**Code complet du livre et figures en couleur**

Ce document rassemble toutes les instructions écrites en police de caractères "**courier style gras**" du livre. En cliquant sur la page du sommaire ci-dessous, vous accéderez directement au code qui vous intéresse. Les figures en couleur non affichées dans le manuel sont également fournies sous le code correspondant.

Ceux qui ne possèdent pas l'ouvrage peuvent employer ce code, notamment celui du chapitre 5 permettant de réaliser des tests à partir de vos données. Il suffit de récupérer le fichier *mais.txt*, de modifier les parties en grisé et de copier-coller le code dans R.

Attention : tous les tests du chapitre 5 ne sont pas exploitables directement. Certains nécessitent de récupérer du code dans d'autres paragraphes suivant les instructions indiquées dans le livre. Le test de Welch (paragraphe 5.13.3), par exemple, est complet mais dans celui de la comparaison de plusieurs proportions observées (paragraphe 5.11), il manque le code de la vérification des conditions de Cochran qu'il faut récupérer dans le test de χ2 d'homogénéité.

Conseil : (1) vérifier systématiquement qu'aucun message d'erreur n'apparaît lorsque vous copiez-collez du code dans R ; (2) utiliser la fonction "rechercher" de Word pour vous aider à retrouver le code que vous souhaitez récupérer.

Dans l'annexe 18, certaines lignes de code sont en rouge. Elles suppriment tous les objets créés de l'environnement de travail. Par sécurité, elles sont précédées d'un "#" qui évite une utilisation non voulue. Pour obtenir exactement les mêmes résultats que ceux affichés dans l'ouvrage, il est conseillé d'enlever le "#" afin qu'elles soient exécutées, sauf si vous avez des objets importants créés dans l'environnement de travail.

**SOMMAIRE**

CHAPITRE 1 : PRESENTATION DE R 9

1.1. Introduction 9

1.2. Comment obtenir et installer R 9

1.2.1. Installation de R sous Windows et Mac OS 9

1.2.2. Installation de *packages* 9

1.2.3. Récupérer des manuels d'aide. 9

1.3. Découverte 9

1.4. Description des principaux attributs de R 9

1.4.1. L'instruction 9

1.4.2. Les objets 9

1.4.2.1. Les objets de données 9

1.4.2.2. Les fonctions 9

1.4.3. Les opérateurs 9

1.4.4. Les attributs spéciaux 9

1.5. Premiers pas : R est une calculatrice 9

1.6. Manipulation des objets de données 10

1.6.1. Création d'objets de données 10

1.6.1.1 Création par écriture 10

1.6.1.2. Création par importation de fichiers texte 11

1.6.1.3. Création par utilisation du tableur de R 11

1.6.2. Description d'un objet de données 11

1.6.2.1. La fonction length() 11

1.6.2.2. La fonction mode() 11

1.6.2.3. La fonction class() 11

1.6.2.4. La fonction summary() 12

1.6.3. Analyse et modification des données dans un objet de données 12

1.7. Fonctions de statistique descriptive 15

1.8. Manipuler les instructions conditionnelles ou répétées en boucles 15

1.8.1. Attributs de condition 15

1.8.2. Attributs de répétition 15

1.9. R est un éditeur de graphiques 16

1.9.1. Découverte de la fenêtre graphique 16

1.9.2. Différents types de graphique 16

1.9.2.1. Description des fonctions classiques 16

1.9.2.2. Arguments communs de ces fonctions 16

1.9.3. Ajout d'éléments sur un graphique 16

1.9.4. Paramètres graphiques 16

1.9.5. Manipulation des polices de caractères 17

1.9.6. Manipulation des couleurs 17

1.9.7. Tracer plusieurs graphiques côte à côte dans une même fenêtre 28

1.9.8. Gérer plusieurs fenêtres graphiques 28

1.9.9. Exporter un graphique 28

CHAPITRE 2 : NOTIONS DE BASE DE LA STATISTIQUE 29

2.1. Introduction 29

2.2. Définitions de la statistique, de la population et de l'échantillon 29

2.3. L'individu 29

2.4. Les variables aléatoires 29

2.4.1. Définition 29

2.4.2. Deux types de variables aléatoires 29

2.4.2.1. Variable quantitative 29

2.4.2.2. Variable qualitative 29

2.4.3. Particularités de certaines variables qualitatives 29

2.4.3.1. Exclusivité des classes 29

2.4.3.2. Classes appariées 29

2.4.3.3. Cas des variables fixées 29

2.4.4. Observer la distribution des valeurs d'une variable quantitative : l'histogramme 29

2.4.4.1. Choix du nombre de classes 29

2.4.4.2. Fixer l'intervalle des classes 29

2.4.4.3. Ordonnée en effectif, fréquence ou densité 29

2.4.4.4. La fonction hist() de R 29

2.4.5. Observer la distribution des valeurs d'une variable qualitative 30

2.4.6. Limite entre l'aspect quantitatif et qualitatif d'une variable 30

2.5. Les différents types de tableaux de données 30

2.5.1. Cas standard 30

2.5.2. Le tableau disjonctif complet pour les variables qualitatives 30

2.5.3. Le tableau de contingence pour une ou deux variables qualitatives 30

2.6. Avant d'entreprendre toute analyse statistique : la check-list 30

2.7. Les paramètres de statistique descriptive les plus employés 30

2.7.1. La moyenne et la médiane 30

2.7.2. Les quantiles 30

2.7.3. La variance, l'écart type et le coefficient de variation 30

2.7.4. La covariance 30

2.7.5. Le coefficient de corrélation linéaire de Pearson 30

2.8. Exercices 30

2.9. Corrections des exercices 30

CHAPITRE 3 : DEMARCHE SCIENTIFIQUE ET ERREURS ASSOCIEES 32

3.1. Formulation de la question scientifique 32

3.1.1. Décalage entre la question posée et l'approche envisagée 32

3.1.2. Faits supposés avérés 32

3.2. Organisation de l'étude scientifique 32

3.2.1. Individus non semblables 32

3.2.2. Conditions environnementales non semblables 32

3.3. Interprétation du résultat 32

3.4. La gestion des individus extrêmes (*outliers* en anglais) 32

3.5. Conclusion 32

CHAPITRE 4 : LES ETAPES D'UN TEST STATISTIQUE 33

4.1. Introduction à lire avant de se lancer dans ce chapitre 33

4.2. Les deux hypothèses statistiques 33

4.3. La Variable de Test (VT) 33

4.3.1. Définition 33

4.3.2. Différents types de VT 33

4.3.2.1. Tests paramétriques et non paramétriques 33

4.3.2.2. Estimateur et VT 33

4.3.2.3. VT et distribution de probabilité 33

4.4. Distributions de probabilité 33

4.4.1. Définition d'une loi de probabilité 33

4.4.2. Paramètres d'une loi de probabilité 33

4.4.2.1. Cas des variables discrètes : quantile, probabilité et fonction de répartition 33

4.4.2.2. Cas des variables continues : quantile, densité de probabilité et fonction de répartition 33

4.4.2.3. Calcul de la probabilité de voir apparaître une valeur de variable continue 33

4.4.3. Comment utiliser les lois de probabilité avec R 33

4.4.4. Différentes lois de probabilité discrètes 33

4.4.4.1. Loi binomiale 33

4.4.4.2. Loi multinomiale 34

4.4.4.3. Loi de Pascal et loi binomiale négative 34

4.4.4.4. Loi géométrique 34

4.4.4.5. Loi hypergéométrique 35

4.4.4.6. Loi de Poisson 35

4.4.5. Différentes lois de probabilité continues 35

4.4.5.1. Loi normale ou de Laplace-Gauss 35

4.4.5.2. Loi normale centrée réduite 36

4.4.5.3. Loi exponentielle 36

4.4.5.4. Loi gamma 36

4.4.5.5. Loi de χ2 37

4.4.5.6. Loi de Fisher-Snedecor 37

4.4.5.7. Loi de Student 37

4.4.6. Distributions de probabilité qui ne suivent pas de loi connue 38

4.4.6.1. Distribution de probabilité de Mann-Whitney 38

4.4.6.2. Distribution de probabilité de Wilcoxon 38

4.4.6.3. Distribution de probabilité du test des signes de Wilcoxon 39

4.4.7. Rapport entre toutes ces distributions de probabilité 39

4.4.8. Remarques importantes 39

4.4.8.1. Ne pas confondre la loi de probabilité d'une variable mesurée et celle d'une VT 39

4.4.8.2. Simulation avec R de la fluctuation d'une VT due à l'échantillonnage 39

4.4.8.3. Importance du tirage aléatoire des individus dans la formation de l'échantillon 40

4.5. Hypothèse H0, distribution de probabilité de la VT et échantillon : le cocktail magique de l'obtention de la *p value* 40

