

ALGORITHMS FOR MODEL-BASED GAUSSIAN HIERARCHICAL CLUSTERING*

CHRIS FRALEY†

Abstract. Agglomerative hierarchical clustering methods based on Gaussian probability models have recently shown promise in a variety of applications. In this approach, a maximum-likelihood pair of clusters is chosen for merging at each stage. Unlike classical methods, model-based methods reduce to a recurrence relation only in the simplest case, which corresponds to the classical sum of squares method. We show how the structure of the Gaussian model can be exploited to yield efficient algorithms for agglomerative hierarchical clustering.

Key words. hierarchical agglomeration, mixture models, model-based cluster analysis

AMS subject classifications. 62H30, 65F99, 65Y20

PII. S1064827596311451

1. Introduction. Multivariate Gaussian models have been proposed for quite some time as a basis for clustering algorithms. Recently, methods of this type have shown promise in a number of practical applications. Examples in the geophysical sciences include seismic data processing, in the biological sciences classification of cell types based on chemical responses, and in the social sciences classification based on attachment theory in psychology. They have also been used for clustering various types of industrial and financial data. Image-processing applications include unsupervised texture image segmentation, tissue classification in biomedical images, identification of objects in astronomy, analysis of images from molecular spectroscopy, and recognition and classification of surface defects in manufactured products.

Agglomerative hierarchical clustering (Murtagh and Raftery [8], Banfield and Raftery [1]), the EM (Expectation-Maximization) algorithm and related iterative techniques (Celeux and Govaert [3]), or some combination of these (Dasgupta and Raftery [4]) are effective computational techniques for obtaining partitions from these models. The subject of efficient computation in this context has, however, received little attention. We aim to fill this gap in the case of agglomerative hierarchical clustering. Although no iterative computation is involved, the issue of efficiency is nevertheless important since the practical value of these methods is limited by a growth in time complexity which is at least quadratic in the initial number of groups. This paper is organized as follows: the remainder of this section gives the necessary background in model-based clustering and hierarchical agglomeration. In section 2, we propose computational techniques for each of the four simplest and most common Gaussian models, and compare the performance of each method to an appropriate benchmark. Finally, extension to more complex Gaussian models is discussed in section 3.

1.1. Model-based cluster analysis. The relevant probability model is as follows: the population of interest consists of G different subpopulations; the density of a p -dimensional observation \mathbf{x}_i from the k th subpopulation is $f_k(\mathbf{x}_i | \theta_k)$ for some

*Received by the editors November 1, 1996; accepted for publication (in revised form) July 6, 1997; published electronically August 7, 1998. Funded by the Office of Naval Research under contracts N00014-96-1-0192 and N00014-96-1-0330.

<http://www.siam.org/journals/sisc/20-1/31145.html>

†Department of Statistics, Box 354322, University of Washington, Seattle, WA 98195 (fraley@stat.washington.edu).

vector of parameters θ_k . Given observations $\mathbf{x} = (\mathbf{x}_1, \dots, \mathbf{x}_n)$, let $\gamma = (\gamma_1, \dots, \gamma_n)^T$ denote identifying labels for the classification, where $\gamma_i = k$ if \mathbf{x}_i comes from the k th subpopulation. The values of the parameters $\theta_1, \dots, \theta_G$ and the classification γ are unknown; in the *classification likelihood* approach to clustering, they are chosen so as to maximize the likelihood

$$(1) \quad \mathcal{L}(\theta_1, \dots, \theta_G; \gamma \mid \mathbf{x}) = \prod_{i=1}^n f_{\gamma_i}(\mathbf{x}_i \mid \theta_{\gamma_i}).$$

Our focus is on the case where $f_k(\mathbf{x}_i \mid \theta_k)$ is multivariate normal (Gaussian). In this instance the parameters θ_k consist of a mean vector μ_k and a covariance matrix Σ_k . The overall approach is much more general and is not restricted to multivariate normal distributions [1]. However, experience to date suggests that clustering based on the multivariate normal distribution is useful in a great many situations of interest ([8], [1], [9], [3], [4]).

When $f_k(\mathbf{x}_i \mid \theta_k)$ is multivariate normal, the likelihood (1) has the form

$$(2) \quad \mathcal{L}(\mu_1, \dots, \mu_G; \Sigma_1, \dots, \Sigma_G; \gamma \mid \mathbf{x}) = \prod_{k=1}^G \prod_{i \in \mathcal{I}_k} (2\pi)^{-\frac{p}{2}} |\Sigma_k|^{-\frac{1}{2}} \exp \left\{ -\frac{1}{2} (\mathbf{x}_i - \mu_k)^T \Sigma_k^{-1} (\mathbf{x}_i - \mu_k) \right\},$$

