Frequency Analysis of Spoken Urdu Numbers Using MATLAB and Simulink

S K Hasnain *, Azam Beg ** and Muhammad Samiullah Awan ***
Pakistan Navy Engineering College (NUST), Karachi-75350(Pakistan)

Abstract
This paper describes the frequency analysis of spoken Urdu numbers from '*sifr*' (zero) to '*nau*' (nine). Sound samples from multiple speakers were utilized to extract different features. Initial processing of data, i.e., normalizing and time-slicing was done using a combination of Simulink and MATLAB. Afterwards, the same tools were used for calculation of Fourier descriptions and correlations. The correlation allowed comparison of the same words spoken by the same and different speakers. The analysis presented in this paper is seen as the first step in creating an Urdu speech recognition system. Such a system can be potentially utilized in implementation of a voice-driven help setup at call centers of commercial organizations operating in Pakistan/India region.

Keywords: Spoken Urdu number processing, Fourier descriptors, Correlation, Speaker independent system, Feature extraction, Simulation.

I. INTRODUCTION
Automatic speech recognition has been an active research topic for more than four decades. With the advent of digital computing and signal processing, the problem of speech recognition was clearly posed and thoroughly studied. These developments were complemented with an increased awareness of the advantages of conversational systems. The range of the possible applications is wide and includes: voice-controlled appliances, fully featured speech-to-text software, automation of operator-assisted services, and voice recognition aids for the handicapped [1].

The speech recognition problem has sometimes been treated as a speech-to-text conversion problem. Many researchers have worked in this regard. Some commercial software is also available in the market for speech recognition, but mainly in English and other European languages.

Correlation exists between objects, phenomena, or signals and occurs in such a way that it cannot be by chance alone. Unconsciously, the correlation is used every day life. When one looks at a person, car or house, one's brain tries to match the incoming image with hundreds (or thousands) of images that are already stored in memory [2]. We based our current work on the premise that same word spoken by different speakers is correlated in frequency domain.

In the speech recognition research literature, no work has been reported on Urdu speech processing. So we consider our work to be the first such attempt in this direction. The analysis has been limited to number recognition. The process involves extraction of some distinct characteristics of individual words by utilizing discrete (Fourier) transforms and their correlations. The system is speaker-independent and is moderately tolerant to background noise.

2. REVIEW OF DISCRETE TRANSFORMATION & ITS MATLAB IMPLEMENTATION
Discrete Fourier transform (DFT) is itself a sequence rather than a function of continuous variable and it corresponds to equally spaced frequency samples of discrete time Fourier transform of a signal. Fourier series representation of the periodic sequence corresponds to discrete Fourier transform of finite length sequence. So we can say that DFT is used for transforming discrete time sequence *x(n)* of finite length into discrete frequency sequence *X[k]* of finite length. This means that by using DFT, the discrete time sequence *x(n)* is transformed into corresponding discrete frequency sequence *X[k]*[2].

DFT is a function of complex frequency. Usually the data sequence being transformed is real. A waveform is sampled at regular time intervals *T* to produce the sample sequence of *N* sample values, where *n* is the sample number from *n=0* to *n=N-1*.

$$\{x(nT)\} = x(0), x(T),...,x[(N-1)T]$$

The data values $x(nT)$ will be real only when representing the values of a time series such as a voltage waveform. The DFT of $x(nT)$ is then defined as the sequence of complex values $\{X[k\omega]\} = X(0), X(\varpi),.....X[(N-1)\omega]$ in the frequency domain, where ω is the first harmonic frequency given by $\omega = 2\pi / NT$. Thus $X[k\omega]$ has real and imaginary components in general, so that for the *k*th harmonic

