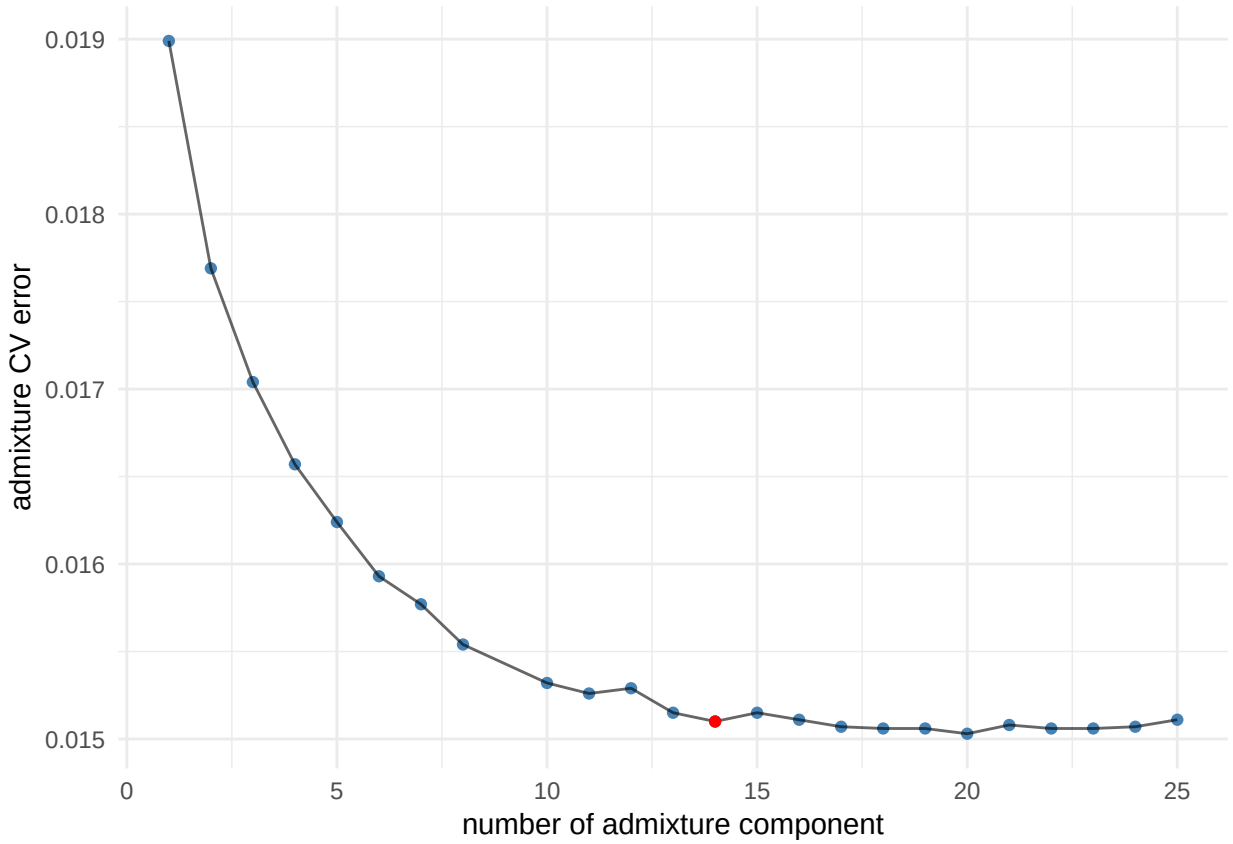
Soil and climate model for predict admixture klass of arabidopsis

