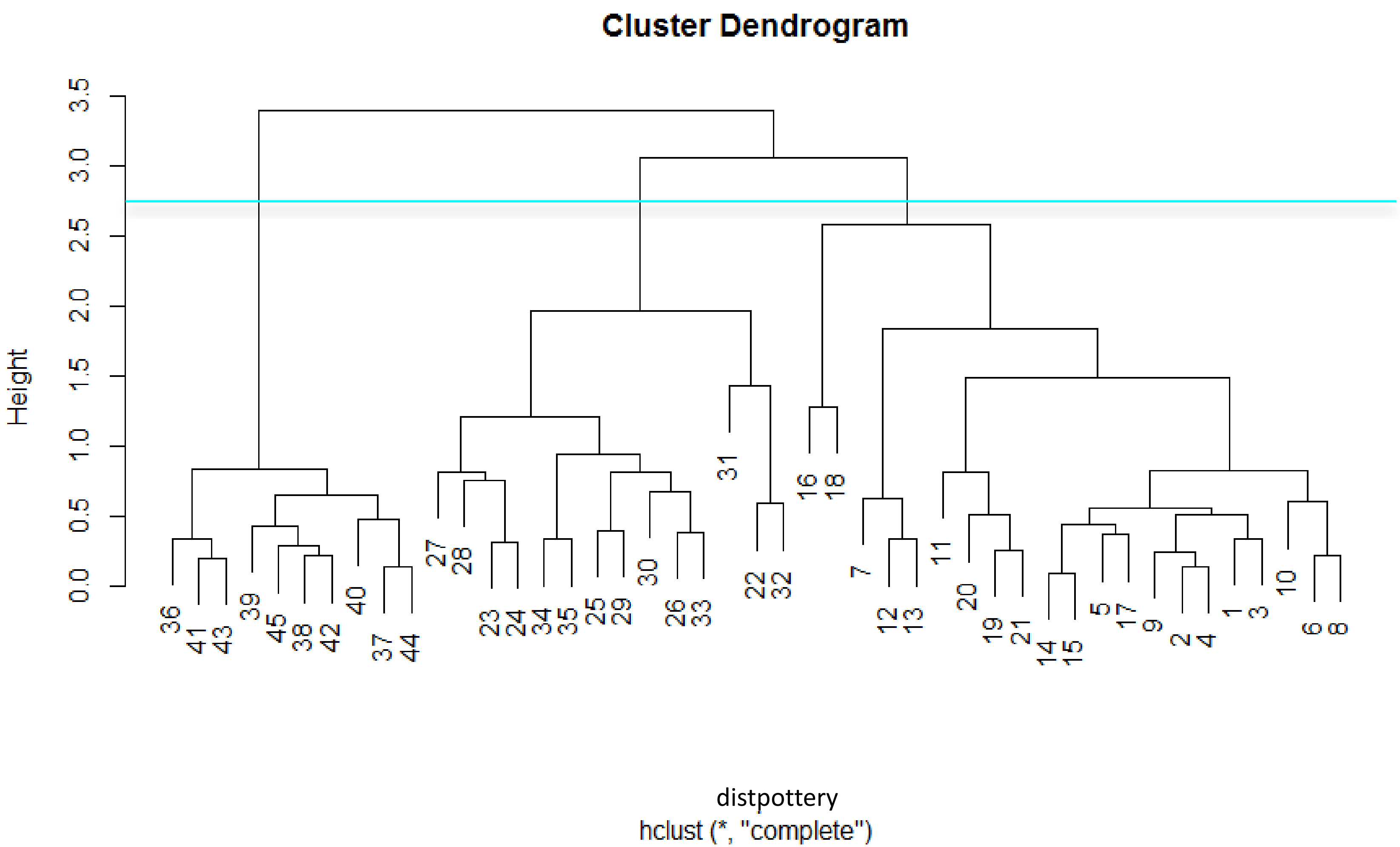
Cluster Analysis II

MENGQIAN LU

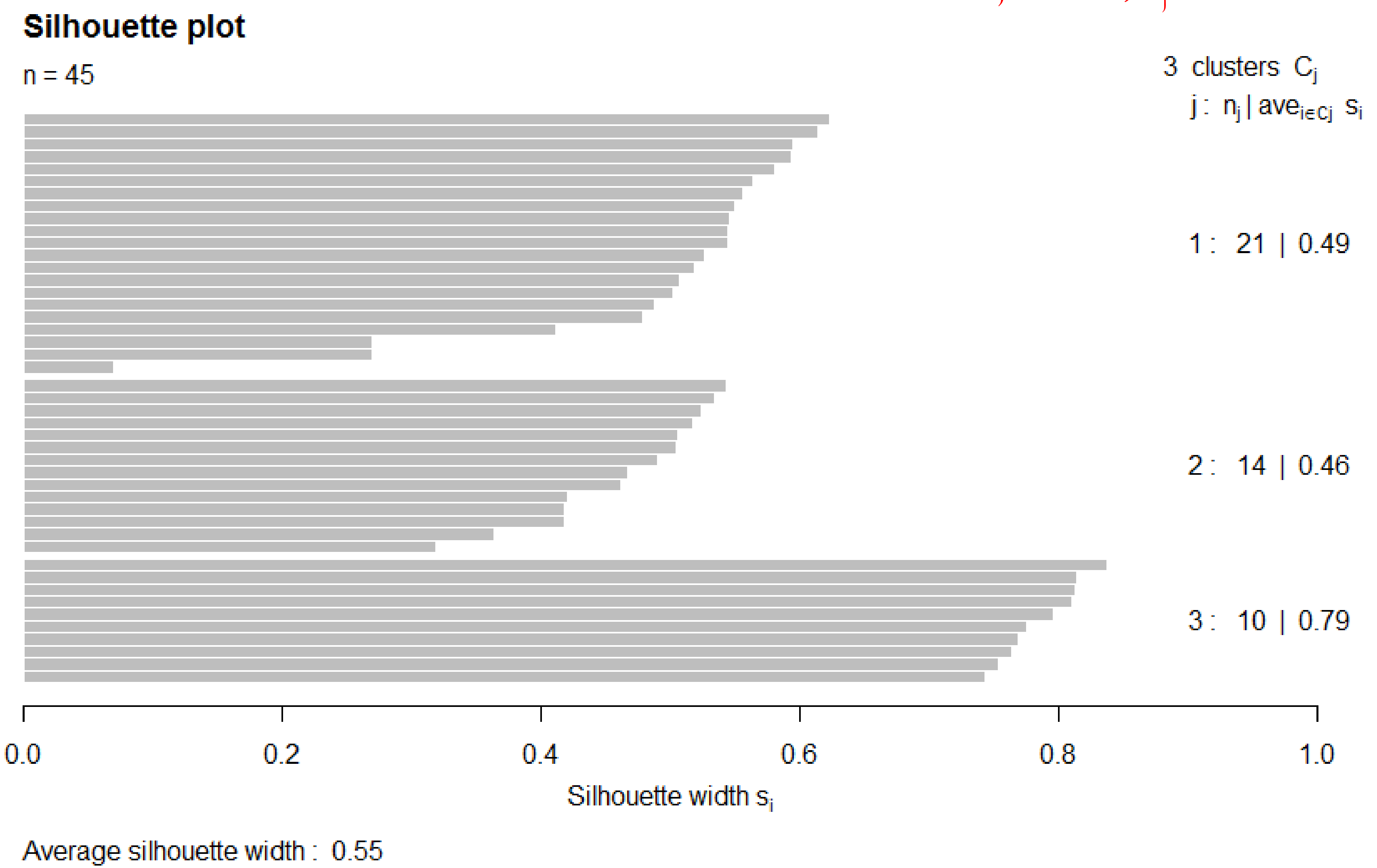
Assess your clusters with visualization `silhouette() in library(cluster)`

