Primer of Biostatistics
The 4th Lecture

Hiroyoshi Iwata
aiwata@mail.ecc.u-tokyo.ac.jp

<Hierarchical cluster analysis>

For many objects, it may be useful to classify them into groups (clusters) based on their multidimensional characteristics. For example, if varieties and lines included in genetic resources can be grouped based on DNA polymorphism data, the variation of traits in genetic resources can be organized based on the group information.

As I mentioned in the last lecture, it is difficult to understand the variation in many features of many samples in data just by looking at the data. In principal component analysis, we tried to summarize variation in data by representing a large number of features with low-dimensional variables. Cluster analysis tries to summarize the variation in data by grouping a large number of samples into a small number of groups. In this lecture, we will first outline hierarchical cluster analysis that classifies a large number of samples hierarchically into groups.

In this lecture, explanations will be given using rice data (Zhao et al. 2011, Nature Communications 2: 467) as before. In this lecture, three data of variety/line data (RiceDiversityLine.csv), phenotype data (RiceDiversityPheno.csv) and marker genotype data (RiceDiversityGeno.csv) are used. All of them are downloaded from the Rice Diversity web page http://www.ricediversity.org/. As described earlier, marker genotype data is imputed for missing data using the software fastPHASE (Scheet and Stephens 2006, Am J Hum Genet 78: 629).

First, let's read three datasets and combine them as we did last time.

```
> line <- read.csv("RiceDiversityLine.csv")
> pheno <- read.csv("RiceDiversityPheno.csv")
> geno <- read.csv("RiceDiversityGeno.csv")
> line.pheno <- merge(line, pheno, by.x = "NSFTV.ID", by.y = "NSFTVID")
> alldata <- merge(line.pheno, geno, by.x = "NSFTV.ID", by.y = "NSFTVID")
> rownames(alldata) <- alldata$NSFTV.ID
```

First, let's classify 374 varieties / lines into clusters based on variations in DNA markers (1,311 SNPs). First, prepare the data for that.

```
> data.mk <- alldata[, 50:ncol(alldata)]
        # marker data start from 50th column
> subpop <- alldata$Sub.population
        #  extract subpopulation data
> dim(data.mk)            #  size of data
[1]  374 1311             #  374 rows x 1311 columns
```

There are various methods for cluster analysis, but here we will perform cluster analysis with one method.

First, based on the DNA marker data, distances amonog varieties and lines are calculated.

```
> d <- dist(data.mk)       #  calculate Euclid distance among 374 var/lines
> head(d)                  #  the result is in the distance matrix format
[1] 54.47141 53.08033 44.70547 52.82571 45.40700 44.36904
> as.matrix(d)[1:6, 1:6]            #  convert it to the matrix format
         1        3        4        5        6        7
1  0.00000 54.47141 53.08033 44.70547 52.82571 45.40700
3 54.47141  0.00000 37.53194 46.79940 37.68502 49.82169
4 53.08033 37.53194  0.00000 44.38481 17.58133 46.49073
5 44.70547 46.79940 44.38481  0.00000 43.85254 42.87989
6 52.82571 37.68502 17.58133 43.85254  0.00000 46.69070
7 45.40700 49.82169 46.49073 42.87989 46.69070  0.00000
```

Note that the value returned by the function dist is not in the form of a matrix, but in the form of a distance matrix. Therefore, if you want to display the distances among the first six varieties in a 6x6 matrix, you need to convert the distance matrix-specific format to the matrix format with the function as.matrix as described above.

Let's do cluster analysis.

```
> tre <- hclust(d)        #  clustering with the function hclust
> tre                     #  show the result

Call:
hclust(d = d)

Cluster method   : complete
Distance         : euclidean
Number of objects: 374
```

After the "Call" the executed command was displayed as it is in regression analysis. Also, "Cluster method" shows the method of cluster analysis (definition of distance between clusters), and "Distance" shows calculation method of distance. Also, "Number of objects" is the number of classified objects (here, varieties and lines).

Let's display the result of cluster analysis as a dendrogram.

```
> plot(tre)               #  単に hclust の結果を関数 plot に入力するだけ
```

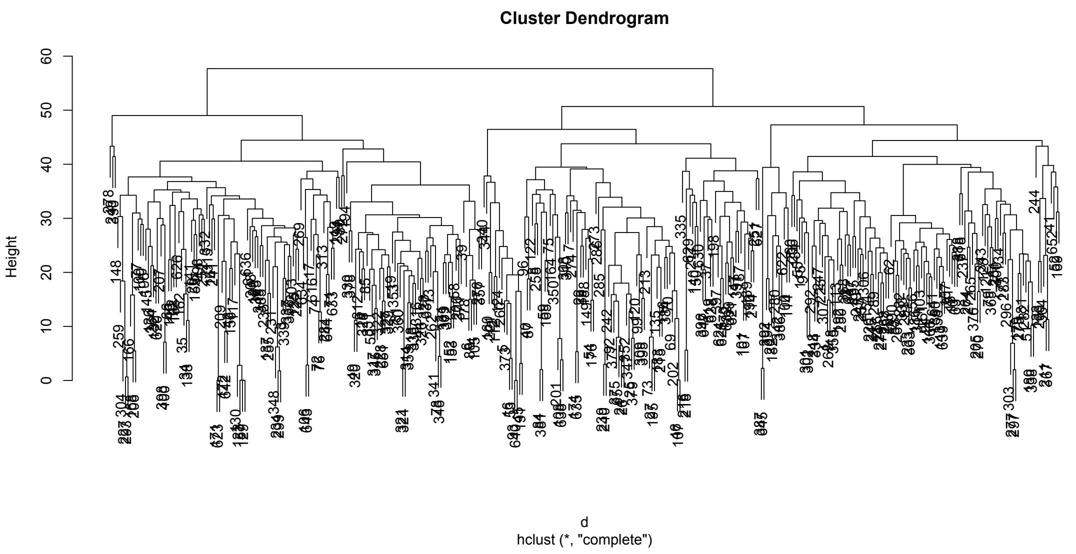

Figure 1. Dendrogram of 374 varieties / lines obtained based on marker genotype data

Figure 1 shows the result obtained with the function hclust in the form of a dendrogram. Using the package ape, you can draw a dendrogram in various expression styles. To do so, you first need to convert the result obtained with the function hclust into a class called phylo, which is defined in the package ape.

```
> require(ape)                          #  package ape
> phy <- as.phylo(tre)                  #  convert hclust to phylo
```

Let's plot the result converted to the phylo class.

```
> plot(phy)                             #  plot the object converted to phylo
```

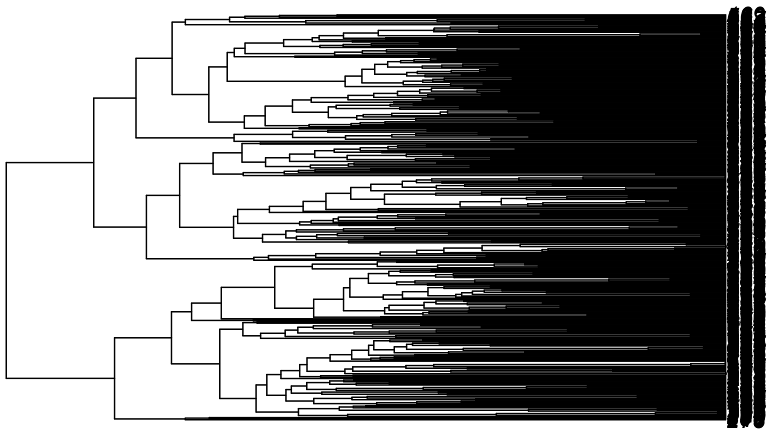

Figure 2. A dendrogram converted to the phylo class of the package ape

Figure 2 is very difficult to see due to the large number of varieties and lines. Let's make it a little easier to see, by making it possible to confirm the relationship between the genetic background of each variety / line (the belonging subpopulation) and the position in the tree diagram with the color of the branches.

```
> phy$edge                  #  the information of connection of blanches
 (omitted)
> subpop[phy$edge[,2]]      #  the second column of phy$edge is
                            #  ID of downstream of each blanch.
                            #  If the blanch is not terminal, the value is <NA>
 (omotted)
> col <- as.numeric(subpop[phy$edge[,2]])
                            #  Use the subpop information as color ID
> edge.col <- ifelse(is.na(col), "gray", col)
                            #  Convert the color of <NA> as “gray”
> plot(phy, edge.color = edge.col, show.tip.label = F)
                            #  Set color of blanches with edge.color option
         #  The terminal lavels are omitted with show.tip.label = F
```

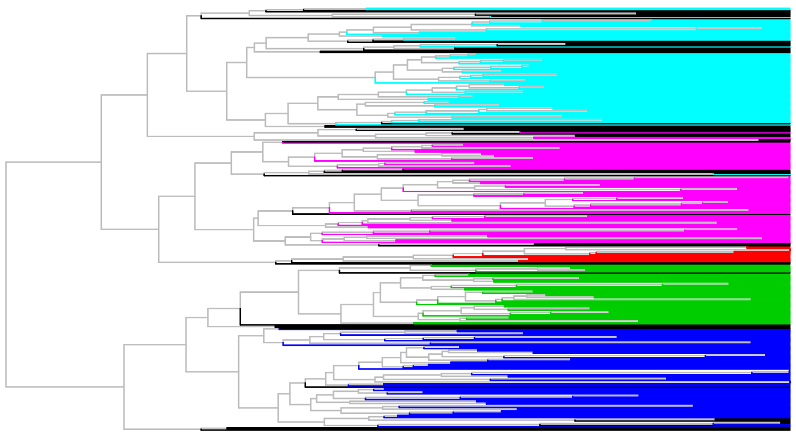

Figure 3. A dendrogram colored for each population group of varieties and lines

As seen in Figure 3, it is possible to confirm the tendency that varieties and lines included in the same subpopulation are included in the same cluster, and it is understood that differences in genetic background of varieties and lines are well reflected in the results of cluster analysis.