4.6. Conclusion d'un test statistique et les deux risques d'erreurs associés 40

4.6.1. Conclure, c'est deux vérités, deux décisions soit quatre probabilités 40

4.6.2. L'hypothèse H0 et le risque α : définitions 40

4.6.3. La correction du seuil de rejet α 40

4.6.3.1. Le problème soulevé 40

4.6.3.2. La technique de Bonferroni 40

4.6.3.3. La technique séquentielle (Holm) 41

4.6.3.4. Quand appliquer la correction ? 41

4.6.4. L'hypothèse H1 et son influence sur le risque α 41

4.6.4.1 Le problème de l'hypothèse H1 41

4.6.4.2. Test bilatéral et unilatéral 41

4.6.4.3. Obtenir la *p value* en test bilatéral et unilatéral 41

4.6.4.4. Placer les seuils α de rejet en test bilatéral et unilatéral 41

4.6.4.5. Comment choisir entre test bilatéral et unilatéral ? 41

4.6.5. Le risque β et la puissance 1-β du test 41

4.6.5.1. Retour sur les définitions de β et 1-β 41

4.6.5.2. Variations de β et 1-β suivant la distribution de probabilité de la VT sous H1 41

4.6.6. α et β en termes de faux positifs et faux négatifs 42

4.6.7. Propriétés de la puissance 1-β 42

4.6.7.1 A lire avant de se lancer dans ce paragraphe 42

4.6.7.2. La puissance d'un test diminue quand décroît le αseuil 42

4.6.7.3. La puissance d'un test croît quand augmente l'effectif n de l'échantillon 42

4.6.7.4. La puissance d'un test augmente avec l'écart entre les paramètres testés 43

4.6.8. Le danger de considérer la *p value* comme un indicateur de forte ou faible significativité 43

4.6.9. Alors comment fixer la puissance d'un test ? 43

4.6.9.1. Considérations générales 43

4.6.9.2. Réaliser des abaques 43

4.6.9.3. Les fonctions disponibles sous R 44

4.6.9.4. Le ncp des lois de probabilité de VT sous R 44

4.6.10. Comment conclure finalement ? 45

4.7. Récapitulation 45

4.8. Exercices 45

4.9. Correction des exercices 45

CHAPITRE 5 : LES TESTS STATISTIQUES 48

5.1. A lire absolument avant d'utiliser un test 48

5.2. Quel test appliquer et quelle fonction de R utiliser ? 48

5.3. χ2 de conformité 48

5.3.1. Méthode 48

5.3.2. Exemple avec R 48

5.3.3. Tests de comparaisons deux à deux 50

5.4. χ2 d'homogénéité 51

5.4.1. Méthode 51

5.4.2. Exemples avec R 51

5.4.3. Tests de comparaisons deux à deux 52

5.5. Test G 54

5.5.1. Méthode 54

5.5.2. Exemples avec R 54

5.5.3. Tests de comparaisons deux à deux 54

5.6. Test exact de Fisher 55

5.6.1. Tableau de contingence 2 × 2 55

5.6.1.1. Méthode 55

5.6.1.2. Exemples avec R 55

5.6.2. Tableau de contingence c × k 58

5.6.2.1. Méthode 58

5.6.2.2. Exemple avec R 58

5.6.2.3. Tests de comparaisons deux à deux 58

5.7. Test de Mantel-Haenszel 59

5.7.1. Méthode 59

5.7.2. Exemples avec R 59

5.7.3. Tests de comparaisons deux à deux 60

5.8. Comparaison d'une proportion observée à une proportion théorique 61

5.8.1. Méthode 61

5.8.2. Exemples avec R 61

5.9. Comparaison de deux proportions observées 63

5.9.1. Méthode 63

5.9.2. Exemples avec R 63

5.10. Comparaison de deux proportions en séries appariées (test de Mac Nemar) 65

5.10.1. Méthode 65

5.10.2. Exemples avec R 65

5.11. Comparaison de plusieurs proportions observées 67

5.11.1. Méthode 67

5.11.2. Exemple avec R 67

5.11.3. Tests de comparaisons deux à deux 68

5.12. Comparaison de plusieurs proportions observées à plusieurs proportions théoriques 69

5.12.1. Méthode 69

5.12.2. Exemple avec R 69

5.12.3. Tests de comparaisons deux à deux 70

5.13. Le test t de Student de comparaison de moyennes 71

5.13.1. Comparaison d'une moyenne observée à une valeur théorique 71

5.13.1.1. Méthode 71

5.13.1.2. Exemples avec R 71

5.13.2. Comparaison de deux moyennes observées 73

5.13.2.1. Méthode 73

5.13.2.2. Exemple avec R 73

5.13.3. Comparaison de deux moyennes observées avec variances différentes (test de Welch) 74

5.13.3.1. Méthode 74

5.13.3.2. Exemples avec R 74

5.13.4. Comparaison de deux moyennes observées en séries appariées 76

5.13.4.1. Méthode 76

5.13.4.2. Exemple avec R 76

5.14. Comparaison d'au moins deux moyennes observées 78

5.14.1. Anova (analyse de variances à un facteur) 78

5.14.1.1. Méthode 78

5.14.1.2. Exemple avec R 78

5.14.2. Anova avec variances différentes (correction de Welch) 79

5.14.2.1. Méthode 79

5.14.2.2. Exemple avec R 79

5.14.3. Tests de comparaisons deux à deux 80

5.15. Comparaison d'une médiane observée à une valeur théorique (test des signes de Wilcoxon) 81

5.15.1. Méthode 81

5.15.2. Exemples avec R 81

5.16. Comparaison de deux médianes observées (test de Mann-Whitney-Wilcoxon) 83

5.16.1. Méthode 83

5.16.2. Exemples avec R 83

5.17. Comparaison de deux médianes observées en séries appariées (test des signes de Wilcoxon) 85

5.17.1. Méthode 85

5.17.2. Exemples avec R 85

5.18. Comparaison d'au moins deux médianes observées 87

5.18.1. Test de Kruskal-Wallis 87

5.18.1.1. Méthode 87

5.18.1.2. Exemple avec R 87

5.18.1.3. Tests de comparaisons deux à deux 88

5.18.2. Test des médianes 89

5.18.2.1. Méthode 89

5.18.2.2. Exemple avec R 89

5.18.2.3. Tests de comparaisons deux à deux 89

5.19.1. Test de Fisher-Snedecor 91

5.19.1.1. Méthode 91

5.19.1.2. Exemple avec R 91

3.0233985.19.2. Test d'Ansari-Bradley 92

5.19.2.1. Méthode 93

5.19.2.2. Exemples avec R 93

5.20. Comparaison d'au moins deux variances observées 96

5.20.1. Test de Bartlett 96

5.20.1.1. Méthode 96

5.20.1.2. Exemple avec R 96

5.20.2. Test de Fligner- Killeen 98

5.20.2.1. Méthode 98

5.20.2.2. Exemple avec R 98

5.20.3. Tests de comparaisons deux à deux 99

5.21. Test du coefficient de corrélation linéaire de Pearson 100

5.21.1. Méthode 100

5.21.2. Exemple avec R 100

5.22. Test du coefficient de corrélation de Spearman 102

5.22.1. Méthode 102

5.22.2. Exemples avec R 102

5.23. Test du coefficient de corrélation de Kendall 104

5.23.1. Méthode 104

5.23.2. Exemples avec R 104

5.24. Test de χ2 105

5.24.1. Méthode 105

5.24.2. Exemple avec R 105

5.25. Tests de corrélations multiples 106

5.26. Ajustement d'une distribution observée à une distribution théorique 107

5.26.1. Introduction 107

5.26.2. Test de χ2 de conformité 107

5.26.2.1. Méthode 107

5.26.2.2. Exemple avec R 107

5.26.3. Test de Kolmogorov-Smirnov 109

5.26.3.1. Méthode 109

5.26.3.2. Exemple avec R 109

5.26.4. Test de Shapiro-Wilk 111

5.26.4.1. Méthode 111

5.26.4.2. Exemple avec R 111

5.27. Comparaison de deux distributions observées (test de Kolmogorov-Smirnov) 112

5.27.1. Méthode 112

5.27.2. Exemple avec R 112

5.28. Tests autour de la régression 114

5.28.1. Introduction 114

5.28.2. Principe de la régression linéaire simple 114

5.28.3. Comparaison d'une régression observée à une régression nulle 114

5.28.3.1. Méthode 114

5.28.3.2. Exemple avec R 114

5.28.4. Comparaison d'une régression observée à une régression théorique 116

5.28.4.1. Méthode 116

5.28.4.2. Exemples avec R 116

5.29. Test autour de la survie 118

5.29.1. Introduction 118

5.29.2. Comparaison de deux courbes de survie (test du logrank) 118

5.29.2.1. Méthode 118

5.29.2.2. Exemple avec R 118

ANNEXES 121

1. Formule développée de la variance et de la covariance 121

2. L'estimateur 121

3. Distribution normale de variables mesurées et théorème central limite 121

4. Rappel des moyennes et variances des distributions de probabilité 121

5. Rappel sur les combinaisons 121

6. Passage du χ2 au Z2 dans le cas de la comparaison d'une proportion observée à une proportion théorique 121

7. Passage du χ2 au Z2 dans le cas de la comparaison de deux proportions observées 121

8. Retrouver la formule de la VT à partir de la formule du χ2 dans le cas de la comparaison de plusieurs proportions observées 121

9. Estimation de la fluctuation de la VT χ2 avec correction de continuité de Yates 121

10. Comment se comportent les différents couples de proportions (pG1/F1 , pG1/F2), (pG2/F1 , pG2/F2), (pF1/G1 , pF1/G2) et (pF2/G1 , pF2/G2) lors d'un test exact de Fisher sur tableau de contingence 2 × 2 122

11. Anova et régression linéaire sont liées 122

12. Procédure lorsque la fonction solve() n'est pas utilisable 123

13. Exemples graphiques avec R 123

14. Exécution des codes du chapitre 5 depuis un fichier de type "texte" et exportation des résultats 125

15. Edition des graphiques du chapitre 5 dans un fichier de type "pdf" 125

16. Différences entre les fonctions sort(), rank() et order() 125

17. Précisions sur la médiane et autres quantiles 125

18. Création de fonctions 126

19. Principaux attributs de R 130

**CHAPITRE 1**

**PRESENTATION DE R**

## 1.1. Introduction

## 1.2. Comment obtenir et installer R

### 1.2.1. Installation de R sous Windows et Mac OS

### 1.2.2. Installation de *packages*

### 1.2.3. Récupérer des manuels d'aide.

## 1.3. Découverte

## 1.4. Description des principaux attributs de R

### 1.4.1. L'instruction

### 1.4.2. Les objets

#### 1.4.2.1. Les objets de données

#### 1.4.2.2. Les fonctions

**sum**

**sum(1,4)**

**help(sum)**

### 1.4.3. Les opérateurs

**x<-NULL**

**x<-4**

**ls()**

**?"+"**

### 1.4.4. Les attributs spéciaux

## 1.5. Premiers pas : R est une calculatrice

**10/3**

**25^0.5**

**5:2**

**8\*(4+2^2)/3-pi**

**abs(54-93)/cos(1-0.62)^0.5 \* exp(-8/3) \* factorial(6) + log10(4^(1/3))-2e+2**

## 1.6. Manipulation des objets de données

### 1.6.1. Création d'objets de données

#### 1.6.1.1 Création par écriture

**x<-4**

**x**

**hauteur<-c(130,300,170)**

**hauteur**

**couleur<-c("Rouge","Jaune","Jaune")**

**couleur**

**enracinement<-rep(c("Faible", "Moyen", "Fort"), c(1,3,2))**

**enracinement**

**logique.1<-c(TRUE,FALSE,TRUE)**

**logique.1**

**vec.vide<-vector("character", 5)**

**vec.vide**

**chiffre<-c(1:15)**

**matrice.1<-matrix(chiffre,ncol=3)**

**matrice.1**

**matrice.1<-1:15**

**dim(matrice.1)<-c(5, 3)**

**matrice.1**

**matrice.2<-matrix((1:15)\*100,nrow=3, byrow=TRUE)**

**matrice.2**

**mat.vide<-matrix(nrow=2, ncol=4)**

**mat.vide**

**array.1<-array(letters, dim=c(3,5,2))**

**array.1**

**dataframe.1<-data.frame(matrice.2)**

**dataframe.1**

**hauteur<-c(130,300,170)**

**masse<-c(1431,1468,1398)**

**nb.grains<-c(320,290,147)**

**couleur<-c("Rouge", "Jaune", "Jaune")**

**dataframe.2<-data.frame(hauteur, masse, nb.grains, couleur)**

**dataframe.2**

**mat.vide<-matrix(nrow=2, ncol=4)**

**dataframe.vide<-as.data.frame(mat.vide)**

**dataframe.vide**

**liste.1<-list(x, hauteur, matrice.1, dataframe.2)**

**liste.1**

**liste.vide<-vector("list", 1)**

**liste.vide**

**enracinement<-rep(c("Faible", "Moyen", "Fort"), c(1,3,2))**

**enracinement**

**facteur.1<-factor(enracinement)**

**facteur.1**

**enracinement.2<-factor(c(5, 0, 10, 5, 40, 5, 0), levels=c(0, 5, 10, 40), labels=c("Faible", "Moyen", "Fort", "Tres.fort"))**

