Cytometry 14:649–659 (1993)

Cluster Analysis of Flow Cytometric List Mode Data on a Personal Computer

Tom C. Bakker Schut, Bart G. De Grooth, and Jan Greve
Cell Characterization Group, Department of Applied Physics, University of Twente, 7500 AE Enschede, The Netherlands

Received for publication June 23, 1992; accepted January 26, 1993

A cluster analysis algorithm, dedicated to analysis of flow cytometric data is described. The algorithm is written in Pascal and implemented on an MS-DOS personal computer. It uses k-means, initialized with a large number of seed points, followed by a modified nearest neighbor technique to reduce the large number of subclusters. Thus we combine the advantage of the k-means (speed) with that of the nearest neighbor technique (accuracy). In order to achieve a rapid analysis, no complex data transformations such as principal components analysis were used.

Results of the cluster analysis on both real and artificial flow cytometric data are presented and discussed. The results show that it is possible to get very good cluster analysis partitions, which compare favorably with manually gated analysis in both time and in reliability, using a personal computer. © 1993 Wiley-Liss, Inc.

Key terms: Data analysis, k-means, software

Most flow cytometric data is analyzed manually using gated analysis programs like PAINT-A-GATE (11). As the number of parameters that are measured increases, the number of two-dimensional dot plots to be inspected increases as n*(n − 1)/2 (where n is the number of parameters). This means that when measuring only six parameters one should inspect 15 plots, which can be tedious and time consuming, even for simple, well-separated clusters.

Although the concept of applying cluster analysis to flow cytometric list mode data is not new (1,4,7,8) and the possibility of obtaining a more objective data partition has been shown (8), cluster analysis has seen little application in flow cytometry so far. The use of cluster analysis is limited by the fact that it is very difficult to obtain a fast algorithm that works well for all flow cytometric data.

Finding the exact solution to the problem is in most cases impossible, first because in many cases there is no exact solution, and second because finding the exact solution would to some extent involve complete enumeration of all the solutions. The number of possible partitions of n events in m clusters is given by a summation of Stirling numbers of the second kind (2,3). Asymptotically for $n \rightarrow \infty$, this is given by $m^n/n!$. Because of the nature of flow cytometric data (n is of the order 1000–100,000, m is unknown), analysis of all the possible subdivisions (complete enumeration) is virtually impossible.

The amount of data points makes it also impossible to use techniques that involve comparison of every particle with every other particle or techniques using the complete distance matrix of all the particles. Calculation times or memory requirements are then proportional to the square of the number of data points, and therefore these techniques, also seem to be inappropriate to achieve a rapid cluster analysis using a personal computer, at least for large numbers of data points.

To obtain a rapid cluster analysis for large amounts of data, one is limited to partitional clustering techniques, which use only a number of single passes through the data set. This can be realized by beginning with some initial partition and then using iterative methods to reach the end solution. One of the most used iterative partitional clustering methods using an initial partition is called the k-means method.

In the k-means method (5), one takes an initial partition, say K cluster centers, also called seed points, in the n-dimensional space where n is the number of parameters. During the first pass, clusters are formed by attributing all data points to the nearest seed point. One then takes the centroids of the clusters as the new cluster centers. In the second pass all data points are attributed to the nearest cluster center. This procedure is repeated until a more or less stable solution is reached.

After completing the k-means passes, one can adjust the number of clusters by merging and/or splitting

clusters using some kind of validity criterion. Iterative partitional clustering has the obvious advantage that it is very fast compared with more accurate techniques that involve comparison of every particle with every other particle, like the nearest neighbor technique, in which the clusters are agglomerated from single data points and merged until a final partition is reached.

K-means clustering, however, when using normal Euclidean distances, can be very dependent on the initialization, distance measures, and centroid calculations. During the k-means passes, no information about the actual distribution of the data is used. Data points are just assigned to the nearest cluster center. K-means methods using normal Euclidean distances and seed point numbers are bound to give wrong results for data distributions in which the parameters are correlated or for data distributions with populations that are close to each other and have different standard deviations and numbers of cells.

These problems can be reduced by using data transformations, principal component analysis, or special distance functions, like the Mahalanobis distance (6). These procedures generally require extensive calculations or knowledge of the data structure (2,3).

In this article we investigated another possible solution by combining the k-means, initialized with a large number of seed points, with a modified nearest neighbor technique. The use of a large number of seed points could significantly reduce the number of mistakes made by the k-means. A more accurate nearest neighbor-like technique was then used to reduce the large number of small clusters found by the k-means.

Our results indicate that the k-means algorithm, initialized with a large number of equally spread seed points, can give good results, even for overlapping populations and hidden distributions.

MATERIALS AND METHODS

The cluster analysis algorithm basically consists of the following three steps: initialization, k-means cluster formation, and merging of the subclusters found.

In our implementation the k-means algorithm is initialized by choosing within the data set a given number of seed points that are to some extent remote from each other. The algorithm takes the first data point as the first seed point and then runs through the rest of the data set, making new seed points of every data point that has at least a minimum sup distance to the seed points already found. The sup distance is defined by the maximum one-dimensional distance in any of the parameters considered during clustering (3). The minimum sup distance is initially made half the maximum value of a parameter. If there are not enough seed points found, this minimum sup distance is divided by 2, and the process is repeated, keeping the seed points that have been found. This whole procedure is repeated until the required number of seed points is found. The advantage of this procedure is that one gets a more or less even distribution of seed points within the data distribution, especially when working with many seed points.

During the k-means passes, all centroid calculations are made using normal Euclidean distances. In this case k-means works best for clusters whose absolute spread in a certain dimension is independent of their distance to the origin in that dimension. For linearly measured parameters, the absolute spread for a certain population will be proportional to the amplification used. In order to get a scale-independent distance measure, a transformation of linearly measured parameters to a logarithmic scale was used.

The k-means passes are stopped when the number of changes between two consecutive passes become smaller than some preset percentage, the so-called stop criterion. This is done in order to prevent instabilities at the end, caused by cells that are on the edge of two adjacent clusters.

