NPS-OR-16-003

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

A SCALE-INDEPENDENT, NOISE-RESISTANT DISSIMILARITY FOR TREE-BASED CLUSTERING OF MIXED DATA

by

Samuel E. Buttrey
Lyn R. Whitaker

April 19, 2016

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
03-25-2016	Technical Report	09-01-2015 to 03-25-2016

4. TITLE AND SUBTITLE

A SCALE-INDEPENDENT, NOISE-RESISTANT DISSIMILARITY FOR TREE-BASED CLUSTERING OF MIXED DATA

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Samuel E. Buttrey, Lyn R. Whitaker

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Postgraduate School
Monterey, CA 93943

8. PERFORMING ORGANIZATION REPORT NUMBER

NPS-OR-16-003

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

None

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited

13. SUPPLEMENTARY NOTES

The views expressed in this document are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol Number: N/A.

14. ABSTRACT

Clustering techniques divide observations into groups. Current techniques usually rely on measurements of dissimilarities between pairs of observations, between pairs of clusters, and between an observation and a cluster. For numeric variables, these dissimilarity measurements often depend on the scaling of the variables, are changed by monotonic transformations, and do not provide for selection of "important" variables. In our scheme, we fit a set of regression or classification trees with each variable acting in turn as the "response" variable. Points are "close" to one another if they tend to appear in the same leaves of these trees. Trees with poor predictive power are discarded. Therefore, "noise" variables will often appear in none of the trees and have no effect on the clustering. Because our technique uses trees, the dissimilarities are unaffected by linear transformations of the numeric variables and resistant to monotonic ones and to outliers. Categorical variables are included automatically and missing values handled in a natural way. We demonstrate the performance of this technique by using these dissimilarities to cluster some well-known data sets to which noise has been added.

15. SUBJECT TERMS
inter-point distance, mixed data, clustering

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Samuel Buttrey
Unclassified	Unclassified	Unclassified	UU	30	19b. TELEPHONE NUMBER *(include area code)* 831-656-3035

NSN 7540-01-280-5500

Standard Form 298 (Rev. 8–98)
Prescribed by ANSI Std. Z39.18

THIS PAGE INTENTIONALLY LEFT BLANK

NAVAL POSTGRADUATE SCHOOL
Monterey, California 93943-5000

Ronald A. Route President	James H. Newman Provost

The report entitled "A Scale-Independent, Noise-Resistant Dissimilarity for Tree-Based Clustering of Mixed Data" was not funded by an outside agency.

Further distribution of all or part of this report is authorized.

This report was prepared by:

Samuel E. Buttrey	Lyn R. Whitaker

Reviewed by:

Patricia A. Jacobs, Chairman Department of Operations Research	Jeffrey D. Paduan Dean of Research

Released by:

THIS PAGE INTENTIONALLY LEFT BLANK

Executive Summary

Clustering techniques divide observations into groups. Current techniques usually rely on measurements of dissimilarities between pairs of observations, between pairs of clusters, and between an observation and a cluster. For numeric variables these dissimilarity measurements often depend on the scaling of the variables, are changed by monotonic transformations, and do not provide for selection of “important" variables. In our scheme we fit a set of regression or classification trees with each variable acting in turn as the “response" variable. Points are “close" to one another if they tend to appear in the same leaves of these trees. Trees with poor predictive power are discarded. Therefore, “noise" variables will often appear in none of the trees and have no effect on the clustering. Because our technique uses trees, the dissimilarities are unaffected by linear transformations of the numeric variables and resistant to monotonic ones and to outliers. Categorical variables are included automatically and missing values handled in a natural way. We demonstrate the performance of this technique by using these dissimilarities to cluster some well-known data sets to which noise has been added.

THIS PAGE INTENTIONALLY LEFT BLANK

1 Introduction

1.1 The Clustering Problem

In the usual clustering problem, we have p measurements on each of n observations, with the measurement for the i-th observation denoted by $\mathbf{x_i} = (x_{i1}, x_{i2}, \ldots, x_{ip})$. We will write $\mathbf{x_{[j]}}$ for the $n-$vector of measurements for variable j, so that $\mathbf{x_{[j]}} = (x_{1k}, x_{2k}, \ldots, x_{nk})$. The entire set of measurements will be denoted by $\mathbf{X}$.

The object of clustering is to separate the observations into groups – "clusters" – such that the observations within each cluster are similar in terms of their measurements, and that the clusters are different one from another. The number of clusters, an important parameter, will generally not be known. The literature is extensive. The seminal book of Hartigan [1] collected much of the theory and practice up until that time. The book by Kaufman and Rousseeuw [2] proposed a set of algorithms, many of which have been implemented in the R statistical environment [3]. Clusters are often modeled as arising from a mixture distribution; the more recent book by Mirkin [4] describes this approach (chapter 6) and gives a good overview of the theory being brought to bear on clustering.

1.2 Measuring Dissimilarity

A crucial part of most clustering algorithms is the measuring of the proximity between two items (where "item" can refer to an individual observation or to a cluster). Most clustering algorithms measure proximity by dissimilarities (or as similarities) between items. Some require that dissimilarities be distances. In many cases it makes sense to separate the dissimilarity measurement from the clustering algorithm, and with a couple of exceptions we will do that here. Treating choice of dissimilarity measurement and clustering algorithm separately is not, it should be said, universal. For example, the recent comprehensive article by Andreopoulous et. al [5] compares 40 clustering algorithms in the context of biomedical research, without specifically defining how these algorithms measure dissimilarity.

A dissimilarity measure can reasonably be expected to have the following qualities:

1. It should incorporate both numeric and categorical variables, including ordinal and asymmetric binary ones;
2. It should be insensitive to linear scalings of numeric variables (say, re-expression in

different units);

3. It should permit the incorporation of variable-specific weights, so that some variables can be made more influential in the dissimilarity measurement than others – in particular, it should permit some weights to be set to zero, if the corresponding variable is merely noise;
4. It should detect the common situation where two variables contain identical, or almost identical, information and prevent those variables from being double-counted in the dissimilarity, and, more generally, should adjust for correlation among variables;
5. It should be insensitive to extreme outliers in the data, either detecting them or at least keeping them from exerting undue influence on dissimilarites;
6. It should be able to operate in the presence of missing data; and
7. It should be straightforward to compute, even in large data sets.

