Journal of Computer Science 3 (5): 259-265, 2007
ISSN 1549-3636

A Bayesian Networks in Intrusion Detection Systems

M. Mehdi, S. Zair, A. Anou and M. Bensebti
Electronics Department, University of Blida, Algeria

Abstract: Intrusion detection systems (IDSs) have been widely used to overcome security threats in computer networks. Anomaly-based approaches have the advantage of being able to detect previously unknown attacks, but they suffer from the difficulty of building robust models of acceptable behaviour which may result in a large number of false alarms caused by incorrect classification of events in current systems. We propose a new approach of an anomaly Intrusion detection system (IDS). It consists of building a reference behaviour model and the use of a Bayesian classification procedure associated to unsupervised learning algorithm to evaluate the deviation between current and reference behaviour. Continuous re-estimation of model parameters allows for real time operation. The use of recursive Log-likelihood and entropy estimation as a measure for monitoring model degradation related with behavior changes and the associated model update show that the accuracy of the event classification process is significantly improved using our proposed approach for reducing the missing-alarm.

Key words: Computer network, Security, Intrusion detection, Probabilistic reasoning, Bayesian learning, Real-time.

INTRODUCTION

Intrusion detection can be defined as the process of identifying malicious behavior that targets a network and its resources. Intrusion detection systems have traditionally been classified as either misuse-based or anomaly-based. Systems that use misuse-based techniques contain a number of attack descriptions, or 'signatures', that are matched against a stream of audit data looking for evidence of the modeled attacks. The audit data can be gathered from the network[1], from the operating system, or from application log files. Signature-based systems have the advantage that they usually generate few false positives (i.e., incorrectly flagging an event as malicious when it is legitimate). Unfortunately, they can only detect those attacks that have been previously specified. That is, they cannot detect intrusions for which they do not have a predefined signature Anomaly-based techniques follow an approach that is complementary with respect to misuse detection. These approaches rely on models, or profiles, of the normal behaviour of users, applications and network traffic. Deviations from the established models are interpreted as attacks. Anomaly detection systems have the advantage that they are able to identify previously unknown attacks. By defining an expected, normal state, any abnormal behavior can be detected, whether it is part of the threat model or not. This capability should make anomaly-based systems a preferred choice. However, the advantage of being able to detect previously unknown attacks is usually paid for in terms of a large number of false positives. This can make the system unusable by flooding and eventually desensitizing the system administrator with large numbers of incorrect alerts.

We have identified two main problems that contribute to the large number of false positives. First, the decision whether an event should be classified as anomalous or as normal is made in a simplistic way. Anomaly detection systems usually contain a collection of models that evaluate different features of an event. These models return an anomaly score or a probability value that reflects the 'normality' of this event according to their current profiles. However, the system is faced with the task of aggregating the different model outputs into a single, final result. The second problem of anomaly-based systems is that they cannot distinguish between anomalous behavior caused by unusual but legitimate actions and activity that is the manifestation of an attack.

This leads to the situation where any deviation from normal behavior is reported as suspicious, ignoring

Corresponding Author:: MEHDI Merouane, Electronics Department, University of Blida, Algeria

potential additional information that might suggest otherwise[2]. Such additional information can be external to the system, received from system health monitors (e.g., CPU utilization, memory usage, process status) or other intrusion detection sensors. Consider the example of an IDS that monitors a web server by analyzing the system calls that the server process invokes. A sudden jump in CPU utilization and a continuous increase of the memory allocated by the server process can corroborate the belief that a certain system call contains traces of a denial-of-service attack [3]. Additional information can also be directly related to the models, such as the confidence in a model output. Depending on the site-specific structure of input events, certain features might not be suitable to distinguish between legitimate and malicious activity.

In such a case, the confidence in the output of the model based on these features should be reduced. In this paper, we propose a new model for intrusion detection following the anomaly detection approach. We are especially interested in information systems that are simultaneously submitted to different behavior profiles. Typical examples are multi-user applications and computer systems or networks carrying different communication protocols. In such cases, audit data reflecting actions or system states associated with each system behavior profile usually can not be separated a priori. Moreover, it is sometimes even impossible to know how many profiles are present in the system behavior. The proposed IDS model is intended to deal with both situations.