The phylo class of package ape can draw dendrograms in various ways of expression. Draw different types of dendrograms.

```
> pdf("fig4.pdf", width = 10, height = 10)
        #  Output in a pdf file
> op <- par(mfrow = c(2, 2), mar = rep(0, 4))
                #  arrange graphs 2x2, mar is the option for margin
> plot(phy, edge.color = edge.col, type = "phylogram", show.tip.label = F)
                                  #  default style
> plot(phy, edge.color = edge.col, type = "cladogram", show.tip.label = F)
> plot(phy, edge.color = edge.col, type = "fan", show.tip.label = F)
> plot(phy, edge.color = edge.col, type = "unrooted", show.tip.label = F)
> par(op)                         #  reset graphic parameters
> dev.off()                       #  close the pdf file
```

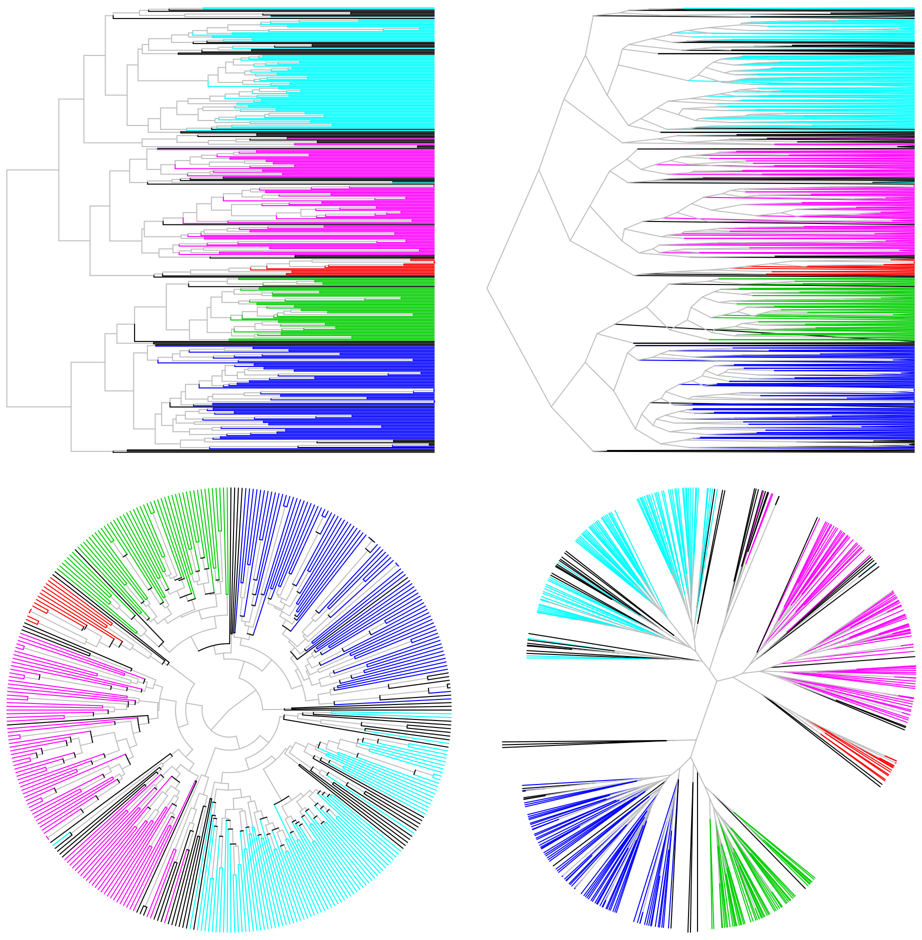

Figure 4. Various styles of dendrograms drawn using package ape

Figure 4 depicts the results of the same cluster analysis in a different style. Impressions and ease of understanding are different when the style is different. If you want to understand the genetic relationship of varieties and lines globally, it is likely that the fourth "unrooted" type tree chart is the most suitable.

The procedure of drawing a cluster analysis result using package ape is somewhat troublesome because it requires conversion to a phylo class on the way. So, let's define a series of tasks as a self-made function to simplify the illustration of the results of cluster analysis.

```
> myplot <- function(tre, subpop, type = "unrooted", ...) {
                  #   Define a self-made function using the function function
                  # Specify arguments of self-made function first.
                  # Here, tre, subpop, type
                  # The default value ("unrooted") has been set for type
                  # Describe the processing to be executed
                  #by the function in the part enclosed by {}
       phy <- as.phylo(tre)
       col <- as.numeric(subpop[phy$edge[,2]])
       edge.col <- ifelse(is.na(col), "gray", col)
       plot(phy,  edge.color  =  edge.col,  type  =  type,  show.tip.label  =
F, ...)
}
```

Let's draw a dendrogram using the self-made function myplot.

```
> d <- dist(data.mk)              #  calculate distances
> tre <- hclust(d)                #  cluster analysis
> myplot(tre, subpop)             #  use the self-made function myplot
> myplot(tre, subpop, type = "cladogram")
                                  #  use the option type to change a style
```

<Definition of distance>
Cluster analysis calculates distances between samples and clusters, and performs clustering based on the calculated distances. Therefore, different definitions of distance will give different results. Here, we will explain the definition of the distance between samples and between clusters.

First of all, about the distance between samples. There are various definitions to calculate the distance between samples. First, let's draw a dendrogram based on different defined distances.

```
> pdf("fig5.pdf", width = 10, height = 10)
> op <- par(mfrow = c(2, 2), , mar = rep(0, 4))
> d <- dist(data.mk, method = "euclidean")  #  Euclid distance (default)
> myplot(hclust(d), subpop) #  draw a dendrogram with the self-made function
> d <- dist(data.mk, method = "manhattan")  #  Manhattan distance
> myplot(hclust(d), subpop)
> d <- dist(data.mk, method = "minkowski", p = 1.5)   #  Minkowski distance
> myplot(hclust(d), subpop)
> d <- as.dist(1 - cor(t(data.mk)))
                #  Distance based on correlation
                #  The matrix should be converted to a dist class
> myplot(hclust(d), subpop)
> par(op)
> dev.off()                         #  close the pdf file
```

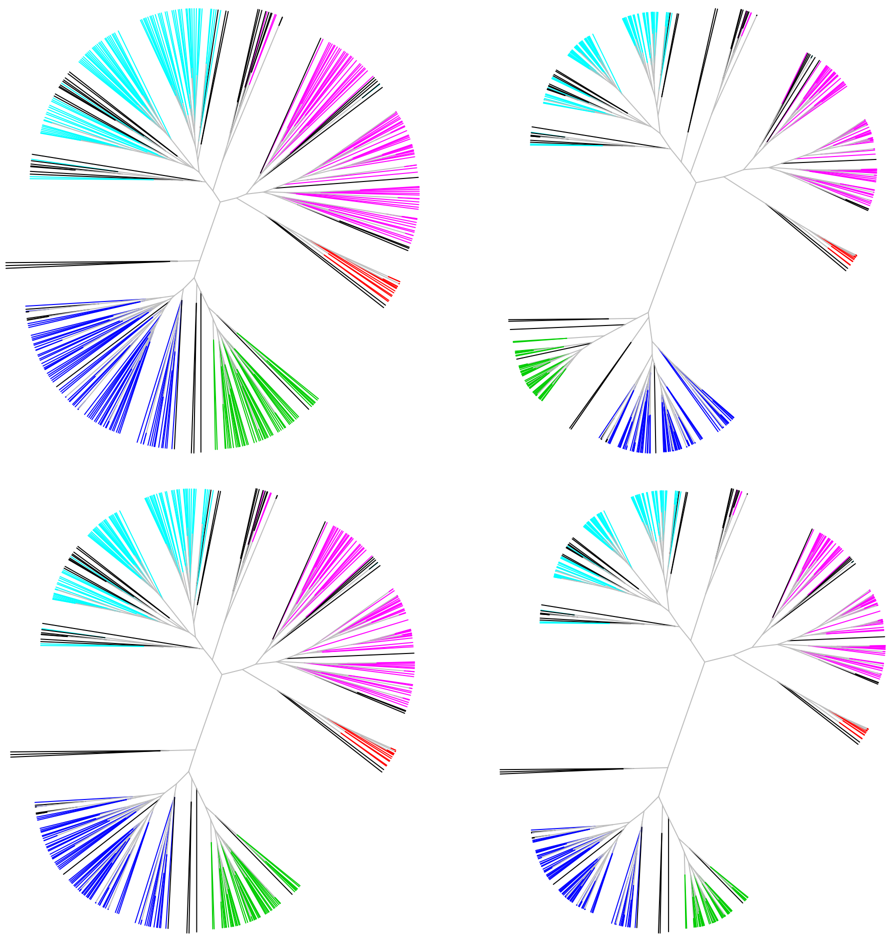
Figure 5. A dendrogram calculated based on different definitions of distances between samples

In this data, the topology of the dendrogram does not change significantly even if the definition of distance is different, but depending on the data, the definition of distance may have a large effect.

Here is the definition of the distance between the samples used above. Note that each sample is described by q features, and let the data vector of the i-th sample be denoted by $\mathbf{x}_i = (x_{i1},\ldots,x_{iq})^{\mathrm{T}}$, and the data vector of the j-th sample be denoted by $\mathbf{x}_j = (x_{j1},\ldots,x_{jq})^{\mathrm{T}}$. At this time, the distance between samples i and j, $d(\mathbf{x}_i,\mathbf{x}_j)$, is defined as follows.

- Euclidian distance

$$d(\mathbf{x}_i,\mathbf{x}_j)=\sqrt{\sum_{k=1}^{q}(x_{ik}-x_{jk})^2}$$

- Manhattan distance

$$d(\mathbf{x}_i,\mathbf{x}_j)=\sum_{k=1}^{q}\left|x_{ik}-x_{jk}\right|$$

- Minkowski distance

$$d(\mathbf{x}_i,\mathbf{x}_j)=\sqrt[p]{\sum_{k=1}^{q}\left|x_{ik}-x_{jk}\right|^p}$$

- Distance based on correlation

$$d(\mathbf{x}_i,\mathbf{x}_j)=1-r_{ij}=1-\frac{\sum_{k=1}^{q}(x_{ik}-\bar{x}_i)(x_{jk}-\bar{x}_j)}{\sqrt{\sum_{k=1}^{q}(x_{ik}-\bar{x}_i)^2}\sqrt{\sum_{k=1}^{q}(x_{jk}-\bar{x}_j)^2}}$$

Here, $\bar{x}_i=\frac{1}{n}\sum_{k=1}^{q}x_{iq}$, $\bar{x}_j=\frac{1}{n}\sum_{k=1}^{q}x_{jq}$

The Manhattan distance is the origin of its name when traveling around a city divided into squares, such as Manhattan in New York City. In such an urban area, for example, when moving from the point (0, 0) to the point (2, 3), it is not possible to move diagonally (Euclidean distance $\sqrt{13}$) because of the building, and move along the road (Manhattan distance 5) is necessary. Minkowski distance is a generalized form of Euclidean distance and Manhattan distance. It corresponds to the Manhattan distance when p = 1 and the Euclidean distance when p = 2.

For correlation-based distances, calculate the correlation coefficient "between samples instead of between variables" and subtract one from it as the distance. When the correlation is 1, the distance is 0. When the correlation is 0, the distance is 1. When the correlation is -1, the distance is 2. That is, the maximum value is 2 for distances based on the correlation coefficient. When performing cluster analysis based on the similarity of expression patterns between genes, "absolute value of correlation" may be reduced instead of reducing correlation from 1. In this case, the distance is 0 when the correlation is -1 or 1, and the distance is 1 when the correlation is 0.

The function dist can also calculate the following distances: Although it was not suitable for this data, it was not used, but depending on the nature of the data to be analyzed, the distances described below may be appropriate.

- Chebyshev distance
  (Set `method="maximum"`)

$$d(\mathbf{x}_i, \mathbf{x}_j) = \max_k \left(\left| x_{ik} - x_{jk} \right| \right)$$

- Canberra distance
  (Set `method="canberra"`)

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sum\nolimits_{k=1}^{q} \frac{\left| x_{ik} - x_{jk} \right|}{\left| x_{ik} \right| + \left| x_{jk} \right|}$$

- Hamming distance
  (Set `method="binary"`)

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sum\nolimits_{k=1}^{q} (1 - \delta_{x_{ik}, x_{jk}})$$

  Here,

$$\delta_{a,b} = \begin{cases} 1 & (a = b) \\ 0 & (a \neq b) \end{cases}$$

The Chebyshev distance is a distance based only on the difference of one of the q features that is the most different. This distance is the limit $p \to \infty$ of the Minkowski distance. The Hamming distance is a commonly used distance in information science, and it counts the number of positions that do not match when comparing values at the same position for a sequence of the same length. For data that uses the Hamming distance, x_{ik} is not a continuous value but a discrete value (0, 1) in most cases.

