UCL

Clustering with the Gaussian mixture model

Christian Hennig

December 16, 2011

0. Overview

1.1 The Gaussian mixture model

Observations $\tilde{\mathbf{x}} = \mathbf{x}_1, \ldots, \mathbf{x}_n \in \boldsymbol{R}^p$ are assumed i.i.d. with density

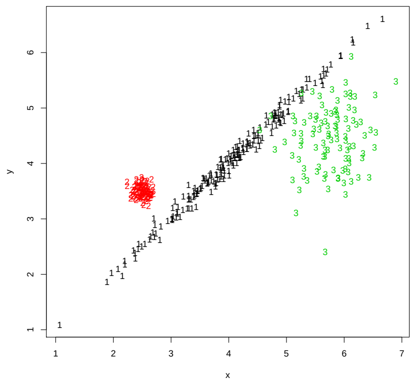
$$f(\mathbf{x}_i) = \sum_{j=1}^{k} \pi_j \varphi_{\mathbf{a}_j, \Sigma_j}(\mathbf{x}_i).$$

Parameters $\pi_j, \mathbf{a}_j, \Sigma_j$ will be estimated by maximum likelihood.
k will be estimated by the BIC (penalised ML).

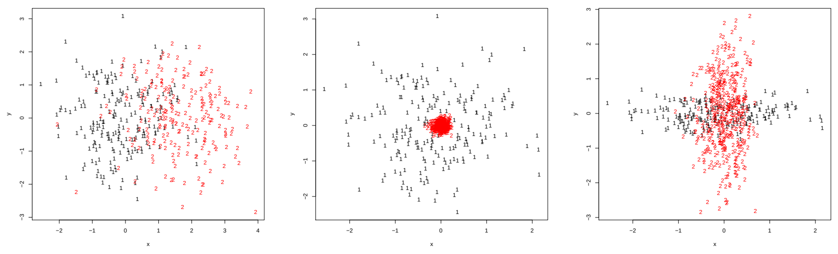
For clustering, normally identify each Gaussian subpopulation with a cluster.

What does this imply?

Gaussian populations are elliptical with flexible shapes.
Within-cluster distances may not be small.

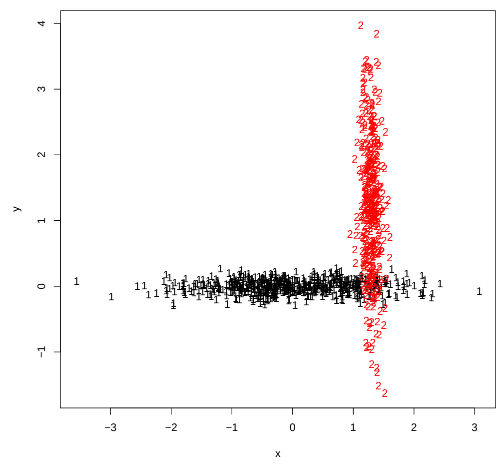


Gaussian mixtures may be unimodal and not heterogeneous.
Sometimes that's desired, sometimes not.

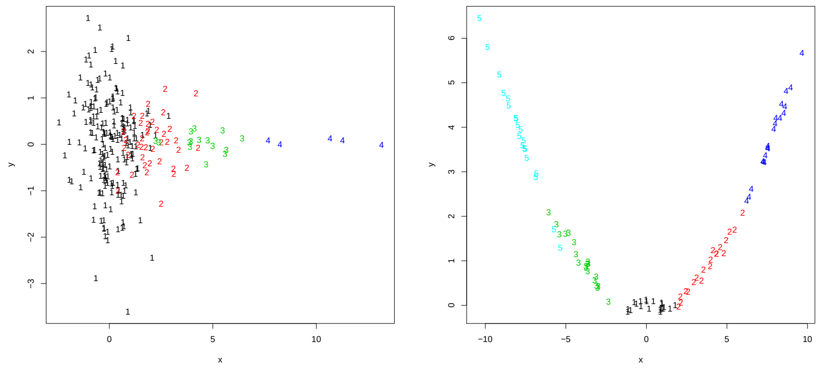


Density approximation vs.
"mode clustering" vs.
"pattern clustering".

Gaussian mixtures may have more modes than mixture components.



Gaussian mixtures can emulate all kinds of distributional shapes.



$$f(\mathbf{x}_i) = \sum_{j=1}^{k} \pi_j \varphi_{\mathbf{a}_j,\Sigma_j}(\mathbf{x}_i).$$

Starting from such a model does *not* mean that it is *required* that the data *really* come from a Gaussian mixture.

Gaussian mixtures are very flexible and "all models are wrong" anyway.

The model "assumption" rather defines the "cluster prototypes" we are looking for.
(Concentrated in centre, linear, maybe large variance.)

It tells us what "view on the data" is implied.
Whether that's suitable depends on the application.

1.2 The two-step version of the model

$(\gamma_1, \mathbf{x}_1), \ldots, (\gamma_n, \mathbf{x}_n)$ i.i.d.,

$$\begin{aligned} j = 1, \ldots, k: \; P\{\gamma_i = j\} &= \pi_j, \\ f(\mathbf{x}_i | \gamma_i = j) &= \varphi_{\mathbf{a}_j, \Sigma_j}(\mathbf{x}_i). \end{aligned}$$

This implies

$$p_{ij} = P\{\gamma_i = j | \mathbf{x}_i\} = \frac{\pi_j \varphi_{\mathbf{a}_j, \Sigma_j}(\mathbf{x}_i)}{\sum_{h=1}^k \pi_h \varphi_{\mathbf{a}_h, \Sigma_h}(\mathbf{x}_i)}.$$

After estimating all parameters, cluster points by

$$\hat{\gamma}_i = \arg\max_j \hat{p}_{ij} = \arg\max_j \frac{\hat{\pi}_j \varphi_{\hat{\mathbf{a}}_j, \hat{\Sigma}_j}(\mathbf{x}_i)}{\sum_{h=1}^k \hat{\pi}_h \varphi_{\hat{\mathbf{a}}_h, \hat{\Sigma}_h}(\mathbf{x}_i)}.$$

1.3 Gaussian mixtures and k-means clustering

k-means clustering is defined by

$$\sum_{i=1}^n \arg\min_{\gamma_i \in \{1,\ldots,k\}} \|\mathbf{x}_i - \mathbf{a}_{\gamma_i}\|^2 = \min!$$

This is maximum likelihood for

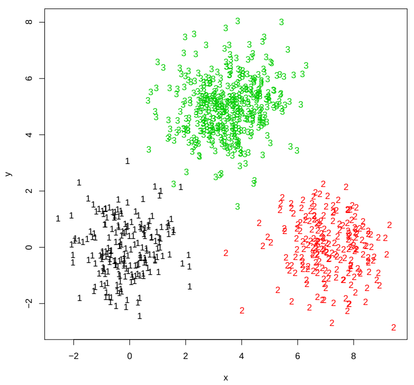
$$f(\tilde{\mathbf{x}}) = \prod_{i=1}^n \varphi_{\mathbf{a}_{\gamma_i}, \Sigma_{\gamma_i}}(\mathbf{x}_i),$$

where $\gamma_i \in \{1, \ldots, k\}$, $\Sigma_j = c\mathbf{I}_p \; \forall j$ ("Fixed Partition Model").

$$f(\mathbf{x}_i | \gamma_i = j) = \varphi_{\mathbf{a}_j, \Sigma_j}(\mathbf{x}_i)$$

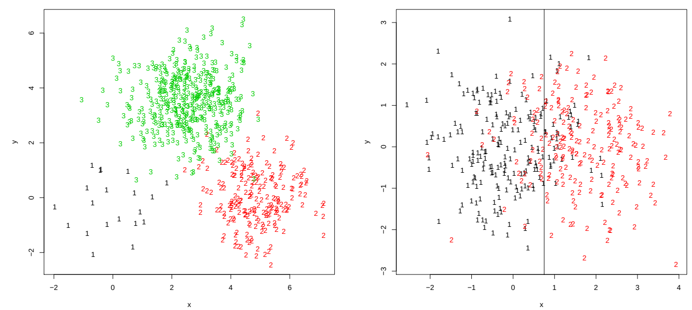
as in mixture, but without component probability π_j.
Can fit Gaussian mixture model with $\Sigma_j = c\mathbf{I}_p \; \forall j$, too.

Constraining covariance matrices,
Gaussian mixtures can emulate k-means cluster shapes
(less flexible, more homogeneous).



...and others.

Gaussian mixtures vs. k-means clustering
Gaussian mixtures allow more flexible cluster shapes.
k-means tends to produce clusters of similar sizes.
k-means is inconsistent because of crisp classification.



1.4 Constrained covariance matrices

$$f(\mathbf{x}_i) = \sum_{j=1}^{k} \pi_j \varphi_{\mathbf{a}_j,\Sigma_j}(\mathbf{x}_i).$$

k-means model: $\Sigma_j = c\mathbf{I}_p \ \forall j$.
Linear discriminant analysis: $\Sigma_j = \Sigma$.

Reasons for constraining the covariance matrices:

- Fewer parameters to estimate (low n, large p).
- Sometimes numerical problems with fully flexible Σ_j.
- Sometimes better interpretation.

But may not fit the data very well. (BIC can decide.)

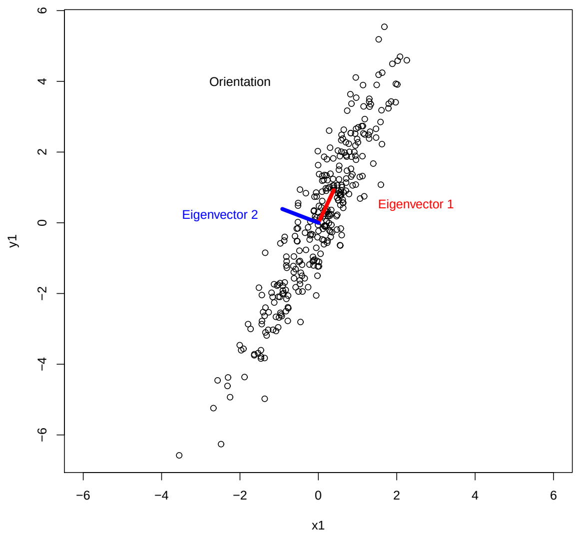
Banfield and Raftery (1993): use spectral decomposition

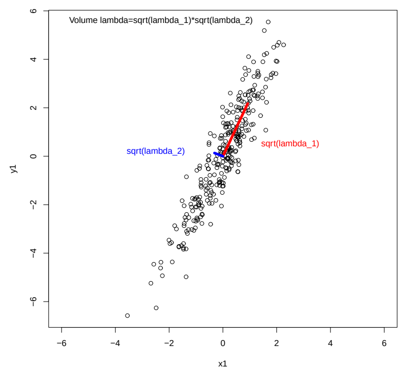
$$\Sigma_j = \lambda_j D_j A_j D_j^T,\ j = 1, \ldots, k,$$

where

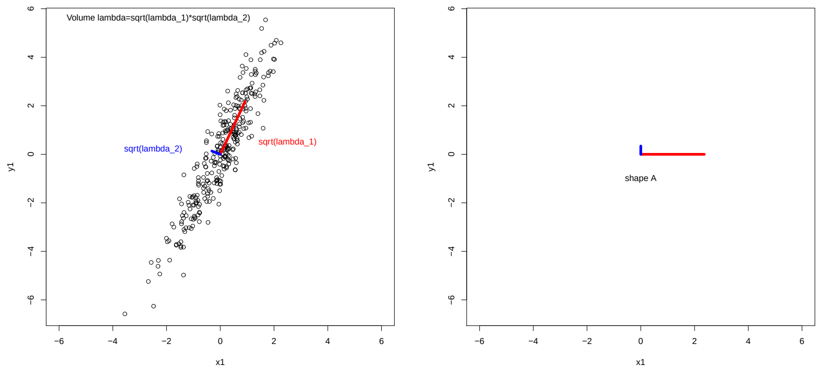
- $(\lambda_{j1}, \ldots, \lambda_{jp})$ eigenvalues,
- $\lambda_j = \prod_{i=1}^{p} (\lambda_{ji})^{1/p}$ hypervolume,
- D_j matrix of eigenvectors,
- $A_j = \frac{1}{\lambda_j} \mathrm{diag}(\lambda_{j1}, \ldots, \lambda_{jp})$ "shape" with $\det A_j = 1$.