Anomaly IDS design consists of three main steps. First, it is necessary to build a reference behavior model for the monitored system. In our case, this reference behavior should be modeled from observed audit data describing the use of the system by a representative set of legitimate, non-malicious entities.

The aim is to model different entities profiles that could not be separated a priori by a learning procedure. A parametrical mixture model [4] is used to construct a Bayesian classification procedure based on the observations and leads to the system behavior model. Unsupervised learning is accomplished by fitting the mixture model parameters by the expectation-maximization (EM) algorithm [4, 5]. As the model order may also be unknown, a minimum entropy criterion is introduced to allow model order estimation [6]. The next step consists in evaluating audit data related to new system activities to detect deviations between the current and the reference behaviors. New observed data is compared to the reference model by means of both a Bayesian classification and cluster pertinence evaluations. As behavior can change, the behavior model should be updated during IDS operation. This is the last step. In our design, the model update is done by re-estimation of model parameters given in the new data presented to the system.

The design of the learning, detection and update phases using Bayesian techniques is the first contribution of the paper, the second lies in the discussion of real-time capabilities of the proposed algorithms, especially in the detection and update phase. Adaptations of the detection algorithm to the case of Gaussian mixture models are proposed, resulting in a linear complexity for the detection algorithm. Recursive parameter estimation is also proposed, as a possible alternative to real-time model update.

IDS Model (Bayesian classification):
The idea is to build a behavior model that takes into account multiple use profiles and allows for a Bayesian classification of data as part of the detection algorithm. A reference audit data set representing the normal system behavior is used to create the model with a learning procedure.

Before starting to describe the model, we should note that audit data must be mapped into random variables. Mapping audit data generated by the system to random variables, both during extraction of reference data in system behavior modeling and system usage, is out of the scope of this discussion. Hereafter, we admit that audit data can be represented by a set of realizations of a continuous random vector y, whose probability distribution function (PDF) will have to be modeled.

Mixture Model and EM-Algorithm:
The PDF of the (d-dimensional) random vector y, for which realizations are mapped from the audit data domain, are represented by a parametrical mixture model [4, 5]. In such models, the realizations of y are regarded as being trials of one of the K simple models designed by a kernel probability function, with each kernel function representing the model of a use profile. Realizations from y are not clustered, e.g. the profile of each realization y_i is not observable. The mixture model fundamental expression, giving the probability of y_i, can be formally expressed as:

$$p(y_i \mid \theta_1,...,\theta_K) = \sum_{k=1}^{K} g_k(y_i,\theta_k)\, p(z_k) \qquad (1)$$

where: $P(z_k)$ denotes the prior probability that a data point is generated by mixture component k, y_i is the i-th

observed data; z is the hidden vector that indicates which source (profile) the data comes from (e.g. $z_k = 1$ if data comes from cluster k and z = 0, otherwise); g_k are kernel distribution functions with respective parameters θ_k, each of them modeling one of the use profiles; K is the model order corresponding to the number of sources being modeled.

The unknown parameters in the model (Eq. (1)) are the set of cluster probabilities $p(z_k)$ and the parameters of kernel distribution functions of each cluster θ_k, represented by

$$\mu = \left[p(z_1), p(z_2),..., p(z_k), \theta_1, \theta_2, .., \theta_k\right] \quad (2)$$

The mixture model represented by (1) has been increasingly used to model the distribution of a wide variety of supposed random phenomena [7]. An iterative algorithm of optimizing the unknown vector μ by a maximum likelihood (ML) criterion has been defined and is also called the expectation-maximization (EM) algorithm [5, 6].

Let:

$$Y = \left[y_1, y_2, ..., y_m\right]^T \quad (3)$$

Where the subscripts 1, 2, k denote an observed m-dimensional realization vector of y (that has to be modeled). Y is regarded as the reference data containing representative normal behavior information and is used to fit μ using the EM algorithm. This algorithm permits both log-likelihood and model parameter estimation to be done in an iterative manner. The recursive process should be repeated until variation in the estimated log-likelihood between two consecutive iterations becomes small, indicating that the algorithm has converged to a (local) maximum of the log-likelihood data function, given that the realization probabilities are expressed as in Eq. (1). As the log-likelihood function evaluation at the point represented by the parameter fitted by the EM-algorithm is not guaranteed to be a global maximum, a finite number of random initializations of the parameters are realized and the EM-algorithm is executed at different times. The results (parameter estimation) corresponding to a maximum log-likelihood evaluation in all executions are kept as the optimal model parameters and are used during detection phase.