After the k-means passes the numbers of clusters are reduced by merging clusters. The criterion for joining two clusters and the order in which to join them is a very important parameter. For good results the data distribution within a cluster should be taken into account. Criteria that involve the individual data points of a cluster would take too much time; therefore only criteria that involve the statistical properties of the clusters can be used. One can use the distance between the cluster centers to determine the order for joining and only join two clusters if for either cluster the means for all parameters fall within x times the standard deviation (SD) of the other cluster (with x an arbitrary constant). This procedure gives good results if the parameters are not correlated and if the populations are well separated, as has been shown by Murphy (8). A less stringent criterion would be to join only if the means for all parameters fall within x times the SD for at least one of the two clusters (with x an arbitrary constant). Because we could not get stable results using these criteria, we experimentally devised a slightly different joining criterion. It was planned to decrease with the number of particles in order to avoid large cluster abosrbing small clusters lying on the edge, and to ensure that neighboring, less dense clusters with large spreads are merged. Because the clusters are often not symmetrical, we calculated for every cluster in all dimensions two "modified spread values": one for the particles with a higher parameter value than the cluster average and one for the particles with a lower value. This modified spread is given by $SD/\sqrt{N}$, with N the number of particles that contribute to the spread. As a "modified distance" measure between two clusters in a certain dimension, we take the distance between the cluster centers in that dimension minus the sum of the modified spread values in that dimension. For each cluster the nearest neighbor is calculated using this modified distance. The two clusters that have the lowest calculated modified distance are now joined. When two clusters have been joined, the minimum modified

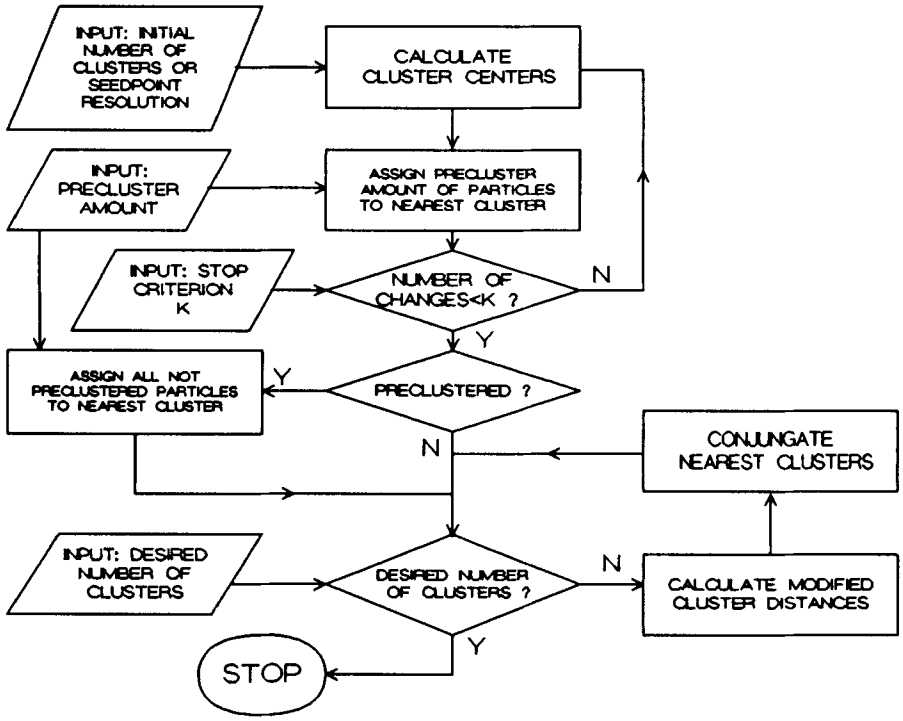

FIG. 1. Flowchart of a typical run of the cluster analysis program AUTOKLUS, described in this article.

distance of the new cluster to all the existing clusters and the minimum modified distances of all clusters that had one of the joined clusters as nearest neighbor are recalculated. The joining of clusters is continued until the final number of large clusters, specified by the user, is reached. We did not implement an automatic stop because we could not find one that gave stable results for all types of flow cytometric data.

In order to get a fast (but less accurate) result, we implemented the possibility of preclustering, i.e., taking only a certain amount of the data (precluster amount) into account during the k-means passes. The data that have not been considered during k-means are assigned to the nearest cluster center directly after the k-means passes.

The algorithm was imbedded in a menu-drived program (called AUTOKLUS), written in Turbo Pascal (Version 6.0, Borland Int. Inc., Scotts Valley, CA). AUTOKLUS was written and examined using a 80486 33 MHz personal computer (Sirex, Enschede, The Netherlands), but it can run on any MS-DOS computer that has an EGA/VGA resolution screen. The program can read FCS 2.0 list mode data files. Source and executable code are available from the authors at a small charge.

As our objective was to investigate the results that the same clustering procedure would have on different types of data, we did not make it interactive with the user. A number of parameters can be changed, but not during the clustering process. A flowchart of a typical ryn is given in Figure 1. When starting the cluster analysis, the program asks for the number of particles to be clustered, the desired number of clusters, the number of seed points, the precluster amount, and the percentage of changes at which the k-means passes are to be stopped. The choice of parameters that are considered during clustering can be changed in a separate menu. The maximum number of parameters that can be considered is eight; parameter values can have a range of 0–255. The maximum number of seed points is 255. The maximum number of events that can be clustered is 65,535. In principle, the number of parameters and the number of events can be increased and are only restricted by the amount of working memory available. In the program, a RAM drive is used for working memory, but in principle one can also use a hard disk.

Both artificial and real data were used to evaluate the cluster analysis program. The artificial data were generated using a Gaussian random generator that can produce clusters with specified means and standard deviations for eight (uncorrelated) parameters. These data were also used to ensure proper coding of the algorithms used. Six parameter data, combined scatter and fluorescence, of stained blood were measured with a home-made flow cytometer. Five parameter data of neutrophil maturation in bone marrow, which are similar to the data published in Terstappen et al. (12), were provided by Dr. L.W.W.M. Terstappen.