So far we have described the definition of the distance between samples. In hierarchical cluster analysis, samples close to each other are grouped into one cluster, and samples and clusters or clusters are further grouped into higher level clusters. Therefore, you need to define not only the distance between samples but also the distance between samples and clusters or between clusters.

First, let's draw a dendrogram based on various definitions of inter-cluster distance. In the hclust function, the calculation method (definition) of the distance between clusters can be specified by the option method.

```
> pdf("fig6.pdf", width = 10, height = 10)
> d <- dist(data.mk)
> op <- par(mfrow = c(2, 3), mar = rep(0, 4))
> tre <- hclust(d, method = "complete")
> myplot(tre, subpop)
> tre <- hclust(d, method = "single")
> myplot(tre, subpop)
> tre <- hclust(d, method = "average")
> myplot(tre, subpop)
> tre <- hclust(d, method = "median")
> myplot(tre, subpop)
> tre <- hclust(d, method = "centroid")
> myplot(tre, subpop)
> tre <- hclust(d, method = "ward.D2")
> myplot(tre, subpop)
> par(op)
> dev.off()
```

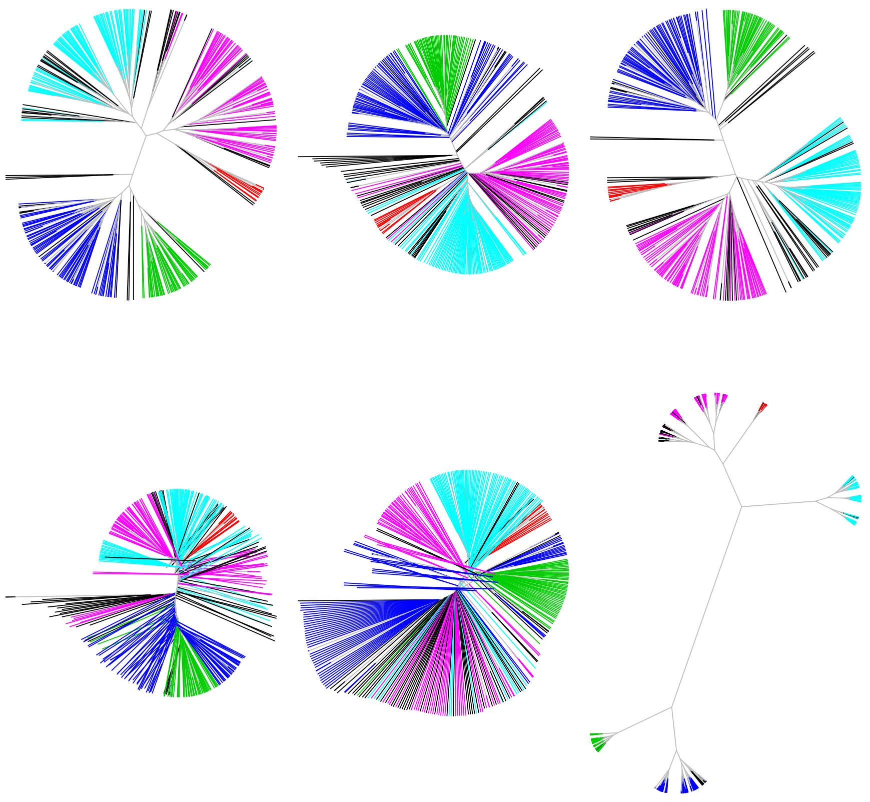

Figure 6. A dendrogram based on various inter-cluster distance definitions

As you can see in Figure 6, the difference in the definition of inter-cluster distance is different from the difference in the definition of inter-sample distance, and the topology of the dendrogram changes significantly. In some cases, the branch length becomes negative and it causes a strange dendrogram (lower left, lower center). Also, differences between clusters may be highly emphasized (lower right). It is difficult to choose which method to use from these definitions. But in many cases, it is chosen that has no major contradiction with known (*a priori*) information. For example, here, it is better to choose one that is less inconsistent with the subpopulation to which the variety/line belongs.

Indicates the definition of the distance between clusters that can be specified by the function hclust. Based on the distance between the samples,

$d(\mathbf{x}_i, \mathbf{x}_j)$, the distance between clusters A and B, d_{AB}, is calculated as follows:

- Maximum distance method (complete connection method)
  (Set `method="complete"`)

$$d_{\mathrm{AB}} = \max_{\substack{i \in A \\ j \in B}} \left(d(\mathbf{x}_i, \mathbf{x}_j) \right)$$

- Minimum distance method (single connection method)
  (Set `method="single"`)

$$d_{\mathrm{AB}} = \min_{\substack{i \in A \\ j \in B}} \left(d(\mathbf{x}_i, \mathbf{x}_j) \right)$$

- Average distance method
  (Set `method="average"`)

$$d_{\mathrm{AB}} = \frac{1}{n_{\mathrm{A}} n_{\mathrm{B}}} \sum_{i \in A} \sum_{j \in B} d(\mathbf{x}_i, \mathbf{x}_j)$$

Here, $n_{\mathrm{A}}, n_{\mathrm{B}}$ represent the numbers of samples included in clusters A and B, respectively.

In the following three definitions, when clusters A and B merge to form a new cluster C, the distance d_{CO} between new clusters C and clusters O other than A and B is defined as follows. The distance between clusters A and B is denoted by d_{AB}, the distance between clusters A and O by d_{AO}, the distance between clusters B and O by d_{BO} and the number of samples contained in clusters A, B and O by n_{A}, n_{B}, and n_{O}.

- Centroid method
  (Set `method="centroid"`)

$$d_{CO}^2 = \frac{n_A}{n_A + n_B} d_{AO}^2 + \frac{n_B}{n_A + n_B} d_{BO}^2 - \frac{n_A n_B}{(n_A + n_B)^2} d_{AB}^2$$

- Median method
  (Set `method="median"`)

$$d_{CO} = \frac{1}{2} d_{AO} + \frac{1}{2} d_{BO} - \frac{1}{4} d_{AB}$$

- Ward's method

(Set `method="ward.D2"`)

$$d_{CO}^2 = \frac{n_A + n_O}{n_A + n_B + n_O} d_{AO}^2 + \frac{n_B + n_O}{n_A + n_B + n_O} d_{BO}^2 - \frac{n_O}{n_A + n_B + n_O} d_{AB}^2$$

Let's compare in more detail the two methods in Figure 6 where the correspondence with the divided groups seems clear.

```
> op <- par(mfrow = c(1, 2))
> d <- dist(data.mk)
> tre <- hclust(d, method = "complete")
> myplot(tre, subpop, type = "phylogram")    #  draw the graph as a phylogram
> tre <- hclust(d, method = "ward")
> myplot(tre, subpop, type = "phylogram")
> par(op)
```

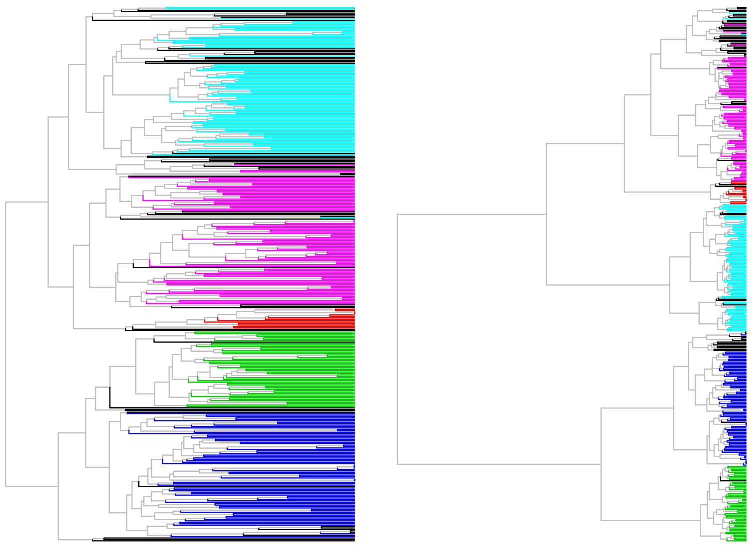

Figure 7. Dendrograms of the longest distance method (left) and Ward method (right)

<Cluster analysis from both sides of multidimensional data>
So far, varieties and lines have been classified into clusters based on DNA marker data. Cluster analysis of varieties and lines can be performed based on trait data as well as DNA marker data. Also, regarding the same data, it is possible to classify traits having similar variation patterns among varieties/lines into the same cluster, considering them as targets for classifying traits instead of vareties/lines. Here we will explain such an approach.

First, let's prepare trait data. Let's extract trait data from all data (alldata), excluding traits not suitable for such analysis.

```
> required.traits <- c("Flowering.time.at.Arkansas",
        "Flowering.time.at.Faridpur", "Flowering.time.at.Aberdeen",
        "Culm.habit", "Flag.leaf.length", "Flag.leaf.width",
        "Panicle.number.per.plant", "Plant.height", "Panicle.length",
        "Primary.panicle.branch.number", "Seed.number.per.panicle",
        "Florets.per.panicle", "Panicle.fertility", "Seed.length",
        "Seed.width","Brown.rice.seed.length", "Brown.rice.seed.width",
        "Straighthead.suseptability","Blast.resistance",
        "Amylose.content", "Alkali.spreading.value", "Protein.content")
> data.tr <- alldata[, required.traits]     #  extract required traits
> missing <- apply(is.na(data.tr), 1, sum) > 0   #  samples with missing
> data.tr <- data.tr[!missing, ]            #  remove the samples
> subpop.tr <- alldata$Sub.population[!missing]  #  subpop info
```

The trait data varies in size (variance) depending on the trait. If this data is used as it is, the large variance trait has a large effect on the distance calculation, and the small variance trait has a small contribution to the distance calculation. Therefore, all traits are normalized to variance 1.

```
> data.tr <- scale(data.tr)        #  Standardization of data
```

Now let's perform cluster analysis with traits classified as variety/line and cluster analysis with traits classified as trait data.

```
> d <- dist(data.tr)                # calculate distance
> tre.var <- hclust(d, method = "ward.D2")
                              # Clustering varieties/lines with Ward’s method
> d <- dist(t(data.tr))            # Calculate distance among traits
> tre.tra <- hclust(d, "ward.D2")
                              # Clustering traits with Ward’s method
> op <- par(mfrow = c(1, 2))
> myplot(tre.var, subpop.tr, type = "phylogram")
> plot(tre.tra, cex = 0.5)
> par(op)
```

