Journal of Global Optimization **16:** 23–32, 2000.
© 2000 *Kluwer Academic Publishers. Printed in the Netherlands.*

k-Plane Clustering

P.S. BRADLEY and O.L. MANGASARIAN
Microsoft Research, Redmond, WA 98052, USA and Computer Sciences Department, University of Wisconsin, Madison, WI 53706, USA
E-mail: bradley@microsoft.com, olvi@cs.wisc.edu

Received 13 October 1998; Accepted in revised form 22 June 1999

Abstract. A finite new algorithm is proposed for clustering m given points in n-dimensional real space into k clusters by generating k planes that constitute a local solution to the nonconvex problem of minimizing the sum of squares of the 2-norm distances between each point and a *nearest* plane. The key to the algorithm lies in a formulation that generates a plane in n-dimensional space that minimizes the sum of the squares of the 2-norm distances to each of m_1 given points in the space. The plane is generated by an eigenvector corresponding to a smallest eigenvalue of an $n \times n$ simple matrix derived from the m_1 points. The algorithm was tested on the publicly available Wisconsin Breast Prognosis Cancer database to generate well separated patient survival curves. In contrast, the k-mean algorithm did not generate such well-separated survival curves.

Key words: Clustering, *k*-Mean, Linear regression

1. Introduction

There are many approaches to clustering such as statistical [2, 6, 9], machine learning [7, 8] and mathematical programming [4, 15, 16]. In this work we take a mathematical programming approach with a novel idea. Instead of generating cluster centers as points that minimize the sum of squares of distances of each given point to a nearest cluster center, we change the entity of the center from being a point to that of being a plane. The justification for this approach is that data sometimes naturally falls into clusters grouped around flat surfaces such as planes. This approach yields interesting theoretical results that lead to an efficiently implementable algorithm which gives better computational results than the standard *k*-mean algorithm [1] on a publicly available dataset.

We outline the contents of the paper now. In Section 2 we formulate the *k*-plane clustering problem and state the *k*-plane clustering algorithm. In Section 3 we derive the theoretical results needed to justify the algorithm and establish its finite termination at a locally optimal solution. In Section 4 we describe our computational results. Section 5 concludes the paper.

Throughout this paper, e will denote a vector of ones of appropriate dimension and a prime will denote the transpose.

2. The *k*-Plane Clustering (kPC) Algorithm

We consider a set $\mathcal{A}$ of m points in the n-dimensional real space R^n represented by the matrix $A \in R^{m\times n}$. We wish to cluster $\mathcal{A}$ into k clusters according to the following nonconvex minimization problem. Determine k *cluster planes* in R^n:

$$P_\ell := \{x \mid x \in R^n, x'w_\ell = \gamma_\ell\}, \quad \ell = 1, \dots, k, \tag{1}$$

that minimize the sum of the squares of distances of each point of $\mathcal{A}$ to a *nearest* plane P_ℓ. The algorithm is similar to the k-mean [1] and k-median [4] algorithms in that it alternates between assigning points to a nearest cluster plane (*Cluster Assignment*) and, for a given cluster, computing a cluster plane that minimizes the sum of the squares of distances to all points in the cluster (*Cluster Update*). It is the latter computation, which is a one step replacement of an algorithm for the Euclidean Regression Problem [5, 17] which does not use squared distances, that makes the following kPC algorithm possible.