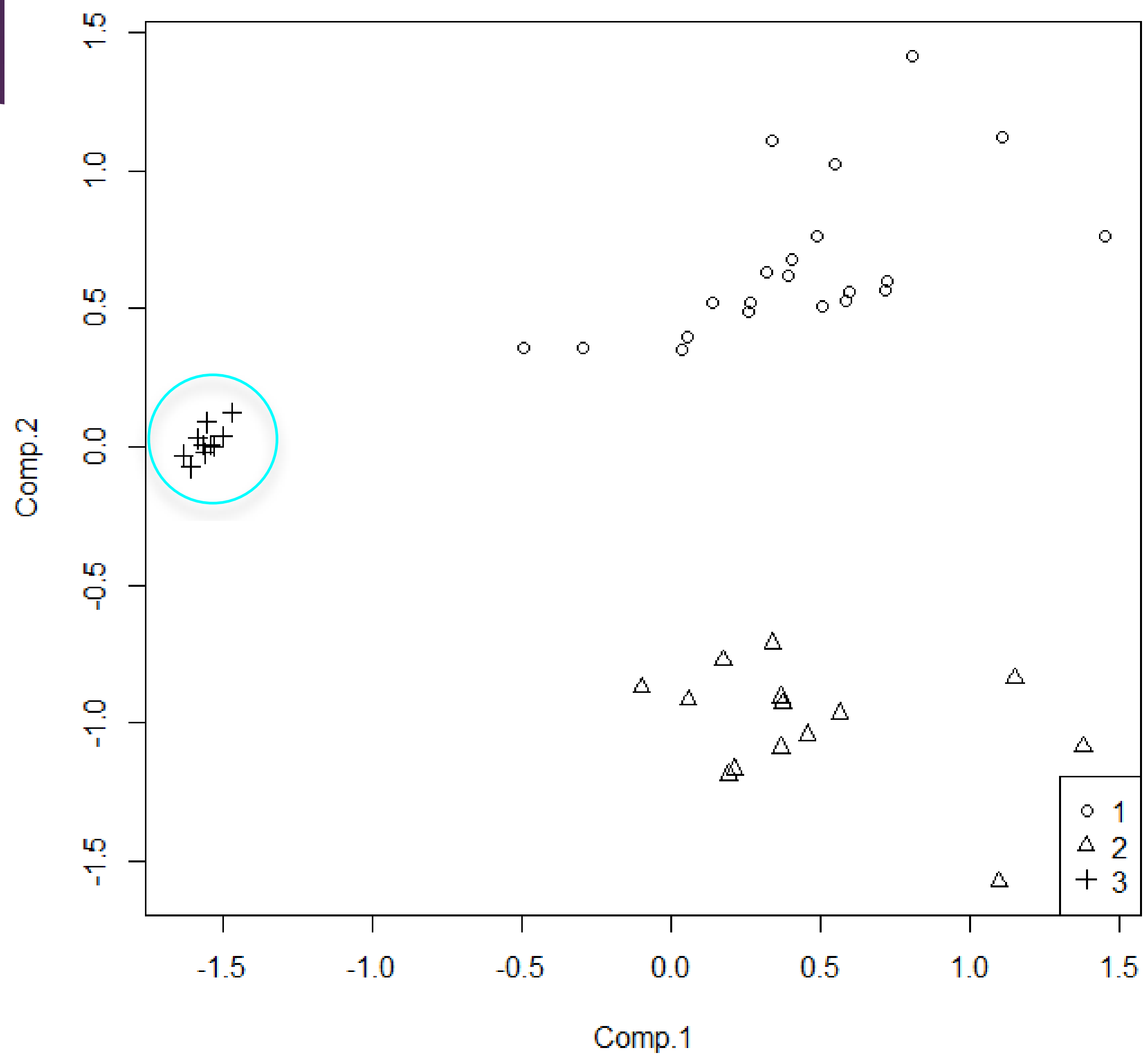
- data(pottery) in package HSAUR2, Chemical composition of Romano-British pottery – a data frame with 45 observations on the following 9 chemicals (Al2O3, Fe2O3, MgO, CaO, Na2O, K2O, TiO2, MnO, BaO, kiln – *site at which the pottery was found*).