where $\mathcal{I}_k = \{i : \gamma_i = k\}$ is the set of indices corresponding to observations belonging to the k th group. The maximum likelihood estimator of μ_k in (2) is known to be the group average or centroid $\bar{\mathbf{x}}_k = s_k/n_k$, where s_k and n_k are the sum and number of observations in the k th group, respectively. Replacing μ_k by $\hat{\mu}_k = \bar{\mathbf{x}}_k$ yields the *concentrated log-likelihood*

$$(3) \quad l(\Sigma_1, \dots, \Sigma_G; \gamma \mid \mathbf{x}; \hat{\mu}_1, \dots, \hat{\mu}_G) = -\frac{pn \log(2\pi)}{2} - \frac{1}{2} \sum_{k=1}^G \{ \text{tr}(W_k \Sigma_k^{-1}) + n_k \log |\Sigma_k| \},$$

in which the covariances Σ_k and the classification γ remain to be determined. The matrix $W_k = \sum_{i \in \mathcal{I}_k} (\mathbf{x}_i - \bar{\mathbf{x}}_k) (\mathbf{x}_i - \bar{\mathbf{x}}_k)^T$ is the sample cross-product matrix for the k th group.

If $\Sigma_k = \sigma^2 I$, then the log-likelihood (3) is maximized by classifications γ that minimize $\text{tr} \left(\sum_{k=1}^G W_k \right)$. This is the well-known sum of squares criterion which, for example, was suggested by Ward [11] as a possible metric when he proposed the agglomerative hierarchical method for clustering. An alternative that allows a different variance for each group is $\Sigma_k = \sigma_k^2 I$; γ is chosen so as to minimize $\sum_{k=1}^G n_k \log \text{tr} \left(\frac{W_k}{n_k} \right)$ [1]. If Σ_k is the same for all groups but otherwise has no structural constraints, then values of γ that minimize $\left| \sum_{k=1}^G W_k \right|$ maximize the log-likelihood [5]. When Σ_k is allowed to vary completely between groups, the log-likelihood is maximized whenever γ minimizes $\sum_{k=1}^G n_k \log \left| \frac{W_k}{n_k} \right|$ [10]. Table 1 summarizes the equivalent criteria to be minimized corresponding to these four parameterizations of Σ_k .

1.2. Hierarchical agglomeration. Agglomerative hierarchical clustering [11] is a stagewise procedure in which “optimal” pairs of clusters are successively merged.

TABLE 1

Four parameterizations of the covariance matrix Σ_k in the Gaussian model with the corresponding criteria to be minimized.

Σ_k	Criterion	Σ_k	Criterion
$\sigma^2 I$	$\text{tr} \left(\sum_{k=1}^G W_k \right)$	Σ	$\sum_{k=1}^G W_k$
$\sigma_k^2 I$	$\sum_{k=1}^G n_k \log \left[\text{tr} \left(\frac{W_k}{n_k} \right) \right]$	Σ_k	$\sum_{k=1}^G n_k \log \left \frac{W_k}{n_k} \right $

Each stage of merging corresponds to a unique number of clusters, and a unique partition of the data. Classifications differ according to the criterion for optimality and the strategy for choosing a single pair when more than one is optimal. In model-based hierarchical clustering, a maximum-likelihood pair is merged at each stage. Although the resulting partitions are suboptimal, agglomerative hierarchical clustering methods are in common use because they often yield reasonable results and are relatively easy to compute. For model-based clustering, another advantage of hierarchical agglomeration is that there is an associated Bayesian criterion for choosing the best partition (hence the optimal number of clusters) from among those defined by the hierarchy [1]. Hierarchical clustering can be accomplished by splitting rather than agglomeration, but the complexity of such algorithms is combinatorial unless severe restrictions on the allowed subdivisions are applicable.

The process of hierarchical agglomeration is usually assumed to start with each observation in a cluster by itself, and to proceed until all observations are in a single cluster. However, it could just as well be started from a given partition and proceed from there to form larger clusters. The value returned consists of a “classification tree” (a list of pairs of clusters merged) and possibly the optimal value of the change in criterion at each stage.

In classical agglomerative methods (e.g., sum of squares, nearest and farthest neighbor (single and complete link) [7]), there is a metric or “cost” based on geometric considerations associated with merging a pair of clusters. For a particular pair this cost remains fixed as long as neither of the clusters in that pair is involved in a merge, so that the time complexity of hierarchical agglomeration can be significantly reduced if the cost of merging pairs is retained and updated during the course of the algorithm. The overall memory usage is then proportional to the square of the initial number of clusters (the initial number of observations if a coarse partition is not available), which could be a severe limitation. For large data sets, one possible strategy is to apply hierarchical agglomeration to a subset of the data and partition the remaining observations via supervised classification or discriminant analysis. Banfield and Raftery [1] used only 522 out of 26,000 pixels in an initial hierarchical phase to successfully classify tissue from an MRI brain-scan image via Gaussian model-based techniques.