Christian Hennig Clustering with the Gaussian mixture model





ChristianHennig ClusteringwiththeGaussianmixturemodel



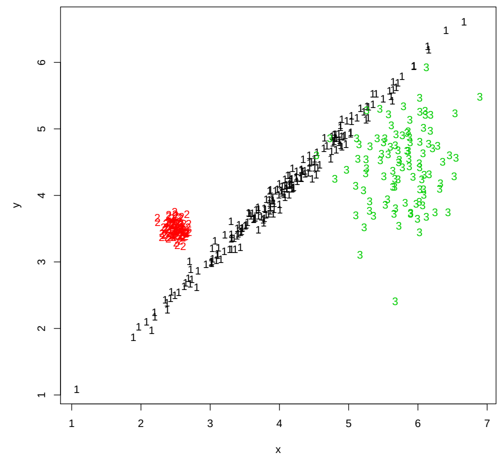
One or more of these can be assumed equal between clusters.
Shape can be assumed to be the unit matrix.

mclust coding

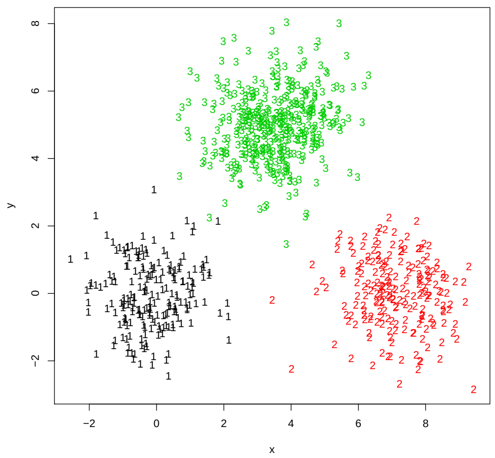
“V” variable, “E” equal, “I” unit matrix.
Models are defined by three letter codes for volume, shape, orientation.


```
From ?mclustModelNames:

univariateMixture: A vector with the following components:
  "E": equal variance (one-dimensional)
  "V": variable variance (one-dimensional)

multivariateMixture: A vector with the following components:
 "EII": spherical, equal volume
 "VII": spherical, unequal volume
 "EEI": diagonal, equal volume and shape
 "VEI": diagonal, varying volume, equal shape
 "EVI": diagonal, equal volume, varying shape
 "VVI": diagonal, varying volume and shape
 "EEE": ellipsoidal, equal volume, shape, and orientation
 "EEV": ellipsoidal, equal volume and equal shape
 "VEV": ellipsoidal, equal shape
 "VVV": ellipsoidal, varying volume, shape, and orientation
```

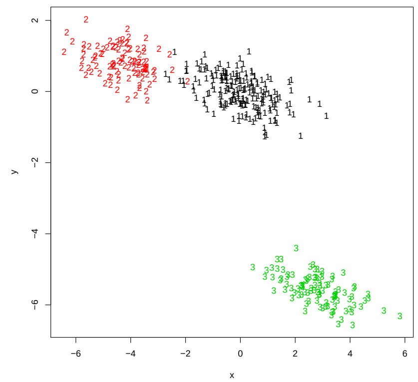

“VVV”: fully flexible model.



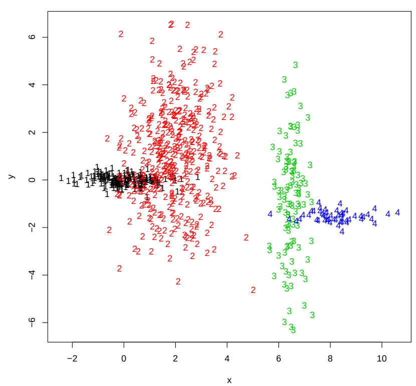
“EII”: equal volume, spherical (*k*-means)



“EEE”: equal (but flexible) volume, shape and orientation.
Assumptions of linear discriminant analysis.



“VVI”: diagonal (“local independence”);
components can be interpreted in terms of marginals



Constraints used for estimation:

Equal volume: clusters are similar
in terms of within-cluster dissimilarity/variation.

Non-unit shape: clustering invariant against variable scaling.

Non-diagonal orientation: clustering rotation invariant.

Optimising over all models: not rotation and scale invariant.

Note again: models are not required to be true,
but determine implications for clustering.

1.5 Identifiability

Can the same dataset be fitted equally well
by two different mixtures of Gaussians?

If so, the found “clusters” cannot be interpreted.

Theoretically: can the same underlying distribution
be written down as a mixture in two different ways?

(If not, there may still be trouble for certain datasets,
which cannot generally be excluded.)

Theorem (Yakowitz and Spragins 1968): Assume $f = g$ with

$$f(\mathbf{x}) = \sum_{j=1}^{k} \pi_j \varphi_{\mathbf{a}_j,\Sigma_j}(\mathbf{x}),\ g(\mathbf{x}) = \sum_{j=1}^{l} \epsilon_j \varphi_{\mathbf{b}_j,\Gamma_j}(\mathbf{x}),$$

$$\sum_{j=1}^{n} \pi_j = \sum_{j=1}^{n} \epsilon_j = 1,\ \forall j:\ \pi_j > 0, \epsilon_j > 0,$$
$$\forall j \neq h:\ (\mathbf{a}_j, \Sigma_j) \neq (\mathbf{a}_h, \Sigma_h),\ (\mathbf{b}_j, \Gamma_j) \neq (\mathbf{b}_h, \Gamma_h).$$

Then $k = l$ and there is a permutation τ so that

$$\forall j = 1, \ldots, k:\ (\pi_j, \mathbf{a}_j, \Sigma_j) = (\epsilon_{\tau(j)}, \mathbf{b}_{\tau(j)}, \Gamma_{\tau(j)}).$$

2. Computation of the ML-estimator: The EM-algorithm

Assume k fixed. Try to maximise

$$\log L_{n,k}(\tilde{\mathbf{x}}) = \sum_{i=1}^{n} \log \left(\sum_{j=1}^{k} \pi_j \varphi_{\mathbf{a}_j,\Sigma_j}(\mathbf{x}_i) \right)$$

under $\pi_j > 0 \forall j,\ \sum_{j=1}^{k} \pi_j = 1$.
Unfortunately there is no straightforward analytic solution.
Need algorithm to find local optima.

Several ones exist, most popular is the EM-algorithm.
Initialisation treated afterwards.

2.1 The general EM-algorithm

EM-algorithm (Dempster, Laird and Rubin 1977):
general principle to find ML-estimator if information is incomplete.

Sometimes “EM-algorithm”
is referred to as “clustering method”,
but EM-algorithm can be used
for many different problems and models.

Missing information in the mixture model:
cluster memberships $\gamma_1, \ldots, \gamma_n$.

General principle:
$\tilde{\mathbf{y}} = \mathbf{y}_1, \dots, \mathbf{y}_n$ unobserved complete data.
$\tilde{\mathbf{x}} = T(\tilde{\mathbf{y}})$ observed data (mixture: $\mathbf{y}_i = (\gamma_i, \mathbf{x}_i)$).

Attempt to maximise $l_{n,k}(\eta) = \sum_{i=1}^n \log f_\eta(\mathbf{x}_i)$.
Define $l_{n,c}(\eta) = \sum_{i=1}^n \log f_{\eta,c}(\mathbf{y}_i)$. η_0 initialisation.

E-step Compute **E**xpected complete likelihood.

$$q(\eta|\eta_{t-1}) = E_{\eta_{t-1}}(l_{n,c}(\eta)|T = \tilde{\mathbf{x}}).$$

M-step **M**aximise conditional likelihood.

$$\eta_t = \arg\max_\eta q(\eta|\eta_{t-1}).$$

Theorem (DLR 1977): Both steps never decrease $l_{n,k}(\eta)$.

2.2 EM in the Gaussian mixture model

$\eta = (\pi_1, ldots, \pi_k, \mathbf{a}_1, \dots \mathbf{a}_k, \Sigma_1, \dots, \Sigma_k)$.
Complete loglikelihood with γ_i known:

$$l_{n,k,c}(\eta) = \sum_{i=1}^n \sum_{j=1}^k 1(\gamma_i = j)(\log \pi_j + \log \varphi_{\mathbf{a}_j,\Sigma_j}(\mathbf{x}_i)),$$

E-step:

$$E_{\eta_{t-1}}(l_{n,k,c}(\eta)|T = \tilde{\mathbf{x}}) =$$
$$= \sum_{i=1}^n \sum_{j=1}^k P(\gamma_i = j|\eta_{t-1}, \mathbf{x}_i)(\log \pi_j + \log \varphi_{\mathbf{a}_j,\Sigma_j}(\mathbf{x}_i)),$$
$$p_{ij}^{(t-1)} = P\{\gamma_i = j|\eta_{t-1}, \mathbf{x}_i\} = \frac{\pi_j^{(t-1)} \varphi_{\mathbf{a}_j^{(t-1)},\Sigma_j^{(t-1)}}(\mathbf{x}_i)}{\sum_{h=1}^k \pi_h^{(t-1)} \varphi_{\mathbf{a}_h^{(t-1)},\Sigma_h^{(t-1)}}(\mathbf{x}_i)}.$$

M-step: maximise

$$\sum_{i=1}^n \sum_{j=1}^k p_{ij}^{(t-1)}(\log \pi_j + \log \varphi_{\mathbf{a}_j,\Sigma_j}(\mathbf{x}_i)).$$

Model VVV: can separately maximise

$$\sum_{i=1}^n \sum_{j=1}^k p_{ij}^{(t-1)} \log \pi_j \Rightarrow \pi_j^t = \frac{1}{n} \sum_{i=1}^n p_{ij}^{(t-1)},$$

$$\sum_{i=1}^n \sum_{j=1}^k p_{ij}^{(t-1)} \log \varphi_{\mathbf{a}_j,\Sigma_j}(\mathbf{x}_i),$$

which yields weighted Gaussian ML estimators for $(\mathbf{a}_j, \Sigma_j)$:

$$\mathbf{a}_j^t = \frac{1}{n} \sum_{i=1}^n p_{ij}^{(t-1)} \mathbf{x}_i, \; \Sigma_j^t = \frac{1}{n} \sum_{i=1}^n p_{ij}^{(t-1)} (\mathbf{x}_i - \mathbf{a}_j^t)(\mathbf{x}_i - \mathbf{a}_j^t)^T.$$

Can iterate these until "convergence",
normally defined by "increase in $l_{n,k}$ smaller than c"
though doesn't guarantee convergence of all parameters.

Note that this gives you (at best) a local optimum.

2.3 Initialisation

EM-algorithm depends on initialisation.
Better initialisation $\Rightarrow$ better local optimum.