Admixture components visualisation

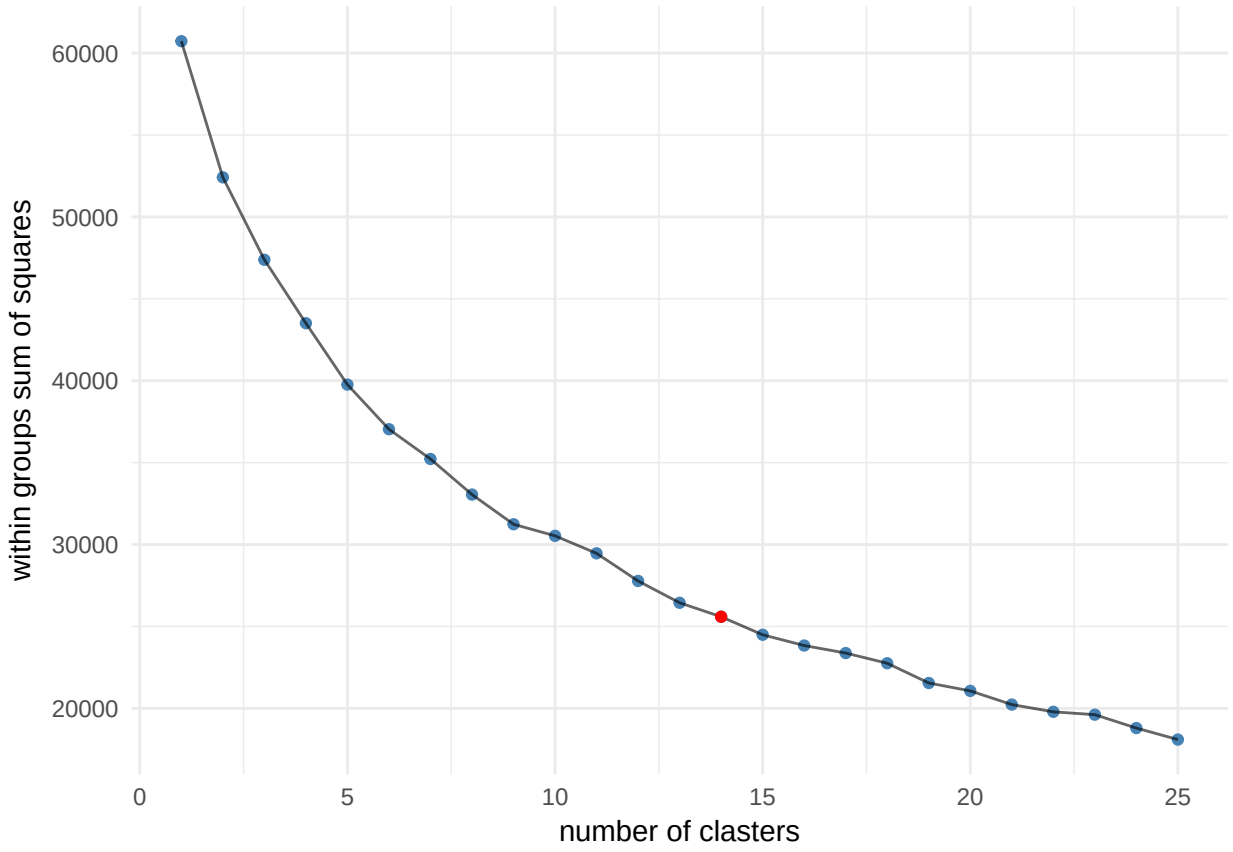
Admixture cross validation

```
cross.validation <- data.frame(
    k = c(1:25)[-9],
    `CV error` = c(0.01899, 0.01769, 0.01704, 0.01657,
                   0.01624, 0.01593, 0.01577, 0.01554,
                   0.01532, 0.01526, 0.01529, 0.01515,
                   0.01510, 0.01515, 0.01511, 0.01507,
                   0.01506, 0.01506, 0.01503, 0.01508,
                   0.01506, 0.01506, 0.01507, 0.01511)
)

ggplot(cross.validation, aes(x = k, y =`CV.error`)) +
    geom_point(col = "steelblue") +
    theme_minimal() +
    geom_line(aes(x = k, y = `CV.error`), alpha = 0.6) +
    xlab("number of admixture component") +
    ylab("admixture CV error") +
    geom_point(data = data.frame(k = 14, `CV error` = 0.01510), colour="red")
```



Predict klasters base on soil and klimate data.