For evaluating the influence of the different cluster analysis parameters, we compared the results with that of a manually gated analysis program. In order to quantify the quality of a cluster analysis result, we

tentatively defined a parameter, the relative error in the cluster analysis result, RE, as:

$$RE = 100 * \sum_{c=1}^{K} \Delta_c / N$$

N is the total number of cells, and Δ_c is the number of cells that are assigned to cluster C in the manually gated analysis result but are assigned to another population when determined by cluster analysis; K is the number of clusters with more than 1% of the total amount of cells in the manually gated analysis result.

No comparison with results of other clustering programs was made. The only comparison was done by implementing the SD joining algorithm described by Murphy (8) and comparing it with our own joining algorithm.

RESULTS

The influence of the different cluster parameters in the algorithm was tested on six real parameter data (4,096 cells), obtained with a home-made flow cytometer described elsewhere (9). Whole blood was lysed and immediately afterwards stained with CD4-FITC (fluoresceinisothiocyanate), CD8-PE (phycoerythrine), and LDS-1751(EXITON, Dayton, OH) as a live-dead indicator, following a procedure described elsewhere (10).

Figure 2A shows six dot plots of the raw data. Parameter 1 is the forward light scatter, parameter 2 is normal orthogonal light scatter, parameter 3 is depolarized orthogonal light scatter, parameter 4 is FITC fluorescence, parameter 5 is PE fluorescence, and parameter 6 is LDS fluorescence. Figure 2B shows the manually gated analysis results for these data. Cell type 1 (light blue) represents neutrophilic granulocytes, cell type 2 (purple) lymphocytes, cell type 3 (red) red blood cells and debris, cell type 4 (dark blue) CD4-positive lymphocytes, cell type 5 (yellow) CD8-positive lymphocytes, cell type 6 (light green) monocytes, cell type 7 (dark green) eosinophilic granulocytes, and cell type 8 CD4-CD8-positive cells. This manually gated analysis result served as a reference to determine the total relative error in the cluster analysis results as defined in Materials and Methods. In order to obtain a valid reference, cells that could not be clearly assigned to one special cluster were left out. Cells belonging to very small (<<1%) clusters were also left out (limitation of the manually gated analysis program). The manually gated analysis result contains 3,907 of the original 4,096 cells, grouped in eight clusters. For evaluation of the cluster analysis results, these 3,907 cells are used. For all cluster analysis results described here we used the following parameters (unless indicated else): 4,096 data points, a stop criterion of 1%, and no preclustering, joined in order of smallest modified distance until seven large clusters remained (based on the outcome of the manually gated analysis).

A typical cluster analysis result is shown in Figure 2C; 75 seed points were used, and calculation time was 117 s using a 33 MHz 80486 MS-DOS PC.

First we investigated the influence of the number of seed points, used for initialization of the k-means, on the cluster analysis result, both with and without logarithmic transformation of the linearly measured parameters. Figure 3 shows the relative error in the cluster analysis result (left scale) as a function of the number of seed points. In the same plot the calculation time for the clustering with logarithmic transformation is shown (right scale). The figure clearly shows that clustering using the logarithmic transformation of the linearly measured parameters gives the best result: good clustering with a fault of less than 1% can be achieved within 50 s.

After the k-means, the subclusters are joined in a particular order until some stop criterion is reached. As described in Materials and Methods, we experimentally devised a joining order based on the smallest modified distance; the joining of clusters is stopped if the total number of big (> 1%) clusters is equal to the desired number of clusters, specified by the user. We compared our joining procedure with the two SD related joining procedures, also described in Materials and Methods. Figure 4 shows the influence of these three different joining criteria on the relative error as a function of the number of seed points. The solid line gives the results using the smallest modified distance: joining is stopped at seven large clusters. The broken lines give the results for joining in order of smallest distance and means within x*SD for either cluster. Results for both x = 3 and x = 3.5 are shown (best results). The dotted lines give the results for joining in order of smallest distance and means within x*SD for at least one of the two clusters. Results for both x = 1.5 and x = 2 are shown (best results). Figure 4 clearly shows that our joining procedure gives less faults than the two SD related joining procedures.

The calculation time of the algorithm is, when many seed points are used, mainly determined by the k-means. It can be reduced by making the k-means stop at a certain number of changes between two consecutive passes. The influence that the stop criterion of the

FIG. 2. **A:** Six-parameter data of whole blood that was lysed and immediately afterwards stained with CD4-FITC, CD8-PE, and LDS as a live-dead indicator. Parameter 1 is the forward light scatter, parameter 2 is normal orthogonal light scatter, parameter 3 is the depolarized orthogonal light scatter, parameter 4 is FITC fluorescence, parameter 5 is PE fluorescence, and parameter 6 is LDS fluorescence. **B:** Manually gated analysis result on data presented in A; cells that cannot be clearly assigned to one special cluster and cells that belong to clusters much smaller than 1% of the total amount are left out. Cell type 1 (light blue) represents neutrophilic granulocytes, cell type 2 (purple) lymphocytes, cell type 3 (red) red blood cells and debris, cell type 4 (dark blue) CD4-positive lymphocytes, cell type 5 (yellow) CD8-positive lymphocytes, cell type 6 (light green) monocytes, cell type 7 (dark green) eosinophilic granulocytes, and cell type 8 CD4-CD8-positive cells. **C:** Cluster analysis result on the data presented in A. The number of data points is 4,096, the number of seed points is 75, stop criterion is 1%, no preclustering is used, k-means clusters were joined in order of smallest modified distance, and joining was stopped at seven big clusters. Execution time (using a 33 MHz 86486 MS-DOS PC) was 117 s.