1. Scale data – calculate Euclidean distance without *kiln* (*distpottery*)
2. Apply agglomerative clustering using complete linkage (*cc = hclust(distpottery, method = 'complete')*
3. Split into 3 groups (*grps = cutree(cc, k = 3))*
4. Plot the tree and use Silhouette plot to check the groups (*plot(silhouette(grps,distpottery))*
5. Visualize results in PC1 and PC2


Cluster Dendrogram
Height
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
36 41 43 39 45 38 42 40 37 44 27 28 23 24 34 35 25 29 30 26 33 31 22 32 16 18 7 12 13 11 20 19 21 14 15 5 17 9 2 4 1 3 10 6 8
distpottery
hclust (*, "complete")

j: cluster; n_j: no. of members





Comp.1
Comp.2
-1.5
-1.0
-0.5
0.0
0.5
1.0
1.5
1
2
3

agnes() in package cluster

- Can directly use the data set

```
> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1       5.1         3.5          1.4         0.2         setosa
2       4.9         3.0          1.4         0.2         setosa
3       4.7         3.2          1.3         0.2         setosa
4       4.6         3.1          1.5         0.2         setosa
5       5.0         3.6          1.4         0.2         setosa
6       5.4         3.9          1.7         0.4         setosa
```

- Two ways to calculate distance: dist() and daisy()

```
> d = dist(iris.use)
> z = agnes(d)
> plot(z) # use which.plots=  to specify which one only to display
> Hit <Return> to see next plot:
```

Example: ***Iris***

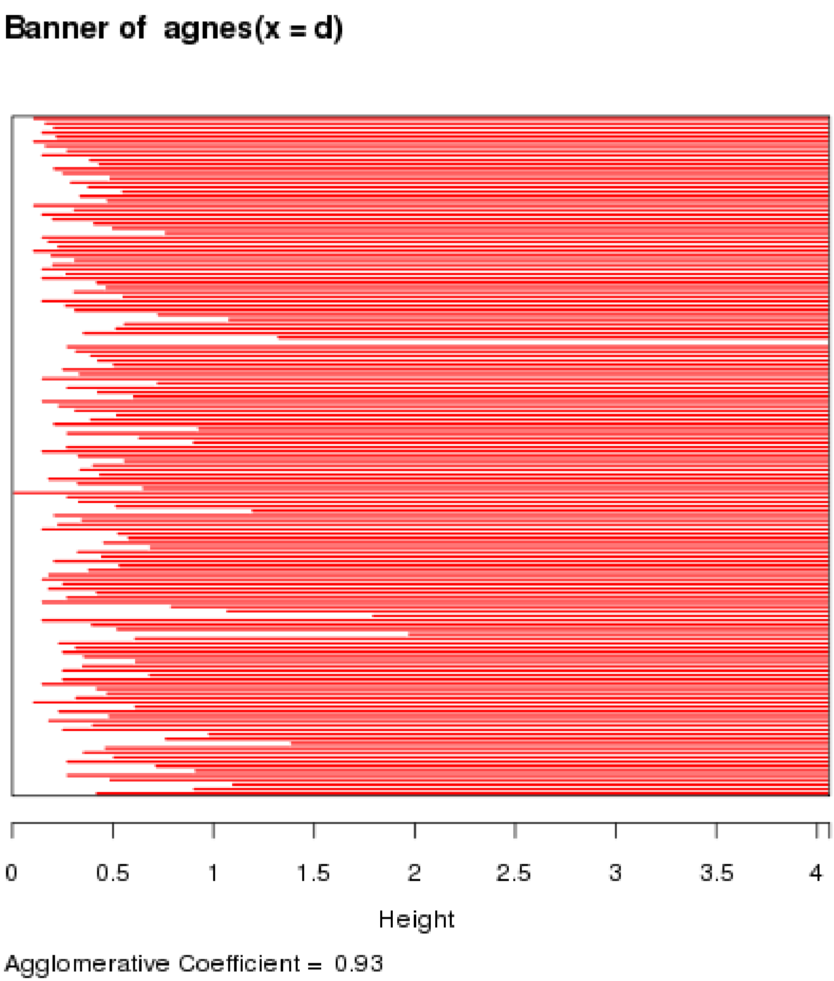
data(iris)