ALGORITHM 1. **kPC: *k*-Plane Clustering Algorithm.** *Start with random* $(w_1^0, \gamma_1^0), \dots, (w_k^0, \gamma_k^0)$, *each in* R^{n+1} *with* $\|w_i^0\|_2 = 1$, $i = 1, \dots, k$. *Having* $(w_1^j, \gamma_1^j), \dots, (w_k^j, \gamma_k^j)$ *at iteration* j *with* $\|w_i^j\|_2 = 1$, $i = 1, \dots, k$, *compute* $(w_1^{j+1}, \gamma_1^{j+1}), \dots, (w_k^{j+1}, \gamma_k^{j+1})$ *by the following two steps:*

- **(a) Cluster Assignment: (Assign each point to closest plane P_ℓ)** *For each* A_i, $i = 1, \dots m$, *determine* $\ell(i)$ *such that*

$$|A_i w_{\ell(i)}^j - \gamma_{\ell(i)}^j| = \min_{\ell=1,\dots,k} |A_i w_\ell^j - \gamma_\ell^j|.$$

- **(b) Cluster Update: (Find a plane P_ℓ that minimizes the sum of the squares of distances to each point in cluster ℓ)** *For* $\ell = 1, \dots, k$ *let* $A(\ell)$ *be the* $m(\ell)\times n$ *matrix with rows corresponding to all* A_i *assigned to cluster* ℓ. *Define* $B(\ell) := [A(\ell)]'(I - \frac{ee'}{m(\ell)})A(\ell)$. *Set* w_ℓ^{j+1} *to be an eigenvector of* $B(\ell)$ *corresponding to the smallest eigenvalue of* $B(\ell)$. *Set* $\gamma_\ell^{j+1} := \frac{e'A(\ell)w_\ell^{j+1}}{m(\ell)}$.

Stop whenever there is a repeated overall assignment of points to cluster planes or a nondecrease in the overall objective function.

We give in the next section the theoretical justification for the kPC algorithm and establish its finite termination.

3. Theoretical Justification of kPC Algorithm

We first note that the cluster assignment rule defined in Step (a) of the kPC Algorithm 1 follows from the well known fact [12] that the 2-norm distance between a point $A_i \in R^n$ and the plane $P_\ell := \{x \mid x \in R^n, x'w_\ell = \gamma_\ell\}$ is $|A_i w_\ell - \gamma_\ell|/\|w_\ell\|_2 = |A_i w_\ell - \gamma_\ell|$. The last equality follows from $\|w_\ell\|_2 = 1$.

The cluster update rule defined in Step (b) of the kPC Algorithm 1 follows from Theorem 5 below. But first we prove a few simple lemmas.

LEMMA 1. *Let $A \in R^{m\times n}$. Then,*

$$\left\langle \begin{array}{c} Aw - e\gamma = 0,\ w \neq 0 \\ \text{has no solution } (w,\gamma) \end{array} \right\rangle \Leftrightarrow \left\langle \begin{array}{c} rank(A) = n, \text{ and} \\ Aw = e \text{ has no solution } w \end{array} \right\rangle. \tag{2}$$

Proof. ($\Rightarrow$) If $rank(A) < n$, then $Aw - e\cdot 0 = 0,\ w \neq 0$ has a solution which is a contradiction. If $Aw = e$ has a solution, then $Aw - e(1) = 0,\ w \neq 0$ has a solution which is again a contradiction.

($\Leftarrow$) If $Aw - e\gamma = 0,\ w \neq 0$ has a solution, then either $\gamma = 0$ or $\gamma \neq 0$. In the first case, $rank(A) < n$. In the second case, by dividing by γ, we have that $Aw = e$ has a solution. In either case, a contradiction ensues. □

LEMMA 2. *Let $A \in R^{m\times n}$, then*

$$A'\left(I - \frac{ee'}{m}\right)A = A'\left(I - \frac{ee'}{m}\right)^2 A. \tag{3}$$

Proof.

$$\left(I - \frac{ee'}{m}\right)^2 - \left(I - \frac{ee'}{m}\right) = I - 2\frac{ee'}{m} + \frac{ee'ee'}{m^2} - I + \frac{ee'}{m} = 0. \tag{4}$$

□

LEMMA 3. $B := A'\left(I - \frac{ee'}{m}\right)A$ *is positive semidefinite.*

Proof. By Lemma 2,

$$w'Bw = \left\|\left(I - \frac{ee'}{m}\right)Aw\right\|_2^2 \geqslant 0. \tag{5}$$