A detailed discussion of the EM-algorithm is out of the scope of this paper, as it has already been extensively discussed in the literature. The reader is asked to refer to [5, 6] for a more general description of the EM-algorithm.

In the particular case of Gaussian mixture models (GMM), e.g. mixture model with Gaussian kernel functions, which is used in our experiments presented further, the Eq. (1) should be rewritten replacing the general distributions (g_k) by the normal distribution (represented by ϕ) and the distribution parameters θk by the mean vector (μ_k) and covariance matrix (R_k), as stated at Eq. (4), where the probability $p(z_k)$ are also replaced by the weighting factor w_k, for notation simplicity.

$$p(y_i) = \sum_{k=1}^{K} w_k \phi(y_i, \mu_k, R_k) \quad (4)$$

For completeness, we provide the EM recursion equations (Eq. (5)-(6)) for the Gaussian mixture models:

$$p(k \mid y_i) = \frac{w_k^i \phi(y_i, \mu_k^i, R_k^i)}{\sum_{k'=1}^{K} w_{k'}^i \phi(y_i, \mu_{k'}^i, R_{k'}^i)} \quad (5)$$

$$w_k^{i+1} = \sum_{i=1}^{n} p(k \mid y_i)/n \quad (6)$$

$$\mu_k^{i+1} = \sum_{i=1}^{n} p(k \mid y_i) y_i \Big/ \sum_{i=1}^{n} p(k \mid y_i) \quad (7)$$

$$R_k^{i+1} = \frac{\sum_{i=1}^{n} p(k \mid y_i)(y_i - \mu_k^{i+1})(y_i - \mu_k^{i+1})^T}{\sum_{i=1}^{n} p(k \mid y_i)} \quad (8)$$

1.1. Entropy-Based Estimation of Model Order K

For the purpose of the EM-algorithm, the model order K (which corresponds to the number of partitions or data sources, when using parametric mixture models for partitioning data) must be provided. Since the number of partitions is not known a priori, it is useful to be able to estimate the most probable number of partitions, as well.

Our objective is to build an "ideal partitioning" estimation for K, which should be regarded as having the posterior probability $p(k \mid y_i)$ (Eq. (5) in the GMM case) close to unity for one value of k and close to zero for all the others, for each realization.

As described in [7], this ideal partitioning should be obtained by minimizing Shannon entropy given observed data, which can be evaluated for each observation by Eq. (9):

$$H_K = -\sum_{k=1}^{K} p(k \mid y_i) \log(p(k \mid y_i)) \quad (9)$$

The expected value of this entropy is evaluated taking the mean of H_K over all observed data (Eq. (10)):

$$E^*(H_K) = \frac{-\sum_{i=1}^{n}\sum_{k=1}^{K} p(k \mid y_i) \log(p(k \mid y_i))}{n} \quad (10)$$

Where: E* denotes an expectation estimator and H_K is the measure in question.

We proceed by fitting K_{max} models with different

order (K = 1, 2,...K_{max}) and we evaluate the expected entropy (13) for each case. The resulting model in a minimum of this measure will be considered the optimum model.

The complete algorithm of the learning phase, used to obtain a mixture model fitted with the EM-algorithm and with optimal model order (K) estimation can be summarised as follows:

EM-Algorithm with Model Order Estimation

K = 0, H_{opt} = 0, K_{opt} = 1.
K = K+1.
Fit the K-order model to data using the EM- Algorithm (Eqs. 2-7).
Calculate expected value of H_K (Eq. (7)).
If $H_K < H_{opt}$ then $H_{opt} = H_K$; $K_{opt} = K$; and $\mu = \mu_{opt}$.
If $K < K_{max}$, then repeat (2).
Update actual model order K with optimal model order: $K = K_{opt}$.
Update actual model parameters μ with optimal model parameters μ_{opt}.

Anomaly Detection: During detection, the behavior model has been already fitted and is available for finding inferences in a new data presented to the system. The aim is to define some penalty λ, which varies from 0 to 1 (e.g. $0 \le \lambda \le 1$), indicating the degree of normality concerning this realization from certainly abnormal ($\lambda = 0$) to a certainly normal ($\lambda = 1$) behavior.