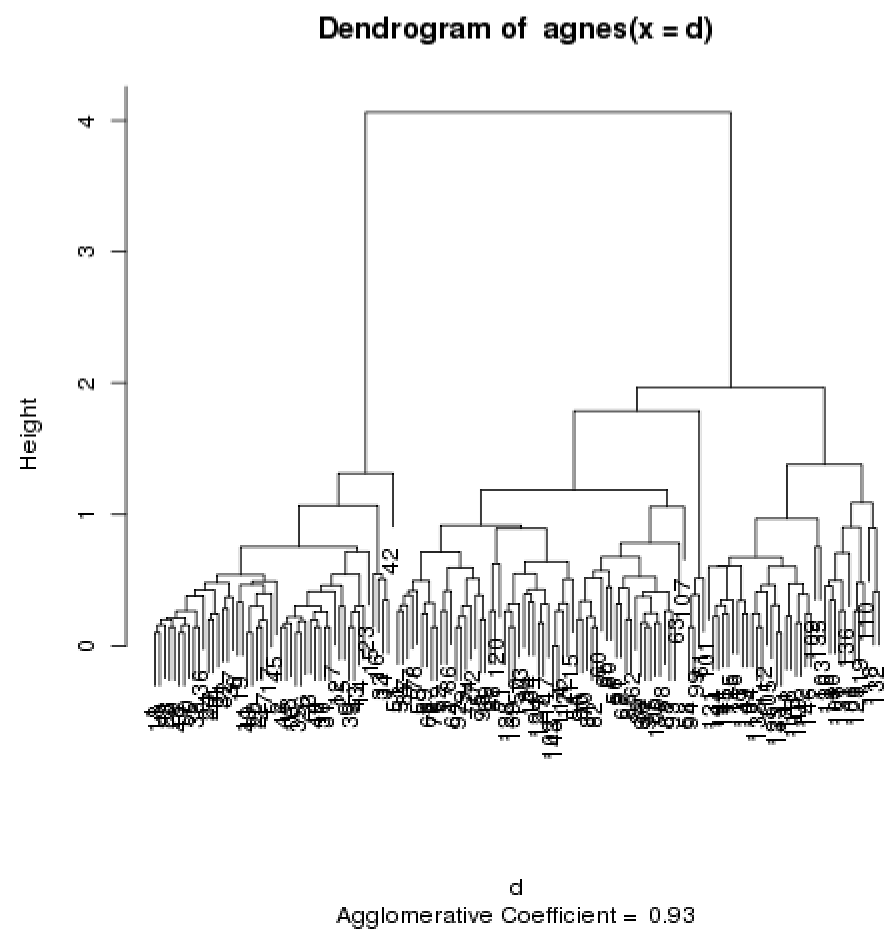
Iris Setosa

Iris Versicolor

Iris Virginica

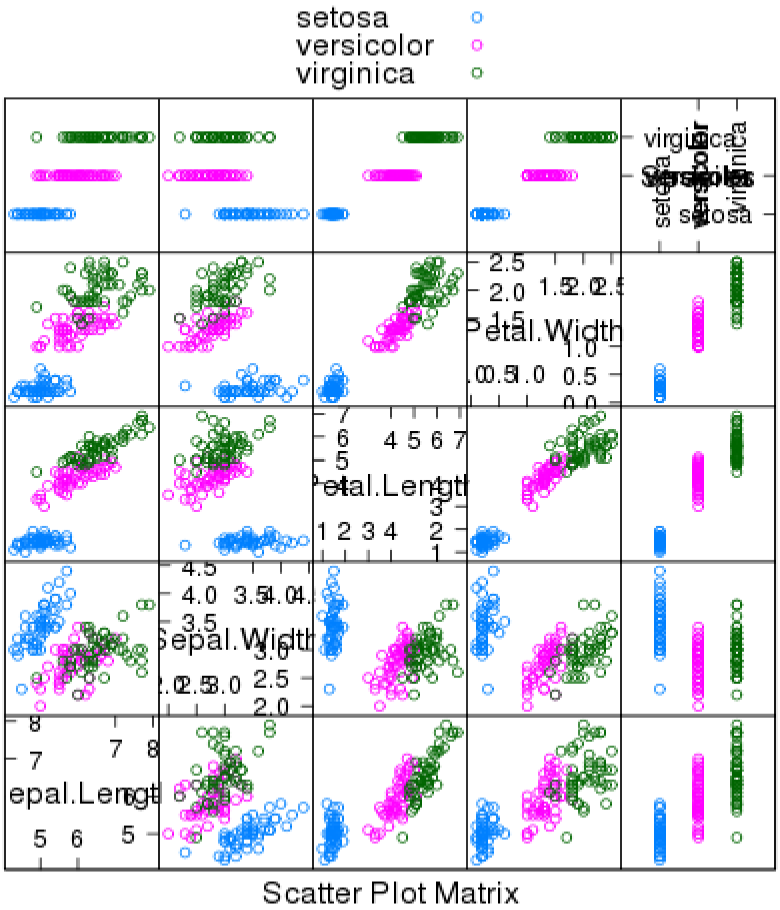
plot(z)
Plot(z, which.plot = 2) # only dendogram





agnes() in package cluster (cont'd)

```
> table(cutree(z,3),iris$Species)

   setosa versicolor virginica
  1     50          0         0
  2      0         50        14
  3      0          0        36
> splom(~iris, groups=iris$Species,
auto.key=TRUE)
```

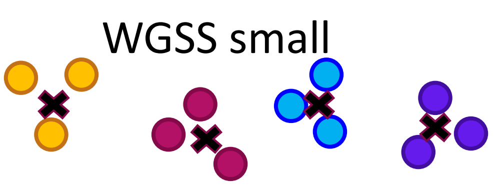


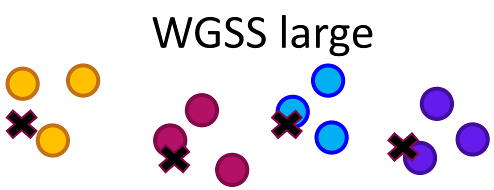
Types of Clustering of our focus

1. Hierarchical techniques
2. K-means clustering　　　**Pre-Specify number of classes**
   - Number of clusters K is fixed in advance
   - Find K cluster centers μi and assignments, so that within-groups Sum of Squares (WGSS) is minimal

组内差异

   - $WGSS = \sum_{all\ Cluster\ C} \sum_{Point\ i\ in\ Cluster\ C} (x_i - \mu_i)^2$





3. Model-based clustering

K-means

- Exact solution computationally infeasible

Table 6.2: Number of possible partitions depending on the sample size n and number of clusters k.

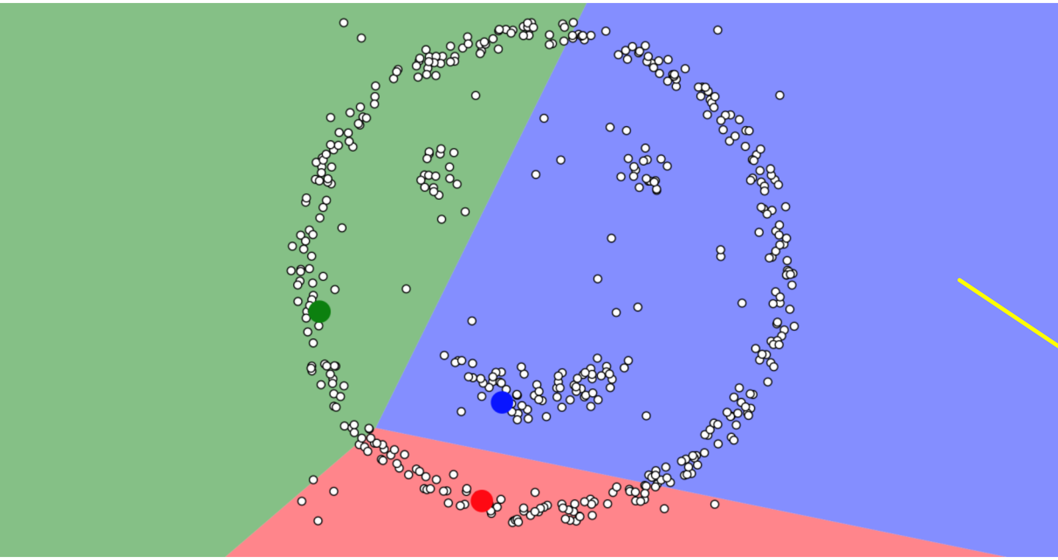
n	k	Number of possible partitions
15	3	$2,375,101$
20	4	$45,232,115,901$
25	8	$690,223,721,118,368,580$
100	5	10^{68}