EM-algorithm can be started from initial parameters or an initial set of p_{ij}^0.
It can therefore be initialised by a partition of the data, in which case p_{ij}^0 is either 0 or 1.

- Start EM q times from random partitions and choose solution that maximises $l_{n,k}$.
- Try to find an "intelligent" starting partition.
- Various alternatives in literature.

Initialisation by hierarchical clustering
(default for mclust package, function hc)

1. Start with every data point as cluster.
2. Merge the two "closest" clusters.
3. Go to 2 until there are k clusters
   (or a single one, to compute a whole hierarchy).

In Step 2, merge clusters that lead to maximum $l_{n,k}$.

Can be computed from pairwise dissimilarity matrix, which requires much memory and time for large n.
For large n do this on subset and extract parameters.

Implemented for VVV, EEE, EII, VII.
(Where not implemented, VVV is default.)

3 Estimating model complexity by the BIC

Estimating k is a model complexity problem.
Models are nested (k mixture components are special case of $k+1$ with $\pi_j = 0$ for some j).
if k increases, $l_{n,k}^* = l_{n,k}(\eta_{k,ML}) = \max_\eta l_{n,k}(\eta)$ increases, too.

Penalised likelihood is a popular approach to estimate model complexity. With $p(k)$ increasing:

$$l_{n,k}^* - p_n(k) = max!$$

Various choices of $p_n(k)$ are in the literature (AIC, BIC, CAIC).

BIC: With $d(k)$ number of free parameters:

$$2l_{n,k}^* - d(k)\log(n) = max!$$

Note that in the literature often $BIC = -2l_{n,k}^* + d(k)\log(n)$.

Motivation 1:
Originally (Schwarz 1978), the BIC has been derived in a Bayesian setup as approximation for

$$p(\tilde{\mathbf{x}}|k) = \int l_{n,k}(\eta, \tilde{\mathbf{x}})h(\eta)d\eta,$$

where h is uniform prior for η.
$p(\tilde{\mathbf{x}}|k)$ is proportional to the posterior for k if all k have the same prior probability.

Motivation 2: Keribin (2000):
BIC estimates k consistently in mixture model under some assumptions, which are fulfilled for a 1-d Gaussian mixture with equal variances bounded from below.

Still seems to be best existing consistency result.

Problem with consistency:
If Gaussian mixture model does not hold precisely, for large n estimated k will become larger and larger in order to give optimal Gaussian mixture approximation.

BIC model selection:

Fit models with all k of interest.
Choose the one with largest BIC.

Can use BIC as well in order to select **covariance constraints**, governed by number of parameters.

4 Model-based clustering with the mclust package

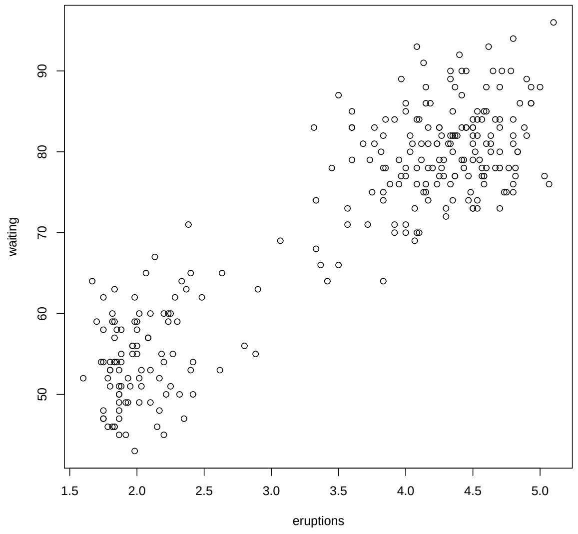
mclust (Fraley and Raftery 2002, 2010) is an add-on package for R (R development core team, 2011) for (Gaussian mixture) model-based clustering.

mclust-documentation: Fraley and Raftery, (2010)
`http://www.stat.washington.edu/fraley/mclust/tr504.pdf`

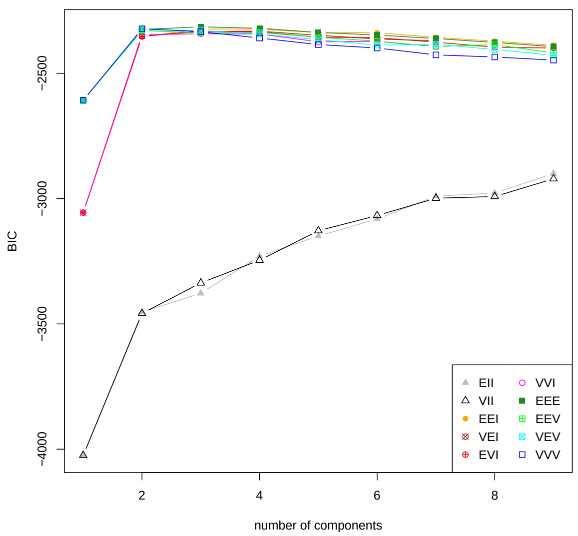
mclust has a nonstandard licence:
`http://www.stat.washington.edu/mclust/license.txt`

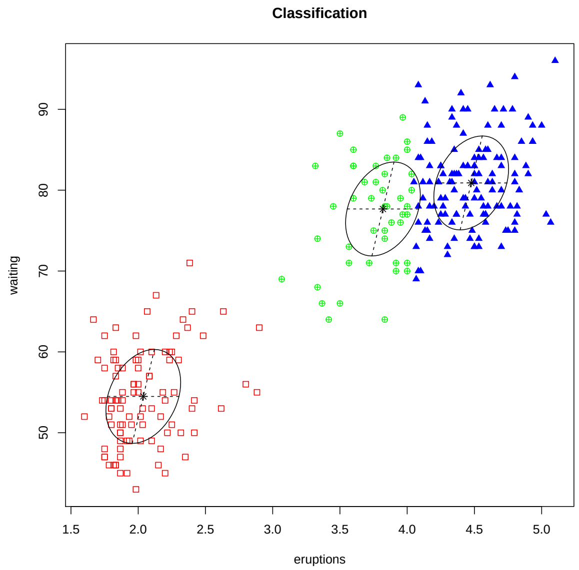
Example: old faithful dataset

```
> library(mclust)
# Loads mclust package

> data(faithful)
# Supplied with R base

> plot(faithful)
# Standard scatterplot of data
```

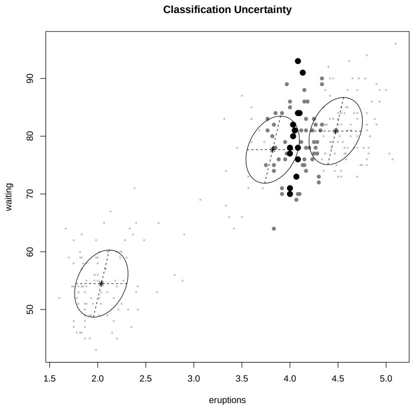



```
faithfulm <- Mclust(faithful)
# Run Mclust on old faithful data

plot(faithfulm,faithful)
# Four mclust default plots

names(faithfulm)
 [1] "modelName"      "n"              "d"              "G"
 [5] "BIC"            "bic"            "loglik"         "parameters"
 [9] "z"              "classification" "uncertainty"
```



Uncertainty of $\hat{\gamma}_i$: $1 - \hat{p}_{i\hat{\gamma}_i}$.
Uncertainty graph shows upward 0.75- and 0.9-quantile.



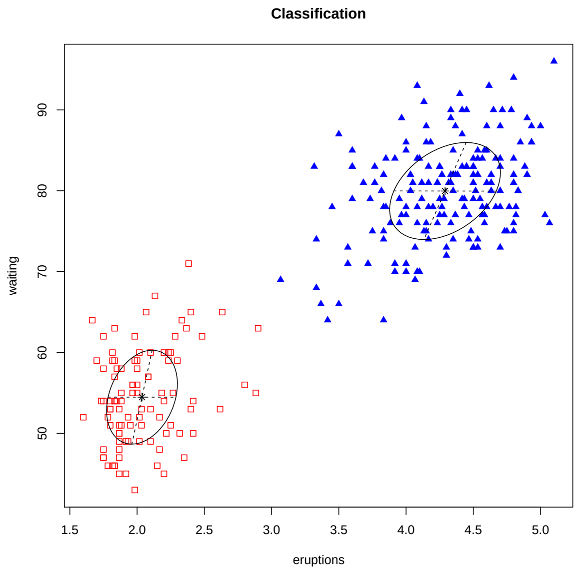

```
> faithfulm

 best model: elliposidal, equal variance with 3 components

> faithfulm$classification
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20
  3   2   3   2   1   2   1   3   2   1   2   3   1   2   1   2   2   1   2   1
 21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40
  2   2   3   3   1   3   2   1   3   1   1   1   3   3   3   2   2   1   2   1
(...)
261 262 263 264 265 266 267 268 269 270 271 272
  1   1   2   1   2   2   1   1   2   1   2   1
> faithfulm$loglik
[1] -1126.361
```



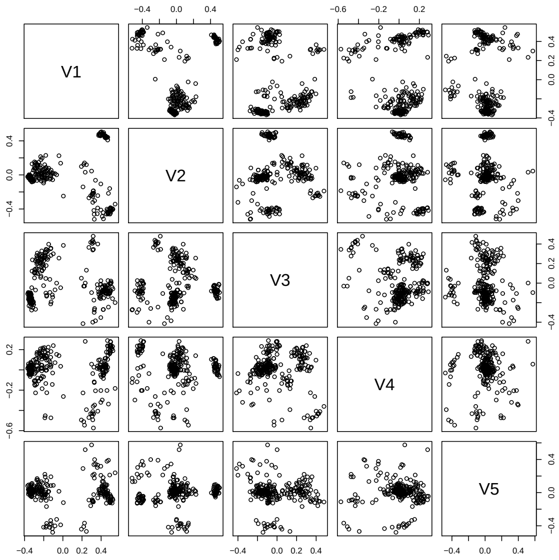
```
# The following emulates the results before; generally mclustBIC allows
# some more.
> faithfulmb <- mclustBIC(faithful)

> plot(faithfulmb)

> faithfulsum <-  summary(faithfulmb,faithful)

> names(faithfulsum)
 [1] "modelName"      "n"              "d"              "G"
 [5] "bic"            "loglik"         "parameters"     "z"
 [9] "classification" "uncertainty"

> mclust2Dplot(data=faithful,parameters=faithfulsum$parameters,
             z=faithfulsum$z,classification=faithfulsum$classification,
             uncertainty=faithfulsum$uncertainty,what = "classification")

> mclust2Dplot(data=faithful,parameters=faithfulsum$parameters,
             z=faithfulsum$z,classification=faithfulsum$classification,
             uncertainty=faithfulsum$uncertainty,what = "uncertainty")

> faithfulsum

classification table:
  1   2   3
130  97  45

best BIC values:
    EEE,3     EEE,4     VVV,2
-2314.386 -2320.207 -2322.192
```