Many different approaches for defining such criteria from the behavior statistical model represented by Eq. (1) are possible. We have defined a detection procedure formed by two basic steps: a (Bayesian) classification inference and a cluster pertinence inference[9].

The classification inference is straightforward for parametrical mixture models and consists of evaluation of the posterior cluster probabilities conditioned to new data y',

$$p(k \mid y'), \quad for \quad k = (1,2,....,K) \tag{11}$$

Cluster pertinence inference is more complex. As all the kernel distributions used in our model have a continuous nature, considering data posterior probabilities conditioned to cluster probability, p(y'|k), by simple evaluation of the cluster probability density function is meaningless. A more realistic approach consists in evaluating the probability of new data being contained in some pertinence interval (Π_k), defined as a function of cluster distribution parameters (μ_k and R_k, for instance) and the observation y', which should be formally expressed as follows (Eq. (12)):

$$p(y' \in \Pi_k \mid k) = \int g_k(y, \theta_k) d\Pi_k \tag{12}$$

Such probability should, indeed, look like some kind of cumulative distribution function, if we define Π_k as stated in Eq. (13), below [10]:

$$\Pi_k = \left| y \in \Re^d \left| \frac{\| y - \mu_k \|^2}{\| R_k \|} \ge \gamma^2 \right. \right| \tag{13}$$

Where $\| \ \|^2$ and $| \ |$ denote given types of norm operators and γ is a constant that should depend on y'. Finally, detection penalty should be defined as Eq. (14):

$$\lambda(y') = \sum_{k=1}^{K} p(k \mid y') p(y' \in \Pi_k \mid k) \tag{14}$$

Model Upgrading: The behavior model should be updated to avoid the raising of erroneous alerts (false positives). Updating should also be regarded as actualization of smooth changes in system behavior, as the basic model should become invalid or incomplete in case of expressive changes.

In our approach, we simply update the estimation of model parameters. Thus, updating is done in the cluster probabilities and in the kernel parameters. Usual estimators can be used for continuous estimation of these model statistics [10]. Note that both log-likelihood and entropy should also be estimated and compared with previous values (e.g. log-likelihood and entropy obtained after learning phase), as it could give an idea about the "goodness" of the new model when compared to the reference one.

Real-Time Capabilities: The learning procedure in the reference behavior model construction is usually executed off-line. Computation complexity constraints are not strong at this stage. However, it is usually desirable to have detection and update phases being executed continuously. Thus, the algorithms for detection and update should be designed for real-time. In this section, we show how detection and updating algorithms presented above should be adapted for real-time execution**.**

Although a formal performance evaluation of the proposed real-time algorithms are still in progress, we regard them as having a linear complexity, both with respect to the number of events (new data) presented to be analyzed by the system and with respect to the model order.

The IDS is designed to be implemented as software for execution in a conventional PC platform, without

need of any special hardware for enhancing computational capacity. Thus, for real-time evaluation, the processing power available for the IDS execution should be comparable with those of a standard PC. As a preliminary reference, IDS in use-intensive systems should deal with thousands or even millions of new events per second.

Real-time detection algorithm with Gaussian Mixture Models is as follows. Eq. (12) can not be usually evaluated analytically. A general solution should use numerical evaluation that can be prohibitive for higher dimensions. Besides, numerical evaluation is computation-intensive even in the one-dimensional or two-dimensional cases, making real-time execution difficult and even impossible.

Although Eq. (12) would be difficult for arbitrary kernel functions g_k, a computational-efficient algorithm for evaluating this integral equation can be established for the particular case of Gaussian distribution. These algorithms, which are discussed in this section, have been successfully used in the experiments, where we are essentially dealing with Gaussian mixture models.

Whenever GMM is being used, evaluation of the Eq. (12) can be done by convenient choice of the undefined elements on Eq. (13).