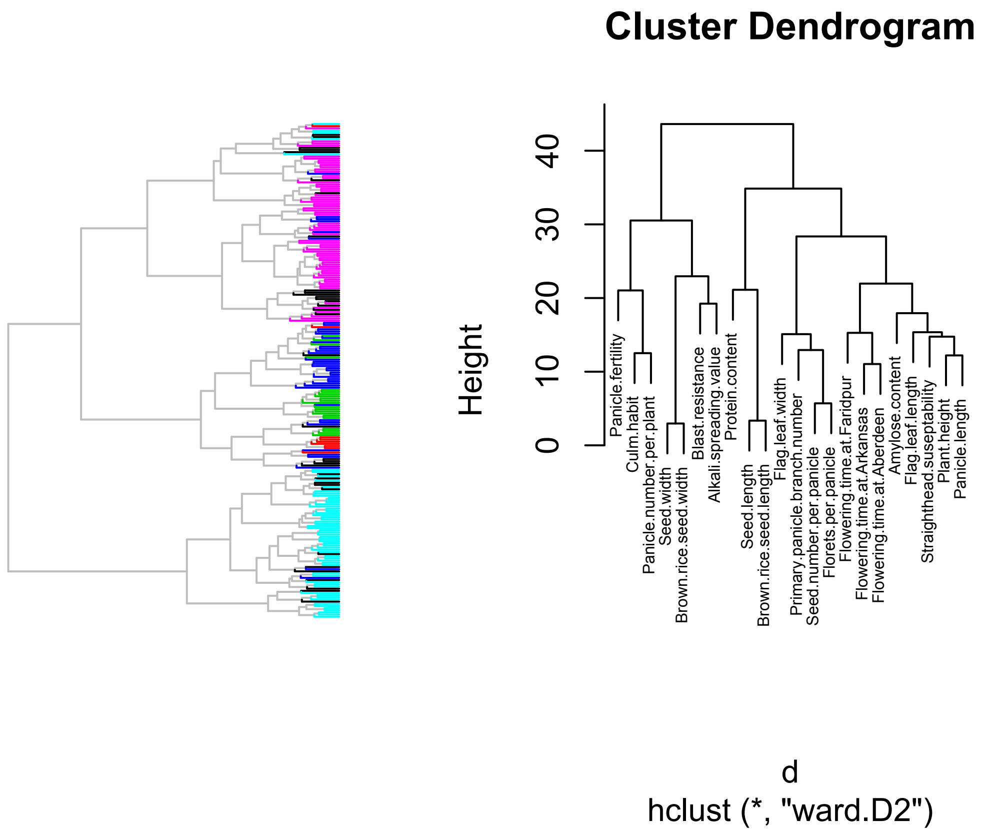

Figure 8. Cluster analysis based on trait data
Dendrograms showing the relationship between varieties and lines (left) and the relationship between traits (right)

From the dendrogram on the right side of Figure 8, it can be seen that the traits (Plant. Height, Panicle. Length, Flag. Leaf. Length) related to the size of the plant are closely related to each other. You can also see that the flowering timing (Flowering.time.at. *****) located in the three environments is located near the cluster. In addition, it is also clear that the flag leaf width (Flag.leaf.width) is different from other size-related traits and strongly associated with the traits that characterize the panicle. In this way, multidimensional data can be cluster analyzed from either side. With this in mind, you will be able to view the same data from a slightly different perspective.

Note that the above analysis can display results more visually using the heatmap function.

```
> pdf("fig9.pdf")
> heatmap(data.tr, margins = c(12,2))
> dev.off()
```

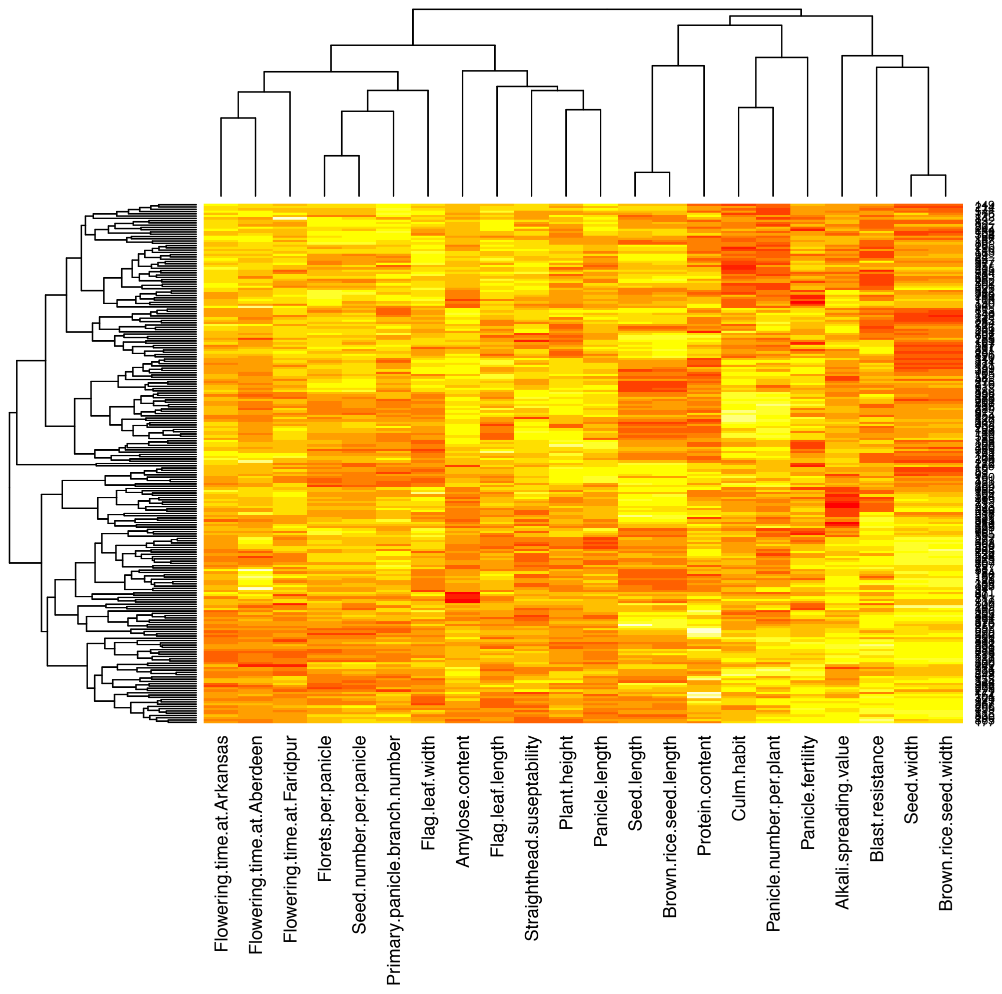

Figure 9. Display of cluster analysis results and heatmap of trait data

There is also a heatmap function, heatmap.2, which is included in the gplots package (I think there are many other functions). Let's draw a heat map using this. Although the way of giving options is slightly different, you can draw a nice-looking figure.

```
> # this part is optional (plot with heatmap.2)
> require(gplots)
(omitted)

> pdf("fig9-2.pdf", height = 12)
> heatmap.2(data.tr, margins = c(12,2), col=redgreen(256), trace = "none",
lhei = c(2,10), cexRow = 0.3)
> dev.off()
quartz
    2
```

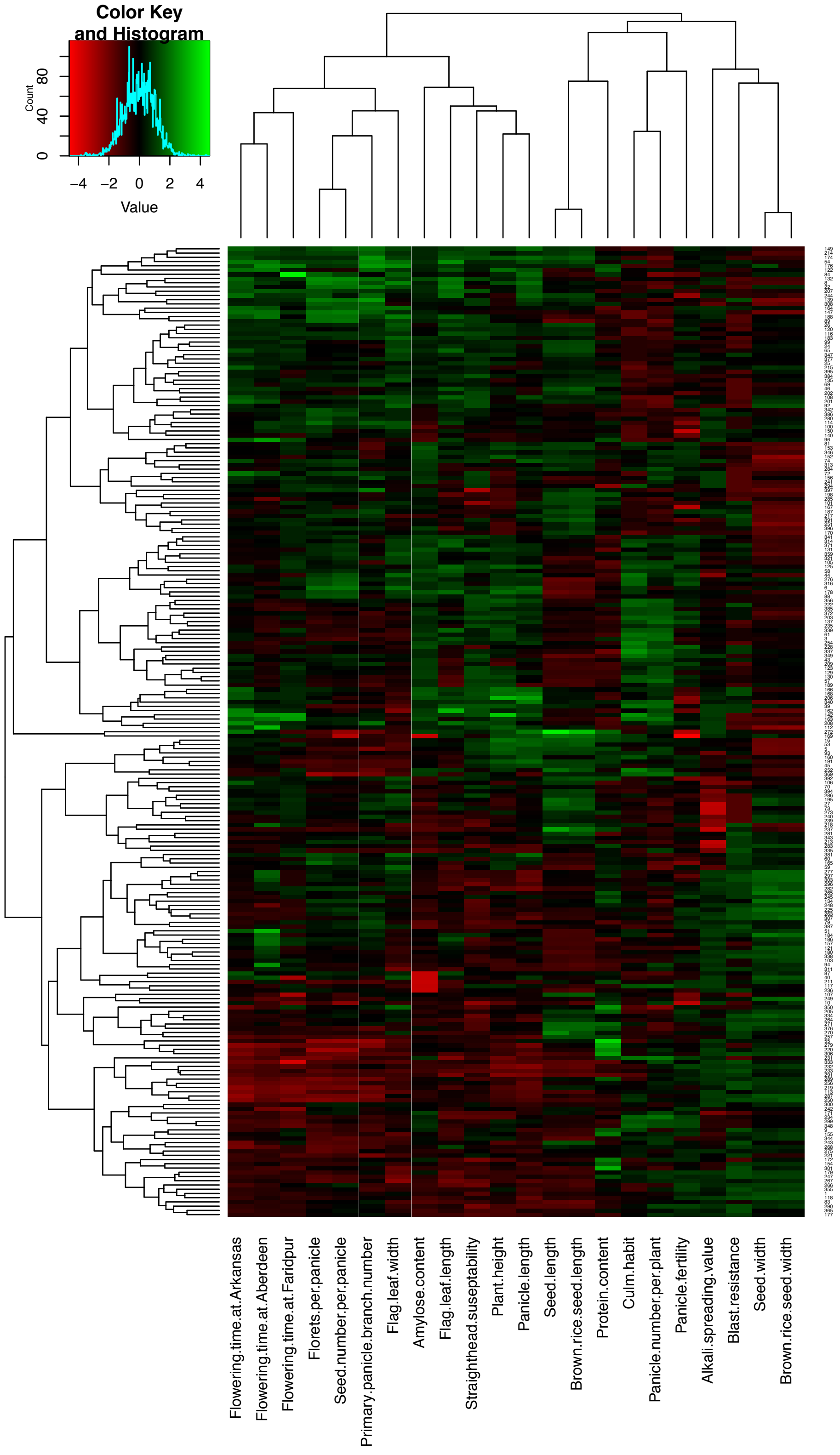

Figure 9-2. Display heat map of trait data using heatmap.2 function

You can reflect the results of another cluster analysis as follows: Reflect the result of the cluster analysis performed for the same data on the heat map display.

```
> pdf("fig10.pdf")
> heatmap(data.tr,                  #  trait data
         Rowv = as.dendrogram(tre.var),      #dendrogram for varieties/lines
         Colv = as.dendrogram(tre.tra),      #  dendrogram for traits
         RowSideColors = as.character(as.numeric(subpop.tr)),
         labRow = subpop.tr,        #  replace the labels
         margins = c(12, 2))        #  set margins of the graph
> dev.off()
```