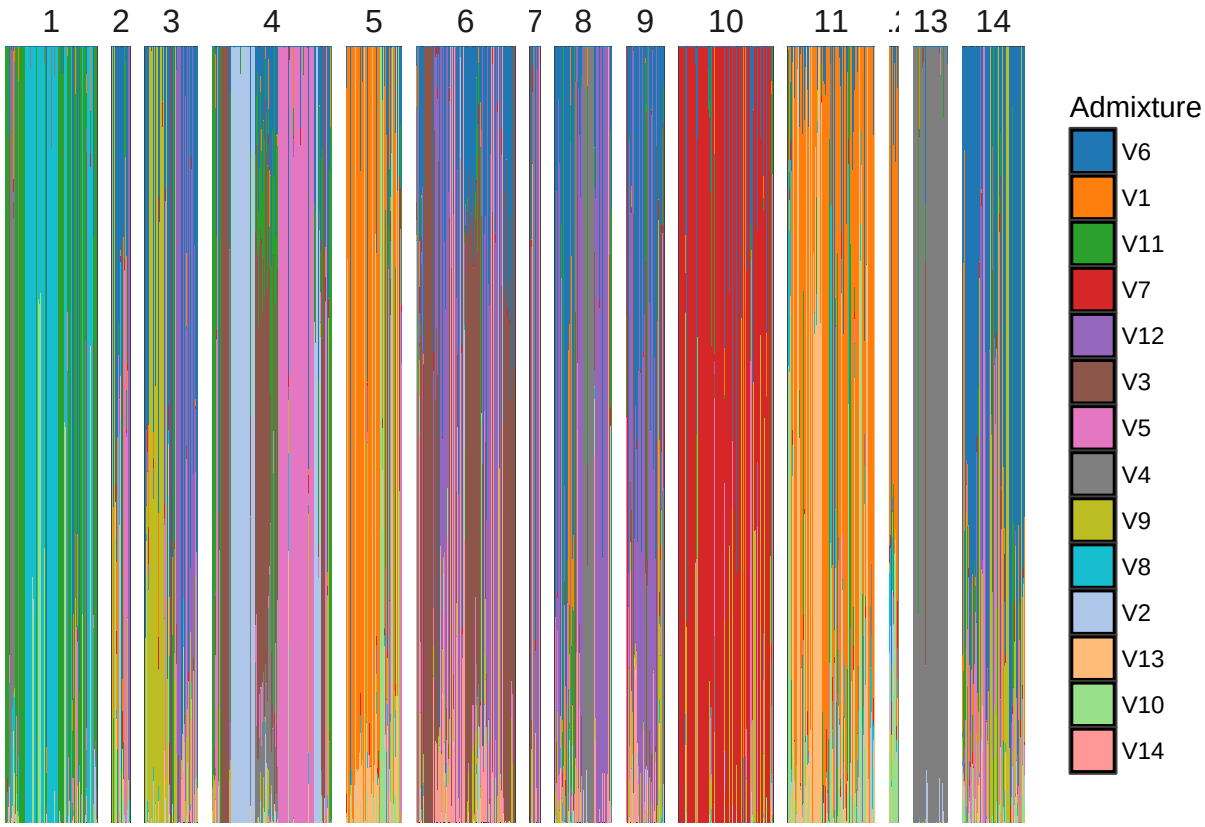
```
just.parametrs <- scale(just.parametrs)
# Determine number of clusters
wss <- rep(0, 25)
for (j in 1:15) {
    wss_ <- (nrow(just.parametrs) - 1) * sum(apply(just.parametrs, 2, var))
    for (i in 2:25) {
        wss_[i] <- sum(kmeans(just.parametrs, centers=i)$withinss)
    }
    wss <- wss + wss_
}
wss <- wss / 15

ggplot(data.frame(id = 1:25, wss = wss), aes(x = id, y = wss)) +
    geom_point(col = "steelblue") +
    theme_minimal() +
    geom_line(aes(x = id, y = wss), alpha = 0.6) +
    xlab("number of clasters") +
    ylab("within groups sum of squares") +
    geom_point(data = data.frame(id = 14, wss = wss[14]), colour="red")
```



