LA-UR- 97-2001

Los Alamos National Laboratory is operated by the University of California for the United States Department of Energy under contract W-7405-ENG-36

TITLE: **ANALYSIS OF WOLVES & SHEEP - FINAL REPORT**

AUTHOR(S): John Hogden, George Papcun, Igor Zlokarnik, David Nix

SUBMITTED TO: Final Report to Sponsor

DISCLAIMER

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

MASTER

Los Alamos

Los Alamos National Laboratory
Los Alamos New Mexico 87545

Analysis of wolves & sheep -- Final Report

John Hogden, George Papcun, Igor Zlokarnik, & David Nix
CIC-3, MS B265
Los Alamos National Laboratory
Los Alamos, New Mexico

I. Executive Summary

In evaluating speaker verification systems, researchers have observed asymmetries in the ease with which people are able to break into other people's voice locks. For example, it might be easier for person 1 to break into the voice lock of person 2 than for person 2 to break into the voice lock for person 1. In such a case, person 1 (who is good at breaking into voice locks) is called a wolf, and person 2 (whose lock is easy to break into) is called a sheep. A third class of people is sometimes discussed: goats -- people who have a difficult time opening their own voice locks.

A priori, it seems odd that there might be wolf/sheep pairs. After all, if we find that speaker 1's voice is similar enough to speaker 2's voice to allow speaker 1 to break into speaker 2's lock, then we would also expect that speaker 2's voice is equally similar to person 1's voice, which should allow speaker 2 to break into speaker 1's lock. If we knew nothing about the voice lock algorithms, we might even wonder whether such asymmetries actually exist, or whether the asymmetries are merely the result of random chance. Unfortunately, as discussed in section II, it is relatively difficult to perform the kinds of experiments that would demonstrate that wolf/sheep asymmetries exist. In order to get around the problems brought up in section II, we do an analysis of the SpeakerKey algorithm in terms of an abstract "speaker space". The way the speaker space relates to the SpeakerKey algorithm is described in section III, then, in section IV, we describe how the speaker space can be used to understand at least one source of the wolf/sheep asymmetry. In section V we show that the speaker space can be inferred from interspeaker similarities using a statistical method called multidimensional scaling. While being able to infer the speaker space allows us to make predictions about which speakers are likely to be wolves and sheep, it is inconvenient and sometimes impossible to collect the kind of data that multidimensional scaling needs in order to infer the speaker space. So, in section VI we show how to infer a person's position in the speaker space using only a sample of the person's speech.

II. Statistical Tests for Wolf/Sheep Asymmetries

If there is no a priori reason to believe that wolf sheep asymmetries actually exist, it is important to show that observed asymmetries are not simply the result of random chance. To see how apparent asymmetries could arise by random chance, consider a case where person 1 actually has a 50% chance of breaking into person 2's lock and person 2 has a 50% chance of breaking into person 1's lock. In such a case, there is no wolf/sheep asymmetry, but an apparent asymmetry can arise. In this example, breaking into a lock is like flipping a coin and having it come up heads. So if we asked each person to try to break into the other's lock 1 time, we can see that there are four equally likely outcomes: person 1 could succeed and person 2 fail, person 2 could succeed and person 1 fail, both could succeed, or both could fail. Note that in 2 out of the 4 possible outcomes, one

person succeeds and the other person fails. Thus there is a 50% chance that there will be an apparent wolf/sheep asymmetry even though no asymmetry actually exists.

Suppose that, instead of having each person try to break into the other's lock 1 time, we had each person try N times and then count the number of successes each person has. To the extent that person 1 succeeds more often than person 2 (or vice versa) we would suspect that there is a real wolf/sheep asymmetry. In fact, if we subtract the number of times person 1 succeeds from the number of time person 2 succeeds and call the difference D, we can use probability theory to determine the likelihood of getting a certain D value if both people had the same probability of breaking into the other's lock. That is, we can find the probability of observing an apparent asymmetry when no real asymmetry exists. Thus, in the case where we have two people, each trying to break into the other's lock N times, we can quantitatively evaluate the likelihood that a real asymmetry exists.