□

LEMMA 4. $Aw - e\gamma = 0,\ w \neq 0$ *has no solution* $\Leftrightarrow$ B *is positive definite.*

Proof. ($\Rightarrow$) By Lemma 3, B is positive semidefinite. If B is *not* positive definite then, by Lemma 3, $(I - \frac{ee'}{m})Aw = 0,\ w \neq 0$ has a solution. But, by Lemma 1, $rank(A) = n$, hence $z = Aw \neq 0$. Thus, $(I - \frac{ee'}{m})z = 0$, or $z = e\frac{e'z}{m} = \alpha e$, where $\alpha = \frac{e'z}{m}$. Since $z \neq 0$, it follows that $\alpha \neq 0$ and $e = \frac{z}{\alpha} = A\frac{w}{\alpha}$, contradicting the fact (from Lemma 1) that $Aw = e$ has no solution.

($\Leftarrow$) If B is positive definite, then by Lemma 3, $(I - \frac{ee'}{m})Aw = 0$ has no solution $w \neq 0$. Hence rank$(A) = n$. Also $Aw = e$ has no solution, else $(I - \frac{ee'}{m})Aw = e - e = 0$. Thus by Lemma 1, $Aw - e\gamma = 0,\ w \neq 0$, has no solution. □

We are ready now to state the theorem that explicitly gives the plane that minimizes the sum of the squares of the 2-norm distances to m given points in R^n.

THEOREM 5. *Let $A \in R^{m \times n}$. Then a global solution of:*

$$\begin{array}{ll} \underset{(w,\gamma)\in R^{n+1}}{\text{minimize}} & \|Aw - e\gamma\|_2^2 \\ \text{subject to} & w'w = 1, \end{array} \tag{6}$$

is attained at any eigenvector w of $B := A'(I - \frac{ee'}{m})A$ corresponding to a minimum eigenvalue of B and $\gamma = \frac{e'Aw}{m}$. The minimum of (6) is positive if and only if B is positive definite or equivalently if and only if $rank(A) = n$ and $Aw = e$ has no solution.

Proof. The second part follows from Lemmas 1 and 4. We now prove the first part. The set of all stationary points of (6) including all its global minima render the partial derivatives of the Lagrangian of (6) equal to zero. That is for:

$$L(w, \gamma, \lambda) := \|Aw - e\gamma\|_2^2 - \lambda(w'w - 1), \tag{7}$$

it follows that:

$$\frac{1}{2}\nabla_w L(w, \gamma, \lambda) = A'(Aw - e\gamma) - \lambda w = 0, \tag{8}$$

$$-\frac{1}{2}\nabla_\gamma L(w, \gamma, \lambda) = e'(Aw - e\gamma) = 0, \tag{9}$$

$$-\nabla_\lambda L(w, \gamma, \lambda) = w'w - 1 = 0. \tag{10}$$

Hence:

$$\lambda = w'A'\left(I - \frac{ee'}{m}\right)Aw, \tag{11}$$

$$\gamma = \frac{e'Aw}{m}. \tag{12}$$

Substitution for λ and γ in (8) gives:

$$A'\left(I - \frac{ee'}{m}\right)Aw - w'A'\left(I - \frac{ee'}{m}\right)Aw \cdot w = 0. \tag{13}$$

By using the definition of B this is equivalent to:

$$Bw - w'Bw \cdot w = 0.$$

That is:

$$Bw = \nu w, \ \ \nu = w'Bw. \tag{14}$$

Thus for each stationary point (w, γ) of (6), it follows that w is an eigenvector of B and $\gamma = \frac{e'Aw}{m}$. Hence,

$$Aw - e\gamma = Aw - e\left(\frac{e'Aw}{m}\right) = \left(I - \frac{ee'}{m}\right)Aw. \tag{15}$$