For each classical method, there is a simple recurrence relation for updating the cost $\Delta(i, j)$ of merging pairs. Once $\Delta(i, j)$ is initialized, computation can proceed without further reference to the data; the size of each group must be retained and updated. In the sum-of-squares method, the recurrence is

$$(4) \quad \Delta(\langle i, j \rangle, k) = \frac{(n_i + n_k)\Delta(i, k) + (n_j + n_k)\Delta(j, k) - n_k\Delta(i, j)}{n_i + n_j + n_k},$$

where $\langle i, j \rangle$ represents the group formed by merging groups i and j . When i and j are singletons, $\Delta(i, j)$ is half the Euclidean distance between them, and in general

$$\Delta(i, j) = \frac{n_i n_j}{n_i + n_j} \|\bar{\mathbf{x}}_i - \bar{\mathbf{x}}_j\|_2^2.$$

The quantity $\Delta(i, j)$ can be shown to be the change in the value of the sum of squares that would result if groups i and j are merged.

The amount of space needed for $\Delta(i, j)$ decreases as the number of groups increases. A memory-efficient scheme for maintaining $\Delta(i, j)$ is as follows. Assume (without loss of generality) that observation number k is the observation of the smallest index in group k in the initial classification. If for each group $j > 1$, values of $\Delta(j, i)$ are stored for all $i < j$, then it is easy to recover space during the course of the computation. Assuming that j is the highest index in a particular merge, and l is the largest current index, the space associated with group j can be used for group l , thereby freeing the (larger) space associated with group l . The original indexes for the groups can easily be recovered at the end. In programming languages such as Fortran 77 in which memory allocation is static, values of $\Delta(j, i)$ can be stored sequentially in the order $\Delta(2, 1), \Delta(3, 1), \Delta(3, 2), \Delta(4, 1), \Delta(4, 2), \Delta(4, 3), \dots$, so that the scheme described above leaves contiguous free space that can be used for the classification tree and other return values. In languages such as C that allow dynamic memory allocation, a separate list of values $\Delta(j, i)$ for all $i < j$ can be maintained for each j ; the space associated with the list for the largest value of j can be freed at each stage under this scheme.

Model-based methods generally require more computational resources than classical methods. In some there is no advantage in storing the cost of merging pairs, and some require quantities that are nontrivial to compute such as determinants of the cross-product matrices. The object of this paper is to show that there are relatively efficient methods for agglomerative hierarchical clustering based on Gaussian models.

2. Efficient algorithms for the four basic models. There is clearly structure to be exploited in the various criteria, in which W_k is a symmetric, positive semidefinite matrix (see Table 1 in section 1.1). Moreover, since only two groups are merged at each stage of hierarchical agglomeration, there should be a close relationship between criteria at successive stages. In fact, the sample cross-product matrix for the merged group can be obtained from the sum of the sample cross-product matrices of its two component groups by means of a symmetric rank-1 update:

$$(5) \quad W_{\langle i, j \rangle} = W_i + W_j + w_{ij} w_{ij}^T,$$

where

$$(6) \quad w_{ij} \equiv \eta_{ji} s_i - \eta_{ij} s_j, \quad \eta_{ij} \equiv \sqrt{\frac{n_i}{n_j(n_i + n_j)}},$$

and s_k denotes the sum of the observations for group k . A derivation is given in the Appendix. In the remainder of this section we show that this relation leads to efficient algorithms for all of the methods of Table 1.

We assume that the input consists of an $n \times p$ matrix whose rows correspond to individual observations and a vector of length n indicating the initial classification of each observation.

2.1. $\Sigma_k = \sigma^2 \mathbf{I}$. When the covariance matrix is constrained to be diagonal and uniform across all groups, the criterion to be minimized at each stage is

$$(7) \quad \text{tr} \left(\sum_{k=1}^G W_k \right) = \sum_{k=1}^G \text{tr} (W_k).$$

This is the sum-of-squares criterion, long known as a heuristic before any relationship to the Gaussian model was recognized: $\text{tr} (W_k)$ is the sum of squares of observations in group k with the group mean subtracted out. Of the classical methods, it is the only one known to have an underlying statistical model. In view of the recurrence relation (4), all that is required to start the hierarchical clustering procedure is a set of values $\Delta(i, j)$ and the number of observations in each group. First, the value of $\Delta(i, j)$ for each pair of observations can be computed; in the absence of other information, the individual observations usually constitute the initial partition of the data, and nothing further need be done. For coarser initial partitions, the recurrence relation could be used to obtain the initial values for hierarchical clustering given $\Delta(i, j)$ for each pair of observations. Merges for initialization are determined by the given partition rather than by the minimum value of $\Delta(i, j)$ at any stage. The process just described, however, requires storage proportional to the square of the number observations n , which is undesirable if there are $m < n$ groups to begin with. The update formula (5) leads to a better initialization procedure for $\Delta(i, j)$, since

$$(8) \quad \begin{aligned} \Delta(i, j) &= \text{tr} (W_{(i,j)}) - [\text{tr} (W_i) + \text{tr} (W_j)] \\ &= \text{tr} (W_{(i,j)} - [W_i + W_j]) = \text{tr} (w_{ij} w_{ij}^T) = w_{ij}^T w_{ij}. \end{aligned}$$