Textbook: IAMA

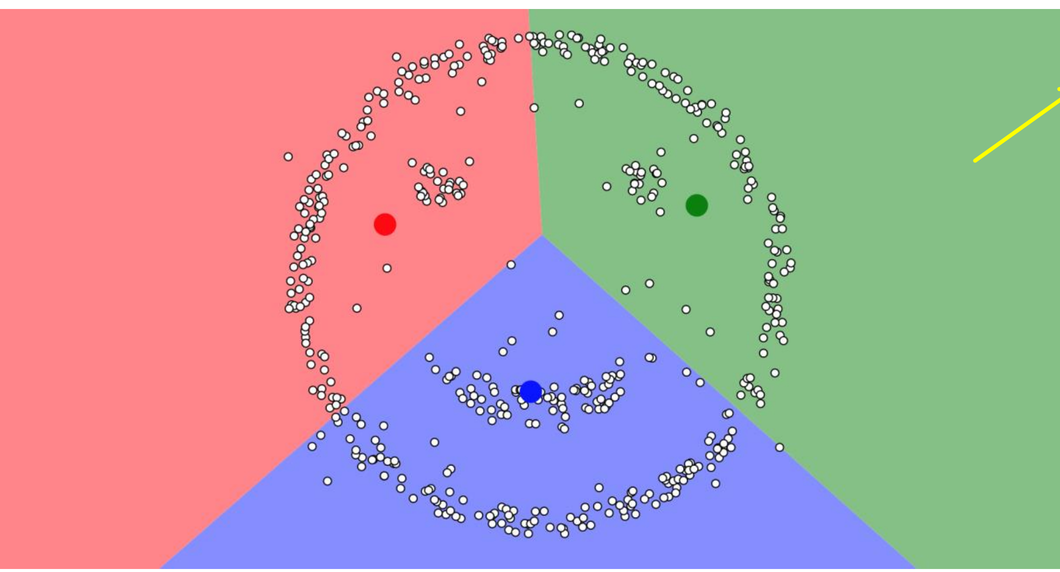
- Essential steps:

1. Final some initial partition, with k groups
2. Calculate the change in the clustering criterion produced by “moving” each member
3. Keep the change that leads to the greatest improvement
4. Repeat 2. and 3. until no useful improvement

Start #1



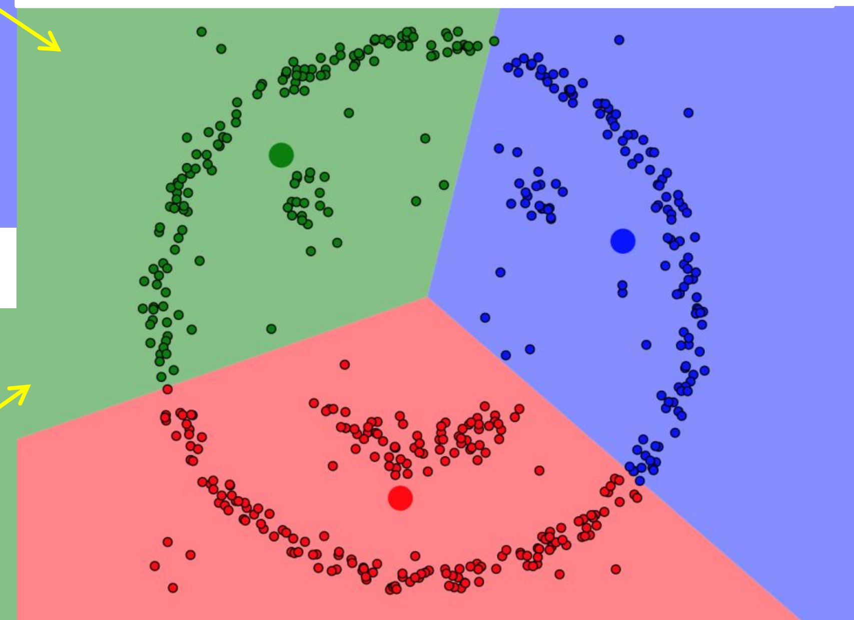
Start #2



http://www.naftaliharris.com/blog/visualizing-dbscan-clustering/

http://www.naftaliharris.com/blog/visualizing-k-means-clustering/

Clusters



http://stanford.edu/class/ee103/visualizations/kmeans/kmeans.html

kmeans() in R

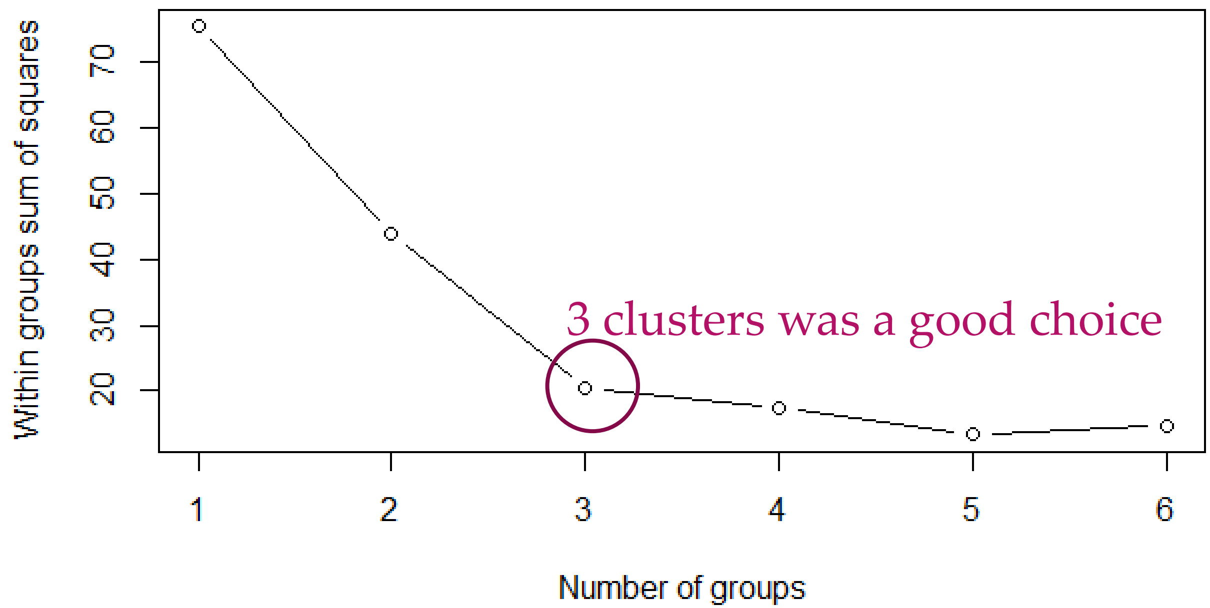
- data(pottery) in package HSAUR2, Chemical composition of Romano-British pottery – a data frame with 45 observations on the following 9 chemicals (Al2O3, Fe2O3, MgO, CaO, Na2O, K2O, TiO2, MnO, BaO, kiln – *site at which the pottery was found*).

```
> ckm = kmeans(pots, centers = 3, nstart = 10)
> grpsKM = ckm$cluster
> grpsKM
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 
 2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2 
20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 
 2  2  1  1  1  1  1  1  1  1  1  1  1  1  1  1  3  3  3 
39 40 41 42 43 44 45 
 3  3  3  3  3  3  3 
```

kmeans() in R (cont'd)