The idea is to define Π_k as the complementary space of the isodensity ellipsoid (in $\Re_d$), whose boundary contains y' and is centered in μ_k. This means that Π_k is bounded internally by a d-dimensional ellipsoidal surface formed by all points having the same density as y'

$$(e.g.\quad \phi(y,\mu_k,R_k)=\phi(y',\mu_k,R_k)) \tag{15}$$

Thus, rewriting of Eq.(13) gives Equation (16):

$$\Pi_k=\left|\begin{array}{l} y\in\Re^d \\ \sum_{\alpha\beta}(y_\alpha-\mu_\alpha)\left[R_k^{-1}\right]_{\alpha\beta}(y_\beta-\mu_\beta)\geq\gamma^2 \end{array}\right| \tag{16}$$

Where: $y=(y_1,y_2,...y_d)^T$; $\mu=(\mu_1,\mu_2,...\mu_d)^T$;

$\left[R_k^{-1}\right]_{\alpha\beta}$ is the element at α-th line and β-th column of the inverse covariance matrix, and γ is done by (Eq. (17)):

$$\gamma^2=\sum_{\alpha\beta}(y'_\alpha-\mu_\alpha)\left[R_k^{-1}\right]_{\alpha\beta}(y_\beta-\mu_\beta) \tag{17}$$

This strategy is illustrated for one and two dimensional spaces as showed in Fig 1 and 2, respectively. The latter was taken from a bivariate Gaussian distribution, with diagonal covariance matrix.

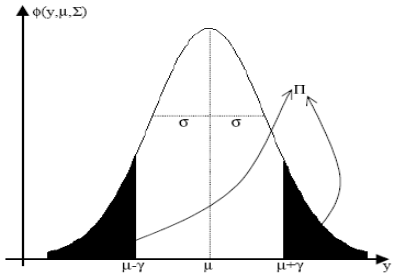

Fig. 1: Π for cluster with one-dimensional Gaussian distribution

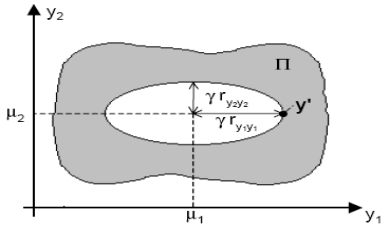

Fig. 2: Π for a cluster with bivariate Gaussian distribution with diagonal covariance matrix.

This procedure can be used even in the case of multivariate Gaussian distributions with unrestricted covariance matrix, as it is always possible to find a linear transformation that maps any multivariate Gaussian distribution in a equivalent new non correlated multivariate Gaussian distribution with same value for γ as in the former distribution [8].

As observed data can belong to a multidimensional space ($\Re$n), a generalized distance γ', defined in Eq. (18), is introduced. This leads to a normalization of the probabilities expressed by Eq.(12) in data models belonging to different dimensional spaces, allowing computation of probabilities to be reduced to the one dimensional space, which can be executed by a simple lookup table procedure, making computational complexity feasible in real time.

$$\gamma'=\gamma/\sqrt{d} \tag{18}$$

RESULTS AND DISCUSSION

The proposed model was implemented and evaluated with artificial data. Figures 3 and 4 present results for a model trained from data extracted from 2 independent and well-separated Gaussian sources. Model order (K) has been estimated by evaluation of minimum entropy for models with K varying from 1 to 3. Estimated entropy for each K and optimal model parameters are shown as well. Figures 3 and 4 also present the detection penalty evaluated for a new occurrence representing, respectively, a clearly normal behavior and a clearly anomalous behavior.

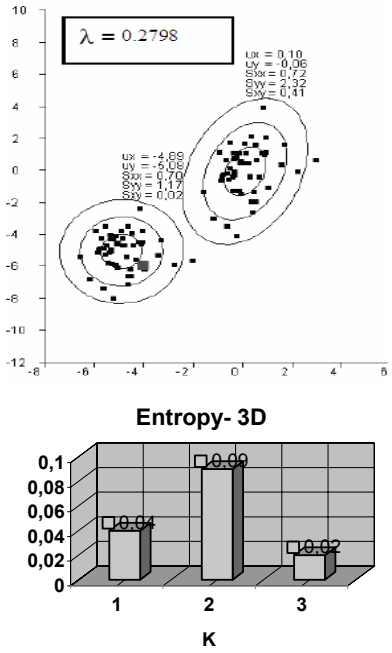

Fig. 3: 3-cluster behavior model and normal behavior recognition.