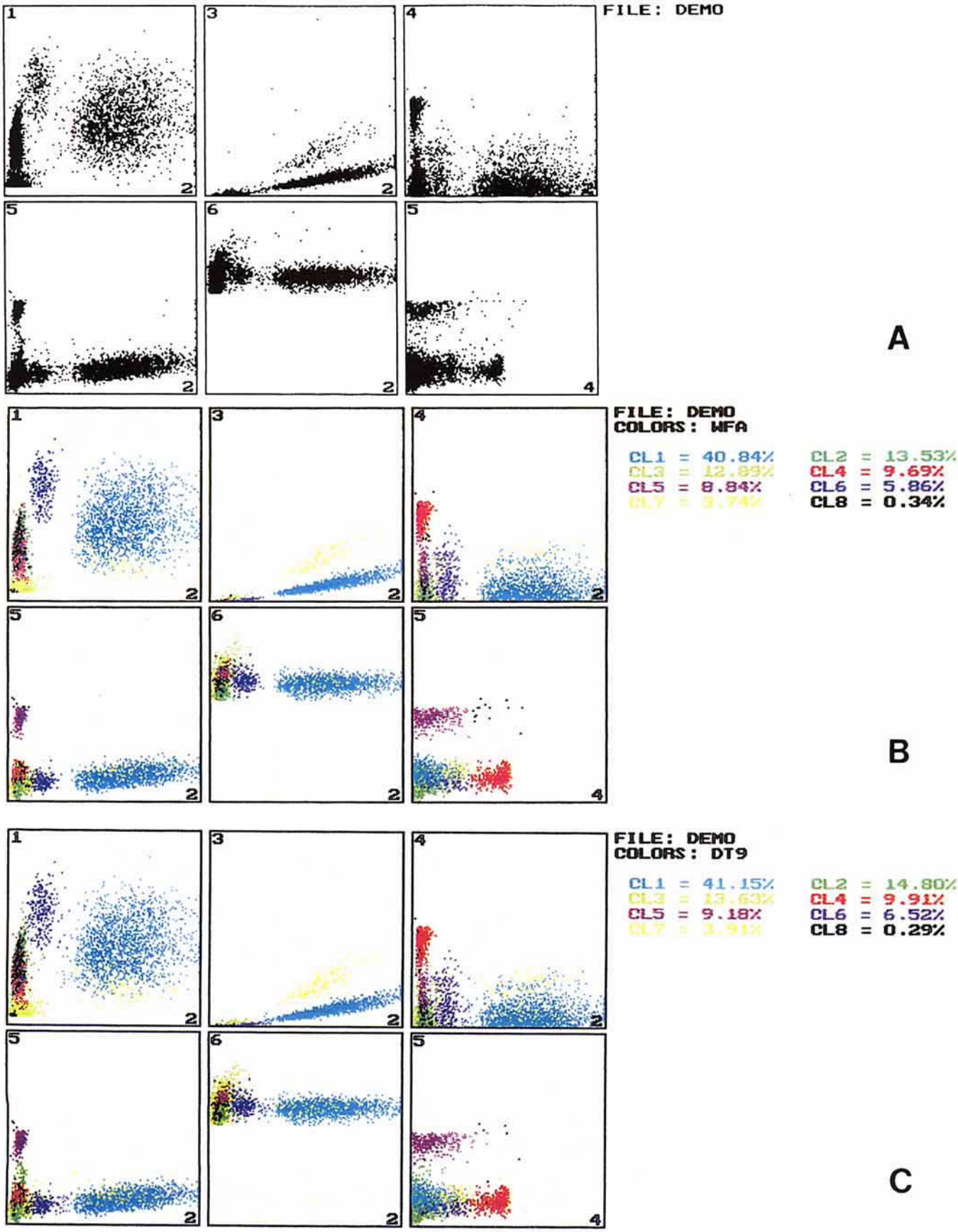
FILE: DEMO
A
FILE: DEMO
COLORS: WFA
CL1 = 40.84%
CL2 = 13.53%
CL3 = 12.89%
CL4 = 9.69%
CL5 = 8.84%
CL6 = 5.86%
CL7 = 3.74%
CL8 = 0.34%
B
FILE: DEMO
COLORS: DT9
CL1 = 41.15%
CL2 = 14.80%
CL3 = 13.63%
CL4 = 9.91%
CL5 = 9.18%
CL6 = 6.52%
CL7 = 3.91%
CL8 = 0.29%
C

FIG. 2.

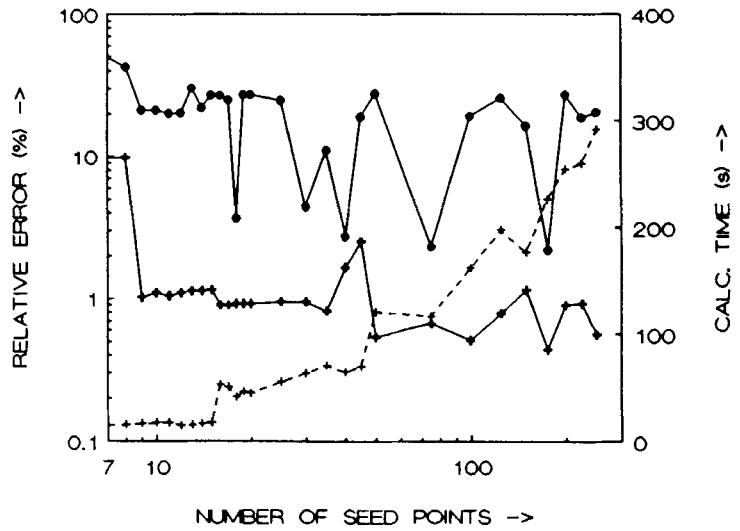

FIG. 3. Relative error in the cluster analysis result (compared with the manually gated analysis result shown in Fig. 2B) and the computing time as a function of the number of seed points used for initialization of the k-means. The solid line with crosses represents the relative error when a logarithmic transformation of the linearly measured parameters is used (left scale); the solid line with circles represents the relative error when no logarithmic transformation is used (left scale); and the broken line with crosses represents the calculation time of the algorithm as a function of the number of seed points for the k-means when the logarithmic transformation is used (right scale). Stop criterion is 1%, no preclustering is used, and joining is stopped at seven big clusters.