* Author for correspondence. E.mail<hasnain@pnec.edu.pk>
** College of Information Technology, UAE University Al-Ain, UAE. E.mail:<abeg@uaeu.ac.ae>
***Iqra University, Karachi Email:< msuawan@yahoo.com>

$$X(k) = R(k) + jI(k)$$
$$|X(k)| = \left[R^2(k) + I^2(k)\right]^{1/2} \quad (2.1)$$

and

X (k) has the associated phase angle

$$\phi(k) = \tan^{-1}\left[I(k)/R(k)\right] \quad (2.2)$$

where *X(k)* is understood to represent *X(kω)*. These equations are therefore analogous to those for the Fourier transform. Note that *N* real data values (in the time domain) transform to *N* complex DFT values (in frequency domain). The DFT values, *X(k)*, are given by:

$$F_D[x(nT)] = \sum_{n=0}^{N-1} x(nT)e^{-jk\omega nT},\ k = 0,1,\ldots,N-1 \quad (2.3)$$

where $\omega = 2\pi/NT$ and F_D denotes the DFT.

$$X[k] = \sum_{n=0}^{N-1} x(nT)e^{-jk2\pi nT/NT}$$

or
$$X[k] = \sum_{n=0}^{N-1} x(nT)e^{-jk2\pi n/N} \quad (2.4)$$

$$r_{xy} = \sum_{n=0}^{N-1} x(n)y(n) \quad (2.5)$$

The Fast Fourier transform (FFT) eliminates most of the repeated complex products in DFT. In C version of signal processing algorithm, there are several different routines for real and complex versions of the DFT and FFT. When these routines are coded into the MATLAB language, they are very slow compared with the MATLAB *fft* routine, which are coded much more efficiently. Furthermore, the MATLAB routines are flexible and may be used to transform real or complex vector of arbitrary length. They meet the requirements of nearly all signal processing applications; consequently, in this paper, the *fft* routines are preferred over all discrete transform operations.

MATLAB's *fft* routine produces a one-dimensional DFT using the FFT algorithm; that is when $[x_K]$ is a real sequence, *fft* produces the complex DFT sequence $[X_m]$. In MATLAB, the length *N* of the vector $[x_K]$ may not be given. Thus both of the following are legal expressions:

```
X=fft(x)
X=fft(x, N)
```
(2.6)

The first expression in (2.6) produces a DFT with the same number of elements as in $[x_K]$, regardless of whether $[x_K]$ is real or complex. In the usual case where $[x_K]$ is real and length *N*, the last *N/2* complex elements of the DFT are conjugates of the first *N/2* elements in the reverse order, in accordance with (2.4). In the unusual case where $[x_K]$ is complex, the DFT consists of *N* independent complex elements. For example, the results of the following commands with N=4 can be easily verified using definition in (2.7).

$$\frac{FFT\ computing\ time}{DFT\ computing\ time} = \frac{1}{2N}\log_2 N \quad (2.7)$$

The results of the following commands with *N=4* can be easily verified with:

```
x=[1 0 0 1];
X=fft(x)
```

In this example, the DFT components $[X_m]=[2, 1-j, 0, 1+j]$ are found from (2.4). The second expression in (2.6) specifies the value of *N* in (2.4), which effectively overrides any previous specification of the length of the vector *x*. thus, the following commands produce the same result:

```
x=[1 0 0 1 3];
X=fft(x,4)
```

The DFT, $x= [X_m]$ has length = 4 is the same as in previous example.

```
x=[1 1];
X=fft(x, 4)
```

$[X_m] =[2, 1-j, 0, 1+j]$

The result here is the same because, when *N* is greater than the length of *x; X* is the DFT of a vector consisting of *x* extended with zeros on the right, from the length of *x* to *N*. (The length of the vector x itself is not increased in the process). The MATLAB library also includes a two dimensional *fft* routine called *fft2*. The routine computes the two-dimensional FFT of any matrix, whose element may be, for example, samples (pixel values) of a two dimensional image.

Usually, some *recognition* occurs when the incoming images bears a strong correlation with an image in memory that "best" corresponds to fit or is most similar to it. This process also helps one distinguish between say, a dog and a cat, a rose and sunflower, or a train and an airplane. A similar approach is used in this investigation, to measure the similarity between two signals. This process is known as *autocorrelation* if the two signals are exactly the same and as *cross-correlation* if the two signals are different. Since correlation measures the similarity between two signals, it is quite useful in identifying a signal by comparing it with a set of known reference signals. The reference signal that results in the lowest value of the correlation with the unknown signals is most likely the identity of the unknown object.