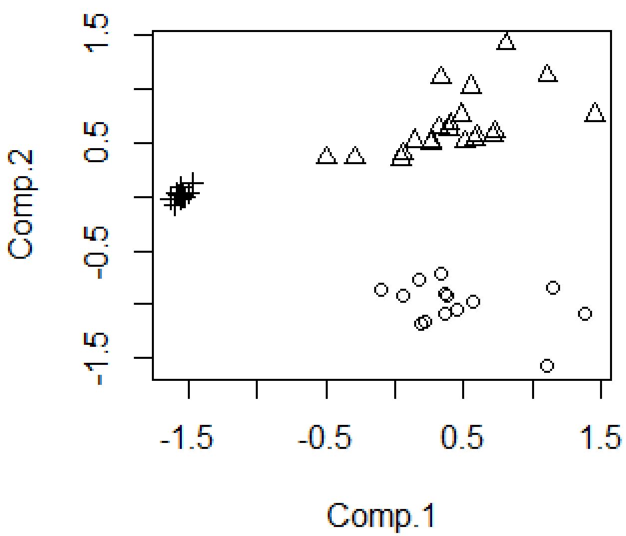
```
## find suitable number of centers
n = nrow(pottery); wss =  rep(0, 6);
for (i in 1:6) wss[i] =  sum(kmeans(pots, centers =
i)$withinss)
plot(1:6, wss, type = "b", xlab = "Number of groups", ylab =
"Within groups sum of squares")
```

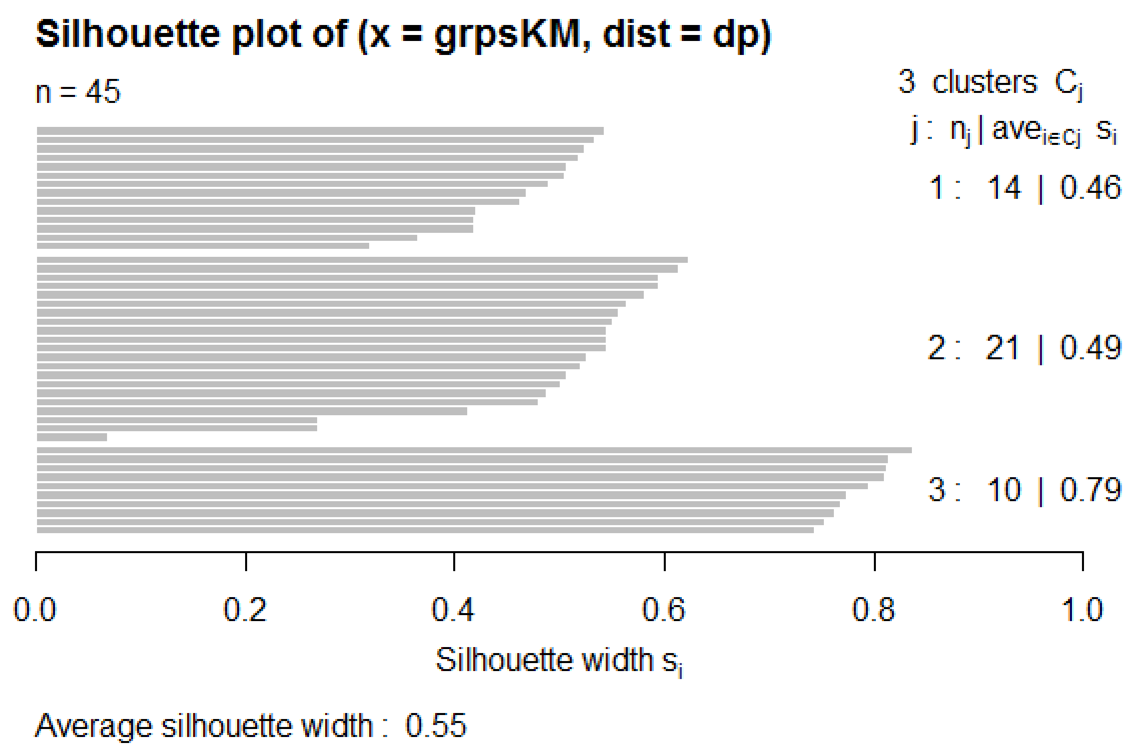
Result may vary, because of random starting configurations in kmeans



kmeans() in R (cont'd)

```
## Silhouette Plot
plot(silhouette(grpsKM, dp))
## visualize in PC 1 & 2
pr =  princomp(pots)$scores[,1:2]
plot(pr, pch = grpsKM)
```