This suggests a simple experiment in which we choose two people, have each try to break the other's lock several times, and then evaluate the likelihood that an asymmetry exists. It is even possible to use probability theory to determine that if we set N=150, we would have a very good chance of being able to find an asymmetry if one exists. However, this experiment could easily fail -- we could, by chance, pick a pair of speakers where neither person is a wolf or a sheep. We might have to repeat this experiment several times to find a wolf/sheep pair. Not only does this mean that we may need to collect a large set of testing data, this also leads to further statistical complications. Remember that given any set of data we are only able to find the probability of getting that data when there is no asymmetry. Suppose we choose N sufficiently large that the probability of getting a D greater than some criterion value is 0.01 if no asymmetry exists. Further suppose we test 100 pairs of people to determine whether any wolf/sheep pairs exist. Even if none of the pairs are wolf/sheep pairs, we would expect that one pair (0.01 x 100) would look like a wolf/sheep pair just by chance.

These considerations are intended to show why it is so difficult to demonstrate that wolf sheep pairs actually exist without taking advantage of our knowledge of the speaker verification algorithms. Because of these difficulties, our approach has been to study the speaker verification algorithms to gain insight into 1) whether there actually are wolf/sheep asymmetries; 2) if there are asymmetries, why?, and 3) who is likely to be a wolf and who is likely to be a sheep?

III. Speaker Space

Figure 1 shows a hypothetical example of what a speaker space might look like. Each letter in Figure 1 is intended to represent an average sample of a speaker's voice. The axes of our hypothetical speaker space correspond to qualities of people's voices, so people with similar voices lie next to each other in the speaker space. At this point, it is not essential to know which voice qualities are represented by the axes -- one can imagine them to be any parameters related to the size and shape of the speaker's vocal tract -- it is the distance between the voice samples that are important. For example, from this plot we can conclude that, on the average, speaker A's voice is more similar to speaker F's voice than to speaker G's voice. Notice that this speaker space is symmetric -- the similarity of A to G is the same as the similarity of G to A. Thus, with this speaker space we would not expect to see wolf/sheep pairs.

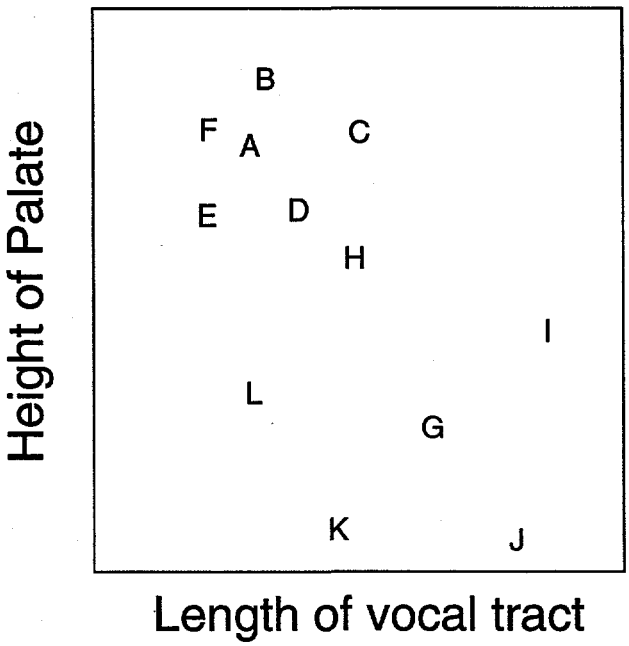


Figure 1. Hypothesized Speaker Space

Even though a speaker's vocal tract length and palate height don't vary, a speech sample does not give full information about the vocal tract length or palate height, so we shouldn't expect every speech sample produced by speaker A to show up in the exact same place in speaker space. Rather, we expect that each speaker will produce a distribution of samples. The area enclosed by the circle drawn around A in **Figure 2** is (hypothetically) the region enclosed by one standard deviation of the distribution of the samples that will be produced at different times by speaker A. The circle drawn around G shows the distribution of samples produced by speaker G.

As described by Higgins, Bahler, & Porter (Higgins, Bahler & Porter, 1991), when one person (the claimant) attempts to open a voice lock, the SpeakerKey system decides whether or not to accept a claimant based on three measures. We will only discuss two of those measures here because the third measure is intended to determine if the correct utterance was produced, not who produced it. The first measure is the forced recognition score (F). The forced recognition score is a measure of how similar the claimant's voice is to the lock's stored voice template. The second measure is the ratio set score (R). The ratio set score is a measure of the similarity between the claimant's voice and a conglomerative template made from the five speakers whose voices are most similar to that of the lock's stored template. Ideally, the claimant's voice should be much more similar to the lock's template than to the template of the lock's ratio set, so SpeakerKey is more likely to accept a claimant if F-R is large.