Correlation involves shifting, multiplication and addition (accumulation). The *cross-correlation function* (CCF) is a measure of the similarities or shared properties between two signals. Application of CCF includes cross spectral density,

detection and recovery of signals buried in noise, for example the detection return signals, pattern, and delay measurement. The general formula for cross-correlation $r_{xY}(n)$ between two data sequences $x(n)$ and $y(n)$ each containing N data might therefore be written as:

$$r_{xx} = \sum_{n=0}^{N-1} x(n)\, x(n) \qquad (2.8)$$

The *autocorrelation function* (ACF) involves only one signal and provides information about the structure of the signal or its behaviour in the time domain. It is special form of CCF and is used in similar applications. It is particularly useful in identifying hidden properties.

3. DATA ACQUISITION AND PROCESSING

One of the obvious methods of speech data acquisition is to have a person speak into an audio device such as microphone or telephone. This act of speaking produces a sound pressure wave that forms an acoustic signal. The microphone or telephone receives the acoustic signal and converts it into an analog signal that can be understood by an electronic system. Finally, in order to store the analog signal on a computer, it must be converted to a digital signal.

The data in this paper is acquired by speaking Urdu numbers into a microphone connected to MS-Windows-XP based PC. The data is saved into '.wav' format files. The sound files are processed after passing through a (Simulink) filter, and are saved for further analysis. We recorded the data for fifteen speakers who spoke the same number set, i.e. zero to nine. The sound sample was curtailed for 0.9 minutes.

In general, the digitized speech waveform has a high dynamic range, and can suffer from additive noise. So first, a Simulink model was used to extract and analyze the acquired data; see Fig. 1.

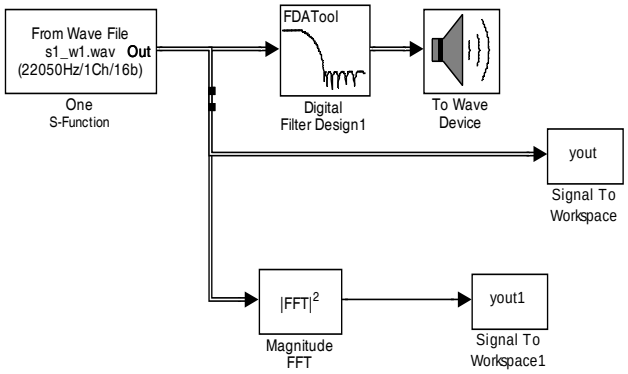

Fig. 1 Simulink model for analyzing Urdu number data

The Simulink model, as shown in Fig. 2, was developed for performing analysis such as standard deviation, mean, median, autocorrelation, magnitude of FFT, data matrix correlation. We also tried a few other statistical techniques, however, most of them failed to provide us any useful insight into the data characteristics. (These are not discussed further for the sake of brevity).

We would also like to mention that we had started our experiments by using Simulink, but soon found this GUI-based tool to be somewhat limited because we did not find it easy to create multiple models containing variations among them. This iterative and variable-nature of models eventually led us to MATLAB's (text-based) .m files. We created these files semi-automatically by using a PERL-language script; the script was developed specifically for this purpose.

Three main data pre-processing steps were required before the data could be used for analysis:

3.1 Pre-Emphasis

By pre-emphasis [5], we imply the application of a *normalization* technique, which is performed by dividing the speech data vector by its highest magnitude.

3.2 Data Length Adjustment

FFT execution time depends on exact number of the samples (N) in the data sequence $[x_K]$, and that the execution time is minimal and proportional to $N*log_2(N)$, where N is a power of two. Therefore, it is often useful to choose the data length equal to a power of two.

