TD2 : Sélection par tests multiples

M2 MMA - Université de Paris

2020-2021

Exercice 1 : Tests Multiples sur un jeu de données simulées

Description du jeu de données

Charger le jeu de données `genes.Rdata` disponible sur Moodle à l'aide des commandes suivantes :

```
load("genes.Rdata")
data <- genes$data
condition <- genes$condition
statut <- genes$statut
```

Ce jeu de données porte sur 5000 gènes et deux conditions (variable `condition`) telles que l'on dispose de 30 réplicats dans chacune d'elle. On suppose l'indépendance entre tous les gènes et tous les réplicats.

Dans la 1ière condition, chacun des gènes a une expression que nous qualifierons de *normale*, qui suit une loi $\mathcal{N}(500, 100)$. Dans la 2nde condition, chaque gène a une probabilité de 0.01 d'être *sur-exprimé* et une probabilité de 0.01 d'être *sous-exprimé*. Suivant s'il est sous-exprimé, normal ou sur-exprimé (-1, 0 ou 1 dans la variable `statut`), son expression dans la condition 2 suit une loi $\mathcal{N}(400, 100)$, $\mathcal{N}(500, 100)$ ou $\mathcal{N}(600, 100)$.

On notera que ce jeu de données est une simplification extrême de la réalité :

- toutes les moyennes et variances sont égales pour un état donné,
- la sur ou sous-expression touche tous les gènes de la même façon,
- le changement d'expression n'affecte que la moyenne alors qu'il peut en réalité affecter uniquement la variance, ou les deux,
- les expressions de gènes sont supposées indépendantes entre elles (niant l'existence de régulations).

Ce jeu simplifié sera cependant suffisant ici pour illustrer la problématique des tests multiples.

Questions

1. Dans un premier temps, on ne s'intéresse qu'au gène 1. Indiquer quel test peut permettre de décider s'il est différentiellement exprimé.
2. Appliquer l'un de ces tests au niveau de risque 5% à l'ensemble des gènes du jeu de données puis examiner la liste des gènes retenus. Que peut-on en dire? Indiquer en particulier le nombre de faux positifs, c'est-à-dire le nombre de gènes qui ne sont pas différentiellement exprimés mais qui sont sélectionnés en tant que tels par la procédure de test.
3. Pour traiter le problème de la surabondance de faux positifs, nous nous intéressons à la procédure de Bonferroni, qui permet de contrôler le FWER en effectuant chaque test avec un risque de première

espèce de α/m (voir cours). Montrer qu'appliquer la procédure de Bonferroni permet d'assurer que :

$$\text{FWER} \leq \alpha,$$

peu importe les relations de dépendance entre variables.

Appliquer la procédure de Bonferroni au jeu de données simulées. Commenter.

4. Sous l'hypothèse d'indépendance des variables testées, la procédure de Sidàk consiste à effectuer chaque test avec un risque de première espèce de $1-(1-\alpha)^{1/m}$. Montrer que cette procédure assure que :

$$\text{FWER} \leq \alpha.$$

Montrer cependant que cette procédure est moins conservatrice que celle de Bonferroni. Que peut-on dire si m est grand?

Appliquer la procédure de Sidàk au jeu de données simulées. Commenter.

5. La procédure de Holm-Bonferroni s'applique de la manière suivante :

- Effectuer les m tests et ordonner les m p-valeurs obtenues

$$p_{(1)} \leq ... \leq p_{(m)}.$$

- Déterminer

$$I = \max \left\{ k, \quad \forall i \leq k, \quad p_{(i)} \leq \frac{\alpha}{m-i+1} \right\}.$$

- Rejeter les p-valeurs inférieures à $\frac{\alpha}{m-I+1}$.

Montrer que cette procédure est moins conservatrice que celle de Bonferroni.

Appliquer la procédure de Holm-Bonferroni au jeu de données simulées. Commenter.

6. Pour les valeurs grandes de m, ce qui est souvent le cas en génomique quand on teste un grand nombre de gènes simultanément, le contrôle du FWER entraîne des tests très conservatifs, et possiblement des listes vides de gènes différentiellement exprimés. Une alternative moins conservative consiste à contrôler, non pas le nombre de faux positifs, mais leur proportion parmi les positifs FDP :

$$\text{FDP} = \frac{\text{FP}}{\text{FP} + \text{TP}},$$

ou son espérance :

$$\text{FDR} = \mathbb{E}(FDP).$$

Pour remplir cet objectif, la procédure de Benjamini-Hochberg est la suivante : - Effectuer les m tests et ordonner les m p-valeurs obtenues

$$p_{(1)} \leq p_{(2)} \leq ... \leq p_{(m)}.$$

- Déterminer

$$I = \max \left\{ k, \quad \forall i \leq k, \quad p_{(i)} \leq \alpha \frac{i}{m} \right\}.$$

- Rejeter les p-valeurs inférieures à $\alpha \frac{I}{m}$.