We can use the speaker space to help understand the relationship between the F and R scores. The distance between the speakers in the speaker space is intended to be analogous to the F score. Furthermore, since SpeakerKey uses the five subjects most similar to speaker A as the ratio set for A, we can determine that the speakers connected by the dashed line in **Figure 3** are the ratio set for speaker A, and the speakers connected by the solid lines are the ratio set for speaker G. Continuing this reasoning, we can conclude that the R score we would obtain if A attempted to access G's lock would be roughly the

distance between A and H, since H is the member of G's ratio set who is closest to A. Similarly, the R score that we would obtain if speaker G tried to access A's lock would be roughly the distance from G to D.

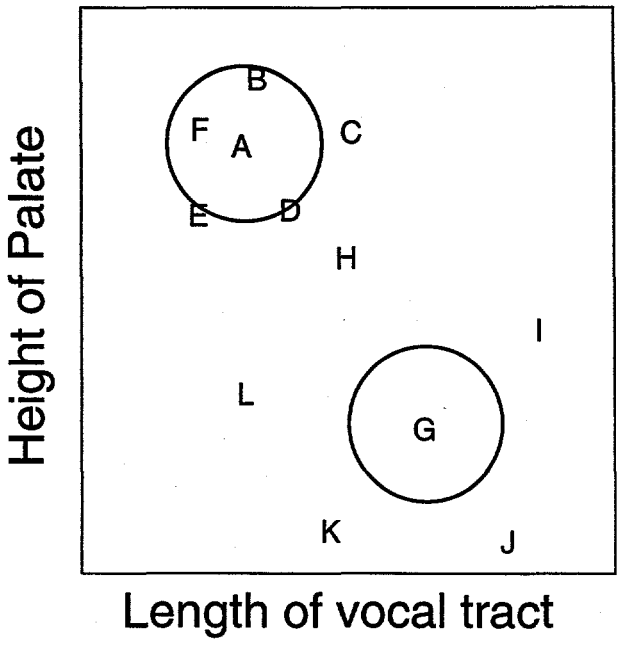


Figure 2. Speaker Sample Distributions

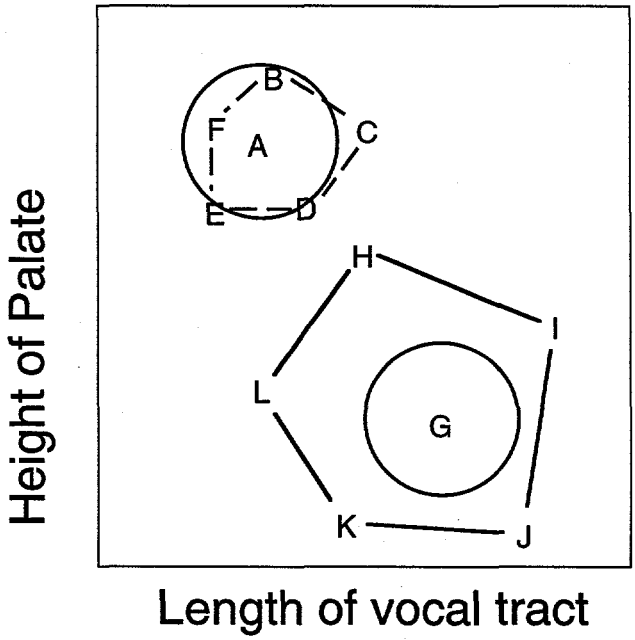


Figure 3. Expected Ratio Sets

IV. Wolves, Sheep, and Goats -- Asymmetries From a Symmetric Space

Notice that the R score obtained when A attempts to access G's lock (R_{AG}) is much smaller than the R score obtained when G tries to access A's lock (R_{GA}). This difference is potentially important because even if F_{AG} (the F score obtained when A tries to access G's lock) is the same as F_{GA}, the difference score F_{AG}- R_{AG} will typically be larger than F_{GA}- R_{GA}. Because of this difference, we should not expect it to be as easy for G to open A's lock as it is for A to open G's lock. Thus A would be a wolf with G as a sheep, but G would not be a wolf to A.

