Latent class analysis

Latent profile analysis a.k.a. Gaussian mixture modeling

DL Oberski & L Boeschoten

Learning goals

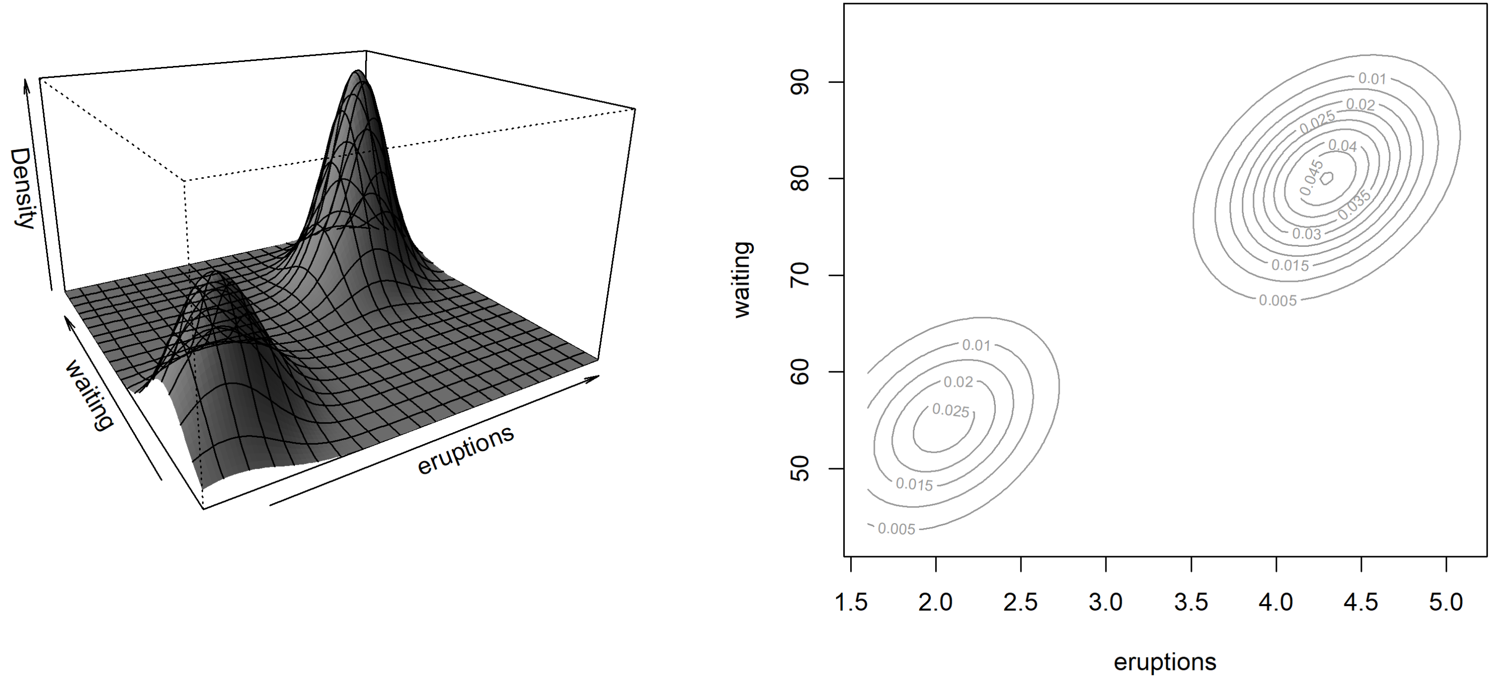
- Understand and apply basic latent profile analysis (a.k.a. Gaussian mixture modeling)
- Apply further clustering evaluation techniques
- Understand & apply component merging
- Use `mclust` in R

Multivariate model-based clustering

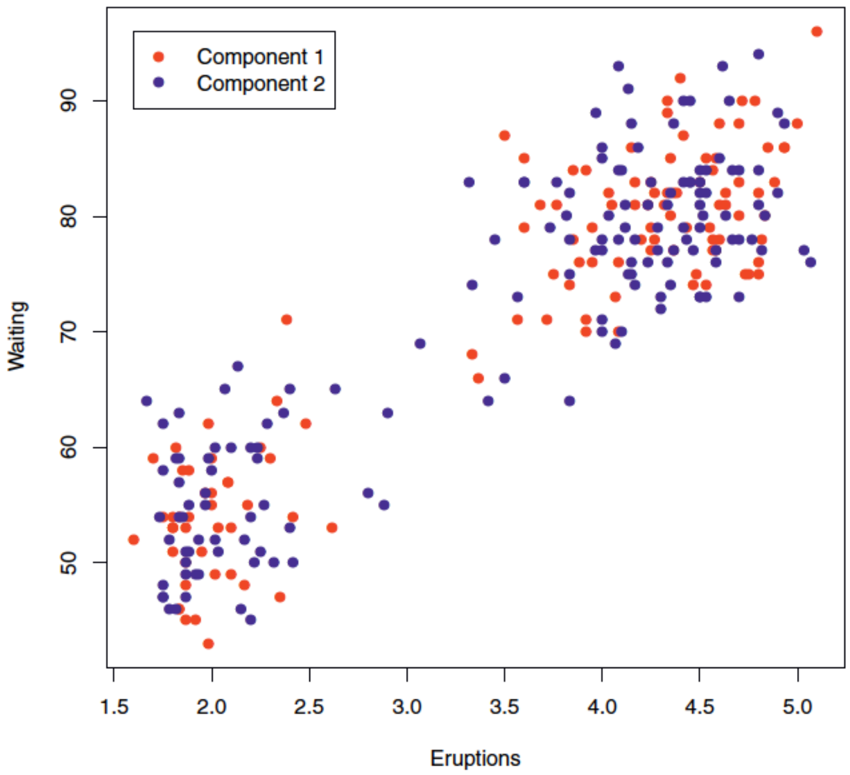
- With 2 observed features:
  - mean becomes a vector of 2 means
  - standard deviation turns into a 2x2 variance-covariance matrix determining the shape of the cluster
- So we have multiple within-cluster parameters:
  - Two means
  - Two variances, one for each observed variable
  - A single covariance among the features
- Together, the 11 parameters define the likelihood in bivariate space, which from the top looks like ellipses

Multivariate model-based clustering

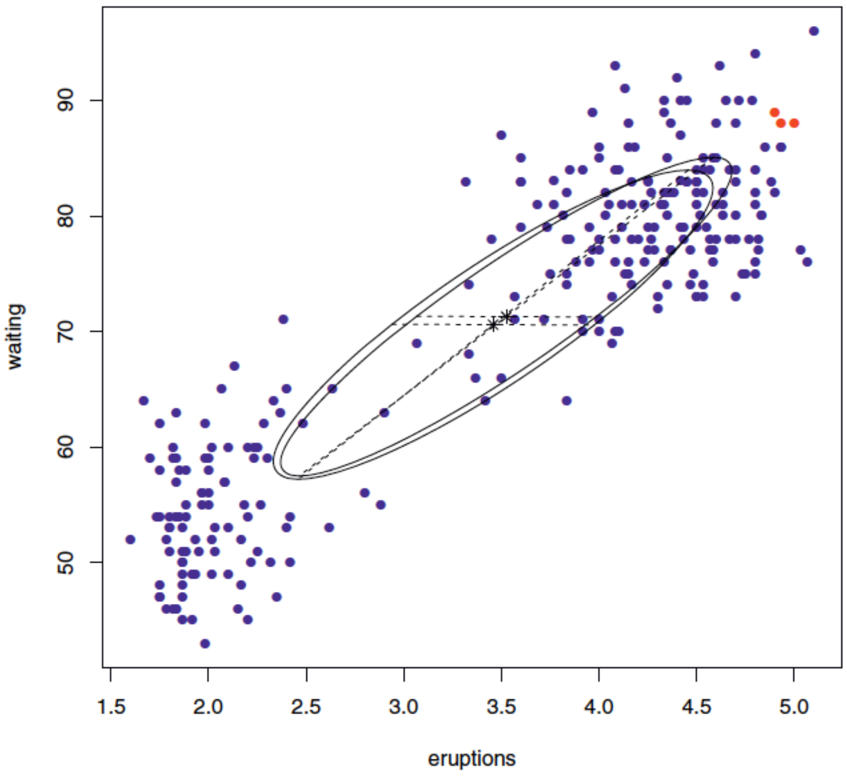
$$p(\boldsymbol{y}|\,\theta) = \pi_1^X MVN(\boldsymbol{\mu_1}, \boldsymbol{\Sigma_1}) + (1 - \pi_1^X) MVN(\boldsymbol{\mu_2}, \boldsymbol{\Sigma_2})$$



