A Spectral Conversion Approach to Feature Denoising and Speech Enhancement

A. Mouchtaris[1,2], *J. Van der Spiegel*[2], *P. Mueller*[3], *and P. Tsakalides*[1]

[1]Institute of Computer Science
Foundation for Research and Technology
Heraklion, Crete
Greece 71110

[2] Electrical and Systems Engineering
University of Pennsylvania
Philadelphia, PA
USA 19104

[3] Corticon Inc.
King of
Prussia, PA
USA 19406

Abstract

In this paper we demonstrate that spectral conversion can be successfully applied to the speech enhancement problem as a feature denoising method. The enhanced spectral features can be used in the context of the Kalman filter for estimating the clean speech signal. In essence, instead of estimating the clean speech features and the clean speech signal using the iterative Kalman filter, we show that is more efficient to initially estimate the clean speech features from the noisy speech features using spectral conversion (using a training speech corpus) and then apply the standard Kalman filter. Our results show an average improvement compared to the iterative Kalman filter that can reach 6 dB in the average segmental output Signal-to-Noise Ratio (SNR), in low input SNR's.

1. Introduction

The spectral conversion method has the objective of estimating spectral parameters with specific target statistics from spectral parameters with specific source statistics, using training data as a means of deriving the estimation parameters. Spectral conversion has been defined within the voice conversion problem, where the objective is to modify the speech characteristics of a particular speaker in such manner, as to sound like speech by a different target speaker (for example [1, 2] and references therein). In this paper (the background noise is assumed to be colored and possibly nonstationary), and in our previous work [3] (the noise was assumed to be white), we have applied spectral conversion to the speech enhancement problem, by considering the problem of speech enhancement in additive noise as similar to voice conversion, where the source speech is the noisy speech, and the target speech is the clean speech. In essence, we practically demonstrate that spectral conversion can be viewed as a very useful estimation method outside the context of voice conversion. In this paper, our objective is to apply spectral conversion as a feature denoising method for speech enhancement, within the Kalman filter framework. Although it is possible to directly use the converted features for synthesizing an enhanced speech signal (using the noisy speech residual), we usually obtain perceptually better speech quality when we use the new features as a means for estimating the parameters of an "optimal" linear filter.

Use of spectral conversion for speech enhancement produces better estimates of the speech spectral features at the expense of the requirement for training data. In many practical scenarios, however, it is possible to have *a priori* access to clean speech signals, and many popular algorithms for speech enhancement have been developed under this assumption, such as HMM-based algorithms [4]. A significant similarity of such approaches is the use of mixture models for the probability density function (pdf) of the spectral features. In contrast with many corpus-based approaches, our spectral conversion methods do not assume any model for the background noise and do not require any noise training data. Our methods (in addition to the clean speech signals) require access to the noisy speech signal for training, which is readily available. The feature denoising approach proposed here is mostly similar to the SPLICE method of [5], which also requires clean and noisy speech for training (mentioned as stereo training data or parallel corpus), and like our methods does not assume noise stationarity. In fact, this method is very similar to the parallel training algorithm that we describe later. The main purpose of this paper, though, is to introduce our previously derived non-parallel training algorithm [2] to the problem of speech enhancement. The advantage of this method when compared to parallel training and SPLICE is the fact that there is no need for parallel training data (*i.e.* the clean and noisy speech need not contain the same context). For this algorithm, initial conversion (estimation) parameters are obtained from a different speaker and noise characteristics pair, and then adapted to the speaker and noise characteristics of interest, through a parameter adaptation procedure similar to what is encountered in speech recognition. The training phase is simplified with this latter approach, since only few sentences of clean speech are needed, while the noisy speech is readily available. It is important to note that in this paper we employ a user-centric approach, *i.e.* the speech data we use for training come from the same speaker whose speech we attempt to enhance. In many scenarios this is possible to implement in practice, however our future plans include testing our algorithm with more restrictions regarding the training procedure.