We then have by Lemma 2 that:

$$\|Aw - e\gamma\|_2^2 = w'A'\left(I - \frac{ee'}{m}\right)^2 Aw = w'Bw = \nu, \tag{16}$$

where the last equality follows from (14). Hence the smallest value that ν can take on is the smallest eigenvalue of B and w is its corresponding eigenvector. □

REMARK 6. **Relation to Singular Value Decomposition.** It can be shown, after some straightforward algebra, that the w obtained in the above Theorem 5 can also be obtained by taking a singular value decomposition USV' [14, 19] of the $m \times n$ matrix:

$$H := \left(I - \frac{ee'}{m}\right)A,$$

where U and V are orthogonal matrices of dimensions $m \times m$ and $n \times n$ respectively, and S is an $m \times n$ diagonal matrix with nonnegative diagonal elements in decreasing order. It can then be shown that the desired w given by Theorem 5 corresponds to the last column of the matrix V corresponding to a smallest singular value of H, and γ is again given by (12) above. This result can be derived by noting that [14, Theorem 8.19] the squares of the singular values of H (possibly with some zeros added) are also the eigenvalues of both HH' and $H'H$ with associated eigenvectors being columns of U and V respectively. A different clustering approach, latent semantic indexing, is given in [3] that also uses singular value decomposition.

We end this section by establishing the finiteness of the kPC Algorithm.

THEOREM 7. (Finite Termination of the kPC Algorithm 1). *The kPC Algorithm 1 terminates in a finite number of steps at a cluster assignment that is locally optimal. That is, the overall objective, the sum of the squares of distances of each point to a closest cluster plane, cannot be decreased by either reassignment of a point to a different cluster plane, or by defining a new cluster plane for any of the clusters.*

Proof. In the cluster assignment part (a) of the algorithm each point is assigned to a closest plane and hence the overall objective cannot increase. Similarly in part (b) of the algorithm, the cluster plane, for each cluster, is recomputed as that plane which minimizes the sum of the squares of distances of points in that cluster to the plane. Hence, again, the overall objective cannot increase. Since there are a finite number of ways that the m points of $\mathcal{A}$ can be assigned to k clusters, since the algorithm does not permit repeated assignments by the explicit choice of the stopping criterion of part (b) of the algorithm, and since the overall objective function is non-increasing and bounded below by zero, it follows that the algorithm must terminate at some clustering assignment that is locally optimal. □

4. Computational Results

Two sets of computational tests were carried out comparing the kPC and k-mean algorithms. In the first set of tests the ability to generate well separated survival curves by clustering medical data was tested. In the second set of tests the ability to recover class labels by clustering unlabeled data was tested.

In the first set of tests the kPC algorithm was tested on the Wisconsin Prognostic Breast Cancer (WPBC) Database [13] along with the k-mean algorithm [1] using only two features: tumor size and lymph node status. These two features were normalized to have zero mean and standard deviation 1. This dataset consists of 198 points in R^2. The number of clusters was set to 3 ($k = 3$) in an attempt to find 3 groups of patients with distinct survival characteristics (see Figure 2).

Kaplan-Meier survival curves [10, 11] were constructed for each cluster, representing expected percent of surviving patients as a function of time, for patients in that cluster. Figure 1 depicts the three planes (lines in R^2) obtained by the kPC Algorithm 1. Figure 2 gives survival curves for the three clusters obtained by the kPC Algorithm 1 and the k-mean algorithm. We note that the survival curves obtained by the kPC Algorithm are well separated and hence can be used as a prognostic tool, whereas those obtained by the k-mean algorithm are not well separated and hence cannot be used as prognostic indicators.