The speaker space also gives us insight about which speakers might be goats. From the relationship between the distribution of speech samples created by A and the ratio set of A, we can easily see that many of the samples created by A will be closer to a member of A's ratio set than to the average A utterance. Since the SpeakerKey system is more likely to accept a speech sample when F-R is large, we can conclude that A will be a goat. In contrast, G will not be a goat, even though the hypothetical standard deviations for A and G are identical.

V. Inferring the Speaker Space

The speaker space concept as described so far explains (at least in part) why we might see wolf/sheep asymmetries. But additional advantages could be obtained if we could determine where people lie in the speaker space. As described above, if we knew the positions of speakers in the speaker space we could make predictions about who would likely be a wolf and who would likely be a sheep, which can help us statistically demonstrate that a pair of speakers is a wolf/sheep pair. In this section we show how the speaker space can be inferred using multidimensional scaling (MDS).

In the discussion so far, we have assumed that the distances in the speaker space roughly correspond to F scores. We can actually construct a space in which that is the case, and thereby infer the speaker space. Multidimensional scaling is a statistical technique for using the distances between points to find the relative positions of points in a space (Dillon & Goldstein, 1984). Given the F scores obtained when each speaker tries to access every other speaker's lock, MDS represents the speakers in a space constructed such that the distances between the speakers are approximately the F scores. Ideally, MDS will place the points representing speakers in a low-dimensional space, making it possible to interpret what voice qualities are represented by the axes.

In order to infer the speaker space with MDS, we used a matrix of interspeaker distance scores supplied by the NSA. The interspeaker distance scores were the F scores obtained from SpeakerKey when using each of 105 speakers to break into each of the speaker locks. In fact, 9 sets of interspeaker distance scores were used to make MDS solutions. To use all 9 matrices, the matrices were averaged on an element by element basis, and the average matrix was used as input to an MDS algorithm.

Since MDS solutions can be made in any number of dimensions, we need to have a way to determine the correct number of dimensions to use. The most commonly used method for determining the number of dimensions to use is to evaluate how well the MDS solution fits the data as we increase the number of dimensions. Stress is a measure of the quality of the fit between the MDS solution and the data. As stress decreases, the MDS solution

improves. Since stress will always decrease as the number of dimensions increases, researchers typically look for a sharp "elbow" in the stress vs. dimensions plot, i.e. we would like to find a point at which increasing the dimensionality of the solution stops producing large increases in the accuracy of the solution.

Figure 4 shows the stress by dimension plot for MDS solutions with between 2 and 6 dimensions. As can be seen, there is approximately a 20% decrease in stress as we go from a 2 dimensional solution to a 3 dimensional solution, a 14% decrease going from 3 dimensions to 4 dimensions, and approximately a 10% decrease per dimension in going from 4 dimensions to 6 dimensions. While this plot does not have a sharp elbow, it does appear that there is relatively little to be gained by using more than 4 dimensions in the solution.

