Analyse exploratoire des données et clustering

Serigne Fallou MBacke NGOM

2023-11-15

Nous allons analyser la structure de corrélation au sein d'un ensemble de données transcriptomiques. Les étapes prévues sont les suivantes :

1. **Chargement et Exploration des Données :** Nous procéderons à l'importation des mesures transcriptomiques de deux tissus de souris, suivie d'une exploration succincte pour mieux appréhender la nature des données.
2. **Analyse de Corrélation avec pairs() :** À l'aide de la fonction pairs(), nous examinerons la structure de corrélation entre les variables, permettant ainsi une visualisation efficace des relations entre les gènes.
3. **Application de k-means et Clustering Hiérarchique :** Nous évaluerons l'impact de l'algorithme k-means et du clustering hiérarchique sur les données, cherchant à identifier des groupes intrinsèques au sein de l'échantillon.
4. **Optimisation du Nombre de Clusters avec clValid :** Nous utiliserons l'outil clValid pour déterminer de manière objective le nombre optimal de clusters, contribuant ainsi à une segmentation pertinente des groupes au sein de l'ensemble de données.

Cette approche méthodique vise à fournir des résultats rigoureux dans l'exploration de la corrélation au niveau transcriptomique, s'appuyant sur des méthodes de clustering bien établies pour dévoiler la structure sous-jacente des données.

Charger les données

- Données intégré au package clValid.
- 147 gènes et étiquettes de séquence exprimées dans deux lignées de souris en développement : les cellules de la crête neurale et les cellules dérivées du mésoderme.
- Trois échantillons par groupe.

```
str(mouse)    # Voir les types de variables dans le jeux de donnees
```

```
## 'data.frame':    147 obs. of  8 variables:
##  $ ID : Factor w/ 147 levels "1415787_at","1415904_at",..: 111 88 93 74 138 103 46 114 112 24 ...
##  $ M1 : num  4.71 3.87 2.88 5.33 5.37 ...
##  $ M2 : num  4.53 4.05 3.38 5.5 4.55 ...
##  $ M3 : num  4.33 3.47 3.24 5.63 5.7 ...
##  $ NC1: num  5.57 5 3.88 6.8 6.41 ...
##  $ NC2: num  6.92 5.06 4.46 6.54 6.31 ...
##  $ NC3: num  7.35 5.18 4.85 6.62 6.2 ...
##  $ FC : Factor w/ 9 levels "ECM/Receptors",..: 3 8 6 6 1 3 1 6 5 6 ...
```

```
head(mouse)
```