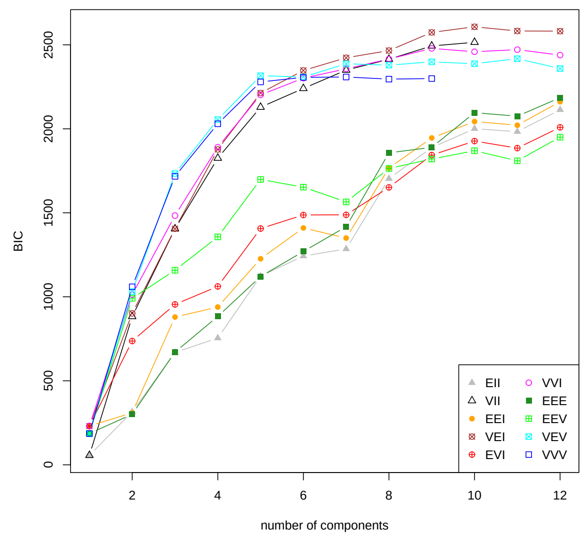
```
> faithfulvvv <- Mclust(faithful,modelNames="VVV")
# Force model to be "VVV"

> plot(faithfulvvv,faithful)
```

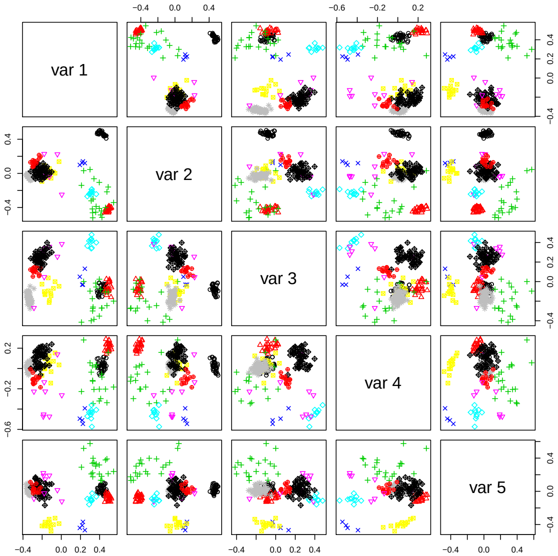


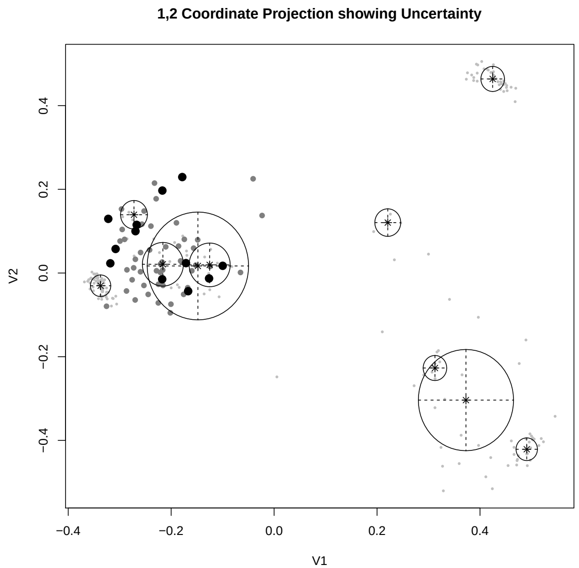
A 5-dimensional dataset



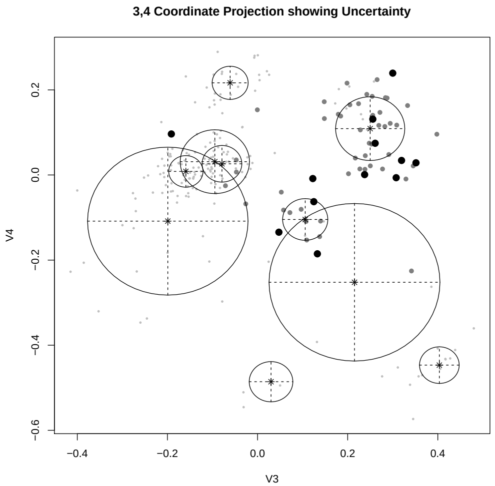

```
> trigonadata <- read.table("trigona.dat")

> trigonam <- Mclust(trigonadata)
Warning messages:
1: In summary.mclustBIC(Bic, data, G = G, modelNames = modelNames) :
  best model occurs at the min or max # of components considered
2: In Mclust(trigonadata) :
  optimal number of clusters occurs at max choice

# G: number of components. Default G is 1:9.
> trigonam <- Mclust(trigonadata,G=1:12)

> plot(trigonam,trigonadata)
```






```
plot(trigonam,trigonadata,what="uncertainty",dimens=c(3,4))
```



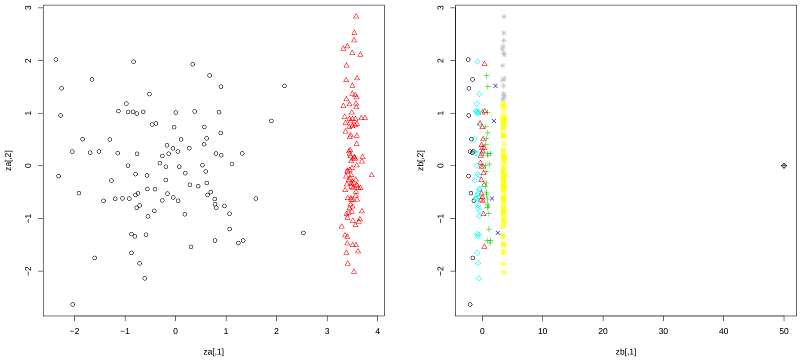
5. Potential problems with mixture model-based clustering

Using mclust (Gaussian mixtures) for aim of clustering.

General attitude: models are not true,
model assumptions are always violated,
what does a method do when faced with different situations,
is this desirable, and if not, how to deal with it?

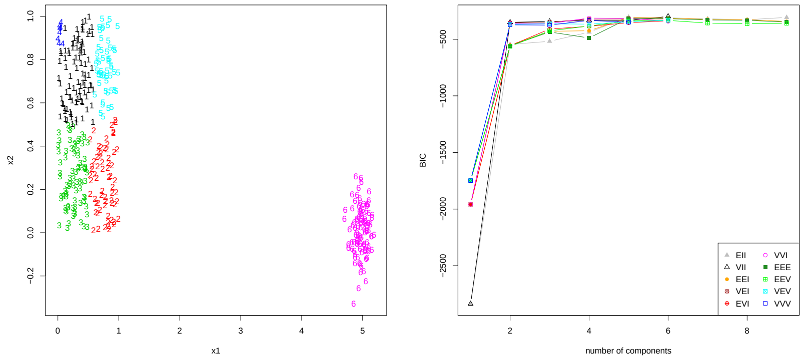
All CA methods are problematic.

5.1 Outliers

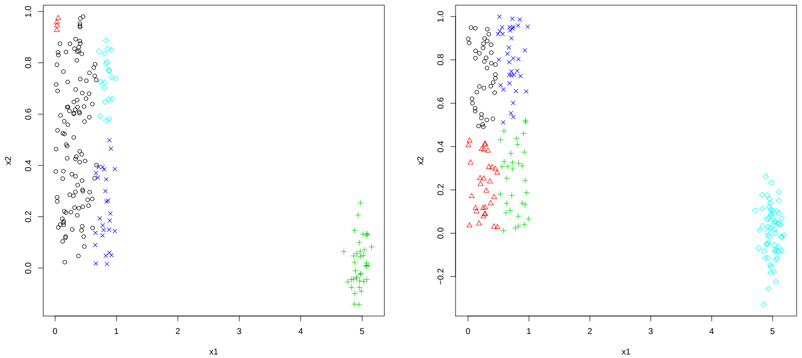


Gaussian mixture ML is sensitive toward outliers.

5.2 Non-normality



5.3 Instability



Sometimes only parts of solution are stable.
Non-normality is one but not only source for instability.

More reasons for instability:

- Gaussian components may not be properly separated,
- Very small “spurious clusters”
- Dataset too small

Instabilities may be tolerated if for example density estimation is of interest and not classification.

6 Degenerating likelihood

Consider k fixed, $(\mathbf{a}_{1m}, \Sigma_{1m})_{m \in N}$ so that $\lambda_{min}(\Sigma_{1m}) \to \infty$, $\exists \mathbf{x}_i = \mathbf{a}_{1m}$, and $\forall \mathbf{x}_i, m \exists j: \ \varphi_{\mathbf{a}_{jm}, \Sigma_{jm}}(\mathbf{x}_i) > c > 0$.

$$\Rightarrow l_n = \sum_{i=1}^{n} \log \left(\sum_{j=1}^{s} \pi_{jm} \varphi_{\mathbf{a}_{jm}, \Sigma_{jm}}(\mathbf{x}_i) \right) \to \infty.$$

The likelihood therefore is unbounded
and “Maximum Likelihood” rather means
“a local non-degenerated likelihood optimum”.

Argument requires variable volumes
(models starting with “V”).
Does not hold where cov-EVs $\to 0$ for *all* j.

Implications of degenerating likelihood

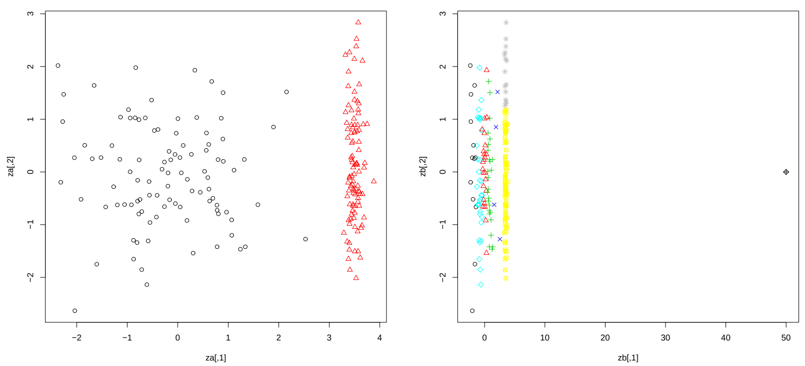
- Consistency proofs for fixed k
  are for local optima and don’t deliver uniqueness
  (which makes asymptotic normality problematic).
- In practice, the EM-algorithm may degenerate.
- The EM-algorithm may find a “spurious” local optimum
  with very small covariance eigenvalue.
  (Few points lying almost precisely on a low-d hyperplane.)

Theoretically, $\lambda_{min}(\Sigma) \geq c$ or $\frac{\lambda_{min}(\Sigma_j)}{\lambda_{max}(\Sigma_k)} \geq c$ prevent degeneration.
But not implemented in mclust (and choice of c tricky).

Default mclust discards solutions with non-invertible Σ.
Will choose other k or covariance matrix model by BIC.

Radical solution: Use models starting with “E” only.

Outliers in data may change the covariance matrix model.



Bayesian maximum posterior

mclust-option for handling degenerating likelihoods:
introduce prior distributions for $\mathbf{a}_j, \Sigma_j$,
compute maximum posterior (MAP) estimator instead of ML.

$$\mu|\Sigma \sim \mathcal{N}(\mu_p, \Sigma/\kappa_p),\ \Sigma \sim \text{inverseWishart}(\nu_p, \Delta_p)$$

MAP maximises

$$l_{n,k}(\eta) + \log p(\eta),$$

and is therefore penalised ML;
should penalise too small EVs of cov-matrices.

Fraley and Raftery (2007):
μ_p, Δ_p overall mean, cov-matrix$/k^{2/p}$,
$\nu_p = p + 2, \kappa_p = 0.01$.