3.3 Endpoint Detection

The goal of endpoint detection is to isolate the word to be detected from the background noise. It is necessary to trim the word utterance to its tightest limits, in order to avoid errors in the modeling of subsequent utterances of the same word. As we can see from the upper part of Fig. 3, a threshold has been applied at both ends of the waveform. The front threshold is normalized to a value that all the spoken numbers trim to a maximum value. These values were obtained after observing the behavior of the waveform and noise in a particular environment. We can see the difference in frequency characteristics of the words *aik* (one), *teen* (three), *chaar* (four) and *paanch (*five) in Fig. 3, 4, 5 and 6, respectively.

3.4 Windowing

Speech signal analysis also involves application of a window with a time less than the complete signal. The window first starts with beginning of the signal and then shifted until it reaches the end. Each application of the window to the part of the speech signal results in a spectral vector.

3.5 Frame Blocking

Since the vocal tract moves mechanically slowly, speech can be assumed to be a random process with slowly varying

properties. Hence the speech is divided into overlapping frames of 100 ms. The speech signal is assumed to be stationary over each frame and this property will prove useful in further operations [5], [6], [8].

3.6 Fourier Transform

The MATLAB algorithm for the two dimensional FFT routine is as follows [9]:

```
fft2(x) =fft(fft(x), ');
```

Thus the two dimensional FFT is computed by first computing the FFT of x, that is, the FFT of each column of x, and then computing the FFT of each row of the result. Note that as the application of *fft2* command produced even symmetric data, we only show the lower half of the frequency spectrum in our graphs.

3.7 Correlation

Calculations for correlation coefficients of different speakers were performed [9]. As expected, the cross-correlation of the same speaker for the same word did come out to be 1. The correlation matrix of a spoken number was generated in a three-dimensional form for generating different simulations and graphs.

4. ANALYSIS & RESULTS

When we compared the frequency content of the same word by different speakers, we found striking similarities among them. This helped us get more confidence in our initial hypothesis that a single word uttered by a diverse set of speakers would exhibit similar characteristics. This phenomenon can be seen in Fig. 5, 6, 7 and 8. Additionally, Fig. 12 and 13 show surface graphs, and Fig. 14 shows a mesh plot for the correlation of frequency content among different speakers, for words *aik* (one) and *teen* (three).

We observed that Fourier descriptor feature was independent of the spoken numbers, with the combination of the Fourier transform and correlation technique commands used in MATLAB, a high accuracy recognition system can be realized. Recorded data was used in Simulink model for introductory analysis [10].