**enracinement.2**

**facteur.2<-gl(2, k=3, labels = c("Moyen", "Faible"))**

**facteur.2**

**facteur.3<-gl(2, k=1, length = 6, labels = c("Moyen", "Faible"))**

**facteur.3**

**enracinement<-rep(c("Faible", "Moyen", "Fort"), c(1,3,2))**

**table.1<-table(enracinement)**

**table.1**

**parcelle<-c("Nord", "Nord", "Nord", "Sud", "Nord", "Sud")**

**table.2<-table(enracinement, parcelle)**

**table.2**

**dataframe.3<-data.frame(enracinement, parcelle)**

**dataframe.3**

**table.2<-table(dataframe.3)**

**table.2**

**mat.vide<-matrix(nrow=2, ncol=4)**

**table.vide<-as.table(mat.vide)**

**table.vide**

#### 1.6.1.2. Création par importation de fichiers texte

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt", header=TRUE)**

**mais**

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt")**

**mais**

**mais.vecteur<-scan("C:/Users/Gael/Desktop/mais.txt", what="")**

**mais.vecteur**

**mais.liste<-scan("C:/Users/Gael/Desktop/mais.txt", what=list("","","",""))**

**mais.liste**

#### 1.6.1.3. Création par utilisation du tableur de R

**dataframe.4<-data.frame()**

**fix(dataframe.4)**

**matrice.2<-matrix((1:15)\*100,nrow=3, byrow=TRUE)**

**dataframe.1<-data.frame(matrice.2)**

**dataframe.4<-edit(dataframe.1)**

### 1.6.2. Description d'un objet de données

#### 1.6.2.1. La fonction length()

**length(x)**

**length(hauteur)**

**length(matrice.1)**

**length(array.1)**

**length(dataframe.1)**

**length(liste.1)**

**length(facteur.1)**

**length(table.2)**

#### 1.6.2.2. La fonction mode()

**mode(x)**

**mode(hauteur)**

**mode(logique.1)**

**mode(matrice.1)**

**mode(array.1)**

**mode(dataframe.1)**

**mode(liste.1)**

**mode(facteur.1)**

**mode(table.2)**

**c(pasbondutout, cevecteurilest)**

**c("couleur", "enracinement")**

**c(couleur, enracinement)**

**2\*c(130,300,170)**

**2\*c("130","300","170")**

#### 1.6.2.3. La fonction class()

**class(hauteur)**

**class(couleur)**

**class(logique.1)**

**class(matrice.1)**

**class(array.1)**

**class(dataframe.2)**

**class(liste.1)**

**class(facteur.1)**

**class(table.2)**

#### 1.6.2.4. La fonction summary()

**summary(hauteur)**

**summary(couleur)**

**summary(logique.1)**

**summary(matrice.1)**

**summary(array.1)**

**summary(dataframe.2)**

**summary(liste.1)**

**summary(facteur.1)**

**summary(table.2)**

### 1.6.3. Analyse et modification des données dans un objet de données

**hauteur<-c(130,300,170)**

**enracinement<-rep(c("Faible", "Moyen", "Fort"), c(1,3,2))**

**hauteur/2**

**hauteur<200**

**hauteur[3]**

**enracinement[c(3,5)]**

**hauteur[-1]**

**hauteur[hauteur>150]**

**hauteur[hauteur>=100 & hauteur<=200]**

**sort(hauteur)[1:2]**

**which(hauteur==300)**

**which(enracinement=="Fort")**

**hauteur.2<-hauteur**

**hauteur.2[which(hauteur.2==300)]<-1000**

**hauteur.2**

**names(hauteur) <- c("a", "b", "c")**

**hauteur**

**names(hauteur) <- NULL**

**hauteur**

**matrice.1<-matrix(1:15,ncol=3)**

**matrice.2<-matrix((1:15)\*100,nrow=3, byrow=TRUE)**

**dim(matrice.2)**

**nrow(matrice.2)**

**ncol(matrice.2)**

**matrice.2[3,2]**

**which(matrice.2==900, arr.ind=TRUE)**

**sort(matrice.2, decreasing=TRUE)[1:3]**

**matrice.2[matrice.2>1050]**

**matrice.2[matrice.2>=500 & matrice.2<=900]**

**matrice.2[,2]**

**matrice.2[,2, drop=FALSE]**

**matrice.2[,-2]**

**matrice.2[,-c(2,4)]**

**rbind(matrice.2,c(-2,-1,0,1,2))**

**rbind(matrice.2,rep("a", ncol(matrice.2)))**

**cbind(matrice.2, matrice.1[1:3,2])**

**matrice.2-2**

**sum(matrice.2)**

**apply(matrice.2,1,sum)**

**dimnames(matrice.2)<-list(c("r1","r2","r3"),c("c1","c2","c3","c4","c5"))**

**matrice.2**

**dimnames(matrice.2)**

**dimnames(matrice.2)[[2]][4]<-"Couleur"**

**matrice.2**

**matrice.2["r1","c2"]**

**dimnames(matrice.2)<-NULL**

**matrice.2**

**matrice.3<-matrice.2**

**for (i in 1:ncol(matrice.3)){matrice.3[,i]<- rev(matrice.3 [,i])}**

**matrice.3<-t(matrice.3)**

**image(1:nrow(matrice.3), 1:ncol(matrice.3), matrice.3, main="Matrice.2", col = heat.colors(length(unique(matrice.2))), xlab="", ylab="", xaxt="n", yaxt="n", bty="n")**



**hauteur<-c(130,300,170)**

**masse<-c(1431,1468,1398)**

**nb.grains<-c(320,290,147)**

**couleur<-c("Rouge","Jaune","Jaune")**

**dataframe.2<-data.frame(hauteur, masse, nb.grains, couleur)**

**dim(dataframe.2)**

**names(dataframe.2)**

**names(dataframe.2[2])**

**dimnames(dataframe.2)**

**dataframe.5<- dataframe.2**

**names(dataframe.5)[2]<-"Masse.totale"**

**dataframe.5**

**row.names(dataframe.5)[2]<-"Bob"**

**dataframe.5**

**dataframe.6<-dataframe.5**

**row.names(dataframe.6)<-NULL**

**dataframe.6**

**names(dataframe.6)<-NULL**

**dataframe.6**

**names(dataframe.6)<-c("a","b","c","d")**

**dataframe.6**

**dataframe.2[1,"masse"]**

**dataframe.2[3]**

**dataframe.2[,3]**

**dataframe.5[2,]**

**dataframe.2$hauteur**

**dataframe.2$h**

**dataframe.2$hauteur[c(2,3)]**

**dataframe.2$hauteur[dataframe.2$hauteur > 150 & dataframe.2$hauteur <= 250]**

**dataframe.2$masse[dataframe.2$hauteur > 150]**

**dataframe.2[dataframe.2$hauteur > 150,]**

**subset(dataframe.2, hauteur > 150)**

**which(dataframe.2=="Rouge" | dataframe.2==300, arr.ind=TRUE)**

**sapply(dataframe.2, class)**

**dataframe.5$hauteur<-dataframe.5$hauteur-50**

**dataframe.5**

**dataframe.5$hauteur[c(1,2)]<-c(1000,2000)**

**dataframe.5**

**cbind(dataframe.5, hauteur-100)**

**transform(dataframe.5,soustraction=hauteur-100)**

**dataframe.7<-dataframe.5**

**dataframe.7$Parcelle<-c("Sud", "Nord", "Sud")**

**dataframe.7**

**rbind(dataframe.5, c("a","a","a","a"))**

**c(dataframe.2)**

**x<-4**

**hauteur<-c(130,300,170)**

**matrice.1<-matrix(1:15,ncol=3)**

**masse<-c(1431,1468,1398)**

**nb.grains<-c(320,290,147)**

**couleur<-c("Rouge","Jaune","Jaune")**

**dataframe.2<-data.frame(hauteur, masse, nb.grains, couleur)**

**liste.1<-list(x, hauteur, matrice.1, dataframe.2)**

**names(liste.1)<-c("a","b","c","d")**

**liste.1**

**liste.1[[2]]**

**liste.1[2]**

**c(liste.1[2], liste.1[4])**

**liste.1[[4]][,3][2]**

**which(liste.1[[3]]==9,arr.ind=TRUE)**

**names(liste.1)<-NULL**

**liste.1**

**enracinement.2<-factor(c(5, 0, 10, 5, 40, 5, 0), levels=c(0, 5, 10, 40), labels=c("Faible", "Moyen", "Fort", "Tres.fort"))**

**enracinement.2[c(3,5)]**

**enracinement.2[-1]**

**which(enracinement.2=="Faible")**

**levels(enracinement.2)**

**levels(enracinement.2)[2]**

**as.numeric(enracinement.2)**

**sort(enracinement.2)**

**enracinement.3<-relevel(enracinement.2, ref="Tres.fort")**

**enracinement.3**

**enracinement.4<-ordered(enracinement.3, levels=c("Faible", "Moyen", "Fort", "Tres.fort"))**

**enracinement.4**

**levels(enracinement.3)[c(1,2,4)]<-"Non.connu"**

**enracinement.3**

**as.factor(as.character(enracinement.3))**

**facteur.2<-gl(2, k=3, labels = c("Moyen", "Faible"))**

**facteur.2**

**facteur.4<-gl(2, k=3, labels = c("Jaune", "Rouge"))**

**facteur.4**

**facteur.5<-interaction(facteur.2, facteur.4)**

**facteur.5**

**enracinement<-rep(c("Faible", "Moyen", "Fort"), c(1,3,2))**

**parcelle<-c("Nord", "Nord", "Nord", "Sud", "Nord", "Sud")**

**table.2<-table(enracinement, parcelle)**

**dim(table.2)**

**dimnames(table.2)**

**margin.table(table.2)**

**margin.table(table.2, 1)**

**margin.table(table.2, 2)**

**table.2[1, "Nord"]**

**table.2[, 2]**

**table.2[table.2 > 1]**

**table.2[table.2[, 2] >=1,]**

**which(table.2==0,arr.ind=TRUE)**

**table.4<-table.2**

**table.4[3,]<-c(1000,2000)**

**table.4**

**dimnames(table.4)[[2]][2] <- "Luxembourg"**

**table.4**

**dimnames(table.4)[[1]][2]<-"Tres.fort"**

**table.4**

**names(dimnames(table.4))[2]<-"Radio"**

**table.4**

**dimnames(table.4)<-NULL**

**table.4**

**cbind(table.4,table.2)**

## 1.7. Fonctions de statistique descriptive

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**sapply(mais[,2:5], mean, na.rm=TRUE)**

**summary(mais[,2:5])**

**var(mais[,2:5], na.rm=TRUE)**

**sapply(mais[,2:5], sd, na.rm=TRUE)**

**sapply(mais[,2:5], sd, na.rm=TRUE)^2**

**cor(mais[,2:5], use="pairwise.complete.obs")**

**sapply(mais[,2:5], median, na.rm=TRUE)**

**sapply(mais[,2:5], max, na.rm=TRUE)**

**sapply(mais[,2:5], min, na.rm=TRUE)**

**sapply(mais[,2:5], quantile, na.rm=TRUE)**

**summary(mais[,6:11])**

**by(mais[,2:5], mais$Parcelle, colMeans, na.rm=TRUE)**

**aggregate(mais[,2:5], list(mais$Parcelle), mean, na.rm=TRUE)**

## 1.8. Manipuler les instructions conditionnelles ou répétées en boucles

### 1.8.1. Attributs de condition

**i<-c(2,25)**

**ifelse(i < 4, i^2, i^0.5)**

**x<-NULL**

**i<-c(2,25)**

**ifelse(i < 4, x<-i^2, i^0.5)**

**x**

**i<-c(2,25)**

**ifelse(i == 2, i^2, NA)**

**i<-25**

**x<-NULL ; y<-NULL**

**if(i < 4){x<-i^2 ; y<-100} else{x<-i^0.5 ; y<-55}**

**x**

**y**

**i<-25**

**x<-NULL ; y<-NULL**

**if(i < 4){x<-i^2 ; y<-100}**

**x**

**y**

**i<-3**

**if(is.numeric(i) == TRUE){i^2}**

**i<-3**

**if(is.numeric(i)){i^2}**

**is.numeric(i)**

### 1.8.2. Attributs de répétition

**valeurs.i<-NULL ; x<-0**

**for(i in 1:10){valeurs.i<-c(valeurs.i, i) ; x<-x+5}**

**i**

**valeurs.i**

**x**

**valeurs.i<-NULL; x<-0**

**for(i in c(20,1,"chose")){valeurs.i<-c(valeurs.i, i); x<-x+5}**

**i**

**valeurs.i**

**x**

**nb<-NULL**

**for(i in 2:5){nb<-c(nb, paste("c'est la ", i,"ème répétition", sep=""))}**

**nb**

**for (i in 1:3){nom.objet<-paste("resultat", i, sep="."); assign(nom.objet, (10-i))}**

**resultat.1**

**resultat.2**

**resultat.3**

**valeurs<-NULL**

**for (i in 1:3){valeurs <- c(valeurs, get(paste("resultat.", i, sep="")))}**

**valeurs**

**dataframe.vide<-as.data.frame(matrix(nrow=3, ncol=1))**

**dataframe.vide**

**for (i in 1:3){dataframe.vide[i, ] <- get(paste("resultat.", i, sep=""))}**

**dataframe.vide**

**dataframe.vide<-as.data.frame(matrix(nrow=3, ncol=1))**

**dataframe.vide**

**for (i in 1:3){assign("dataframe.vide", {tempo <- get("dataframe.vide") ; tempo[i,] <- get(paste("resultat.", i, sep="")) ; tempo })}**