The first item is one of the the prime motivations behind our effort. Dissimilarities for numeric data start with the ordinary Euclidean distance and are widely used. Similarly, substantial effort has gone into developing dissimilarities for purely categorical data. Some of this work is summarized in Boriah, Chandola and Kumar [6]. Of the many possible flavors of dissimilarities, an indicator of mismatch is often used when clustering categorical data. It should be noted that binary categoricals are substantially less daunting than those with many categories. A number of authors have sought to construct dissimilarities for "mixed" data – that is, data with both categorical and numerical variables. For example, Ahmad and Dey [7] extend the widely-used k-means algorithm to mixed data by treating numerical and categorical variables separately. McCane and Albert [8] describe measures for mixed data that adjust for inter-variable correlation (our point 4) above. As other authors often do, however, their algorithm assumes that all categories are equidistant from one another, an assumption which seems generally too strong.

Points 2 and 3 above overlap substantially. In many applications, scaling and weighting are both implemented with a specific multiplier. For our purposes, "scaling" is applied so as to counteract the effects of linear scaling – e.g. changes in units – whereas "weighting" aims to perform variable selection by down-weighting or omitting variables which should not contribute to the dissimilarity. Often scaling can be done automatically, as in the Gower dissimilarity we describe below. Weighting is more difficult. Sometimes this can be done with the help of subject-matter experts; other approaches seek to estimate weights so as

to optimize some criterion regarding, for example, cluster "quality." We will describe one such effort below.

Missing values are always an issue in real data. When these are rare, almost any reasonable approach can be applied. It is common to omit observations which have any missing entries. When missing values are widespread, though, this prescription can lead to the omission of large chunks of data. It is important that a dissimilarity measure be able to handle missing values gracefully.

The final point also deserves mention. A number of clustering algorithms begin by computing all $n(n-1)/2$ pairwise dissimilarities or distances among observations. In modern-day analysis, however, we might expect data sets with millions of rows, thousands of columns, and dozens or perhaps hundreds of clusters. Computing, storing, and using all pairwise distances is a computational burden best avoided. For this paper, our concern is focused on computing dissimilarities, rather than on the clustering mechanism, but it is important to be aware that clustering algorithms need to be able to operate on large data sets.

1.3 Organization

The rest of the paper proceeds as follows. In the next section, we describe our proposal for measuring dissimilarities between observations in mixed data. Our dissimilarities are insensitive to affine transformation and indeed resistant to non-linear monotonic ones. They also provide for automatic variable selection. Our approach handles missing data and outliers automatically and cleanly. The next section also describes two dissimilarities that will act as competitors: the Gower dissimilarity and random forest dissimilarity. Section 3 describes our implementation of our dissimilarity measures, the data sets that will be used as input to the dissimilarity measurements, and the clustering algorithms that are used once the dissimilarities are computed. We also describe how we measure the performance of any particular combination of dissimilarity and clustering algorithm. Section 4 presents results and, finally, Section 5 gives our conclusions and directions for future exploration.

2 Dissimilarities

In this section, we describe our proposals for measuring dissimilarities, first mentioned in Buttrey [9]. Our scheme involves using classification and regression trees, and so we

consider also, as a competitor, the random forest proximity of Breiman [10]. Finally, we describe the widely-used dissimilarity of Gower [11].

2.1 The TreeClust Dissimilarities

In our scheme, which we call treeClust, we capitalize on many of the advantages of classification and regression trees (Breiman et. al [12]) to produce a dissimilarity and, from that, a clustering that is oblivious to any affine scaling (and, indeed, resistant in some sense to any monotone transformation) of any or all of the variables. Categorical and numeric variables are combined and weighted in an obvious way, and it is possible to detect and omit large numbers of “noise" variables.

The central idea of our approach is this: two observations are similar if they tend to fall in the same leaves of classification or regression trees. A regression or classification tree needs a response variable, though, and in the clustering problem there is no such thing. So we create p trees, one for each variable where each variable in turn serves as the response variable, with the others acting as predictors. Trees corresponding to categorical variables will be classification trees and numeric variable trees will be regression trees. We use cross-validation to “prune" each tree to an “optimal" size. That is, we select the size for which the cross-validated error rate is minimized.

A tree built with a noise variable as the response will often have an optimal size of 1. Such a tree classifies every observation into the same leaf, and therefore contributes nothing to our dissimilarity computations. At the same time, noise variables will rarely be chosen as splitting variables in other trees. So some variables will never enter the clustering either as response or as predictor: those variables are ignored entirely. Of course, we hope that all noise variables, and no others, will be excluded in this way. A tree is unaffected by a linear scaling of the response, or by any monotonic transformation of a predictor, so the entire scheme is insensitive to linear scaling. A non-linear transformation of a variable affects only the tree built with that variable as a response, not any of the trees in which it acts as a predictor.

After the trees are built, a simple measure of dissimilarity between two observations is the number of trees in which those observations fall into different leaves. Let the label of the leaf of the t^{th} tree into which the i^{th} observation falls be denoted by $L_t(i)$. Then we measure

the dissimilarity between observations i and j by

$$d(i,j) = \sum_{t=1}^{p} \begin{cases} 0 & \text{if} \quad L_t(i) = L_t(j) \\ d^t(i,j) & \text{if} \quad L_t(i) \neq L_t(j) \end{cases}, \tag{1}$$

taking all $d^t(i,j) = 1$ initially to construct our first dissimilarity d_1. We note that d_1 and its extensions are not metric on the set of observations with the original variables. Two different observations i and j, not equal, can fall in the same leaf for every tree giving $d_1(i,j) = 0$. But d_1 is metric for the observations measured with the p categorical variables indicating leaf membership for the p trees.

The advantages of this approach include the advantages of trees generally. Their insentivity to affine transformation is automatically reflected in the d_1 dissimilarity. Trees are easy to build, even in the presence of missing values, and being local-type models they are resistant to outliers.

Furthermore, the p or fewer trees in any data set can be built independently. This allows a straightforward parallelization of many of the necessary computations.

