DETERMINING STOCHASTIC DEPENDENCE FOR NORMALLY DISTRIBUTED VECTORS USING THE CHI-SQUARED METRIC

R.S. Valiveti and B.J. Oommen

SCS-TR-178, JULY 1990

School of Computer Science, Carleton University
Ottawa, Canada, KIS 5B6

Determining Stochastic Dependence for Normally Distributed Vectors Using the Chi-squared Metric

R.S. Valiveti and B.J. Oommen *
School Of Computer Science
Carleton University
Ottawa, Canada, K1S 5B6

Abstract

A fundamental problem in information theory involves computing and estimating the probability density function associated with a set of random variables. In estimating this density function, one can either assume that the form of the density function is known, and that we are merely estimating parameters that characterize the distribution or that no information about the density function is available. This problem has been extensively studied if the random variables are independent.

If the random variables are dependent and are of the discrete sort, the problem of capturing this dependence between variables has been studied in [2]. The analogous problem for normally distributed continuous random variables has been tackled in [3]. In both these instances, the determination of the best dependence tree hinges on the well-acclaimed Expected Mutual Information Measure (EMIM) Metric. Valiveti and Oommen studied the suitability of the chi-squared based metric in-lieu of the EMIM metric, for the discrete variable case[10].

In this paper, we study the use of the chi-squared metric for determining dependence trees for normally distributed random vectors. We show that for such vectors, the chi-squared metric yields the optimal tree and that it is identical to the one obtained using the EMIM metric. The computational gain obtained using the chi-squared metric is discussed.

Keywords:

Statistical Information, Probability distribution, Estimation, Approximation, Closeness of Approximation, Normal Distribution, Dependence trees.

*Both authors partially supported by Natural Sciences and Engineering Research Council Of Canada.

1 Introduction

Many information theoretic applications require the extraction of statistical information from the problem domain. The input to the data analysis procedures consists of observations of features that characterize the sample space. In this context, probably the most frequent scenario is that of computing and estimating the underlying probability distribution of a set of random variables. Once this distribution has been obtained, it can be used in host of applications such as pattern recognition, system identification and information retrieval [4].

Let $\mathbf{X} = [x_1, x_2, \ldots, x_N]^T$ be an n-dimensional random vector. The components of the vector $\mathbf{X}$, i.e. $x_1, x_2, \ldots, x_N$, are the random variables in question and will be synonymously referred to as the features of the problem space. The basic problem that we wish to address is one of estimating the joint probability density $P(\mathbf{X})$, from measurements on the vector itself. Due to the fact that the number of samples may be small, and that real machines often impose limitations on available storage, it is often necessary to restrict the amount of information that can be collected from the samples. Keeping this constraint in mind, it is common to make various simplifying assumptions about the distribution $P(\mathbf{X})$. The simplest assumption made about this distribution is that the random variables or features are statistically independent. Obviously, this assumption does not necessarily model the underlying real life situation adequately.

This assumption, namely that regarding stochastic independence, typically simplifies the solution that is obtained. In the interest of brevity, we shall clarify this for the case when the random vector is jointly normal. Let us consider a pattern recognition scheme, and focus on the aspect of analyzing the features of only one class. Since the features may be completely dependent, in general, the covariance matrix Σ may be full. However, by redefining the features so that the feature vector is a linear combination of the original vector, it is easy to see that the transformed set of features can be made statistically independent [5, pp.31-35]. Indeed, the linear transformation essentially involves pre-multiplying the original feature vector by the matrix of eigenvectors of Σ, and since Σ is symmetric and positive definite, these eigenvectors will be orthonormal and the resulting covariance matrix of the transformed feature vector will be diagonal.

The question is a little more complicated when we are studying the features of two classes, say ω_1 and ω_2. Let us suppose that the class-conditional densities of these classes have covariance matrices Σ_1 and Σ_2. Using a transformation analogous to the one described above, it is easy to diagonalize Σ_1 (i.e. make the features of class ω_1 independent) while the features of the class ω_2 are still dependent. Let us denote the transformed covariance matrices as Σ_1' and Σ_2' respectively. By appropriately scaling the transformed features of the class ω_1, is possible to render *it* to have the identity matrix as its covariance matrix. If the new covariance matrices are Σ_1'' and Σ_2'', then, this operation renders $\Sigma_1'' = I$. Subsequently the axes of the features can be "rotated" so as to be parallel to the eigenvectors of the covariance matrix Σ_2''. Thus even in the two class problem, after the appropriate rotation and scaling operations, we can always assume that we are working with independent features. This process is typically referred to as *whitening* and since this is not the primary topic of this paper, we refer the reader to Fukunaga's book for an excellent treatise on the topic [5, pp.31-35].

When the number of classes is more than two, the question of dependence is extremely pertinent. Notice now that although the vectors can be transformed to permit a whitening process between any two classes, such a scheme will not be applicable for the rest of the classes. Thus the Bayes' classifier will have to resort to computing the Mahalanobis' distance in the original feature space.

The question is now one of modeling the dependence between features. In the strictest sense, it is possible that every feature is dependent on every other one. Thus, in general, both the covariance matrix Σ and its inverse Σ^{-1} are full. However if the dependence is of the tree-type (i.e. x_i is dependent on x_j only if x_j is the parent of x_i) it is interesting to note that although Σ is full, Σ^{-1} is sparse and fully captures the dependence information[3]. Since it is *this* quantity which is required in computing the Mahalanobis' distance, a significant gain in computing can be achieved in all such cases by modeling the dependence as a tree-type dependence.

In this paper, we shall consider the problem of studying the dependence of an arbitrary random vector. To pose the problem in all its generality, we will not pay attention to the number of classes, but merely concentrate on a single random vector $\mathbf{X} = [x_1, x_2, \ldots, x_N]^T$ whose covariance matrix is Σ. In the first case we assume that there is a tree-type dependence between the components of $\mathbf{X}$. We shall study the problem of estimating this dependence tree and show that the estimate of the dependence tree converges to the true underlying (unknown) tree. However, if the underlying dependence is not of the tree type, our endeavor is to obtain the best tree-type dependence which adequately models Σ.

At this juncture, it is appropriate to query the word "best". Indeed, by this we mean that we are investigating how one can quantify the "goodness" of any tree. In this context, the most straightforward metric is the well-known Information Theoretic comparison between the "real" distribution and the distribution based on a given tree dependence. The computation of the best dependence tree naturally leads to the Expected Mutual Information Measure (EMIM) metric for capturing the dependence between *pairs* of variables. Although this metric is exact, it is difficult to compute, because of the logarithms, exponentials, and ratios involved.

In contrast to this, we advocate the use of an alternate metric, the chi-squared metric, to represent dependence information between pairs of random variables. The latter involves no logarithms but is merely computed using sum of squares. In the case of discrete valued features, in an earlier paper [10] we showed analytically that for various families of distributions, the chi-squared metric **always** yields the optimal tree. In other cases, the experimentally obtained accuracy of it yielding the optimal tree was 100%. However, in this paper we shall show that for the case of continuous normal vectors, the result is much more powerful. Amazingly enough, we show that the chi-squared metric **always** yields the optimal tree, implying that the entire process of computing and manipulating the EMIM metric can be short-circuited.

Apart from the above results, we shall also prove results that involve the estimate of Σ (denoted as $\hat{\Sigma}$) as opposed to Σ itself. We shall show that the tree obtained by using the estimate $\hat{\Sigma}$ is the Maximum Likelihood Estimate (MLE) of the dependence tree, chosen from among all the possible trees, and that this estimate converges to the "real" unknown underlying tree as the number of samples increases. We note in passing, that in typical cases, this estimate converges to the true underlying tree when the number of samples is even as small as 50.

2 Approximating Discrete Distributions

In this section, we shall briefly discuss the problem of representing the information regarding the stochastic dependence between the random variables when they are discrete-valued. Clearly, this constitutes a large number of cases in the field of traditional pattern recognition, and indeed includes the entire field of problems encountered in information retrieval. But in the context of this paper, this section is included to present the problem in the right perspective and to motivate the obvious generalizations.

The central issue in tackling this problem is one of approximating the joint distribution/density $P(\mathbf{X})$, where $\mathbf{X}$ is the vector $[x_1, x_2, \ldots, x_N]^T$. Because of the largeness of the dimensionality of the vector $\mathbf{X}$, it is both infeasible and impractical to store estimates of the joint density function $P(\mathbf{X})$ for all possible values of $\mathbf{X}$. It is infeasible because, if each feature x_i could take one of a set of k distinct values, the number of estimates that would have to be maintained is of the order of k^N. Obviously such a scheme would be impractical also. A second alternative is one of approximating $P(\mathbf{X})$ by a well defined and easily computable density function $P_a(\mathbf{X})$.

When using an approximation density function $P_a(\mathbf{X})$, the question is now one of measuring how well this function approximates the "real" density function $P(\mathbf{X})$. In order to quantify this "goodness" of approximation, we must rely on a distance measure, between these two density functions. The most prominent one is the an information theoretic measure referred to earlier, and is given by:

$$I(P, P_a) = \sum_{\mathbf{X}} P(\mathbf{X}) \log \frac{P(\mathbf{X})}{P_a(\mathbf{X})} \tag{1}$$

It is well known that $I(P, P_a) \geq 0$ and $I(P, P_a) = 0$ only if the approximation is exact (i.e. $P(\mathbf{X}) = P_a(\mathbf{X})$, for all $\mathbf{X}$). Hence $I(P, P_a)$ is a measure of the closeness of the approximation; the smaller this measure, the better the approximation. In other words, the intent in finding the best approximation for a given density function $P(\mathbf{X})$ is to find a density function $P_a(\mathbf{X})$ from the set of available approximations such that $I(P, P_a)$ is minimized.

A discussion of the techniques for approximating discrete distributions appears in [7]. Since our present focus is based on the approximation technique based on dependence trees, it is appropriate to summarize the method due to Chow and Liu, in all brevity [2].

2.1 Dependence Tree Approximation

To introduce the method due to Chow *et. al.* [2], we use the following well established rule, *the Chain Rule*, which expresses the joint probability distribution in terms of conditional probabilities:

$$P(X) = Pr(x_1)Pr(x_2 \mid x_1)Pr(x_3 \mid x_1, x_2) \cdots Pr(x_N \mid x_1 \ldots x_{N-1}) \tag{2}$$

We notice that in this expression, each variable is conditioned on an increasing number of other variables. In this context, it is important to point out that estimating the k^{th} term of this equation (i.e. a conditional probability term, in which a variable is conditioned on $k-1$ other variables), requires maintaining the estimates of all the k^{th} order marginals. If we restrict ourselves to computing only the (first and) second-order marginals, we are assured that we can compute all the conditional probabilities of the form $Pr(x_i \mid x_j)$. Thus we explore the approximation that results if we ignore the conditioning on multiple variables, implying that we shall attempt to retain only dependencies on at most one variable. This leads us to the following approximation:

$$P_a(X) = \prod_{i=1}^{N} Pr(x_i \mid x_{j(i)}) \qquad \text{where} \qquad 0 \leq j(i) < i. \tag{3}$$

Notice that in this equation, each variable is dependent on exactly one variable that has appeared earlier in the equation. To include the case when $j(i) = 0$, we use the convention that x_0 is a dummy variable, which does not influence any other variable. With this understanding, $Pr(x_i \mid x_0)$ will be the same as $Pr(x_i)$.