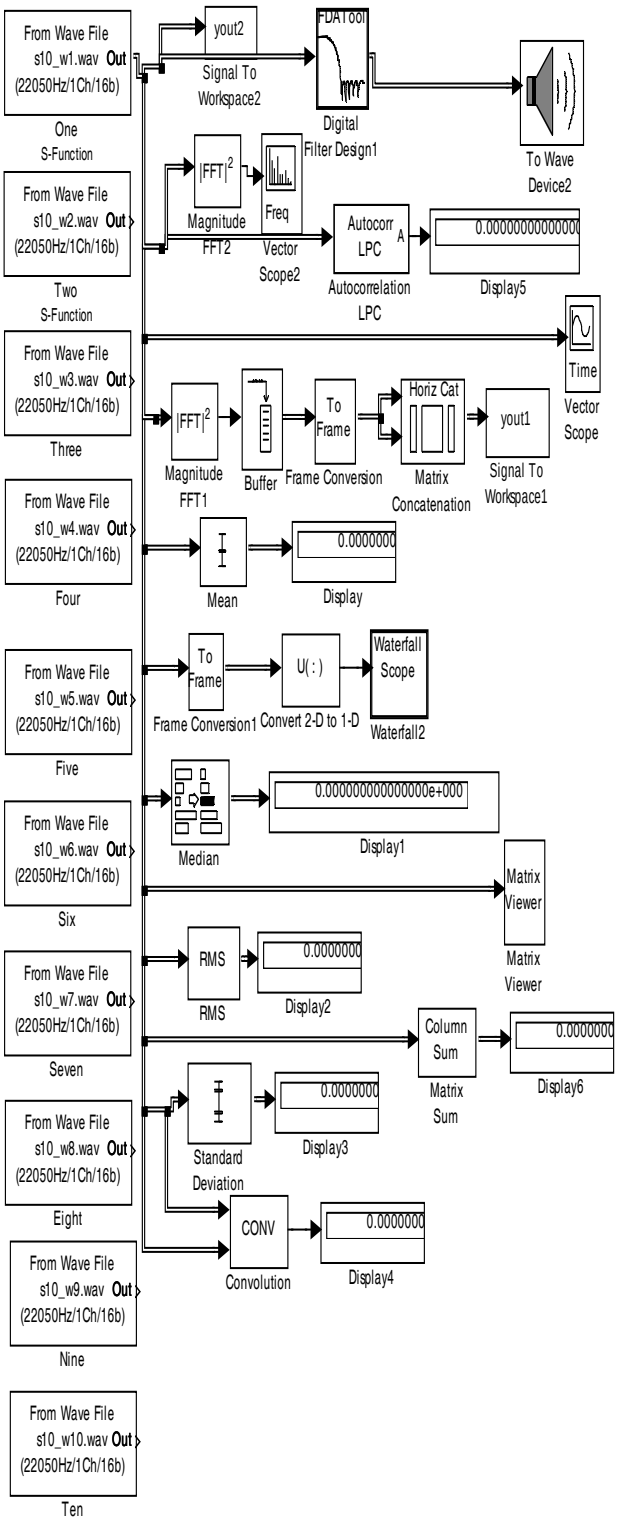

Fig. 2 Extended Simulink model for analysis of Urdu spoken numbers

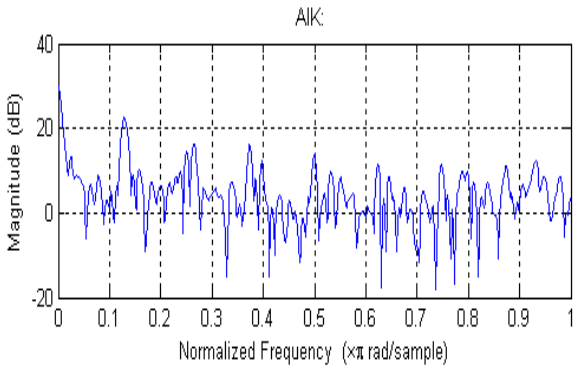

Fig. 3 The waveform of the correlation of the spoken Urdu numbers spoken *aik* (one)

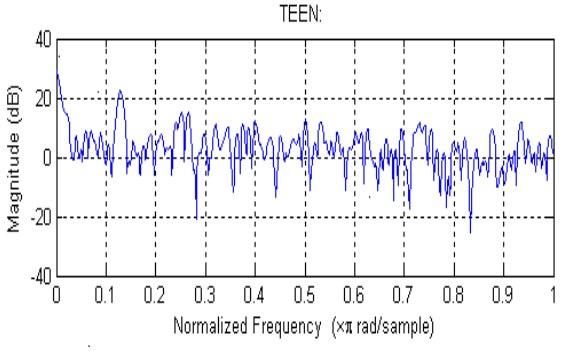

Fig. 4 The waveform of the correlation of the spoken Urdu number *teen* (three)

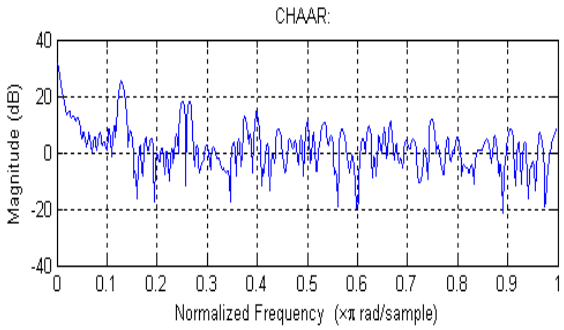

Fig. 5 The waveform of the correlation of the spoken Urdu number *chaar* (four)