Admixture barplot

```
set.seed(1)

cluster.count = 14
just.parametrs <- all_data[,-c(1:(adm.count + 3))]
just.parametrs <- scale(just.parametrs)

fit <- kmeans(just.parametrs, cluster.count)
just.parametrs <- data.frame(just.parametrs, cluster = fit$cluster)

admixture.and.clasters <- data.frame(all_data[, 1:adm.count], cluster = just.parametrs$cluster)
admixture.and.clasters <- admixture.and.clasters[order(admixture.and.clasters$cluster),]

df <- admixture.and.clasters[, 1:adm.count]

df$population = admixture.and.clasters$cluster
df$subject_id = paste("id", 1:nrow(df), sep="")

mdat = melt(df, id.vars=c("subject_id", "population"),
            variable.name="Admixture", value.name="Fraction")

mdat$Admixture = factor(mdat$Admixture,
                        levels=names(sort(colSums(df[, 1:adm.count]), decreasing=TRUE)))
```

```
custom.colors = c("#1f77b4", "#ff7f0e", "#2ca02c", "#d62728", "#9467bd",
          "#8c564b", "#e377c2", "#7f7f7f", "#bcbd22", "#17becf",
          "#aec7e8", "#ffbb78", "#98df8a", "#ff9896")
names(custom.colors) = levels(mdat$Admixture)

ggplot(mdat, aes(x=subject_id, y=Fraction, fill=Admixture, order=Admixture)) +
    geom_col(width = 1) +
    facet_grid(. ~ population, drop=TRUE, space="free", scales="free") +
    theme(panel.grid=element_blank()) +
    theme(panel.background=element_rect(fill=NA, colour="grey25")) +
    theme(axis.title.x=element_blank()) +
    theme(axis.text.x=element_blank()) +
    theme(axis.ticks.x=element_blank()) +
    theme(strip.background=element_blank()) +
    theme(strip.text=element_text(size=12)) +
    theme(legend.key=element_rect(colour="grey25")) +
    scale_x_discrete(expand=c(0, 0)) +
    scale_y_continuous(expand=c(0, 0)) +
    scale_fill_manual(values = custom.colors) +
    guides(fill=guide_legend(override.aes=list(colour=NULL))) +
    ylab(NULL) +
      theme(axis.title.y = element_blank(),
        axis.text.y = element_blank(),
        axis.ticks.y = element_blank())
```



Group metrics

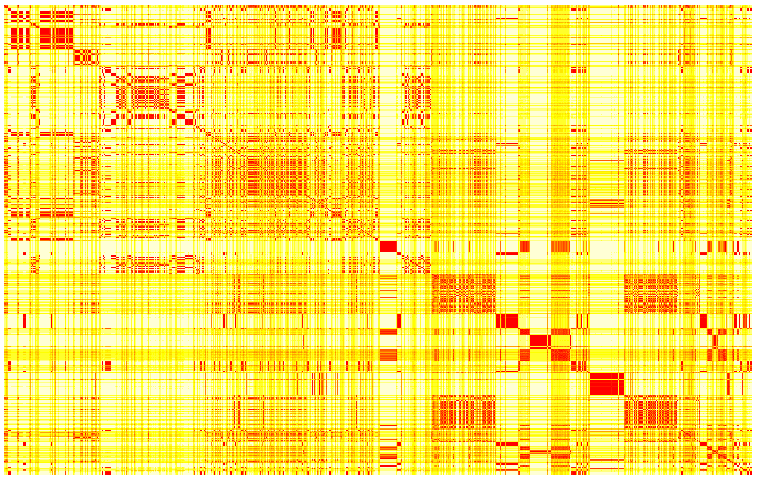
This metrics can give understanding how similar we predict groups base on environment with groups based on admixture.

```
metrics <- lapply(1:cluster.count, function(cluster.number) {
    cluster.subset <- subset.data.frame(df, population == cluster.number)
    max(colSums(cluster.subset[,1:adm.count])) / nrow(cluster.subset)
})
unlist(metrics)
```