Because we are assuming that k is the smallest index associated with observations in group k , we can overwrite the k th observation by the sum s_k of observations in that group and the k th element of the classification vector with n_k . This can be accomplished in $\mathcal{O}(np)$ time and requires no additional storage since the input is overwritten. Then (8) and (6) can be used to initialize $\Delta(i, j)$. The total storage required would then be $\mathcal{O}(np + m^2)$: $\mathcal{O}(np)$ for the input, and $\mathcal{O}(m^2)$ for $\Delta(i, j)$.

2.2. $\Sigma_k = \sigma_k^2 \mathbf{I}$. When the covariance of each group is constrained to be diagonal, but otherwise allowed to vary between groups, the criterion to be minimized at each stage is

$$(9) \quad \sum_{k=1}^G n_k \log \text{tr} \left(\frac{W_k}{n_k} \right) = \sum_{k=1}^G n_k \log \left[\frac{\text{tr} (W_k)}{n_k} \right].$$

As for the sum-of-squares and the other classical methods, $\Delta(i, j)$ remains unchanged from stage to stage unless either group i or group j is involved in a merge, so that storing $\Delta(i, j)$ results in a gain in time efficiency for hierarchical clustering. From the update formula (5), we have

$$\begin{aligned} \Delta(i, j) &= (n_i + n_j) \log \left[\frac{\text{tr} (W_i) + \text{tr} (W_j) + w_{ij}^T w_{ij}}{n_i + n_j} \right] \\ &\quad - \left\{ n_i \log \left[\frac{\text{tr} (W_i)}{n_i} \right] + n_j \log \left[\frac{\text{tr} (W_j)}{n_j} \right] \right\}. \end{aligned}$$

Unlike the classical methods, there is no simple recurrence relation for $\Delta(\langle i, j \rangle, k)$ given $\Delta(i, j)$, $\Delta(i, k)$, and $\Delta(j, k)$. However, there is a reasonably efficient update; all that is required is to maintain values of n_k , $\text{tr}(W_k)$, and $n_k \log[\text{tr}(W_k)/n_k]$ in addition to s_k . The vector s_k can overwrite the k th observation, as was done for the sum of squares. But this time, for each group k that has more than one element, the index k' of the next observation in that group is stored in the k th element of the classification vector. The number of elements in the group is stored in the k' th entry of the classification vector, while the trace of the sample cross-product matrix and the corresponding term of the criterion overwrite the first two elements of the k' th observation. With this scheme no additional storage is necessary ($p \geq 2$) beyond that required for $\Delta(i, j)$ and the input.

The issue of terms in which $\text{tr}(W_k) = 0$ remains to be resolved; these are terms corresponding to groups which consist of a single observation or in which all observations coincide. We replace (9) with a modified criterion in order to handle these cases transparently:

$$(10) \quad \sum_{k=1}^G n_k \log \left[\frac{\text{tr}(W_k) + \alpha \frac{\text{tr}(W)}{np}}{n_k} \right],$$

where W is the sample cross-product matrix for the group consisting of all observations; a suitable default value for α is 1. The factor $\text{tr}(W)/np$ is an attempt to take into account scaling in the data, since the resulting criterion is scale dependent.

2.3. Constant Σ_k . When the covariance is uniform across all groups but otherwise has no structural constraints, the criterion to be minimized at each stage is

$$(11) \quad \left| \sum_{k=1}^G W_k \right|.$$

In contrast to the methods discussed up to this point, the change in criterion caused by merging two groups is affected by merges among other groups in the current classification. Hence it is of no advantage here to store $\Delta(i, j)$ for the duration of the computation. Nevertheless, (5) leads to an efficient update, because the change in $W = \sum_{k=1}^G W_k$ when groups i and j are merged can be represented as

$$(12) \quad W \leftarrow W + \{W_{\langle i, j \rangle} - (W_i + W_j)\} = W + w_{ij}w_{ij}^T.$$