2. Kalman Filter for Speech Enhancement

In this paper, we assume that $y(n)$ is the noisy signal, $s(n)$ is the clean speech signal, and $d(n)$ is the additive noise that is uncorrelated with $s(n)$. We follow the method of [6]. The algorithms that we describe operate successively in analysis *frames* of the signals (*i.e.* batch processing). For each frame, the speech signal is assumed to follow an autoregressive (AR) model

$$s(n) = -\sum_{i=1}^{p} a_i s(n-i) + \sqrt{g_s}u(n), \tag{1}$$

where $u(n)$ is the excitation signal, assumed to be white noise with zero mean and unit variance, g_s is the spectral level, and a_i are the AR coefficients (order p). The noise is assumed as possibly non-white and more specifically to follow an AR model similar to (1) with $w(n)$ as the zero-mean unit-variance white noise, g_d the noise spectral level, and b_i the AR coefficients (order q). These equations can be written in state-space form as

$$\begin{aligned} \boldsymbol{x}(n) &= \boldsymbol{\Phi}\boldsymbol{x}(n-1) + \mathbf{G}\boldsymbol{r}(n) \\ y(n) &= \boldsymbol{h}^T\boldsymbol{x}(n), \end{aligned} \tag{2}$$

where the state vector $\boldsymbol{x}(n)$ is given by

$$\begin{aligned}\boldsymbol{x}^T(n) &= [\boldsymbol{s}_{p-1}^T(n-1)\ s(n)\ \boldsymbol{d}_{q-1}^T(n-1)\ d(n)] \\ \boldsymbol{s}_p^T(n) &= [s(n-p+1)\ s(n-p+2)\ \cdots\ s(n)]^T \\ \boldsymbol{d}_q^T(n) &= [d(n-q+1)\ d(n-q+2)\ \cdots\ d(n)]^T. \quad (3)\end{aligned}$$

The state transition matrix $\mathbf{\Phi}$ can be easily found from the AR speech and noise models, and contains the AR coefficients of the speech and noise processes. Finally, $\mathbf{G}$ is a matrix containing $\sqrt{g_s}$ and $\sqrt{g_d}$, $\boldsymbol{h}$ is a vector of $p+q$ zeros with 1 placed only in the p^{th} and $(p+q)^{th}$ places, and $\boldsymbol{r}(n) = [u(n)\ w(n)]^T$. If the parameters a_i, b_i, g_s and g_d were known, then matrices $\mathbf{\Phi}$ and $\mathbf{G}$ would be known and the standard Kalman filter would be obtained, that provides the optimal MMSE (minimum mean-squared error) estimate of the state vector (and thus the clean speech signal). In practice, however, these parameters are not available. The KEMI algorithm of [6] estimates these parameters iteratively, within the Kalman filter algorithm.

The Kalman EM Iterative (KEMI) algorithm, uses the Expectation Maximization (EM) algorithm for iteratively estimating the speech and noise AR model parameters, applying the Kalman filter at each iteration. We use the notation $\boldsymbol{a} = [a_p\ a_{p-1}\ \cdots\ a_1]^T$, $\boldsymbol{b} = [b_q\ b_{q-1}\ \cdots\ b_1]^T$, $\boldsymbol{\theta} = [\mathbf{a}^T\ g_s\ \boldsymbol{b}^T\ g_d]^T$. Also, $\hat{\boldsymbol{\theta}}^{(l)}$ denotes the estimate of $\boldsymbol{\theta}$ after the l^{th} iteration. We denote $\boldsymbol{y} = [y(1)\ y(2)\ \cdots\ y(N)]^T$ as the vector of measurements for the current analysis frame. We denote as $\widehat{(\cdot)} = E_{\hat{\boldsymbol{\theta}}^{(l)}}(\cdot|\boldsymbol{y})$. To obtain the parameter estimate at iteration $l+1$ we use the following two-step iterative procedure:
E-STEP: We denote the current state estimate and state covariance estimate respectively as

$$\boldsymbol{\mu}(n|N) = \widehat{\boldsymbol{x}(n)},\ \mathbf{P}(n|N) = \widehat{\boldsymbol{x}(n)\boldsymbol{x}^T(n)} - \widehat{\boldsymbol{x}(n)}\widehat{\boldsymbol{x}(n)}^T. \quad (4)$$