Appliquer la procédure de Benjamini-Hochberg au jeu de données. Déterminer, à l'aide de la variable `statut` donnant le vrai état des gènes, la vraie valeur de la FDR obtenue.

Commenter les résultats en les comparant à ceux obtenus avec les procédures de contrôle de la FWER.

7. Comparer les résultats précédents à ceux obtenus à l'aide de la fonction `p.adjust()` de `R`.

Exercice 2 : Tests Multiples sur un jeu de données réelles

Chargement et description du jeu de données

Charger les données `Golub_Merge` disponibles dans le package `golubEsets` de `Bioconductor`.

```
BiocManager::install("golubEsets")
library(golubEsets)
data("Golub_Merge")
```

Ces données sont extraites de Golub et al. et contiennent des informations concernant 47 patients atteints de leucémie lymphoblastique `ALL` et 25 atteints de leucémie myéloide `AML` (pour un total de $n = 72$ patients donc). On accède aux données d'expression des gènes (pour $p = 7129$ gènes) à l'aide de la commande suivante.

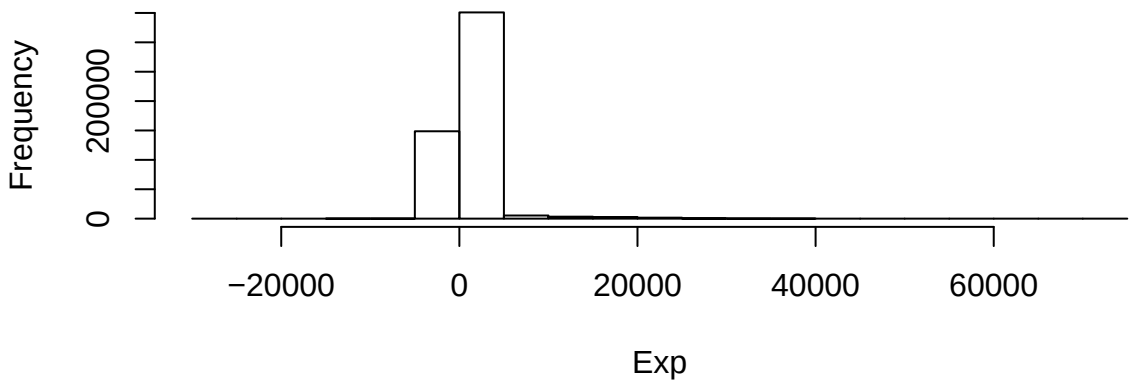
```
Exp <- exprs(Golub_Merge)
dim(Exp)
```

```
## [1] 7129   72
```

Questions

1. En traçant l'histogramme représentant la distribution de l'expression des gènes, nous pouvons nous rendre compte qu'il y a des valeurs aberrantes. Cela vient du fait que le bruit de fond a déjà été enlevé et parfois sur-estimé.

```
hist(Exp, main="")
```



Pour pouvoir traiter ce jeu de données,

- remplacer toutes les valeurs inférieures à 100 par 100 et toutes celles supérieures à 16000 par 16000,
- enlever les gènes dont l'expression n'est pas assez importante ou dont la variance est trop faible. Pour cela, déterminer l'expression maximale *emax* et minimale *emin* de chaque gène et ne garder que les gènes tels que $\frac{emax}{emin} > 5$ et $emax - emin > 500$.
- passer ensuite les données en logarithme en base 10.
- centrer et réduire les données par gène à l'aide de la fonction `scale()`.

2. Effectuer une analyse différentielle entre les deux cancers `ALL` et `AML` à l'aide d'un test de Student et d'une FDR à 1%.

Créer une sous-matrice `DEgenes` des données précédentes en ne gardant que les gènes différentiellement exprimés.

3. Charger la librarie `gplots`. A l'aide de la fonction `heatmap()`, représenter alors l'expression des gènes différentiellement exprimés. Que peut-on dire des conditions `AML` et `ALL`?

4. Garder uniquement les 50 gènes les plus différentiellement exprimés. On se demande si cet ensemble de 50 gènes peut être une bonne signature du type de cancer, c'est-à-dire permet bien de séparer les deux types de conditions.

Pour répondre à cette question, mettre en oeuvre une méthode de classification (par exemple `randomForest`), avec ensemble d'apprentissage et test, et tester son efficacité.