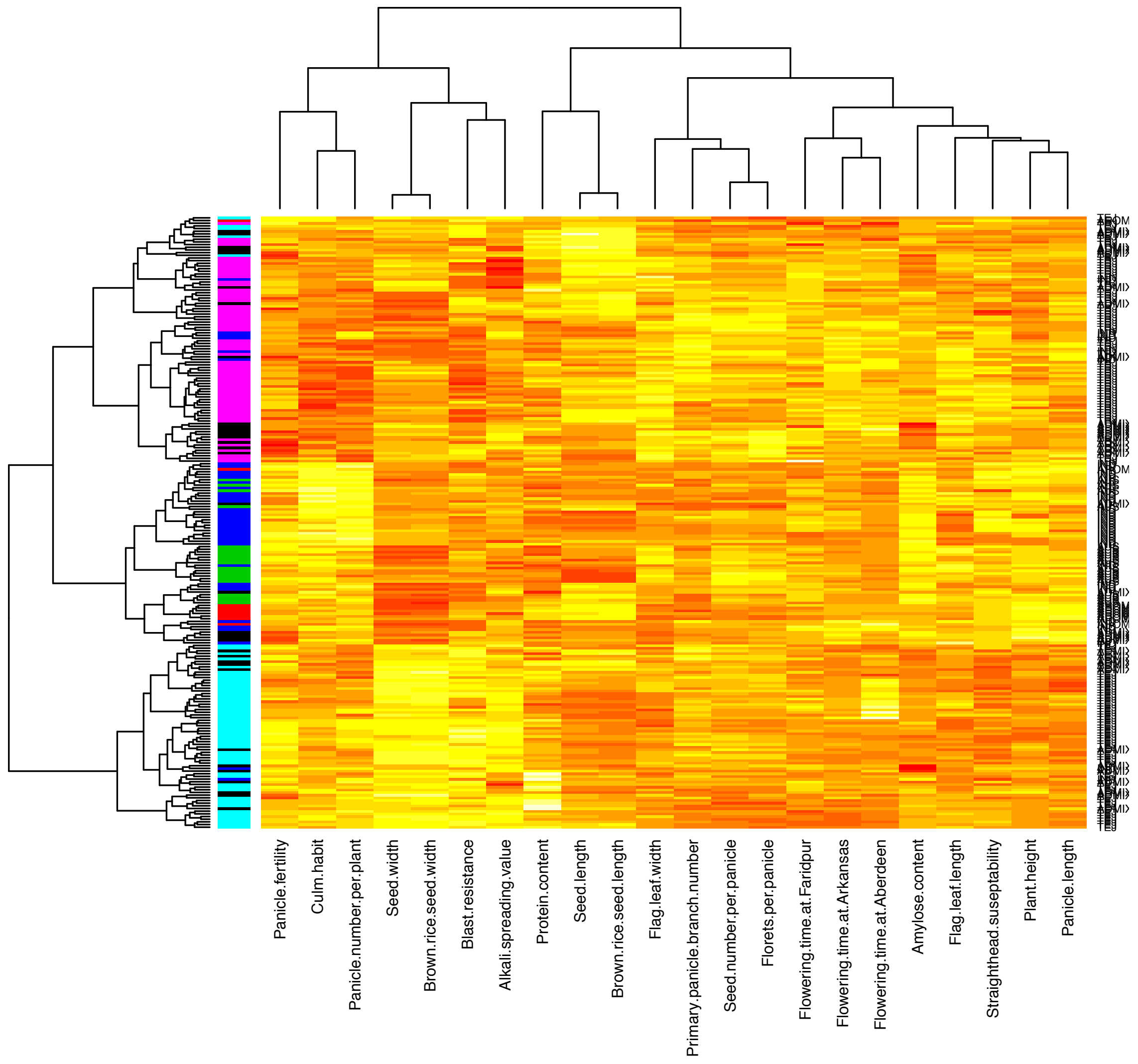

Figure 10. The result of changing the cluster analysis method in Figure 9 to Ward's method

This can also be drawn using the heatmap.2 function as before.

```
> # perform clustering with appropriate methods
> pdf("fig10.pdf")
> heatmap(data.tr, Rowv = as.dendrogram(tre.var),
+                                   Colv = as.dendrogram(tre.tra),
+                                   RowSideColors                          =
as.character(as.numeric(subpop.tr)),
+                                   labRow = subpop.tr,
+                                   margins = c(12, 2))
> dev.off()
quartz
    2
```

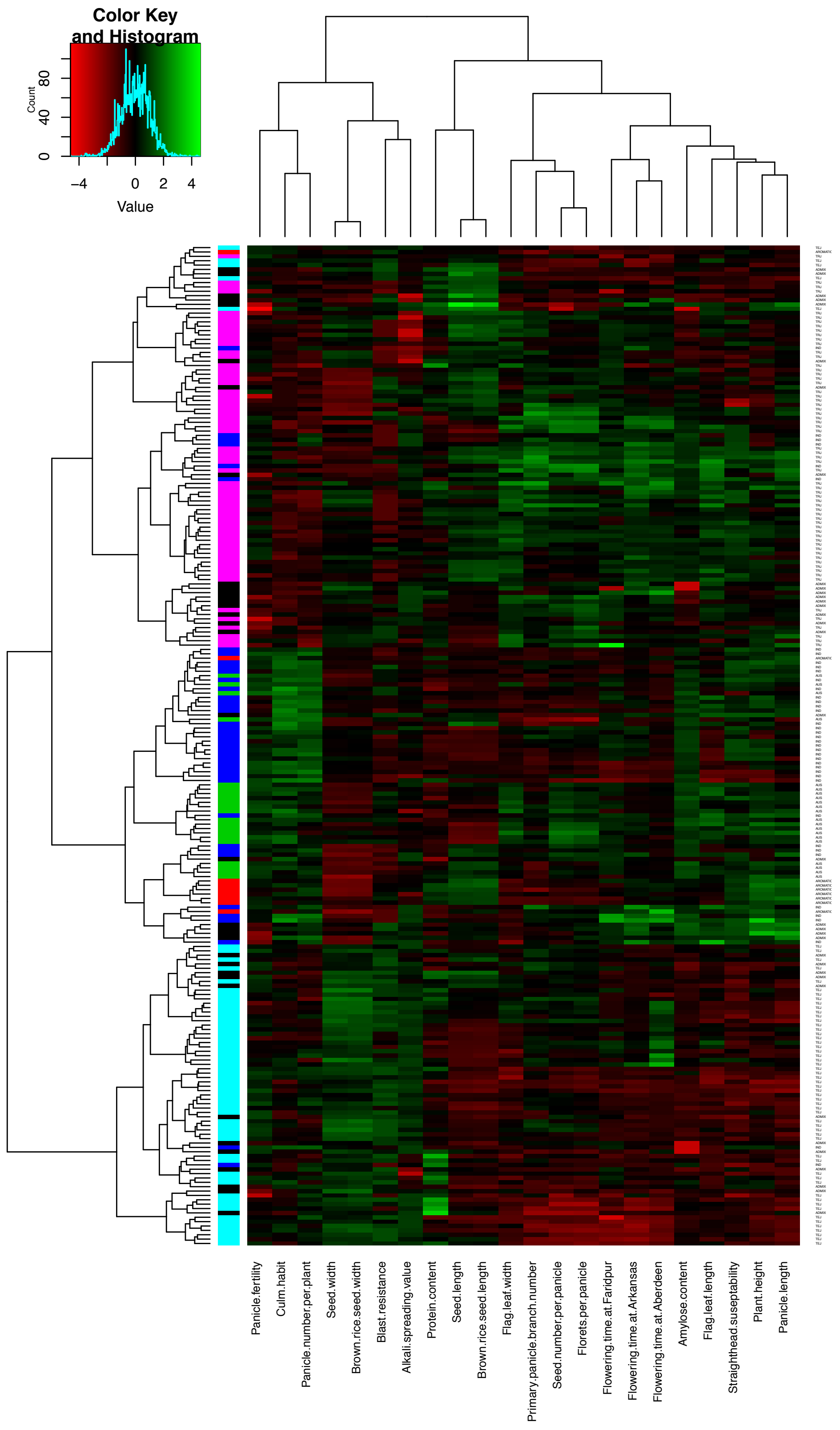

Figure 10-2. The result of changing the cluster analysis method of Figure 9-2 to Ward's method

The heatmap function does not have to perform cluster analysis on the same data in both vertical and horizontal directions. For example, you can also apply the results of cluster analysis using DNA marker data for the row side.

```
> data.mk2 <- data.mk[!missing, ]   #  remove samples with missing trait data
          #  to make the sample number same between two datasets
> d <- dist(data.mk2)               #  calculate distance with DNA data
> tre.mrk <- hclust(d, method = "ward.D2")  # Clustering with Wald's method
> pdf("fig11.pdf")
> heatmap(data.tr, Rowv = as.dendrogram(tre.mrk),
                           #  this part is different from the above
                 Colv = as.dendrogram(tre.tra), #  the remainders are same
                 RowSideColors = as.character(as.numeric(subpop.tr)),
                 labRow = subpop.tr,
                 margins = c(12, 2))
> dev.off()
```

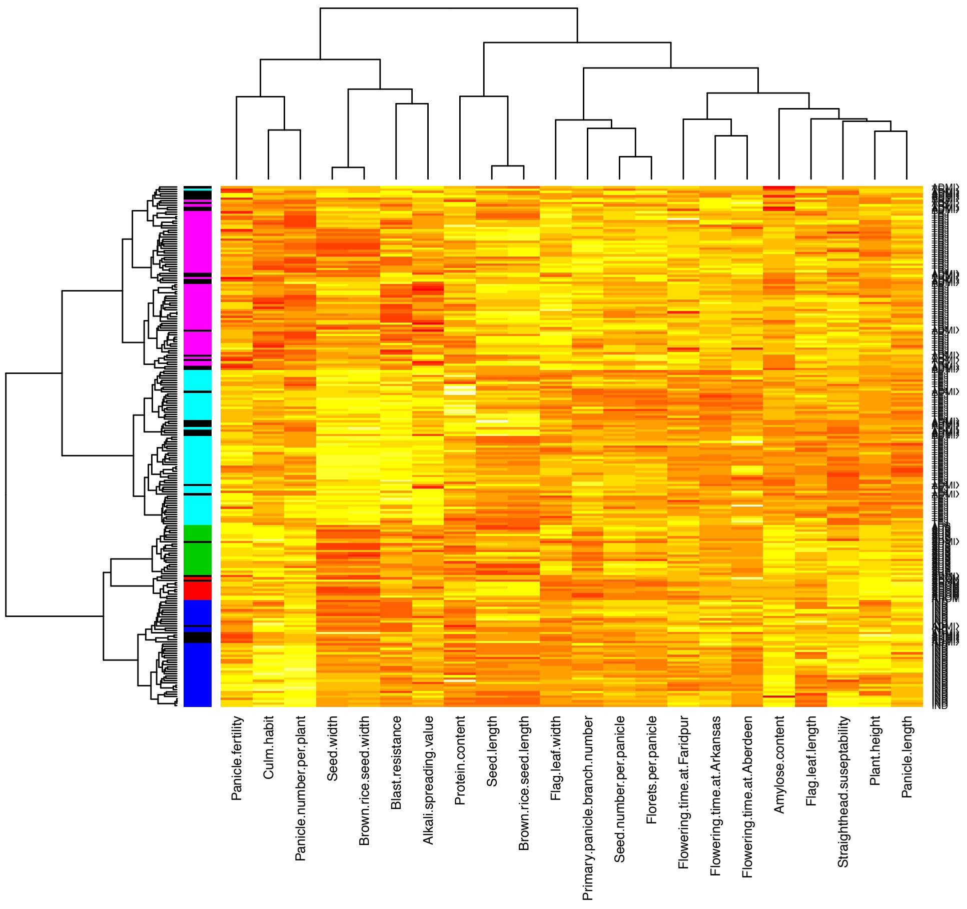

Figure 11. Relationship between the results of cluster analysis based on

This can also be drawn using the heatmap.2 function as before.

```
> # perform clustering with appropriate methods
> pdf("fig10.pdf")
> heatmap(data.tr, Rowv = as.dendrogram(tre.var),
+                                   Colv = as.dendrogram(tre.tra),
+                                   RowSideColors                          =
as.character(as.numeric(subpop.tr)),
+                                   labRow = subpop.tr,
+                                   margins = c(12, 2))
> dev.off()
quartz
    2
```