These can be found using the well-known Kalman filter recursion, followed by the smoothing recursion. We omit the equations here due to lack of space, the interested reader is referred to [6]. The estimation equations are similar to the standard Kalman filter, with the difference that matrices $\mathbf{\Phi}$ and $\mathbf{G}$ are only estimates (from the M-Step of the previous iteration), which is the reason that this iterative EM procedure is needed. The E-Step is followed by the M-Step providing the parameter estimates for the next iteration:
M-Step: The parameter estimates are given by:

$$\hat{\boldsymbol{a}}^{(l+1)} = -\left[\sum_{i=1}^{N} \widehat{\boldsymbol{s}_p(n-1)\boldsymbol{s}_p(n-1)^T}\right]^{-1} \sum_{i=1}^{N} \widehat{\boldsymbol{s}_p(n-1)s(n)}$$

$$\hat{\boldsymbol{b}}^{(l+1)} = -\left[\sum_{i=1}^{N} \widehat{\boldsymbol{d}_q(n-1)\boldsymbol{d}_q(n-1)^T}\right]^{-1} \sum_{i=1}^{N} \widehat{\boldsymbol{d}_q(n-1)d(n)}$$

$$\begin{aligned} g_s^{(\hat{l+1})} &= \frac{1}{N}\sum_{i=1}^{N}[\hat{s}^2(n) + (\hat{\boldsymbol{a}}^{(l+1)})^T \widehat{\boldsymbol{s}_p(n-1)s(n)}] \\ g_d^{(\hat{l+1})} &= \frac{1}{N}\sum_{i=1}^{N}[\hat{d}^2(n) + (\hat{\boldsymbol{b}}^{(l+1)})^T \widehat{\boldsymbol{d}_q(n-1)d(n)}]. \quad (5)\end{aligned}$$

All the various estimates that are necessary in the above equations can be obtained as submatrices of $\widehat{\boldsymbol{x}(n)\boldsymbol{x}^T(n)} = \mathbf{P}(n|N) + \boldsymbol{\mu}(n|N)\boldsymbol{\mu}^T(n|N)$. It is of interest to note the similarity of the above equations with the Yule-Walker equations. For the remainder of this paper, we use the delayed Kalman filter estimate (fixed-lag smoothing) for reducing the computational complexity of the algorithm. This means that we use $\hat{s}(n-p+1)$ as the current signal estimate (delay of $p-1$ samples), which is the first entry of $\boldsymbol{\mu}(n|N)$, and similarly for the noise estimate. The advantage of fixed-lag smoothing is that the smoothing equations need not be computed, which results in significantly less computations, while good performance is retained. Note that an initialization of the speech and noise AR parameters is required, which can be simply obtained from the noisy speech. Higher-order statistics can alternatively be used for the initialization [6]; in our experiments this procedure did not offer any advantage and thus was not applied.

In the next two sections we provide an alternate approach to the initialization of the AR speech parameters needed in KEMI. In Section 3, we present an estimation procedure of the clean speech AR parameters based on the noisy parameters, using a parallel training corpus, while in Section 4 a similar procedure is applied, which does not require a parallel speech corpus.

3. Spectral Conversion

From the reference and target training waveforms, we extract the parameters that model their short-term spectral properties (here we use the line spectral frequencies - LSF's - due to their desirable interpolation properties [1]). Note that LSF's have a 1-1 correspondence with the AR filter coefficients that are needed for the Kalman filter. The objective of spectral conversion methods is to derive a function $\mathcal{F}(\cdot)$ which, when applied to spectral vector $\boldsymbol{x}_k$, produces a vector close in some sense to vector $\boldsymbol{y}_k$. For the noise enhancement problem, the vector sequence $\boldsymbol{x}_k$ corresponds to the spectral vectors of noisy speech, while the sequence $\boldsymbol{y}_k$ corresponds to the spectral vectors of clean speech. Gaussian mixture models (GMM's) have been successfully applied to the voice conversion problem. Here we follow the approach of [1].

Assuming that $\boldsymbol{x}$ and $\boldsymbol{y}$ are jointly Gaussian for each class ω_i, in MS sense the optimal choice for the function $\mathcal{F}$ is

$$\mathcal{F}(\boldsymbol{x}_k) = \sum_{i=1}^{M} p(\omega_i|\boldsymbol{x}_k)\left[\boldsymbol{\mu}_i^y + \boldsymbol{\Sigma}_i^{yx}{\boldsymbol{\Sigma}_i^{xx}}^{-1}(\boldsymbol{x}_k - \boldsymbol{\mu}_i^x)\right], \quad (6)$$

where the conditional probabilities $p(\omega_i|\boldsymbol{x}_k)$ are given from

$$p(\omega_i|\boldsymbol{x}_k) = \frac{p(\omega_i)\mathcal{N}(\boldsymbol{x}_k;\boldsymbol{\mu}_i^x,\boldsymbol{\Sigma}_i^{xx})}{\sum_{j=1}^{M} p(\omega_j)\mathcal{N}(\boldsymbol{x}_k;\boldsymbol{\mu}_j^x,\boldsymbol{\Sigma}_j^{xx})}. \quad (7)$$