**dataframe.vide**

**valeurs.mois<-NULL; x<-NULL**

**calendrier<-month.name**

**mois<-1**

**while(mois <= 5){valeurs.mois <- c(valeurs.mois, mois) ; x<-c(x, calendrier[mois]) ; mois<-mois+1}**

**mois**

**valeurs.mois**

**x**

**x<-0**

**repeat{x<-x+1 ; if(x==500){break}}**

**x**

## 1.9. R est un éditeur de graphiques

### 1.9.1. Découverte de la fenêtre graphique

**x<-1:20**

**plot(x, x^2)**

### 1.9.2. Différents types de graphique

#### 1.9.2.1. Description des fonctions classiques

**hauteur**

**couleur**

#### 1.9.2.2. Arguments communs de ces fonctions

**x<-1:20**

**plot(x, x^2, xlim=c(0, 30), ylim=c(-100, 500), xlab="Variable x", ylab="Variable x au carré", main="Carré des valeurs 1 à 20", cex.axis=1.5, cex.lab=1.5, cex.main=2, bty="l", pch=16)**

### 1.9.3. Ajout d'éléments sur un graphique

### 1.9.4. Paramètres graphiques

**par()**

**par.ini<-par(no.readonly=TRUE)**

**par(par.ini)**

**par.ini<-par(cex = 3)**

**par.ini**

**par(par.ini)**

**par.modif<-par(cex = 3)**

**par.modif<-par(cex = 3)**

**par.modif**

**par(par.modif)**

**par(omd=c(0.05, 0.97, 0.05, 0.97))**

**par(xpd=NA)**

**plot(x=0, y=0)**

**rect(-100,-100,100,100, density=-1, col="grey0")**

**par(xpd=TRUE)**

**rect(-100,-100,100,100, density=-1, col="grey50")**

**par(xpd=FALSE)**

**rect(-100,-100,100,100, density=-1, col="grey100")**

### 1.9.5. Manipulation des polices de caractères

**plot(1:10, font.lab=2)**

**plot(1:10, font.lab=5)**

**quartzFonts()**

**par(family="Helvetica-Bold")**

**plot(1:10)**

**par(family="Helvetica-Oblique")**

**plot(1:10)**

**plot(1:10, font.lab=4)**

**par(family="mono")**

**plot(1:10, font.lab=4)**

**par(family="HersheySymbol")**

**plot(1:10, font.lab=3)**

**plot(1:10)**

**text(4, 6, "test de fonte", vfont=c("serif", "oblique cyrillic"))**

### 1.9.6. Manipulation des couleurs

**colors()**

**par(ann=FALSE, xaxt="n", yaxt="n", bty="n")**

**plot(1, col="aquamarine", pch=16, cex=10)**



**palette()**

**plot(1, col=6, pch=16, cex=10)**



**palette(colors())**

**tempo<-NULL**

**for(i in 14:1){tempo<-c(tempo, rep(i,50))}**

**windows(width=14)**

**par(ann=FALSE, xaxt="n", yaxt="n")**

**plot(rep(1:50,14)[1:657], tempo[1:657], pch=22, bg=1:657, cex=1.5, bty="n")**

**par(xpd=TRUE)**

**axis(side=2, at=14:1, labels=, cex.axis=1.5, srt=90)**

**text(rep(-2, 14), 14:1, as.character((0:13)\*50+1), srt=0, cex=1)**

**text(c(10,20,30,40,50), rep(-0.5,5), c(10,20,30,40,50), srt=0, cex=1)**

****

**palette(colors())**

**par(ann=FALSE, xaxt="n", yaxt="n", bty="n")**

**plot(1, col=630, pch=16, cex=10)**



**palette("default")**

**palette()**

**(0:255)/255**

**par(ann=FALSE, xaxt="n", yaxt="n", bty="n")**

**plot(1, col=rgb(0, 0, 255, maxColorValue=255), pch=16, cex=10)**



**plot(1, col=rgb(0, 0, 170, maxColorValue=255), pch=16, cex=10)**



**par(ann=FALSE, xaxt="n", yaxt="n", bty="n")**

**stripchart(0, pch="|", cex=10, xlim=c(-20,275))**

**for(i in 0:255){stripchart(i, col=rgb(i, 0, 0, maxColorValue=255), pch="|", cex=10, add=TRUE)}**

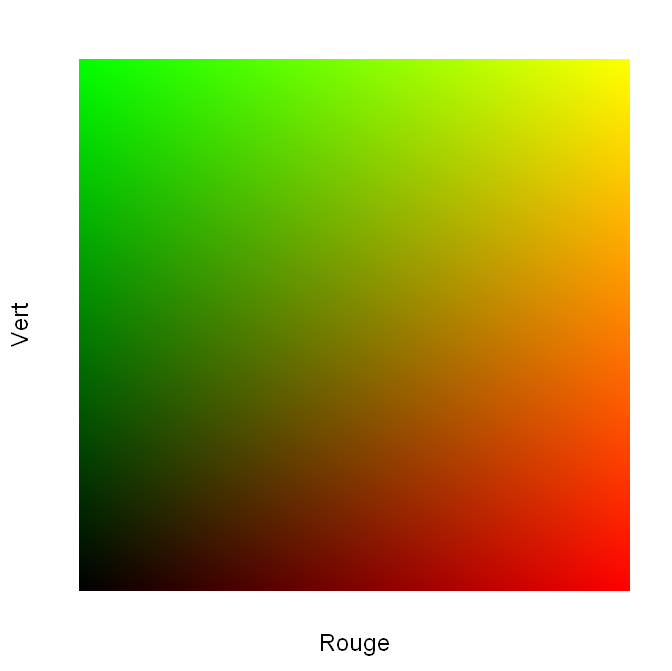
**stripchart(256, col="white", pch="|", cex=10, add=TRUE)**



**par(mar=c(6,6,4,2), xaxt="n", yaxt="n", bty="n", cex=0.5)**

**plot(x=0, y=0, pch=15, xlim=c(-1, 256), ylim=c(-1,256), xlab="Rouge", ylab="Vert", cex.lab=3)**

**for(i in 0:255){for(j in 0:255){points(x=i, y=j, col=rgb(i, j, 0, maxColorValue=255), pch=15)}}**



**rgb(0:255, 0, 0, maxColorValue=255)**

**par(ann=FALSE, xaxt="n", yaxt="n", bty="n")**

**plot(1, col="#0000FF", pch=16, cex=10)**



**col2rgb("#0000AB")**

**matrice.RGB <- matrix(c(255, 0, 0, 0, 255, 0, 0, 0, 255), ncol=3, byrow=TRUE)**

**matrice.RGB**

**rgb(matrice.RGB, maxColorValue=255)**

**par(ann=FALSE, xaxt="n", yaxt="n", bty="n")**

**plot(0:4, col=rgb(0:4, 0, 0, maxColorValue=4), pch=16, cex=5)**



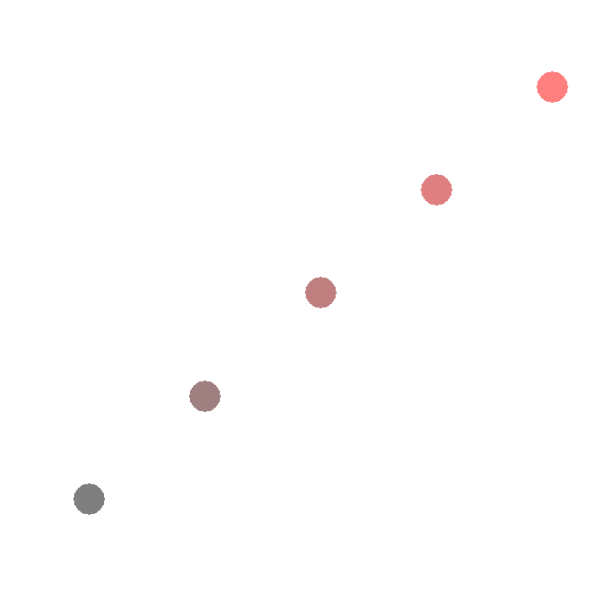
**VAL.RGB<-rgb(17.5, 0, 0, maxColorValue=255)**

**VAL.RGB**

**col2rgb(VAL.RGB)**

**par(ann=FALSE, xaxt="n", yaxt="n", bty="n")**

**plot(0:4, col=rgb(0:4, 0, 0, maxColorValue=4, alpha=2), pch=16, cex=5)**



**rgb(255, 0, 0, maxColorValue=255, alpha=254)**

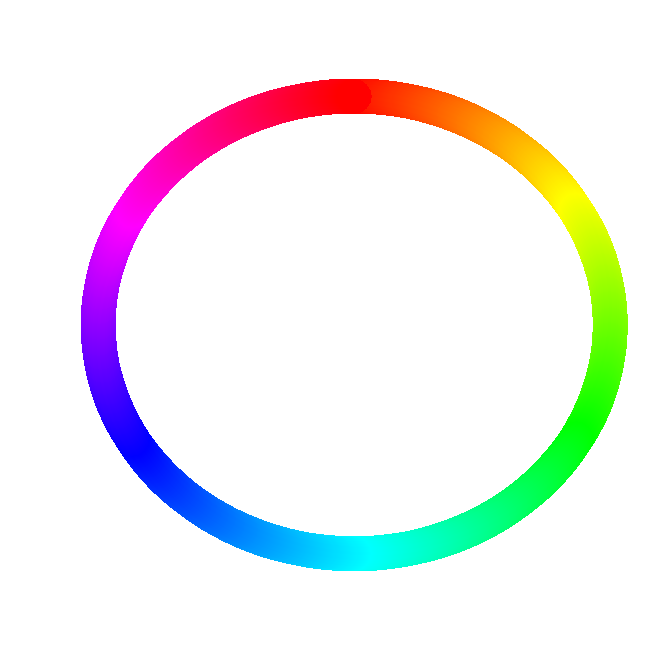
**gray(119/255)**

**theta<-seq(0, 2\*pi, 0.0001)**

**theta2<-rev(c(theta[trunc(length(theta)/4):length(theta)], theta[1:trunc(length(theta)/4)]))**

**par(ann=FALSE, xaxt="n", yaxt="n", bty="n")**

**plot(cos(theta2), sin(theta2), col=rainbow(length(theta2)), pch=16, cex=5)**



**par(ann=FALSE, xaxt="n", yaxt="n", bty="n")**

**plot(1, col=hsv(h=0, s=1, v=1), pch=16, cex=10)**



**plot(1, col=hsv(h=0.33, s=1, v=1), pch=16, cex=10)**



**plot(1, col=hsv(h=0, s=0.5, v=1), pch=16, cex=10)**



**plot(1, col=hsv(h=0, s=1, v=0.5), pch=16, cex=10)**



**plot(1, col=hsv(h=0, s=0, v=0.5), pch=16, cex=10)**



**par(ann=FALSE, xaxt="n", yaxt="n", bty="n")**

**stripchart(0, pch="|", cex=10, xlim=c(-0.05,1.05))**

**for(i in (0:255)/255){stripchart(i, col=hsv(0, 1, i), pch="|", cex=10, add=TRUE)}**

**stripchart(256/255, col="white", pch="|", cex=10, add=TRUE)**

**hsv(0, 1, (0:255)/255)**



**par(ann=FALSE, xaxt="n", yaxt="n", bty="n")**

**stripchart(0, pch="|", cex=10, xlim=c(-0.05,1.05))**

**gamma<-1**

**for(i in (0:255)/255){stripchart(i, col=rgb(i^gamma, 0^gamma, 0^gamma, maxColorValue=1), pch="|", cex=10, add=TRUE)}**

**stripchart(256/255, col="white", pch="|", cex=10, add=TRUE)**

**rgb(((0:255)/255)^gamma, 0^gamma, 0^gamma, maxColorValue=1)**



**par(ann=FALSE, xaxt="n", yaxt="n", bty="n")**

**plot(0:11, col=rainbow(12, start=0, end=10/12), pch=16, cex=5)**



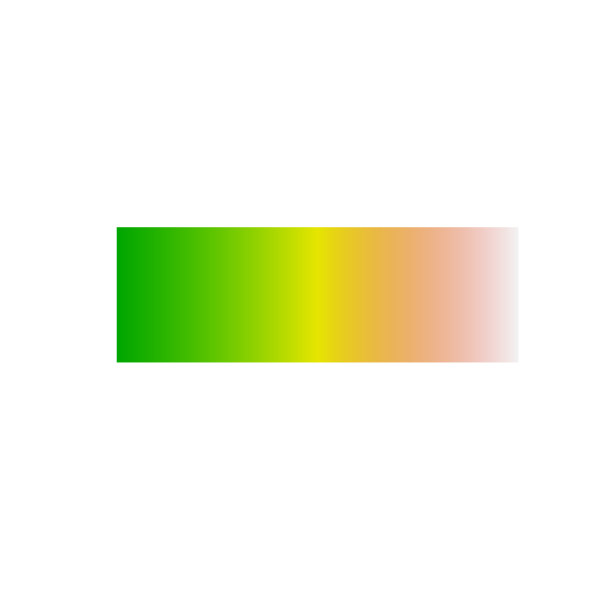
**par(ann=FALSE, xaxt="n", yaxt="n", bty="n")**