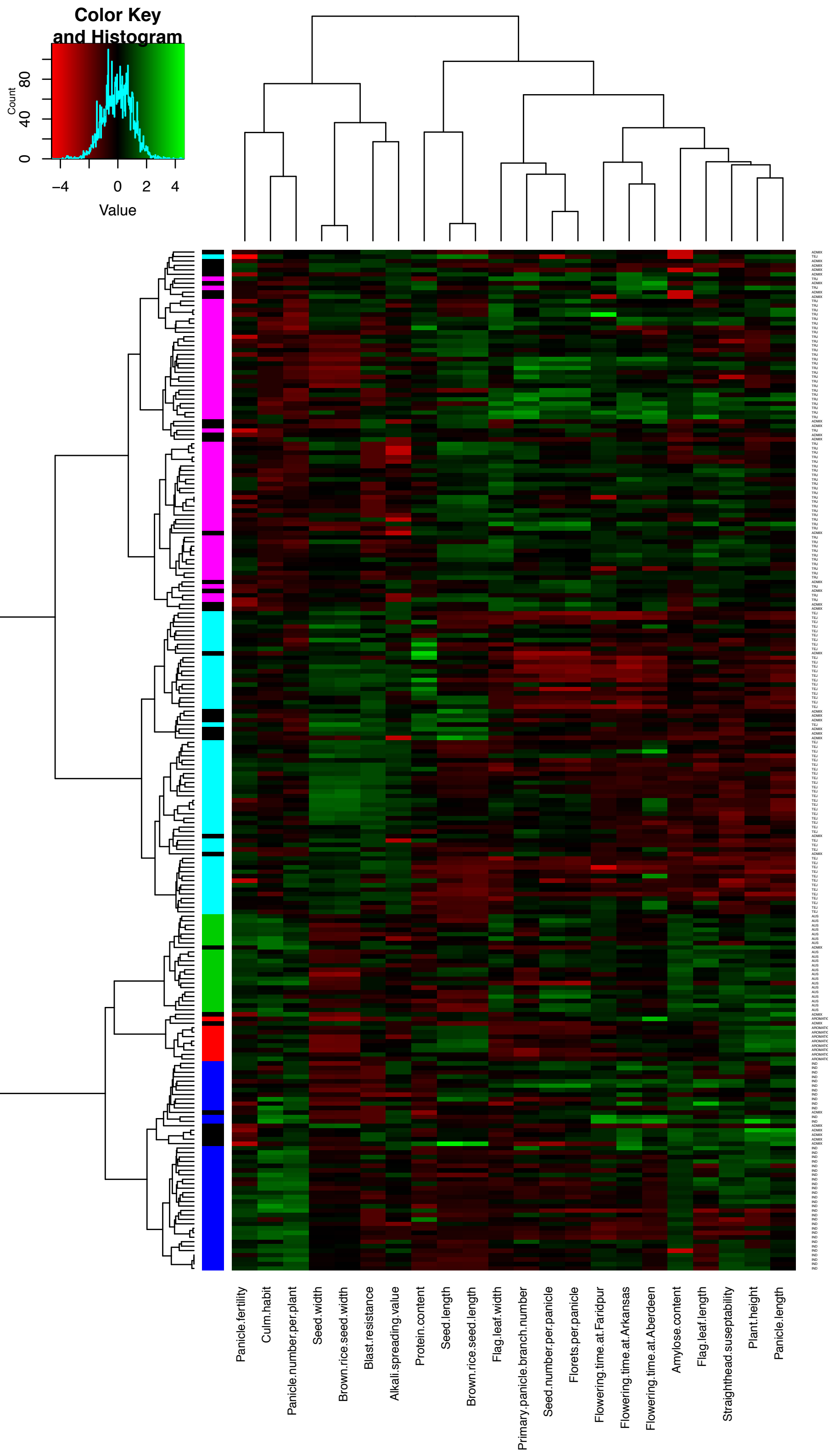

Fig. 11-2. Relationship between the results of cluster analysis and traits based on genetic marker data expressed using heatmap. 2 function

<Classification based on hierarchical cluster analysis>
Sometimes you want to delineate samples into places where there is similarity between samples and to categorize samples discretely into groups. This section describes how to classify samples into a fixed number of clusters based on the results of hierarchical cluster analysis.

Based on the results of hierarchical cluster analysis based on DNA marker data, let's classify varieties/lines into five groups. Five is a number according to the number of division groups to which varieties/lines belong. From the result of hierarchical cluster analysis, we use the function cutree to find discrete groups.

```
> d <- dist(data.mk)
> tre <- hclust(d, method = "ward.D2")                タ解析
> cluster.id <- cutree(tre, k = 5)
         #  classify samples into 5 groups
> cluster.id                              #  show the result
```

Let's illustrate the results of classification into five groups based on cluster analysis.

```
> op <- par(mfrow = c(1,2), mar = rep(0, 4))  #  set graphic options
> myplot(tre, cluster.id, type = "phylogram")
                   #  color the terminals with the result of cutree
> myplot(tre, subpop, type = "phylogram", direction = "leftwards")
         #  color the terminal with subpop info
> par(op)
```

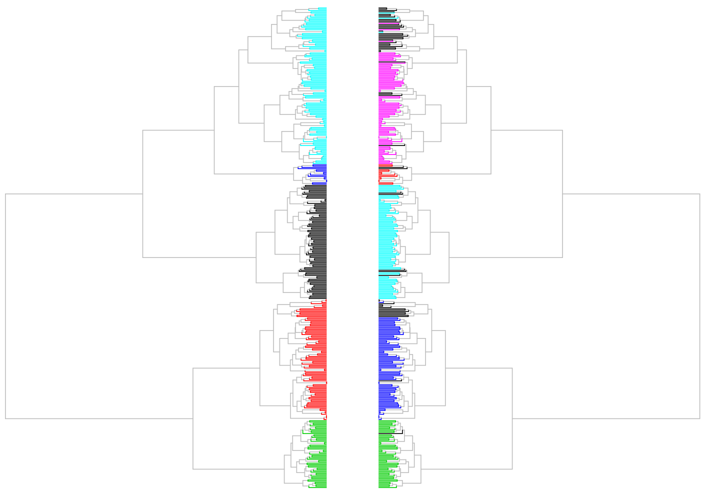

Fig. 12. Relationship between classification by cluster analysis (left) and divided groups (right)

Let's check the relationship between classification and subpopulation based on cluster analysis by creating cross tabulation table.

```
> table(cluster.id, subpop)      #  cluster.id と subpop のクロス集計表を表示
                 subpop
cluster.id ADMIX AROMATIC AUS IND TEJ TRJ
         1     5        0   0   0  84   0
         2    14        0   0  80   0   0
         3     1        0  52   0   0   0
         4     2       14   0   0   0   0
         5    34        0   0   0   3  85
```

It can be seen that the two match very well, except for the three varieties / lines of Indica (IND). This is thought to be because the divided population structure itself was estimated based on DNA marker data. In addition, it is also understood that varieties that are estimated to be a mixture of multiple subpopulations (ADMIX) are classified into various groups.

Let's confirm the result of classification by hierarchical cluster analysis on the principal component axis.

```
> pca <- prcomp(data.mk)           #  PCA
> op <- par(mfrow = c(1,2))        #
> plot(pca$x[,1:2], pch = cluster.id, col = as.numeric(subpop))
        #  draw scatterplot with PC1 and 2
        #  shapes of dots represent the result
        # colors of dots represent subpop info
> plot(pca$x[,3:4], pch = cluster.id, col = as.numeric(subpop))
```

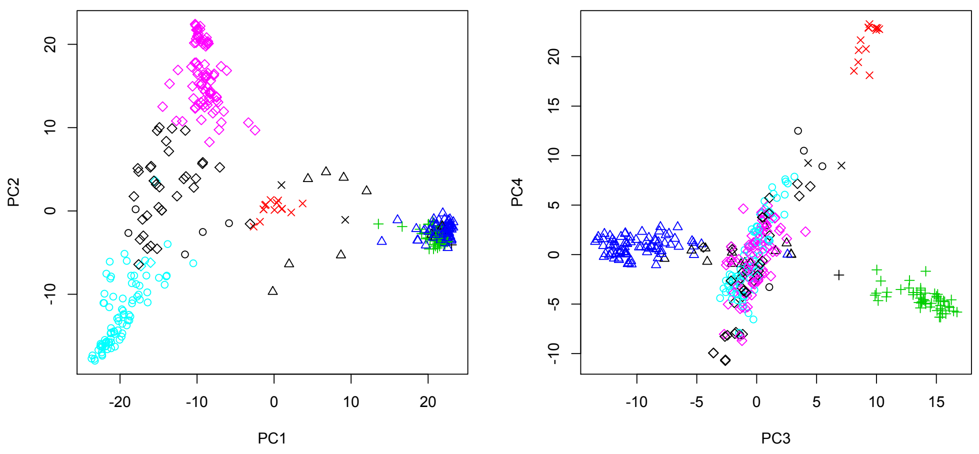

Figure 13. Classification by cluster analysis and relationship between subgroups

<Non-hierarchical clustering>
When classifying into a certain number of groups, it is not necessary to classify hierarchically. Here we introduce the k-means method, which is one of the non-hierarchical cluster analysis methods.

For the same data as before, let's classify it into five groups using the function kmeans.

```
> kms <- kmeans(data.mk, centers = 5) #  classify samples into 5 groups
> kms                                  #  show the result
 (omitted)
```

The k-means method performs classification into the number of groups determined by the following algorithm.

1. randomly choose k samples as k cluster centers
2. Find the distance between all data points and k cluster centers, and classify each data point into the closest cluster (centering on the center of gravity)
3. Update the center (center of gravity) of the formed cluster
4. Repeat 2-3 until the cluster center (center of gravity) does not change

In the k-means method, the results may vary depending on the first randomly chosen sample. In fact, let's repeat the analysis with the same data and check the variation of the results.

```
for(i in 1:5) {            #  repeat the same analysis five times
        kms <- kmeans(data.mk, centers = 5, nstart = 50)
#  nstart = 50 repeat analysis with 50 different sets of chosen samples
        print(table(kms$cluster, subpop))
 }
```

Unlike before, you can see that the results are stable. Note that although the “number” of the group into which each sample is classified varies among different analyzes, this is not a particular problem because it is an arbitrary number assigned to five groups.