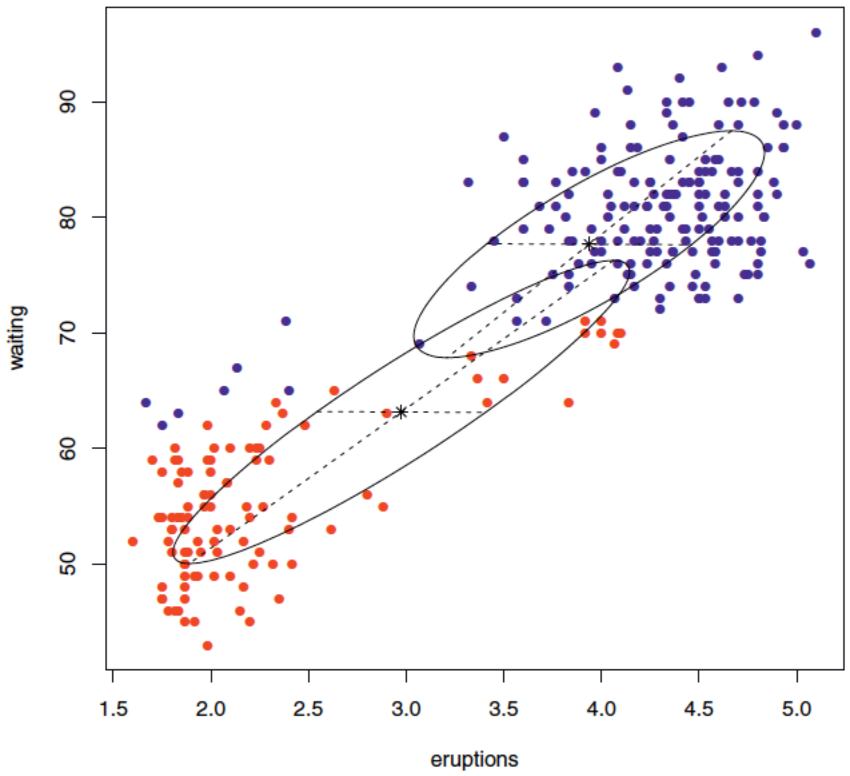
EM algorithm for Gaussian mixture model (LPA)



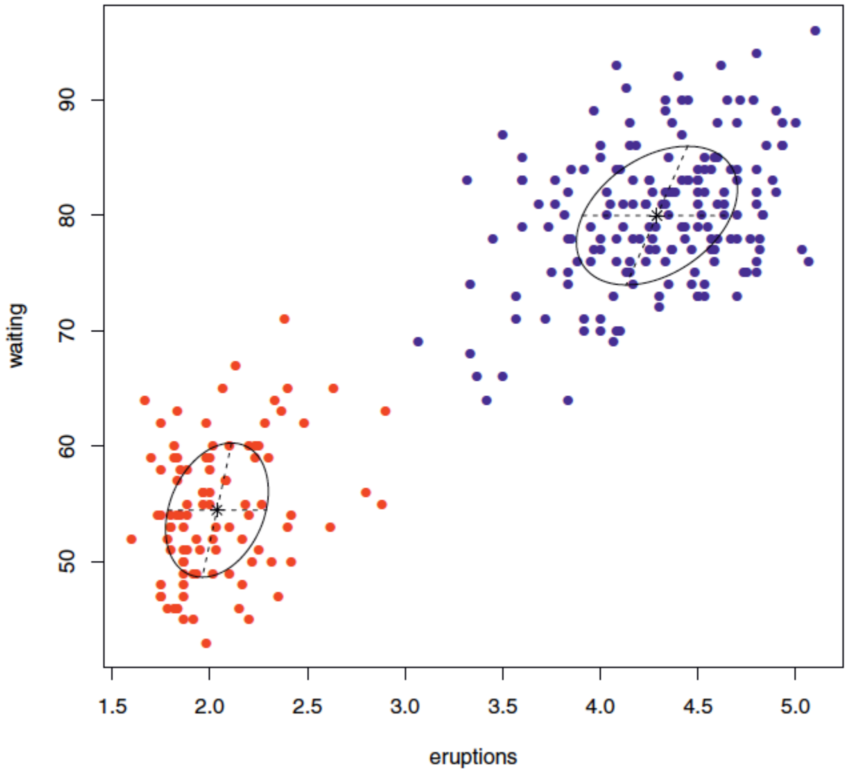
Component 1
Component 2
Waiting
Eruptions
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
50
60
70
80
90



waiting
50
60
70
80
90
eruptions
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0



waiting
eruptions
50
60
70
80
90
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0



waiting
eruptions
50
60
70
80
90
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0

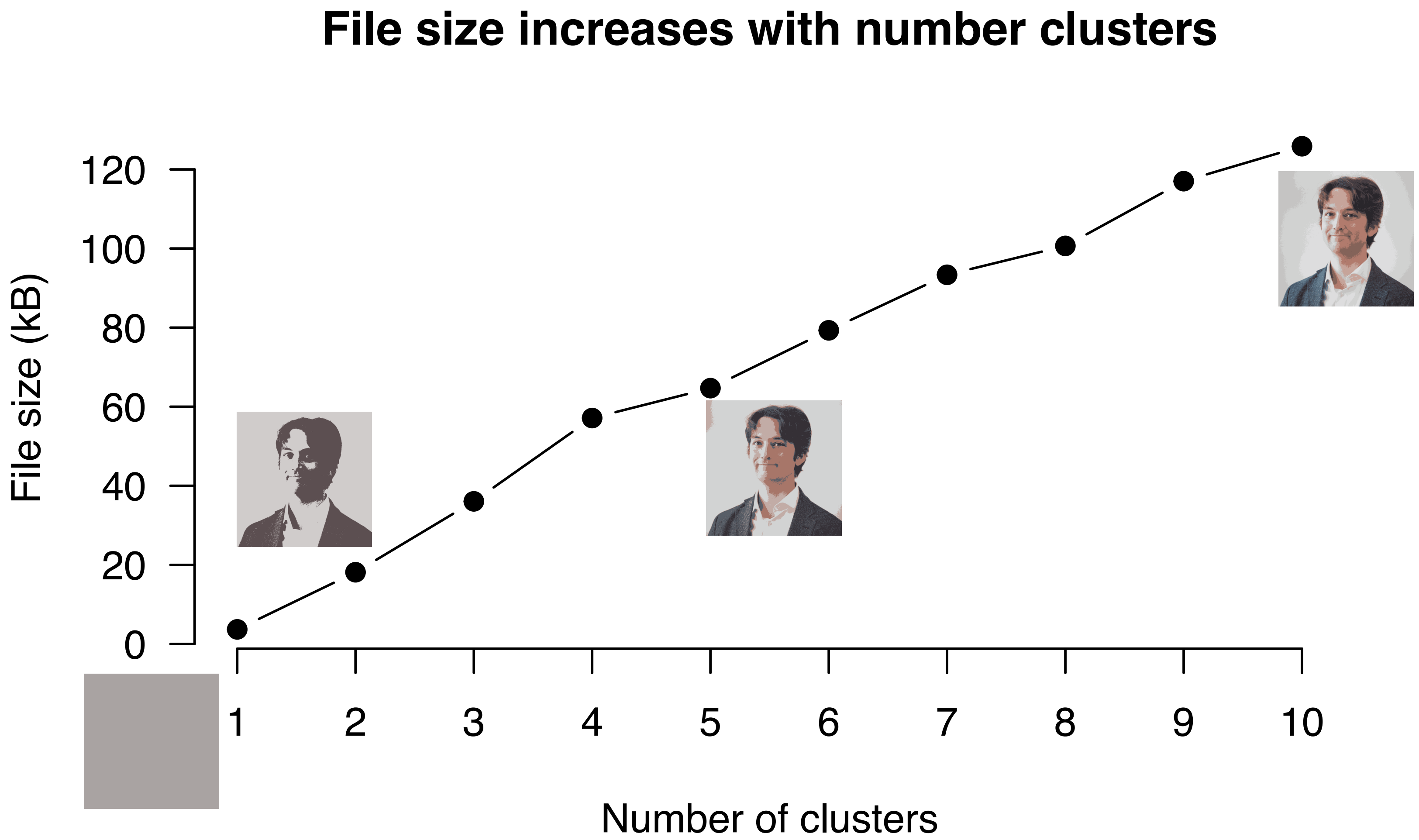
Multivariate model-based clustering

- Cluster shape parameters (the variance-covariance matrix) *can* be constrained to be *equal* across clusters
- Can also be *different* across clusters
- More flexible, complex model
- Think: **bias-variance tradeoff**

How to evaluate clustering results

1. Use of external information
2. Visual exploration
3. Stability assessment / sensitivity analysis
4. Internal validation indexes
5. **Testing for clustering structure**

Much more info & helpful advice: Clustering strategy & method selection (ch 31 of Handbook of clustering), https://arxiv.org/pdf/1503.02059.pdf


File size increases with number clusters
File size (kB)
0
20
40
60
80
100
120
1
2
3
4
5
6
7
8
9
10
Number of clusters