As an extension, we also propose a second dissimilarity $d_2(i,j)$. This measurement takes into account the relative "strength" of the various trees. Each reponse variable starts with a quantification of its variability in the form of deviance, computed at the tree's root (that is, using all the response values). Deviance is measured by the sum of squares of deviations from the mean for a numeric response variable, and by the usual multinomial deviance for a categorical one. After the tree is built, we compute the sum of deviances in its leaves. A tree's strength can be measured by the ratio of the change in deviance between root and leaves to the deviance at the root. We denote this ratio, which will be between zero and 1, by q. A tree with a large q is presumably better able to help cluster individual observations. Two observations that land in different leaves of such a "good" tree might be judged to be more dissimilar than two observations landing in different leaves of a "bad" tree. In this approach, each tree gets a weight based on how big its q is compared to the largest q observed across all trees. That is, we set $d_2^t(i,j) = 0$ when observations i and j fall in the same leaf of tree t, and otherwise to $q_t / \max_k(q_k)$.

Figure 1 shows a hypothetical example of a tree of the sort that might be built in our scheme.

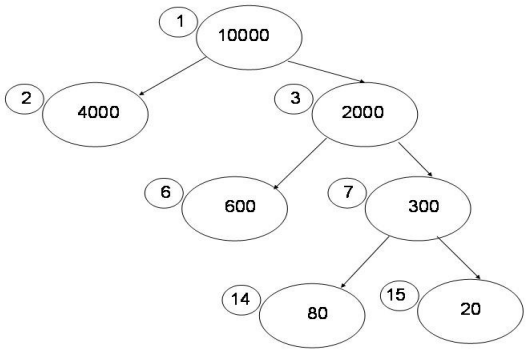

Figure 1: Example tree, showing node numbers (small circles) and deviances

Large ovals show the deviances of each node, while small circles indicate the leaf numbers. Then, under our first measure, observations i and j will have dissimilarity $d_1^t(i, j) = 0$ for this tree if they fall in the same leaf, and 1 otherwise. The second measure requires knowing the maximum q for any of the trees. Suppose that maximum is 0.80. The tree in Figure 1 exhibits a q of 0.47, that being the ratio of the root deviance of the sum of deviances in the leaves, 4,700, to the deviance at the root, 10,000. So the tree in Figure 1 would contribute 0.47/0.80 = 0.59 when two observations fall in different leaves, and 0 otherwise. The overall dissimilarity measure $d_2(i, j)$ is then the sum of contributions from each of the individual trees.

In a third dissimilarity, observations in the same leaf continue to have dissimilarity 0. The dissimilarity between two observations for a particular tree depends on how far apart they are. In Figure 1, two observations falling in leaves 14 and 15 seem much closer together than two falling in leaves 2 and 15. We imagine minimally pruning the tree until the two observations fall in the same leaf. The dissimilarity between two observations is then the ratio of the increase in deviance associated with the pruning to the original tree's decrease in deviance.

For example, observations in leaves 14 and 15 would be in the same leaf if those two leaves were pruned back to leaf 7. The tree resulting from that pruning operation would have deviance 4,900, so those two observations would have a dissimilarity of $(4,900 - 4,700)/(10,000 - 4,700) = 0.038$ for this tree. Observations in leaves 2 and 15 would only be in the same leaf if the tree were pruned back to the root, an operation which would

produce a tree with deviance 10000. So those two observations would have dissimilarity $(10,000-4,700)/(10,000-4,700) = 1.0$ for this tree, and the overall dissimilarity measure $d_3(i,j)$ is then the sum of the contributions from the set of trees. Our fourth dissimilarity uses the d_3 dissimilarities, together with the tree-specific weights described above for d_2. In our example, where the maximum q was 0.80, the weight for the tree in the figure was 0.59. Two observations maximally far apart on that tree contribute dissimilarity 0.59 for this tree, whereas observations in leaves 14 and 15 would receive a contribution of 0.59×0.038.

Missing values can cause problems with many algorthms. Trees are robust to missing values in the predictors; it is easy to devise schemes for use in tree construction and prediction for observations that are missing some predictors. For example, we might take advantage of surrogate splits for missing predictors. When missing values are present in the response, the usual approach is to omit those observations from the tree-construction phase. However, it is still possible to make predictions for every variable on the entire data set. In our examples, we have removed missing values for ease of comparison across methods, but our code handles data sets with missing values.

2.2 Random Forest and Other Tree-Based Methods

Random forests were introduced in Breiman [10] as a ensemble technique for regression and classification. However, the software manual (Breiman and Cutler [13]) notes that the set of trees produced by this technique provide a natural way to measure proximities between two observations, even those of mixed variable type. The data are augmented by simulated data whose marginal distributions match the data, but whose variables are independent. The random forest is built on the combined data where the response variable takes value 1 for the original observations and 0 for the simulated observations. Proximity is measured by counting the number of times two observations fall in the same leaves. Dividing this number by twice the number of trees and setting each observation's proxmity with itself to 1 yields a proxmity measure that Brieman recommends for clustering. In practice we subtract that measure from 1 to yield a dissimilarity. That dissimilarity was put to the test by Shi and Horvath [14], who note that the corresponding dissimilarity measure is, like our treeClust measure, invariant to affine transformation and robust with regard to outliers.

We note that other clustering approaches have been based on trees. Fisher [15] proposes us-

ing trees as part of a scheme to categorize items, an approach called "conceptual clustering." It appears to be specifically relevant to categorical predictors arranged in a natural hierarchy. Ooi [16] uses trees to partition the feature space as a step towards density estimation, which is related to clustering, and Smyth et. al [17] cluster using multivariate regression trees with principal component and factor scores. The latter two approaches are intended for numeric data.