Note that MAP estimators are biased. M-step change for VVV:

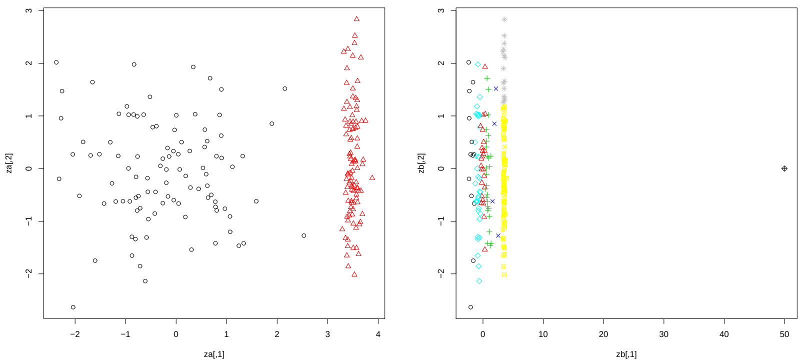
$$\mathbf{a}_{k,MAP-M} = \frac{n_k \mathbf{a}_{k,ML-M} + \kappa_p \mu_p}{n_k + \kappa_p},$$

$$\Sigma_{k,MAP-M} = \frac{\Delta_p + \frac{\kappa_p n_k}{\kappa_p + n_k}(\mathbf{a}_{k,ML-M} - \mu_p)(\mathbf{a}_{k,ML-M} - \mu_p)^T + n_k \Sigma_{k,ML-M}}{\nu_p + n_k + p + 2},$$

push cov-EVs closer to Δ_p's and deviation of $\mathbf{a}_k$ from μ_k,
means closer to μ_p.

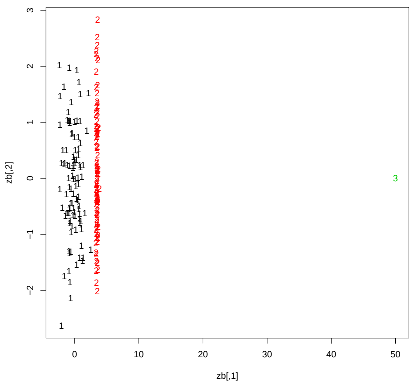
Not proper Bayes, no posterior distribution, no prior for π_j.
Compute MAP estimator and BIC based on MAP likelihood.
Improves problems with spurious clusters
and degenerating likelihood.


```
> set.seed(11111)
> z1 <- rnorm(100,0,1)
> z2 <- rnorm(100,3.5,0.1)
> z3 <- rnorm(100,0,1)
> z4 <- rnorm(100,0,1)
> za <- cbind(c(z1,z2),c(z3,z4))
> zb <- rbind(za,c(50,0))
> plot(zb)

> mza <- mclustBIC(za)
> smza <- summary(mza,za)
> plot(za,col=smza$classification)

> mzb <- mclustBIC(zb)
> smzb <- summary(mzb,zb)
```




```
mzbp <- mclustBIC(zb,prior=priorControl())
smzbp <- summary(mzbp,zb)
plot(zb,col=smzbp$classification)
```

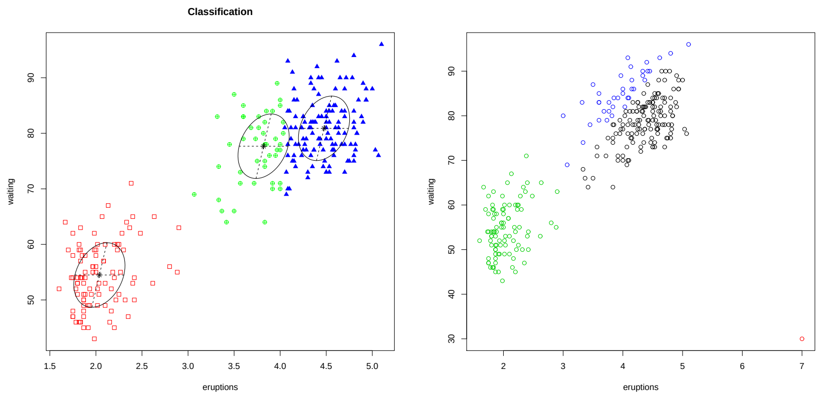
Prior parameters can be set in `priorControl`, e.g. `priorControl(shrinkage=0.1,scale=diag(2))` to set κ_p, Δ_p, see `?priorControl, ?defaultPrior`.



7 The noise component to deal with outliers

Unfortunately priors can’t solve all outlier problems.

```
> faithfulx <- rbind(faithful,c(7,30),c(3,80))
> mfaithfulx <- mclustBIC(faithfulx,prior=priorControl())
> smfaithfulx <- summary(mfaithfulx,faithfulx)
> plot(faithfulx,col=smfaithfulx$classification)
```



The “noise component” (Banfield and Raftery, 1993)

$$f(\mathbf{x}) = \pi_0 \frac{1}{V} + \sum_{j=1}^{s} \pi_j \varphi_{\mathbf{a}_j, \Sigma_j}(\mathbf{x}),$$

V is fixed during EM-algorithm (mclustBIC)
as volume of smallest hyperrectangle covering data,
but initial π_0 is needed and outliers should not affect
initialisation of Gaussian components.

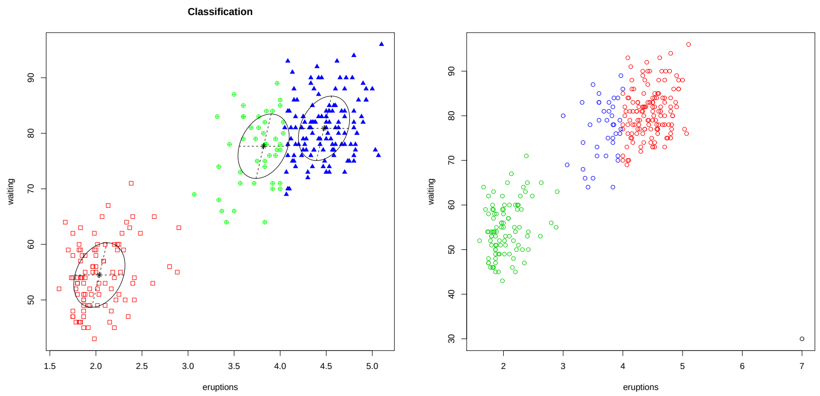
In mclustBIC: `initialization=list(noise=initnoise)`.

May draw initial noise points at random.

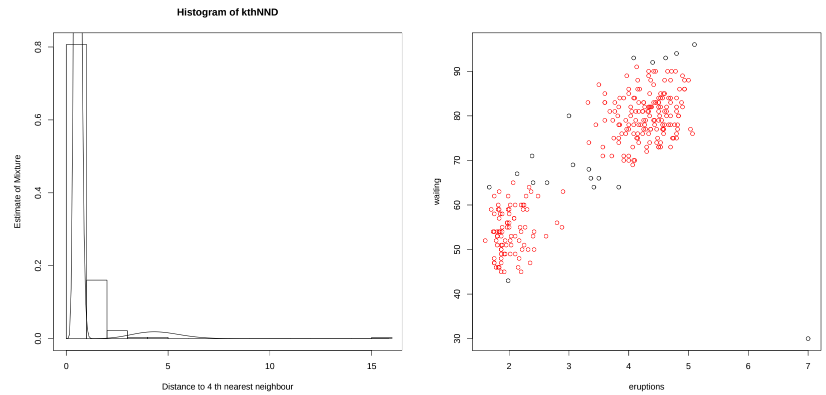
Better (reproducible):
`NNclean` (Byers and Raftery 1998) in prabclus.
Fits mixture of transformed Gamma-distributions
on distances to K-nearest neighbor
based on mixture of
two homogeneous (uniform) Poisson processes for data.
Component with larger mean is “noise”.

Specification of K required.
Isolated groups of fewer than K points may still be regarded as noise.
Decide based on application and size of dataset.


```
> library(prabclus)

> initnoise <- as.logical(1-NNclean(faithfulx,k=4)$z)
> mfaithfulxn <- mclustBIC(faithfulx,
                   initialization=list(noise=initnoise))
> smfaithfulxn <- summary(mfaithfulxn,faithfulx)
> plot(faithfulx,col=smfaithfulxn$classification+1)
```

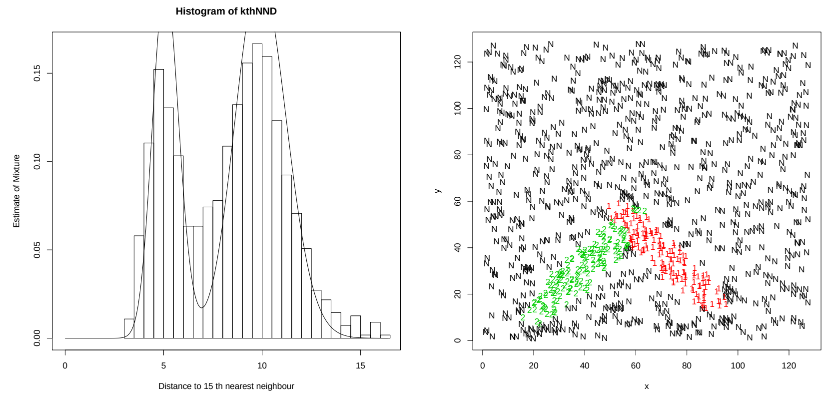



```
> faithfulnn <- NNclean(faithfulx,k=4,plot=TRUE)
> plot(faithfulx,col=1+faithfulnn$z)
```



An example with lots of noise:

```
> data(chevron)
> nnc <-  as.logical(1-NNclean(chevron[,2:3],15,plot=TRUE)$z)
> mc <- mclustBIC(chevron[,2:3],initialization=list(noise=nnc))
> smc <- summary(mc,chevron[,2:3])
> plot(chevron[,2:3],col=1+smc$classification)
```

The noise component can break down with extreme outliers.
Much recent work on robust clustering, for example
Coretto and Hennig (2010)
on finding an optimal value for the "noise density",
trimmed clustering, mixtures of t-distributions,
forward search etc.

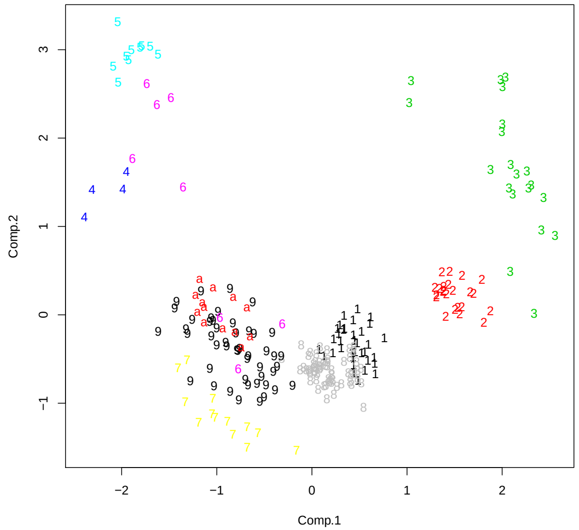
8. Cluster validation

Check whether outcome of clustering method makes sense.
Strategies:

- External/subject matter information
- Significance tests for structure
- Compare different clusterings on same dataset
- Validation indexes
- Visual inspection
- Stability assessment

Some indexes, validation information by
`cluster.stats` in fpc based on distance matrix.