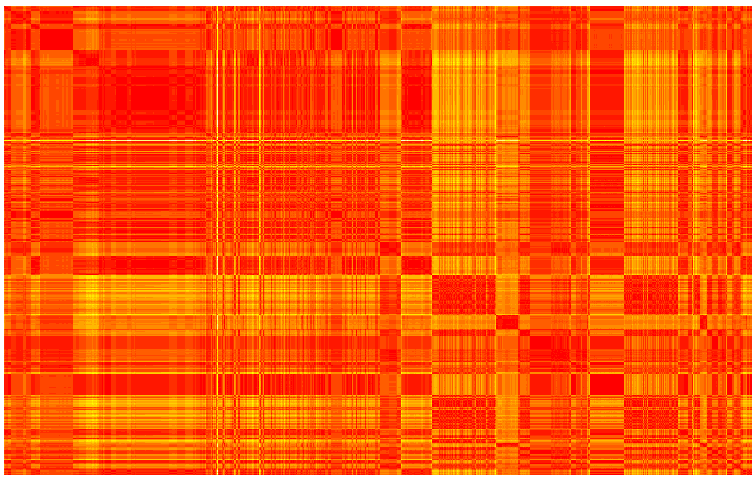
```
##  [1] 0.4043470 0.3431238 0.3380788 0.3278287 0.5375775 0.3700436 0.3642322
##  [8] 0.3360332 0.4000593 0.7439718 0.5072440 0.5046123 0.9389331 0.4815323
```

Mantel test

```
genetic.distance.matrix <- dist(all_data[, 1:adm.count])
geography.distance.matrix <-dist(all_data[,-c(1:(adm.count + 3))])
heatmap.2(as.matrix(genetic.distance.matrix),
          dendrogram = 'none',
          Rowv = FALSE,
          Colv = FALSE,
          trace='none',
          labRow = FALSE,
          labCol = FALSE,
          key = FALSE)
```



```
heatmap.2(as.matrix(geography.distance.matrix),
          dendrogram = 'none',
          Rowv = FALSE,
          Colv = FALSE,
          trace='none',
          labRow = FALSE,
          labCol = FALSE,
          key = FALSE)
```



```
(RoughCorrels <- mantel.rtest(genetic.distance.matrix, geography.distance.matrix, nrepet = 100))
```