2.3 Gower's Dissimilarity

Many of the issues associated with constructing dissimilarities are addressed by Gower's coefficient [11]. As implemented in the function `daisy()` in the `cluster` package [18] in the R statistical environment [3], the Gower dissimilarity for mixed data starts with a measurement of dissimilarity between two observations i and j for a single variable k. In this implementation, numeric distances are scaled by the variable's range, so that each numeric variable's contribution is between 0 and 1, and each categorical variable's contribution is exactly 0 or 1:

$$\begin{aligned} d_G^k(i,j) &= \frac{|x_{ik} - x_{jk}|}{\max \mathbf{x_{[k]}} - \min \mathbf{x_{[k]}}} && \text{when } \mathbf{x_{[k]}} \text{ is numeric,} \\ &= \left\{ \begin{array}{ll} 1 & \text{if } x_{ik} \neq x_{jk} \\ 0 & \text{if } x_{ik} = x_{jk} \end{array} \right\} && \text{when } \mathbf{x_{[k]}} \text{ is categorical.} \end{aligned}$$

Then the different contributions are combined in a weighted sum, with an adjustment for missing values and for asymmetric binary variables. Specifically, `daisy()` computes the weighted mean

$$d_G(i,j) = \frac{\sum_{k=1}^{p} w_k \delta_k(i,j) d_G^k(i,j)}{\sum_{k=1}^{p} w_k \delta_k(i,j)} \tag{2}$$

where, as before, the set of w_k serves to weight each variable separately. The quantites $\delta_k(i,j)$ serve two purposes. First, $\delta_k(i,j)$ is set to 0 if either x_{ik} or x_{jk} is missing; this ensures that the denominator of $d_G(i,j)$ is set correctly when variables are missing. Second, if variable $\mathbf{x_{[k]}}$ is specified to be an asymmetric binary one, then $\delta_k(i,j)$ gets the value 0 if $x_{ik} = x_{jk} = 0$. Across a set of asymmetric binary variables, two observations' mutual dissimilarity is given by the proportion of times they both have the value 1, among all the times either one has a 1.

2.4 Variable Selection

The weights in the Gower dissimilarity are often taken to be 1 for every variable, but a number of researchers have considered how weights might be automatically chosen in the clustering context. Friedman and Muelman [19] seek to optimize a criterion on aggregated sums of Euclidean distances within clusters by selecting weights and cluster memberships simultaneously. Witten and Tibshirani [20] propose another, simpler optimization that they have included in the R package `sparcle` [21]. We include two algorithms from this package in our comparison (see Section 3.4) to compare the relative variable selection strategies.

3 Data and Algorithms

3.1 Implementing the TreeClust Dissimilarities

Our approach is implemented in an R package called `treeClust` that can be downloaded from the CRAN repository, `cran.r-project.org`. Trees are built using the `rpart` package [22]. The user supplies a data set and an indicator that takes values 1 to 4, depending on whether dissimilarity d_1, d_2, d_3 or d_4 is preferred.

3.2 Data Sets

In this section, we describe the data sets on which we evaluate our algorithm. A number of measures are available by which the quality of a clustering solution can be evaluated, each of which inevitably has strengths and weaknesses. We have circumvented the problem of evalation to some extent by choosing data sets in which the "true" classifications are known. Then we measure the clustering solution's quality by Cramér's V, which is the usual χ^2 measure of association for the two-way table, scaled to produce a number between 0 and 1. Specifically, when that measure has value χ^2 from a table computed from n observations with r rows and c columns, we have

$$V = \sqrt{\frac{\chi^2}{n \min(r-1, c-1)}}, \tag{3}$$

although we note that in some references the square root is omitted.

Cramér's V will be small when the cluster labels do not follow class labels well, and close

to 1 when most clusters correspond to classes and *v*ice versa. The measure does depend on the true number of classes, though, and so is not something that will be useful to compare across different data sets.

Although the correct number of classes is clearly known in these cases, that information might not correspond directly to the "correct" number of clusters. For example, in an optical digit recognition task there should be at least one cluster of observations corresponding to each digit. However, handwritten 7's, for example, come in at least two styles (one with a horizontal line across the vertical staff, and one without); some 2's have a loop at the base and others do not, and so on. For purposes of evaluating the clustering algorithms, then, we computed Cramér's coefficient both when the algorithm was asked to find k clusters (where k is the number of distinct class labels) and also $2k$ clusters.

All of our data sets are available at the UC Irvine Machine Learning Repository (Bache and Lichman, [23]). We have chosen one in which all of the variables are numeric, one in which they are all categorical, and one with mixed variables.

3.2.1 Seeds

The seeds data set [24] concerns 210 wheat seeds from three varieties. There are seven predictors, all numeric, on quite different scales, with standard deviations ranging from 0.0006 to 8.5. The three varieties are represented by 70 examples each.

3.2.2 Credit Approval

The credit approval data set describes attributes of 690 borrowers. There are fifteen predictors, six numeric, four binary, and five that are categorical with more than two levels. We removed any observation with missing values; the final data set had 653 observations in two classes.

3.2.3 Splice

The final data set, more fully named "Molecular Biology (Splice-Junction Gene Sequences)" at the repository, describes junctions on sequences of DNA. The problem is to determine whether particular sites on the sequences represent junctions of one sort or another, or no junction at all. There are three classes, the third having about twice as many members as

the other two. Each variable is categorical, containing the letters A, C, T, or G in each observation. A small number of observations containing other letters are deleted, resulting in a data set with 3,175 rows and 60 columns. The junction, if there is one, is located "in the middle of" the window formed by the 60 measurements.

3.3 Adding Noise

For each data set we test the ability of our algorithm by adding fifteen and fifty variables of random noise to the original data. In order to present noise that "looks plausible," each of our noise variables consists of one of the variables from the original data, selected at random, with its values permuted. The data sets with 50 noise variables are constructed by adding 35 more noise variables to the data sets with 15 noise variables.

Table 1 summarizes the data sets used: "Rows" and "Cols" give the number of rows and columns in the data set (not counting the column giving the correct classification); "Type" is "Numeric," "Categorical" or "Mixed"; and "Classes" gives the true number of classes in the data (which may not, as noted, be the correct number of clusters).