Instead of W itself, we maintain a lower triangular Cholesky factor L for W (see, e.g., [6]), since then the determinant can be easily computed as the square of the product of the diagonals of L :

$$|W| = |LL^T| = |L|^2 = \left\{ \prod \text{diag}(L) \right\}^2.$$

Noting that since $W = 0$ (and hence $L = 0$) when each observation is in a group by itself, we can then either build the Cholesky factor for a coarse initial partition or else merge optimal groups in hierarchical agglomeration as follows:

$$W + w_{ij}w_{ij}^T = LL^T + w_{ij}w_{ij}^T = (L \quad w_{ij}) \begin{pmatrix} L^T \\ w_{ij}^T \end{pmatrix};$$

$$\begin{pmatrix} L^T \\ w_{ij}^T \end{pmatrix} = \begin{pmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times \\ \times & \times & \times & \times \end{pmatrix} \xrightarrow{\text{Givens}} \begin{pmatrix} \tilde{\times} & \tilde{\times} & \tilde{\times} & \tilde{\times} \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times \\ \tilde{0} & \tilde{\times} & \tilde{\times} & \tilde{\times} \end{pmatrix} \xrightarrow{\text{Givens}} \begin{pmatrix} \times & \times & \times & \times \\ 0 & \tilde{\times} & \tilde{\times} & \tilde{\times} \\ 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times \\ 0 & \tilde{0} & \tilde{\times} & \tilde{\times} \end{pmatrix}$$

$$\xrightarrow{\text{Givens}} \begin{pmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \tilde{\times} & \tilde{\times} \\ 0 & 0 & 0 & \times \\ 0 & 0 & \tilde{0} & \tilde{\times} \end{pmatrix} \xrightarrow{\text{Givens}} \begin{pmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & 0 & \tilde{\times} \\ 0 & 0 & 0 & \tilde{0} \end{pmatrix} = \begin{pmatrix} \tilde{L}^T \\ 0 \end{pmatrix};$$

$$W + w_{ij}w_{ij}^T = \tilde{L}\tilde{L}^T.$$

The symbol $\xrightarrow{\text{Givens}}$ stands for application of a *Givens rotation*, an elementary orthogonal transformation that allows selective and numerically stable introduction of zero elements in a matrix. The marked entries are values changed in the last transformation. The time efficiency for the Cholesky update is $\mathcal{O}(p^2)$, in contrast to $\mathcal{O}(p^3)$ for forming a new Cholesky factor from the updated $p \times p$ matrix $W + w_{ij}w_{ij}^T$. For details of the Cholesky update via Givens rotations, see, e.g., [6].

Although the criterion is defined for all possible partitions, the first stages of hierarchical clustering using this criterion will be arbitrary if initially each observation is in a cluster by itself, since merging any pair of observations i and j will result in $W = W_{\langle i,j \rangle}$ and $|W_{\langle i,j \rangle}| = 0$ whenever i and j are singletons ($p \geq 2$). More generally, $|W| = 0$ whenever W has rank less than p . To circumvent this, we use the sum-of-squares criterion $\text{tr}(W)$ to determine merges until the value of $|W|$ is positive, while maintaining the Cholesky factor of W . As the computation proceeds, s_k overwrites the data and n_k overwrites the classification vector; most of the information needed to recover the classification tree and optimal values of the criterion can be stored in the portions of these structures that are no longer needed in the algorithm. Other than the necessary $\mathcal{O}(p^2)$ storage for maintaining L^T , additional storage of size $\mathcal{O}(M)$, where M is the number of stages, is needed when $p < 4$ to store the merge indexes in order to completely reconstruct the classification tree.

2.4. Unconstrained Σ_k . When the covariance is allowed to vary completely between groups, the criterion to be minimized at each stage is

$$(13) \quad \sum_{k=1}^G n_k \log \left| \frac{W_k}{n_k} \right|.$$

Like the criteria discussed in sections 2.1 and 2.2, each group contributes a separate additive term to (13), so that it is time efficient to save values of $\Delta(i, j)$. If L_k denotes the Cholesky factor of W_k , then, in view of (5),

$$L_{\langle i,j \rangle} L_{\langle i,j \rangle}^T = L_i L_i^T + L_j L_j^T + w_{ij}w_{ij}^T = (L_i \quad L_j \quad w_{ij}) (L_i \quad L_j \quad w_{ij})^T.$$

$L_{\langle i,j \rangle}$ can be computed efficiently from L_i , L_j , and w_{ij} by applying Givens rotations (see section 2.3) to the composite matrix:

$$(14) \quad \begin{pmatrix} L_i^T \\ L_j^T \\ w_{ij}^T \end{pmatrix} \rightarrow \dots \rightarrow \begin{pmatrix} L_{\langle i,j \rangle}^T \\ 0 \\ 0 \end{pmatrix}.$$

The composite matrix is never explicitly formed; instead, w_{ij} and each row of L_j^T are treated as separate updates to L_i^T . Although the time-efficiency to form the updated Cholesky factor is of the same order of magnitude as that for forming a new Cholesky factor from the updated $W_{\langle i,j \rangle}$, the use of (14) has an advantage in storage efficiency. Maintaining the upper triangle of each W_k and updating directly via (5) would require more storage since W_k has p rows regardless of n_k , whereas L_k has at most $\min(n_k - 1, p)$ nonzero rows ($W_k = 0$ whenever $n_k = 1$). Moreover, s_k and L_k^T can overwrite the data, and the entries corresponding to the lower triangle of L_k^T can be used for the necessary pointers and values to be updated (the k th term in (13)). Besides what is required for the data and for $\Delta(i, j)$, additional $\mathcal{O}(p^2)$ storage is needed for the Cholesky factors when updating $\Delta(i, j)$.

