A walk through the SigClust algorithm

In a 2008 paper[1] Liu et al. present the clustering algorithm SigClust. The algorithm takes as a framework the assumption that clusters are by definition, generated by single, multivariate Gaussian distributions. Thus the algorithm considers the null hypothesis that the data in question (viewed as the rows of $X \in \mathbb{R}^{m \times n}$) come from a single, multivariate Gaussian. A translation and rotation invariant test static ($CI_2(\cdot)$, the k-means cluster index for $k = 2$) is used, and thus we can assume without loss that the distribution of the null hypothesis is distributed as $D_0 \sim N(0, \Sigma_0)$ where Σ_0 is a diagonal matrix. This matrix is estimated from data, taking into account some symmetric (fixed σ^2), multivariate Gaussian background noise. Given D_0, one can estimate a distribution on the test statistic $CI_k(Z)$ on datasets Z generated by D_0. From here it is straightforward to compute the p-value of the CI_k of the clustered *original* dataset with respect to this CDF. When the p-value is below some threshold α, the null hypothesis is rejected. The data may then be clustered as desired, in particular one may use the two clusters given when computing $CL_2(X)$.

Here we will present a short tour of SigClust. Since Liu et al. break the algorithm down into seven steps, we will devote a small section to each step (after making some initial definitions).

Definition 1 (Preliminary)**.** *Throughout let $X \in \mathbb{R}^{m \times n}$ be a data matrix for a dataset with m samples and n features. So for each $1 \leq i \leq m$, the ith* row

$$X^i := X(i, \cdot)$$

gives all the feature values for the ith sample*, and the for each $1 \leq i \leq n$ the ith* column

$$X_i := X(\cdot, i)$$

gives a vector of data for the ith feature*. Also, let*

$$rows(X) := \{X^i : 1 \leq i \leq m\}$$

[1] Yufeng Liu, David Neil Hayes, Andrew Nobel and J. S. Marron, *Statistical Significance of Clustering for High-Dimension, Low-Sample Size Data* Journal of the American Statistical Association, Vol. 103, No. 483 (Sep., 2008), pp. 1281-1293
See also http://arxiv.org/abs/1305.5879v2

refer to the set of all m of the $1 \times n$ row vectors of X and let

$$cols(X) := \{X_i : 1 \leq i \leq n\}$$

denote the n, $m \times 1$ column vectors of X.

Definition 2 (The k-means Cluster Index (CI_k))**.** *Let $k, m, n \in \mathbb{Z}^+$ (positive integers).*

Given $X \in \mathbb{R}^{m \times n}$, the k-means algorithm delivers a partition $\mathscr{Q}$ of $rows(X)$ with $|\mathscr{Q}| = k$ and determines a corresponding partition

$$\mathscr{P}_k(X) := \langle C_1, C_2, \ldots, C_k \rangle$$

of the index set $I = \{1, 2, \ldots, m\}$.

Given the latter partition, we define the k-means cluster index *for X as*

$$CI_k(X) = \frac{\sum_{j=1}^{k} \sum_{i \in C_j} \|X^i - c_j\|^2}{\sum_{i=1}^{m} \|X^i - \bar{X}\|^2}$$

where $\|\cdot\|$ here is just the Euclidean norm for one dimensional (row or column) vectors,

$$\bar{X} := mean(rows(X)),$$

and for each $1 \leq i \leq k$

$$c_i := mean(X(C_i, \cdot))$$

is the mean of all rows of X with row index in C_i.

Lin et. al. note that "The smaller the CI, the larger the proportion of the overall variance that is explained by clustering."

Also, it is easy to see that this function is invariant under translation and isometric transformation of the rows of the input data matrix.

Lemma 3. *Let 1_m be the $m \times 1$ column vector of ones.*
For any data matrix $X \in \mathbb{R}^{m \times n}$, any row vector $\mu \in \mathbb{R}^n$, and any isometry $M \in \mathscr{O}(n)$ we have

$$CI_k(XM + 1_m\mu) = CI_k(X)$$

Proof. (Partial) This should be geometrically clear, but we give a partial proof of invariance under isometric transformation and leave invariance under translation to the reader. Let $M \in \mathscr{O}_n$, and let

$$Y = XM = \begin{bmatrix} X^1 \\ X^2 \\ \vdots \\ X^m \end{bmatrix} \begin{bmatrix} M_1 & M_2 & \cdots & M_n \end{bmatrix}$$

so that $Y(i, j) = X^i M_j$ for all $1 \leq i \leq m$ and $1 \leq j \leq n$. First notice that the when we apply k-means to X and to Y we have

$$\mathscr{P}_k(X) = \langle C_1, C_2, \ldots, C_k \rangle = \mathscr{P}_k(Y)$$

(from Definition 2). Now let $1 \leq i \leq m$ and consider the kth component $(Y^i - \bar{Y})_k$ of the difference $(Y^i - \bar{Y})$. We see that

$$(Y^i - \bar{Y})_k = X^i M_k - \frac{1}{m} \sum_{r=1}^{m} Y(r,k) = X^i M_k - \frac{1}{m} \sum_{r=1}^{m} X^r M_k$$

$$= (X^i - \frac{1}{m} \sum_{r=1}^{m} X^r) M_k = (X^i - \bar{X}) M_k.$$

Thus

$$Y^i - \bar{Y} = \left[\begin{array}{cccc} (X^i - \bar{X})M_1 & (X^i - \bar{X})M_2 & \cdots & (X^i - \bar{X})M_n \end{array} \right] = (X^i - \bar{X})M,$$

and so the deviation is just

$$\|Y^i - \bar{Y}\| = \|(X^i - \bar{X})M\| = \|((X^i - \bar{X})M)^T\| = |M^T (X^i - \bar{X})^T\| = \|(X^i - \bar{X})^T\| = \|X^i - \bar{X}\|$$

where the second to last equality follows since M (and hence M^T) is an isometry. This shows that the denominators of $CI_k(X)$ and $CI_k(Y)$ from Definition 2 are the same; the cases for each term in the numerators are proved similarly. □

We will also need

Definition 4 (Median absolute deviation from the median (MAD)). *Let Y be a univariate (scalar-valued) data set. Then we define*

$$MAD(Y) := median(|Y - median(Y)|).$$

1 The SigClust Algorithm

Let $m, n \in \mathbb{Z}^+$ and $X \in \mathbb{R}^{m \times n}$.