k-means has on the relative error and the computing time is shown in Figure 5. In all analyses seven large clusters were requested. The solid lines give the cluster analysis results for different numbers of seed points,

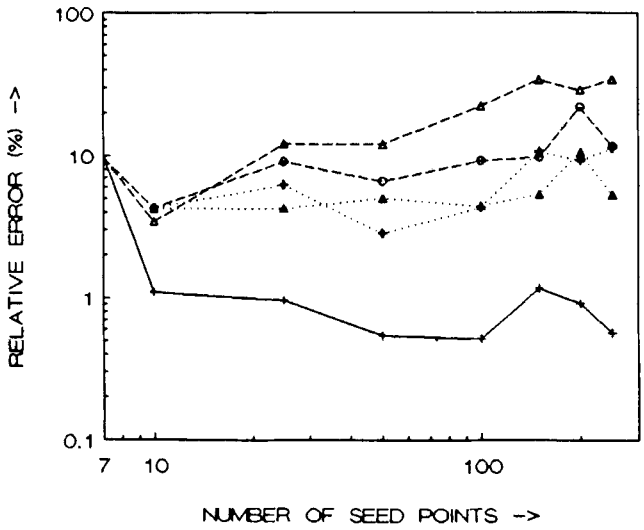

FIG. 4. Relative error in the cluster analysis result (compared with the manually gated analysis result shown in Fig. 2B) as a function of the number of seed points used for the initialization of the k-means for different joining criteria. Stop criterion is 1%, and no preclustering is used. Solid line, smallest modified distance, stop at seven large clusters; broken line and triangles, smallest distance and means within x*SD for either cluster with x = 3.0; broken line and circles, smallest distance and means within x*SD for either cluster with x = 3.5; dotted line and crosses, smallest distance and means within x*SD for at least one of the clusters with x = 1.5; dotted line and triangles, smallest distance and means within x*SD for at least one of the clusters with x = 2.

and the broken lines give the corresponding calculation times. The figure shows that for large numbers of seed points, a less stringent stop criterion is sufficient to give good results, and it significantly reduces the calculation time.

To enhance the speed of the algorithm, we further implemented the possibility of letting the k-means work on only a part of the data (preclustering). We found that the amount of data used during the k-means could be reduced to 50% without giving rise to a large increase in errors. This is illustrated in Figure 6, in which the relative error and the calculation time are given as a function of the amount of data that is used for preclustering, for different numbers of seed points.

The ability of the algorithm to analyze complicated flow cytometric data was tested on three-color immunofluorescence data showing the maturation of neutrophils in normal bone marrow; these data are similar to the data used by Terstappen et al. (12). The cluster analysis result for these data is shown in Figure 7. Clustering was done on 20,000 cells using 200 seed points; 15 large clusters were requested. Figure 7A shows six plots of all the clusters; because the separate clusters can hardly be recognized in these plots, Figure 7B shows these six plots for the largest 15 clusters separately; the cluster indicated is colored black, and the rest of the data is colored gray. The results show a pattern similar to that reported by Terstappen et al. using manually gated analysis.

DISCUSSION

Application of k-means algorithm to flow cytometric data seems to be more justified by the speed of the

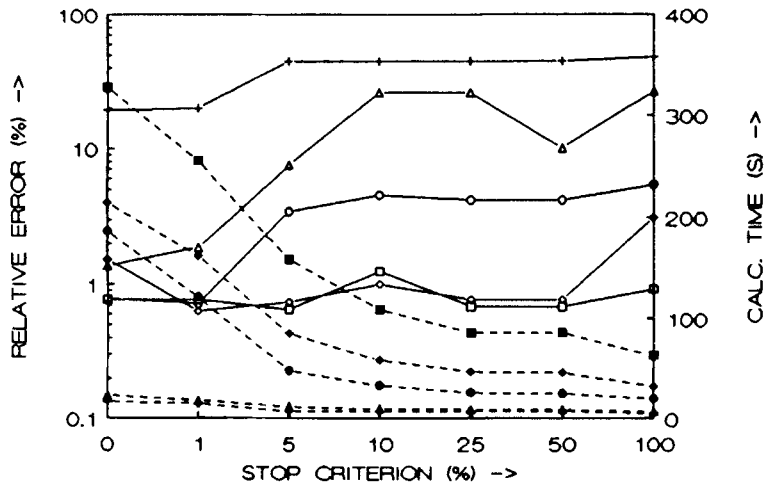

FIG. 5. Relative error in the cluster analysis result (compared with the manually gated analysis result shown in Fig. 2B) and the corresponding computing time as a function of the value of the stop criterion for the k-means; graphs for different numbers of seed points, used for the initialization of the k-means, are shown. Solid lines represent relative errors (left scale), and broken lines represent calculation times (right scale); crosses, seven seed points; triangles, ten seed points; circles, 50 seed points; diamonds, 100 seed points; squares, 200 seed points. Data points (4,096) were clustered, no preclustering was used, k-means clusters were joined in order of smallest modified distance and joining was stopped at seven big clusters.