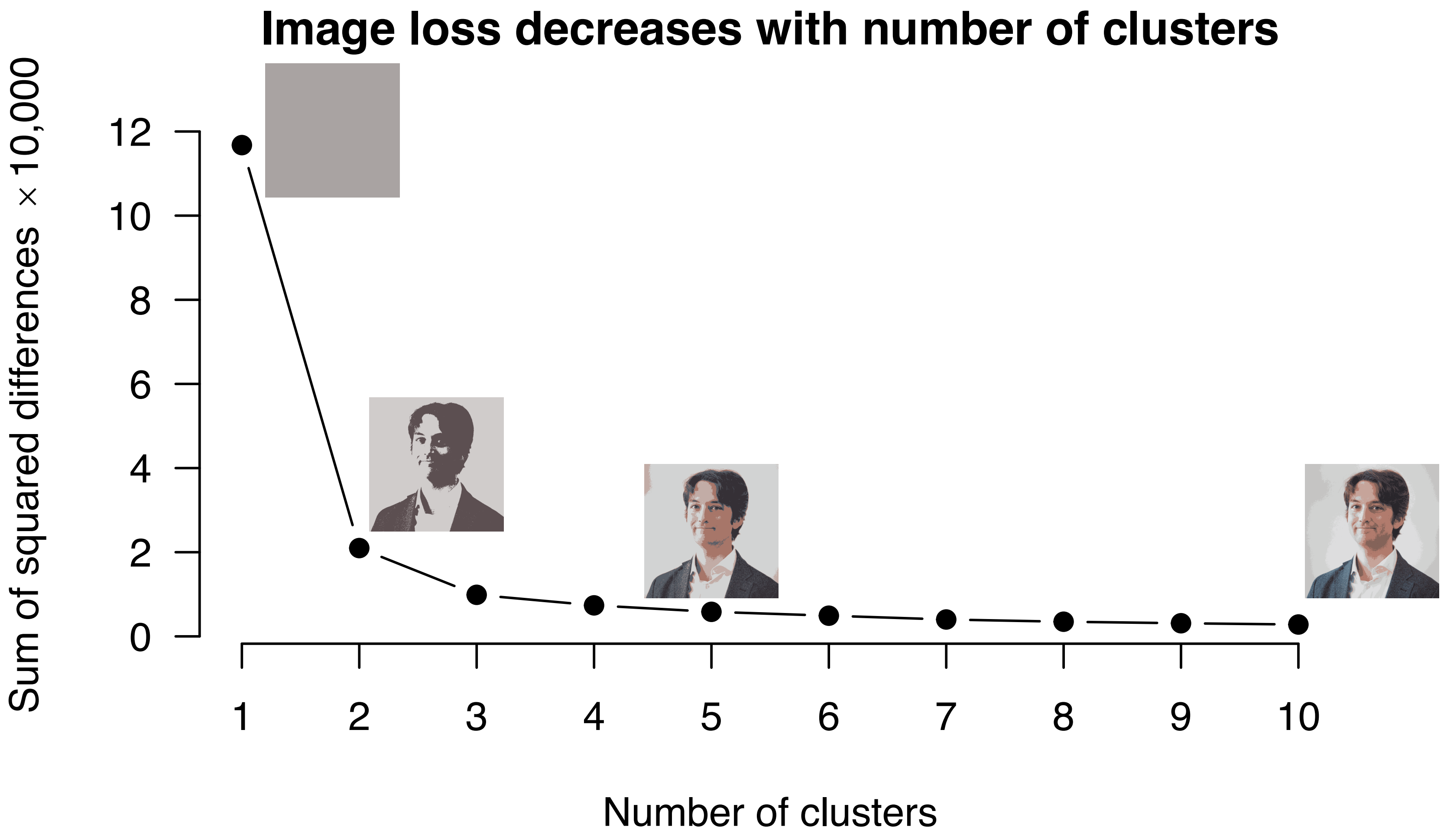
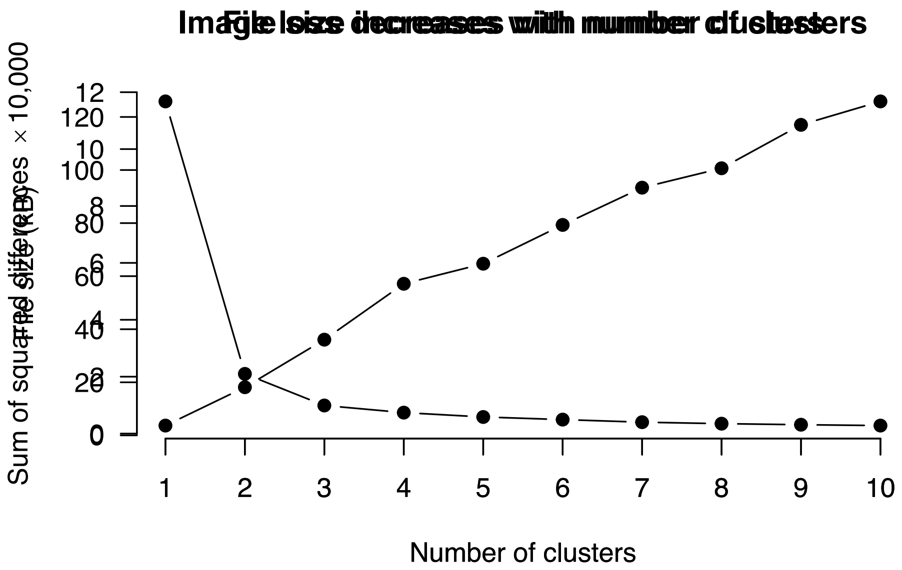


Image loss decreases with number of clusters
Sum of squared differences × 10,000
0
2
4
6
8
10
12
1
2
3
4
5
6
7
8
9
10
Number of clusters



- More clusters gives **better “fit”** in terms of reconstruction of the image
  (compression is less “lossy”)
- More clusters gives **bigger file size**
  (solution is more complex, takes more bytes to store)
- So the **model loss and model complexity trade off against each other**
- This is a common theme in (unsupervised) machine learning.

Model fit criteria

- BIC: “Schwarz/Bayesian information criterion”
- AIC: “Another/Akaike information criterion”
  (same as BIC but penalty is m)
- AIC3: The same as AIC but penalty is $\frac{3}{2}m$
- ICL: “Integrated information criterion” (Biernacki et al. 2000)
  (Same as BIC but penalized by entropy of classification)
- *(Others based on):*
  - *Minimum description length (MDL)*
  - *Bayesian marginal likelihood*

Model-based clustering in R

- `mclust` implements multivariate model-based clustering
- Provides an easy interface to fit several parameterizations
- Model comparison with BIC
- Plotting functionality

```
> library(mclust)
    __  ___________    __  _____________
   /  |/  / ____/ /   / / / / ___/_  __/
  / /|_/ / /   / /   / / / /\__ \ / /
 / /  / / /___/ /___/ /_/ /___/ // /
/_/  /_/\____/_____/\____//____//_/    version 5.4.6
```

The full model

- We again have a K-mixture model, this time for multivariate continuous variables $\boldsymbol{y}$, whose p.d.f., f, is modeled as

$$f(\boldsymbol{y}) = \sum_{k=1}^{K} \pi_k \cdot \mathrm{MVN}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

- Where $\boldsymbol{\mu}_k$ is the class-specific mean vector and
- Where $\boldsymbol{\Sigma}_k$ is the class-specific covariance matrix
- Note: no conditional independence assumption by default
- Would correspond to $\boldsymbol{\Sigma}_k$ being a diagonal matrix for each k

Covariance matrices in `mclust`

- All the models used in mclust are multivariate normal (Gaussian) (key assumption 1)
- Further ”juice” is in structure of the covariance matrices, $\boldsymbol{\Sigma}_k$
- Would be tedious to restrict the elements of these matrices directly
- Therefore the mclust people use a trick :

$$\Sigma_g = \lambda_g D_g A_g D_g^T.$$

- This trick is very closely related to factor analysis with orthogonal factors, where D would be loading matrix, and A a diagonal “relative factor variance” matrix, with overall amount of variance lambda

Covariance matrices in `mclust`

"Volume-Shape-Orientation” decomposition:

$$\Sigma_g = \lambda_g D_g A_g D_g^T.$$

- λ_g: volume
- A_g: shape
- D_g: orientation

- Each of these can be equal (E) or different (V) across classes
- In addition, A_g and/or D_g can be “identity” (I)

Model-based clustering in R

- Mclust uses an identifier for each possible parametrization :
- **E** for **e**qual, **V** for **v**ariable, **I** for identity matrix:
  - Volume (size of the clusters in data space):
  - Shape (circle or ellipse)
  - Orientation (the angle of the ellipse)

- E.g. an "EEE" model has equal volume, shape and orientation
- A VVV model has variable volume, shape, and orientation
- A VVE model has variable volume and shape but equal orientation