In the above case, there is a "natural ordering" among the components $\{x_i \mid i = 1, 2, \ldots, N\}$ of the vector $\mathbf{X}$, such that the variable x_i was conditioned on $x_{j(i)}$, where $j(i) < i$. In the general case, the product approximation takes the form:

$$P_a(X) = \prod_{i=1}^{N} Pr(x_{m_i} \mid x_{m_{j(i)}}) \tag{4}$$

where, $m_1, m_2, \ldots, m_N$ is a permutation of the integers $\{1, 2, \ldots, N\}$.

The dependence assumed in the above equation can be given a graph theoretical interpretation. Consider a graph G, with N nodes, labelled as $\{x_1, x_2, \ldots, x_N\}$. In this graph, the edge $(x_{m_i}, x_{m_{j(i)}})$, represents the fact the variable x_{m_i} is (statistically) dependent on the variable $x_{m_{j(i)}}$. It is easy to see that G is indeed a tree, with the node x_{m_1} as the root. This tree is completely defined by the permutation $\langle m_1, m_2, \ldots m_N \rangle$ and the function $j(\cdot)$.

It is well known that there are $N^{(N-2)}$ spanning trees on a graph with N nodes. Also, each tree is associated with a unique approximation of the form given in (4). The problem of finding the "dependence tree" is one of finding the tree, for which the associated approximation is the best.

It is proved in [2] that the closeness measure, $I(P, P_a)$, computed for the product approximation in (4) can be expressed as:-

$$I(P, P_a) = -\sum_{i=1}^{N} I^*(x_{m_i}, x_{m_{j(i)}}) + \sum_{i=1}^{N} H(x_i) - H(\mathbf{P}) \tag{5}$$

where,

$$\begin{aligned} H(x_i) &= -\sum_{x_i} Pr(x_i) \log Pr(x_i) \\ H(\mathbf{P}) &= -\sum_{\mathbf{X}} P(\mathbf{X}) \log P(\mathbf{X}) \end{aligned}$$

and

$$I^*(x_i, x_j) = \sum_{x_i, x_j} Pr(x_i, x_j) \log \frac{Pr(x_i, x_j)}{Pr(x_i) Pr(x_j)} \tag{6}$$

The problem of finding the dependence tree, is one of finding the permutation $\langle m_1, m_2, \ldots, m_N \rangle$ and the function $j(\cdot)$ (or equivalently an array $(j_1, j_2, \ldots j_N)$) which will minimize the right-hand side of (5). Clearly $\sum_{i=1}^{N} H(x_i)$ and H(P) are independent of the approximation since they only depend on the underlying distribution $P(\mathbf{X})$. Therefore, in order to minimize the RHS of (5), we must maximize $\sum_{i=1}^{N} I^*(x_{m_i}, x_{m_{j(i)}})$.

It can be seen that this minimization process is essentially the same as the operation of finding the maximum spanning tree (MST) of the graph G, with N nodes, $\{x_1, x_2, \ldots, x_N\}$, where the edge between nodes x_i, x_j is assigned the weight $I^*(x_i, x_j)$. In practice, the probabilities required for computing the edge weights of the graph are not known *a priori*, and their estimates must be used instead of their "real" values. In [2] it was proved that the estimate of the dependence tree so obtained, is also the MLE of the dependence tree. This is by no means an obvious result, and it is no small task to generalize this for the case involving a vector of continuous random variables.

In the light of the above result, it is clear that the metric defined in (6) is the best possible metric for capturing dependence information between *pairs* of variables, if the overall approximation is to be ideal in the sense of obtaining the minimum value of $I(P, P_a)$, as defined in (1). The only drawback with the I^* measure is that it is computationally very expensive and time consuming (especially for large values of N), as it involves the evaluation of $O(N^2k^2)$ logarithms where, as before, k is the number of values which each feature of the sample can take. To alleviate this problem, a new metric was earlier proposed [10], called the I_χ metric, which essentially captures the χ^2-statistic of the distributions and which can be used as a measure of dependence between the variables. Applications of approximations based on dependence trees can be found in [4, 12].

2.2 The χ^2 statistic

The metric $I_\chi(x_i, x_j)$ to quantify the degree of dependence between two discrete (random) variables is defined as follows [10]:-

$$I_\chi(x_i, x_j) = \sum_{x_i, x_j} \frac{\{Pr(x_i, x_j) - Pr(x_i)Pr(x_j)\}^2}{Pr(x_i)Pr(x_j)} \tag{7}$$

From the definition in (7), it is clear that I_χ has the following desirable characteristics of a metric capturing dependency information:

1. $I_\chi(x_i, x_j) \geq 0$

2. $I_\chi(x_i, x_j) = 0$ iff $Pr(x_i, x_j) = Pr(x_i)Pr(x_j)$.

Observe that the latter equation represents the case when the variables x_i, x_j are *statistically independent.*

At this juncture, it is worth recapitulating that Section 2.1 concluded with the very important result due to Chow *et. al.* which is that the I^* metric defined in (6) naturally appears in the process of selecting the MLE of the dependence tree. This is true, since we are constrained by the laws of probabilities, to choose a product form approximation for the joint distribution $P(\mathbf{X})$, and the only metric which minimizes the closeness measure defined in (1) is the EMIM given in (6).

In this light, it is clear that the metric defined in (7), cannot be related to the closeness of approximation defined by (1). Although $I_\chi(x_i, x_j)$ cannot be related to $I(P, P_a)$ defined in (1), it is not entirely futile to pursue it. Observe that for the first part, it is far easier to compute it rather than to compute I^* because the latter involves the evaluation of numerous logarithms. Furthermore, this metric possesses the following properties [10]:

(i) In the case when the random variables x_i, x_j are binary valued, the metric $I_\chi(x_i, x_j)$ defined in (7), increases or decreases monotonically with $I^*(x_i, x_j)$.

(ii) For a restricted class of probability distributions, i.e. for a specific type of dependence between the individual random variables, the quantities $I_\chi(x_i, x_j)$ and $I^*(x_i, x_j)$ are shown to be equivalent. Thus within this family, the I_χ metric indeed yields the MLE of the underlying tree, even though the computation is achieved without computing the matrix $I^*(\cdot, \cdot)$.

Notice that in general, the I_χ metric yields marginally sub-optimal dependence trees. As reported in [10], the computational gain in using the I_χ metric is significant — since a good approximation to the optimal dependence tree can be found in 20-25% of the time required to computed the latter. Also, if the underlying dependence is tree-type, the experimentally observed accuracy of the I_χ metric yielding the optimal tree was 100%, thereby proving that it is an efficient metric – both theoretically and experimentally.

We shall now study the effectiveness of chi-squared metric for the case of continuous random variable case.

3 Approximating Continuous Distributions

In the case when the vector $\mathbf{X} = [x_1, x_2, \ldots, x_N]^T$ is composed of random variables x_i, which are all continuous random variables. $P(\mathbf{X})$ denotes the joint density function of the random vector $\mathbf{X}$. This joint distribution is said to be of *tree dependence* if $P(\mathbf{X})$ can be written in the form:

$$P(\mathbf{X}) = \prod_{i=1}^{N} P(x_{m_i} | x_{m_{j(i)}}) \tag{8}$$

where $0 \leq j(i) < i$ and $\langle m_1, m_2, \ldots, m_N \rangle$ is a permutation of the integers $\{1, 2, \ldots, N\}$ and $m_0 = 0$.

Observe that in the RHS of (8), the term $P(x_{m_i} | x_{m_{j(i)}})$ represents the conditional density of the random variable x_{m_i}, conditioned on the random variable $x_{m_{j(i)}}$. Equation (8) can be seen to be a straightforward extension of (4) for the discrete vector case, obtained by substituting probability density functions in place of absolute probabilities. Observe too, that, in the dependence tree τ, corresponding to the approximation (8), there is an edge between the nodes x_{m_i} and $x_{m_{j(i)}}$. (8) can be easily rewritten in the symmetric form as follows:

$$\begin{aligned} P(\mathbf{X}) &= \prod_{i=1}^{N} \frac{P(x_{m_i}, x_{m_{j(i)}})}{P(x_{m_i})P(x_{m_{j(i)}})} \prod_{i=1}^{N} P(x_{m_i}) \\ &= \prod_{[i,j] \in \tau} \frac{P(x_i, x_j)}{P(x_i)P(x_j)} \prod_{i=1}^{N} P(x_i) \quad \{\text{since } \langle m_1, m_2, \ldots, m_N \rangle \text{ is a permutation}\} \end{aligned} \tag{9}$$

Since the notion of tree dependence in continuous distributions has been established, we now proceed to determine the best dependence tree [1] for a given continuous distribution.

3.1 Optimum Dependence Tree

In selecting the best approximating density function, we use as the measure of closeness of approximation, the natural extension to the information theoretic measure defined in (1). The metric for closeness between the "real" density function $P(\mathbf{X})$ and the approximating density $P_a(\mathbf{X})$ is quite simply given by:

$$I(P, P_a) = \int P(\mathbf{X}) \log \frac{P(\mathbf{X})}{P_a(\mathbf{X})} d\mathbf{X}. \tag{10}$$

Note that the above integral is an $N-$fold *vector* integral with respect to the variables $x_1, x_2, \ldots, x_N$. Again, as in the discrete case, it is well known that $I(P, P_a) \geq 0$ with the equality holding only if $P(\mathbf{X}) = P_a(\mathbf{X})$ almost everywhere.

[1] If the dependence tree is of the tree-type, the best dependence tree will be the underlying tree — and this is merely a consequence of the *metric* properties of the closeness measure. In all other cases, the best dependence tree τ is the one which minimizes the closeness measure between the "real" density function $P(\mathbf{X})$ and the density $P_\tau(\mathbf{X})$ based on the tree dependence model with the dependence tree being τ.

In the case when the tree dependence is being used as model for approximation, the probability density derived from the dependence tree replaces the term $P_a(\mathbf{X})$ in (10). If the dependence tree in question is denoted by τ, the following equation gives the measure of closeness for the approximate density function derived from τ[3]:

$$I(P, P_\tau) = K - \sum_{[i,j]\in\tau} I^*(x_i, x_j) \tag{11}$$

where K is a constant independent of the approximation, and

$$I^*(x_i, x_j) = E\left[\log \frac{P(x_i, x_j)}{P(x_i)P(x_j)}\right] \tag{12}$$

Notice that $I^*(x_i, x_j)$ is simply the Expected Mutual Information Measure (EMIM) between the variables x_i and x_j. An observation of (11) shows that its LHS can be minimized by choosing the dependence tree to be the MST of the complete graph with the EMIM values as the edge weights. This result is quite analogous to the discrete variable case.

There is one important difference to note here. In the discrete case, the evaluation of the EMIM between two variables x_i and x_j only depends on actual discrete probabilities of the joint events. In the continuous case, we encounter probability densities instead of actual discrete probabilities, and hence, there is no straightforward method to numerically compute the quantity defined by (12) if the analytic form of the density function is not known. Even if the analytic form of the densities are known, the quantity defined by (12) may not be computable, if the integrals cannot be evaluated to closed-form expressions[2]. We now investigate the best dependence tree for the case when the random vector $\mathbf{X}$ is normally distributed.