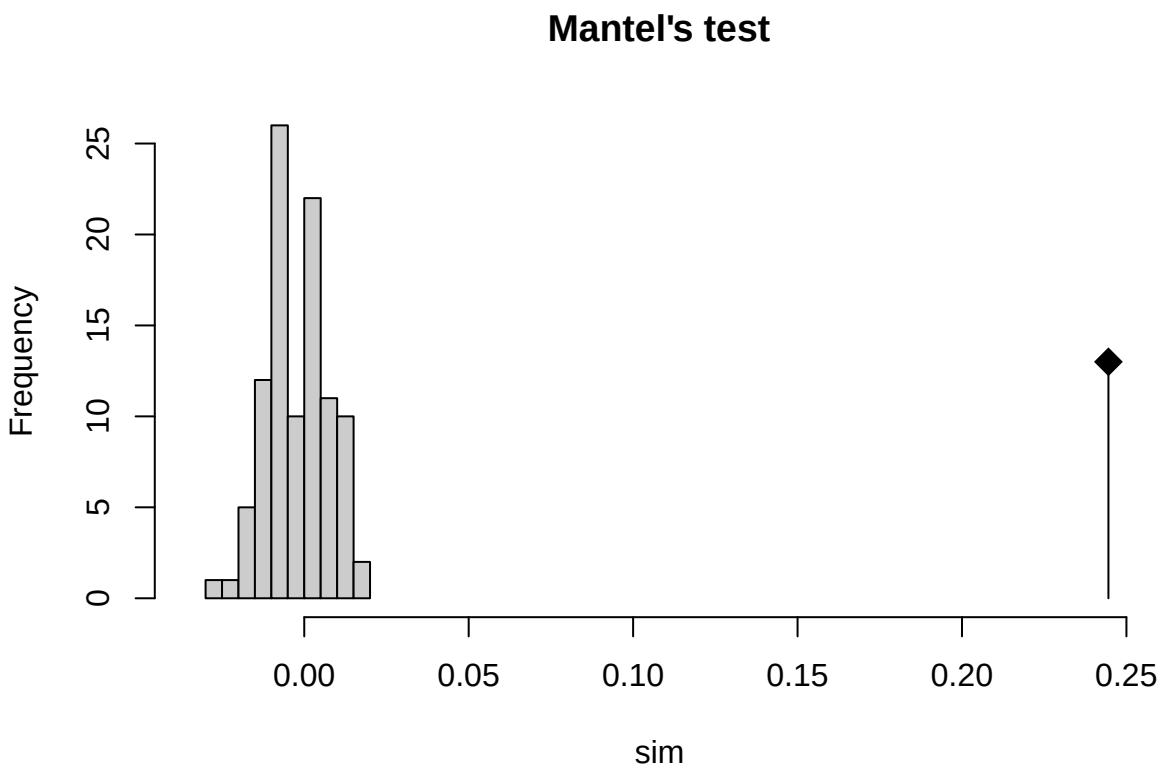
```
## Warning in is.euclid(m1): Zero distance(s)
```

```
## Warning in is.euclid(m2): Zero distance(s)
```

```
## Warning in is.euclid(distmat): Zero distance(s)
```

```
## Monte-Carlo test
## Call: mantel.rtest(m1 = genetic.distance.matrix, m2 = geography.distance.matrix,
##     nrepet = 100)
##
## Observation: 0.2445221
##
## Based on 100 replicates
## Simulated p-value: 0.00990099
## Alternative hypothesis: greater
##
##       Std.Obs   Expectation      Variance
##  2.637797e+01 -2.153254e-03  8.745181e-05
```

```
plot(RoughCorrels, main = "Mantel's test")
```



Random forest based on all variables

Match admixture group for every sample base on the next simple rule, if component number N in vector > 0.7, then this sample belong to the group N.

```
clear.just.parametrs <- just.parametrs

lbound <- .7
groups <- apply(all_data[, c(1:adm.count)], 1, function(current.row) {
    group <- ((current.row > lbound) * c(1:adm.count))[current.row > lbound]
    if (length(group) == 0) {
        return(NA)
    }
    return(group)
})
str(groups)
```

```
##  int [1:1048] 6 6 6 NA NA NA 11 11 11 11 ...
```

```
clear.just.parametrs <- just.parametrs
prepared.data <- data.frame(group = groups)
prepared.data <- na.omit(cbind(prepared.data, clear.just.parametrs))
prepared.data$group <- factor(prepared.data$group)
ncol(prepared.data)
```

```
## [1] 60
```

```
set.seed(666)
```

```
train.indexes <- sample(1:nrow(prepared.data), 0.8 * nrow(prepared.data))
train <- prepared.data[train.indexes, ]
test <- prepared.data[-train.indexes, ]

res.forest <- randomForest(group ~ ., data = train)
t_pred.forest <- predict(res.forest, test)
confMat <- table(test$group, t_pred.forest)
sum(diag(confMat)) / sum(confMat)
```

```
## [1] 0.8591549
```

```
important.dot.plot(res.forest)
```