Let's create a cross-tabulation table and check the relationship between groups classified by k-means, groups classified by hierarchical cluster analysis, and division groups to which varieties / lines belong.

```
> table(kms$cluster, subpop)       #  compare k-means and subpop
   subpop
    ADMIX AROMATIC AUS IND TEJ TRJ
  1     1        0  52   0   0   0
  2    23        0   0   0   1  85
  3    17        0   0   0  86   0
  4    12        0   0  80   0   0
  5     3       14   0   0   0   0
 > table(cluster.id, subpop)
                    #  compare hierarchical clustering and subpop
cluster.id ADMIX AROMATIC AUS IND TEJ TRJ
         1     5        0   0   0  84   0
         2    14        0   0  80   0   0
         3     1        0  52   0   0   0
         4     2       14   0   0   0   0
         5    34        0   0   0   3  85
> table(kms$cluster, cluster.id)
                    #  compare hierarchical clustering and k-means

   cluster.id
      1   2   3   4   5
  1   0   0  53   0   0
  2   0   1   0   0 108
  3  88   1   0   0  14
  4   0  92   0   0   0
```

From cross-tabulation tables, it can be seen that varieties and lines belonging to the subpopulations other than ADMIX and IND are classified in the same way by k-means and hierarchical cluster analysis. Looking at the third crosstabulation table, the classification results of both methods are almost identical, while some differences can be seen. This is mainly due to the fact that the classification of varieties/lines in which the subpopulation is ADMIX (mixed) differs in both methods.

Let's confirm the result of classification by k-means method and hierarchical cluster analysis by plotting on principal component axis.

```
> convert.table <- apply(table(kms$cluster, cluster.id), 1, which.max)
   #  to match the ID of clusters between two different methods
> convert.table            #  table for converting IDs
1 2 3 4 5
5 3 4 2 1
> cluster.id.kms <- convert.table[kms$cluster]       #  convert IDs
> pdf("fig14.pdf", width = 8, height = 8)    #
> op <- par(mfrow = c(2,2))
> plot(pca$x[,1:2], pch = cluster.id, col = as.numeric(subpop),
                main = "hclust")           #  the result of hclust
> plot(pca$x[,3:4], pch = cluster.id, col = as.numeric(subpop),
                main = "hclust")
> plot(pca$x[,1:2], pch = cluster.id.kms, col = as.numeric(subpop),
                main = "kmeans")           #  the result of k-means
> plot(pca$x[,3:4], pch = cluster.id.kms, col = as.numeric(subpop),
                main = "kmeans")
> par(op)
> dev.off()
```

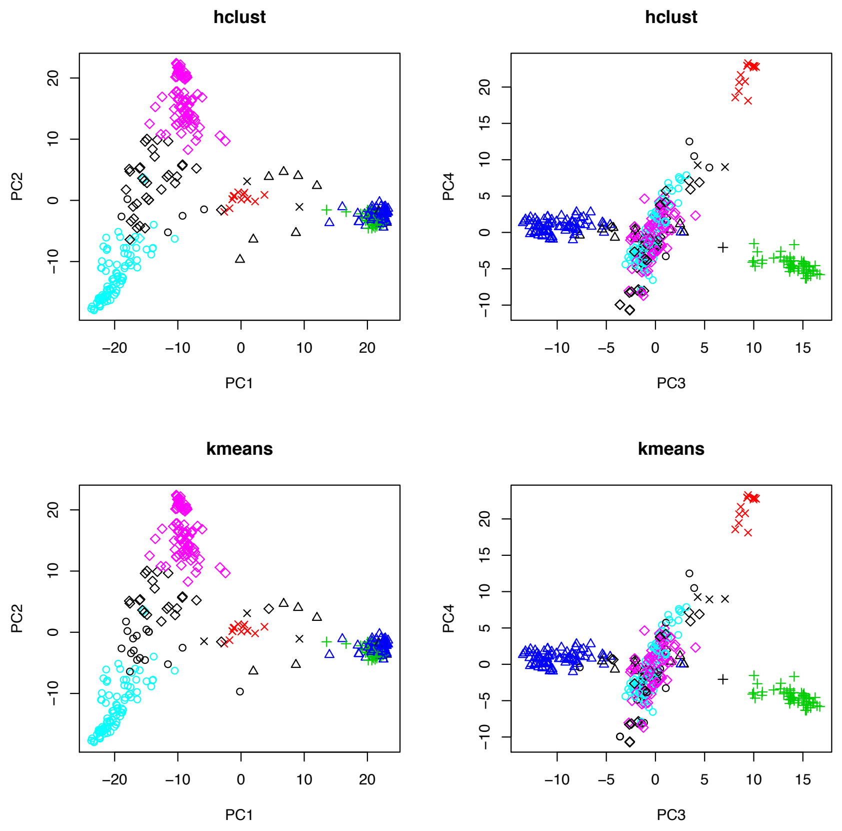

Figure 14. Classification by hierarchical cluster analysis (top) and k-means (bottom) Relationship between the score and the principal component score

<Determination of appropriate number of groups>
The number of groups classified up to this point has been set to 5 according to the number of divided groups. What should we do to check if the number of 5 groups is really appropriate? One way to determine the appropriate number of groups is to classify them into various numbers of groups and determine the degree of decrease in the within groups sum of squares at that time. there is.

As explained in the analysis of variance, the sum of squares is divided into the sum of squares between groups and the sum of squares within groups. Therefore, when the number of groups is 1, the sum of squares is the sum of squares within groups. Then, as the number of groups increases to 2, 3, 4 ..., the sum of squares between groups increases and the sum of squares within groups decreases. When the number of groups finally matches the number of samples, the sum of squares within groups becomes 0. Therefore, the rule of minimizing the sum of squares within a group is meaningless because the number of groups is always the number of samples. Therefore, as with the rules for determining the number of components in principal component analysis, find the point where the reduction of the sum of squares within a group changes from a sudden change to a gradual change, and use it as the number of groups to adopt.

Now let's change the number of groups from 1 to 10 and calculate the sum of squares within a group. Graphically illustrate how it decreases.

```
> n <- nrow(data.mk)                # number of samples
> wss <- rep(NA, 10)       # prepare a vector for within-group sum of squares
> wss[1] <- (n - 1) * sum(apply(data.mk, 2, var))
                  # calculate variance and convert it to sum of squares
> for(i in 2:10) {
         print(i)          # show the progress with printing i
         res <- kmeans(data.mk, centers = i, nstart = 50)
                                    # k-means with k = 2-10
         wss[i] <- sum(res$withinss)
}
 (omitted)
> plot(1:10, wss, type = "b", xlab = "Number of groups",
                            ylab = "Within groups sum of squares")
```

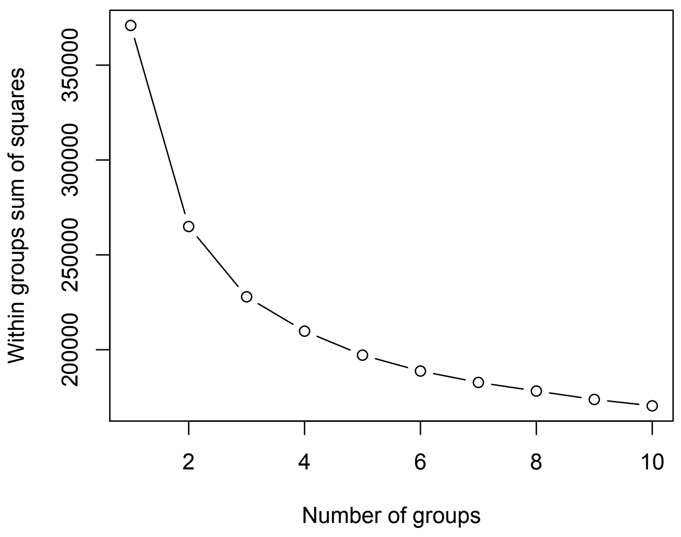

Figure 15. Change in the within group variation when the number of clusters is 1 to 10 by k-means method

Looking at Fig. 15, we can see that the decrease in the within group sum of squares becomes linear after the number of clusters exceeds five. Also from this figure, the number 5 is considered to be a suitable number of groups.

<Detection of Ambiguous Classification Samples>
So far, each sample has always been classified into one group. Then, among the samples classified into a certain group, there will be those that are clearly classified into that group and those that are "barely" classified into that group. In order to clarify the certainty of classification, it is useful to be able to detect a sample with an ambiguous classification like the latter by some standard. Here, we will introduce a method to evaluate classification ambiguity based on statistics called shadow value (Everitt and Hothorn 2011, An introduction to applied multivariate analysis with R. Springer).

In the kms function, the position of the center of gravity of each group determined by the k-means method is calculated. Here, we calculate the distance to the center of gravity of these groups from each sample, calculate the distance to the nearest group and the distance to the next closest group, and evaluate the ambiguity based on the difference.

First we calculate the distance between all samples and group centroids using the function rdist which is included in the package fields.

```
> require(fields)                  #  Use fields
> d2ctr <- rdist(kms$centers, data.mk)
 #  calculate distance between group centroids (kms$centers) and samples
> d2ctr                            # check the result
 (omitted)
> apply(d2ctr, 2, which.min)       #  find the closest group
 (omitted)
> kms$cluster                      #  the result of k-means
 (omitted)
```

The k-means classifies each sample into the closest group to the center of gravity, as mentioned earlier. Therefore, note that the two results displayed in the upper box match.

From the distances calculated for each sample, you can use the min function to derive the distance to the nearest centroid. But how do you get the distance to the second closest center of gravity? Here, we realize this by creating a self-made function nth.min. The self-made function nth.min is a function that rearranges the array given as the argument x in order of size (ascending order) and returns the nth value.

```
> nth.min <- function(x, n) {       #  define self-made function
        sort(x)[n]                  #  return nth sample after sorting
 }
> nth.min(-10:10, 3)               # test the function
> d.1st <- apply(d2ctr, 2, min)     #  obtain distance to the closest center
> d.2nd <- apply(d2ctr, 2, nth.min, n = 2)
                 #  use nth.min to obtain distance to the send closest center
```

When the command in the upper box is executed, d.1st is assigned the distance from each sample to the nearest centroid, and d.2nd is assigned the distance to the second closest centroid.

Next, let's calculate the shadow value. The shadow value for the ith sample is defined as

.

Where is the distance from the observed value of the ith sample to the centroid of the nearest group (the centroid of the group into which the sample was classified), and represents the distance to the centroid of the second closest group. This value is from 0 to 1. If this value is close to 0, it indicates that the sample is located near the center of gravity of the classified group; conversely, if it is close to 1, the gravity center of the classified group and the second closest group are It means that the distance to the center of gravity is almost the same. Therefore, in order to detect a sample whose classification is ambiguous, it means that shadow value should find a sample close to 1.

Let's calculate the shadow value using the distances d.1st and d.2nd calculated above and find out if the value is 0.9 or more.

```
> shadow <- 2 * d.1st / (d.1st + d.2nd)
> unclear <- shadow > 0.9  #  if shadow is larger than 0.9, unclear = T
```

The result of detection is assigned to "unclear". If this value is T (true), the classification is considered ambiguous, if it is F (false), the classification is considered relatively clear.