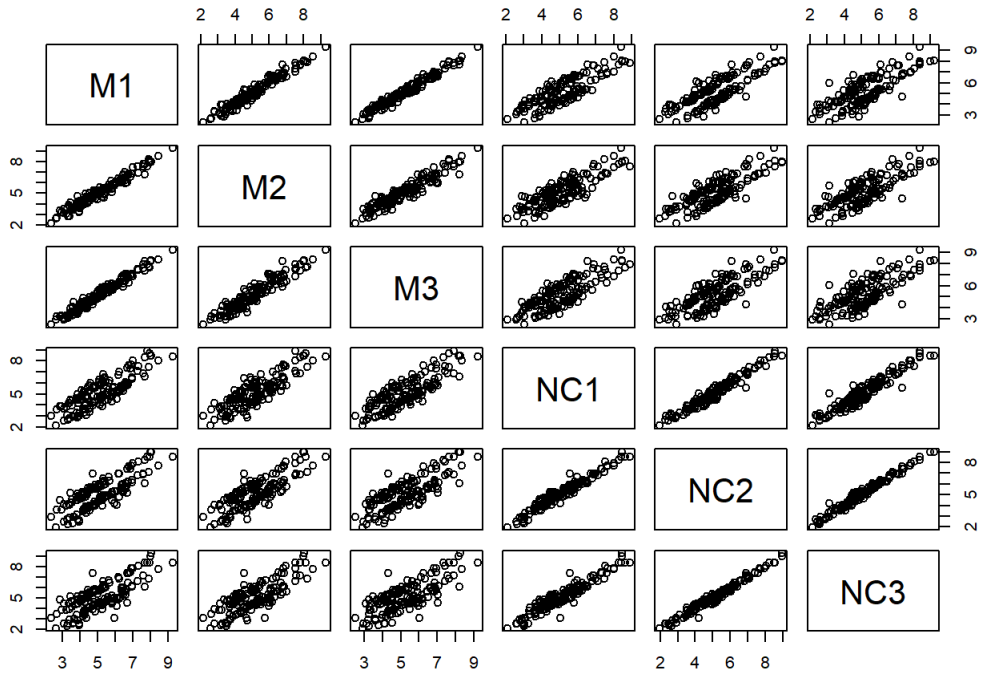
```
##            ID       M1       M2       M3      NC1      NC2      NC3
## 1   1448995_at 4.706812 4.528291 4.325836 5.568435 6.915079 7.353144
## 2 1436392_s_at 3.867962 4.052354 3.474651 4.995836 5.056199 5.183585
## 3 1437434_a_at 2.875112 3.379619 3.239800 3.877053 4.459629 4.850978
## 4   1428922_at 5.326943 5.498930 5.629814 6.795194 6.535522 6.622577
## 5 1452671_s_at 5.370125 4.546810 5.704810 6.407555 6.310487 6.195847
## 6   1448147_at 3.471347 4.129992 3.964431 4.474737 5.185631 5.177967
##                       FC
## 1 Growth/Differentiation
## 2   Transcription factor
## 3          Miscellaneous
## 4          Miscellaneous
## 5          ECM/Receptors
## 6 Growth/Differentiation
```

```
summary(mouse)
```

```
##          ID          M1              M2              M3
##  1415787_at: 1   Min.   :2.352   Min.   :2.139   Min.   :2.500
##  1415904_at: 1   1st Qu.:4.188   1st Qu.:4.151   1st Qu.:4.207
##  1415993_at: 1   Median :4.994   Median :5.043   Median :5.054
##  1416164_at: 1   Mean   :5.166   Mean   :5.140   Mean   :5.231
##  1416181_at: 1   3rd Qu.:6.147   3rd Qu.:6.015   3rd Qu.:6.129
##  1416221_at: 1   Max.   :9.282   Max.   :9.273   Max.   :9.228
##  (Other)   :141
##       NC1             NC2             NC3                          FC
##  Min.   :2.100   Min.   :1.996   Min.   :2.125   EST                      :31
##  1st Qu.:4.174   1st Qu.:4.136   1st Qu.:4.293   Transcription factor     :28
##  Median :4.996   Median :5.056   Median :4.974   Miscellaneous            :25
##  Mean   :5.120   Mean   :5.134   Mean   :5.118   ECM/Receptors            :16
##  3rd Qu.:5.860   3rd Qu.:5.920   3rd Qu.:5.826   Growth/Differentiation:16
##  Max.   :8.905   Max.   :8.954   Max.   :9.251   Unknown                  :10
##                                                  (Other)                  :21
```

Le résumé fournit des informations utiles sur la distribution des variables. Notons que FC est une variable catégorielle.

```
mouse_exp = mouse[,c("M1","M2","M3","NC1","NC2","NC3")]
pairs(mouse_exp)
```



Corrélations, distances et clustering

Nous avons six échantillons, la matrice de corrélation devrait donc être 6x6.

```
library(corrplot)
```

```
## Warning: le package 'corrplot' a été compilé avec la version R 4.2.3
```

```
## corrplot 0.92 loaded
```