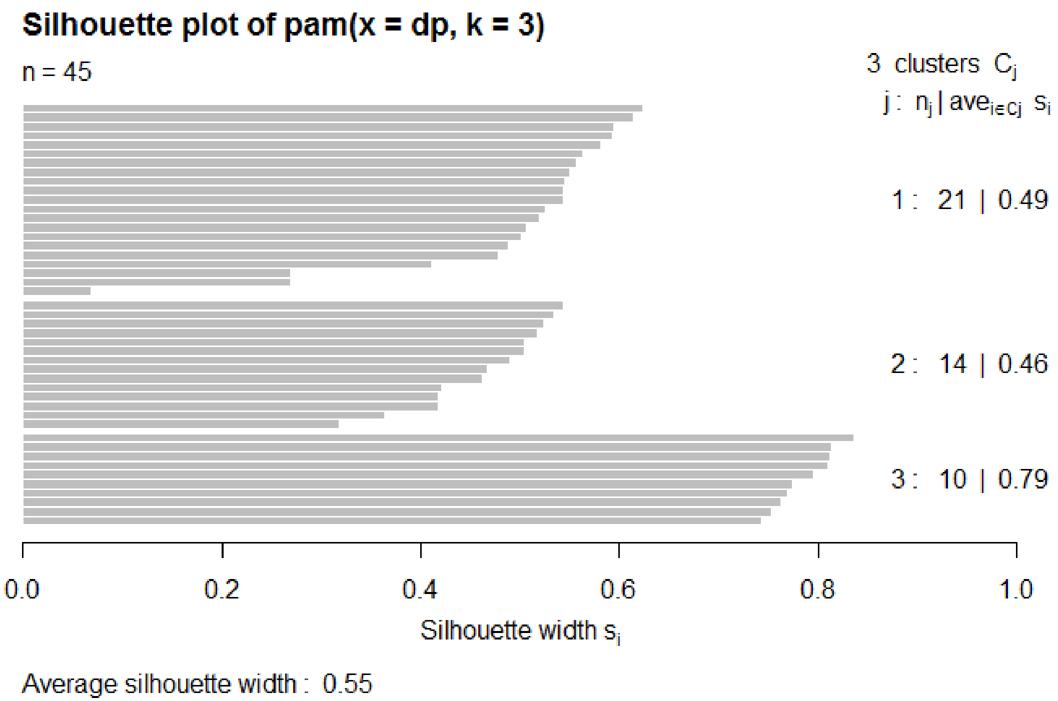
pam() in package cluster

- The k-Medoids Clustering
  - A cluster is represented with the object closest to the center of the cluster;
  - More robust in presence of outliers

```
> pamC = pam(x = distpottery, k = 3)
> pamC$medoids
[1] "2" "26" "38"
> pamC$clustering
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38
1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3
39 40 41 42 43 44 45
3 3 3 3 3 3 3
```

pam() in package cluster (cont'd)

```
> plot(pamC) ## Plots Shilouette directly
```



Types of Clustering of our focus

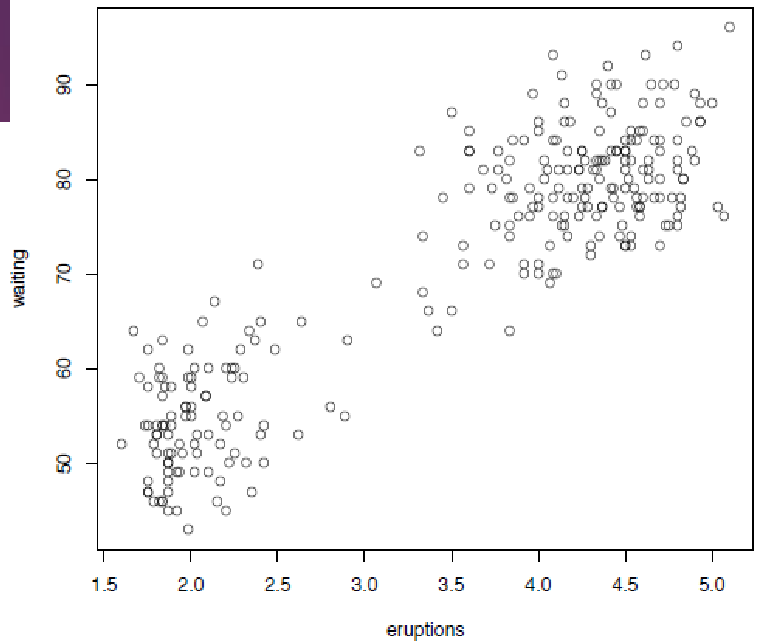
1. Hierarchical techniques
2. K-means clustering
3. Model-based clustering
   - Assume underlying statistical model of the population, from which we sampled our data;
   - Model assumes this population consists of a number of subpopulations (clusters)
   - Each subpopulation has variables with a different multivariate probability density function, together they result in a *finite mixture density* for the population as a whole
   - Cluster analysis → estimation of parameters of the assumed mixture
   - Determine number of clusters → model selection

Model-based clustering

- Read IAMA Ch 6.5
- R package *mclust*
  - *Model-based hierarchical clustering*
  - *Expectation-Maximization for Gaussian mixture models*
  - *Bayesian Information Criterion*

mclust

- data(faithful): Waiting time between eruptions and the duration of the eruption for the Old Faithful geyser in Yellowstone National Park, Wyoming, USA.