Now let's draw a scatter plot on the principal component axis, representing the ● of the sample whose classification is determined to be ambiguous.

```
> cluster.id.kms[unclear] <- 20     #  make the value 20 when unclear = F
                                    #  20 is the code for ●
> op <- par(mfrow = c(1,2))
> plot(pca$x[,1:2], pch = cluster.id.kms, col = as.numeric(subpop),
                     main = "kmeans")  #  図 15 と同様に散布図を描く
> plot(pca$x[,3:4], pch = cluster.id.kms, col = as.numeric(subpop),
                     main = "kmeans")
> par(op)
```

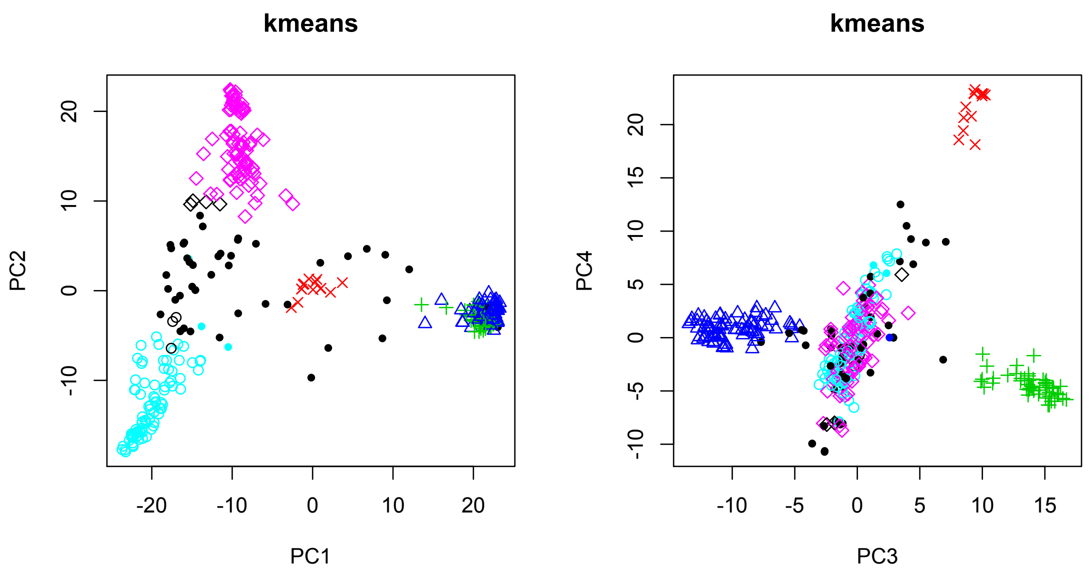

Figure 16. Scatter plot of the sample with unclear classification by ●

It can be seen from Fig. 16 that most of the varieties and lines (black points) in which the subpopulations are ADMIX (mixed) are scattered with ●. In this way, by evaluating the ambiguity (in other words, the certainty) of the classification of each sample, it becomes possible to grasp the classification result in more detail. In this example, we were able to find varieties/lines that seemed to be mixed backgrounds resulting from multiple subpopulations.

<Selection of representative sample>
Cluster analysis can also be used to select a small number of representative samples from a large number of samples. For example, classification can be performed by cluster analysis based on existing data collected for a large number of genetic resources, and representative varieties/lines can be selected based on the classification results. In this way, representative varieties/lines are often selected, and field trials and molecular biological experiments that require time and cost are often performed using these varieties/lines.

Here, we introduce the k-medoids method as a method of cluster analysis suitable for selection of such representative samples. The k-medoids method is similar to the k-means method, but instead of grouping based on the distance to the center of gravity of the group, grouping based on the distance to the representative samples of the groups (medoids). More specifically, this algorithm does not use the center of a cluster as the center of gravity, but as the coordinate point of the representative sample of the group.

```
> require(cluster)   #  package pam for k-medoids method
> kmed <- pam(data.tr, k = 48)     #  perform k-medoids
                                   #  k = 48
> kmed                             # show the result
 (omitted)
> kmed$id.med                      #  IDs of representative samples
 [1]   1  28   8 192 121  80   7  53  10 218  33  15  63 191  18  83 126  27
[19]  93  98  36  38 202 106  52  54 148 136 101  62 211 161  86 145  85  91
[37]  92 207 123 203 124 159 178 137 138 162 166 214
```

Note that the phenotypic data (data.tr) used here has already been normalized to have variance 1. If you perform the same analysis on your own data, carefully consider whether it is necessary to scale the data, and if necessary, use the scale function to scale.

Now, let's illustrate the variation in the samples selected as representatives with scatter plots on the principal component axis.

```
> pca.tr <- prcomp(data.tr)         #  PCA
> mypch <- rep(1, nrow(data.tr))    #  repeat 1 the number of samples
> mypch[kmed$id.med] <- 19  #  change 1 to 19 for the representativess
                                    #  1 is code for ○, 19 for ●
> op <- par(mfrow = c(1,2))
> plot(pca.tr$x[,1:2], col = as.numeric(subpop.tr), pch = mypch)
> plot(pca.tr$x[,3:4], col = as.numeric(subpop.tr), pch = mypch)
> par(op)
```

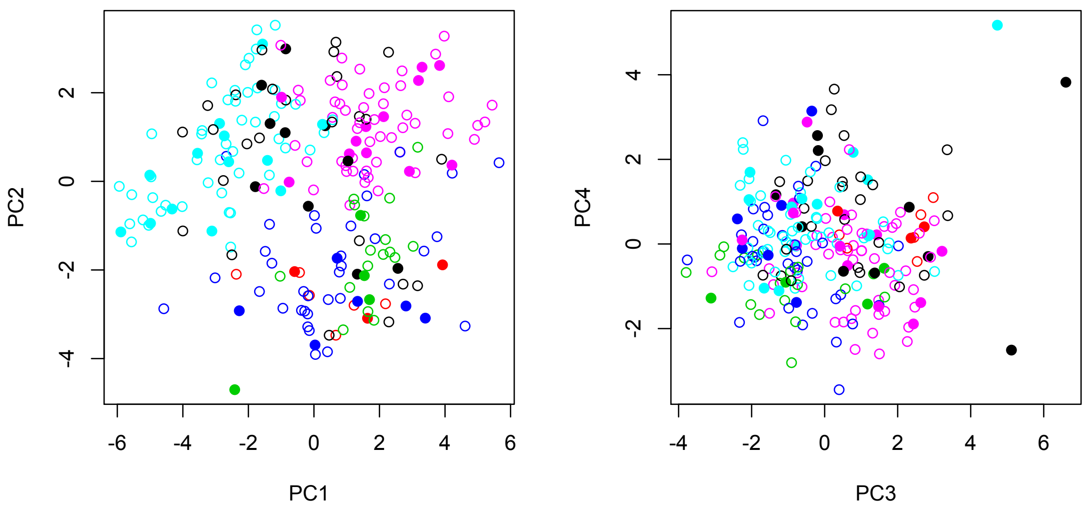

Figure 17. Distribution of representative 48 varieties / lines selected by k-medoids method

Finally, let's compare the distribution of principal component scores of 48 varieties / lines selected using the k-medoids method with the distribution of principal component scores of all varieties / lines by drawing a histogram.

```
> op <- par(mfcol = c(2,4))
> for(i in 1:4) {          #  For PC1 to PC4
          res <- hist(pca.tr$x[,i], main = paste("PC", i, "all"))
          #  histogram of all varieties/lines
          #  the information of the histogram is used in the next line
          hist(pca.tr$x[kmed$id.med, i], breaks = res$breaks,
                    main = paste("PC", i, "k-medoids"))
                    #  histogram of representatives
#  Use the same breaks for the histogram of all varieties (res$breaks)
}
> par(op)
```

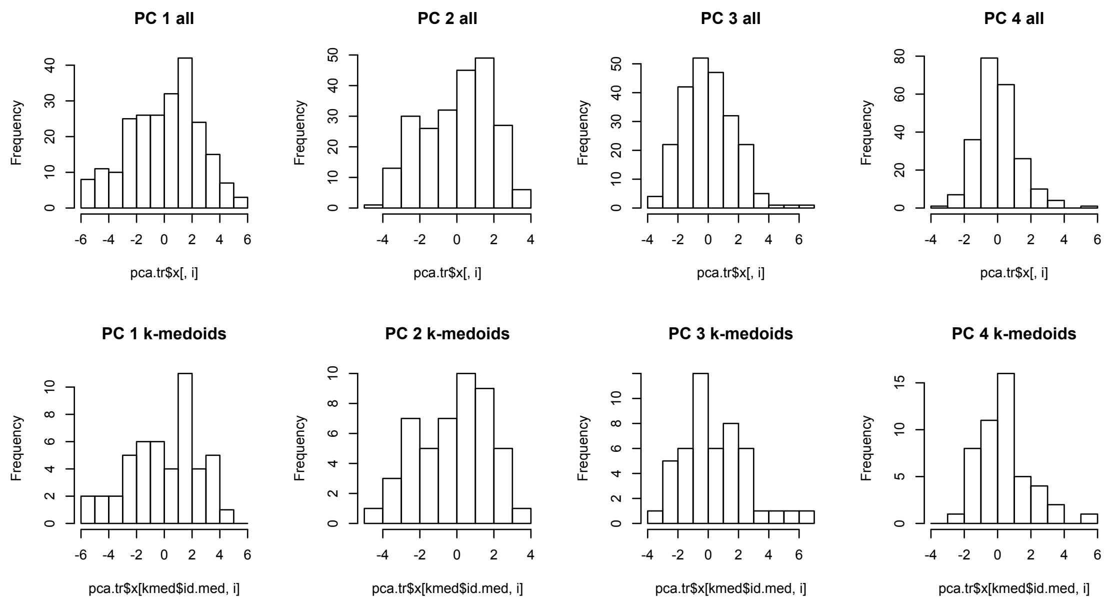

Figure 19. Distribution of principal component scores of all varieties / lines (top) and varieties / lines selected as representatives (bottom)

It can be seen from Fig. 19 that the 48 varieties / lines selected by the k-medoids method represent the variation of the traits possessed by all the varieties / lines.

Thus, cluster analysis can also be used to select a smaller number of representatives from a larger number of samples. It would be useful to remember this use of cluster analysis as well.

<Report assignment>

(1) Using hierarchical cluster analysis and non-hierarchical cluster analysis, classify varieties/lines into “5 groups” based on rice phenotypic data data.tr (page 16). For hierarchical cluster analysis, classify based on the definition of distance between several clusters. Also, use a crosstabulation table to find out how these relate to subpopulations.

(2) From the 229 varieties / lines contained in data.mk2 (page 24) using the k-medoids method, let us select 48 representative varieties / lines based on the mutations found in the DNA markers. Also, perform principal component analysis based on the DNA marker, and let us illustrate the genetic variation of the selected cultivar and strain as a scatter plot on the principal component axis.

(3) For the varieties / lines selected in (2), use the table function to find out how many samples belonging to each subpopulation (subpop) are selected. In addition, for the cultivar / line selected in (2), draw Figure 19 (Histogram of the variation of traits of all cultivars / line and selected cultivar / line), and find out how much it is represented by the varieties and lines selected in).

Submission method:

- Prepare a report as a pdf file and submit it as an email attachment.
- Send an e-mail to "report@iu.a.u-tokyo.ac.jp".
- At the beginning of the report, please do not forget to write your affiliation, student number, and name.
- Deadline for submission is 14 June 2019.