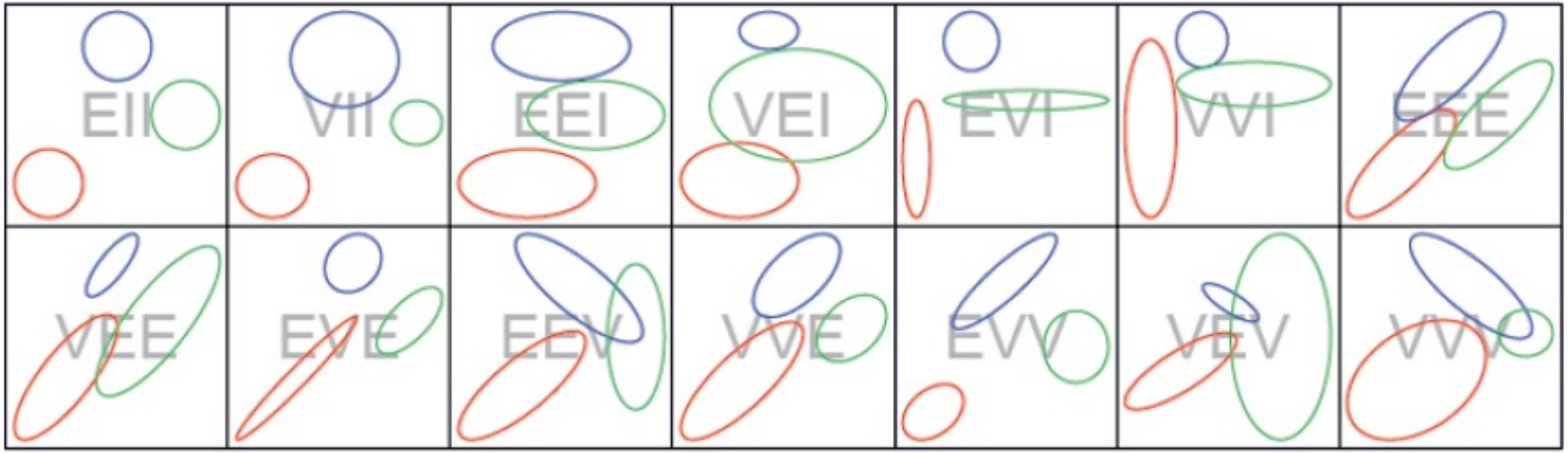


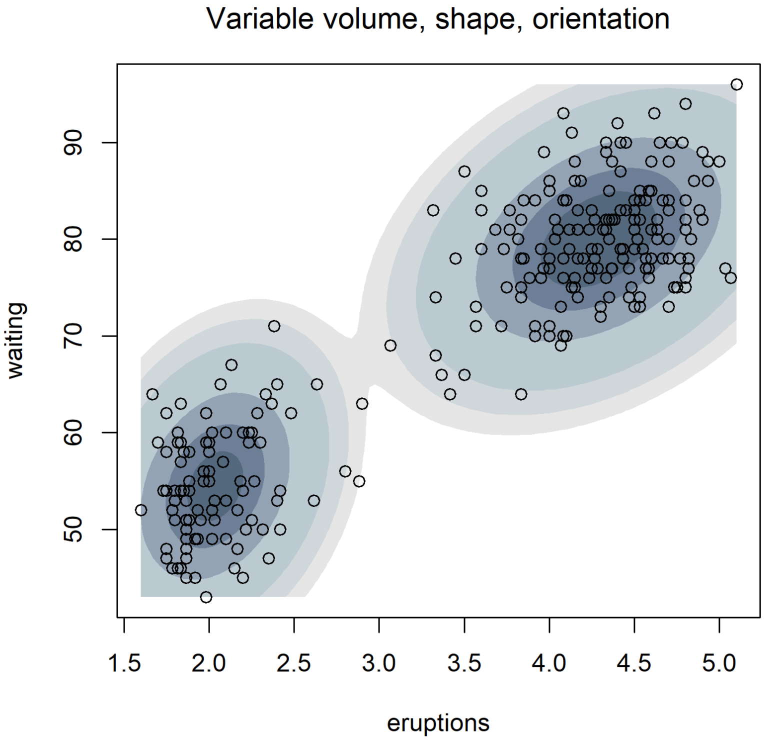
Figure 2.3 Models used in model-based clustering: examples of contours of the bivariate normal component densities for the 14 parameterizations of the covariance matrix used in model-based clustering.

Source: Bouveyron et al. (2021)

Identifier	Model	Distribution	Volume	Shape	Orientation
E		Univariate	Equal		
V		Univariate	Variable		
EII	λI	Spherical	Equal	Equal	NA
VII	$\lambda_g I$	Spherical	Variable	Equal	NA
EEI	λA	Diagonal	Equal	Equal	Axis-aligned
VEI	$\lambda_g A$	Diagonal	Variable	Equal	Axis-aligned
EVI	λA_g	Diagonal	Equal	Variable	Axis-aligned
VVI	$\lambda_g A_g$	Diagonal	Variable	Variable	Axis-aligned
EEE	Σ	Ellipsoidal	Equal	Equal	Equal
VEE	$\lambda_g DAD^T$	Ellipsoidal	Variable	Equal	Equal
EVE	$\lambda DA_g D^T$	Ellipsoidal	Equal	Variable	Equal
EEV	$\lambda D_g AD_g^T$	Ellipsoidal	Equal	Equal	Variable
VVE	$\lambda_g DA_g D^T$	Ellipsoidal	Variable	Variable	Equal
EVV	$\lambda D_g A_g D_g^T$	Ellipsoidal	Equal	Variable	Variable
VEV	$\lambda_g D_g AD_g^T$	Ellipsoidal	Variable	Equal	Variable
VVV	Σ_g	Ellipsoidal	Variable	Variable	Variable

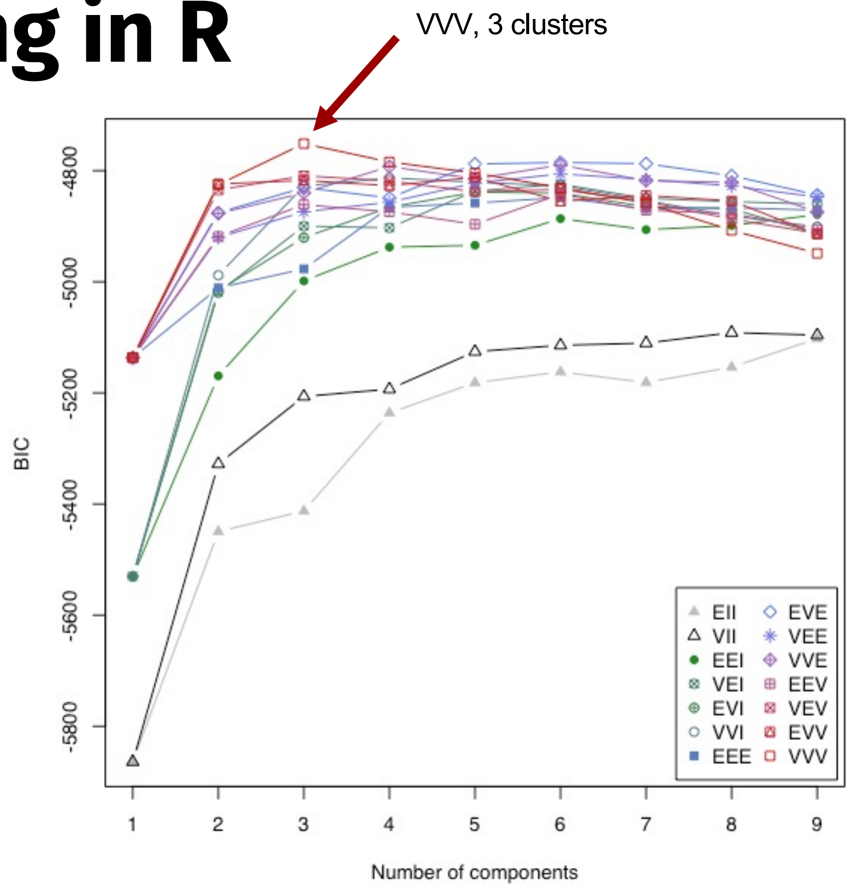
Source: Bouveyron et al. (2021)

Model-based clustering in R:
EEE vs. VVV



Model-based clustering in R

- How `mclust` optimizes hyperparameters:
  - Fit all the models with up to 9 clusters (or more, your choice!)
  - Compute the BIC (or ICL) of each model
  - Choose the model with the best BIC



```
> fit_mc <- Mclust(im_ar, G = 1:10)
fitting ...
  |========================================================| 100%
> summary(fit_mc)
----------------------------------------------------
Gaussian finite mixture model fitted by EM algorithm
----------------------------------------------------
Mclust VVV (ellipsoidal, varying volume, shape, and orientation)
model with 8 components:

 log-likelihood      n df     BIC     ICL
        3808542 640000 79 7616028 7530927

Clustering table:
     1      2      3      4      5      6      7      8
151032  48661 155542  34602  82621  49494  41665  76383
```

GMM in Latent GOLD

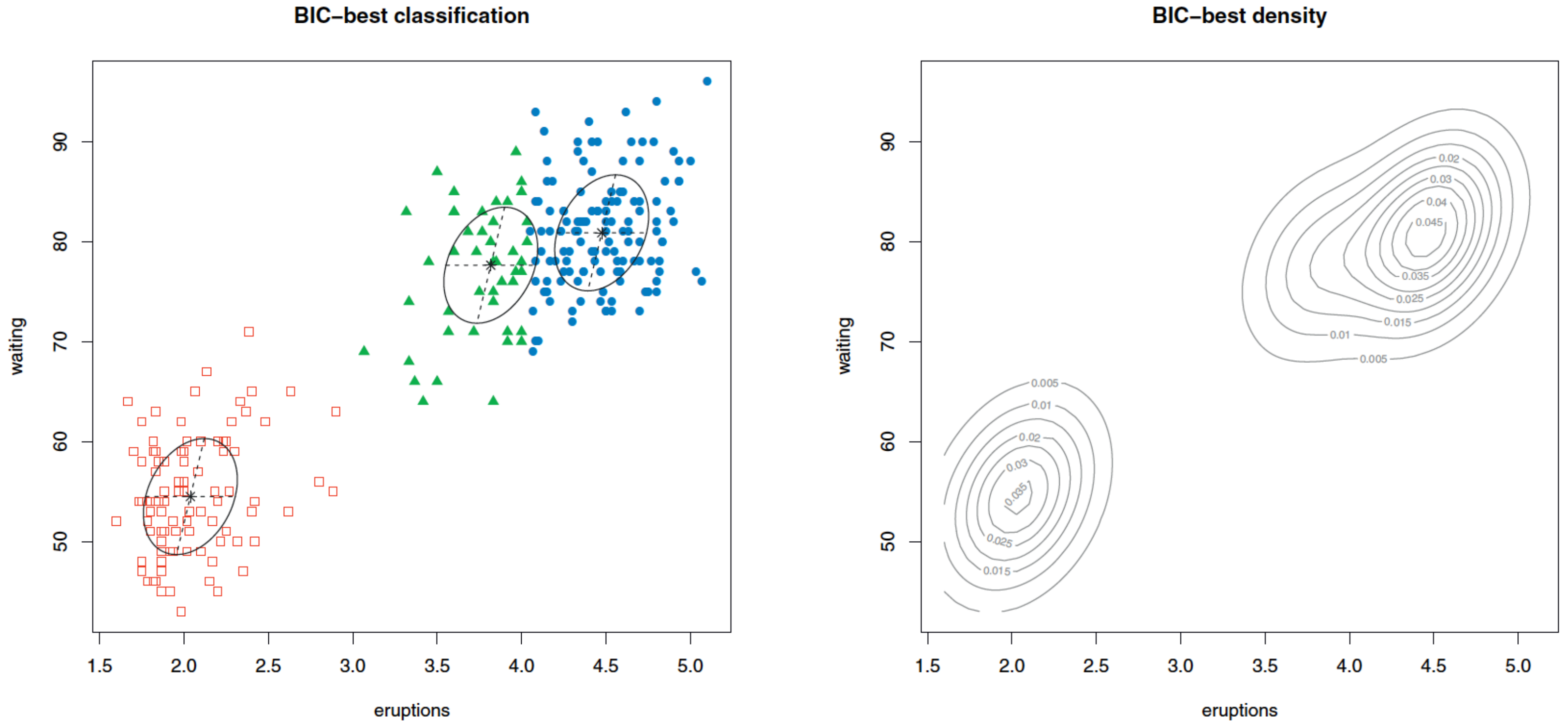
- Latent GOLD does not expose the same covariance structure parametrizations as mclust does.
- It implicitly uses a specific kind of parametrization by assuming a diagonal covariance matrix by default, meaning:
  - No correlation between variables within components.
  - Each component has independent variances for each variable.
  - Equivalent to “VVI” in mclust