3.2 Optimal Dependence Tree for Normal Distributions

We assume that the random vector $\mathbf{X} = [x_1, x_2, \ldots, x_N]^T$ is jointly normal with mean vector $\boldsymbol{\mu} = [\mu_1, \mu_2, \ldots, \mu_N]^T$ and the covariance matrix $\Sigma = [\sigma_{ij}]$. The distribution of $\mathbf{X}$ will be denoted as $N(\boldsymbol{\mu}, \Sigma)$ and the density function $P(\mathbf{X})$ is given by:

$$P(\mathbf{X}) = (2\pi)^{-N/2}|\Sigma|^{-0.5} \exp\left\{-\frac{1}{2}(\mathbf{X}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{X}-\boldsymbol{\mu})\right\}, \tag{13}$$

where $|\Sigma|$ is the determinant and Σ^{-1} is the inverse of the covariance matrix Σ. Notice that Σ is an $N \times N$ symmetric, positive definite matrix and hence its inverse exists.

The marginal density of the random variable x_i has the usual uni-variate normal density $P(x_i)$ given by:

[2]For example, for the Cauchy distributions, the mean is not defined.

$$P(x_i) = \frac{1}{\sqrt{2\pi}\sigma_i} \exp\left\{-\frac{1}{2}\left(\frac{x_i - \mu_i}{\sigma_i}\right)^2\right\}, \tag{14}$$

where σ_i is the square root of the diagonal entry σ_{ii} of the covariance matrix Σ. Also the joint distribution of the pair of random variables x_i and x_j has the density $P(x_i, x_j)$ given by:

$$P(x_i, x_j) = \frac{1}{2\pi\sigma_i\sigma_j(1 - \rho_{ij}^2)^{1/2}} \exp\{-q(x_i, x_j)\} \tag{15}$$

where,

$$q(x_i, x_j) = \frac{1}{2(1 - \rho_{ij}^2)}\left[\left(\frac{x_i - \mu_i}{\sigma_i}\right)^2 + \left(\frac{x_j - \mu_j}{\sigma_j}\right)^2 - 2\rho_{ij}\left(\frac{x_i - \mu_i}{\sigma_i}\right)\left(\frac{x_j - \mu_j}{\sigma_j}\right)\right] \tag{16}$$

and $\rho_{ij} = \sigma_{ij}/\{\sigma_i\sigma_j\}$ is the correlation coefficient between the two variables x_i, x_j.

The reader must observe that (15) can be obtained from the general form (13) by taking the mean vector $\boldsymbol{\mu}$ and Covariance matrix Σ to be column vector of dimension 2×1 and a square matrix of size 2×2 respectively. The entries in these matrices involve only parameters related to the variables x_i and x_j.

In order to motivate the algorithm to compute the optimum dependence tree, we now state the following result.

Lemma 1 *For all jointly normal pairs of random variables x_i, x_j, the EMIM metric between these variables is given by:*

$$E\left[\log \frac{P(x_i, x_j)}{P(x_i)P(x_j)}\right] = -\frac{1}{2}\log(1 - \rho_{ij}^2).$$

Proof:

The ratio of the joint and marginal densities can be obtained by combining (14) and (15) as follows:

$$\begin{aligned} &\frac{P(x_i, x_j)}{P(x_i)P(x_j)} \\ &\quad = (1 - \rho_{ij}^2)^{-1/2} \exp\left[\frac{\rho_{ij}}{1 - \rho_{ij}^2}\left(\frac{x_i - \mu_i}{\sigma_i}\right)\left(\frac{x_j - \mu_j}{\sigma_j}\right) - \right. \\ &\qquad\qquad \left. \frac{\rho_{ij}^2}{2(1 - \rho_{ij}^2)}\left\{\left(\frac{x_i - \mu_i}{\sigma_i}\right)^2 + \left(\frac{x_j - \mu_j}{\sigma_j}\right)^2\right\}\right] \end{aligned} \tag{17}$$

and hence,

$$\begin{aligned} \log \frac{P(x_i, x_j)}{P(x_i)P(x_j)} &= -\frac{1}{2}\log(1 - \rho_{ij}^2) + \left[\frac{\rho_{ij}}{1 - \rho_{ij}^2}\left(\frac{x_i - \mu_i}{\sigma_i}\right)\left(\frac{x_j - \mu_j}{\sigma_j}\right) - \right. \\ &\quad \left. \frac{\rho_{ij}^2}{2(1 - \rho_{ij}^2)}\left\{\left(\frac{x_i - \mu_i}{\sigma_i}\right)^2 + \left(\frac{x_j - \mu_j}{\sigma_j}\right)^2\right\}\right] \end{aligned} \tag{18}$$

Taking expectations of both sides of (18), we get,

$$E\left[\log \frac{P(x_i, x_j)}{P(x_i)P(x_j)}\right]$$
$$= -\frac{1}{2}\log(1-\rho_{ij}^2) + \left[\frac{\rho_{ij}}{1-\rho_{ij}^2} E\left[\left(\frac{x_i-\mu_i}{\sigma_i}\right)\left(\frac{x_j-\mu_j}{\sigma_j}\right)\right] - \frac{\rho_{ij}^2}{2(1-\rho_{ij}^2)}\left\{E\left[\left(\frac{x_i-\mu_i}{\sigma_i}\right)^2\right] + E\left[\left(\frac{x_j-\mu_j}{\sigma_j}\right)^2\right]\right\}\right] \quad (19)$$

The RHS of (19) can simplified making use of the fact that $E[(x_i-\mu_i)(x_j-\mu_j)] = \sigma_{ij} = \rho_{ij}\sigma_i\sigma_j$ and that $E[(x_i-\mu_i)^2/\sigma_i^2] = 1$. This leads us to:

$$E\left[\log \frac{P(x_i, x_j)}{P(x_i)P(x_j)}\right] = -\frac{1}{2}\log(1-\rho_{ij}^2). \quad (20)$$

and the lemma is proved. □

As a result of Lemma 1, we can now algorithmically present the procedure to compute the best dependence tree for jointly normal random variables. This is given formally[3] as Algorithm DepTree_EMIM in Program 1.

```
Algorithm DepTree_EMIM($\mu, \Sigma, \tau^*$)
Input:
    The parameters $\mu$ and $\Sigma$ for the normal distribution obeyed by $\mathbf{X}$.
Output:
    The best dependence tree, $\tau^*$, as per the EMIM metric.
Method:
    1.  From $\Sigma$, compute $R := [\rho_{ij}]$, the matrix of correlation coefficients.
    2.  Form a complete undirected weighted graph $G^* = \langle V, E, W^* \rangle$, where
            $V := \{x_1, x_2, \ldots, x_N\}$
            $E := \{\langle x_i, x_j \rangle | i, j = 1, 2, \ldots, N; i \neq j\}$
            $W^*(\langle x_i, x_j \rangle) := I^*(x_i, x_j) = -\frac{1}{2}\log(1-\rho_{ij}^2)$.
    3.  $\tau^* :=$ MaximumSpanningTree($G^*$)
End Algorithm DepTree_EMIM
```

Program 1: Algorithm to Determine the Optimal Dependence Tree for Normal Distributions.

Having described the procedure to compute the best dependence tree for Gaussian Distributions, it can be beneficial to try to visualize the distribution characterized by *tree*

[3]It is interesting that although the random variables may have "functionally dependent" means, the mean vector itself does not come into the picture when evaluating stochastic dependence.

dependence. Consider (9) which determines the form of the approximating density function generated by the dependence tree. Notice that the term $\prod_{i=1}^{N} P(x_i)$ is a product of densities of the form defined by (14) and leads to expressions which possess purely quadratic terms in their exponents. Furthermore, the first product in (9) groups terms whose form is given by (17). The net result is a density function with a form exactly identical to a normally distributed variable. In other words, the tree dependence for jointly normal vectors, approximates $N(\boldsymbol{\mu}, \Sigma)$ by another normal distribution $N(\boldsymbol{\mu}, \Sigma_\tau)$. Whereas the means of the two are identical, [4] the covariance matrix of the latter is a "condensed" form of the covariance matrix of the former.

As a consequence of the above, it can be seen that the "cross-product" terms of the form $(x_i - \mu_i)(x_j - \mu_j)/\{\sigma_i \sigma_j\}$ appear only for those indices for which there is an edge between nodes x_i and x_j in the dependence tree. The number of such "cross-terms" is clearly $N - 1$, which is the number of branches in the trees. Combining this information with the fact that such cross terms are contributed by the off-diagonal terms of the symmetric matrix Σ_τ^{-1}, we can conclude that although Σ_τ may be full, Σ_τ^{-1} is sparse and has at most $3N - 2$ elements, which are the N diagonal and $2(N - 1)$ off-diagonal elements !!

The main strength of using the tree approximation is that the computation of Σ_τ^{-1} can be achieved without the need to invert Σ_τ. This is natural consequence of the fact that the elements of this square matrix can be simply expressed in terms of the relevant means, the standard deviations, and the correlation coefficients. The applications of approximating normal distributions with tree dependence has been presented in [3]. We now investigate the dependence tree that results if the chi-squared metric is used for the edge weights.

It must be emphasized that Algorithm DepTree_EMIM assumes that the "true" covariance matrix of the distribution of the random vector $\mathbf{X}$ is known. On the other hand, if only a finite number of samples of the vector $\mathbf{X}$ are available, the question is now one of obtaining a MLE (or for that matter any consistent estimate) of the dependence tree. We shall tackle this matter in a later section and turn our attention to the chi-squared metric being proposed in this paper.

3.3 χ^2 Metric For Determining Dependence Trees

Let us now turn our attention to the problem of using a more computationally effective metric for determining the dependence tree. This metric is the chi-squared metric, $I_\chi(x_i, x_j)$, which, for continuous random variables, quantifies the degree of dependence between random

[4] Although this is not obvious, it becomes clear when one studies the quadratic form in the exponent of the exponential.

variables x_i, x_j as

$$I_\chi(x_i, x_j) = \int\int \frac{\{P(x_i, x_j) - P(x_i)P(x_j)\}^2}{P(x_i)P(x_j)} dx_i dx_j \tag{21}$$

From the definition of $I_\chi(x_i, x_j)$, it is clear that I_χ has the following desirable properties that any metric which captures dependence information should possess:

1. $I_\chi(x_i, x_j) \geq 0$
2. $I_\chi(x_i, x_j) = 0$ if $P(x_i, x_j) = P(x_i)P(x_j)$

In view of the fact that the I^* metric appears naturally in the process of selecting the best dependence tree for continuous distributions, it might be inferred that I_χ cannot possibly lead to the optimal dependence tree. We shall prove that for Gaussian distributions I_χ can be used instead of I^* to yield the *optimal* dependence tree — and this increase in computational efficiency is not forfeited by any optimality.

To do this, we first state our next result which involves the expression for $I_\chi(x_i, x_j)$ for the case of the normal random vector $\mathbf{X}$. In the interest of brevity, many of the intermediate steps in the derivations are omitted.

Theorem 1 *If* $\mathbf{X}$ *has a Gaussian distribution* $N(\boldsymbol{\mu}, \Sigma)$, *then* $I_\chi(x_i, x_j) = \dfrac{\rho_{ij}^2}{1 - \rho_{ij}^2}$.

Proof:

$I_\chi(x_i, x_j)$ given by (21) can be rewritten as:

$$I_\chi(x_i, x_j) = \int\int \frac{(P(x_i, x_j))^2}{P(x_i)P(x_j)} dx_i dx_j - 1.$$

Now, from (15) we can easily write down the expression for $(P(x_i, x_j))^2$ (denoted for convenience as $P^2(x_i, x_j)$) as:

$$P^2(x_i, x_j) = \frac{1}{4\pi^2\sigma_i^2\sigma_j^2(1 - \rho_{ij}^2)} \exp\{-2q(x_i, x_j)\} \tag{22}$$

where, $q(x_i, x_j)$ is given by (16).