**stripchart(1, pch="|", cex=10, xlim=c(-20, 275))**

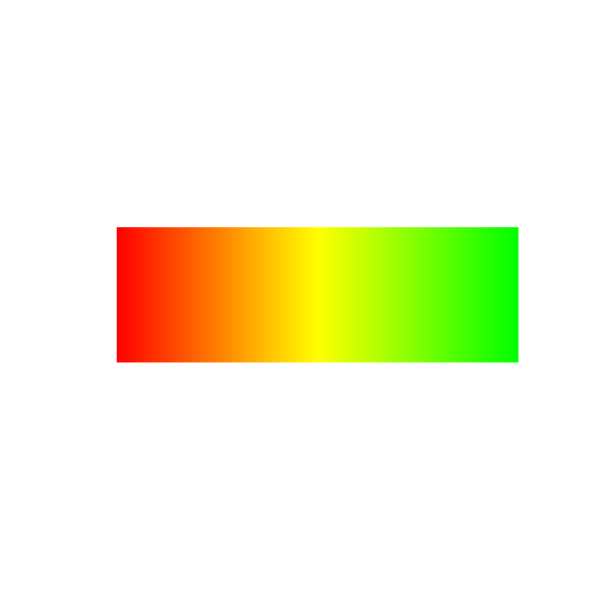
**test.palette <- terrain.colors(256)**

**for(i in 1:256){stripchart(i, col=test.palette[i], pch="|", cex=10, add=TRUE)}**

**stripchart(257, col="white", pch="|", cex=10, add=TRUE)**



**chip.colors <- hsv(seq(0, 1/3, length.out=256), 1, 1)**



**library(tcltk)**

**tclvalue(tcl("tk\_chooseColor"))**

**library(dichromat)**

**perception.standard <- rainbow(6, start=0, end=4/12)**

**perception.standard**

**perception.deutan <- dichromat(perception.standard, type="deutan")**

**perception.deutan**

**par(ann=FALSE, xaxt="n", yaxt="n", bty="n")**

**plot(0:5, col=perception.standard, pch=16, cex=5)**



**plot(0:5, col=perception.deutan, pch=16, cex=5)**

****

**par(ann=FALSE, xaxt="n", yaxt="n", bty="n")**

**plot(0:9, col=colorschemes$BrowntoBlue.10, pch=16, cex=5)**



### 1.9.7. Tracer plusieurs graphiques côte à côte dans une même fenêtre

**par(mfrow=c(3,2))**

**plot(1)**

**zones <- matrix(c(1, 1, 1, 2, 3, 4), ncol=2)**

**zones**

**layout(zones)**

**layout.show(max(zones))**

**plot(1)**

**layout(zones, widths=c(1,3), heights=c(2,1,1))**

**layout.show(max(zones))**

### 1.9.8. Gérer plusieurs fenêtres graphiques

**windows()**

**windows(width=3.5, height=3.5, pointsize=6, rescale="fit")**

**windows.options(width=3.5)**

**windows.options()**

**windows.options(width=7)**

### 1.9.9. Exporter un graphique

**graphics.off()**

**windows()**

**pdf()**

**dev.cur()**

**dev.list()**

**graphics.off()**

**pdf(file="C:/Users/Gael/Desktop/Exemple A.pdf")**

**plot(1)**

**plot(1:6)**

**dev.off()**

**pdf(file="C:/Users/Gael/Desktop/Exemple B%03d.pdf", onefile = FALSE)**

**plot(1)**

**plot(1:6)**

**dev.off()**

**CHAPITRE 2**

**NOTIONS DE BASE DE LA STATISTIQUE**

## 2.1. Introduction

## 2.2. Définitions de la statistique, de la population et de l'échantillon

## 2.3. L'individu

## 2.4. Les variables aléatoires

### 2.4.1. Définition

### 2.4.2. Deux types de variables aléatoires

#### 2.4.2.1. Variable quantitative

#### 2.4.2.2. Variable qualitative

### 2.4.3. Particularités de certaines variables qualitatives

#### 2.4.3.1. Exclusivité des classes

#### 2.4.3.2. Classes appariées

#### 2.4.3.3. Cas des variables fixées

### 2.4.4. Observer la distribution des valeurs d'une variable quantitative : l'histogramme

#### 2.4.4.1. Choix du nombre de classes

#### 2.4.4.2. Fixer l'intervalle des classes

#### 2.4.4.3. Ordonnée en effectif, fréquence ou densité

#### 2.4.4.4. La fonction hist() de R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**par(mfrow=c(1,2))**

**hist(mais$Hauteur, freq=TRUE, main=NULL, xlab= "Hauteur (cm)", ylab="Nombre de pieds de maïs (effectif)", cex.lab=1.5, cex.axis=1.5)**

**hist(mais$Hauteur, freq=FALSE, main=NULL, xlab= "Hauteur (cm)", ylab="Densité de la distribution", cex.lab=1.5, cex.axis=1.5)**

**n<-sum(hist(mais$Hauteur, plot=FALSE)$counts)**

**frequences<-hist(mais$Hauteur, plot=FALSE)$counts / n**

**frequences**

**frequences<-diff(hist(mais$Hauteur, plot=FALSE)$breaks) \* hist(mais$Hauteur, plot=FALSE)$density**

**frequences**

**barplot(frequences, space=0, col="white", xlab= "Hauteur (cm)", ylab="Proportion de pieds de maïs", cex.lab=1.5, cex.axis=1.5, bty="l")**

**axis(side=1, at=0.5:(length(frequences)-0.5), labels=hist(mais$Hauteur, plot=FALSE)$mids, cex.axis=1.5)**

### 2.4.5. Observer la distribution des valeurs d'une variable qualitative

**plot(mais$Enracinement, ylab="Nombre de pieds de maïs (effectif)", main="Enracinement", cex.main=2, cex.axis=1.5, cex.lab=1.5, cex=1.5)**

**mais$Enracinement**

**enracinement.5 <- ordered(mais$Enracinement, levels=c("Faible", "Moyen", "Fort", "Tres.fort"))**

**plot(enracinement.5, ylab="Nombre de pieds de maïs (effectif)", main="Enracinement", cex.main=2, cex.axis=1.5, cex.lab=1.5, cex=1.5)**

### 2.4.6. Limite entre l'aspect quantitatif et qualitatif d'une variable

**plot(c(150, 250, 400),c(10, 70, 17), type="h", xlab="Hauteur (cm)", ylab="Nombre de pieds de maïs (effectif)", lwd=10, cex.axis=1.5, cex.lab=1.5, cex=1.5, bty="l")**

## 2.5. Les différents types de tableaux de données

### 2.5.1. Cas standard

### 2.5.2. Le tableau disjonctif complet pour les variables qualitatives

### 2.5.3. Le tableau de contingence pour une ou deux variables qualitatives

**table6<-table(mais$Couleur, mais$Enracinement)**

**table6**

**addmargins(table6)**

## 2.6. Avant d'entreprendre toute analyse statistique : la check-list

## 2.7. Les paramètres de statistique descriptive les plus employés

### 2.7.1. La moyenne et la médiane

### 2.7.2. Les quantiles

### 2.7.3. La variance, l'écart type et le coefficient de variation

### 2.7.4. La covariance

### 2.7.5. Le coefficient de corrélation linéaire de Pearson

## 2.8. Exercices

## 2.9. Corrections des exercices

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**hist(mais$Hauteur, freq=TRUE, breaks=c(100,200,300,500), main=NULL, xlab= "Hauteur (cm)", ylab="Nombre de pieds", cex.lab=1.5, cex.axis=1.5)**

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**par(cex.axis=1.5 , cex.lab=1.5, bty="n")**

**stripchart(na.omit(mais$Hauteur), pch=16, at=0.65, xlab="Hauteur (cm)")**

**stripchart(mean(mais$Hauteur, na.rm=TRUE), add=TRUE, pch="x", cex=2.5, at=0.65)**

**abline(v=median(mais$Hauteur, na.rm=TRUE), lty="89",lwd=2)**

**abline(v=quantile(mais$Hauteur, probs=0.25, na.rm=TRUE), lty="44",lwd=2)**

**obs1<-mais$Nb.grains[which(mais$Parcelle=="Nord")]**

**par(cex.axis=1.5 , cex.lab=1.5, bty="n")**

**stripchart(na.omit(obs1), pch=16, at=0.65, xlab="Nombre de grains sur Nord")**

**stripchart(mean(obs1, na.rm=TRUE), add=TRUE, pch="x", cex=2.5, at=0.65)**

**abline(v=median(obs1, na.rm=TRUE), lty="89",lwd=2)**

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**summary(mais[,2:11])**

**sapply(mais[,2:5], sd, na.rm=TRUE)^2**

**sapply(mais[,2:5], sd, na.rm=TRUE)**

**cor(mais[,2:5], use="pairwise.complete.obs", method="pearson")**

**par(mfrow=c(4,3))**

**for (i in 2:5){hist(mais[,i], xlab="", ylab="Effectif", main=names(mais)[i], cex.axis=1.5, cex.lab=1.5)}**

**for (i in 6:11){plot(mais[,i], ylab="Effectif", main=names(mais)[i], cex.axis=1.5,cex.lab=1.5,cex=1.5)}**

**CHAPITRE 3**

**DEMARCHE SCIENTIFIQUE ET ERREURS ASSOCIEES**

## 3.1. Formulation de la question scientifique

### 3.1.1. Décalage entre la question posée et l'approche envisagée

### 3.1.2. Faits supposés avérés

## 3.2. Organisation de l'étude scientifique

### 3.2.1. Individus non semblables

**sample(1:50)**

**sample(rep(c(1, 2),c(25, 25)))**

### 3.2.2. Conditions environnementales non semblables

## 3.3. Interprétation du résultat

## 3.4. La gestion des individus extrêmes (*outliers* en anglais)

## 3.5. Conclusion

**CHAPITRE 4**

**LES ETAPES D'UN TEST STATISTIQUE**

## 4.1. Introduction à lire avant de se lancer dans ce chapitre

## 4.2. Les deux hypothèses statistiques

## 4.3. La Variable de Test (VT)

### 4.3.1. Définition

### 4.3.2. Différents types de VT

#### 4.3.2.1. Tests paramétriques et non paramétriques

#### 4.3.2.2. Estimateur et VT

#### 4.3.2.3. VT et distribution de probabilité

## 4.4. Distributions de probabilité

### 4.4.1. Définition d'une loi de probabilité

### 4.4.2. Paramètres d'une loi de probabilité

#### 4.4.2.1. Cas des variables discrètes : quantile, probabilité et fonction de répartition

#### 4.4.2.2. Cas des variables continues : quantile, densité de probabilité et fonction de répartition

#### 4.4.2.3. Calcul de la probabilité de voir apparaître une valeur de variable continue

### 4.4.3. Comment utiliser les lois de probabilité avec R

### 4.4.4. Différentes lois de probabilité discrètes

#### 4.4.4.1. Loi binomiale

**p=0.2; n=4; k=2**

**choose(n,k)\*p^k\*(1-p)^(n-k)**

**dbinom(k, n, p)**

**p=0.2; n=4; k=0:4**

**plot(k, dbinom(k, n, p), pch=16, cex=2, xlim=range(0, 5), ylim=range(0, 0.5), xlab="Nombre de boules noires k", ylab="Probabilité f(k)", cex.lab=1.5, cex.axis=1.5, bty="l")**