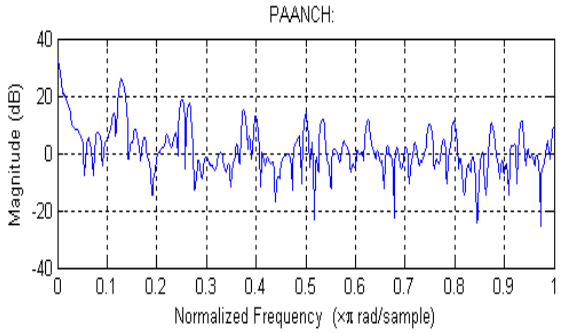

Fig. 6 The waveform of the correlation of the spoken Urdu number *paanch* (five)

5. CONCLUSION

In this paper, we presented frequency analysis of Urdu numbers (one to nine). The data was acquired in moderate noisy environment by word utterances of 15 different speakers. FFT algorithm was used in MATLAB to analyze the data. As expected, we found high correlation among frequency contents of the same word, when spoken by many different speakers.

We are currently investigating creation of neural network models for automatically recognizing individual Urdu words, numbers to be specific. This recognition system could be many potential applications, for example, voice-driven menu selection in a telephone-based customer service in Urdu/Hindi speaking countries such as Pakistan/India.

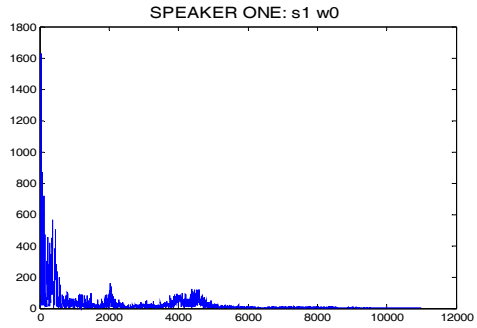

Fig. 7 The waveform of the correlation of the spoken Urdu number *sifr* (zero) by speaker-1

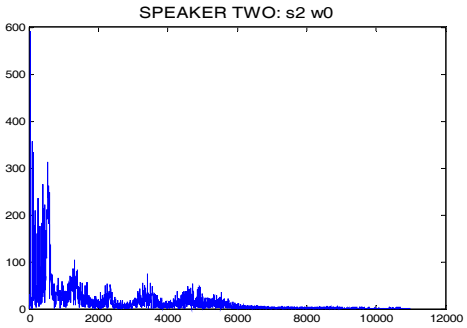

Fig. 8 The waveform of the correlation of the spoken Urdu number *sifr* (zero) by speaker-2

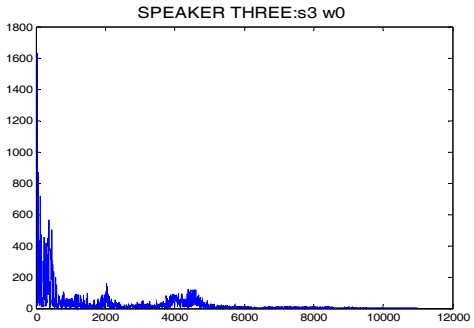

Fig. 9 The waveform of the correlation of the spoken Urdu number *sifr* (zero) by speaker-3

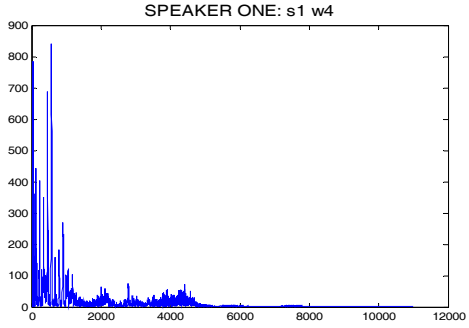

Fig. 10 The waveform of the correlation of the spoken Urdu number spoken *chaar* (four) by speaker-1