Combining (22) and (14), we get,

$$\frac{P^2(x_i, x_j)}{P(x_i)P(x_j)} = \frac{1}{2\pi\sigma_i\sigma_j(1 - \rho_{ij}^2)} \exp\{r(x_i, x_j)\}$$

where,

$$r(x_i, x_j) = -\left\{\left(\frac{x_i - \mu_i}{\sigma_i}\right)^2 + \left(\frac{x_j - \mu_j}{\sigma_j}\right)^2\right\} \frac{1 + \rho_{ij}^2}{2(1 - \rho_{ij}^2)} + \frac{2\rho_{ij}}{1 - \rho_{ij}^2} \left(\frac{x_i - \mu_i}{\sigma_i} \cdot \frac{x_j - \mu_j}{\sigma_j}\right).$$

Consider the integral:

$$I' = \int\int \frac{P^2(x_i, x_j)}{P(x_i)P(x_j)} dx_i dx_j.$$

By making the simple substitution of variables

$$\xi = \frac{x_i - \mu_i}{\sigma_i} \frac{1}{\sqrt{1-\rho_{ij}^2}}, \quad \text{and,} \quad \eta = \frac{x_j - \mu_j}{\sigma_j} \frac{1}{\sqrt{1-\rho_{ij}^2}}$$

the integral I' can be seen to simplify to:

$$I' = \frac{1}{2\pi} \int_{-\infty}^{\infty} \exp\left\{(-\frac{1+\rho_{ij}^2}{2}) \cdot \xi^2\right\} \int_{-\infty}^{\infty} \exp\left\{(-\frac{1+\rho_{ij}^2}{2}) \left(\eta^2 - \frac{4\xi\rho_{ij}\eta}{1+\rho_{ij}^2}\right)\right\} d\xi d\eta.$$

Observe that the vector integral can now be evaluated by first evaluating the integral w.r.t. the variable η and then performing the integration w.r.t. variable ξ. The integration w.r.t. η can be completed, if we observe that the exponent of the exponential can be reduced to a purely quadratic term by "completing the square". Subsequently, we can make use of the standard definite integral given below:

$$\int_{-\infty}^{+\infty} \exp\left\{-a^2x^2\right\} dx = \frac{\sqrt{\pi}}{a}.$$

By carrying out the steps mentioned above, the integral w.r.t. the variable η can be shown to be:

$$\int_{-\infty}^{\infty} \exp\left\{(-\frac{1+\rho_{ij}^2}{2}) \left(\eta^2 - \frac{4\xi\rho_{ij}\eta}{1+\rho_{ij}^2}\right)\right\} d\eta = \exp\left\{\frac{2\rho_{ij}^2\xi^2}{1+\rho_{ij}^2}\right\} \cdot \sqrt{\frac{2\pi}{1+\rho_{ij}^2}}$$

Therefore, the integral I' can be simplified to:

$$\begin{aligned} I' &= \frac{1}{2\pi} \int_{-\infty}^{\infty} \exp\left[\left\{-\frac{1+\rho_{ij}^2}{2} + \frac{2\rho_{ij}^2}{1+\rho_{ij}^2}\right\}\xi^2\right] d\xi \cdot \sqrt{\frac{2\pi}{1+\rho_{ij}^2}} \\ &= \frac{1}{2\pi} \int_{-\infty}^{\infty} \exp\left[\left\{-\frac{(1-\rho_{ij}^2)^2}{2(1+\rho_{ij}^2)}\right\}\xi^2\right] d\xi \cdot \sqrt{\frac{2\pi}{1+\rho_{ij}^2}} \\ &= \frac{1}{1-\rho_{ij}^2} \end{aligned}$$

Hence the chi-squared metric between the two variables x_i and x_j has the form:

$$I_\chi(x_i, x_j) = I' - 1 = \frac{\rho_{ij}^2}{1-\rho_{ij}^2}. \quad (23)$$

and the theorem is proved. □

Thus far, we have shown how to compute the edge weights $I_\chi(x_i, x_j)$ in terms of the correlation coefficient ρ_{ij}. It is now a trivial matter to incorporate these edge weights to yield

```
Algorithm DepTree_CHI ($\mu, \Sigma, \tau_X$)
Input:
    The parameters $\mu$ and $\Sigma$ for the normal distribution obeyed by $\mathbf{X}$.
Output:
    The dependence tree, $\tau_X$, as per the chi-squared metric.
Method:
    1.  From $\Sigma$, compute $R := [\rho_{ij}]$, the matrix of correlation coefficients.
    2.  Form a complete undirected weighted graph $G_X = \langle V, E, W_X \rangle$, where
            $V := \{x_1, x_2, \ldots, x_N\}$
            $E := \{\langle x_i, x_j \rangle | i, j = 1, 2, \ldots, N; i \neq j\}$
            $W_X(\langle x_i, x_j \rangle) := I_X(x_i, x_j) = \frac{\rho_{ij}^2}{1-\rho_{ij}^2}$.
    3.  $\tau_X :=$ MaximumSpanningTree($G_X$)
End Algorithm DepTree_CHI
```

Program 2: Algorithm to Determine the Dependence Tree for Normal Distributions as per the chi-squared metric.

a dependence tree. The procedure to do this is given below in Algorithm DepTree_CHI[5] (see Program 2).

Having given a procedure to obtain the dependence tree τ_X, the question of how τ_X relates to the optimal tree τ^* has now to be considered. Indeed we shall show that τ_X is exactly identical to τ^*, the best dependence tree. To establish this claim, we first state an auxiliary result.

Theorem 2 *Let x_i and x_j be any two components of the normal random vector* $\mathbf{X}$. *Then $I_X(x_i, x_j)$ is a* **monotonic function** *of $I^*(x_i, x_j)$.*

Proof:

Equation (20) which gave the expression for $I^*(x_i, x_j)$ can be rewritten as:

$$\rho_{ij}^2 = 1 - \exp\{-2I^*(x_i, x_j)\} \tag{24}$$

Substituting the expression for ρ_{ij}^2 into (23), we get

$$\begin{aligned} I_X(x_i, x_j) &= \frac{1 - \exp\{-2I^*(x_i, x_j)\}}{\exp\{-2I^*(x_i, x_j)\}} \\ &= \exp\{2I^*(x_i, x_j)\} - 1 \end{aligned} \tag{25}$$

[5]Please see footnote for the input parameters in Algorithm DepTree_EMIM

whence the claim that $I_\chi(x_i, x_j)$ is a monotonic function of $I^*(x_i, x_j)$ follows. □

This theorem naturally leads us to our fundamental result which establishes that τ_χ and τ^* are the same.

Theorem 3 *Let* **X** *be a normally distributed random vector. Then* τ_χ *(obtained by Algorithm DepTree_CHI) is either* **identical to,** *or is an* **equally good approximation** *as the tree* τ^* *produced by the Algorithm DepTree_EMIM.*

Proof:

Consider the complete graphs G^* and G_χ. By virtue of (25), we can arrive at the straightforward conclusion that

$$W^*(x_i, x_j) > W^*(x_k, x_l) \Rightarrow W_\chi(x_i, x_j) > W_\chi(x_k, x_l) \quad \forall i \neq j, \quad k \neq l.$$

This implies that the relative ordering among the $W^*(\cdot, \cdot)$ and the $W_\chi(\cdot, \cdot)$ weights are identical.

The equivalence of the tree produced by the I^* and the I_χ metrics will be shown by simulating Kruskal's algorithm [1]. This algorithm essentially sorts the edges in the descending order of weights and selects the $(N-1)$ edges that do not form cycles. It is apparent that since the relative ordering of the edge weights under the two metrics are identical, the corresponding MSTs computed will be identical.

It must be noted however that the MST is unique only if all the edge weights are distinct. If two edges have identical weights under I^* metric, it can be trivially observed from (25) that they have identical weights under the I_χ metric. Thus in these cases, the MST's produced by the two metrics can only differ in the selection of an edge from the set of edges with equal weight. The sum of the weights of the edges included in the MST will indeed be identical and hence both the MSTs will be equally good approximations. Hence the theorem. □

We shall now turn our attention to the more fundamental task of estimating the dependence tree from the set of samples $\mathcal{X}$ instead of the parameters $\boldsymbol{\mu}$ and Σ.

4 Estimating Dependence Tree From Samples

Throughout our presentation, we implicitly worked with the assumption that the "true" covariance matrix of the random vector **X** was available to us. This knowledge enabled us to determine the optimal dependence tree by Algorithm DepTree_EMIM, or equivalently, by the computationally efficient Algorithm DepTree_CHI. The situation however changes if instead of the true parameters of the normal distribution of the random vector **X**, only a finite number of representative samples are presented to the procedure which estimates the dependence tree.

An obvious method can be envisioned. Since the covariance matrix is unknown, one can first estimate it and subsequently follow either the procedure of Algorithm DepTree_EMIM or DepTree_CHI. This procedure is intuitively appealing, but its properties as an estimator of the *dependence tree* are not obvious. We shall now prove that this simple minded and straightforward procedure indeed produces the MLE of the dependence tree when the MLEs of the corresponding parameters $\hat{\boldsymbol{\mu}}$ and $\hat{\Sigma}$ are used in Algorithm DepTree_CHI.

Consider Algorithm Estimate_DepTree_CHI (in Program 3). The reader will observe that this algorithm is identical to Algorithm DepTree_CHi except that instead of having the input parameters as $\boldsymbol{\mu}$ and Σ, the input is a set $\mathcal{X}$ of s statistically independent observations of $\mathbf{X}$.

```
Algorithm Estimate_DepTree_CHI ($\mathcal{X}, \hat{\tau}$)
Input:
    $\mathcal{X} = s$ statistically independent samples $X^1, X^2, \ldots, X^s$.
Output:
    An estimate of the dependence tree, $\hat{\tau}$, as per the chi-squared metric.
Method:
    1.  Compute the MLEs of the parameters of $\mathbf{X}$
            $\hat{\boldsymbol{\mu}} := \frac{1}{s}\sum_{k=1}^{s} X^k$,   and,
            $\hat{\Sigma} = \frac{1}{s}\sum_{k=1}^{s}(X^k - \hat{\boldsymbol{\mu}})(X^k - \hat{\boldsymbol{\mu}})^T$
    2.  Invoke Algorithm DepTree_CHI ($\hat{\boldsymbol{\mu}}, \hat{\Sigma}, \hat{\tau}$)
End Algorithm Estimate_DepTree_EMIM
```

Program 3: Estimate Dependence Tree from Samples.

We now state and prove the fundamental property of the tree estimated by Algorithm Estimate_DepTree_CHI.

Theorem 4 *Algorithm Estimate_DepTree_CHI produces the Maximum Likelihood Estimate of the dependence tree.*

Proof:

First of all note that a tree type approximation to the given density function is completely described by the structure of the tree and the parameters associated with this tree. The parameters to be chosen consist of the Mean Vector, the Variances of all the components of the random vector and the correlation coefficient for the pair of nodes connected by an edge in the dependence tree. In the following derivation, we shall denote the set of parameters as $\boldsymbol{\theta}$ and the tree by τ.