**p=0.2; n=4**

**dbinom(2,n,p)+dbinom(3,n,p)+dbinom(4,n,p)**

**p=0.2; n=4**

**1-dbinom(0,n,p)-dbinom(1,n,p)**

**pbinom(1,n,p, lower.tail=FALSE)**

**p=0.2; n=4**

**qbinom(0.1808,n,p, lower.tail=FALSE)**

**qbinom(0.15,n,p, lower.tail=FALSE)**

**qbinom(0.5,n,p)**

**p=0.2; n=4; k=0:20**

**plot(k,dbinom(k,n,p), type="l", xlab="Nombre de boules noires k", ylab="Probabilité f(k)", cex.lab=1.5, cex.axis=1.5)**

**p=0.2; n=20**

**lines(k,dbinom(k,n,p),lwd=3)**

**p=0.2; n=50**

**lines(k,dbinom(k,n,p), lty="39",lwd=2)**

#### 4.4.4.2. Loi multinomiale

**k1=2;k2=7;k3=1;p1=0.1;p2=0.4;p3=0.5**

**factorial(k1+k2+k3)/(factorial(k1)\*factorial(k2)\*factorial(k3))\*p1^k1\*p2^k2\*p3^k3**

**dmultinom(c(k1,k2,k3),prob=c(p1,p2,p3))**

**p1=0.1; p2=0.4; p3=1-p1-p2**

**n=10**

**x<-NULL; y<-NULL; z<-NULL**

**for(k2 in 0:n){for(k1 in 0:n){x<-c(x, k1); y<-c(y, k2); if((n-k1-k2)>=0){z<-c(z, dmultinom(c(k1, k2, (n-k1-k2)), prob=c(p1, p2, p3)))} else{z<-c(z, 0)}}}**

**library(scatterplot3d)**

**scatterplot3d(x, y, z, type="h", pch=16, xlab="Nombre de boules noires k1", ylab="", zlab="Probabilité", highlight.3d=TRUE, box=FALSE)**

**par(xpd=NA)**

**text(x=7.5, y=0.5, labels="Nombre de boules rouges k2", srt=38)**

#### 4.4.4.3. Loi de Pascal et loi binomiale négative

**p=10/50; n=4; k=2**

**choose(n-1,k-1)\*p^k\*(1-p)^(n-k)**

**p=10/50; n=4; k=2; r=2**

**dnbinom(n-k,k,p)**

**p=10/50; n=3; k=2**

**pnbinom(n-k,k,p, lower.tail=FALSE)**

**p=10/50; r=1; k=2**

**pnbinom(r,k,p, lower.tail=FALSE)**

**p=10/50; r=2; k=5**

**pnbinom(r,k,p)**

**p=0.2; n=1:80; k=2**

**plot(n,dnbinom(n-k,k,p), type="l", xlab="Nombre de tirages n (loi de Pascal)", ylab="Probabilité f(n)", cex.lab=1.5, cex.axis=1.5)**

**p=0.5; k=1**

**lines(n,dnbinom(n-k,k,p),lwd=3)**

**p=0.02; k=2**

**lines(n,dnbinom(n-k,k,p), lty="39",lwd=2)**

**p=0.2; k=15**

**lines(n,dnbinom(n-k,k,p), lty="83",lwd=2)**

**p=0.2; n=1:80; k=2**

**plot(n-k,dnbinom(n-k,k,p), type="l", xlab="Nombre de boules rouges n-k (loi binomiale négative)", ylab="Probabilité f(n-k)", cex.lab=1.5, cex.axis=1.5)**

#### 4.4.4.4. Loi géométrique

**p=10/50; n=4; r=3**

**p\*(1-p)^(n-1)**

**p=10/50; n=4; r=3**

**dgeom(r, p)**

**p=10/50; r=2**

**pgeom(r, p, lower.tail=FALSE)**

**p=10/50; r=2**

**pgeom(r, p)**

**p=0.2; n=1:50**

**plot(n, dgeom(n-1,p), type="l", xlab="Nombre de tirages n", ylab="Probabilité f(n)", cex.lab=1.5, cex.axis=1.5)**

**p=0.05**

**lines(n, dgeom(n-1,p),lwd=3)**

**p=0.5**

**lines(n, dgeom(n-1,p), lty="39",lwd=2)**

**p=0.2; n=1:50**

**plot(n-1, dgeom(n-1,p), type="l", xlab="Nombre r de boules rouges", ylab="Probabilité f(r)", cex.lab=1.5, cex.axis=1.5)**

#### 4.4.4.5. Loi hypergéométrique

**k=2; Np=10; N=50; n=4**

**choose(Np,k)\*choose(N-Np,n-k)/choose(N,n)**

**k=2; Np=10; N=50; n=4**

**dhyper(k,Np,N-Np,n)**

**k=1; Np=10; N=50; n=4**

**phyper(k,Np,N-Np,n, lower.tail=FALSE)**

**k=1; Np=10; N=50; n=4**

**phyper(k,Np,N-Np,n)**

**k=0:20; Np=10; N=50; n=4**

**plot(k,dhyper(k, Np, N-Np, n), type="l", ylim=range(0, 0.7), xlab="Nombre de boules noires k", ylab="Probabilité f(k)", cex.lab=1.5, cex.axis=1.5)**

**Np=10; n=20**

**lines(k,dhyper(k,Np,N-Np,n),lwd=3)**

**Np=10; n=48**

**lines(k,dhyper(k,Np,N-Np,n), lty="39",lwd=2)**

**Np=40; n=20**

**lines(k,dhyper(k,Np,N-Np,n), lty="83",lwd=2)**

#### 4.4.4.6. Loi de Poisson

**k=2; p=1/50; n=50**

**(n\*p)^k\*exp(-n\*p)/factorial(k)**

**k=2; p=1/50; n=50**

**dpois(k,n\*p)**

**k=1; p=1/50; n=50**

**ppois(k,n\*p, lower.tail=FALSE)**

**k=1; p=1/50; n=50**

**ppois(k,n\*p)**

**k=0:20; p=1/50; n=50**

**plot(k,dpois(k,n\*p), type="l", xlab="Nombre de boules noires k", ylab="Probabilité f(k)", cex.lab=1.5, cex.axis=1.5)**

**p=1/200; n=50**

**lines(k,dpois(k,n\*p),lwd=3)**

**p=1/50; n=200**

**lines(k,dpois(k,n\*p), lty="39",lwd=2)**

**p=1/50; n=500**

**lines(k,dpois(k,n\*p), lty="83",lwd=2)**

### 4.4.5. Différentes lois de probabilité continues

#### 4.4.5.1. Loi normale ou de Laplace-Gauss

**x=160; mu=165; sigma=7**

**1/(sqrt(2\*pi)\*sigma)\*exp(-((x - mu)^2/(2\*sigma^2)))**

**x=160; mu=165; sigma=7**

**dnorm(x,mu,sigma)**

**x=160; mu=165; sigma=7**

**pnorm(x,mu,sigma, lower.tail=FALSE)**

**x=160; mu=165; sigma=7**

**pnorm(x,mu,sigma)**

**mu=165; sigma=7**

**pnorm(mu-1.96\*sigma,mu,sigma, lower.tail=FALSE)- pnorm(mu+1.96\*sigma,mu,sigma, lower.tail=FALSE)**

**mu=165; sigma=7**

**pnorm(mu+1.96\*sigma,mu,sigma)- pnorm(mu-1.96\*sigma,mu,sigma)**

**F=0.5; mu=165; sigma=7**

**qnorm(F,mu,sigma, lower.tail=TRUE)**

**F=0.025; mu=165; sigma=7**

**qnorm(F,mu,sigma, lower.tail=FALSE)**

**mu+1.96\*sigma**

**x=seq(-10,10,0.01); mu=0; sigma=1**

**plot(x,dnorm(x,mu,sigma), type="l", xlab="Variable continue x", ylab="Densité f(x)", cex.lab=1.5, cex.axis=1.5)**

**mu=4; sigma=1**

**lines(x,dnorm(x,mu,sigma),lwd=3)**

**mu=0; sigma=4**

**lines(x,dnorm(x,mu,sigma), lty="39",lwd=2)**

#### 4.4.5.2. Loi normale centrée réduite

**x=-2.57**

**1/sqrt(2\*pi)\*exp(-x^2/2)**

**x=-2.57**

**dnorm(x)**

**x=-2.57**

**pnorm(x)**

**x=1**

**pnorm(x, lower.tail = FALSE)**

**F=0.025**

**qnorm(F, lower.tail=FALSE)**

**x=seq(-3, 3, 0.01)**

**plot(x,dnorm(x), type="l", xlab="Variable continue x", ylab="Densité f(x)", cex.lab=1.5, cex.axis=1.5)**

**x2=seq(1, 3, 0.01)**

**polygon(x=c(x2, 3, 1), y=c(dnorm(x2), 0, 0), col="grey")**

#### 4.4.5.3. Loi exponentielle

**x=2;lambda=1**

**lambda\*exp(-lambda\*x)**

**x=2**

**dexp(x)**

**x=2**

**pexp(x)**

**x=2**

**pexp(x, lower.tail=FALSE)**

**F=0.025**

**qexp(F, lower.tail=FALSE)**

**x=seq(-0.1,6,0.01); lambda=1**

**plot(x,dexp(x,lambda),type="l", xlim=range(0,5),ylim=range(0,2), xlab="Variable continue x", ylab="Densité f(x)", cex.lab=1.5, cex.axis=1.5)**

**lambda=3**

**lines(x, dexp(x, lambda),lwd=3)**

**lambda=0.3**

**lines(x, dexp(x, lambda), lty="39",lwd=2)**

#### 4.4.5.4. Loi gamma

**x=2;alpha=1;beta=1**

**1/(beta^alpha\*gamma(alpha))\*x^(alpha-1)\*exp(-x/beta)**

**x=2;alpha=1;beta=1**

**dgamma(x,shape=alpha,scale=beta)**

**x=2;alpha=1;beta=3**

**dgamma(x,shape=alpha,scale=beta)**

**x=2;alpha=1;beta=3**

**pgamma(x,shape=alpha,scale=beta)**

**x=2;alpha=1;beta=3**

**pgamma(x,shape=alpha,scale=beta, lower.tail=FALSE)**

**F=0.025; alpha=1;beta=3**

**qgamma(F, shape=alpha,scale=beta, lower.tail=FALSE)**

**x=seq(-0.1,8,0.01);alpha=1;beta=1**

**plot(x, dgamma(x,shape=alpha,scale=beta),type="l", xlim=range(0,7), xlab="Variable continue x", ylab="Densité f(x)", cex.lab=1.5, cex.axis=1.5)**