All the parameters in the two above equations are estimated using the EM algorithm on the joint model of $\boldsymbol{x}$ and $\boldsymbol{y}$. This means that the EM algorithm is performed on the concatenated vector sequence $[\boldsymbol{x}_k^T\boldsymbol{y}_k^T]^T$. A time-alignment procedure is required in this case, which is only possible when a parallel corpus is used. We use a diagonal covariance implementation, as in [2].

4. Constrained GMM Estimation

We assume that a parallel speech corpus is available for a different speaker and noise conditions (*reference pair*), in addition to the particular pair of speaker and noise for which only a non-parallel corpus exists (*target pair*). The random vector $\boldsymbol{x}'$ represents the spectral vectors of the noisy speech of the target pair, while random vector $\boldsymbol{x}$ corresponds to the noisy speech of the reference pair. Consider that these are related by a probabilistic linear transformation [7], for each Gaussian class ω_i of $\boldsymbol{x}$

$$\boldsymbol{x}' = \mathbf{A}_j\boldsymbol{x} + \boldsymbol{b}_j \ \text{ with probability } p(\lambda_j|\omega_i), \quad j = 1,\ldots,N. (8)$$

In the above equations $\mathbf{A}_j$ is a $K \times K$ matrix (K is the dimensionality of $\boldsymbol{x}$), and $\boldsymbol{b}_j$ is a vector of the same dimension with $\boldsymbol{x}$. Random vectors $\boldsymbol{y}'$ and $\boldsymbol{y}$ correspond to the clean speech of the target and reference pairs respectively, and are related by another probabilistic linear transformation, similar to (8), where matrix $\mathbf{A}_j$ is now substituted by $\mathbf{C}_\rho$, vector $\boldsymbol{b}_j$ becomes $\boldsymbol{d}_\rho$, and $p(\lambda_j|\omega_i)$ becomes $p(\kappa_\rho|\omega_i)$, for $\rho = 1,\ldots,L$. Note that classes ω_i are the same for $\boldsymbol{x}$ and $\boldsymbol{y}$ by design in Section 3. All the unknown parameters can be estimated by use of the non-parallel corpus and the GMM of the parallel corpus, by applying the EM algorithm. It can be shown that the conversion function for the non-parallel case becomes [2]

$$\begin{aligned}\mathcal{F}(\boldsymbol{x}_k^{'}) &= \sum_{i=1}^{M}\sum_{j=1}^{N}\sum_{\rho=1}^{L} p(\omega_i|\boldsymbol{x}_k^{'})p(\lambda_j|\boldsymbol{x}_k^{'},\omega_i)p(\kappa_\rho|\omega_i)\\ &\quad \Big[\mathbf{C}_\rho\boldsymbol{\mu}_i^y + \boldsymbol{d}_\rho + \mathbf{C}_\rho\boldsymbol{\Sigma}_i^{yx}\boldsymbol{\Sigma}_i^{xx^{-1}}\mathbf{A}_j^{-1}\\ &\quad \Big(\boldsymbol{x}_k^{'} - \mathbf{A}_j\boldsymbol{\mu}_i^x - \boldsymbol{b}_j\Big)\Big],\end{aligned}$$

$$p(\omega_i|\boldsymbol{x}_k^{'}) = \frac{p(\omega_i)\sum_{j=1}^{N}p(\lambda_j|\omega_i)\mathrm{g}(\boldsymbol{x}_k^{'}|\omega_i,\lambda_j)}{\sum_{i=1}^{M}\sum_{j=1}^{N}p(\omega_i)p(\lambda_j|\omega_i)\mathrm{g}(\boldsymbol{x}_k^{'}|\omega_i,\lambda_j)}, \quad (9)$$

$$p(\lambda_j|\boldsymbol{x}_k^{'},\omega_i) = \frac{p(\lambda_j|\omega_i)\mathrm{g}(\boldsymbol{x}_k^{'}|\omega_i,\lambda_j)}{\sum_{j=1}^{N}p(\lambda_j|\omega_i)\mathrm{g}(\boldsymbol{x}_k^{'}|\omega_i,\lambda_j)}, \quad (10)$$

$$\mathrm{g}(\boldsymbol{x}^{'}|\omega_i,\lambda_j) = \mathcal{N}(\boldsymbol{x}^{'};\mathbf{A}_j\boldsymbol{\mu}_i^x + \boldsymbol{b}_j, \mathbf{A}_j\boldsymbol{\Sigma}_i^{xx}\mathbf{A}_j^T). \quad (11)$$