Let $X^1, X^2, \ldots, X^s$ be the s *statistically independent* samples drawn from the underlying normal distribution. Let $p(X^k|\boldsymbol{\theta}, \tau)$ denote the value of the density function for the sample X^k, given the set of current parameter values and the dependence tree τ. Then, using (9), we can write:

$$p(X^k|\boldsymbol{\theta},\tau) = \prod_{[i,j]\in\tau} \frac{P(x_i^k, x_j^k)}{P(x_i^k)P(x_j^k)} \cdot \prod_{l=1}^{N} P(x_l^k),$$

where x_j^k is the j^{th} component of the sample X^k. Equivalently, by taking logarithms,

$$\log p(X^k|\boldsymbol{\theta},\tau) = \sum_{[i,j]\in\tau} \log \frac{P(x_i^k, x_j^k)}{P(x_i^k)P(x_j^k)} + \sum_{l=1}^{N} \log P(x_l^k).$$

Utilizing (18), we obtain:

$$\log \frac{P(x_i^k, x_j^k)}{P(x_i^k)P(x_j^k)} = -\frac{1}{2}\log(1-\rho_{ij}^2) - R(x_i^k, x_j^k)$$

where,

$$R(x_i^k, x_j^k) \;=\; -\left[\frac{\rho_{ij}}{1-\rho_{ij}^2}\left(\frac{x_i^k-\mu_i}{\sigma_i}\right)\left(\frac{x_j^k-\mu_j}{\sigma_j}\right) - \frac{\rho_{ij}^2}{2(1-\rho_{ij}^2)}\left\{\left(\frac{x_i-\mu_i}{\sigma_i}\right)^2 + \left(\frac{x_j-\mu_j}{\sigma_j}\right)^2\right\}\right]$$

Also, utilizing the expression for the univariate Gaussian density given by (14), we can obtain the log-likelihood function ℓ as:

$$\begin{aligned}
\ell &= \sum_{k=1}^{s} \log p(X^k|\boldsymbol{\theta},\tau) \\
&= \sum_{k=1}^{s}\left[\sum_{[i,j]\in\tau}\left\{-\frac{1}{2}\log(1-\rho_{ij}^2) - R(x_i^k,x_j^k)\right\}\right. \\
&\qquad \left. -\sum_{l=1}^{N}\left\{\frac{1}{2}\left(\frac{x_l^k-\mu_l}{\sigma_l}\right)^2 + \frac{1}{2}\log(2\pi) + \log\sigma_l\right\}\right] \\
&= \sum_{[i,j]\in\tau}\left[-\frac{s}{2}\log(1-\rho_{ij}^2) - \sum_{k=1}^{s} R(x_i^k,x_j^k)\right] \\
&\qquad -\sum_{l=1}^{N}\sum_{k=1}^{s}\left\{\frac{1}{2}\left(\frac{x_l^k-\mu_l}{\sigma_l}\right)^2 + \frac{1}{2}\log(2\pi) + \log\sigma_l\right\}.
\end{aligned} \tag{26}$$

In trying to arrive at the best approximation, we are faced with a complex optimization problem, which is that of choosing the dependence tree, and of choosing its associated parameters. The problem would be in a sense "decomposable" if we can independently choose the tree and the values of the parameters. Fortunately we can show that the "best" values of the parameters, $\hat{\boldsymbol{\theta}}$, is the same, irrespective of what the estimated dependence tree is. This, in turn, permits us to perform the maximization of ℓ in a straightforward fashion.

We shall first investigate the values of the parameters in $\boldsymbol{\theta}$ that would maximize the quantity ℓ for a *given tree*. In order to achieve this, we merely have to differentiate ℓ w.r.t. $\boldsymbol{\theta}$ and set the result to 0, which in turn would yield the estimate $\hat{\boldsymbol{\theta}}$. The set of equations,

$$\frac{\partial l}{\partial \mu_\alpha} = 0 \quad \text{for } \alpha = 1, 2, \ldots N$$

simplifies to

$$-\sum_{[\alpha,j]\in\mathcal{T}} \sum_{k=1}^{s} \frac{\partial R(x_\alpha^k, x_j^k)}{\partial \mu_\alpha} + \sum_{k=1}^{s} \left(\frac{x_\alpha^k - \mu_\alpha}{\sigma_\alpha} \right) = 0,$$

which, after some manipulation, yields,

$$-\sum_{[\alpha,j]\in\mathcal{T}} \frac{1}{2(1-\rho_{\alpha j}^2)} \left[\left\{ \sum_{k=1}^{s} 2 \left(\frac{x_\alpha^k - \mu_\alpha}{\sigma_\alpha} \right) (-1) \right\} \rho_{\alpha j}^2 - \sum_{k=1}^{s} 2\rho_{\alpha j} (-\frac{1}{\sigma_\alpha}) \left(\frac{x_j^k - \mu_j}{\sigma_j} \right) \right]$$
$$+ \sum_{k=1}^{s} \left(\frac{x_\alpha^k - \mu_\alpha}{\sigma_\alpha} \right) = 0,$$

which equivalently is:

$$\sum_{[\alpha,j]\in\mathcal{T}} \frac{\rho_{\alpha j}^2}{(1-\rho_{\alpha j}^2)} \cdot \left\{ \sum_{k=1}^{s} \left(\frac{x_\alpha^k - \mu_\alpha}{\sigma_\alpha} \right) \right\} - \frac{\rho_{\alpha j}}{(1-\rho_{\alpha j}^2)\sigma_\alpha} \cdot \left\{ \sum_{k=1}^{s} \left(\frac{x_j^k - \mu_j}{\sigma_j} \right) \right\}$$
$$+ \left\{ \sum_{k=1}^{s} \left(\frac{x_\alpha^k - \mu_\alpha}{\sigma_\alpha} \right) \right\} = 0 \tag{27}$$

Differentiating ℓ w.r.t. $\rho_{\alpha\beta}$, where $[\alpha, \beta] \in \tau$, we get:

$$\frac{\partial \ell}{\partial \rho_{\alpha\beta}} = 0$$

which can be written down as [6]:

$$-\frac{s}{2} \frac{(-2\rho_{\alpha\beta})}{1-\rho_{\alpha\beta}^2} - \sum_{k=1}^{s} \frac{\partial R(x_\alpha^k, x_\beta^k)}{\partial \rho_{\alpha\beta}} = 0,$$

and can be further simplified to:

$$\frac{s\rho_{\alpha\beta}}{1-\rho_{\alpha\beta}^2} - \frac{\rho_{\alpha\beta}}{(1-\rho_{\alpha\beta}^2)^2} \left[\sum_{k=1}^{s} \left\{ \left(\frac{x_\alpha^k - \mu_\alpha}{\sigma_\alpha} \right)^2 + \left(\frac{x_\beta^k - \mu_\beta}{\sigma_\beta} \right)^2 \right\} \right]$$
$$+ \frac{1+\rho_{\alpha\beta}^2}{(1-\rho_{\alpha\beta}^2)^2} \sum_{k=1}^{s} \frac{(x_\alpha^k - \mu_\alpha)(x_\beta^k - \mu_\beta)}{\sigma_\alpha \sigma_\beta} = 0 \tag{28}$$

Also,

[6]Notice that we have made use of the results that the derivatives of the functions $\rho/(1-\rho^2)$ and $\rho^2/(1-\rho^2)$ w.r.t. ρ are $(1+\rho^2)/(1-\rho^2)^2$ and $2\rho/(1-\rho^2)^2$ respectively.

$$\frac{\partial \ell}{\partial \sigma_\alpha} = 0 \quad \text{for } \alpha = 1, 2, \ldots N,$$

yields (on simplification) the following equations:

$$- \sum_{[\alpha,j]\in\tau} \left[\frac{1}{1-\rho_{\alpha j}^2} \left\{ \rho_{\alpha j}^2 (-2) \sum_{k=1}^{s} \frac{(x_\alpha^k - \mu_\alpha)^2}{\sigma_\alpha^3} \right\} + 2\rho_{\alpha j} \sum_{k=1}^{s} \frac{(x_\alpha^k - \mu_\alpha)(x_\beta^k - \mu_\beta)}{\sigma_\alpha^2 \sigma_\beta} \right]$$
$$+ \sum_{k=1}^{s} \frac{(x_\alpha^k - \mu_\alpha)^2}{\sigma_\alpha^3} - \frac{s}{\sigma_\alpha} = 0. \qquad (29)$$

By direct substitution we can verify that (27), (28) and (29) are satisfied if:

$$\begin{aligned} \hat{\mu}_i &= \frac{1}{s} \sum_{k=1}^{s} x_i^k \\ \hat{\sigma}_i &= \frac{1}{s} \sum_{k=1}^{s} (x_i^k - \hat{\mu}_i)^2 \\ \hat{\rho}_{ij} &= \frac{1}{s\hat{\sigma}_i\hat{\sigma}_j} \sum_{k=1}^{s} (x_i^k - \hat{\mu}_i)(x_j^k - \hat{\mu}_j). \end{aligned}$$

which interestingly enough are the MLEs of the parameters of the distribution of $\mathbf{X}$.

Notice that all these parameters are **independent** of the tree τ. With these values of the parameters $\boldsymbol{\theta}$, let us consider the quantity $\sum_{k=1}^{s} R(x_i^k, x_j^k)$ which appears in (26). Now,

$$\begin{aligned} &\textstyle\sum_{k=1}^{s} R(x_i^k, x_j^k) \\ &= \frac{\hat{\rho}_{ij}^2}{2(1-\hat{\rho}_{ij}^2)} \left[\sum_{k=1}^{s} \left\{ \left(\frac{x_i^k - \hat{\mu}_i}{\hat{\sigma}_i} \right)^2 + \left(\frac{x_j^k - \hat{\mu}_j}{\hat{\sigma}_j} \right)^2 \right\} \right] - \frac{\hat{\rho}_{ij}}{1-\hat{\rho}_{ij}^2} \sum_{k=1}^{s} \frac{(x_i^k - \hat{\mu}_i)(x_j^k - \hat{\mu}_j)}{\hat{\sigma}_i\hat{\sigma}_j} \\ &= \frac{\hat{\rho}_{ij}^2}{2(1-\hat{\rho}_{ij}^2)} \cdot 2s - \frac{\hat{\rho}_{ij}}{1-\hat{\rho}_{ij}^2} \cdot \frac{s\hat{\sigma}_i\hat{\sigma}_j\hat{\rho}_{ij}}{\hat{\sigma}_i\hat{\sigma}_j} \\ &= 0. \end{aligned}$$

And hence, ℓ in (26) can be simplified to:

$$\ell = -\frac{s}{2} \sum_{[i,j]\in\tau} \log(1 - \hat{\rho}_{ij}^2) + k'. \qquad (30)$$

where k' is a constant independent of the tree and only depends only on the samples. An examination of (30) reveals that the MLE of the dependence tree can be found as the Maximum Spanning Tree of a weighted graph, whose edge weights are the values of the corresponding EMIM metric, calculated using the *estimates* of the correlation coefficients. The theorem follows since we have already established that when presented with identical input, Algorithms DepTree_EMIM and DepTree_CHI *always* produce identical (or equivalent) trees. □

Remark:

The reader will recall that the maximization of the log-likelihood function ℓ required the choice of the dependence tree, as well the parameters that completely specify the approximation. Also worth noting is the fact that the set of parameters associated with *different trees* are not identical. To clarify this, consider two dependence trees τ_1, and τ_2, where τ_1 includes the edge $\langle x_i, x_j \rangle$ and τ_2 does not. Clearly, the correlation coefficient, ρ_{ij} will be part of the parameter set associated with τ_1. However, ρ_{ij} is not needed when approximating with the tree τ_2. Viewed in this perspective, the proof presented above demonstrates that the parameters **common** to any two trees, must have the **same value**, in order to maximize the log-likelihood function. This simplifies the search process for the best approximation. Indeed, in every case, we merely choose all the means, variances and the correlation coefficients to be their respective MLEs and subsequently compute the best estimate of the dependence tree as the MST of the appropriately weighted graph.