**alpha=1;beta=4**

**lines(x, dgamma(x,shape=alpha,scale=beta),lwd=3)**

**alpha=4;beta=1**

**lines(x, dgamma(x,shape=alpha,scale=beta), lty="39",lwd=2)**

**alpha=2;beta=2**

**lines(x, dgamma(x,shape=alpha,scale=beta), lty="83",lwd=2)**

#### 4.4.5.5. Loi de χ2

**x=2;ddl=4**

**1/(2^(ddl/2)\*gamma(ddl/2)) \*x^(ddl/2-1)\*exp(-x/2)**

**x=2; ddl=4**

**dchisq(x,ddl)**

**x=2; ddl=4**

**pchisq(x,ddl)**

**x=2; ddl=4**

**pchisq(x,ddl, lower.tail=FALSE)**

**F=0.025; ddl=4**

**qchisq(F,ddl, lower.tail=FALSE)**

**x=seq(-0.1,10,0.01);ddl=1**

**plot(x, dchisq(x,ddl),type="l", ylim=range(0,1), xlab="Variable continue x", ylab="Densité f(x)", cex.lab=1.5, cex.axis=1.5)**

**ddl=2**

**lines(x, dchisq(x,ddl),lwd=3)**

**ddl=4**

**lines(x, dchisq(x,ddl), lty="39",lwd=2)**

**ddl=8**

**lines(x, dchisq(x,ddl), lty="83",lwd=2)**

#### 4.4.5.6. Loi de Fisher-Snedecor

**x=2;ddl1=2;ddl2=3**

**((ddl1/ddl2)^(ddl1/2) \* gamma((ddl1+ddl2)/2) / (gamma(ddl1/2) \* gamma(ddl2/2))) \* (x^(ddl1/2-1) / (1+ddl1/ddl2\*x)^((ddl1+ddl2)/2))**

**x=2;ddl1=2;ddl2=3**

**df(x,ddl1,ddl2)**

**x=2;ddl1=2;ddl2=3**

**df(x,ddl2,ddl1)**

**x=2;ddl1=2;ddl2=3**

**pf(x,ddl1,ddl2)**

**x=2;ddl1=2;ddl2=3**

**pf(x,ddl1,ddl2, lower.tail=FALSE)**

**F=0.025; ddl1=2;ddl2=3**

**qf(F,ddl1,ddl2, lower.tail=FALSE)**

**x=seq(-0.1,2,0.01);ddl1=1;ddl2=1**

**plot(x, df(x,ddl1,ddl2),type="l", xlab="Variable continue F", ylab="Densité f(F)", cex.lab=1.5, cex.axis=1.5)**

**ddl1=5;ddl2=1**

**lines(x, df(x,ddl1,ddl2),lwd=3)**

**ddl1=1;ddl2=5**

**lines(x, df(x,ddl1,ddl2), lty="39",lwd=2)**

**ddl1=50;ddl2=500**

**lines(x, df(x,ddl1,ddl2), lty="83",lwd=2)**

#### 4.4.5.7. Loi de Student

**t=2;ddl=3**

**gamma((ddl+1)/2)/(sqrt(ddl\*pi)\*gamma(ddl/2))\*(1+t^2/ddl)^-((ddl+1)/2)**

**t=2;ddl=3**

**dt(t,ddl)**

**t=2;ddl=3**

**pt(t,ddl)**

**t=2;ddl=3**

**pt(t,ddl, lower.tail=FALSE)**

**F=0.025; ddl =3**

**qt(F,ddl, lower.tail=FALSE)**

**x=seq(-3,3,0.01);ddl=1**

**plot(x, dt(x,ddl),type="l", ylim=range(0,0.5),xlab="Variable continue t", ylab="Densité f(t)", cex.lab=1.5, cex.axis=1.5)**

**ddl=3**

**lines(x, dt(x,ddl),lwd=3)**

**ddl=30**

**lines(x, dt(x,ddl), lty="39",lwd=2)**

### 4.4.6. Distributions de probabilité qui ne suivent pas de loi connue

#### 4.4.6.1. Distribution de probabilité de Mann-Whitney

**k=3; n1=2; n2=2**

**dwilcox(k,n1,n2)**

**n1=1; n2=3**

**choose(4,1)**

**k=0:3; n1=1; n2=3**

**dwilcox(k,n1,n2)**

**k=2; n1=2; n2=2**

**pwilcox(k,n1,n2, lower.tail=FALSE)**

**k=2; n1=2; n2=2**

**pwilcox(k,n1,n2)**

**k=0:50; n1=2; n2=2**

**plot(k,dwilcox(k,n1,n2), type="l", xlab="Valeur de U", ylab="Probabilité f(U)", cex.lab=1.5, cex.axis=1.5)**

**n1=5; n2=5**

**lines(k,dwilcox(k,n1,n2),lwd=3)**

**n1=8; n2=8**

**lines(k,dwilcox(k,n1,n2), lty="39",lwd=2)**

**n1=8; n2=2**

**lines(k,dwilcox(k,n1,n2), lty="83",lwd=2)**

#### 4.4.6.2. Distribution de probabilité de Wilcoxon

**W=3; n1=2; n2=2**

**k=W-0.5\*n1\*(n1+1)**

**k**

**dwilcox(k,n1,n2)**

**W=7; n1=2; n2=2**

**k=W-0.5\*n2\*(n2+1)**

**k**

**dwilcox(k,n1,n2)**

**n1=4; n2=2**

**n1\*n2**

**U=0:8; n1=4; n2=2**

**U+0.5\*n1\*(n1+1)**

**U=0:8; n1=4; n2=2**

**U+0.5\*n2\*(n2+1)**

**W=12; n1=4; n2=2**

**W-0.5\*n1\*(n1+1)**

**k=2; n1=4; n2=2**

**pwilcox(k,n1,n2,lower.tail=FALSE)**

**W=13:18; n1=4; n2=2**

**k= W-0.5\*n1\*(n1+1)**

**sum(dwilcox(k,n1,n2))**

**U=0:30; n1=8; n2=2**

**plot(U,dwilcox(U,n1,n2), type="l", xlab="Valeur de U", ylab="Probabilité f(U) = f(W)", cex.lab=1.5, cex.axis=1.5)**

**W=U+0.5\*n1\*(n1+1)**

**n1<-2**

**n2<-2**

**n<-n1+n2**

**vecteur.ini<-c(rep("A", n1), rep("O", n2))**

**library(gtools)**

**comb<-as.data.frame(unique(permutations(n, n, vecteur.ini, set=FALSE)))**

**rang<-1:n**

**tempo<-t(comb)**

**serie.W<-NULL**

**for(i in 1:ncol(tempo)){serie.W<-c(serie.W, tapply(rang, tempo[,i], sum)[1])}**

**serie.U<-serie.W-0.5\*n1\*(n1+1)**

**comb.W.U<-cbind(comb, serie.W, serie.U)**

**comb.W.U**

**W<-sort(unique(serie.W))**

**prob.W<-table(serie.W)/length(serie.W)**

**prob.W**

**plot(W, as.vector(prob.W), pch=16, xlab="Valeur de W", ylab="Probabilité f(W)", cex.lab=1.5, cex.axis=1.5)**

#### 4.4.6.3. Distribution de probabilité du test des signes de Wilcoxon

**k=3; n=3**

**dsignrank(k,n)**

**k=0:3; n=2**

**dsignrank (k,n)**

**k=2; n=3**

**psignrank(k,n, lower.tail=FALSE)**

**k=2; n=3**

**psignrank(k,n)**

**k=0:50; n=3**

**plot(k,dsignrank(k,n), type="l", xlab="Valeur de V", ylab="Probabilité f(V)", cex.lab=1.5, cex.axis=1.5)**

**n=5**

**lines(k, dsignrank(k,n),lwd=3)**

**n=10**

**lines(k, dsignrank(k,n), lty="83",lwd=2)**

**n<- 3**

**seed<-vector("list", n)**

**for(i in 1:n){seed[[i]]<-c("A", "O")}**

**comb<-expand.grid(seed)**

**rang<-1:n**

**tempo<-t(comb)**

**serie.V<-NULL**

**for(i in 1:(ncol(tempo)-1)){serie.V<-c(serie.V, tapply(rang, tempo[,i], sum)[1])}**

**serie.V<-c(serie.V, 0)**

**comb.V<-cbind(comb, serie.V)**

**comb.V**

**V<-sort(unique(serie.V))**

**prob.V<-table(serie.V)/length(serie.V)**

**prob.V**

**plot(V, as.vector(prob.V), pch=16, xlab="Valeur de V", ylab="Probabilité f(V)", cex.lab=1.5, cex.axis=1.5)**

### 4.4.7. Rapport entre toutes ces distributions de probabilité

### 4.4.8. Remarques importantes

#### 4.4.8.1. Ne pas confondre la loi de probabilité d'une variable mesurée et celle d'une VT

#### 4.4.8.2. Simulation avec R de la fluctuation d'une VT due à l'échantillonnage

**m.theo<-165**

**n<-10**

**t<-NULL**

**for(i in 1:500){Echantillon<-rnorm(n,165,7) ; m<-mean(Echantillon) ; s<-sd(Echantillon) ; t.calc<-(m-m.theo)/(s/n^0.5) ; t<-c(t, t.calc)}**

**hist(t, freq=FALSE, main="Fluctuation de la VT t", ylim=range(0, 0.5), xlab= "Valeurs de la VT t", ylab="Densité de la distribution", cex.lab=1.5, cex.axis=1.5, cex.main = 1.8)**

**VT.t<-seq(-5,10,0.01)**

**lines(VT.t,dt(VT.t,n-1), lwd=2)**

**legend("topright", c("A cause de l'échantillonnage","Prévue par le test t"), lty=c(NA,1), lwd=c(NA,2), pch=c(22,NA), pt.cex=c(3,NA), pt.bg=c("white",NA), bty="n", cex=1.5,y.intersp=1.5)**

#### 4.4.8.3. Importance du tirage aléatoire des individus dans la formation de l'échantillon

## 4.5. Hypothèse H0, distribution de probabilité de la VT et échantillon : le cocktail magique de l'obtention de la *p value*

**pnorm(168,165,0.99, lower.tail=FALSE)**

## 4.6. Conclusion d'un test statistique et les deux risques d'erreurs associés

### 4.6.1. Conclure, c'est deux vérités, deux décisions soit quatre probabilités

### 4.6.2. L'hypothèse H0 et le risque α : définitions

### 4.6.3. La correction du seuil de rejet α

#### 4.6.3.1. Le problème soulevé

**nb.coups.partis=1; tentatives=1:10; p=1/20**

**nb.coups.vides = tentatives - nb.coups.partis**

**probabilite <- dbinom(nb.coups.partis, tentatives, p)**

**data.frame(tentatives, nb.coups.partis, nb.coups.vides, probabilite)**

**nb.coups.partis=1:10; tentatives=10; p=1/20**

**sum(dbinom(nb.coups.partis, tentatives, p))**

**m.theo<-165**

**n<-10**

**moyenne<-NULL**

**t<-NULL**

**p.value<-NULL**

**for(i in 1:20){Echantillon<-rnorm(10,165,7) ; m<-mean(Echantillon) ; moyenne<-c(moyenne, m) ; s<-sd(Echantillon) ; t.calc<-(m-m.theo)/(s/n^0.5) ; t<-c(t, t.calc) ; p<-pt(t.calc, n-1, lower.tail=FALSE) ; p.value<-c(p.value, p)}**

**length(which(p.value<=0.05))**

**data.frame(moyenne, t, p.value)**

**x<-seq(-5,5,0.01)**

**plot(x,dt(x,9), type="l", ylim=range(0, 0.55), xlab= "Valeurs de la VT t", ylab="Densité de la distribution", cex.lab=1.5, cex.axis=1.5, bty="l")**

**t.seuil<-qt(0.05,9,lower.tail=FALSE)**

**lines(c(t.seuil, t.seuil), c(0,0.4), lty=2)**

**points(t, y=rep(0,20), pch="|", cex=2.5)**

**legend(x=-4, y=0.6, c("Loi de la VT t sous Ho", "Valeur de la VT t d'un échantillon","Seuil de rejet de Ho (5%)"), lty=c(1, NA, 2), lwd=c(1, NA, 1.5), pch=c(NA, "|", NA), pt.cex=c(NA, 3, NA), bty="n", cex=1.5, y.intersp=1.5)**