$s(i) = \frac{b(i)-a(i)}{\max(a(i),b(i))}$ is called the "silhouette width" (Kaufman and Rousseeuw, 1990),
$a(i)$ is average distance of $\mathbf{x}_i$ to another point of its own cluster,
$b(i)$ is average distance to another point of closest cluster.
This can be averaged clusterwise over points.



```
> cs <- cluster.stats(dist(trigonadata),trigonam$classification)
> cs
$n
[1] 236

$cluster.number
[1] 10

$cluster.size
 [1] 35 23 20  4 10  8 13 62 48 13

$diameter
 [1] 0.2220615 0.2011110 0.8882174 0.2466013 0.2520631
0.7895725 0.3429880
 [8] 0.2464109 0.3996499 0.2268971

$average.distance
 [1] 0.10960597 0.10530936 0.42058017 0.14797559
0.11524152 0.49448545
 [7] 0.18921780 0.09693295 0.18436365 0.11742765
```



```
$median.distance
 [1] 0.10757334 0.10478257 0.40924474 0.13831797
0.11075841 0.52140322
 [7] 0.19024028 0.09521223 0.18188913 0.11888133

$separation
 [1] 0.5889131 0.3425002 0.3425002 0.5002507 0.3354944
0.0897763 0.3193068
 [8] 0.1922279 0.1604022 0.0897763

$average.toother
 [1] 0.8898844 0.9043505 0.8773002 0.8711378 0.9062254
0.7031898 0.6800758
 [8] 0.7083479 0.6734774 0.5841906
```



```
$separation.matrix
          [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]
 [1,] 0.0000000 0.8214897 0.6121101 0.7355199 0.9163432 0.5889131 0.6446088
 [2,] 0.8214897 0.0000000 0.3425002 0.9149291 0.7642136 0.8000080 0.7271676
 [3,] 0.6121101 0.3425002 0.0000000 0.8453088 0.5350112 0.6501032 0.7943997
 [4,] 0.7355199 0.9149291 0.8453088 0.0000000 0.5467675 0.6286042 0.5002507
 [5,] 0.9163432 0.7642136 0.5350112 0.5467675 0.0000000 0.3354944 0.8125327
 [6,] 0.5889131 0.8000080 0.6501032 0.6286042 0.3354944 0.0000000 0.4271635
 [7,] 0.6446088 0.7271676 0.7943997 0.5002507 0.8125327 0.4271635 0.0000000
 [8,] 0.8023756 0.8801732 0.6508053 0.8341647 0.8896053 0.2331168 0.3685067
 [9,] 0.7274789 0.7901756 0.6227011 0.7106608 0.6295933 0.1891964 0.3193068
[10,] 0.6927242 0.9830951 0.7581647 0.7185608 0.7778999 0.0897763 0.4601516
          [,8]      [,9]     [,10]
 [1,] 0.8023756 0.7274789 0.6927242
 [2,] 0.8801732 0.7901756 0.9830951
 [3,] 0.6508053 0.6227011 0.7581647
 [4,] 0.8341647 0.7106608 0.7185608
 [5,] 0.8896053 0.6295933 0.7778999
 [6,] 0.2331168 0.1891964 0.0897763
 [7,] 0.3685067 0.3193068 0.4601516
 [8,] 0.0000000 0.2245057 0.1922279
 [9,] 0.2245057 0.0000000 0.1604022
[10,] 0.1922279 0.1604022 0.0000000
```



```
$average.between
[1] 0.7680693

$average.within
[1] 0.1413954

(...)

$clus.avg.silwidths
         1          2          3          4          5
 0.8616234  0.8269252  0.2727838  0.7614558  0.8142473
         6          7
-0.1954790  0.6117464
         8          9         10
 0.7245092  0.4113044  0.6319789

$avg.silwidth
[1] 0.6147748

(...)
```

Cluster validation is not about estimating the number of clusters!
The results of such a method still need to be validated.

8.1 Cluster validation by visualisation

Generally use different colours and symbols.
Here: **projection methods**.

Given: $n \times p$-dataset $\mathbf{X}$.
Find $p \times s$-matrix $\mathbf{C}$ (eg, $s = 2$), so that
$\mathbf{Y} = \mathbf{XC}$ is optimally "informative".

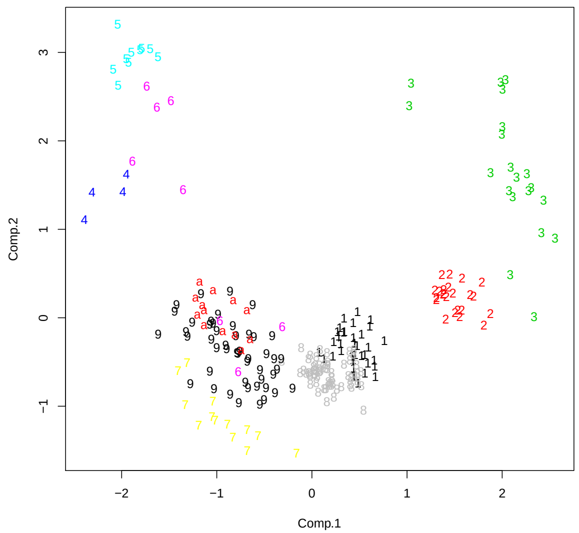
Definition. The first s projection vectors defined by the choice of $\mathbf{Q}$ and $\mathbf{R}$) $\mathbf{c}_1, \ldots \mathbf{c}_s$ are defined as the vectors maximising

$$F_{\mathbf{c}} = \frac{\mathbf{c}'\mathbf{Qc}}{\mathbf{c}'\mathbf{Rc}}$$

subject to $\mathbf{c}_i'\mathbf{Rc}_j = \delta_{ij}$, where $\delta_{ij} = 1$ for $i = j$ and $\delta_{ij} = 0$ else.

Corollary. The first s projection vectors of $\mathbf{X}$ are the eigenvectors of $\mathbf{R}^{-1}\mathbf{Q}$ corresponding to the s largest eigenvalues.

Definition. PCA is defined by $\mathbf{Q} = Cov(\mathbf{X})$ and $\mathbf{R} = \mathbf{I}_p$.



PCA: "Information" = variance. Clusters ignored.

Notation:
Let $\mathbf{x}_{i1}, \ldots, \mathbf{x}_{in_i}$ the p-dimensional points of group $i = 1, \ldots, k$, $n = \sum_{i=1}^{k} n_i$. Let $\mathbf{X}_i = (\mathbf{x}_{i1}, \ldots, \mathbf{x}_{in_i})'$, $i = 1, \ldots, k$, and $\mathbf{X} = (\mathbf{X}_1', \ldots, \mathbf{X}_k')'$. Let

$$\mathbf{m}_i = \frac{i}{n_i} \sum_{j=1}^{n_i} \mathbf{x}_{ij}, \; \mathbf{m} = \frac{1}{n} \sum_{i=1}^{k} \sum_{j=1}^{n_i} \mathbf{x}_{ij},$$
$$\mathbf{U}_i = \sum_{j=1}^{n_i} (\mathbf{x}_{ij} - \mathbf{m}_i)(\mathbf{x}_{ij} - \mathbf{m}_i)', \; \mathbf{U} = \sum_{i=1}^{s} \mathbf{U}_i,$$
$$\mathbf{S}_i = \frac{1}{n_i - 1} \mathbf{U}_i, \; \mathbf{W} = \frac{1}{n-k} \mathbf{U}, \; \mathbf{B} = \frac{1}{n(k-1)} \sum_{i=1}^{k} n_i (\mathbf{m}_i - \mathbf{m})(\mathbf{m}_i - \mathbf{m})',$$

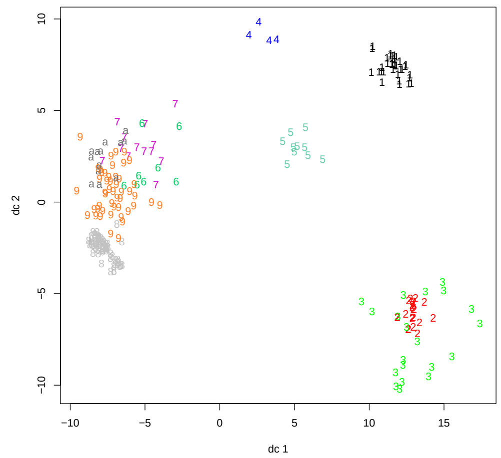
that is, $\mathbf{S}_i$ is the covariance matrix of group i with mean vector $\mathbf{m}_i$, $\mathbf{W}$ is the pooled within groups-scatter matrix and $\mathbf{B}$ is the between groups-scatter matrix.

Definition. DCs (Rao 1952) are defined by $\mathbf{Q} = \mathbf{B}$ and $\mathbf{R} = \mathbf{W}$.

Corollary. Only $k - 1$ eigenvalues of $\mathbf{W}^{-1}\mathbf{B}$ are larger than 0. The whole information about the mean differences can be displayed in $k - 1$ dimensions (cf. Gnanadesikan, 1977).

Use R-function `plotcluster` in fpc.

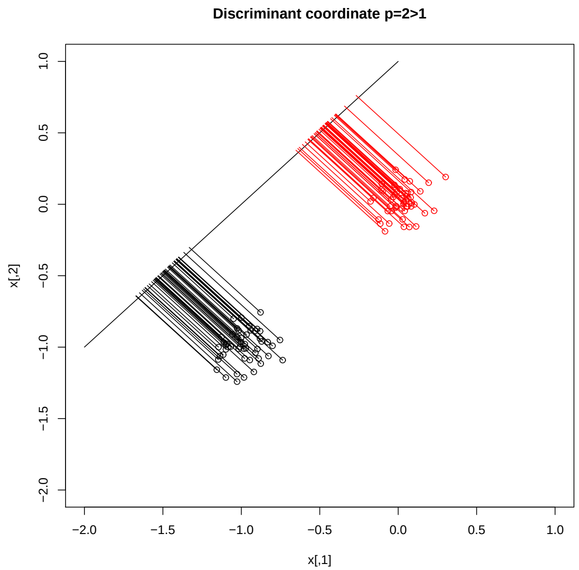

```
library(fpc)
clusym <- c(sapply(1:9,toString),"a")

plotcluster(trigonadata,trigonam$classification,
  pch=clusym[trigonam$classification])
```

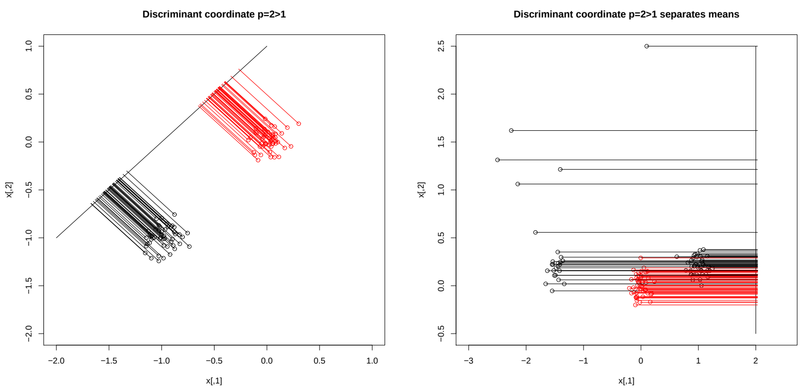



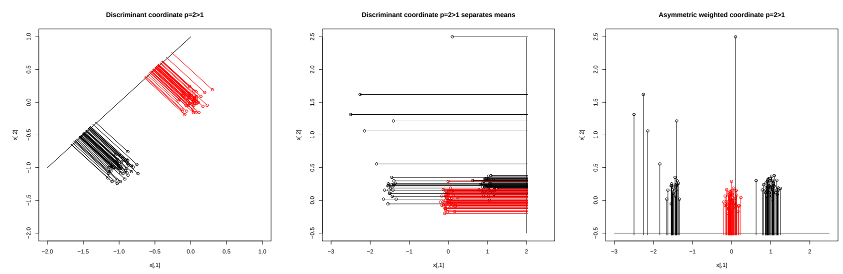
More than 3 clusters: cannot see everything in 2-d.