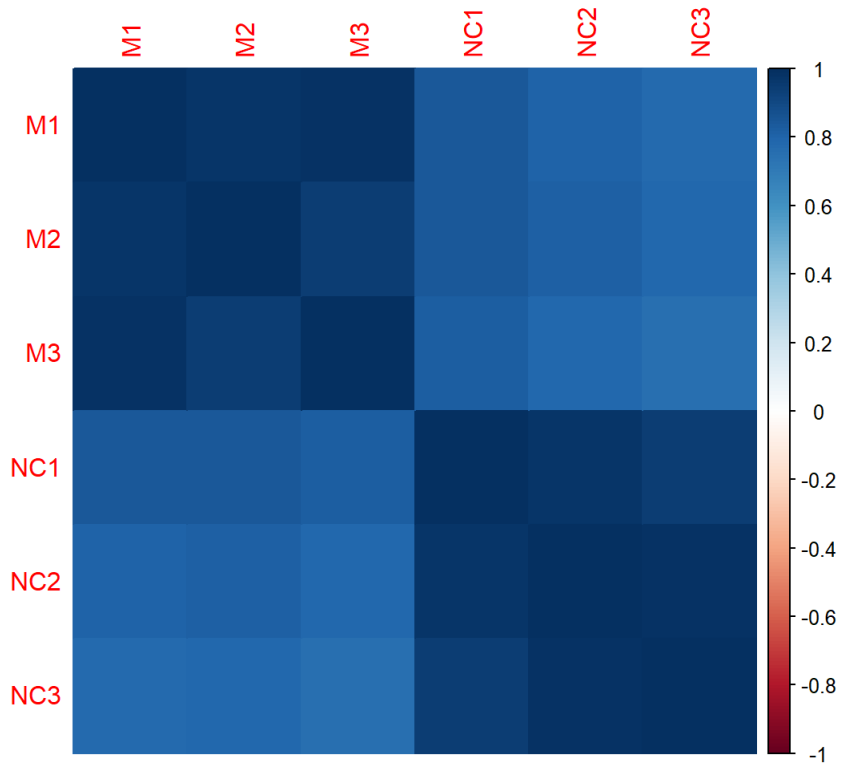
```
mouse_cor <- cor(mouse_exp)
dim(mouse_cor)
```

```
## [1] 6 6
```

```
round(mouse_cor,2)
```

```
##       M1   M2   M3 NC1 NC2 NC3
## M1  1.00 0.98 0.98 0.84 0.81 0.78
## M2  0.98 1.00 0.95 0.84 0.81 0.79
## M3  0.98 0.95 1.00 0.82 0.78 0.75
## NC1 0.84 0.84 0.82 1.00 0.97 0.95
## NC2 0.81 0.81 0.78 0.97 1.00 0.99
## NC3 0.78 0.79 0.75 0.95 0.99 1.00
```

```
corrplot(mouse_cor, method="color")
```