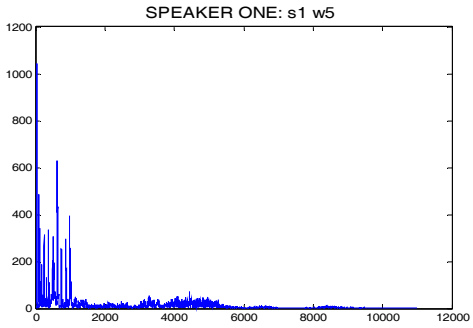

Fig. 11 The waveform of the correlation of the spoken Urdu number spoken *paanch* (five) by speaker-1

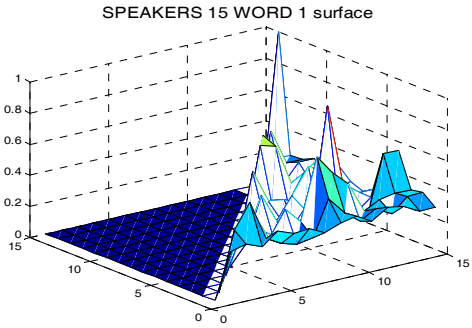

Fig. 12 The surface plot of the correlation of the spoken Urdu numbers spoken *aik* (one) by speaker-15

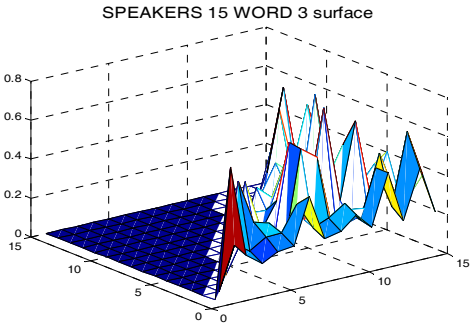

Fig. 13 The surface plot of the correlation of the spoken Urdu numbers spoken *teen* (three) by speaker-15

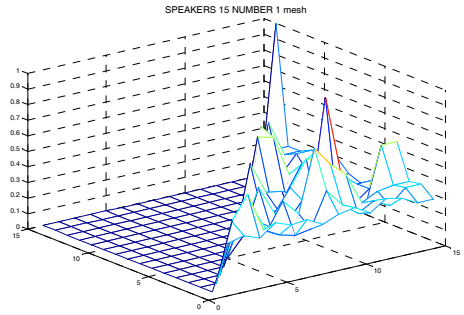

Fig. 14 The mesh plot of the correlation of the spoken Urdu number spoken *aik* (one) by speaker-15

6. REFERENCES

[1]. S K Hasnain, Nighat Jamil, "Implementation of Digital Signal Processing real time Concepts Using Code Composer Studio 3.1, TI DSK TMS 320C6713 and DSP Simulink Blocksets," IC-4 conference, Pakistan Navy Engineering College, Karachi, Nov. 2007

[2]. S K Hasnain, Pervez Akhter, "Digital Signal Processing, Theory and Worked Examples",January 2007.

[3]. M M El Choubassi, H E El Khoury, C E Jabra Alagha, J A Skaf, M A AL Alaoui, "Arabic Speech Recognition Using Recurrent Neural Networks," Symp. Signal Processing & Info. Tech., 2003, ISSPIT 2003, Dec. 2003, pp. 543- 547.

[4]. S K Hasnain, Aisha Tahir, "Digital Signal Processing Laboratory Workbook", 2006.

[5]. "MATLAB User's Guide," Mathworks Inc., 2006.

[6]. J Koolwaaij, "Speech Processing," //www.google.com/ search (current 5 May 2004).

[7]. M A Al-Alaoui, R Mouci, M M Mansour, R Ferzli, "*A Cloning Approach to Classifier Training*," IEEE Trans. Systems, Man and Cybernetics – Part A: Systems and Humans, vol. 32, no. 6, pp. 746-752, 2002.

[8]. "TMS320C6713 DSK User's Guide," Texas Instruments Inc., 2005.

[9]. D O'Shaughnessy, "Speech Communication: Human and Machine," Addison Wesley Publishing Co., 1987.

[10]. Samuel D Stearns, Ruth A David, "Signal Processing Algorithms in MATLAB," Prentice Hall, 1996.