5. Simulation Results

In this section we measure the performance of our two proposed algorithms (parallel and non-parallel conversion) as an improvement to the Kalman filter for speech enhancement. We use a clean speech corpus named *VOICES*, available from OGI's CSLU [8, 9]. This is a parallel corpus and is used for both the parallel and non-parallel training cases that are examined in this section, in a manner explained later in this section. The background noise, added artificially to the speech signals, is car interior noise (with constant acceleration). This type of noise is colored with a low degree of nonstationarity. The noise and speech signals were downsampled to 8 kHz for reducing the implementation demands of the various methods. We implemented and tested, in addition to our two proposed algorithms, the original KEMI algorithm of [6], as well as the LSAE algorithm of [10], for comparison. The latter has been shown to exhibit very desirable performance in [6] compared to the KEMI algorithm in output signal-to-noise ratio (SNR) sense.

In our implementation we use a 32 msec. analysis frame and (for the Kalman-based methods) LSF vectors of 22^{nd} order for the speech signal (4^{th} for the noise). The noise parameters were initialized (noise estimation for LSAE) using very few signal segments that did not contain any speech (initial segments of each recording). The error measure employed is the output average segmental SNR,

$$\text{ASSNR(dB)} = \frac{1}{n}\sum_{k=1}^{n} 10\log_{10}\left(\frac{\boldsymbol{x}_k^T\boldsymbol{x}_k}{(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_{\boldsymbol{k}})^T(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_{\boldsymbol{k}})}\right),$$

where $\boldsymbol{x}_k$ is the clean speech signal for segment k, and $\hat{\boldsymbol{x}}_k$ is the estimated speech signal for segment k (batch processing). We test the performance of the algorithms using the ASSNR for various values of input (global) SNR. We test the performance of the two algorithms proposed here (one case (6) for parallel training and one (9) for non-parallel training), in comparison to the original KEMI algorithm and LSAE. The ideal error for both our methods (the desired LSF's with zero prediction error, only available in the simulation environment) is also given. It is important to note that the corpus used contains a total of 50 sentences, of which a total of 40 is used for training purposes (as explained next) and the remaining 10 are used for testing. All the results given in this section are averaged over these 10 sentences, with different noise segments added to each sentence.

In Fig. 1, the ASSNR is given for the five cases tested, for various values of input SNR. The five cases are: the two

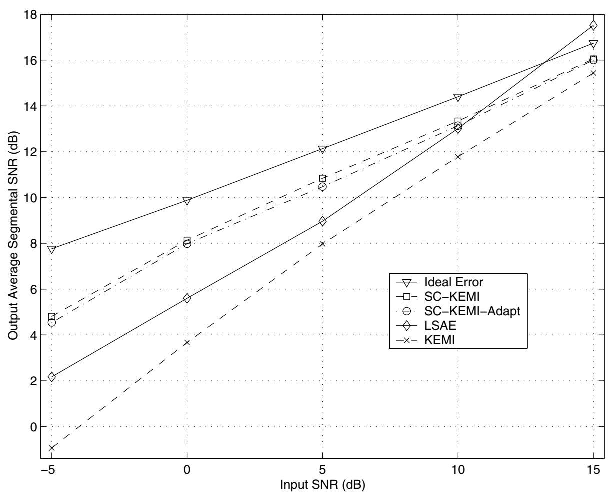

Figure 1: ASSNR (dB) for different values of input SNR, for the five cases tested, *i.e.* perfect prediction (Ideal Error), the Iterative Kalman filter (KEMI), Spectral Conversion followed by KEMI (SC-KEMI, parallel corpus), Spectral Conversion by adaptation followed by KEMI (SC-KEMI-Adapt, non-parallel corpus), and Log-Spectral Amplitude Estimation (LSAE).