Difficulties with DC:

- Separation between cluster means is shown.
- All within-cluster cov-matrices equal implicitly assumed.
- More than 3 clusters: cannot see everything in 2-d.
- DCs may still be dominated by outliers.







Definition (Hennig 2005) Let

$$\mathbf{B}^* = \frac{1}{n_1 n_2} \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} (\mathbf{x}_{1i} - \mathbf{x}_{2j})(\mathbf{x}_{1i} - \mathbf{x}_{2j})',$$

denoting now by $\mathbf{x}_{2j}$ all points that are not in cluster 1. ADCs for cluster 1 are defined by $\mathbf{Q} = \mathbf{B}^*$ and $\mathbf{R} = \mathbf{S}_1$.

Definition. Let

$$\mathbf{B}^{**} = \frac{1}{n_1 \sum_{j=1}^{n_2} w_j} \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} w_j (\mathbf{x}_{1i} - \mathbf{x}_{2j})(\mathbf{x}_{1i} - \mathbf{x}_{2j})', \text{ where}$$

$$w_j = \min\left(1, \frac{d}{(\mathbf{x}_{2j} - \mathbf{m}_1)' \mathbf{S}_1^{-1} (\mathbf{x}_{2j} - \mathbf{m}_1)}\right), \ j = 1, \ldots, n_2, \qquad (1)$$

$d > 0$ being some constant, for example the 0.99-quantile of the χ^2_p-distribution.
AWCs for cluster 1 are defined by $\mathbf{Q} = \mathbf{B}^{**}$ and $\mathbf{R} = \mathbf{S}_1$.

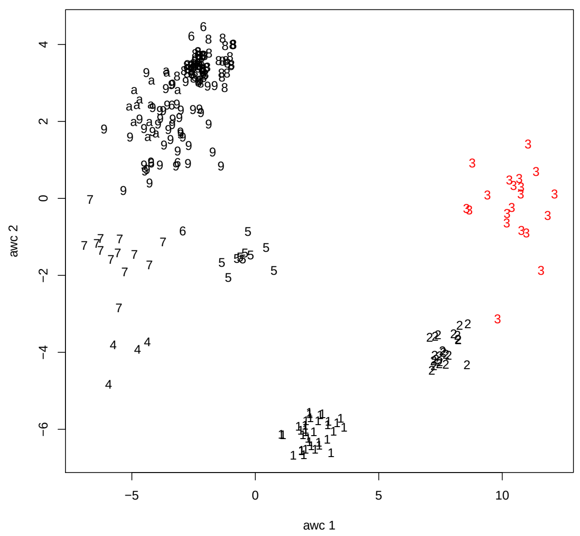
Motivation for weights: Consider $\mathbf{x}_{2j} = \mathbf{m}_1 + q\mathbf{v}$, where $\mathbf{v}$ is a unit vector w.r.t. $\mathbf{S}_1$ giving the direction of the deviation of x_{2j} from the mean $\mathbf{m}_1$ of cluster 1 and $q > 0$ is the amount of deviation. The contribution of $\mathbf{x}_{2j}$ to $\mathbf{B}^{**}$ is, for q large enough,

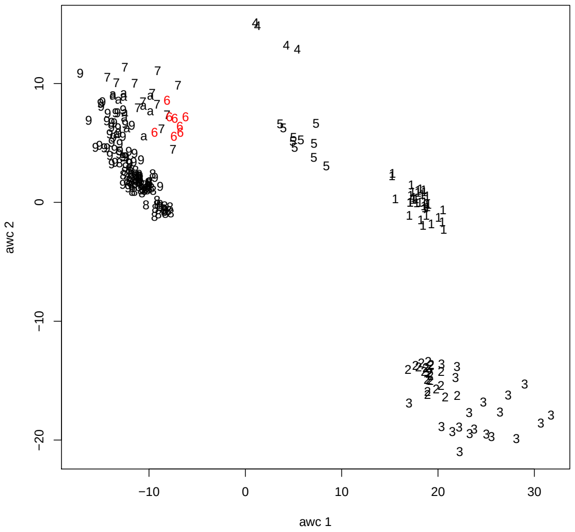
$$\sum_{i=1}^{n_1} \frac{d}{(\mathbf{x}_{2j} - \mathbf{m}_1)' \mathbf{S}_1^{-1} (\mathbf{x}_{2j} - \mathbf{m}_1)} (\mathbf{x}_{1i} - \mathbf{x}_{2j})(\mathbf{x}_{1i} - \mathbf{x}_{2j})',$$

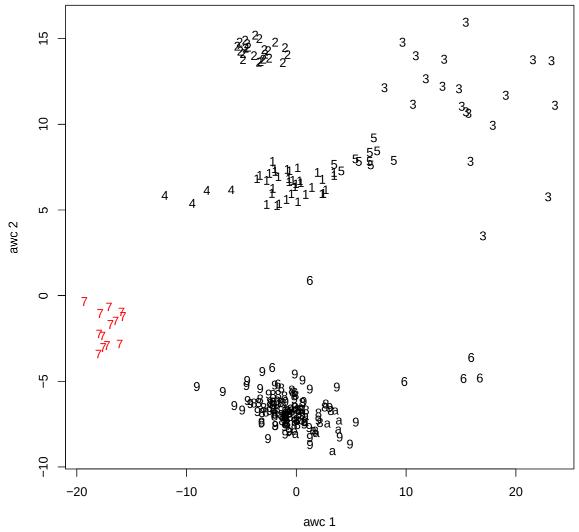
$$\to n_1 d \frac{\mathbf{v}\mathbf{v}'}{\mathbf{v}' \mathbf{S}_1^{-1} \mathbf{v}} \text{ for } q \to \infty.$$

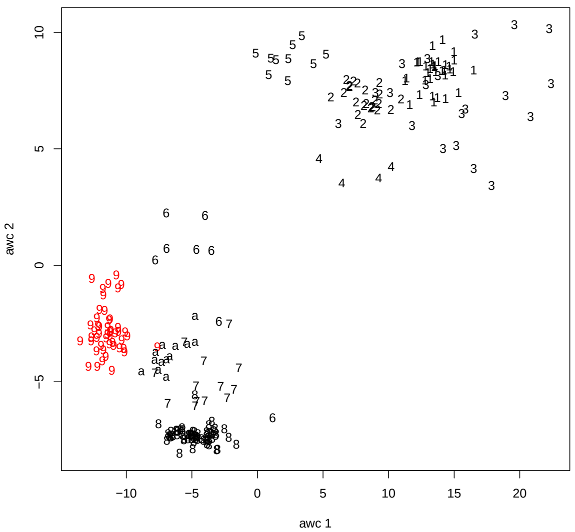
Look for a single cluster at a time.

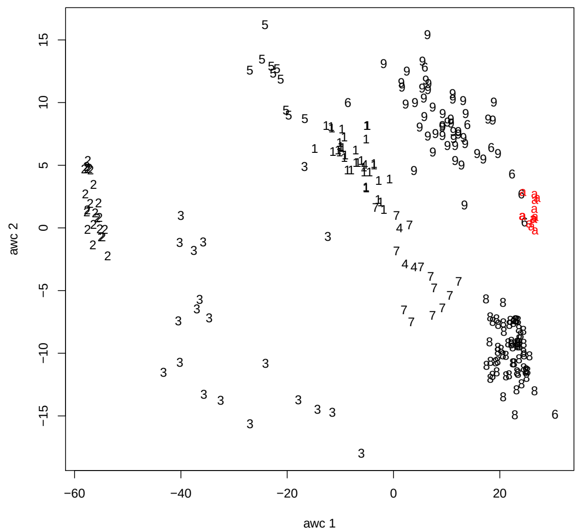
```
> plotcluster(trigonadata,trigonam$classification,
  3,method="awc",pch=clusym[trigonam$classification],
  col=1+(trigonam$classification==3))

> plotcluster(trigonadata,trigonam$classification,
  6,method="awc",pch=clusym[trigonam$classification],
  col=1+(trigonam$classification==6))
```


awc 1
awc 2
−10
0
10
20
30
−20
−10
0
10


awc 1
awc 2
−20
−10
0
10
20
−10
−5
0
5
10
15


awc 1
awc 2
−10
−5
0
5
10
15
20
−5
0
5
10


awc 1
awc 2
−60
−40
−20
0
20
−15
−10
−5
0
5
10
15

Things to keep in mind:

- Clusters can still be heterogeneous in other directions.
- Cluster may be separated but surrounded. (Check `cluster.stats`)
- Outliers are influential if members of cluster to plot. Alternative methods in Hennig (2005), `plotcluster`.

8.2 Stability assessment

General principle for stability assessment

- Generate several new datasets out of the original one.
- Cluster all these new datasets.
- Define statistic to formalise how similar new clusterings are to the original one.
- If they are very similar, it's stable.

Most clusterings are unstable in one way or another.
Want to know which clusters are stable
$\Rightarrow$ here *cluster-wise* methodology,
`clusterboot` in package fpc (Hennig 2007).

1. Use the Jaccard coefficient
$$\gamma(C, D) = \frac{|C \cap D|}{|C \cup D|}.$$
to measure similarity between two subsets of a set.
2. Repeat B times steps 2-4:
resample new data sets from the original one,
3. apply the same clustering method to them.
4. For $C \in \mathcal{C}$ record $m_i = \max_{D \neq C} \gamma(C, D)$
5. Use $\bar{\gamma} = \frac{1}{B} \sum_{i=1}^{B} m_i$ to assess stability of C.

Various methods to resample are possible.

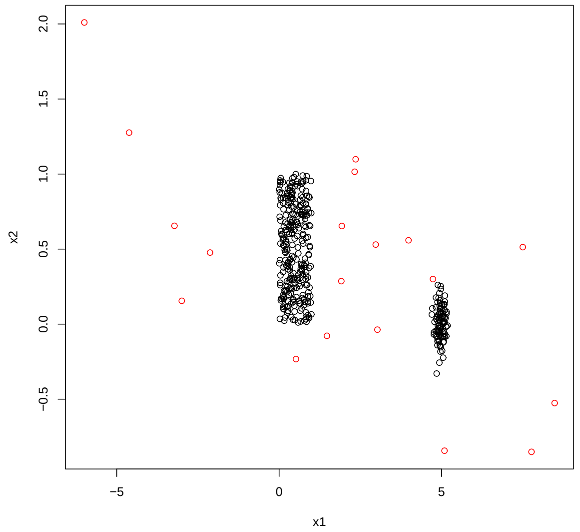
Use two different methods,
can discover different kinds of instability.

Bootstrap method discarding multiple points

Replacement by noise Draw 5%, say, of points and replace them by uniform “noise”.

1. Sphere the dataset to unit covariance matrix.
2. Draw points from $U[-4,4]^p$.
3. Rotate data back.

Problem with bootstrap: can only increase separation.
Problem with noise: unclear what “realistic” noise would be.



For computing γ for given original cluster and cluster in resampled dataset,
use only points that are both in original dataset and in resampled one.

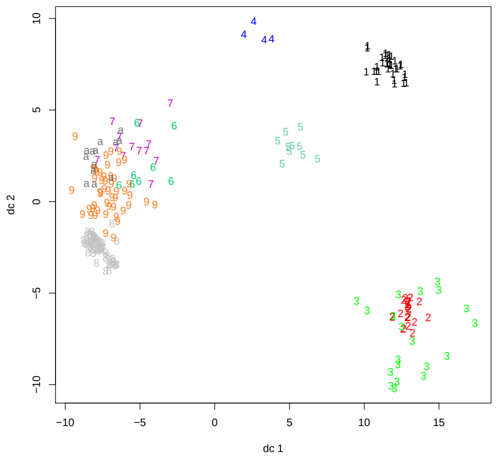
In practice, use $B = 100$ if time allows.
But need some patience.