5 Experimental Results

Simulations were carried out to validate the results in this paper. Since the Chi-squared and the EMIM metric were analytically shown to be exactly equivalent for normal vectors, numerous experiments conducted to verify the optimality of the chi-squared metric are not reported here, because they do not add to the fundamental contribution of this paper.

In our simulations, we chose to monitor the only characteristic that was not analytically studied — the rate of convergence of the algorithm. In these experiments, we chose an underlying dependence tree (denoted as τ_r) and observed how quickly the estimates τ_X (and hence τ^*) converged to it. Besides observing the convergence to the real tree, we also observed the improvement in the generated approximation, as more and more samples were seen by the system attempting to learn the characteristics of the distribution. The simulations were carried out for various values of the dimension of the random vector $\mathbf{X}$. The following procedure was followed for each value of the dimension parameter N.

Once the dimension of the random vector $\mathbf{X}$ was assigned, a predefined dependence tree was selected as follows. The structure of the dependence tree was initially arbitrarily selected. The means and variances of each of the variables $\{x_i\}$ were then assigned to be independent, uniformly distributed values in the interval (0,1). Subsequently, for each edge $\langle x_i, x_j \rangle$ in the dependence tree, the correlation coefficient ρ_{ij} was uniformly chosen from the interval $(-1 \ldots 1)$. All the parameters characterizing the tree dependence being specified, the inverse of the covariance matrix (i.e. Σ^{-1}) was computed. This matrix was inverted to obtain the "real" underlying covariance matrix Σ from which the samples, $\mathcal{X}$, were generated. These pre-processing steps were accomplished in order to completely describe the distribution of

X. The subsequent steps consisted of generating the samples and estimating the dependence tree from them.

Using the technique described in Appendix A samples were generating according to the distribution $N(\boldsymbol{\mu}, \Sigma)$.[7] We denote the k^{th} sample as $\mathbf{X}^k$. Let m_k and C_k denote the MLEs of the mean vector and the covariance matrix, after the k^{th} sample vector were seen. These estimates were updated as the samples were processed, using the recurrence relations[4, pp.82]:

$$\begin{aligned} m_{n+1} &= \frac{1}{n+1}\left[n m_n + \mathbf{X}^{n+1}\right] \\ C_{n+1} &= \frac{n}{n+1} C_n + \frac{n}{(n+1)^2}(\mathbf{X}^{n+1} - m_n)(\mathbf{X}^{n+1} - m_n)^T. \end{aligned}$$

where, $m_1 = X^1$, and C_1 is the null matrix. Notice that the above recurrence relation for C_{n+1} is slightly different from the relation that appears in [4, pp.82]. This is due to the fact that C_n refers to the MLE of the covariance matrix, and is not the unbiased estimator used in [4, pp.82].

Once the estimated covariance matrix was available, Algorithm DepTree_CHI was invoked to estimate the dependence tree. In our simulations, we opted not to compute the dependence tree for the first 50 samples so as to ensure that we were dealing with "reasonable" estimates for the mean vector and the covariance matrix.

It was observed during the simulations that the estimate $\hat{\tau}$ produced by Algorithm Estimate_DepTree_EMIM converges very quickly to the underlying "real" dependence tree τ_r. However, the *distribution* generated by the estimate $\tau_{\mathbf{X}}$ converges more sluggishly to the underlying distribution, even though $\tau_{\mathbf{X}}$ and τ_r are structurally identical. This is essentially due to inaccurate estimates for the mean vector and the covariance matrix.

As far as experimental results are concerned, the gradual improvement in the "quality" of approximation, as the number of samples increases, is reported in the graphs given below. The quality of approximation, is of course the information theoretic closeness measure, $I(P, P_\tau)$, between the "real" distribution $N(\boldsymbol{\mu}, \Sigma)$ and the approximating distribution $N(\hat{\boldsymbol{\mu}}, \hat{\Sigma})$. This closeness of approximation has the following explicit form:

$$I(P, P_\tau) = 0.5 \log \frac{|\hat{\Sigma}|}{|\Sigma|} - 0.5\left\{N - \mathrm{tr}(\hat{\Sigma}^{-1}\Sigma) - (\boldsymbol{\mu} - \hat{\boldsymbol{\mu}})^T \hat{\Sigma}^{-1}(\boldsymbol{\mu} - \hat{\boldsymbol{\mu}})\right\}. \tag{31}$$

where $\mathrm{tr}(A)$ denotes the trace of the square matrix A. That $I(P, P_\tau)$ has the form of (31) is shown in Appendix B.

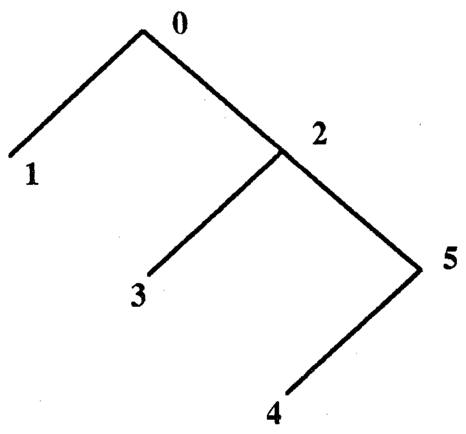
[7]The question of generating random vectors with the distribution $N(\boldsymbol{\mu}, \Sigma)$ is so common and yet a systematic procedure to achieve it not readily available. We have included it here for the sake of completeness. A secondary purpose of including it here is to enable other researchers to verify and expand on our work.

We now present below the results obtained for two typical distributions.

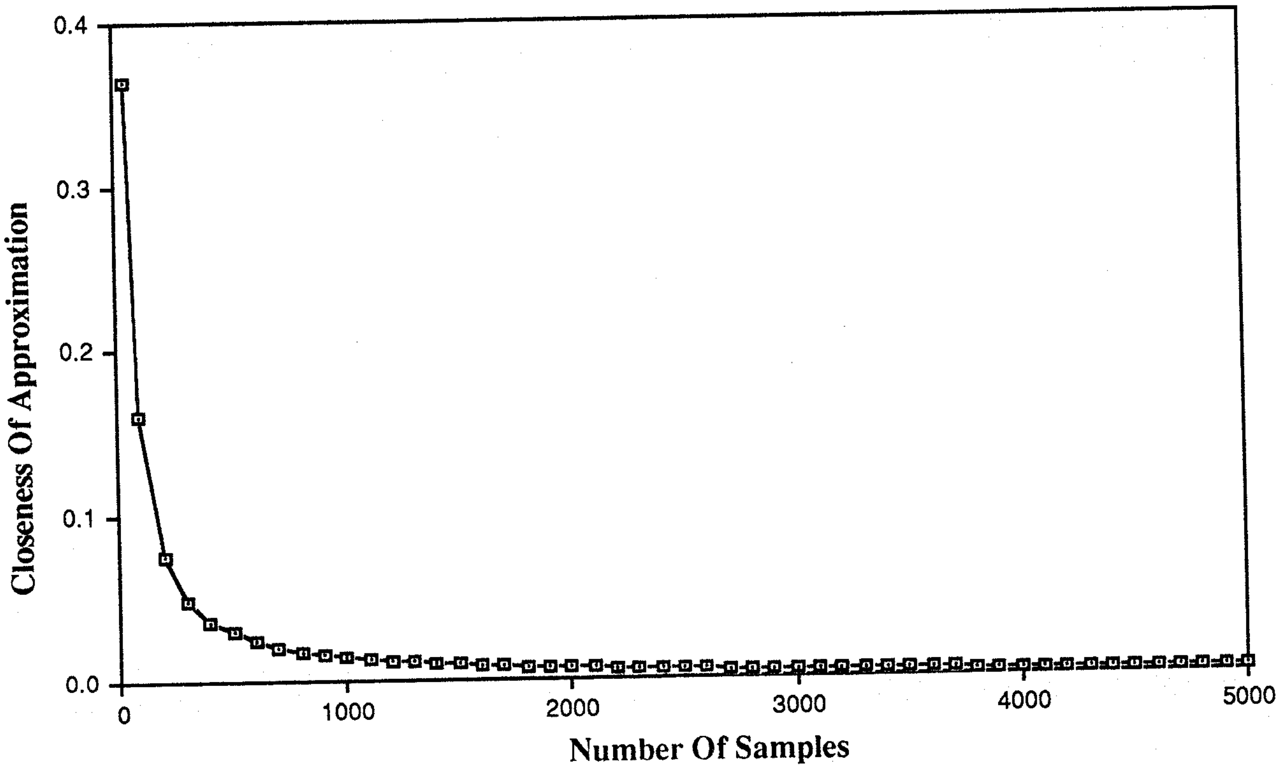
In the first case, the dimension of the vector, N was 6. The underlying dependence tree is shown in Figure 1a. Figure 1b shows the closeness measure as a function of n, the number of samples. Note that the closeness measure reported in this figure is the ensemble average over 50 experiments, with each experiment lasting for 5000 samples. Observe the monotonic decrease in the closeness measure with the number of samples. Thus, for example, in Figure 1b, the closeness measure was 0.363854 after 50 samples were processed. This value dropped to 0.01 after processing 1000 samples. It was observed that the average closeness measure attained a value of 0.002681 after the 5000 samples were processed.

Figure 2 presents similar results for the case when the dimension of the vector, N, was 10. The experiments were conducted in a fashion identical to the one reported above. Again the initial value of the closeness measure was 0.996056, at the end of 50 samples. This value attains the value of 0.03220 and 0.0061, after 1000 and 5000 samples were processed, respectively. The power of the scheme is obvious.

Other experimental results justifying the claims of the paper are found in [11] but are omitted here, in the interest of brevity.

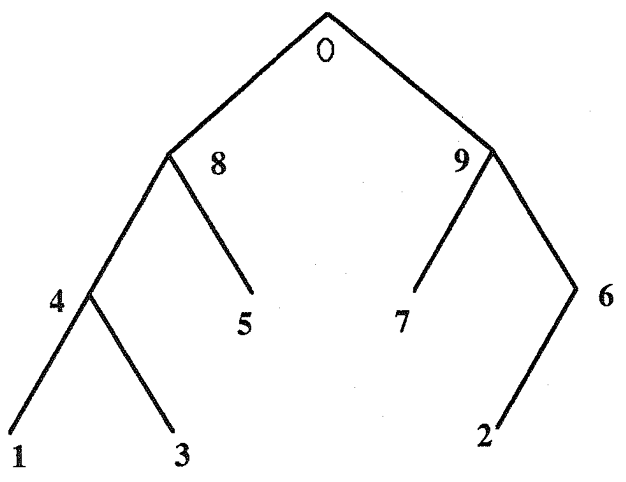


(a) Underlying dependence tree used for generating the samples to measure the rate of convergence of τ_X. In this case, N, the dimensionality of the random vector is 6, which is equal to the number of nodes in this tree.