proposed algorithms for parallel and non-parallel training as an initialization to the KEMI algorithm (SC-KEMI and SC-KEMI-Adapt respectively), the KEMI algorithm (iterative Kalman filter), the Log-Spectral Amplitude Estimation (LSAE) algorithm, as well as the theoretically best possible performance of our two proposed methods (desired LSF's with zero prediction error). It is important to mention that the results for both our methods, as well as their ideal error performance, were obtained without use of the iterative Kalman procedure. In other words, the results were obtained by LSF estimation followed by the standard Kalman filter. We found that further iterations did not offer any significant improvement. For the KEMI algorithm we obtained good results after 15 iterations. For the results in Fig. 1, we used around 20,000 training LSF vectors, which correspond to 40 sentences of the corpus. Later in this section we discuss the effect of the size of the training corpus to the final results. Also, the number of (diagonal) GMM classes used for both the parallel and non-parallel methods is 16 ($M = 16$ in (6) and (9)), while the number of adaptation parameters is 4 for both the source and target speech ($L = N = 4$ in (9)). For this figure, we plot the performance of the SC-KEMI-Adapt algorithm based on adaptation of the GMM conversion parameters of a different speaker from our corpus, in car interior noise of 10 dB SNR (*i.e.* the SNR is accurate only for the 10 dB input SNR case). From Fig. 1, we can see that the improvement in the KEMI algorithm using both the methods proposed in this paper is significant, especially for low input SNR's. For input SNR of -5 dB for example, the improvement is almost 6 dB for both methods, which is perceptually significant. A very important observation is that the adaptation algorithm performs almost as good as the parallel algorithm. This was not expected, given that we have previously explained (for voice conversion) that adaptation will always perform worse than the parallel method since in parallel training we exploit an additional property of the corpus in an explicit manner. In [2], we have shown that the variations in the estimation error are small between these two algorithms when compared to the distance between the initial and desired parameters. We can conclude that the Kalman filter does not exhibit much sensitivity to the small variations in the estimation error for the initialization parameters in contrast to the case of large estimation errors that are encountered in the original KEMI algorithm (*i.e.* estimating the clean parameters directly from the noisy speech) This is also encountered later

GMM's	2	4	8	16	32
ASSNR	8.1769	8.1338	8.1564	8.1329	8.0073
Vectors	500	1000	2000	5000	20000
ASSNR	7.9333	7.9784	8.0715	8.0611	8.1329

Table 1: Resulting ASSNR in dB (parallel training, 0 dB input SNR), for different numbers of GMM parameters (for 20,000 vectors) and training vectors (for 16 GMM parameters).

in this section, when comparing the ASSNR when fine-tuning the GMM and adaptation parameters (Tables 1 and 2). In high input SNR's the algorithms perform similarly (with the LSAE resulting in the best estimation results for 15 dB SNR), which is sensible since in high SNR's the speech initial parameters estimation from the noisy speech is very close to the desired.

In Table 1, the ASSNR is given for the parallel case (SC-KEMI) for 0 dB input SNR, for various numbers of GMM parameters and training vectors. When comparing the performance of the various numbers of GMM parameters, the vectors in training are 20,000. The number of GMM parameters does not have an influence on the performance of the algorithm. For the second case examined in this table, namely the effect of the training dataset size on the algorithm performance, we use 16 GMM parameters. We can see that the performance of the algorithm improves slightly when more training vectors are available. The fact that only a small number of training data results in significant improvement over KEMI is important, since this corresponds to requiring only a small amount of clean speech data. The fact that we have such a significant improvement in the KEMI algorithm without large variations in the number of GMM parameters or training data is consistent with our previous observation (when comparing parallel *vs.* non-parallel training), that the Kalman filter is not influenced much by small variations in the LSF estimation error.

