.4 Objective Function and Termination

Evolution is driven by an objective function defined in terms of a distance measure between the noisy MFCCs, projected by using the individuals (weights), and the clean MFCCs. The fittest individual is the set of weights which corresponds to the minimum of that distance. As we are using MFCCs, Euclidean distance is considered. The GA must search all the axes generated by the KLT of the Mel-frequency space to find the closest to those of the clean MFCCs. The fittest individual is the axis corresponding to the minimum of that distance. Let's consider two vectors $\mathbf{C}$ and $\hat{\mathbf{C}}$ representing two frames, each with N components, where the geometric distance is defined as:

$$d(\mathbf{C}, \hat{\mathbf{C}}) = \left(\sum_{k=1}^{N} (\mathbf{C_k} - \hat{\mathbf{C}}_\mathbf{k})^l \right)^{1/l}. \tag{5.3}$$

The Euclidean distance corresponds to ($l = 2$). The opposite of this distance, $-d(\mathbf{C}, \hat{\mathbf{C}})$ is used since we have to maximize the fitness function. The evolution process is terminated when a certain number of maximum generations is reached. This number corresponds to the beginning of the objective function convergence.

5.5 Experiments

The evaluation of the hybrid KLT-GA enhancement method is carried out by testing its robustness as it performs a speech recognition over a telephone channel. It is well-known that the limitation of the analysis bandwidth in the telephone channel yields higher speech recognition error rates. In these experiments a HMM-based speech recognition system is trained with high-quality speech and tested by using simulated telephone speech.

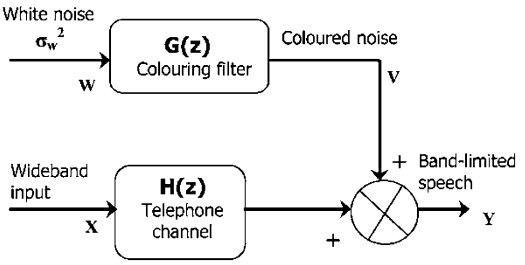

Fig. 5.2 Model of the telephone channel [77].

5.5.1 Speech Databases

In the first set of experiments, the training set providing the clean speech models is composed of the Train subdirectories of the TIMIT database described in [133]. The speech recognition system uses the Test subdirectories of NTIMIT as a test set [72]. The NTIMIT database was created by transmitting TIMIT sentences over a physical telephone network. Previous work on speech recognition systems has demonstrated that the use of speech over the telephone line yields a reduction in accuracy of about 10% [96]. The model used to simulate the telephone channel is described in [77]. Figure 5.2 shows that the wideband input sequence corresponding to TIMIT speech, is bandlimited by $H(z)$, the transfer function simulating the frequency response characteristics of a telephone channel. The channel noise is created by passing zero mean white noise with variance through a second filter $G(z)$ producing a coloured noise. This coloured noise is added to the $H(z)$ output to obtain the telephone speech. In the second set of experiments, NTIMIT is used for both training and test.

5.5.2 Experimental Setup

The baseline HMM-based speech recognition system is designed through the use of the HTK toolkit [66]. Here three systems are compared: the KLT-based system as detailed in Chapter 3, the KLT-GA-based ASR system and the baseline HMM-based system which uses MFCCs and their first and second derivatives as input features (MFCC_D_A). The parameters used to control the run of the genetic algorithm are as follows. The initial population is composed of 250 individuals and was created by duplicating (cloning) the elements of the weighting matrix. In order to insure convergence, we allow the population to evolve through 300 generations, even if no improvement in fitness is observed beyond 200 generations, as is shown in Figure 5.3. The percentages of crossover rate and mutation rate are fixed respectively at 35% and 3%. The number of total runs was fixed at 80. In order to make an adequate comparison with the baseline front-end, after the GA processing, the MFCCs static vectors are expanded to produce a 39-dimensional (static+dynamic) vector.