In the second set of tests the kPC Algorithm and the k-mean algorithm were further compared in their ability to recover class labels on a holdout data subset. The datasets used here had two classes, hence $k = 2$ in these tests. A ten-fold cross-validation [18] scheme was employed. In this procedure the dataset is randomly divided into 10 disjoint sets of approximately equal size, $T_1, T_2, \ldots, T_{10}$. Then 10 trials are conducted. At trial j, the clustering algorithms are applied to the union of $T_1, \ldots, T_{j-1}, T_{j+1}, \ldots, T_{10}$ (training data) without making use of the class label for each point. Then the data points in T_j (test data) were assigned to the closest cluster plane or cluster center. Training correctness at trial j is the percentage of training data $T_1, T_2, \ldots, T_{j-1}, T_{j+1}, \ldots, T_{10}$ correctly classified by the majority label of the cluster that each point was assigned to. Similarly, testing correctness

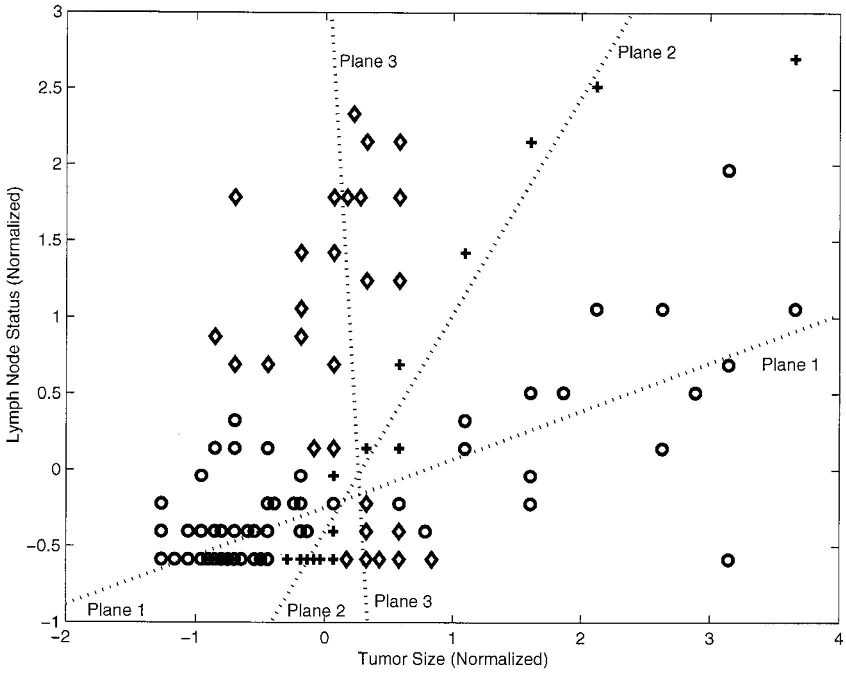

Figure 1. Three cluster lines obtained by the kPC Algorithm for the Wisconsin Prognostic Breast Cancer Database (WPBC). Data assigned to Plane 1 is indicated by ○. Data assigned to Plane 2 is indicated by +. Data assigned to Plane 3 is indicated by ◇.

at trial j is the percentage of T_j correctly classified by the majority label of the cluster that the point was assigned to.

Table 1 summarizes average training and testing results on 2 publicly available datasets [13]. The Johns Hopkins Ionosphere dataset consists of 351 data points with 34 real-valued features characterizing radar returns from the ionosphere. One class corresponds to radar returns showing evidence of structure. The other class corresponds to those returns showing no structure. The BUPA Liver Disorders dataset consists of 345 data points with 6 real-valued features. A 7th feature indicates the class of the corresponding feature. Both the Ionosphere and BUPA datasets have been normalized so that the mean of each feature is zero and standard deviation is one.

We note that the kPC clusters were better able to recover original class labels on the BUPA liver disorders dataset over both the training and testing subsets. The *k*-mean clusters were better on the Ionosphere dataset. On both the BUPA and Ionosphere datasets, the kPC algorithm converged faster than *k*-mean, as much as 6.21 times faster on the BUPA dataset.