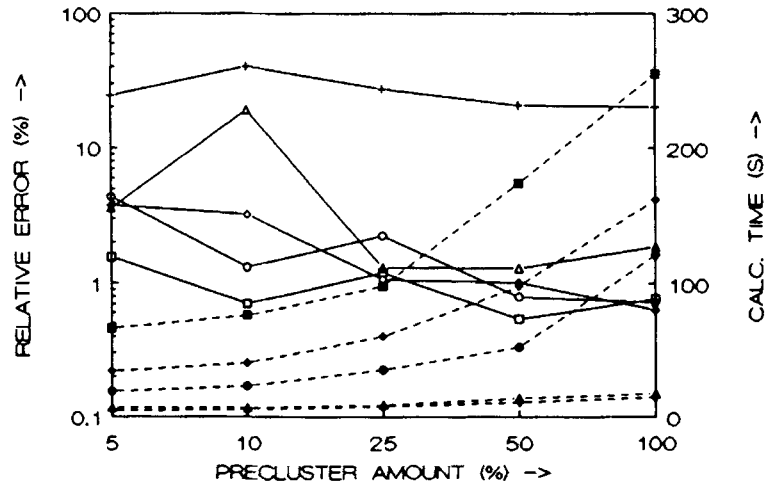

FIG. 6. Relative error in the cluster analysis result (compared with the manually gated analysis result shown in Fig. 2B) and the corresponding computing time as a function of the amount of data used for the k-means (preclustering); graphs for different numbers of seed points, used for the initialization of the k-means, are shown. Solid lines represent relative errors (left scale), and broken lines represent calculation times (right scale); crosses, seven seed points; triangles, ten seed points; circles, 50 seed points; diamonds, 100 seed points; squares, 200 seed points. Data points (4,096) were clustered, the stop criterion was set to 1%, k-means clusters were joined in order of smallest modified distance, and joining was stopped at seven big clusters.

algorithm than by the suitability for this type of data. For overlapping, correlated, and unevenly distributed populations, common in flow cytometry data, much effort has to be made to achieve a reasonable partition using the k-means algorithm. The use of a more accurate technique, like nearest neighbors clustering, is not suitable because of the amount of data, typical for flow cytometric measurements. Our results show that cluster analysis using k-means, initialized with a large number of seed points, followed by a nearest neighbor-like technique to merge the subclusters, can produce good results for flow cytometric list mode data in a reasonable time, using only a personal computer. This is illustrated by Figure 3, in which the relative error in the cluster analysis result is given as a function of the number of seed points.

The relative error is, to some extent, a subjective parameter. It is highly dependent on the manually gated reference analysis. We used a filtered reference, in which all questionable events were left out, in order

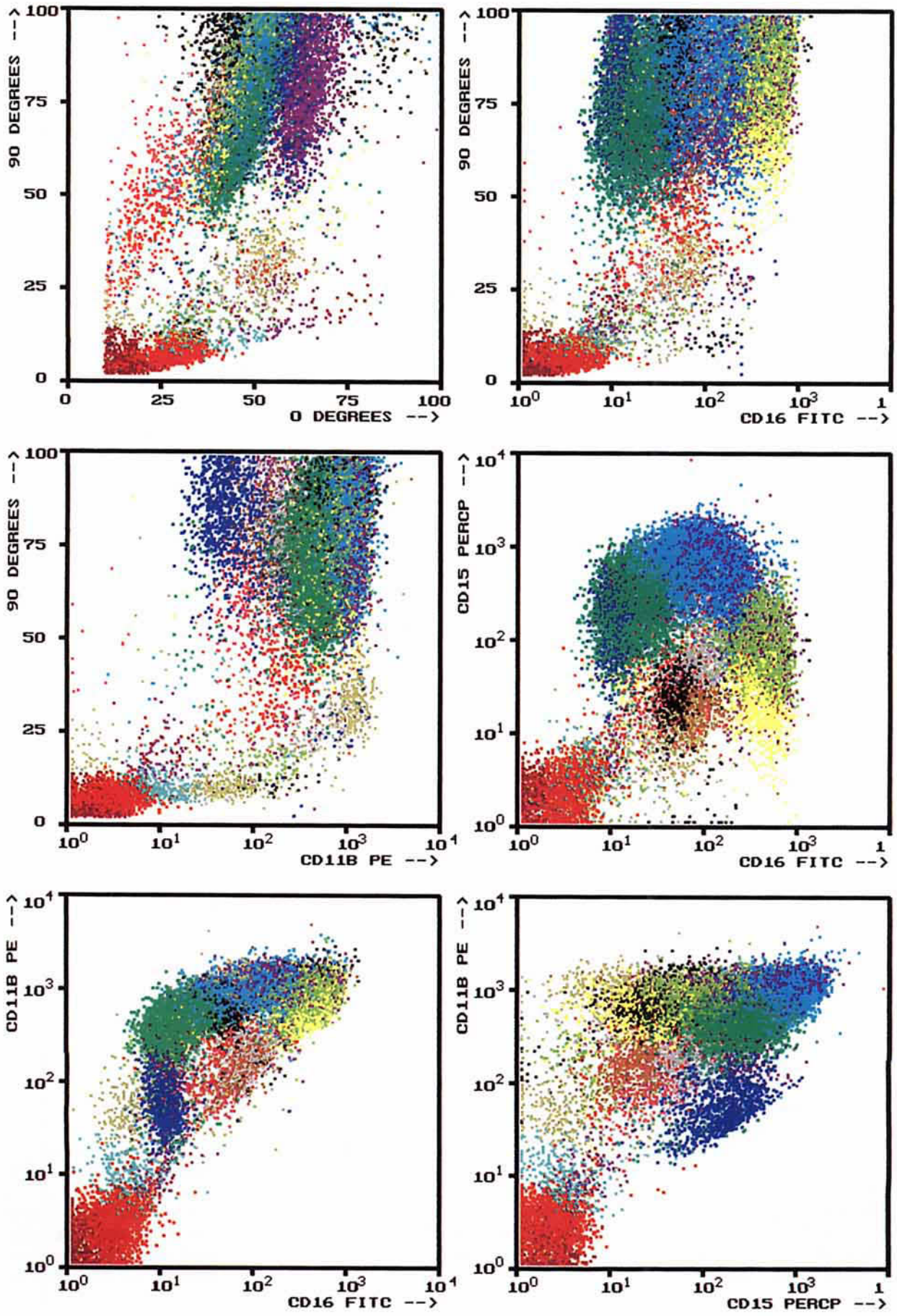

FIG. 7A.