Finally, because $|W_k| = 0$ whenever $n_k \leq p$, there is even greater ambiguity with criterion (13) than with either (11) or (9). For this reason, we use

$$(15) \quad \sum_{k=1}^G n_k \log \left[\left| \frac{W_k}{n_k} \right| + \beta \left\{ \frac{\text{tr}(W_k) + \alpha \frac{\text{tr}(W)}{np}}{n_k} \right\} \right],$$

in place of (13), which gives a hybrid between the modified criterion for $\Sigma_k = \sigma_k^2 I$ (10) and (13). A suitable default value for both α and β is 1.

2.5. Benchmark comparisons. Figure 1 shows marked gains in time efficiency for the methods of section 2 relative to methods that do not use the update. Randomly generated observations of dimension $p = 5$ were used, with the default initial partition in which each singleton observation constitutes a cluster. The basic methods were written in Fortran with an S-Plus interface and were invoked using S-Plus version 3.3¹ on a Silicon Graphics Iris workstation under the IRIX 5.2 operating system. The solid line represents the performance of algorithms based on the update formula (5), while the dashed line represents performance of algorithms in which the necessary quantities are obtained without updating. In every case, the gain improves with increasing number of observations. Timings depend on the data as well as its dimension and the clustering method; Figure 1 shows the average times over nine trials. The results for $\Sigma_k = \sigma^2 I$ are a point of comparison for all methods, since in that case the solid line represents that classical sum of squares approach via the well-known recurrence relation (4). Note that the time scale for the constant-variance method differs from that of the other methods, which use more memory in exchange for improved time efficiency.

3. Extension to more complex Gaussian models. Banfield and Raftery [1] developed a model-based framework that subsumes all of the parameterizations in Table 1. The resulting clustering methods include some criteria that are more general than $\Sigma_k = \sigma^2 I$ or constant Σ_k , while still constraining the structure of Σ_k . This is

¹*S-Plus Version 3.3 for Unix*, MathSoft, Inc., Seattle, WA (1995).

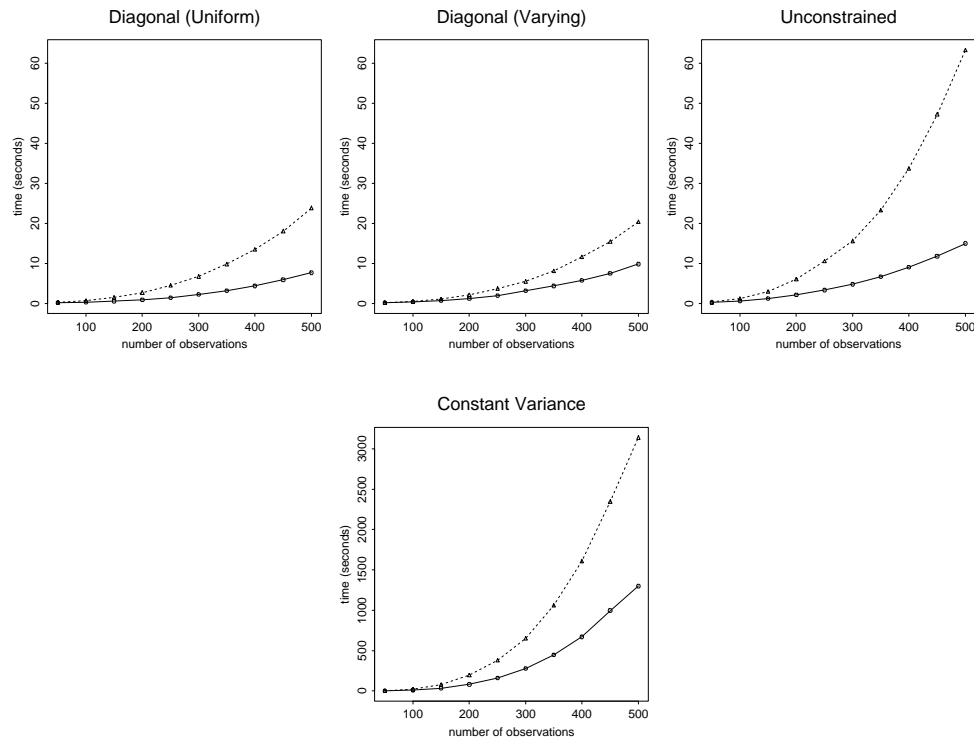


FIG. 1. CPU time vs. number of observations for the four basic models. The solid line represents the methods proposed in this paper.