In Table 2, the ASSNR is given for the non-parallel case (SC-KEMI-Adapt) and input SNR of 0 dB, for various choices of adaptation parameters ($L = N$ in (9)) and training dataset. When varying the number of adaptation parameters, the training dataset contains 20,000 vectors, and when varying the number of vectors in the training dataset, the number of adaptation parameters is $L = N = 4$. For the results in this table, the noise conditions of the parallel (*reference*) pair (*i.e.* initial conversion parameters) were obtained for white noise of 10 dB SNR. This choice was made so that we can show more evidently the effect of adaptation on the algorithm performance, since in this case the initial error (*i.e.* with no adaptation) is much larger than in the case when the reference pair contains the same type (car interior) noise. With no adaptation, *i.e.* simply applying the GMM parameters of a different speaker/noise pair to the speaker in car noise environment, the ASSNR is only 0.3359, which is worse than the original KEMI results for 0 dB SNR (3.6702 to be specific). On the other hand, we observe once again the lack of sensitivity of the Kalman filter to small LSF estimation errors (as long as the adaptation procedure is employed). We also observe that, similarly to the parallel case of Table 1, increasing the number of training vectors consistently improves the algorithm performance, although not significantly. The fact that even a small number of training data results in good algorithm performance is very positive, since in many cases gathering large numbers of data is impractical.

6. Conclusions

Two algorithms were presented in this paper for improving the initial parameter estimate of the iterative Kalman filter for speech enhancement. They were both based on previous work in the area of voice conversion, one of them on a well-known parallel training procedure and the other one on our method for non-parallel training. They were both shown to provide good estimates of the clean speech parameters (from the noisy speech) needed by the Kalman filter, by producing results that are far superior to the original iterative Kalman filtering method. Both methods are useful only in the case that training speech data are *a priori* available. However, our non-parallel method offers important practical advantages regarding the collection of the speech corpus, since there is only need for a small amount of clean speech data, and from the noisy speech (which is readily available), in a non-parallel fashion.

Param.	0	1	2	4	6
ASSNR	0.3359	8.1757	8.0450	8.2340	8.0674
Vectors	500	1000	2000	5000	20000
ASSNR	7.7370	8.1359	7.9530	8.2912	8.2340

Table 2: Resulting ASSNR in dB (non-parallel training, 0 dB input SNR), for different numbers of adaptation parameters (for 20,000 vectors) and training vectors (4 adaptation parameters).

7. Acknowledgments

This research was supported by the Greek General Secretariat for Research and Technology, Program ΕΠΑΝ Code 01EP111 and by a Marie Curie International Reintegration Grant within the 6th European Community Framework Programme.

8. References

[1] A. Kain and M. W. Macon, "Spectral voice conversion for text-to-speech synthesis," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, (Seattle, WA), pp. 285–289, May 1998.

[2] A. Mouchtaris, J. Van der Spiegel, and P. Mueller, "Non-parallel training for voice conversion by maximum likelihood constrained adaptation," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, (Montreal, Canada), pp. 1–4, May 2004.

[3] A. Mouchtaris, J. Van der Spiegel, and P. Mueller, "A spectral conversion approach to the iterative Wiener filter for speech enhancement," in *IEEE Proc. Int. Conf. Multimedia and Expo (ICME)*, (Taipei, Taiwan), pp. 1971–1974, June 2004.

[4] Y. Ephraim, D. Mallah, and B.-H. Juang, "On the application of hidden Markov models for enhancing noisy speech," *IEEE Trans. Acoust., Speech, and Signal Process.*, vol. 37, pp. 1846–1856, December 1989.

[5] L. Deng, J. Droppo, and A. Acero, "Recursive estimation of nonstationary noise using iterative stochastic approximation for robust speech recognition," *IEEE Trans. Speech and Audio Processing*, vol. 11, pp. 568–580, November 2003.

[6] S. Gannot, D. Burshtein, and E. Weinstein, "Iterative and sequential Kalman filter-based speech enhancement algorithms," *IEEE Trans. Speech and Audio Processing*, vol. 6, pp. 373–385, July 1998.

[7] V. D. Diakoloukas and V. V. Digalakis, "Maximum-likelihood stochastic-transformation adaptation of hidden Markov models," *IEEE Trans. Speech and Audio Processing*, vol. 7, pp. 177–187, March 1999.

[8] A. Kain, *High Resolution Voice Transformation*. PhD thesis, OGI School of Science and Engineering at Oregon Health and Science University, October 2001.

[9] http://www.cslu.ogi.edu/corpora/voices/.

[10] Y. Ephraim and D. Mallah, "Speech enhancement using a minimum mean-square error log-spectral amplitude estimator," *IEEE Trans. Acoust., Speech, and Signal Process.*, vol. ASSP-33, pp. 443–445, April 1985.