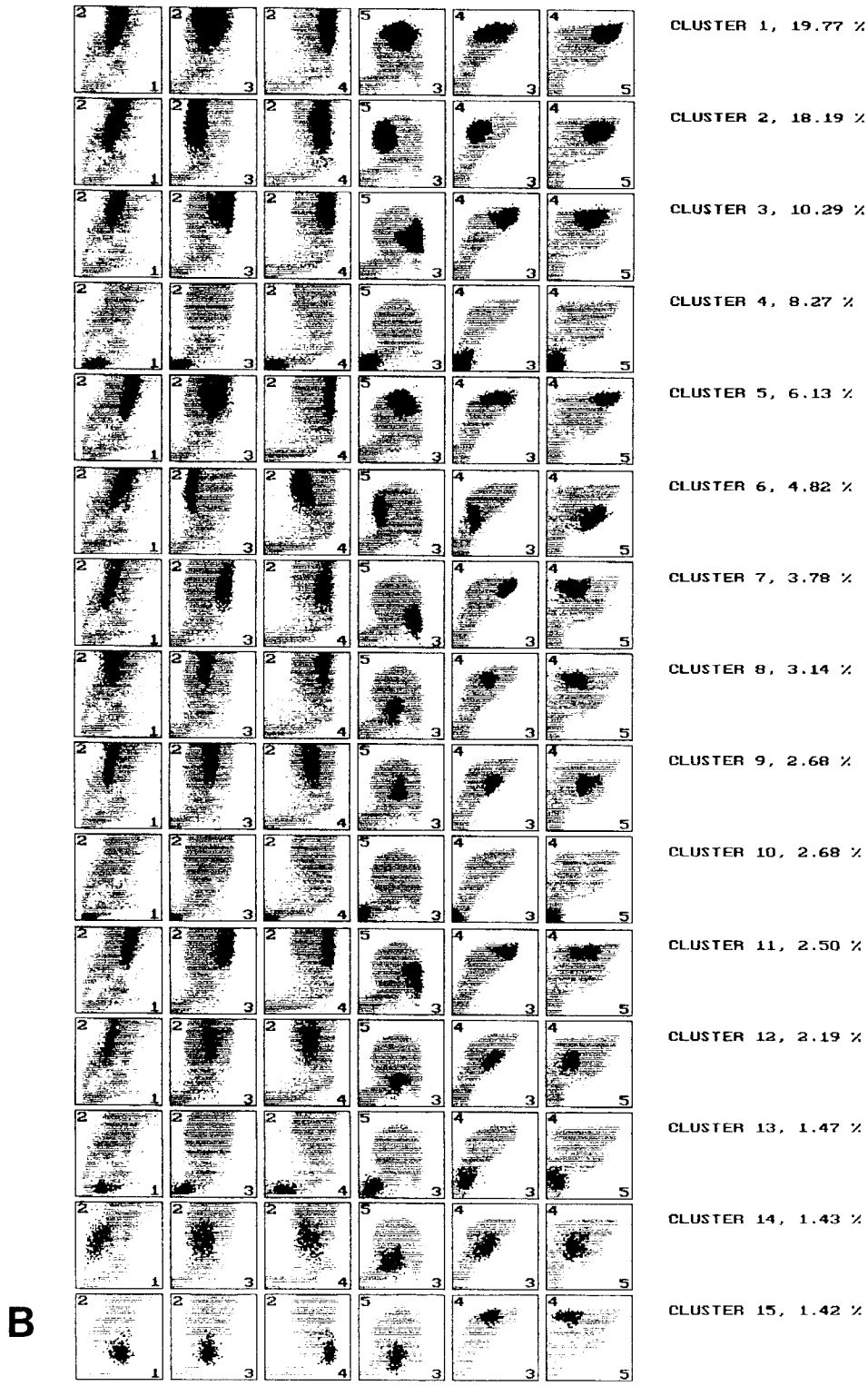

FIG. 7. Cluster analysis result on data showing neutrophil maturation. Clustering was done using 200 seed points, 20 clusters were requested, the stop criterion was set to 1%, k-means clusters were joined in order of smallest modified distance, and no preclustering was used. **A:** Six dot plots of all the clusters. **B:** Same six dot plots as A but now for the 15 biggest clusters separately; the cluster is colored black, and the rest of the data is colored gray.

to get a measure for the apparent mistakes made during cluster analysis. One could object that, if such a reference is used, cluster analysis should also be done using the filtered data set. If we were to use only these data, the clusters would have been clearly separated, and few mistakes would have been made, thus making the evaluation trivial. Cluster analysis was done also using the questionable events in order to investigate whether adjacent clusters disturb the cluster algorithm so that apparent mistakes are made.

Our definition of the relative error does not discriminate between "clear errors" and "uncertainties." In Figure 3 one can see that the relative error in the cluster analysis result is about 1% when only nine seed points are used. This result, however, contains many clear errors, whereas the results using more than 50 seed points contain more uncertainties, i.e., small clusters at the edge of large clusters that stay apart during clustering. Because of these uncertainties, the relative error does not completely reach zero.

In the data set, using large numbers of seed points that are remote from each other has a number of advantages. First, the seed points do not move far before reaching the end solution; one can even get good results using only one k-means pass if many seed points are used (see Fig. 5, stop criterion at 100%). Second, outliers will get their own seed points and will not influence the nearest clusters.

When many seed points are used, the number of possible instable points is large, and therefore one needs a certain stop criterion. The criterion we chose for our evaluation of the algorithm does not alter the outcome of the analysis much, as can be seen in Figure 5. It was used in the evaluation of the algorithm to show that good results can be obtained in a time that is competitive with manual analysis.