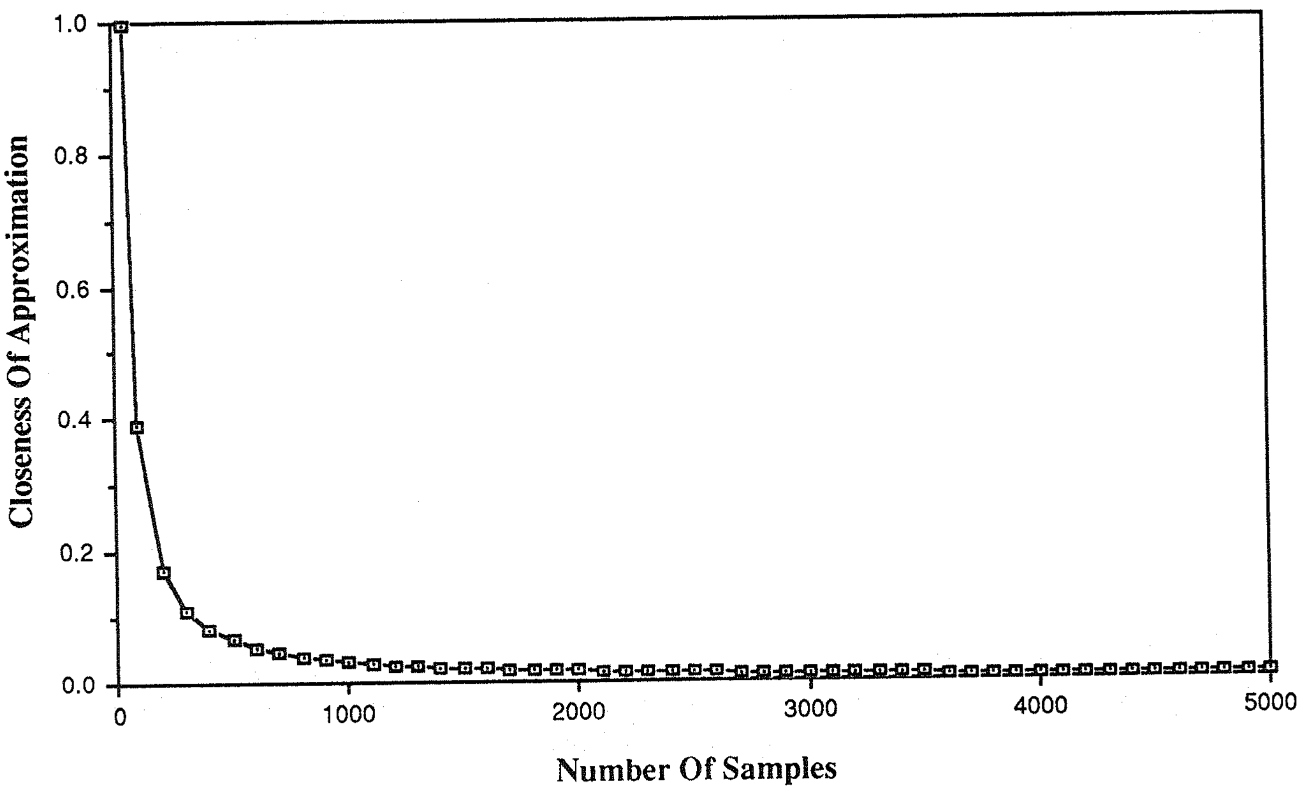


(b) This figure shows the ensemble average of the closeness of approximation $I(P, P_a)$ between the actual density P and the approximating density P_a derived from the estimated dependence tree. The underlying tree describing P is shown in Figure 1a.

Figure 1: Rate of Convergence (N=6).



(a) Underlying dependence tree used for generating the samples to measure the rate of convergence of τ_X. In this case, N, the dimensionality of the random vector is 10, which is equal to the number of nodes in this tree.



(b) This figure shows the ensemble average of the closeness of approximation $I(P, P_a)$ between the actual density P and the approximating density P_a derived from the estimated dependence tree. The underlying tree describing P is shown in Figure 2a.

Figure 2: Rate of Convergence (N=10).

6 Conclusion

In this paper, we have considered the problem of approximating the a multi-variate normal distribution with one derived from a dependence tree. The best dependence tree τ^* is known to be the MST of a complete graph, with $I^*(x_i, x_j)$ as the edge weight between the pair of nodes x_i and x_j, where I^* is the EMIM metric between the corresponding variables.

This paper proposes a chi-squared based metric, I_χ to capture the dependence information between pairs of random variables. For the case of normally distributed vectors, the I_χ metric is shown to be a monotonic function of the metric I^*. As a natural consequence of this property, τ_χ, the dependence tree found using the I_χ metric was shown to be identical to, or as good an approximation as τ^*.

The paper also considers the situation in which only a finite number of samples are available, instead of the actual parameters characterizing the normal distribution in question. It is shown in this paper that the "natural" sequence of first estimating the parameters of the distribution, and then using these estimates *as if they were the true parameters*, to determine the dependence tree yields the MLE of the dependence tree.

The paper also includes simulations verifying and demonstrating the power of the strategy proposed.

References

[1] A.V. Aho, J.E. Hopcroft, J.D. Ullman, *The Design and Analysis of Algorithms*, Addison-Wesley, 1974.

[2] C.K. Chow and C.L. Liu, "Approximating Discrete Probability Distributions Using Dependence Trees", *IEEE Trans. on Information Theory*, Vol. IT-14, 1968, pp. 462-467.

[3] C.K. Chow, S.S.M. Wang, and, J.H. Siegel, "Sequential Classification of Patient Recovery Patterns After Coronary Artery Bypass Graft Surgery", *Computers and Biomedical Research*, Vol.12, 1979, pp.589-613.

[4] R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis*, Wiley Interscience, 1973.

[5] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, Academic Press, London, 1972.

[6] D.E. Knuth, *The Art Of Computer Programming: Vol.2 Seminumerical Algorithms*, Addison-Wesley, 1981.

[7] H. Ku and S. Kullback, "Approximating Discrete Probability Distributions", *IEEE Trans. on Information Theory*, Vol. IT-14, 1968, pp. 462-467.

[8] K. Press, B.P. Flannery, S.A. Teukolsky, W.T. Vetterling, *Numerical Recipes : The Art of Scientific Computing*, Cambridge University Press, New York, 1986.

[9] J.R. Magnus, and H. Neudecker, *Matrix Differential Calculus: with Applications in Statistics and Econometrics*, John Wiley, New York, 1988.

[10] R.S. Valiveti and B.J. Oommen, "A New Metric For Determining Dependence Trees For Pattern Recognition", *Proc. of the 1989 Intl. Symp. on Computer Arch. and Digital Signal Processing*, Oct.11-14 1989, Hong Kong, pp. 474-479.

[11] R.S. Valiveti, Ph.D. thesis, Carleton University, In preparation.

[12] C.J. Van Rijsbergen, "A Theoretical Basis For the Use of Co-occurrence Data in Information Retrieval", *Journal Of Documentation*, Vol. 33, No. 2, June 1977, pp. 106-119.

[13] S.K.M. Wong, and F.C.S. Poon, "Comments on approximating Discrete Probability Distribution with Dependence Trees", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 11, No. 3, March 1989, pp. 333-335.

Appendices

A Generating Samples of Normally Distributed Vectors

In this section, we consider the problem of generating a random vector **X** whose distribution is $N(\boldsymbol{\mu}, \Sigma)$, where $\boldsymbol{\mu}$ and Σ are **known** parameters. The generation technique is based on the following property (stated without proof).

Theorem A.1 *Let* **X** *be the vector* $[x_1, x_2, \ldots, x_N]^T$ *which has the distribution* $N(\boldsymbol{\mu}, \Sigma)$*. Then the random vector* $\mathbf{Y} = A\mathbf{X}$ *has the distribution* $N(A\boldsymbol{\mu}, A\Sigma A^T)$*.*

Corollary A.1 *Let* $\boldsymbol{\mu}, \Sigma$ *be the mean vector and the covariance matrix of a normal distribution respectively. Let* Z *be a normal random vector which has the distribution* $N(0, I)$*. Also let* $\Lambda = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_N)$ *and* Φ *be the diagonal eigenvalue matrix, and the eigenvector matrix of* Σ *respectively. Then*

$$\mathbf{X} = \boldsymbol{\mu} + \Phi\Lambda^{1/2}Z$$

has the normal distribution $N(\boldsymbol{\mu}, \Sigma)$*.*

Proof:

It is clear that the random vector **X** has the mean vector $\boldsymbol{\mu}$. Furthermore, using Theorem A.1, the covariance matrix of **X** can evaluated as:

$$\text{Covariance}(\mathbf{X}) = \Phi\Lambda^{1/2}\text{Covariance}(Z)\left\{\Phi\Lambda^{1/2}\right\}^T = \Phi\Lambda^{1/2}I\Lambda^{1/2}\Phi^T = \Phi\Lambda\Phi^T.$$

Because of the fact that Φ is the orthogonal eigenvector matrix for the covariance matrix Σ, it follows that $\Phi\Lambda\Phi^T = \Sigma$. Hence the result. □

The above corollary leads to a straightforward technique to generate samples drawn from a normal distribution, given its parameters. This is given below in Algorithm GenerateNormalSamples (in Program 4).

```
Algorithm GenerateNormalSamples($\boldsymbol{\mu}, \Sigma, X$)
Input:
    The parameters $\boldsymbol{\mu}$ and $\Sigma$ for the normal distribution obeyed by $\mathbf{X}$.
Output:
    A random vector $X$ drawn from the distribution $N(\boldsymbol{\mu}, \Sigma)$.
Memory requirements:
    $Z$ – a vector of dimension $N \times 1$
    $\Phi, \Lambda$ – square matrices of dimension $N \times N$.
Assumptions:
    Assumes that the function "GetUnivariateNormal", for generating nor-
    mal variates with the distribution $N(0,1)$ is available. One method of
    realizing this function can be found in [6, pp.117-118]
    The procedure "GetEigenValuesAndVectors" is assumed to be present,
    for determining the eigenvalues and the eigenvectors of a real symmetric
    matrix. The most common procedure for achieving this computation
    is the Jacobi method described in [8].
Method:
    1.   GetEigenValuesAndVectors ($\Sigma, \Phi, \Lambda$)
    1.   for $I := 1$ to $N$ do
                 $Z[I]$ := GetUnivariateNormal
    3.   Compute $X = \boldsymbol{\mu} + \Phi\Lambda^{1/2}Z$.
End Algorithm GenerateNormalSamples
```

Program 4: Algorithm to Generate Samples from a Multi-variate Normal Distribution.

B Computing Closeness Measure Between Two Normal Distributions

In this section, we consider the problem of computing the closeness measure between two normal distributions described as $N(\boldsymbol{\mu}, \Sigma)$ and $N(\boldsymbol{\mu}_a, \Sigma_a)$. The expression for the closeness measure between these distributions was reported in (31) and is given by Theorem B.1. The derivation makes use of a central result, which is given as Lemma B.1 below.

Lemma B.1 *Let* $\mathbf{X}$ *be a random (column) vector of dimension* $N \times 1$*, whose mean vector is* $\boldsymbol{\mu}$ *and the covariance matrix is* Σ*. Then, if* A *is any square matrix of dimension* $N \times N$*, the expected value of* $\mathbf{X}^T A\mathbf{X}$ *is given by:*

$$E[\mathbf{X}^T A\mathbf{X}] = \mathrm{tr}(A\Sigma) + \boldsymbol{\mu}^T A\boldsymbol{\mu}.$$

where $\mathrm{tr}(A)$ denotes the trace of the square matrix A.