accomplished by means of a reparameterization of the covariance matrix in terms of its eigenvalue decomposition

$$(16) \quad \Sigma_k = \lambda_k D_k A_k D_k^T,$$

where D_k is the orthogonal matrix of eigenvectors, A_k is a diagonal matrix whose elements are proportional to eigenvalues of Σ_k , and λ_k is a scalar. The orientation of the principal components of Σ_k is determined by D_k , while A_k determines the shape of the density contours; $\lambda_k^p |A_k|$ is proportional to the volume of the corresponding ellipsoid. This paradigm is particularly useful for two- and three-dimensional data, where geometric features can often be identified visually. It may also be applicable for higher-dimensional data when multivariate visualization analysis reveals some structure. For example, Banfield and Raftery [1] were able to closely match the clinical classification of a biomedical data set using Gaussian hierarchical clustering after analyzing its geometric features. The parameterization can be selected so as to allow some but not all of the characteristics (orientation, volume, and shape) of distributions to vary between groups, while constraining others to be the same.

Analysis of the model that leads to the sum of squares criterion ($\Sigma_k = \lambda I$ or $\sigma^2 I$) in terms of (16) suggests that it is likely to be most appropriate when groups are spherical and of approximately the same size. The constant-variance assumption in which D_k , λ_k , and A_k are the same for all groups but otherwise unconstrained favors clusters that are ellipsoidal with the same orientation, shape, and volume. If all

elements of Σ_k are allowed to vary between groups, the resulting classification is likely to contain elliptical groups with differing geometric features. Metrics appropriate for various intermediate situations can also be formulated. For example, assuming that $\Sigma_k = \lambda_k I$ or $\sigma_k^2 I$ implies that the underlying densities are spherical, while variation in λ_k between groups allows their volumes to differ. Celeux and Govaert [2] analyzed this criterion and showed that it can give classification performance that is much better than traditional methods. In one example, they successfully apply the method to an astronomical image in which one tightly clustered galaxy is contained within another more dispersed one.

Table 2 shows relationships between orientation, volume, and shape discussed in [1]. Criteria based on other combinations of these factors are also possible [3]. An implementation of hierarchical clustering based on the algorithms described in this paper is available via the internet; see <http://www.stat.washington.edu/fraley/software.html> for details. This supersedes an earlier version which has been used in a variety of applications with some success [9].

TABLE 2

Parameterizations of the covariance matrix Σ_k in the Gaussian model and their geometric interpretation. The models shown here are those discussed in Banfield and Raftery [1].

Σ_k	Distribution	Volume	Shape	Orientation	Reference
λI	Spherical	Fixed	Fixed	NA	[11], [8], [1]
$\lambda_k I$	Spherical	Variable	Fixed	NA	[1], [3]
$\lambda D A D$	Elliptical	Fixed	Fixed	Fixed	[5], [10], [1], [3]
$\lambda_k D_k A_k D_k$	Elliptical	Variable	Variable	Variable	[10], [1], [3]
$\lambda D_k A D_k$	Elliptical	Fixed	Fixed	Variable	[8], [1], [3]
$\lambda_k D_k A D_k$	Elliptical	Variable	Fixed	Variable	[1], [3]

Efficient computational methods for the first four models in Table 2 were given in section 2. We conclude this section by showing how those techniques can be applied to the remaining models: $\Sigma_k = \lambda D_k A D_k$ and $\Sigma_k = \lambda_k D_k A D_k$. The relevant criteria are $\sum_{k=1}^G \text{tr}(A^{-1} \Omega_k)$ and $\sum_{k=1}^G n_k \log[\text{tr}(A^{-1} \Omega_k) / n_k]$, respectively, where Ω_k is the diagonal matrix of eigenvalues of W_k . In both cases, efficient algorithms are possible if s_k and L_k^T are maintained and updated in the storage provided for the original data, as described in section 2.4. Instead of information pertaining to the terms of (15), the k th term of the sum in the appropriate criterion is stored in the space corresponding to the lower triangle of L_k^T . As in all of the methods in section 3, the matrix $W_{(i,j)}$ is never explicitly formed. Rather, its nonzero eigenvalues are obtained as the squares of the singular values of $L_{(i,j)}^T$, which has $\min(n_k - 1, p)$ rows and p columns. For $\Sigma_k = \lambda_k D_k A D_k$, we include the additive term inside the logarithm that appears in (10) for $\Sigma_k = \lambda_k I$ or $\sigma_k^2 I$ in order to accommodate those cases in which W_k (and hence Ω_k) vanishes.

4. Concluding remarks. This paper has made several contributions toward computational efficiency in agglomerative hierarchical clustering. First, we gave a memory-efficient scheme suitable for any method that stores the change in criterion for each merged pair. Second, we showed that the sample cross-product matrix for the union of two Gaussian clusters can be formed by a rank-one update of the sum of sample cross-matrices of its constituent clusters, and described how this can be used to obtain efficient algorithms for model-based clustering. The update leads to a memory-efficient initialization strategy for the sum of squares method, which corresponds to

the simplest Gaussian model, as well as time- and memory-efficient algorithms for three other Gaussian models that have no counterpart in classical hierarchical agglomeration. At the same time, we gave strategies to resolve the inherent ambiguities in some of the models. Finally, we showed how these techniques can be easily extended to two additional Gaussian models based on a more sophisticated parameterization of the covariance matrix that has recently shown promise in practical applications.