Due to the large number of subclusters, the most critical part of the algorithm is the criterion for joining two clusters and stopping the merging of clusters. Many authors use a joining criterion that is somehow related to the standard deviation (2). Clusters are said to overlap if their means are within x times the standard deviations of either cluster, where x is an arbitrary multiplication constant. If there are no more clusters that satisfy that criterion, the algorithm is stopped. Murphy (8) points out that the results can be very dependent on the value of that multiplication constant; he had the most reproducible results using a multiplication constant of 2. Our own experience is that normal standard deviation related measures fail when there is correlation between parameters or when there is overlap between different clusters; clusters with a large spread will then tend to absorb adjacent clusters. Both the normal standard deviation related joining criteria we investigated are also not well suited for joining very small subclusters, as can be seen from Figure 4: the relative error in the cluster analysis results rises with the number of seed points. The main reason for this high relative error is that the joining procedures stop at a wrong number of clusters. In the case of the data shown in Figure 2A, errors occurred especially in the separation of the eosinophilic and neutrophilic granulocytes (yellow and blue clusters in Fig. 2B and C).

It is very difficult to define a simple criterion that gives a valid decision for joining two clusters that works for all types of data. Often the distance between two cluster centers seems to be the most valid determination factor. The joining criterion we used is devised to protect widespread clusters from absorbing everything in their surroundings as their modified spread decreases with the number of particles. Because of the dependency of our joining criterion on the number of events in a cluster, one might suspect that samples consisting of the same populations in different proportions would give different results. This is true only when few seed points are used. If a large number ofseed points is used, the cluster analysis result is not really dependent on the relative numbers in the clusters but more on the distance between the cluster centers.

Compared with joining in order of smallest distance, our joining method gives better results for separating less dense widespread clusters at the edge of dense clusters. Therefore it seems to be a good joining criterion in order to avoid the chaining tendencies one gets when merging many little clusters. Using this criterion, we were able to separate overlapping clusters and hidden distributions.

When to stop the merging of clusters is a very difficult problem, especially for overlapping clusters. For well-separated clusters, simple SD related criteria seem to be satisfactory, but for overlapping or adjacent clusters these criteria give false results, as shown in Figure 4. We therefore did not implement an automatic stop criterion but let the user specify the final number of big clusters.

The calculation time required for the k-means is linearly dependent on the number of data points, but it is not a simple function of the number of seed points. The algorithm can become unstable in reaching the final partition for large numbers of seed points, but using a termination criterion that ends the k-means if the number of changes is smaller than a few percent solves this problem and does not have much influence on the final results, as can be seen from Figure 5. Using a larger stop criterion further reduces the calculation time, but then more mistakes occur, especially for small numbers of seed points. The calculation time can be further reduced by using only a certain amount of the data in the k-means part (preclustering) without giving rise to large errors, as can be seen from Figure 6. Execution times are still reasonably low using an 80486 33 MHz microprocessor computer and could be decreased using macro-routines for the calculational parts of the program. Execution times on an 80386 and an 80286 will be longer.

The results on the quick lysed blood show that cluster analysis might be very useful for automated immu-

nofluorescence analysis; the calculation time for a good cluster analysis result using 4,096 data points is about 50 s and is therefore less than the time needed for lysis and immunostaining.

The algorithm is also able to detect patterns in complicated immunofluorescence data like the data shown in Figure 7. When using manually gated analysis on this type of data, it is very difficult and time consuming to identify all subpopulations. The results show that cluster analysis can give very good results for these types of data. It is even possible to find cohesive subpopulations within the "smear" that is present in the data. These subpopulations show a maturation pattern similar to the pattern that Terstappen et al. suggested (12). It should be noted that the exact borders of the subpopulations, in case they are lying next to each other in all the parameters, are to some extent subjective; initialization with different parameters can give slightly different subpopulations. Nevertheless, the resulting analysis can be very helpful in visualizing certain patterns, present in the data.

In conclusion, our results show that it is possible to get very good cluster analysis partitions that compare favorable with manually gated analysis in both time and in reliability using a personal computer. Improvements can be made by finding good criteria for cluster validity in flow cytometric data.

LITERATURE CITED

1. Dean PN: Data processing. In: Flow Cytometry and Sorting, Ed. 2, Melamed MR, Mullaney PF, Mendelsohn ML (eds.) John Wiley & Sons, New York, 1990, pp 415–444.
2. Duran BS, Odell PL: Cluster Analysis, a Survey. Springer-Verlag, Berlin, 1974.
3. Jain AK, Dubes RC: Algorithms for Clustering Data. Prentice Hall, Englewood Cliffs, NJ, 1988.
4. Kosugi Y, Sato R, Genka S, Shitara N, Takakura K: An interactive multivariate analysis of FCM data. Cytometry 9:405–408, 1988.
5. MacQueen J: Some methods for classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkeley Symposion on Mathematical Statistics and Probability, Le Cam LM, Neyman J (eds). University of California Press, Berkeley, 1967, pp 281–279.
6. Mahalanobis PC: On the generalized distance in statistics. Proc Natl Inst Sci India 2:49–55, 1936.
7. Mann RC: On multiparameter data analysis in flow cytometry. Cytometry 8:184–189, 1987.
8. Murphy RF: Automated identification of subpopulations in flow cytometric list mode data using cluster analysis. Cytometry 6:302–309, 1985.
9. Terstappen LWWM, de Grooth BG, Nolten GMJ, ten Napel CHH, van Berkel W, Greve J: Physical discrimination between human T-lymphocyte subpopulations by means of light scattering revealing two populations of T8 positive cells. Cytometry 7:178–183, 1986.
10. Terstappen LWWM, Loken MR: Five-dimensional flow cytometry as a new approach for blood and bone marrow differentials. Cytometry 9:548–556, 1988.
11. Terstappen LWWM, Shah VO, Datillo KL, Civin CI, Hurwitz CA, Loken MR: Multidimensional flow cytometry as a new approach for discrimination between normal and leukemic cells in peripheral blood and bone marrow. In: Progress in Cytometry—Flow and Image. Beckton Dickinson, Erebodegem, 1989, pp 4–29.
12. Terstappen LWWM, Safford M, Loken MR: Flow cytometric analysis of human bone marrow III. Neutrophil maturation. Leukemia 4:657–663, 1990.