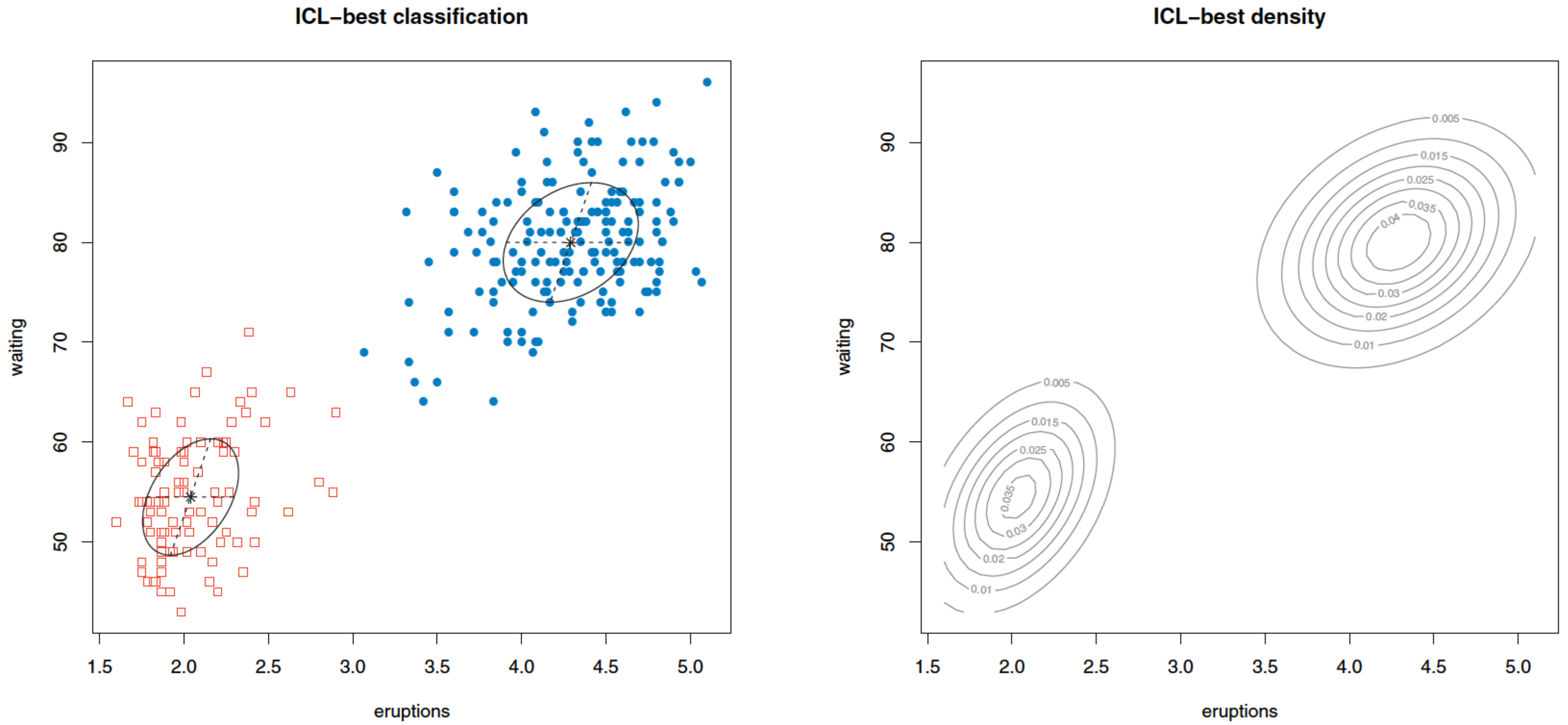
What if “clusters” are not normally distributed?

What if our “clusters” are not normally distributed?

- Example:

- At this point we could distinguish between:
  a. “clusters”: the groups we are actually interested in
  b. “mixture components”: the groups we find when assuming multivariate normal distributions (Gaussians)
- What can we do when clusters ≠ components?
- Two ideas discussed here:
  1. Use ICL or entropy directly to select classes
  2. Use component merging


BIC−best classification
waiting
eruptions
BIC−best density
waiting
eruptions
0.005
0.01
0.015
0.02
0.025
0.03
0.035
0.04
0.045


ICL–best classification
ICL–best density
waiting
eruptions
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
50
60
70
80
90
0.005
0.01
0.015
0.02
0.025
0.03
0.035
0.04

Using entropy (e.g. ICL) to select K

- Provided there is good separation between our clusters,
- Using ICL (entropy) to select number of “classes” gives:
  - The right number of components, and
  - The components more or less correspond to the clusters

- Disadvantage: the model does not fit very well
- (density is not well estimated)
- Can matter for generative models

Merging *components* to get *clusters*

- GMM obviously has trouble with clusters that are not ellipses
- Secret weapon: **merging**

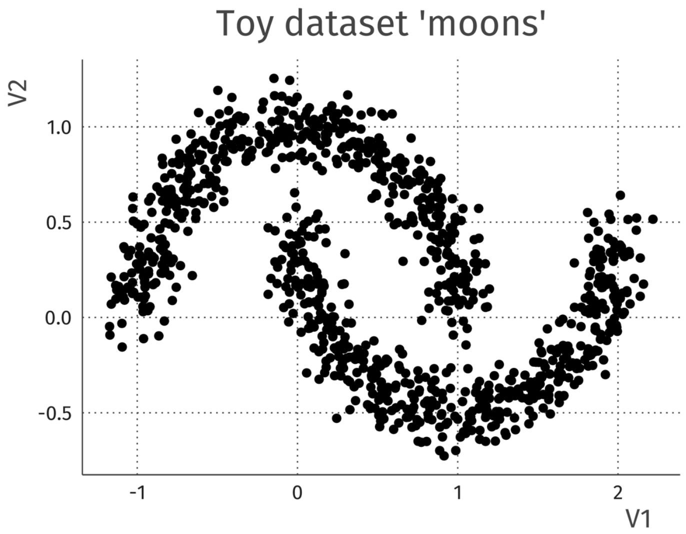
Powerful idea:

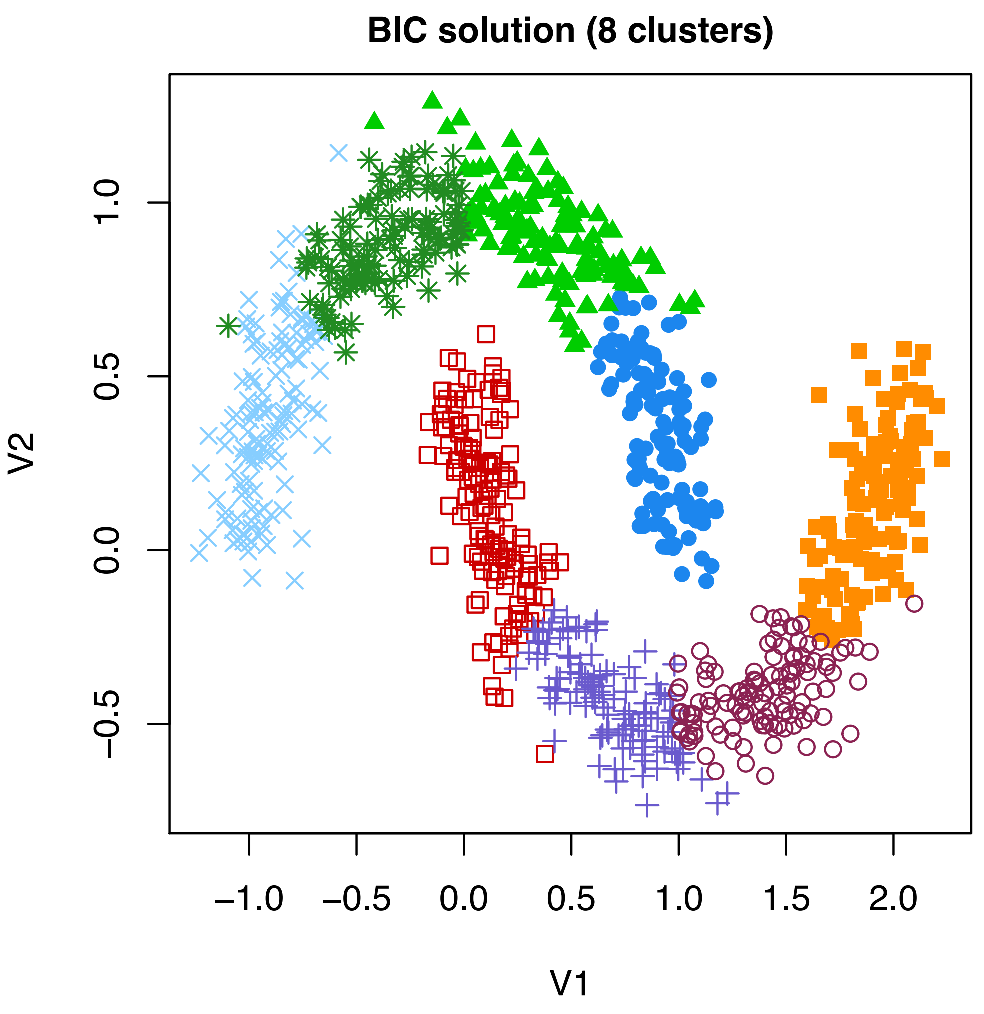
- Start out with the usual Gaussian mixture solution;
- **merge** “similar” *components* to create non-Gaussian *clusters*.