Table 1: Data sets used

Name	Rows	Cols	Type	Classes
Seeds	210	8	Numeric	3
Seeds + 15 noise	210	23	Numeric	3
Seeds + 50 noise	210	58	Numeric	3
Credit	653	15	Mixed	2
Credit + 15 noise	653	40	Mixed	2
Credit + 50 noise	653	65	Mixed	2
Splice	3190	60	Categorical	3
Splice + 15 noise	3190	75	Categorical	3
Splice + 50 noise	3190	110	Categorical	3

3.4 Clustering Algorithms

Each of our data sets is used to produce six dissimilarity measures: the newly proposed d_1, d_2, d_3, and d_4 measures, the random forest (RF) dissimilarity, and that of Gower. Each dissimilarity measure was employed in two widely-used algorithms, "Pam" and "Agnes," which are described briefly below. In the all-numeric Seeds data, we also used k-means, Pam, and Agnes on both the original data and on the data when each column is scaled by its

range. This scaling is intended as an analog to Gower's dissimilarity. In addition we used the "Sparse Hierarchical" algorithm of Witten and Tibshirani [20] (see below) on every data set and the "Sparse K-means" algorithm from the same paper on the all-numeric data set. The algorithms used were:

K-means In this algorthm, the number of clusters k is specified in advance and an initial set of cluster centers chosen at random. Observations are assigned to the nearest cluster; cluster centers are updated; and the process repeats until convergence. The traditional k-means clustering as implemented in [3] can only accept data in the form of a numeric matrix or data frame, so we use it only on the all-numeric Seeds data set. Distance is Euclidean. No scaling is performed by the algorithm, so we run it twice, once with the original, unscaled data, and then again with the data where each variable has been scaled by its range. We allow 100 random starting points for each run and up to 500 iterations; other than that we use the function's default values. K-means has proven to be very useful in all-numeric data sets in which little scaling is needed, like, for example, optical digit recognition and natural computer vision tasks.

Pam The partitioning around medoids (Pam) algorithm is described in Kaufman and Rousseeuw [2]. It operates in a manner similar to that of k-means, but using medoids, a multidimensional analog to the median. It allows for pre-computed dissimilarities and is therefore suited to categorical or mixed, as well as numeric, data. We use the implementation in R's `cluster` library [18]. We use all of the algorthm's default values in our runs. As with k-means, we use Euclidean distance with both the original and scaled data for the Seeds data set. We also use Pam with the inter-point dissimilarity matrices computed by `daisy()` using the original data (that is, with Gower dissimilarities), with the random forest dissimilarities and with the treeClust dissimilarities.

Agnes Agnes is a hierarchical clustering method which, like Pam, is implemented in R's `cluster` library. This algorithm starts by computing all pairwise dissimilarities – as with Pam, the user can supply pre-computed ones – and progressively merges the two closest items. The algorithm requires the user to specify what it means for clusters to be close to one another; we used the default method, under which the dissimilarity between two clusters is the average of all the dissimilarites between members of one

group and members of the other. This algorithm is used just as Pam is: directly on the original and scaled data for the Seeds data, and with Gower, random forest, and treeClust dissimilarities.

Sparse K-means Sparse k-means is one of the algorithms focused on variable selection described in Witten and Tibshirani [20] (see Section 2.4). This algorithm has been implemented in the `sparcl` package for R. It operates like k-means, but optimizes over a set of weights, one for each column, using a lasso penalty to shrink the weights towards zero, thereby removing some columns from the distance computation. The penalty value can be provided by the user or selected by a permutation approach; we used the latter approach with default values. This program requires a numeric data matrix, since the variable selection takes place during the running of the algorithm, not in the computation of dissimilarity. Therefore, as with standard k-means, we use this algorithm only on the numeric Seeds data set. Since the algorithm imposes its own scaling, we use the original (unscaled) data.

Sparse Hierarchical Sparse Hierarchical is the other of the algorithms from Witten and Tibshirani [20]. This algorithm produces a hierarchical clustering like that of Agnes, but using only a subset of variables. The program will accept a numeric data set, but it also allows the user to specify pre-computed, component-wise dissimilarities in the form of an $n(n-1)/2 \times p$ numeric matrix, where the jth column specifies all $n(n-1)/2$ pairwise dissimilarities between observations on variable j. These pre-computed dissimilarities are required in the case of categorical or mixed data; we use the Gower dissimilarities for each variable. Clearly, this requirement will be computationally troublesome when n is very large, but our data sets are small enough to allow this algorithm to be run in every case. With the numeric data set, we pass the data directly, in its original unscaled form. As with the sparse k-means algorithm, a tuning parameter is chosen by a permutation algorithm using default settings when the data is numeric. For mixed or categorical data, the permutation is unavailable, so we manually set the tuning parameter to 1.5 and otherwise use default settings.

Table 2 shows the possible dissimilarity measurements (rows) and clustering (algorithms). An "N" shows a combination that is only possible with all-numeric data; "X" shows combinations that are possible with numeric, mixed or categorical data. The asterisk indicates that the sparse algorithms provide their own scaling.

3.5 Randomness

There are three sources of randomness in our results. First, some of the algorithms are inherently random, and can return different outputs for the same inputs. K-means and Pam find local optima of a particular objective function, relying on a randomized initialization. (For the former, as we have noted, we use multiple starting points so as to make finding the global optimum more likely.) Agnes is not random, but the sparse methods select a parameter through a permutation test.

A second source of randomness is in the computations of inter-point dissimilarities. For a particular set of input data, the Euclidean distance and Gower dissimilarity are not random. Our tree dissimilarities rely on cross-validation for pruning, so they are random, and so too are the Random Forest dissimilarities, which rely on simulated data.

Finally, we have added random noise to our data sets in order to examine the different approaches' success in variable selection. This last piece of randomness is not directly of value in comparing the approaches. So in our experiment we produced the data sets first and saved them. We then ran each algorithm on each data set, with k and $2k$ clusters, using the same starting data but a different random-number seed each time. Since the data remains unchanged from run to run, variability we see in the results must be due to variability in the distance computations or in the algorithms themselves.

4 Results

Tables 3, 4, and 5 show the average values of Cramér's coefficient, multiplied by 100 for readability, for 20 replications on each of the data sets, using k and $2k$ clusters, for each of

Table 2: Combinations of Measurements and Algorithms; "N" indicates numeric only; "X" indicates numeric or mixed; blanks show untested combinations; asterisks show that the sparse algorithms implement their own scaling.

Dissimilarity	K-Means	Pam	Agnes	Sparse KM	Sparse Hier
Euclidean (original)	N	N	N	N*	N*
Euclidean (scaled)	N	N	N		
Gower		X	X		X*
Random Forest		X	X		
TreeClust		X	X		

the clustering techniques considered. In table 3, we have abbreviated "Noise" to "Ns" and "Classes" to "C" to save space. The bold-face number in each row shows the highest value of the Cramér coefficient achieved in that row.