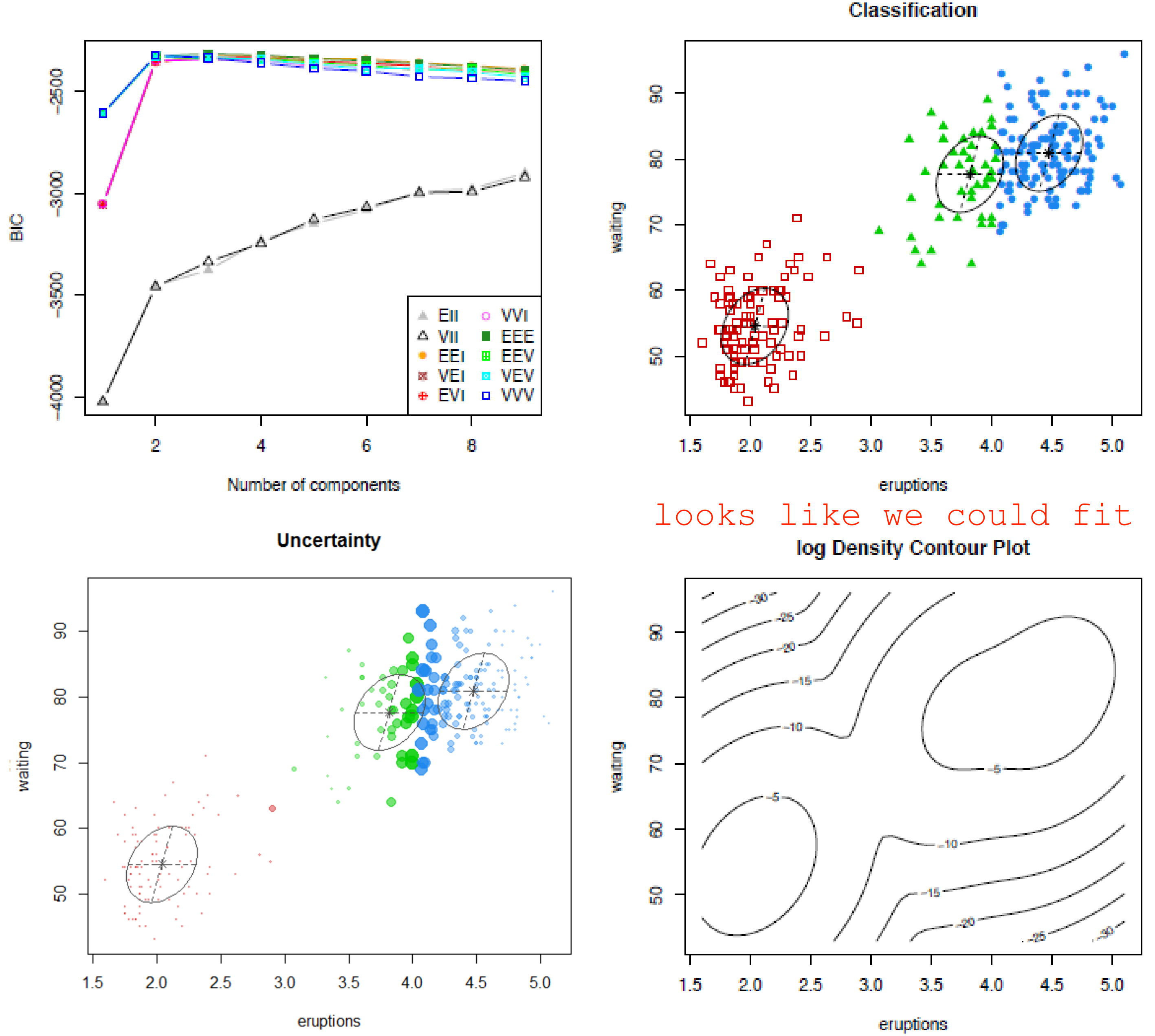

```
> faithfulMclust <- Mclust(faithful)
> summary(faithfulMclust)
----------------------------------------------------
Gaussian finite mixture model fitted by EM algorithm
----------------------------------------------------

Mclust EEE (elliposidal, equal volume, shape and orientation) model with 3 components:

 log.likelihood   n df     BIC
        -1126.4 272 11 -2314.4

Clustering table:
  1   2   3
130  97  45
```

mclust (cont'd)

```
> summary(faithfulMclust, parameters = TRUE)
----------------------------------------------------
Gaussian finite mixture model fitted by EM algorithm
----------------------------------------------------

Mclust EEE (elliposidal, equal volume, shape and orientation) model with 3 components:

 log.likelihood   n df     BIC
        -1126.4 272 11 -2314.4

Clustering table:
  1   2   3
130  97  45

Mixing probabilities:
      1       2       3
0.46190 0.35646 0.18164

Means:
             [,1]    [,2]    [,3]
eruptions  4.4761  2.0378  3.8199
waiting   80.8922 54.4935 77.6711
```

```
Variances:
[,,1]
          eruptions waiting
eruptions   0.07728  0.4765
waiting     0.47650 33.7485
[,,2]
          eruptions waiting
eruptions   0.07728  0.4765
waiting     0.47650 33.7485
[,,3]
          eruptions waiting
eruptions   0.07728  0.4765
waiting     0.47650 33.7485
```

```
> plot(faithfulMclust)
```



looks like we could fit an LDA

The covariance structures defining the models available in mclust

identifier	Model	HC	EM	Distribution	Volume	Shape	Orientation
E		•	•	(univariate)	equal		
V		•	•	(univariate)	variable		
EII	λI	•	•	Spherical	equal	equal	NA
VII	λ_k I	•	•	Spherical	variable	equal	NA
EEI	λA		•	Diagonal	equal	equal	coordinate axes
VEI	$\lambda_k A$		•	Diagonal	variable	equal	coordinate axes
EVI	λA_k		•	Diagonal	equal	variable	coordinate axes
VVI	$\lambda_k A_k$		•	Diagonal	variable	variable	coordinate axes
EEE	$\lambda D A D^T$	•	•	Ellipsoidal	equal	equal	equal
EEV	$\lambda D_k A D_k^T$		•	Ellipsoidal	equal	equal	variable
VEV	$\lambda_k D_k A D_k^T$		•	Ellipsoidal	variable	equal	variable
VVV	$\lambda_k D_k A_k D_k^T$	•	•	Ellipsoidal	variable	variable	variable

mclustBIC()

```
> faithfulBIC <- mclustBIC(faithful)
> faithfulSummary <- summary(faithfulBIC, data = faithful)
> faithfulSummary
 classification table:
   1   2   3
 130  97  45

 best BIC values:
   EEE,3   EEE,4   VVV,2
 -2314.4 -2320.2 -2322.2

 > faithfulBIC
  BIC:
        EII     VII     EEI     VEI     EVI     VVI     EEE     EEV     VEV     VVV
 1 -4024.7 -4024.7 -3055.8 -3055.8 -3055.8 -3055.8 -2607.6 -2607.6 -2607.6 -2607.6
 2 -3453.0 -3458.3 -2354.6 -2350.6 -2352.6 -2346.1 -2325.2 -2329.1 -2325.4 -2322.2
 3 -3377.7 -3336.5 -2323.0 -2332.7 -2332.2 -2342.4 -2314.4 -2339.0 -2329.4 -2333.9
 4 -3230.2 -3245.7 -2323.7 -2331.8 -2334.8 -2343.1 -2320.2 -2336.8 -2342.5 -2359.2
 5 -3149.4 -3128.2 -2337.7 -2348.3 -2355.9 -2374.3 -2337.0 -2356.2 -2366.2 -2385.3
 6 -3081.4 -3067.6 -2338.1 -2363.1 -2357.7 -2372.7 -2347.3 -2371.7 -2387.4 -2399.0
 7 -2990.3 -2998.5 -2356.5 -2370.1 -2375.9 -2393.1 -2361.2 -2393.0 -2384.2 -2426.5
 8 -2978.1 -2991.9 -2371.8      NA -2396.0      NA -2376.9 -2385.8 -2404.9 -2435.0
 9 -2899.8 -2921.0 -2388.6      NA -2399.1      NA -2393.7 -2418.3 -2428.4 -2447.3
```

mclustBIC() (cont'd)

```
> plot(faithfulBIC, G = 1:7, ylim = c(-2500,-2300),
       legendArgs = list(x = "bottomright", ncol = 5))
```