Interpretation:

- 0.5 is minimum v so that for given partition it’s possible for every cluster to find another partition so that maximum γ is $\leq v$.
- New partition with m clusters, original one with $k > m \Rightarrow \exists$ at least $k - m$ clusters in original partition for which no $\gamma > v$.

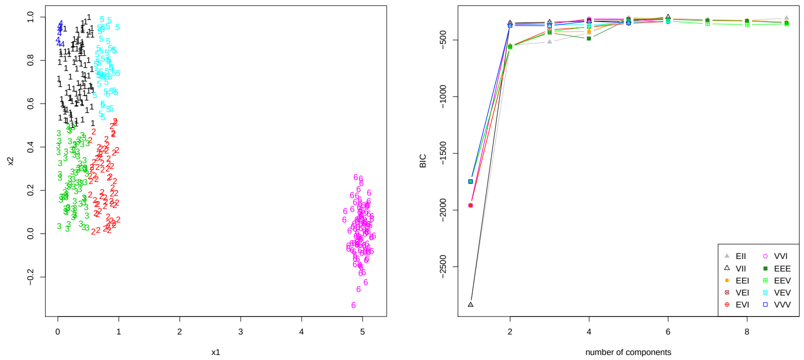
Consider clusters with max $\gamma \leq 0.5$ as “dissolved”.
Demand $\bar{\gamma} >> 0.5$ for stability.

```
> trigonaboot <- clusterboot(trigonadata,B=20,
  multipleboot=FALSE,
  clustermethod=noisemclustCBI,nnk=0,G=1:15)

* Cluster stability assessment *
Cluster method:  mclustBIC
Full clustering results are given as parameter result
of the clusterboot object, which also provides
further statistics of the resampling results.
Number of resampling runs:  20

Number of clusters found in data:  10
```


```
 Clusterwise Jaccard bootstrap (omitting multiple points) mean:
 [1] 1.0000000 1.0000000 0.9493590 0.9236111 0.9833333
0.6884722 1.0000000
 [8] 0.9955763 0.9820907 0.9156313
dissolved:
 [1] 0 0 0 2 0 3 0 0 0 0
recovered:
 [1] 20 20 20 18 20  7 20 20 20 19
 Clusterwise Jaccard replacement by noise mean:
 [1] 1.0000000 1.0000000 0.9034211 0.9687500 1.0000000
0.5488095 1.0000000
 [8] 0.9974430 1.0000000 0.9288795
dissolved:
 [1]  0  0  0  1  0 13  0  0  0  0
recovered:
 [1] 20 20 20 19 20  1 20 20 20 20
```


Example where uniform is split up into Gaussians.



```
set.seed(234567)
x1 <- runif(300,0,1)
x2 <- runif(300,0,1)
x3 <- rnorm(100,5,0.1)
x4 <- rnorm(100,0,0.1)
x <- rbind(cbind(x1,x2),cbind(x3,x4))
mx <- mclustBIC(x)
smx <- summary(mx,x)
plot(x,col=smx$classification,
     pch=clusym[smx$classification])

uniboot <- clusterboot(x,B=20,multipleboot=FALSE,
                 clustermethod=noisemclustCBI,nnk=0)
```



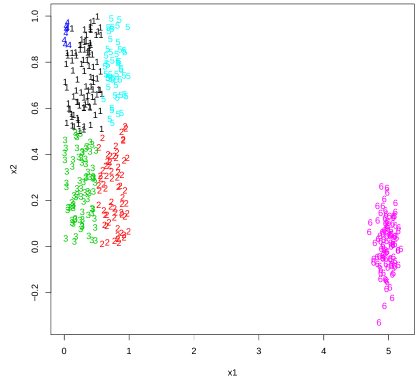
```
(For uniform plus Gaussian dataset)
* Cluster stability assessment *
Cluster method:  mclustBIC
Number of resampling runs:  20

Number of clusters found in data:  6

 Clusterwise Jaccard bootstrap (omitting multiple points) mean:
[1] 0.78226138 0.90698801 0.93042938 0.08628977 0.81728134
1.00000000
dissolved:
[1]  2  1  1 20  1  0
recovered:
[1] 17 18 18  0 18 20
 Clusterwise Jaccard replacement by noise mean:
[1] 0.35669233 0.26304825 0.31162945 0.07258365 0.19932778
1.00000000
dissolved:
[1] 17 20 17 19 20  0
recovered:
[1]  0  0  0  1  0 20
```


Instabilities can result from

- features of the data,
- instabilities of clustering method,
- mismatch between the two.

Stable clusters are not necessarily good.
(Fixing $k = 1$ is always stable.)
Unstable clusters can be tolerated if stability is not the aim.

9. Merging Gaussian components



mclustBIC may fit homogeneous non-Gaussian sets
by too many components.

May want to merge components that “belong together” in a clustering sense.

Problem: “mixture of mixtures” is not identifiable.
Not a statistical estimation problem.

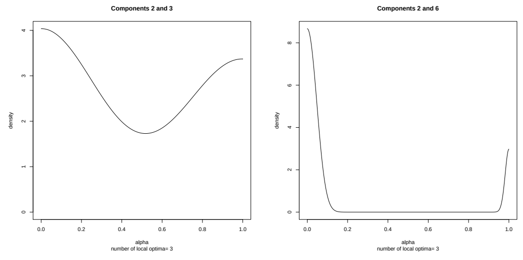
Need to formalise “component similarity”.
There are various possibilities,
implemented in fpc’s `mergenormals` (Hennig 2010).

The ridgeline (Ray and Lindsay 2005)

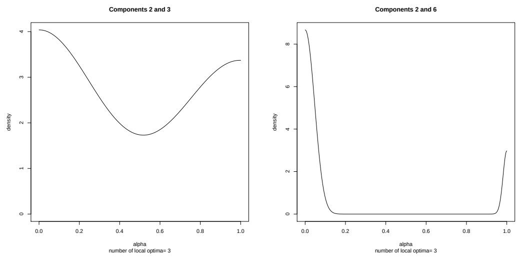
Density on $k-1$-dimensional manifold containing all density extrema of k-component Gaussian mixture
$\Rightarrow$ 1-d density for 2-component Gaussian.

$$\mathbf{x}^*(\alpha) = [(1-\alpha)\Sigma_1^{-1} + \alpha\Sigma_2^{-1}]^{-1}[(1-\alpha)\Sigma_1^{-1}\mathbf{a}_1 + \alpha\Sigma_2^{-1}\mathbf{a}_2],$$

$\alpha \in [0,1]$.



Ridgelines can be evaluated easily for 2 components.
Ridgeline ratio: $r =$ratio minimum/min.maximum density.



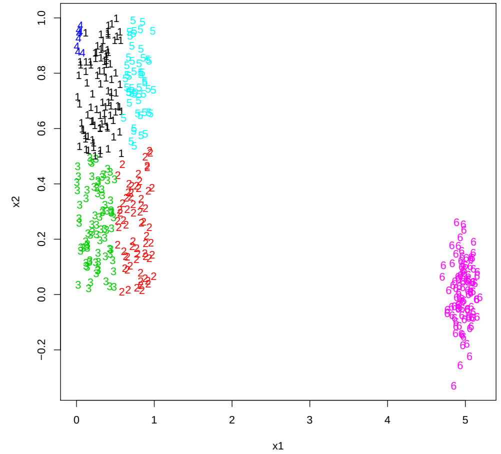
Should not insist on unimodality for merging ($r = 1$), because mclustBIC separates tiny insignificant gaps.
Suggest merge for $r \geq 0.2$.

How to join more than two components?
Hierarchically...

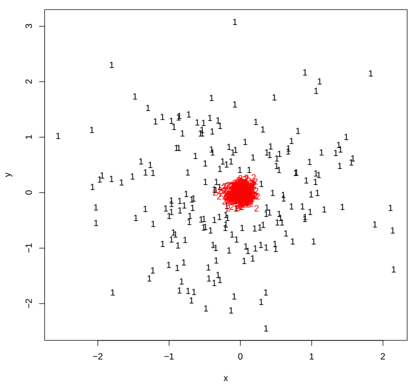
1. Compute all pairwise ridgeline ratios.
2. Unless all ratios below cutoff,
   join pair of components with max. ratio.
3. Recompute mean and cov-matrix for new cluster.
4. Go to 1.

```
> mnx <- mergenormals(x,smx,method="ridge.ratio")
# could specify cutoff=0.2
> summary(mnx)
* Merging Gaussian mixture components *

 Method:  ridge.ratio , cutoff value:  0.2
 Original number of components:  6
 Number of clusters after merging:  2
 Values at which clusters were merged:
     [,1]         [,2]
[1,]    5 6.257516e-01
[2,]    4 5.004525e-01
[3,]    3 6.990044e-01
[4,]    2 2.071673e-01
[5,]    1 4.856773e-30
 Components assigned to clusters:
     [,1]
[1,]    1
[2,]    1
[3,]    1
[4,]    1
[5,]    1
[6,]    2
```


This merges 1-5, as it should.



However, one may not always waht to merge for modality.



Alternative methods available in Hennig(2010), `mergenormals`

References

- Banfield, J. D. and Raftery, A. E. (1993) Model-Based Gaussian and Non-Gaussian Clustering, *Biometrics* 49, pp. 803-821.
- Byers, S. and Raftery, A. E. (1998), "Nearest-Neighbor Clutter Removal for Estimating Features in Spatial Point Processes", *Journal of the American Statistical Association*, 93, pp. 577-584.
- Coretto, P. and Hennig, C. (2010) A simulation study to compare robust clustering methods based on mixtures. *Advances in Data Analysis and Classification* 4, 111-135.
- Dempster, A. P., Laird, N. M. and Rubin, D. B. (1977) Maximum likelihood from incomplete data via the EM algorithm, *Journal of the Royal Statistical Society B*, 39, pp. 1-38.

- Fraley, C. and Raftery, A. E. (2002) Model-based clustering, discriminant analysis, and density estimation, *Journal of the American Statistical Association* 97, 611-631.
- Fraley, C. and Raftery, A.E. (2007) Bayesian Regularization for Normal Mixture Estimation and Model-Based Clustering. *Journal of Classification*, 24, 155-181.
- Fraley, C. and Raftery, A. E. (2010) MCLUST Version 3 for R: Normal Mixture Modeling and Model-based Clustering, Technical Report No. 504, Department of Statistics, University of Washington, September 2006 (revised July 2010).
- Hennig, C. (2005) "Asymmetric linear dimension reduction for classification", *Journal of Computational and Graphical Statistics*, 13, 930-945.

- Hennig, C. (2007) "Clusterwise assessment of cluster stability", *Computational Statistics and Data Analysis*, 52, 258-271.
- Hennig, C. (2010) Methods for merging Gaussian mixture components. *Advances in Data Analysis and Classification* 4, 3-34.
- Keribin, C. (2000) Consistent estimation of the order of a mixture model, *Sankhya A*, 62, 49-66.
- R Development Core Team (2011) *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, `http://www.R-project.org`
- Rao, C. R. (1952) *Advanced Statistical Methods in Biometric Research*, Wiley, New York.
- Schwarz, G. (1978) Estimating the dimension of a model, *Annals of Statistics* 6, pp. 461-464.

- Yakowitz, S. J. and Spragins, J. D. (1968) On the identifiability of finite mixtures. *Annals of Mathematical Statistics* 39, 209-214.