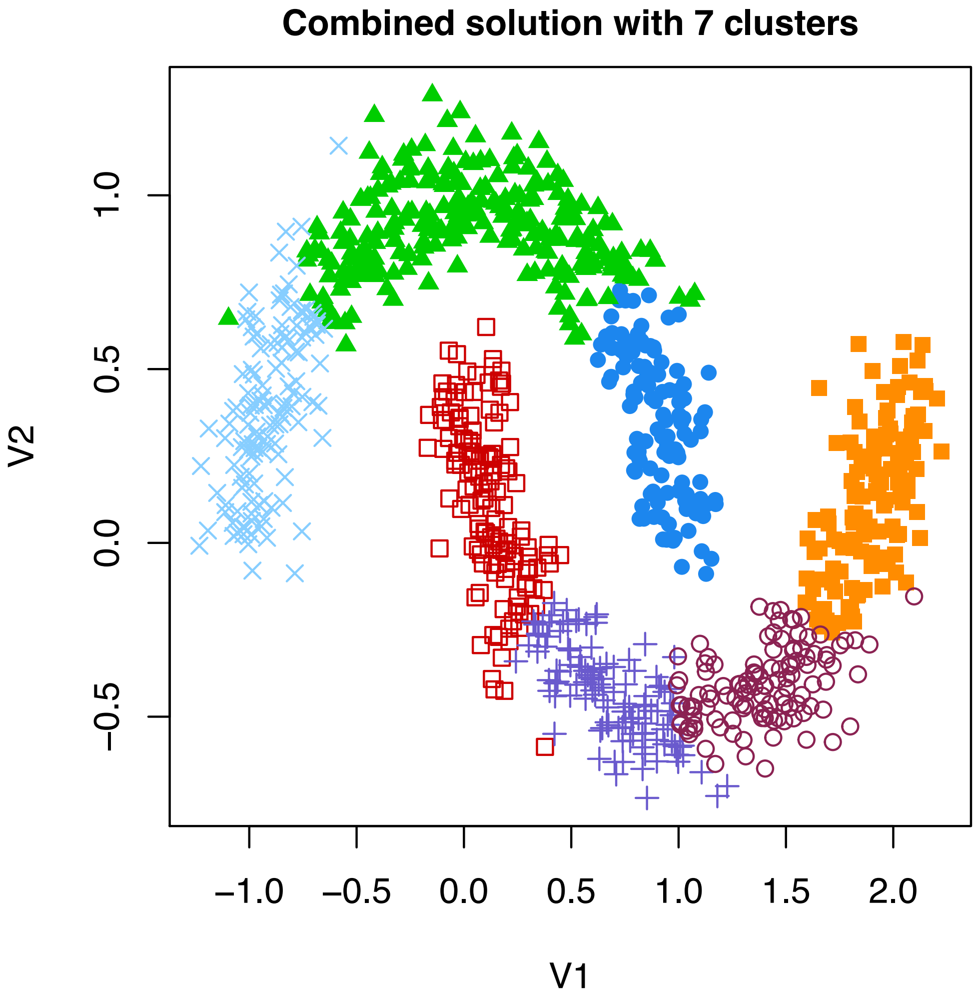
Note: we’re distinguishing “components” from “clusters” now.

Merging components to get clusters

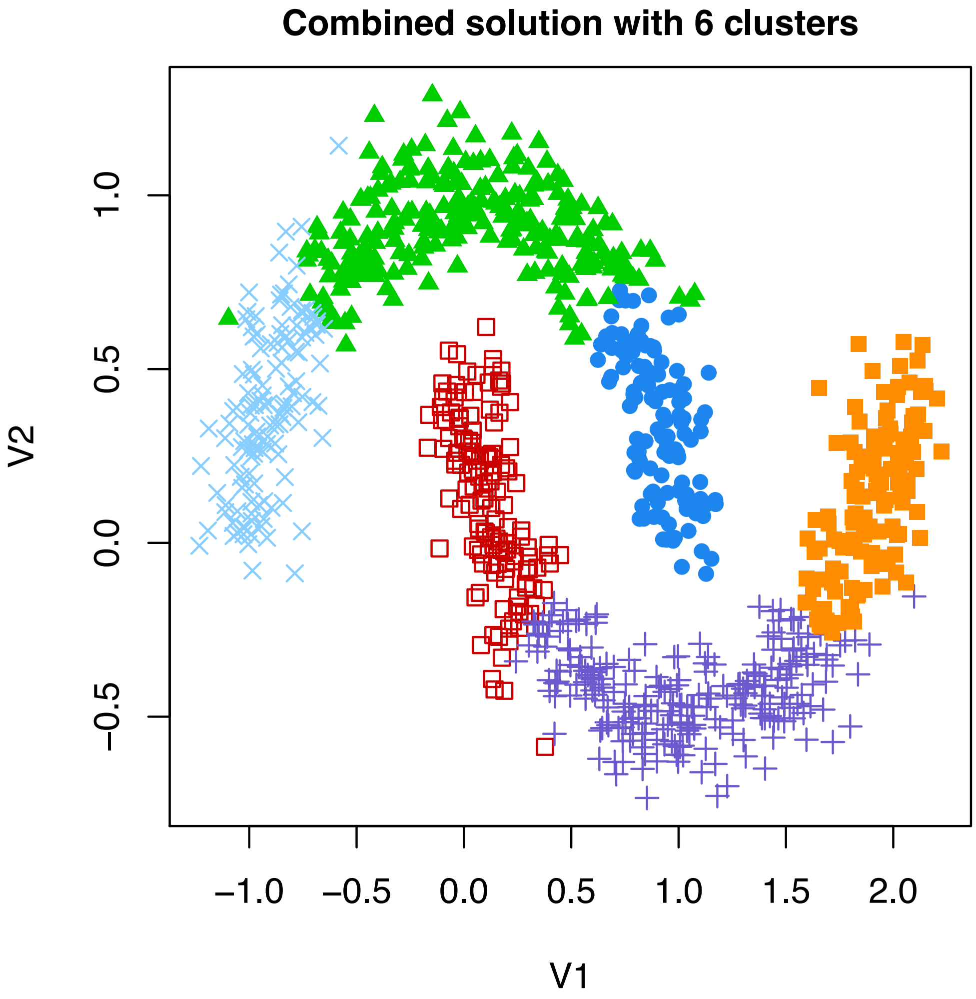
```
library(mclust)

output <- clustCombi(data = x)
plot(output)
```