An implementation of hierarchical clustering based on these considerations has been made available on the internet; see <http://www.stat.washington.edu/fraley/software.html> for details.

Appendix A. Derivation of the update formula. *The following holds for the group $\langle i, j \rangle$ formed by merging groups i and j :*

$$W_{\langle i, j \rangle} = W_i + W_j + w_{ij} w_{ij}^T,$$

where W_k denotes the sample cross-product matrix for group k , and

$$w_{ij} \equiv \eta_{ji} s_i - \eta_{ij} s_j; \quad \eta_{ij} \equiv \sqrt{\frac{n_i}{n_j(n_i + n_j)}},$$

where s_k is the sum of the observations and n_k is the cardinality of group k .

Proof. Let X_k be the matrix of observations corresponding to group k . If

$$\widetilde{X}_k \equiv X_k - \frac{e_{n_k} s_k^T}{n_k},$$

where e_{n_k} denotes the vector of length n_k in which every element is equal to 1 (\widetilde{X}_k is X_k with the mean subtracted out of each column), then

$$W_k = \widetilde{X}_k^T \widetilde{X}_k = X_k^T X_k - \frac{s_k s_k^T}{n_k},$$

since $s_k = X_k^T e_{n_k}$. Because $\langle i, j \rangle$ consists of those observations in groups i and j ,

$$W_{\langle i, j \rangle} = \begin{pmatrix} X_i \\ X_j \end{pmatrix}^T \begin{pmatrix} X_i \\ X_j \end{pmatrix} - \frac{(s_i + s_j)(s_i + s_j)^T}{n_i + n_j}.$$

Hence

$$\begin{aligned} W_{\langle i, j \rangle} - (W_i + W_j) &= \left(\frac{s_i s_i^T}{n_i} + \frac{s_j s_j^T}{n_j} \right) - \frac{(s_i + s_j)(s_i + s_j)^T}{n_i + n_j} \\ &= \frac{n_j}{n_i(n_i + n_j)} s_i s_i^T + \frac{n_i}{n_j(n_i + n_j)} s_j s_j^T - \left(\frac{s_i s_j^T}{n_i + n_j} + \frac{s_j s_i^T}{n_i + n_j} \right) \\ &= \eta_{ji}^2 s_i s_i^T + \eta_{ij}^2 s_j s_j^T - (\eta_{ji} \eta_{ij} s_i s_j^T + \eta_{ij} \eta_{ji} s_j s_i^T) = w_{ij} w_{ij}^T. \quad \square \end{aligned}$$

Acknowledgments. We are indebted to principal investigator Adrian Raftery for his expertise and enthusiastic support, as well as to the associate editor for suggesting a number of significant improvements.

REFERENCES

- [1] J. D. BANFIELD AND A. E. RAFTERY, *Model-based Gaussian and non-Gaussian clustering*, *Biometrics*, 49 (1993), pp. 803–821.
- [2] G. CELEUX AND G. GOVAERT, *Comparison of the mixture and the classification maximum likelihood in cluster analysis*, *J. Statist. Comput. Simulation*, 47 (1993), pp. 127–146.
- [3] G. CELEUX AND G. GOVAERT, *Gaussian parsimonious clustering models*, *Pattern Recognition*, 28 (1995), pp. 781–793.
- [4] A. DASGUPTA AND A. E. RAFTERY, *Detecting features in spatial point processes with clutter via model-based clustering*, *J. Amer. Statist. Assoc.*, 93 (1998), pp. 294–302.
- [5] H. P. FRIEDMAN AND J. RUBIN, *On some invariant criteria for grouping data*, *J. Amer. Statist. Assoc.*, 62 (1967), pp. 1159–1178.
- [6] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, 3rd ed., The Johns Hopkins University Press, 1996.
- [7] L. KAUFMAN AND P. J. ROUSSEEUW, *Finding Groups in Data*, Wiley, 1990.
- [8] F. MURTAGH AND A. E. RAFTERY, *Fitting straight lines to point patterns*, *Pattern Recognition*, 17 (1984), pp. 479–483.
- [9] A. E. RAFTERY, *Transitions from ONR Contract N00014-91-J-1074 ‘Time Series and Image Analysis’*, Manuscript, Department of Statistics, University of Washington, December 1993.
- [10] A. J. SCOTT AND M. J. SYMONS, *Clustering methods based on likelihood ratio criteria*, *Biometrics*, 27 (1971), pp. 387–397.
- [11] J. H. WARD, *Hierarchical groupings to optimize an objective function*, *J. Amer. Statist. Assoc.*, 58 (1963), pp. 234–244.