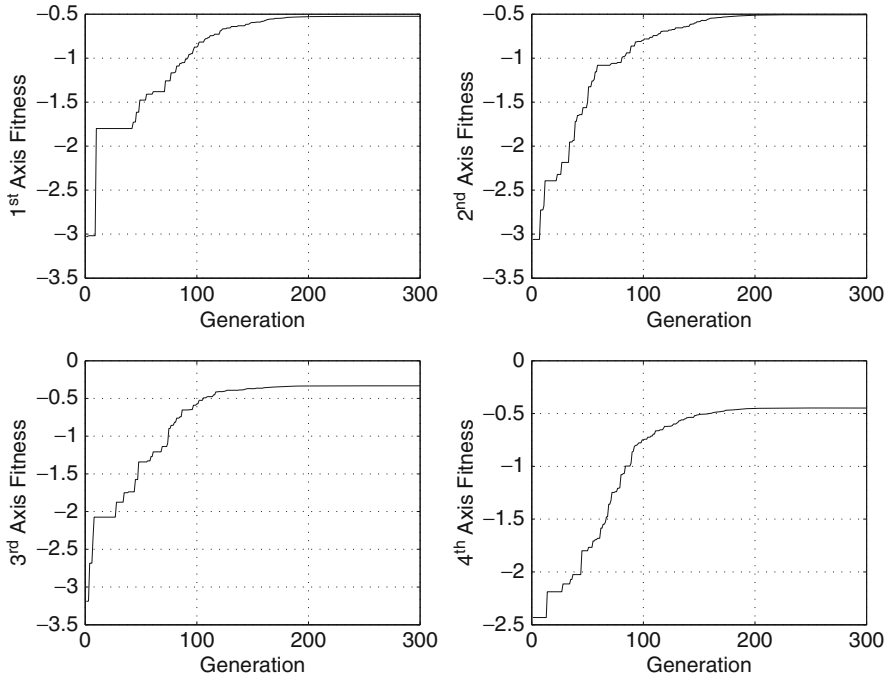

Fig. 5.3 Fitness variation of the first optimized weights of **G** matrix with respect to the number of generations within the evolutionary process.

5.5.3 Performance Evaluation

The results presented in Table 5.1 show that the use of the KLT-GA as a pre-processing approach to enhance the MFCCs that were used for recognition with 8-mixture Gaussian HMMs using tri-phone models, leads to a significant improvement in the accuracy of speech recognition. A correct rate of 34.49% is reached by the KLT-GA-MFCC_D_A-based CSR system when the baseline and the KLT-baseline (using the MDL criterion) systems achieve 18.02% and 27.73% respectively. This represents an improvement of more than 16% compared to the baseline system. Expanding to more than 8 mixtures did not improve the performance. Another set of experiments is carried out by applying the Cepstral Mean Normalization (CMN) to the MFCCs prior to the evolutionary subspace filtering. CMN is a widely used method for improving the robustness of speech recognition to channel distortions. The principle of CMN consists of performing a bias subtraction from the observation sequence (MFCC vector) resulting in a sequence that has a zero mean vector. In these experiments, the CMN included in the HTK toolkit is used [66]. The results show that the CMN has a significant impact on the baseline system using the MFCCs and their derivatives. An improvement of more than 4% is noticed.

Table 5.1 Percentages of word recognition rate ($\%C_{Wrd}$), insertion rate ($\%\epsilon_{Ins}$), deletion rate ($\%\epsilon_{Del}$), and substitution rate ($\%\epsilon_{Sub}$) of the MFCC_D_A, KLT-MFCC_D_A, and KLT-GA-MFCC_D_A ASR systems using 8-mixture tri-phone models. The cepstral mean normalisation (CMN) is also tested as preprocessing for all ASR systems. (Best rates are highlighted in boldface).

	$\%\epsilon_{Sub}$	$\%\epsilon_{Del}$	$\%\epsilon_{Ins}$	$\%C_{Wrd}$
MFCC_D_A	78.02	3.96	40.83	18.02
KLT-MFCC_D_A	66.95	5.32	31.74	27.73
KLT-GA-MFCC_D_A	60.85	4.66	29.56	**34.49**
[a] TIMIT is used for the training and NTIMIT for the test.				
MFCC_D_A	47.02	2.78	24.62	50.20
KLT-MFCC_D_A	39.36	3.37	18.62	57.61
KLT-GA-MFCC_D_A	33.64	3.24	10.39	**63.12**
[b] NTIMIT is used for the training and NTIMIT for the test.				
CMN-MFCC_D_A	74.54	3.28	38.36	22.18
CMN-KLT-MFCC_D_A	66.09	5.76	30.48	28.15
CMN-KLT-GA-MFCC_D_A	60.39	4.79	29.85	**34.82**
[c] TIMIT is used for the training and NTIMIT for the test. The CMN preprocessing is applied to the MFCCs.				
CMN-MFCC_D_A	43.58	2.04	21.58	54.38
CMN-KLT-MFCC_D_A	40.30	3.28	18.47	56.42
CMN-KLT-GA-MFCC_D_A	33.42	3.35	10.56	**63.23**
[d] NTIMIT is used for the training and NTIMIT for the test. The CMN preprocessing is applied to the MFCCs.				

However, the effect of the CMN preprocessing is very limited on the systems using KLT and GAs. Indeed, for the CMN-KLT-GA-MFCC_D_A, a little decrease (less than 1%) is noticed when NTIMIT is used for both training and test.

5.6 Summary

The approach described in this chapter can be viewed as a transformation via a mapping operator using a Mel-frequency subspace decomposition and GAs. The results show that this evolutionary eigendomain KLT-based transformation achieves an enhancement of MFCCs in the context of telephone speech. The improvement obtained over telephone lines demonstrates that the KLT-GA hybrid enhancement scheme succeeds in obtaining less-variant MFCC parameters under telephone-channel degradation. This indicates that both subspace filtering and GA-based optimization gained from the hybridization of the two approaches. It should be noted that the use of soft-computing technique leads to less complexity than many other enhancement techniques that need to either model or compensate for the noise.