Table 3: Results for Seeds Data, Expressed as 100 × Cramér's Coefficient

		Orig			Scaled			Gower		Sparse	
Ns	C	KM	Ag	Pa	KM	Ag	Pa	Ag	Pa	KM	Hier
0	3	85.2	**87.2**	84.7	84.3	85.4	84.5	82.6	83.8	79.0	77.3
0	6	85.5	87.7	88.9	88.7	88.8	**89.6**	83.5	85.6	84.8	82.4
15	3	58.9	49.1	55.8	**85.1**	58.0	65.5	64.3	73.3	59.7	41.4
15	6	57.4	49.8	59.0	80.4	59.2	64.8	81.5	75.6	63.7	64.9
50	3	55.2	12.6	25.6	67.0	61.1	50.1	63.2	57.2	59.7	47.5
50	6	51.1	52.4	45.5	72.0	62.1	56.6	63.7	65.8	60.6	69.1

		RandFor		d_1		d_2		d_3		d_4	
Ns	C	Ag	Pa	Ag	Pa	Ag	Pa	Ag	Pa	Ag	Pa
0	3	84.1	84.6	76.0	85.0	76.3	84.4	68.9	76.6	68.4	76.6
0	6	87.6	87.0	82.6	83.5	83.7	83.3	82.4	85.3	82.4	85.0
15	3	72.0	58.2	72.6	81.7	66.9	80.1	76.6	76.9	69.0	76.6
15	6	77.1	68.7	83.6	83.2	83.6	83.1	81.0	82.2	81.1	**85.2**
50	3	51.2	32.5	73.5	80.2	67.4	**81.3**	72.2	76.9	67.7	76.6
50	6	60.3	41.5	83.8	83.4	83.7	83.0	80.4	80.3	80.7	**85.1**

We can see that in the all-numeric Seeds data, all of the approaches perform well on the easiest task (no noise, six clusters). Here, Pam on the scaled data produces the largest average V, whereas with three clusters, Agnes on the original data has the edge. Interestingly Agnes performs comparatively poorly when paired with the treeClust dissimilarity in the three-cluster condition, and the "more complicated" d_3 and d_4 perform worse than their simpler siblings. In the presence of moderate noise, all of the approaches except k-means on the scaled data, and the treeClust dissimilarities, show degradation of performance. In the presence of 50 noise variables, the treeClust dissimilarities show uniformly better performance than any of the competitors. There was very little variability across the 20 trials for most of the methods, with the exception of the random forest dissimilarities and, to a lesser extent, d_3 under Pam.

In the credit data (table 4), Pam performs best with no noise, as well as with moderate noise and four clusters (column Cl). In the presence of a lot of noise, though, its performance is badly degraded. The combination of Pam and the treeClust dissimilarities performs well

under all conditions, but Agnes does poorly in all of the two-cluster configurations. This is because that algorithm is more likely to create one cluster with only a few observations in it. The Sparse Hierarchical algorithm experiences the same problem.

Table 4: Results for Credit Data, Expressed as 100 × Cramér's Coefficient

		Gower		RandFor		Sparse
Name	Cl	Ag	Pa	Ag	Pa	Hier
Credit	2	3.3	**61.5**	26.5	37.0	0.4
Credit	4	7.5	**57.0**	39.8	45.4	8.6
Credit + 15	2	0.4	45.2	17.8	20.2	0.4
Credit + 15	4	7.5	**54.4**	28.8	32.6	5.8
Credit + 50	2	0.4	1.0	3.5	9.8	0.0
Credit + 50	4	9.7	32.5	13.1	17.8	5.8

		d_1		d_2		d_3		d_4	
Name	Cl	Ag	Pa	Ag	Pa	Ag	Pa	Ag	Pa
Credit	2	20.9	50.0	16.2	44.9	18.6	44.9	16.2	44.9
Credit	4	47.1	50.4	46.1	46.4	49.9	46.9	46.1	46.1
Credit + 15	2	19.8	**46.7**	16.2	44.9	16.4	45.4	16.2	44.9
Credit + 15	4	43.1	49.0	46.1	46.1	49.8	47.3	46.1	46.1
Credit + 50	2	11.3	**45.9**	16.2	44.9	18.5	45.3	16.2	44.9
Credit + 50	4	38.6	**46.9**	46.1	46.1	47.5	47.4	46.1	46.1

Table 5 gives the averages of 20 replications in the Splice data. In this example, the treeClust dissimilarity paired with Pam perform much better than any of their competitors. The weighted dissimilarities d_2 and d_4 outperform their unweighted counterparts for this data set, with the d_4 dissimilarity producing the largest values of average V in every case.

Figure 2 shows an example of the variability across the set of twenty trials, in this case when using the splice data with 50 noise variables and three clusters. There was no variability in the results for Sparse Hierarchical clustering, which uses the pre-computed Gower dissimilarity, nor for Pam and Agnes using the Gower. There was substantial variability with the random forest dissimilarities, about a low mean, and for the d_2 dissimilarities, about a higher mean.

Figure 3 compares the variable selection algorithms in treeClust and Sparse Hierarchical for one particular random number seed. The vertical scale gives the weights assigned to each tree (treeClust) or, for Sparse Hierarchical, to each variable (grey circles). Only six

Table 5: Results for Splice Data, Expressed as 100 × Cramér's Coefficient

		Gower		RandFor		Sparse
Name	Cl	Ag	Pa	Ag	Pa	Hier
Splice	3	34.3	18.7	7.8	9.6	6.2
Splice	6	36.9	30.3	13.5	14.0	8.0
Splice + 15	3	23.6	22.0	12.2	5.8	1.2
Splice + 15	6	29.8	28.1	21.1	9.7	3.2
Splice + 50	3	7.8	17.4	8.1	5.6	1.1
Splice + 50	6	13.3	26.5	14.9	9.1	2.6

		d_1		d_2		d_3		d_4	
Name	Cl	Ag	Pa	Ag	Pa	Ag	Pa	Ag	Pa
Splice	3	3.4	50.0	54.0	59.1	5.4	57.4	47.2	**68.9**
Splice	6	7.8	54.4	61.2	65.6	31.5	58.4	58.1	**68.6**
Splice + 15	3	4.0	48.2	54.0	60.8	9.1	57.8	45.4	**69.0**
Splice + 15	6	7.6	55.0	61.1	66.5	34.9	58.8	60.7	**68.3**
Splice + 50	3	4.0	50.3	51.4	58.7	8.7	57.8	45.7	**69.0**
Splice + 50	6	8.1	53.9	60.8	65.3	40.1	58.4	58.1	**68.4**

variables are retained in the latter case. For the treeClust points, the black digits shows the size of each tree (59 of 60 are retained). We see that the "best" trees according to treeClust are in the middle, which suits our belief that, because the junction is near the middle of the sequence, variables near the middle ought to be best associated with the class of the junction.