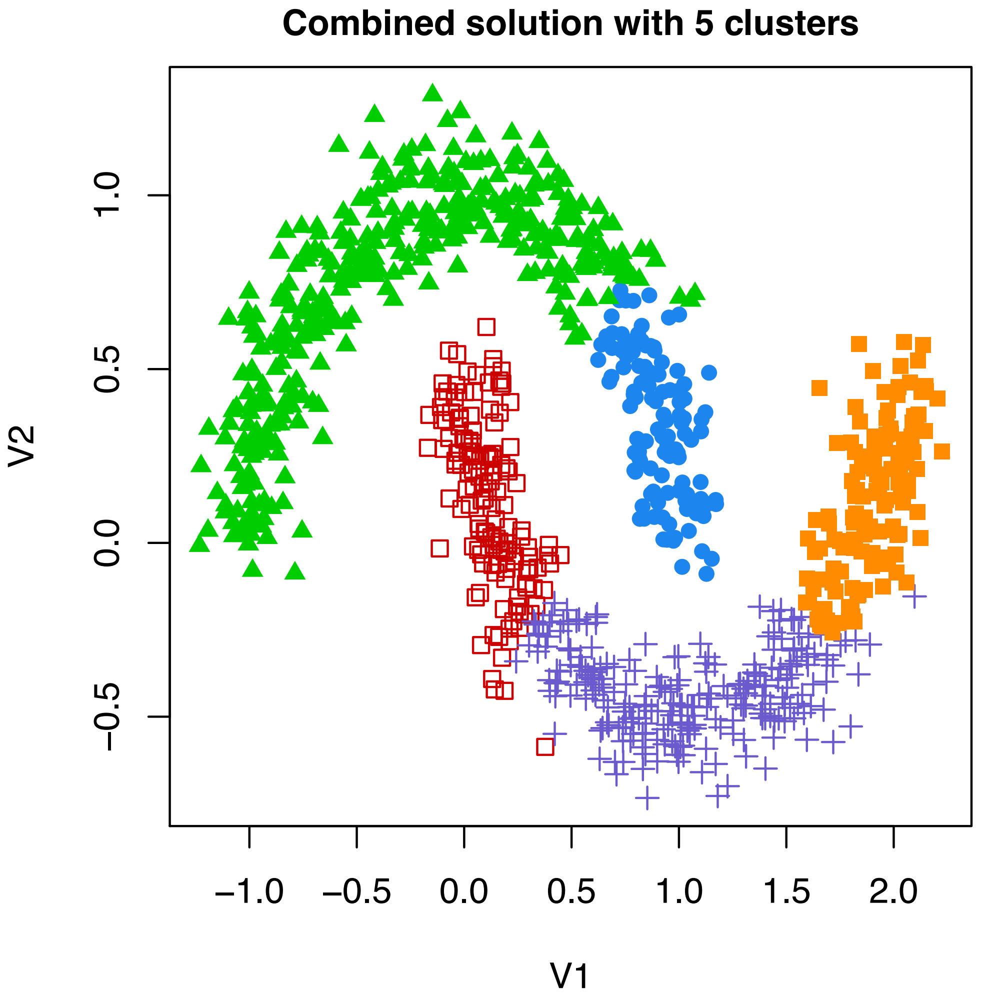


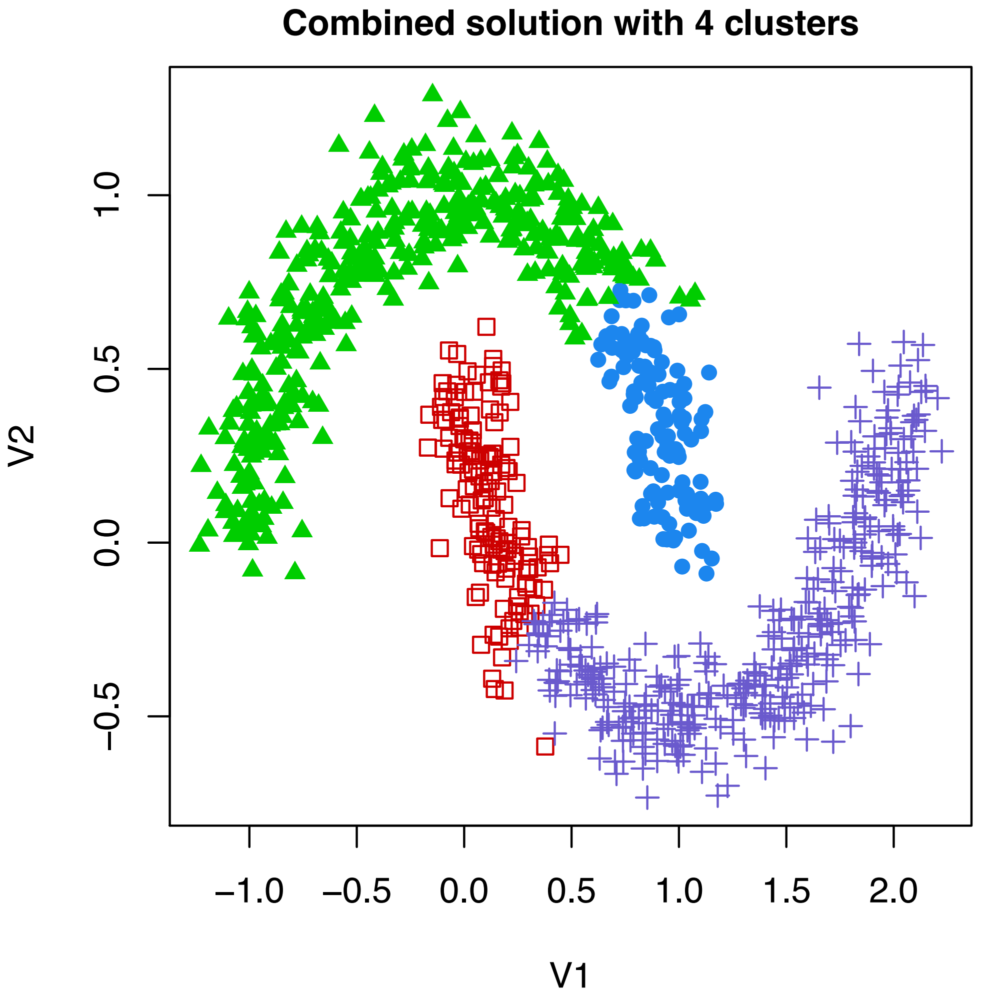

BIC solution (8 clusters)
V2
1.0
0.5
0.0
−0.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
V1

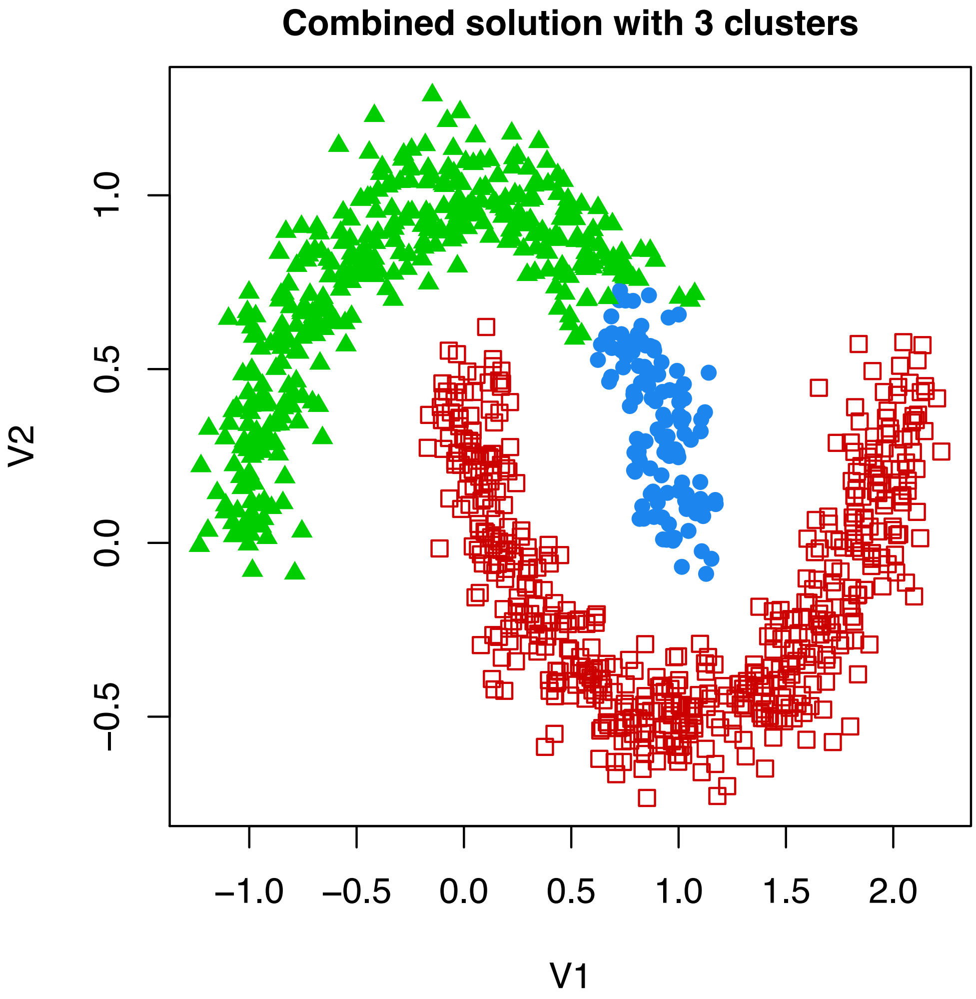


Combined solution with 7 clusters
V1
V2
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
−0.5
0.0
0.5
1.0

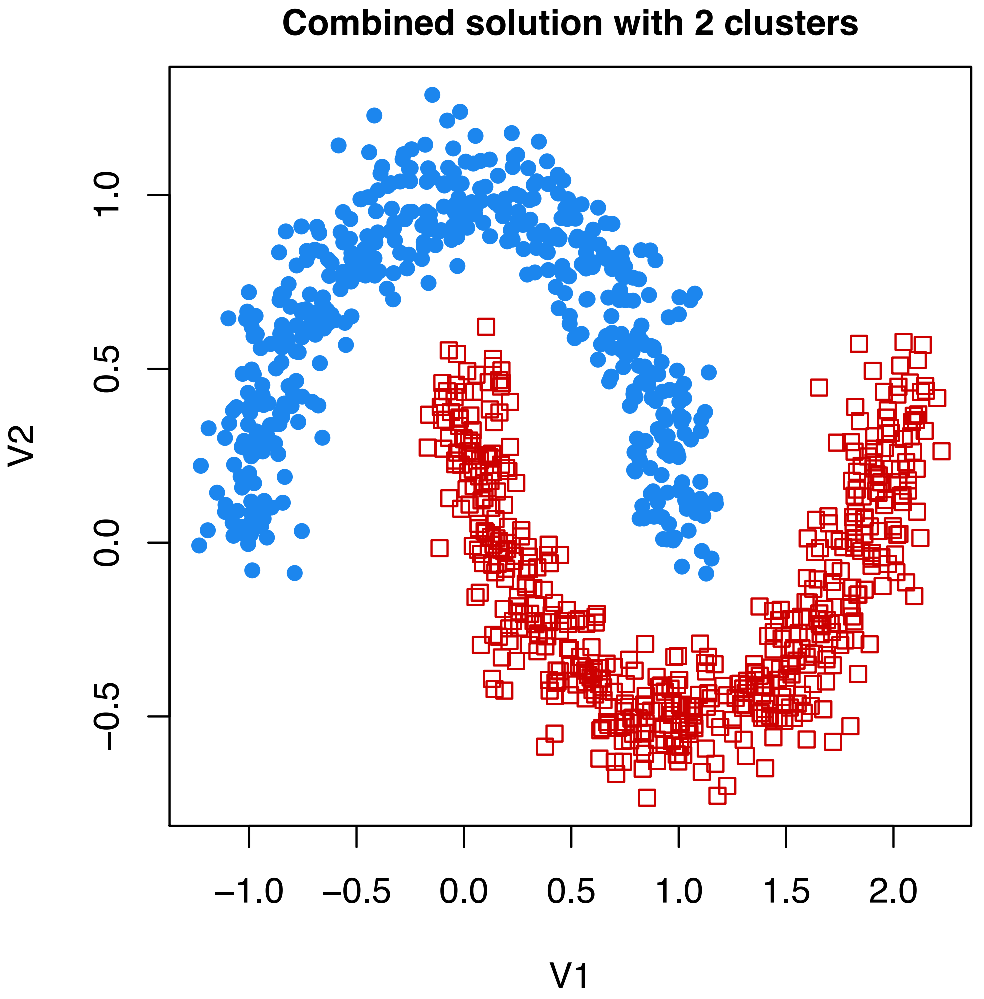

Combined solution with 6 clusters
V2
V1
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
−0.5
0.0
0.5
1.0