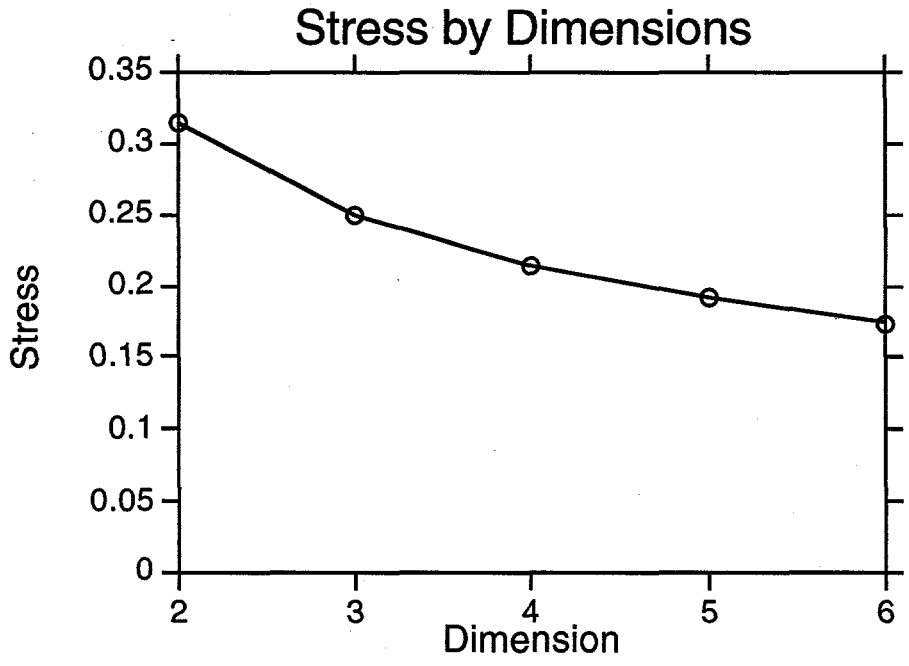


Figure 4. MDS Stress

Another aid to determining the number of dimensions to use is to evaluate the robustness of the dimensions obtained in solutions generated from different subsets of the data. After all, we should also be able to perform MDS analyses on two different subsets of data and get the same results. Clearly, if we don't get the same results (i.e. if the solutions are not robust), we should not trust the solutions. However, there are three different reasons why we might not get the same MDS solution from 2 different subsets of the data.

1. MDS occasionally gives a "degenerate solution", which is not a good representation of the dissimilarity data. Such degenerate solutions are relatively easy to detect because they show an abnormally small stress value and the solution is generally a simplex. Since degenerate solutions are easy to detect

and were not encountered with this data set, we will not discuss these solutions further.

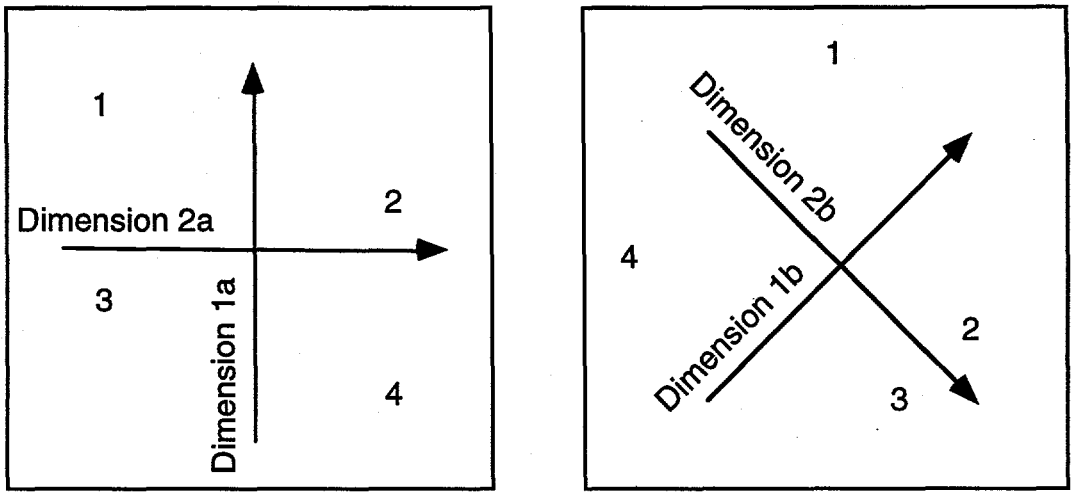
2. MDS, like all analyses that require using minimization techniques, occasionally finds a local minimum in the error space that is not as good as some other global minimum. Local minima can usually be avoided by running the MDS algorithm from different initial configurations and choosing the solution that has the lowest stress. However, it is possible for two very different solutions to have nearly the same stress value, in which case the soution should not be trusted.

3. The data in one subset may be different from the data in a different subset, suggesting that more data should be collected in each subset so that the subsets give sufficient information to let us generalize.

To evaluate the robustness of the solutions, we broke the data into 2 subsets by randomly selecting 4 of the interspeaker distances matrices to put in the first subset, and then randomly choosing 4 different interspeaker difference matrices to put in the second subset, with one matrix left unused. The four matrices in subset one were averaged to get a single matrix to use for the MDS analysis. Similarly, the four matrices in subset two were also averaged and used as input to an MDS analysis.