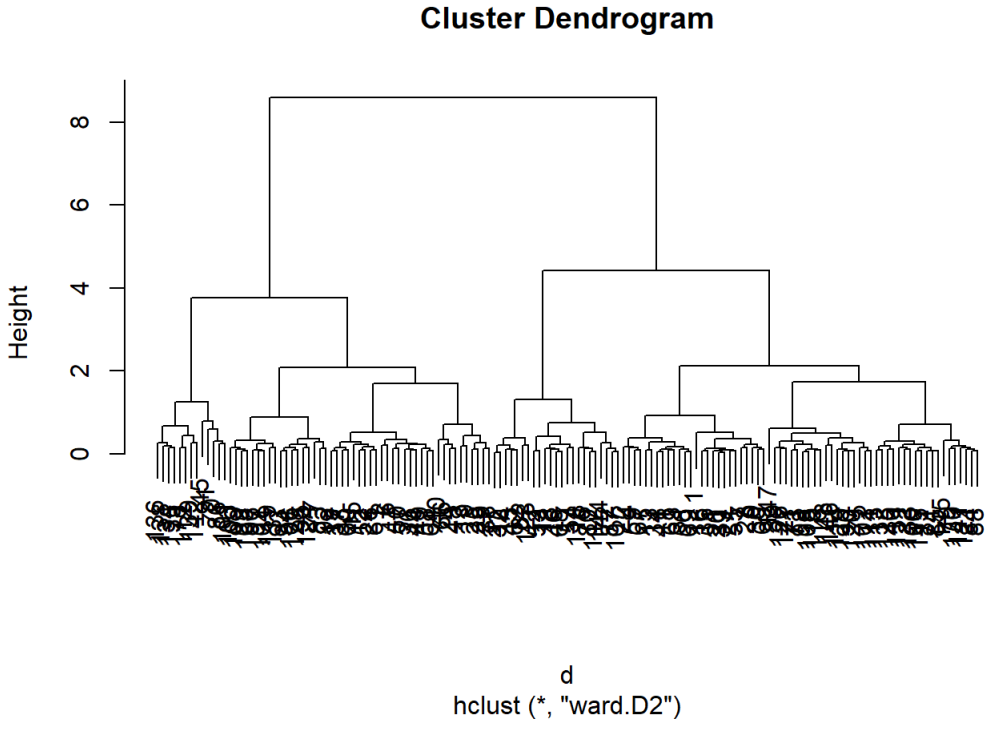


Classification hiérarchique

Le regroupement hiérarchique nécessite des distances entre les échantillons. Utilisons dist()pour calculer ces distances et hclust()générer l'objet de clustering hiérarchique.

Différentes valeurs pour methodpeuvent produire des résultats différents. En effet complete et ward.D2 produisent des résultats stables.

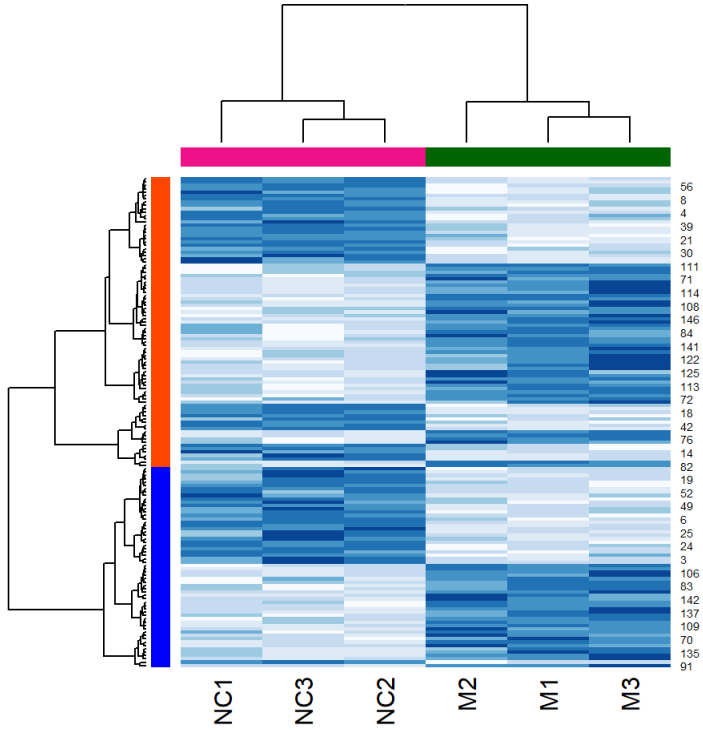
```
d <- dist(log(mouse_exp))
h <- hclust(d,method="ward.D2")
plot(h)
```



Ajoutons maintenant une carte thermique à ce dendrogramme, afin que nous puissions voir les valeurs des gènes dans chaque cluster. Pour cela, nous utiliserons la heatmap()fonction, qui nécessite l'attribution d'étiquettes de cluster à chaque échantillon, ainsi qu'une fonction génératrice de dendrogrammes.

Nous obtenons des affectations de clusters en « coupant » le dendrogramme en deux clusters (ce que nous attendons de notre conception expérimentale). Nous utilisons cutree()pour cela.