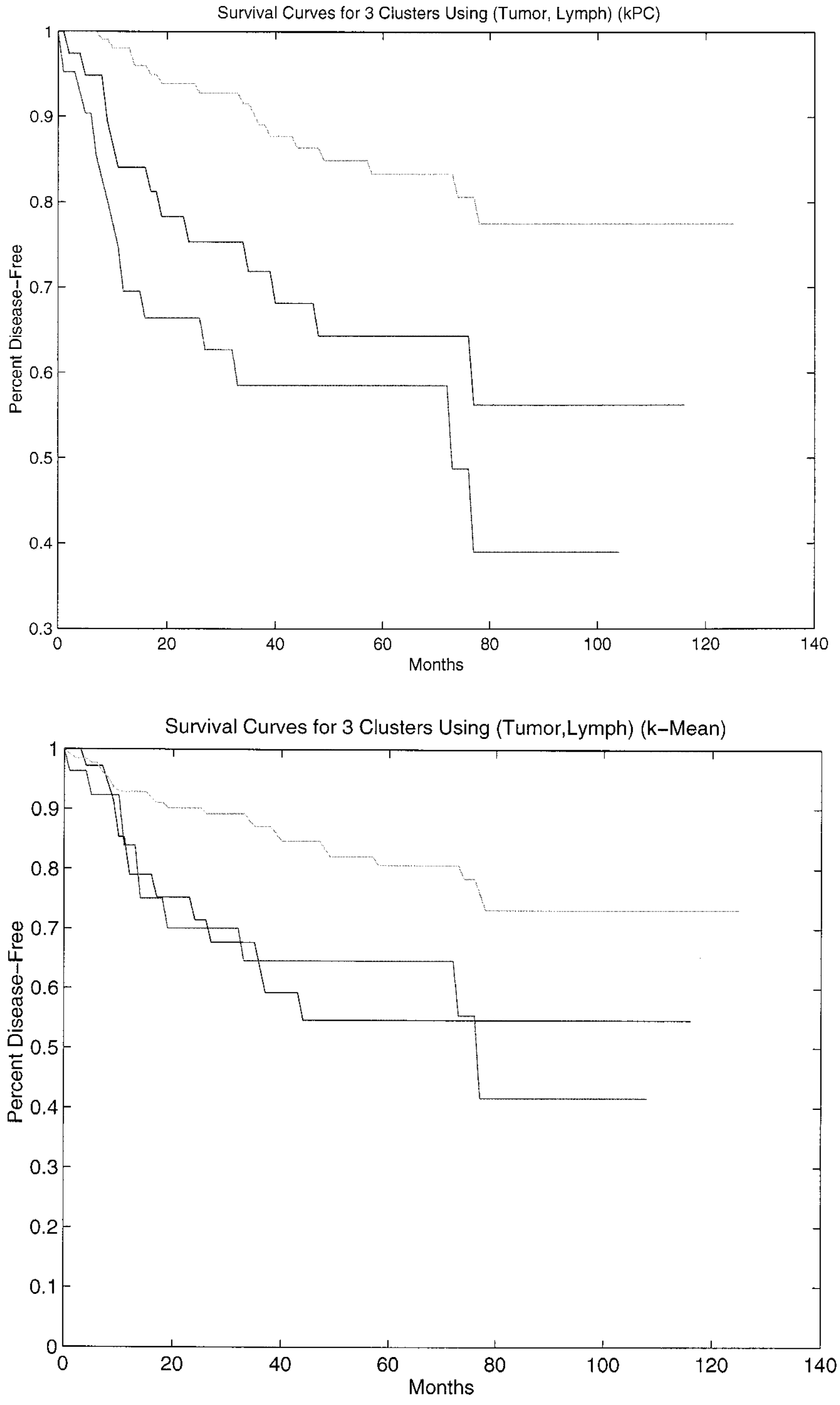

Figure 2. Survival curves for the 3 clusters obtained by kPC and *k*-Mean Algorithms