Proof:

The proof can be found in [9, pp.247]. □

Using Lemma B.1, we prove the main result of this Appendix.

Theorem B.1 *Let* $N(\boldsymbol{\mu}, \Sigma)$ *and* $N(\boldsymbol{\mu}_a, \Sigma_a)$ *denote two normal distributions. Then* $I(P, P_a)$*, the information theoretic closeness measure between these distributions is given by:*

$$I(P, P_a) = 0.5 \log \frac{|\Sigma_a|}{|\Sigma|} - 0.5 \left\{ N - \mathrm{tr}(\Sigma_a^{-1}\Sigma) - (\boldsymbol{\mu} - \boldsymbol{\mu}_a)^T \Sigma_a^{-1} (\boldsymbol{\mu} - \boldsymbol{\mu}_a) \right\}. \tag{B.1}$$

Proof:

We begin by writing the explicit forms of the the density functions for the two normal distributions under consideration:

$$\begin{aligned} P(\mathbf{X}) &= (2\pi)^{-N/2}|\Sigma|^{-0.5} \exp\left\{-\frac{1}{2}(\mathbf{X} - \boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{X} - \boldsymbol{\mu})\right\}, \\ P_a(\mathbf{X}) &= (2\pi)^{-N/2}|\Sigma_a|^{-0.5} \exp\left\{-\frac{1}{2}(\mathbf{X} - \boldsymbol{\mu}_a)^T\Sigma_a^{-1}(\mathbf{X} - \boldsymbol{\mu}_a)\right\}. \end{aligned} \tag{B.2}$$

From (B.2) we can easily obtain the ratio of $P(\mathbf{X})$ to $P_a(\mathbf{X})$ as:

$$\frac{P(\mathbf{X})}{P_a(\mathbf{X})} = \frac{|\Sigma_a|^{0.5}}{|\Sigma|^{0.5}} \cdot \exp\left\{-\frac{1}{2}(\mathbf{X} - \boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{X} - \boldsymbol{\mu}) + \frac{1}{2}(\mathbf{X} - \boldsymbol{\mu}_a)^T\Sigma_a^{-1}(\mathbf{X} - \boldsymbol{\mu}_a)\right\}.$$

or equivalently,

$$\log \frac{P(\mathbf{X})}{P_a(\mathbf{X})} = 0.5 \log \frac{|\Sigma_a|}{|\Sigma|} - \frac{1}{2}(\mathbf{X} - \boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{X} - \boldsymbol{\mu}) + \frac{1}{2}(\mathbf{X} - \boldsymbol{\mu}_a)^T\Sigma_a^{-1}(\mathbf{X} - \boldsymbol{\mu}_a). \tag{B.3}$$

From the definition of the closeness measure $I(P, P_a)$, it clear that this quantity is simply the expected value of the random variable $\log \frac{P(\mathbf{X})}{P_a(\mathbf{X})}$. Thus,

$$\begin{aligned} I(P, P_a) & \\ &= E\left[\log \frac{P(\mathbf{X})}{P_a(\mathbf{X})}\right] \\ &= 0.5 \log \frac{|\Sigma_a|}{|\Sigma|} - \frac{1}{2} E\left[(\mathbf{X}-\boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{X}-\boldsymbol{\mu})\right] + \frac{1}{2} E\left[(\mathbf{X}-\boldsymbol{\mu}_a)^T \Sigma_a^{-1} (\mathbf{X}-\boldsymbol{\mu}_a).\right] \end{aligned} \quad \text{(B.4)}$$

We can make use of Lemma B.1 to compute the expected values of the two quadratic forms that are encountered in (B.4). We first concentrate on computing the expected value of the first quadratic form on the RHS of (B.3). We note that since the mean of the random vector $\mathbf{X}$ is $\boldsymbol{\mu}$, the mean of the random vector $\mathbf{X} - \boldsymbol{\mu}$ is 0. Thus by Lemma B.1:

$$E\left[(\mathbf{X}-\boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{X}-\boldsymbol{\mu})\right] = \mathrm{tr}(\Sigma^{-1}\Sigma) = N. \quad \text{(B.5)}$$

Also noting that $\mathbf{X} - \boldsymbol{\mu}_a$ has the mean $\boldsymbol{\mu} - \boldsymbol{\mu}_a$, a second application of Lemma B.1 yields:

$$E\left[(\mathbf{X}-\boldsymbol{\mu}_a)^T \Sigma_a^{-1} (\mathbf{X}-\boldsymbol{\mu}_a)\right] = \mathrm{tr}(\Sigma_a^{-1}\Sigma) + (\boldsymbol{\mu}-\boldsymbol{\mu}_a)^T \Sigma_a^{-1} (\boldsymbol{\mu}-\boldsymbol{\mu}_a). \quad \text{(B.6)}$$

Substituting the results of (B.5) and (B.6) into (B.4) we obtain the result stated in (B.1).

□

School of Computer Science, Carleton University
Bibliography of Technical Reports

SCS-TR-136	**Optimal Visibility Algorithms for Binary Images on the Hypercube** Frank Dehne, Quoc T. Pham and Ivan Stojmenovic, April 1988.
SCS-TR-137	**An Efficient Computational Geometry Method for Detecting Dotted Lines in Noisy Images** F. Dehne and L. Ficocelli, May 1988.
SCS-TR-138	**On Generating Random Permutations with Arbitrary Distributions** B. J. Oommen and D.T.H. Ng, June 1988.
SCS-TR-139	**The Theory and Application of Uni-Dimensional Random Races With Probabilistic Handicaps** D.T.H. Ng, B.J. Oommen and E.R. Hansen, June 1988.
SCS-TR-140	**Computing the Configuration Space of a Robot on a Mesh-of-Processors** F. Dehne, A.-L. Hassenklover and J.-R. Sack, June 1988.
SCS-TR-141	**Graphically Defining Simulation Models of Concurrent Systems** H. Glenn Brauen and John Neilson, September 1988
SCS-TR-142	**An Algorithm for Distributed Mutual Exclusion on Arbitrary Networks** H. Glenn Brauen and John E. Neilson, September 1988
	SCS-TR-143 to 146 are unavailable.
SCS-TR-147	**On Transparently Modifying Users' Query Distributions** B.J. Oommen and D.T.H. Ng, November 1988
SCS-TR-148	**An O(N Log N) Algorithm for Computing a Link Center in a Simple Polygon** H.N. Djidjev, A. Lingas and J.-R. Sack, July 1988 Available in STACS 89, 6th Annual Symposium on Theoretical Aspects of Computer Science, Paderborn, FRG, February 16-18, 1989, Lecture Notes in Computer Science, Springer-Verlag No. 349
SCS-TR-149	**Smallscript: A User Programmable Framework Based on Smalltalk and Postscript** Kevin Haaland and Dave Thomas, November 1988
SCS-TR-150	**A General Design Methodology for Dictionary Machines** Frank Dehne and Nicola Santoro, February 1989
SCS-TR-151	**On Doubly Linked List ReOrganizing Heuristics** D.T.H. Ng and B. John Oommen, February 1989
SCS-TR-152	**Implementing Data Structures on a Hypercube Multiprocessor, and Applications in Parallel Computational Geometry** Frank Dehne and Andrew Rau-Chaplin, March 1989
SCS-TR-153	**The Use of Chi-Squared Statistics in Determining Dependence Trees** R.S. Valiveti and B.J. Oommen, March 1989
SCS-TR-154	**Ideal List Organization for Stationary Environments** B. John Oommen and David T.H. Ng, March 1989
SCS-TR-155	**Hot-Spot Contention in Binary Hypercube Networks** Sivarama P. Dandamudi and Derek L. Eager, April 89
SCS-TR-156	**Some Issues in Hierarchical Interconnection Network Design** Sivarama P. Dandamudi and Derek L. Eager, April 1989
SCS-TR-157	**Discretized Pursuit Linear Reward-Inaction Automata** B.J. Oommen and Joseph K. Lanctot, April 1989
SCS-TR-158 (revised)	**Parallel Fractional Cascading on a Hypercube Multiprocessor** Frank Dehne, Afonso Ferreira and Andrew Rau-Chaplin, May 1989 (Revised April 1990)

SCS-TR-159	**Epsilon-Optimal Stubborn Learning Mechanisms** J.P.R. Christensen and B.J. Oommen, June 1989
SCS-TR-160	**Disassembling Two-Dimensional Composite Parts Via Translations** Doron Nussbaum and Jörg-R. Sack, June 1989
SCS-TR-161 (revised)	**Recognizing Sources of Random Strings** R.S. Valiveti and B.J. Oommen, January 1990 Revised version of SCS-TR-161 "On the Data Analysis of Random Permutations and its Application to Source Recognition", published June 1989
SCS-TR-162	**An Adaptive Learning Solution to the Keyboard Optimization Problem** B.J. Oommen, R.S. Valiveti and J. Zgierski, October 1989
SCS-TR-163	**Finding a Central Link Segment of a Simple Polygon in O(N Log N) Time** L.G. Alexandrov, H.N. Djidjev, J.-R. Sack, October 1989
SCS-TR-164	**A Survey of Algorithms for Handling Permutation Groups** M.D. Atkinson, January 1990
SCS-TR-165	**Key Exchange Using Chebychev Polynomials** M.D. Atkinson and Vincenzo Acciaro, January 1990
SCS-TR-166	**Efficient Concurrency Control Protocols for B-tree Indexes** Ekow J. Otoo, January 1990
SCS-TR-167	**A Hierarchical Stochastic Automaton Solution to the Object Partitioning Problem** B.J. Oommen, January 1990
SCS-TR-168	**Adaptive List Organizing for Non-stationary Query Distributions. Part I: The Move-to-Front Rule** R.S. Valiveti and B.J. Oommen, January 1990
SCS-TR-169	**Trade-Offs in Non-Reversing Diameter** Hans L. Bodlaender, Gerard Tel and Nicola Santoro, February 1990
SCS-TR-170	**A Massively Parallel Knowledge-Base Server using a Hypercube Multiprocessor** Frank Dehne, Afonso Ferreira and Andrew Rau-Chaplin, April 1990
SCS-TR-171	**Parallel Processing of Quad Trees on the Hypercube (and PRAM)** Frank Dehne, Afonso Ferreira and Andrew Rau-Chaplin, April 1990
SCS-TR-172	**A Note on the Load Balancing Problem for Coarse Grained Hypercube Dictionary Machines** Frank Dehne and Michel Gastaldo, May 1990
SCS-TR-173	**Self-Organizing Doubly-Linked Lists** R.S. Valiveti and B.J. Oommen, May 1990
SCS-TR-174	**A Presortedness Metric for Ensembles of Data Sequences** R.S. Valiveti and B.J. Oommen, May 1990
SCS-TR-175	**Separation of Graphs of Bounded Genus** Ljudmil G. Aleksandrov and Hristo N. Djidjev, May 1990
SCS-TR-176	**Edge Separators of Planar and Outerplanar Graphs with Applications** Krzystof Diks, Hristo N. Djidjev, Ondrej Sykora and Imrich Vrto, May 1990
SCS-TR-177	**Representing Partial Orders by Polygons and Circles in the Plane** Jeffrey B. Sidney and Stuart J. Sidney, July 1990
SCS-TR-178	**Determining Stochastic Dependence for Normally Distributed Vectors Using the Chi-squared Metric** R.S. Valivetei and B.J. Oommen, July1990