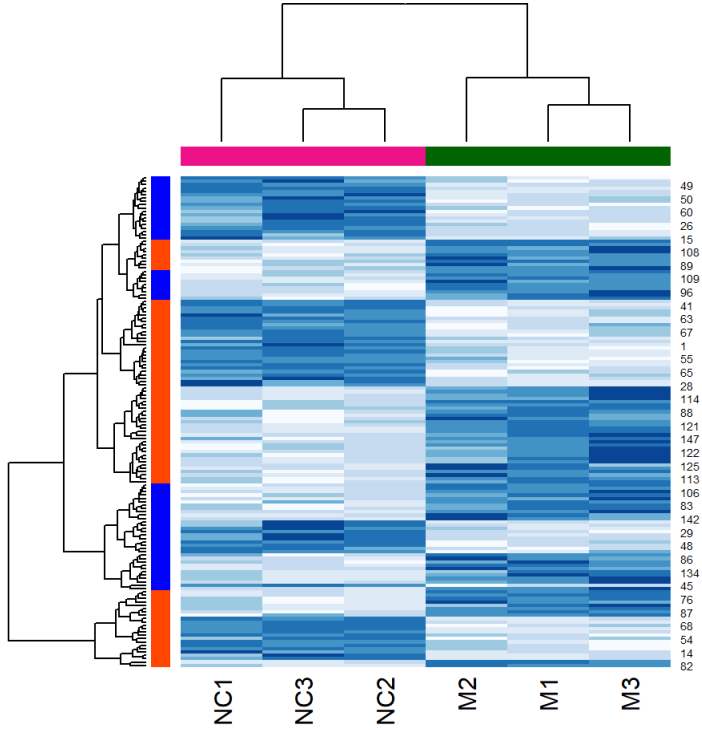
```
library("RColorBrewer")

h2 <- cutree(h, k = 2)
h2cols <- c("orangered","blue")[h2]

hclust_fun <- function(x){
   f <- hclust(x, method = "ward.D2");
   return(f)
}

heatmap(
   as.matrix(mouse_exp),
   col = brewer.pal("Blues",n=8),
   hclustfun = hclust_fun,
   RowSideColors = h2cols,
   ColSideColors = c(
      rep("darkgreen",3),
      rep("deeppink2",3)
   )
)
```



Notez que les deux couleurs sont complètement divisées (c'est-à-dire qu'il n'y a pas de rouge et de bleu entrecoupés). C'est un bon signe !

Comparez ce résultat à ce qui se passe si nous essayons le même appel de fonction sans clustering :

```
heatmap(
   as.matrix(mouse_exp),
   col = brewer.pal("Blues",n=8),
   RowSideColors = h2cols, # use colours from cutree call above
   ColSideColors = c(
      rep("darkgreen",3),
      rep("deeppink2",3)
   )
)
```