Table 1. 10-fold Cross-Validation Results

		Ionosphere	BUPA
Ave. Test	kPC	0.6411	0.6503
Correct.	*k*-Mean	0.7060	0.5564
Ave. Train	kPC	0.6410	0.6488
Correct.	*k*-Mean	0.7091	0.5485
Ave. Time	kPC	0.55	0.64
(sec.)	*k*-Mean	2.49	3.98
Ave. # of	kPC	1.0	7.8
Iterations	*k*-Mean	5.6	11.7

5. Conclusion

We have proposed a new clustering algorithm based on minimizing the sum of the squared distances of points to a closest cluster plane instead of the conventional closest cluster center that is used in the *k*-mean algorithm. Clustering around such planes appears to have advantages over clustering around points. One such advantage is well separated survival curves for prognostic data. Other possible advantages include the ability to cluster points that naturally fall into a subspace of the original data space and hence may be better approximated by a plane.

Acknowledgements

This research is supported by National Science Foundation Grants CCR-9322479, CCR-9729842 and CDA-9623632, and by Air Force Office of Scientific Research Grant F49620-97-1-0326 as Mathematical Programming Technical Report 98-08, August 1998.

References

1. Anderberg, M.R., (1973), *Cluster Analysis for Applications.* Academic Press, New York.
2. Andrews, H.C., (1972), *Introduction to Mathematical Techniques in Pattern Recognition.* John Wiley & Sons, New York.
3. Berry, M.W., Dumais, S.T. and O'Brein, G.W., (1995), Using linear algebra for intelligent information retrieval. *SIAM Review*, 37:573–595. http://www.cs.utk.edu/∼berry.
4. Bradley, P.S., Mangasarian, O.L. and Street, W.N., (1997), Clustering via concave minimization. In: M.C. Mozer, M.I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems -9-*, pages 368–374, Cambridge, MA. MIT Press. ftp://ftp.cs.wisc.edu/math-prog/tech-reports/96-03.ps.Z.

5. Cavalier T. and Melloy, B., (1995), An iterative linear programming solution to the Euclidean regression model. *Computers and Operations Research*, 28:781–793.
6. Celeux, G. and Govaert, G., (1995), Gaussian parsimonious clustering models. *Pattern Recognition*, 28:781–793.
7. Fisher, D., (1987), Knowledge acquisition via incremental conceptual clustering. *Machine Learning*, 2:139–172.
8. Hassoun, M.H., (1995), *Fundamentals of Artificial Neural Networks*. MIT Press, Cambridge, MA.
9. Jain, A.K. and Dubes, R.C., (1988), *Algorithms for Clustering Data*. Prentice-Hall, Inc, Englewood Cliffs, NJ.
10. Kaplan, E.L. and Meier, P., (1958), Nonparametric estimation from incomplete observations. *Journal of the American Statistical Association*, 53:457–481.
11. Kleinbaum, David G., (1996), *Survival Analysis*. Springer-Verlag, New York.
12. Mangasarian, O.L., (1999), Arbitrary-norm separating plane. *Operations Research Letters*, 24(1–2). ftp://ftp.cs.wisc.edu/math-prog/tech-reports/97-07.ps.Z.
13. Murphy, P.M. and Aha, D.W., (1992), UCI repository of machine learning databases. Technical report, Department of Information and Computer Science, University of California, Irvine. www.ics.uci.edu/∼mlearn/MLRepository.html.
14. Noble, B. and Daniel, J.W., (1988), *Applied Linear Algebra*. Prentice Hall, Englewood Cliffs, New Jersey, third edition.
15. Rao, M.R., (1971), Cluster analysis and mathematical programming. *Journal of the American Statistical Association*, 66:622–626.
16. Selim, S.Z. and Ismail, M.A., (1984), K-Means-Type algorithms: a generalized convergence theorem and characterization of local optimality. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6:81–87.
17. Späth, H., (1992), *Mathematical Algorithms for Linear Regression*. Academic Press, San Diego.
18. Stone, M., (1974), Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society*, 36:111–147.
19. Strang, G., (1993), *Introduction to Linear Algebra*. Wellesley-Cambridge Press, Wellesley, MA.