MDS solutions having between 2 and 6 dimensions were obtained for each of the average matrices. Canonical correlation was used to evaluate the similarity of the solutions obtained from different subsets of the data. Canonical correlation first finds a dimension in both solutions such that the positions of the points projected onto the dimensions are maximally correlated between solutions. Then the next most highly correlated dimensions are found, etc. **Figure 5** shows an example of a canonical correlation analysis of two hypothetical MDS solutions. Both of the hypothetical MDS solutions have 4 points, as opposed to the 105 points (one point for each speaker) in the MDS solutions for the SpeakerKey interspeaker distances. Recall that the 4 points represent speakers and the distances between the points tell us about the F scores that would be generated by SpeakerKey. The axes drawn in the solutions (labeled "Dimension 1a" etc.) show the canonical dimensions that would be found by canonical correlation for these configurations of points. Notice that the positions of the solution 1 points on dimension 1a have a 100% correlation with the positions of solution 2 points on dimension 1b. In contrast, there is a 0% correlation between the solution 1 points projected onto dimension 2a and the solution 2 points projected onto dimension 2b. Dimension 1 is robust since we find the same dimension in both solutions, but dimension 2 is not at all robust between the solutions. In this example, we would conclude that canonical dimension 1 contains real information but that dimension 2 is unlikely to be useful.

Canonical Correlation Example



Solution 1 Solution 2

Figure 5

The results of canonical correlation analysis for two, three, four, and six-dimensional solutions are shown in **Figure 6**. The results of comparing the five-dimensional solutions are similar but were omitted to make the figure clearer. Figure 6 is complex and requires some explanation. The triangles represent correlations for canonical dimensions found when comparing the two-dimensional solution obtained from data subset 1 to the two-dimensional solution obtained for data subset 2. As can be seen, the first dimension of the two-dimensional solution is robust -- the correlations between the solution 1 positions of the point projected on dimension 1 and the solution 2 positions of the points projected onto dimension 1 is about 98%. Dimension 2 of the two-dimensional solution is not very robust; it has a correlation of about 34%. Strikingly, all of the dimensions of the solutions with three or more dimensions are fairly robust.

When we compare the stress values (recall that stress is larger for worse MDS solutions) obtained for the 2 dimensional solutions, we find equal stress values for solutions obtained in both subsets (.318 for both). From the results described so far, it is impossible to know whether the data in the two subsets is just sufficiently different that different solutions are best for each subset, or whether the data are roughly equivalent but there are multiple equally good solutions. Further MDS runs reveal that different solutions can be found even using a single data set. Depending on the values used to start the MDS minimization algorithm, we can find solutions for data subset 1 that are very different, despite having stress values that only differ by 0.001. Since these solutions were not the result of the minimization algorithm reaching the maximum number of iterations, or the minimization algorithm stopping because some minimum stress value was reached, this result suggests that more than one good 2 dimensional solution exists for the data.

While researchers typically prefer MDS solutions with fewer dimensions, it is clear that the two-dimensional solution should not be chosen due to it's lack of robustness. In contrast,

the three-dimensional solution was robust and accounted for 59% of the variance of the interspeaker distances, despite the fact that such data must be fairly noisy. Thus, a three dimensional solution is probably sufficient for this problem.