1.1 Step 1: Compute $CI_2(X)$

See above definition, and use $k = 2$. This value will not be needed until the final step 7, so it can be computed at any point.

1.2 Step 2: Compute background noise level σ_N^2

In order to apply some dimensionality reduction later, we assume some level $\epsilon \sim N(0, \sigma_N^2 I_n)$ background noise on the data. We are told to estimate the standard deviation σ_N by

$$\sigma_N = \frac{MAD(X)}{MAD(N(0,1))}$$

Note that our definition for MAD is for a uni-variate data set and not for a vector data set such as $rows(X)$ whose elements come from $\mathbb{R}^n$. So in order for the above to make sense we need a multi-variate MAD.

The Mathematica implementation of MAD defines its value on an array to be the row vector of the original MAD values of each of the columns of X. This would make sense for us, but it would give us a scalar $\sigma_{N,i}$ for all $1 \leq i \leq n$, and our goal is to get a single scalar value σ_N.

There seem to be two possible approaches.

1. Taking the column-wise MAD of each of the features, and then take either the mean or the median of the resulting row vector.
2. Use some sort of multi-variate median (such as the *geometric median*), and replace the "absolute value" $|\cdot|$ by a vector norm on $\mathbb{R}^n$ such as the standard Euclidean $\mathscr{L}^2$ norm $\|\cdot\|_2$.

1.3 Step 3: PCA

We compute the eigenvalues (with multiplicities) for the covariance matrix for the mean-shifted data. Let's break this into steps.

1. Let $\mu \in \mathbb{R}^n$ be the column vector $\mu = \langle \mu_1, \ldots, \mu_n \rangle$ where for $1 \leq i \leq n$, μ_i is the mean of the ith feature, that is, the mean of column $X_i = X(\cdot, i)$. Also let $1_n = \langle 1, \ldots, 1 \rangle$ be the column vector of all 1s. The mean centered data is given by
$$Y = X - 1_n \mu^T.$$
2. Compute the covariance matrix for the data matrix Y by
$$\Sigma = \left(\frac{1}{n-1}\right) Y^T Y.$$
Using $n-1$ instead of n here is Bessel's correction. More on this in the future.
3. Compute the eigenvalues of Σ. Strictly speaking we only need the eigenvalues and not the eigenvectors for the rest of the algorithm. But note that Σ is a real symmetric matrix and thus has is diagonalizable by a orthogonal matrix M (a rigid rotation about the origin). In particular, Σ has an eigendecomposition
$$\Sigma = MDM^T$$
where
$$D = diag(\lambda_1, \lambda_2, \ldots, \lambda_n),$$
$$M = [v_1 v_2 \ldots v_n],$$
the λ_i are the eigenvalues of Σ in non-increasing order, and the v_i are corresponding eigenvectors of length $\|v_i\|_2 = 1$ which are pairwise orthogonal ($v_i \cdot v_j = 0$ for all $i \neq j$).
Note that we are only going to use eigenvalues greater than σ_N^2 from step 2, so using a method (such as Raleigh quotients) to compute eigenvalues from largest to smallest could save computation.

4. Upon finding the necessary eigenvalues λ_i, for each $1 \leq i \leq n$, set
$$\hat{\lambda}_i = max(\lambda_i, \sigma_N^2).$$
5. Define
$$\Sigma_0 = diag(\hat{\lambda}_1 \ldots, \hat{\lambda}_n)$$
6. We now have a mean-centered, axis-aligned multivariate Gaussian distribution
$$N(0, \Sigma_0)$$
from which to simulate data. We will refer to this distribution as the *null distribution.*

We now begin the "Monte Carlo" part of this algorithm.

1.4 Step 4: Simulation

Generate a data set $Z \in \mathbb{R}^{m \times n}$ from the null distribution. If this is the tth iteration of the Simulation step we can call this data set Z_t.

1.5 Step 5: Clustering

Compute $z = CL_2(Z)$ where Z is from the previous step. If this is the tth iteration of the Clustering step we can call this value $z_t = CI_2(Z_t)$.

1.6 Step 6: Iterate, then compute distribution on CL_2

Repeat the previous two steps N_L times for some large N_L. Let $S := \{z_t\}_{1 \leq t \leq N_L}$.

1.7 Step 7: Compute p-value of the test statistic

We can now estimate the CDF of $CL_2(Z)$ for $Z \in \mathbb{R}^{m \times n}$ with rows generated by $D_0 \sim N(0, \Sigma_0)$. Specifically, for any $r \in \mathbb{R}$ we estimate that
$$P_{Z \sim D_0}(CL_2(Z) \leq r) = \frac{1}{N_L} |\{t : 1 \leq t \leq N_L, z_t \leq r\}|.$$
In particular, letting $r = r_X := CL_2(X)$ from step 1, we can compute the p-value
$$P_{Z \sim D_0}(CL_2(Z) \leq r_X).$$
If this p-value is smaller than some threshold α, we reject the null hypothesis and split $rows(X)$ into two clusters according to the clusters discovered in step 1. We may then repeat the SigClust algorithm on either or both of the two clusters.