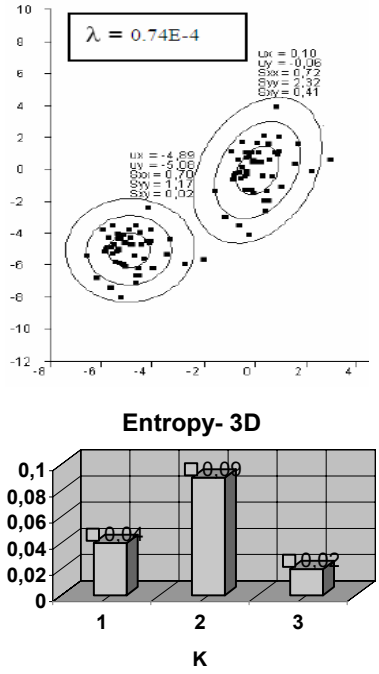

Fig. 4: 3-cluster behavior model and abnormal behavior detection.

Training data consists of all realizations of the same operation in all connections logged during the reference data extraction. The use of recursive log-likelihood and entropy estimation as a measure for monitoring model degradation related with behavior changes and the associated model update.

In our preliminary experiments, the proposed IDS model has been tested in a standard PC platform.

Execution of detection and updating algorithms were verified for data rates of 500 requests per second in a third order model (k=3).

Conclusion

This probabilistic approach is a tentative research on the possibility of applying Bayesian unsupervised learning in the detection of network intrusions. Based on unsupervised learning algorithms (Bayesian classification), some novel detection methods are proposed showing a very high detection rate with a reasonable true positive rate as results, much better than the classic method. By training with unsupervised learning algorithms, namely, Bayesian classification procedure, the log analyzer performs well in discovering the inherent nature of the dataset, clustering similar instances into the same classes.

We have presented a new anomaly IDS design using a parametric mixture model for behavior modeling and Bayesian based detection. Continuous model update is accomplished by model parameter re-estimation. Algorithms for detection and update phases are designed for real-time operations. Preliminary experimentations show that proposed algorithms have some limitations such as that the kernel distributions are used to model numerical data with continuous and unbounded nature, the Gaussian parametrical model may not be suitable for complex data and that the use of mixed models assumes statistical independence between trials, which can be restrictive in some cases. Despite these drawbacks the system presents real-time feasibility with no special hardware requirement. Moreover, it is being extended to detect security violations in a heterogeneous networked environment. The scalability, performance and fault tolerance can be improved when mobile agents perform distributed detection and do not need a central location where data is gathered. For instance, for wireless networks such as Mobile Ad hoc Networks are greatly prone to security threats. Since intrusion to the transmission support is relatively easy, compared to fixed networks, the use of such IDS is greatly recommended.

REFERENCES

1. H. Debar, M. Dacier and A. Wespi, 1999. A Revised Taxonomy for Intrusion-Detection Systems, IBM Research Report.
2. C. Krügel and T. Toth, 2002. Flexible, mobile agent based intrusion detection for dynamic networks, European Wireless, Florence Italy.
3. H. Luo, P. Zerfos, J. Kong, S. Lu and L. Zhang, 2004. Self Securing Ad Hoc Wireless Networks, proceeding of the 7th Inter. Symp. On computers and communications.
4. P. Cheeseman and J. Stutz, 1996. Bayesian classification (Auto Class): theory and results in Advances in Knowledge Discovery and Data Mining, edited by U.M. Fayyad et al., California: The AAAI Press, pp: 61-83.
5. A. P. Dempster, N. M. Laird and D. B. Rubin, 1977. Journal of the Royal Statistical Society B 39, pp: 1-38.
6. G. J. McLachlan, D. Peel, K. E. Basford and P. Adams, 1999. Journal of Statistical Software 04.
7. S. J. Roberts, R. Everson and I. Rezek, 1999. Pattern Recognition, 33:5, pp: 833-839.
8. Z. Marrakchi, 2002. Détection d'Intrusion Comportementale dans les Systèmes à Objets Répartis: Modélisation des Séquences de Requêtes et de la Répartition de leurs Paramètres, PhD Thesis, Université de Rennes I.
9. Z. Marrakchi, L. Mé, B. Vivinis and B. Morin, 2000. Flexible Intrusion-Detection Using Variable-Length Behavior Modeling in Distributed Environment: Application to CORBA Objects, Proceedings of 3rd International Workshop on the Recent Advances in Intrusion Detection, pp: 130-144.
10. R. A. Johnson, D.A. Wichern and D. W. Wichern, 1998. Applied Multivariate Statistical Analysis – 4th Edition, Prentice-Hall, pp: 198-207.