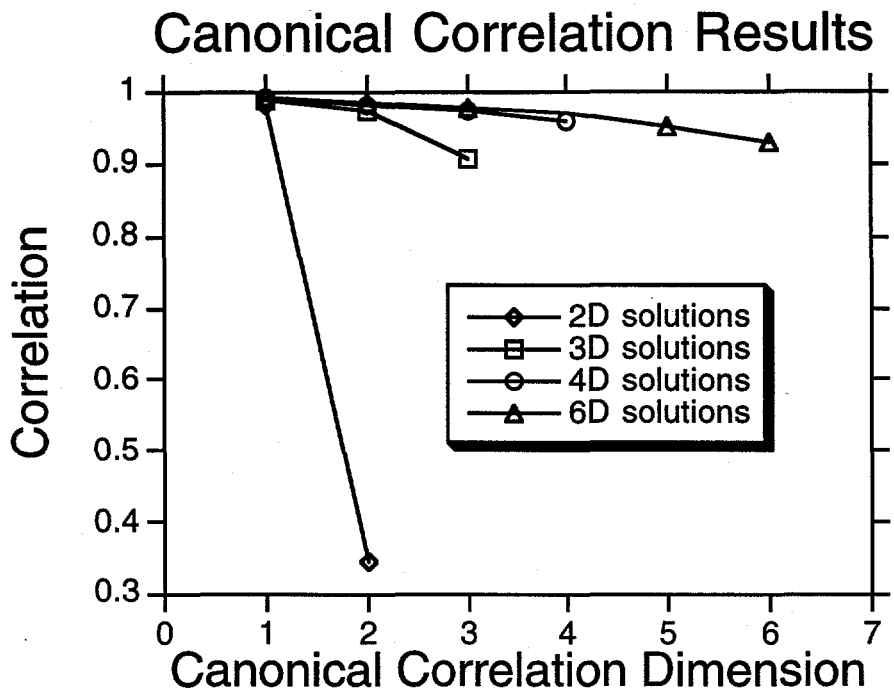


Figure 6

VII. Mapping from a speaker's voice to the position in speaker space

The speaker space idea could prove even more useful if we had a way to map from an unenrolled speaker's voice directly into the speaker space. This is not possible using the MDS method discussed above, in which we must first enroll the speaker and then have the speaker try to break into a variety of other locks. Clearly, the MDS method would be difficult to use with an uncooperative speaker. In this section, we show that a neural network is capable of performing this task.

A neural network was created that used processed speech samples (10 LPC coefficients and 10 cepstrum coefficients for each 10ms window of speech) as inputs and the position of the person in speaker space as the output. Ideally, the neural network should take in each individual window of a speech sample from a given speaker and output the speaker's position in speaker space. To be more concrete, call the ith window from the jth speaker w(i,j), and the position of the jth speaker in the speaker space p(j). If the neural network performed perfectly, then for speaker 17 it would take in w(1,17) as the input and output p(17), then take w(2,17) and get p(17), etc., until w(100,17) is reached, which should also give the output p(17).

Of course the neural network did not perform perfectly, so when given window (1,17) the network gives an approximation of p(17). To get a single position in the speaker space for

each speaker (instead of a position for each window), we averaged the estimates of a speaker's position in speaker space.

Neural networks can be trained to perform arbitrarily well on one data set but then may not generalize well to different data sets. Thus, neural network performance is typically determined by training on a subset of the data and testing how well the neural network performs on a different subset of the data. We made a neural network training set from a subset of 83 speakers enrolled in the SpeakerKey systems, and made a testing set composed of 20 different speakers from among those enrolled. Note that only 103 out of the 105 speakers were used in the training and testing sets. Speech samples from two speakers were omitted because they spoke the incorrect SpeakerKey combination. Since all the other speakers spoke the same combination, and therefore the neural network was trained on the same utterance for each of the other speakers, we believed it would only add noise to include the two invalid utterances.

Canonical correlation was used to determine how well the speaker space positions estimated by the neural network matched the speaker space positions determined by MDS. The correlations between the canonical dimensions of the solutions were 95%, 87%, and 59%. The relatively high correlations (particularly for the first two dimensions) show that it is possible to get a good estimate of a speaker's position in speaker space, and therefore get information about whose locks are vulnerable to attack by the speaker.

VIII. Conclusion

Starting from the observation that a tremendous amount of data would be required to statistically demonstrate the existence of wolf/sheep pairs we concluded that analyses of speaker verification algorithms should be used to help us understand wolf/sheep asymmetries. Using the notion of a "speaker space", we demonstrated that such asymmetries could arise even though the similarity of voice 1 to voice 2 is the same as the similarity of voice 2 to voice 1. This provides a partial explanation of wolf/sheep asymmetries but not a complete explanation. There may well be other factors that contribute to the asymmetry but which have not yet been evaluated.

In addition, we demonstrated that the speaker space can be computed from interspeaker similarity data using multidimensional scaling, and that such a speaker space can be used to give a good approximation of the interspeaker similarities. The derived speaker space can be used to predict which of the enrolled speakers are likely to be wolves and which are likely to be sheep. This work lays the foundation for further work that will experimentally determine the accuracy of the predictions about who is a wolf and who is a sheep.

However, since the speaker space derived using multidimensional scaling was created from interspeaker similarity data, a speaker must first enroll in the speaker key system, and then be compared to each of the other speakers (or to an adequately large subset of speakers) to place the person in the speaker space. This would be inconvenient in general, and impossible in cases where the person we want to place in the speaker space is uncooperative. To get around these difficulties, we also demonstrated that a good estimate of a person's speaker space position could be obtained using only a speech sample.

References

Dillon, W., & Goldstein, M. (1984). Multivariate Analysis: Methods and Applications. New York: John Wiley & Sons.

Hays, W. (1981). Statistics. (3rd ed.). New York: Holt, Rinehart, and Winston.

Higgins, A., Bahler, L., & Porter, J. (1991). Speaker verification using randomized phrase prompting. Digital Signal Processing, 1, 89-106.