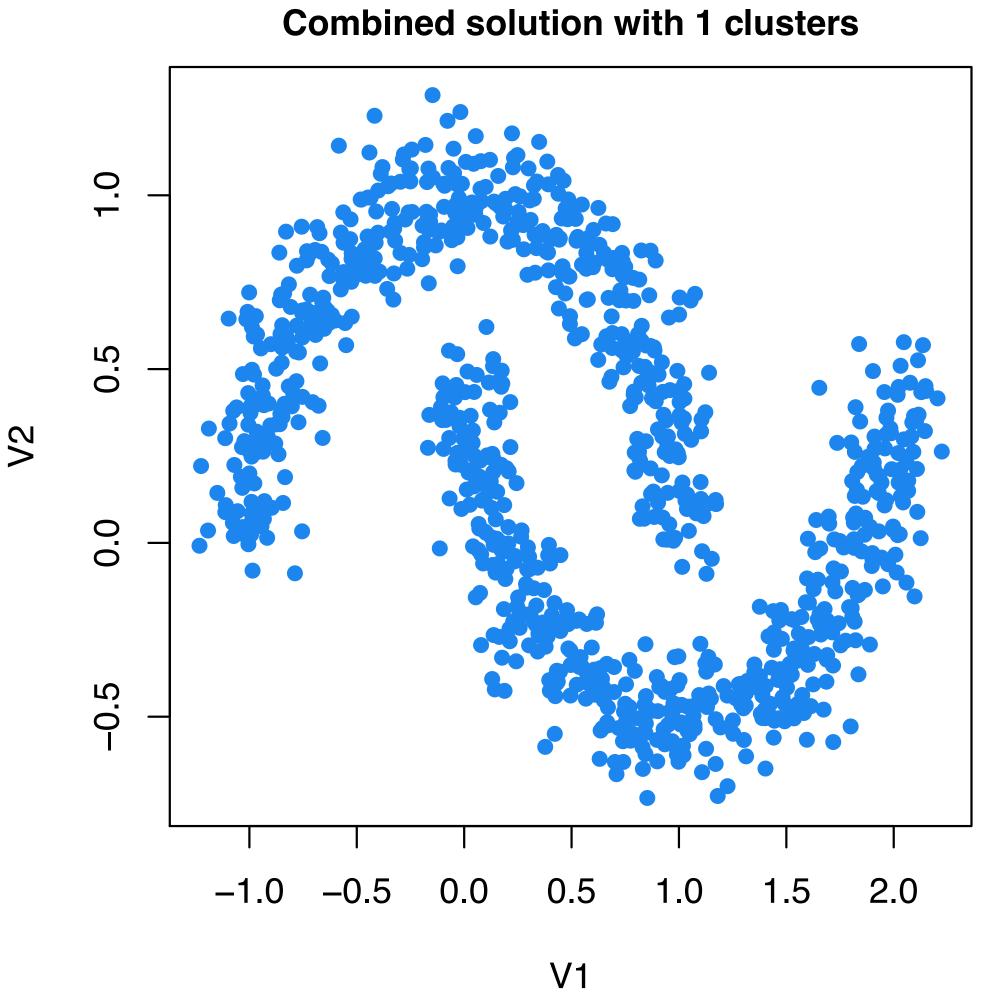

Combined solution with 5 clusters
V2
V1
1.0
0.5
0.0
−0.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0

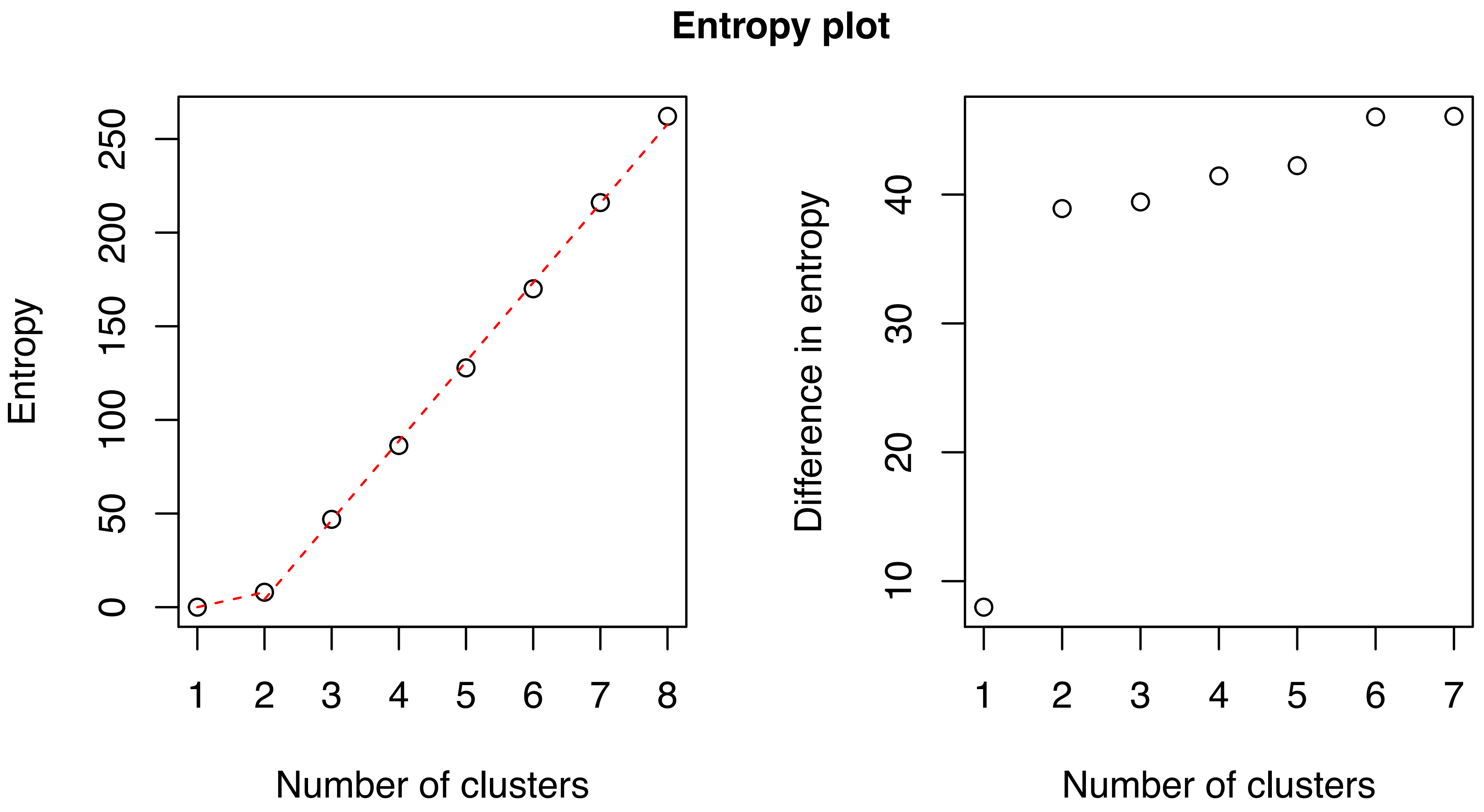

Combined solution with 4 clusters
V2
V1
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
−0.5
0.0
0.5
1.0



Combined solution with 3 clusters
V2
1.0
0.5
0.0
−0.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
V1


Combined solution with 2 clusters
V2
1.0
0.5
0.0
−0.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
V1


Combined solution with 1 clusters
V2
V1
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
−0.5
0.0
0.5
1.0



Entropy plot
Entropy
0
50
100
150
200
250
1
2
3
4
5
6
7
8
Number of clusters
Difference in entropy
10
20
30
40
1
2
3
4
5
6
7
Number of clusters

The `tidyLPA` package

```
library(tidyLPA)
library(dplyr)
```

```
pisaUSA15[1:100, ] %>%
    select(broad_interest, enjoyment, self_efficacy) %>%
    single_imputation() %>%
    estimate_profiles(3)
#> tidyLPA analysis using mclust:
#>
#>  Model Classes AIC    BIC    Entropy prob_min prob_max n_min n_max BLRT_p
#>  1     3       639.57 676.04 0.71    0.67     0.91     0.11  0.60  0.06
```