5 Conclusions

We have described a set of inter-point dissimilarity measures useful for clustering. They define the dissimilarity between two observations as a function of the number of times those two observations fall in different leaves of classification or regression trees. These dissimilarities can be computed for numeric, categorical, or mixed data, and are insensitive to linear transformation and resistant, in some sense, to non-linear monotonic ones. They also handle missing values in a clean and reasonable way.

Our dissimilarities can be fed into standard clustering algorithms and the combination performs well compared to Gower dissimilarities, or, for numeric data, Euclidean distances. In a few cases, other approaches produce somewhat better results, but our approach seems to

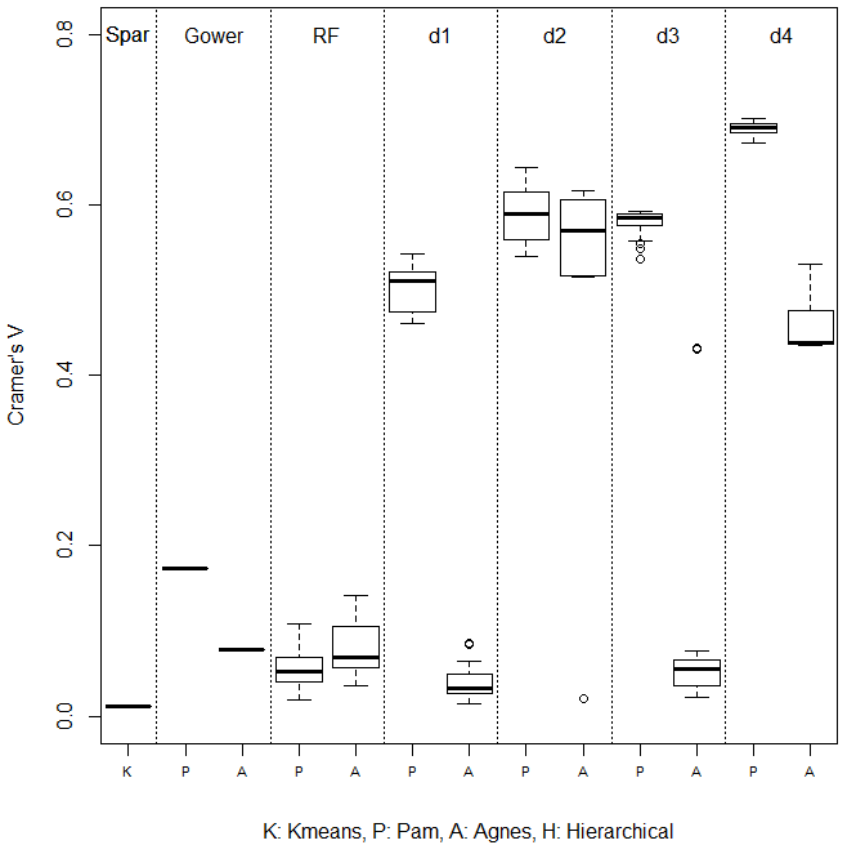

Figure 2: Boxplots of 20 Cramér values, by data or distance (top label) and clustering algorithm (bottom), using the Splice data with 50 added noise variables and three clusters

resist noise better than using Gower or random forest dissimilarities or the sparse techniques. Our approach will not, however, downweight redundant variables when those exist. While the specific best performer depends on the data, the d_2 dissimilarity with the Pam algorithm generally produces good results.

A number of directions for future work might be fruitful. First, our technique is somewhat expensive computationally, and we are implementing it in a parallel mode that allows the various trees to be built on separate processors. Second, our approach resembles cluster ensembles (Strehl and Ghosh, [25]); other approaches to consensus building, especially those that give us a computational boost, are worth looking into. Further, trees are a natural choice for building ensembles. Our methods might be used to generate ensembles