K-means clustering

Essayons d'utiliser le clustering k-means, en demandant trois clusters :

```
kclust <- kmeans(
   mouse_exp,
   centers = 3
)
kclust
```

```
## K-means clustering with 3 clusters of sizes 64, 61, 22
##
## Cluster means:
##        M1       M2       M3      NC1      NC2      NC3
## 1 5.553148 5.499583 5.642404 5.426931 5.435363 5.353342
## 2 3.947440 3.946048 4.012209 3.922949 3.950984 4.004362
## 3 7.416536 7.406216 7.414799 7.548674 7.534414 7.520608
##
## Clustering vector:
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20
##   1   2   2   1   1   2   1   1   1   2   3   2   1   3   2   1   3   3   2   1
##  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40
##   1   1   2   2   2   2   3   1   2   1   1   3   2   1   1   2   2   2   1   2
##  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60
##   1   3   3   2   2   3   2   2   2   2   1   2   3   3   1   1   2   2   1   2
##  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80
##   2   3   1   2   1   2   1   3   1   2   1   1   1   1   1   3   2   2   1   3
##  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100
##   1   3   2   1   1   2   3   1   2   1   2   2   2   2   2   2   2   1   2   1
## 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120
##   2   1   2   2   2   2   3   1   2   1   1   2   1   1   1   1   3   3   2   1
## 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140
##   1   1   3   2   1   1   1   2   1   1   1   1   1   2   2   2   2   2   2   1
## 141 142 143 144 145 146 147
##   1   2   1   3   1   1   1
##
## Within cluster sum of squares by cluster:
## [1] 166.24343 193.59229  81.44264
##  (between_SS / total_SS =  74.3 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

Utilisation clValidpour déterminer le nombre de clusters

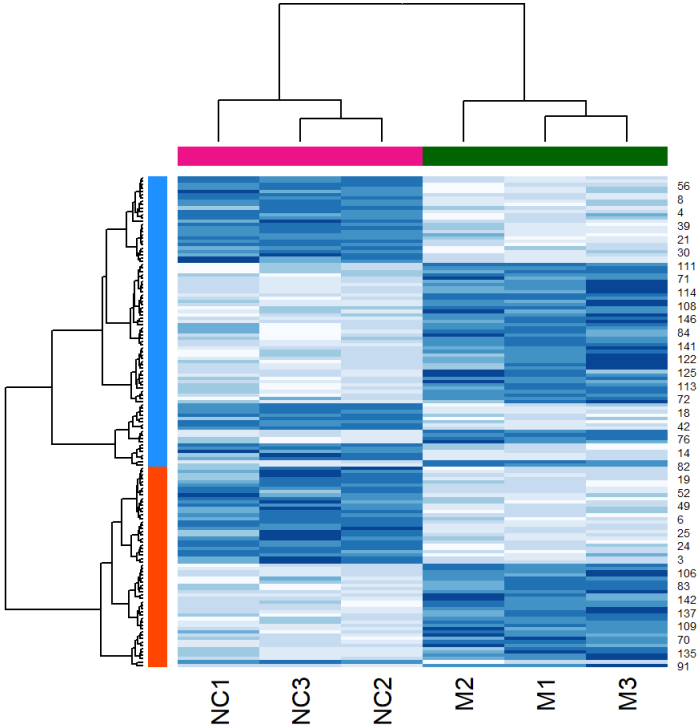
Utilisez la clValid()fonction pour valider les clusters à l'aide de : - Indice Dunn; - scores de silhouette; - connectivité.

```
validation_data <- clValid(
    mouse_exp,
    2:6, # num. clusters to evaluate
    clMethods = c("hier","kmeans"), # methods to eval.
    validation = "internal"
)

summary(validation_data)
```

```
##
## Clustering Methods:
##  hierarchical kmeans
##
## Cluster sizes:
##  2 3 4 5 6
##
## Validation Measures:
##                              2      3      4      5      6
##
## hierarchical Connectivity   5.3270 14.2528 20.7520 27.0726 30.6194
##              Dunn           0.1291  0.0788  0.0857  0.0899  0.0899
##              Silhouette     0.5133  0.4195  0.3700  0.3343  0.3233
## kmeans       Connectivity  13.2548 17.6651 37.3980 43.2655 50.6095
##              Dunn           0.0464  0.0873  0.0777  0.0815  0.0703
##              Silhouette     0.4571  0.4182  0.3615  0.3367  0.3207
##
## Optimal Scores:
##
##              Score  Method       Clusters
## Connectivity 5.3270 hierarchical 2
## Dunn         0.1291 hierarchical 2
## Silhouette   0.5133 hierarchical 2
```

Toutes les mesures de clustering indiquent systématiquement que deux clusters correspondent le mieux aux données.

```
d <- dist(log(mouse_exp))
h <- hclust(d,method="ward.D2")
cluster_ids <- cutree(h, k = 2)
clust_colors <- c("dodgerblue","orangered")[cluster_ids]

heatmap(
    as.matrix(mouse_exp),
    col = brewer.pal("Blues",n=8),
    hclustfun = hclust_fun,
    RowSideColors = clust_colors, # kmeans labels
    ColSideColors = c(
        rep("darkgreen",3),
        rep("deeppink2",3)
    )
)
```


56
8
4
39
21
30
111
71
114
108
146
84
141
122
125
113
72
18
42
76
14
82
19
52
49
6
25
24
3
106
83
142
137
109
70
135
91
NC1
NC3
NC2
M2
M1
M3