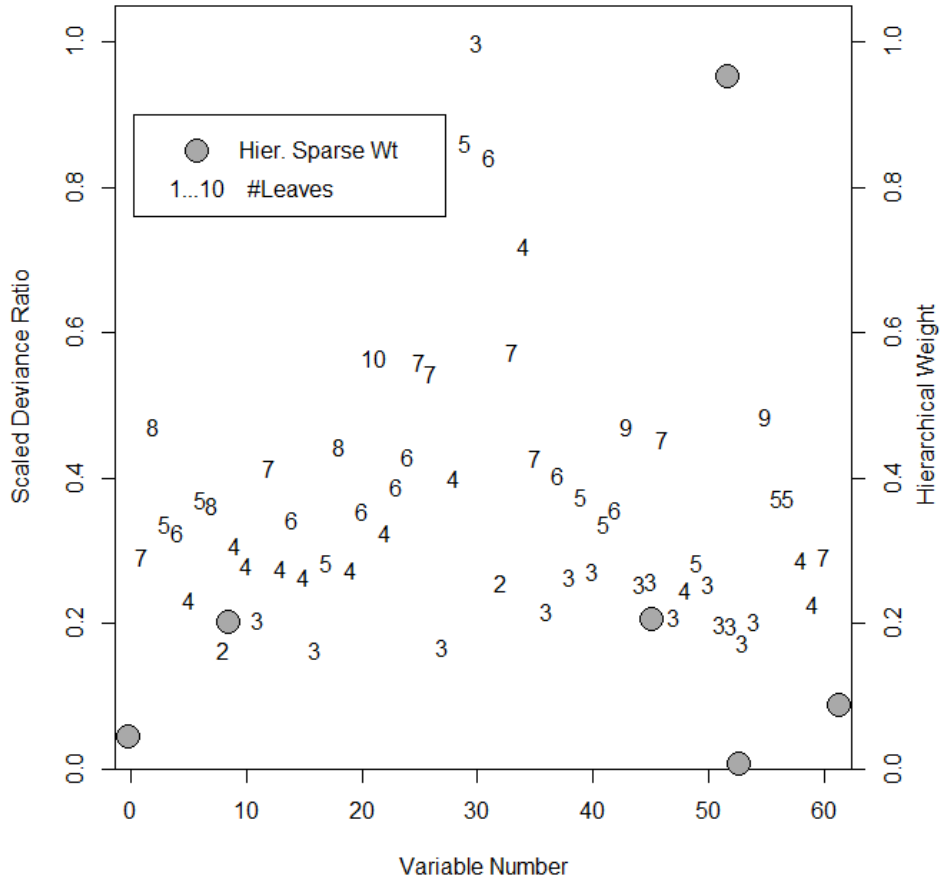

Figure 3: Trees built for the Splice data by variable number (x-axis), giving ratio of decrease in deviance to maximal decrease (left y-axis). Black numbers indicate the number of leaves in the tree. Grey circles shows variables kept by the Sparse Hierarchical algorithm, with heights given by variable weights (right y-axis)

of weak clusterers. We have implemented our approach in an R package that is available at the CRAN website, `cran.r-project.org` under the name `treeClust`, and scripts to produce all the results in this paper are available from the authors.

References

[1] J. A. Hartigan, *Clustering Algorithms*, ser. Probability and Mathematical Statistics. New York, NY: John Wiley and Sons Inc, 1975.

[2] L. Kaufman and P. J. Rousseeuw, *Finding Groups in Data: An Introduction to Analysis*, ser. Probability and Statistics. New York, NY: Wiley-Interscience, 1990.

[3] R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2013. [Online]. Available: http://www.R-project.org/

[4] B. Mirkin, *Clustering: A Data Recovery Approach, Second Edition*. London, UK: Chapman and Hall/CRC, 2012.

[5] B. Andreopoulos, A. An, X. Wang, and M. Schroeder, "A roadmap of clustering algorithms: Finding a match for a biomedical application," *Briefings in Bioinformatics*, vol. 10, no. 3, pp. 297–314, 2009.

[6] S. Boriah, V. Chandola, and V. Kumar, "Similarity measures for categorical data: A comparative evaluation." Presented at SIAM Conference on Data Mining, Atlanta, GA, 2008, 2008.

[7] A. Ahmad and L. Dey, "A k-mean clustering algorithm for mixed numeric and categorical data," *Data and Knowledge Engineering*, vol. 63, pp. 503–527, 2007.

[8] B. McCane and M. Albert, "Distance functions for categorical and mixed variables," *Pattern Recognition Letters*, vol. 29, pp. 986–993, 2008.

[9] S. E. Buttrey, "A scale-independent clustering method with automatic variable selection based on trees." Presented at the Joint Statistical Meetings, Seattle, WA, 2006, 2006.

[10] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.

[11] J. Gower, "A general coefficient of similarity and some of its properties," *Biometrics*, vol. 27, no. 4, pp. 857–871, 1971.

[12] L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification and Regression Trees*. Monterey, CA: Wadsworth and Brooks, 1984.

[13] L. Breiman and A. Cutler, *Manual–Setting Up, Using, and Understanding Random Forests v4.0*, 2003. [Online]. Available: https://www.stat.berkeley.edu/~breiman/Using_random_forests_v4.0.pdf

[14] T. Shi and S. Horvath, "Unsupervised learning with random forest predictors," *J. Computational and Graphical Statistics*, vol. 15, no. 1, pp. 118–138, 2006.

[15] D. Fisher, "Knowledge acquisition via incremental conceptual clustering," *Machine Learning*, vol. 2, pp. 139–172, 1987.

[16] H. Ooi, "Density visualization and mode hunting using trees," *J. Computational and Graphical Statistics*, vol. 11, no. 2, pp. 328–347, 2002.

[17] C. Smyth, D. Coomans, and Y. Everingham, "Clustering noisy data in a reduced dimension space via multivariate regression trees," *Pattern Recognition*, vol. 39, pp. 424–431, 2006.

[18] M. Maechler, P. Rousseeuw, A. Struyf, M. Hubert, and K. Hornik, *cluster: Cluster Analysis Basics and Extensions*, 2013. [Online]. Available: http://www.R-project.org/

[19] J. H. Friedman and J. J. Meulman, "Clustering objects on subsets of attributes," *J. Royal Statistical Society B*, vol. 66, pp. 815–849, 2004.

[20] D. M. Witten and R. Tibshirani, "A framework for feature selection in clustering," *J. Am. Stat. Assoc.*, vol. 105, no. 490, pp. 713–726, 2010.

[21] D. M. Witten and R. Tibshirani, *sparcl: Perform sparse hierarchical clustering and sparse k-means clustering*, 2013, r package version 1.0.3. [Online]. Available: http://CRAN.R-project.org/package=sparcl

[22] T. Therneau, B. Atkinson, and B. Ripley, *rpart: Recursive Partitioning and Regression Trees*, 2014, r package version 4.1-8. [Online]. Available: http://CRAN.R-project.org/package=rpart

[23] K. Bache and M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[24] M. Charytanowicz, J. N. P. Kulczycki, P. Kowalski, S. Lukasik, and S. Zak, *A Complete Gradient Clustering Algorithm for Features Analysis of X-ray Images*. Springer-Verlag, 2010.

[25] A. Strehl and J. Ghosh, "Cluster ensembles - a knowledge reuse framework for combining multiple partitions," *J. Machine Learning Research*, vol. 3, pp. 583–617, 2003.

Initial Distribution List

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Research Sponsored Programs Office, Code 41
   Naval Postgraduate School
   Monterey, California

4. Richard Mastowski (Technical Editor)
   Graduate School of Operational and Information Sciences (GSOIS)
   Naval Postgraduate School
   Monterey, California