#### 4.6.3.2. La technique de Bonferroni

**0.05/1:10**

#### 4.6.3.3. La technique séquentielle (Holm)

#### 4.6.3.4. Quand appliquer la correction ?

### 4.6.4. L'hypothèse H1 et son influence sur le risque α

#### 4.6.4.1 Le problème de l'hypothèse H1

#### 4.6.4.2. Test bilatéral et unilatéral

#### 4.6.4.3. Obtenir la *p value* en test bilatéral et unilatéral

**pnorm(166.9,165,0.99, lower.tail=FALSE)**

**pnorm(165.8,165,0.99)**

**pnorm(166.9,165,0.99)**

**2\*pnorm(166.9,165,0.99, lower.tail=FALSE)**

**2\*pnorm(164.5,165,0.99)**

**Mcalc=166.9**

**Mcalc.oppose=165-(Mcalc-165)**

**Mcalc.oppose**

**pnorm(Mcalc.oppose,165,0.99)**

#### 4.6.4.4. Placer les seuils α de rejet en test bilatéral et unilatéral

**qnorm(0.05,165,0.99,lower.tail=FALSE)**

**qnorm(0.05,165,0.99)**

**alpha=0.025**

**Mseuil.inf<-qnorm(alpha,165,0.99)**

**Mseuil.inf**

**Mseuil.sup<-qnorm(alpha,165,0.99, lower.tail=FALSE)**

**Mseuil.sup**

#### 4.6.4.5. Comment choisir entre test bilatéral et unilatéral ?

### 4.6.5. Le risque β et la puissance 1-β du test

#### 4.6.5.1. Retour sur les définitions de β et 1-β

#### 4.6.5.2. Variations de β et 1-β suivant la distribution de probabilité de la VT sous H1

**Mseuil=166.63**

**mu.h=seq(163,170,0.01)**

**x.label<-expression(mu[h])**

**y.label<-expression(paste("Puissance 1-", beta))**

**titre<-expression(paste("Variation pour ", alpha, " = 5% et ", M[seuil], " = 166,63"))**

**par(mar=c(5,5,4,2))**

**plot(mu.h, pnorm(Mseuil, mu.h, 0.99, lower.tail=FALSE), type="l", main=titre, xlab=x.label, ylab=y.label, bty="l", cex.lab=1.7, cex.axis=1.5, cex.main=1.7)**

**abline(v=165, lty="39", lwd=2)**

**Mseuil=163.37**

**mu.h=seq(161,168,0.01)**

**x.label<-expression(mu[h])**

**y.label<-expression(paste("Puissance 1-", beta))**

**titre<-expression(paste("Variation pour ", alpha, " = 5% et ", M[seuil], " = 163,37"))**

**par(mar=c(5,5,4,2))**

**plot(mu.h, pnorm(Mseuil, mu.h, 0.99), type="l", main=titre, xlab=x.label, ylab=y.label, bty="l", cex.lab=1.7, cex.axis=1.5, cex.main=1.7)**

**abline(v=165, lty="39", lwd=2)**

**Mseuil.inf=163.06**

**Mseuil.sup=166.94**

**mu.h=seq(160,170,0.01)**

**x.label<-expression(mu[h])**

**y.label<-expression(paste("Puissance 1-", beta))**

**titre<-expression(paste("Variation pour ", alpha, " = 5% ", M[seuil.inf], " = 163,06 et ", M[seuil.sup], " = 166,94"))**

**par(mar=c(5,5,4,2))**

**plot(mu.h, pnorm(Mseuil.inf, mu.h, 0.99) + pnorm(Mseuil.sup, mu.h, 0.99, lower.tail=FALSE), type="l", main=titre, xlab=x.label, ylab=y.label, bty="l", cex.lab=1.7, cex.axis=1.5, cex.main=1.7)**

**abline(v=165, lty="39", lwd=2)**

### 4.6.6. α et β en termes de faux positifs et faux négatifs

**x.seuil<-qnorm(0.05,165,7,lower.tail=FALSE)**

**x.seuil**

**faux.positif<- pnorm(x.seuil,165,7,lower.tail=FALSE)**

**faux.positif**

**faux.negatif <- pnorm(x.seuil,169,7)**

**faux.negatif**

### 4.6.7. Propriétés de la puissance 1-β

#### 4.6.7.1 A lire avant de se lancer dans ce paragraphe

#### 4.6.7.2. La puissance d'un test diminue quand décroît le αseuil

**alpha<-0.05**

**Mseuil<-qnorm(alpha,165,0.99,lower.tail=FALSE)**

**Mseuil**

**puissance<-pnorm(Mseuil,169,0.99,lower.tail=FALSE)**

**puissance**

**alpha<-0.001**

**Mseuil<-qnorm(alpha,165,0.99,lower.tail=FALSE)**

**Mseuil**

**puissance<-pnorm(Mseuil,169,0.99,lower.tail=FALSE)**

**puissance**

#### 4.6.7.3. La puissance d'un test croît quand augmente l'effectif n de l'échantillon

**alpha<-0.05**

**n=30**

**Mseuil<-qnorm(alpha,165, 7/n^0.5,lower.tail=FALSE)**

**Mseuil**

**puissance<-pnorm(Mseuil,169, 7/n^0.5,lower.tail=FALSE)**

**puissance**

**n=100**

**Mseuil<-qnorm(alpha,165, 7/n^0.5,lower.tail=FALSE)**

**Mseuil**

**puissance<-pnorm(Mseuil,169, 7/n^0.5,lower.tail=FALSE)**

**puissance**

**windows(width=13)**

**par(mar=c(5,5,9,4), mfrow=c(1,2))**

**n=30**

**M<-seq(160,175,0.01)**

**plot(M, dnorm(M,165,7/n^0.5), type="l", lwd=2, ylim=range(0, 0.6), xlab="Valeurs de la VT M ",ylab="Densité de la distribution", main ="n = 30", cex.lab=1.7, cex.axis=1.7, cex.main=2.5, bty="l")**

**lines(M, dnorm(M,169, 7/n^0.5), type="l", lwd=2, col="grey")**

**n=100**

**plot(M, dnorm(M,165,7/n^0.5), type="l", lwd=2, ylim=range(0, 0.6), xlab="Valeurs de la VT M ",ylab=" ", main ="n = 100", cex.lab=1.7, cex.axis=1.7, cex.main=2.5, bty="l")**

**lines(M, dnorm(M,169, 7/n^0.5), type="l", lwd=2, col="grey")**

**par(xpd=TRUE)**

**legend(x=171, y=0.75, c("H0","H1"), lty=c(1,1), col=c("black","grey"), lwd=2, cex=2, bty="n", y.intersp=1.5)**

#### 4.6.7.4. La puissance d'un test augmente avec l'écart entre les paramètres testés

**alpha<-0.05**

**Mseuil<-qnorm(alpha,165, 0.99,lower.tail=FALSE)**

**Mseuil**

**mu.h<-167**

**puissance<-pnorm(Mseuil,mu.h, 0.99,lower.tail=FALSE)**

**puissance**

**mu.h<-169**

**puissance<-pnorm(Mseuil,mu.h, 0.99,lower.tail=FALSE)**

**puissance**

### 4.6.8. Le danger de considérer la *p value* comme un indicateur de forte ou faible significativité

### 4.6.9. Alors comment fixer la puissance d'un test ?

#### 4.6.9.1. Considérations générales

#### 4.6.9.2. Réaliser des abaques

**alpha<-0.05**

**sigma<-7**

**mu.h=seq(165,190,0.01)**

**delta=mu.h-165**

**n<-30**

**Mseuil<-qnorm(alpha,165,(sigma/n^0.5),lower.tail=FALSE)**

**par(mar=c(6,5,6,2))**

**plot(delta, pnorm(Mseuil,mu.h,(sigma/n^0.5),lower.tail=FALSE), type="l", lwd=2, xlab=expression(Delta[(mu[h]-165)]),ylab=expression(1-beta), main =expression(paste(alpha[seuil], " = 5% et ", sigma, " = 7 cm")), xlim=range(0,5), cex.lab=1.9, cex.axis=1.5, cex.main=2, bty="l")**

**i=1**

**for (n in c(50,100,200,500,1000)){i<-i+1 ; Mseuil=qnorm(alpha, 165, (sigma/n^0.5), lower.tail=FALSE) ; lines(delta, pnorm(Mseuil, mu.h, (sigma/n^0.5), lower.tail=FALSE), lty=i, lwd=2)}**

**legend("bottomright", c("n=30", "n=50", "n=100", "n=200", "n=500", "n=1000"), lty=c(1:6), lwd=2, cex=1.5, bty="n", y.intersp=1.5)**

**rug(seq(0,5,0.5),-0.01) ; rug(seq(0.05,1,0.05),-0.01,side=2)**

**mu.h<-166**

**sigma<-7**

**delta=mu.h-165**

**n=seq(30,1000,0.1)**

**alpha<-0.15**

**Mseuil<-qnorm(alpha,165,(sigma/n^0.5),lower.tail=FALSE)**

**par(mar=c(6,5,6,2))**

**plot(n, pnorm(Mseuil,mu.h,(sigma/n^0.5),lower.tail=FALSE), type="l", lwd=2, xlab=" n", ylab=expression(1-beta), main = expression(paste(Delta[(mu[h]-165)], " = 1 cm et ", sigma, " = 7 cm")), xlim=range(30,1000), ylim=range(0,1), cex.lab=1.9, cex.axis=1.5, cex.main=2, bty="l")**

**i=1**

**for (alpha in c(0.1, 0.05, 0.01, 0.005, 0.001)){i<-i+1 ; Mseuil=qnorm(alpha, 165, (sigma/n^0.5), lower.tail=FALSE) ; lines(n, pnorm(Mseuil, mu.h, (sigma/n^0.5), lower.tail=FALSE), lty=i, lwd=2)}**

**legend("bottomright", c(expression(paste(alpha[seuil], " = 15%")), expression(paste(alpha[seuil], " = 10%")), expression(paste(alpha[seuil], " = 5%")), expression(paste(alpha[seuil], " = 1%")), expression(paste(alpha[seuil], " = 0,5%")), expression(paste(alpha[seuil], " = 0,1%"))), lty=c(1:6), lwd=2, cex=1.5, bty="n", y.intersp=1.5)**

#### 4.6.9.3. Les fonctions disponibles sous R

**power.t.test(n=20, delta=4, sd=7, sig.level=0.05, power=NULL, type="one.sample", alternative="one.sided")**

**power.t.test(n=NULL, delta=2, sd=20, sig.level=0.05, power=0.8, type="one.sample", alternative="one.sided")**

#### 4.6.9.4. Le ncp des lois de probabilité de VT sous R

**par(mar=c(5,5,5,2))**

**par(xpd=TRUE)**

**n<-20**

**VT.t<-seq(-5, 10, 0.01)**

**m.theo<-165**

**m.testee<-169**

**sigma<-7**

**ncp.test = (m.testee-m.theo)/(sigma/n^0.5)**

**plot(VT.t, dt(VT.t, n-1), type="l", lwd=2, xlab="Valeurs de la VT t", ylab="Densité de la distribution", cex.lab=1.7, cex.axis=1.7, bty="l")**

**lines(VT.t, dt(VT.t, n-1, ncp= ncp.test), type="l", lwd=2, col="grey")**

**legend(x=4, y=0.4, c(expression(H[0]), expression(H[1])), lty=c(1,1), col=c("black","grey"), lwd=2, cex=2, bty="n", y.intersp=1.5)**

**alpha<-0.05**

**t.seuil<-qt(alpha, n-1, lower.tail=FALSE)**

**puissance<-pt(t.seuil, n-1, ncp= ncp.test, lower.tail=FALSE)**

**puissance**

**power.t.test(n=20, delta=2, sd=7, sig.level=0.05, power=NULL, type="one.sample", alternative="two.sided", strict=TRUE)**

**n<-20**

**m.theo<-165**

**m.testee<-167**

**sigma<-7**

**ncp.test = (m.testee-m.theo)/(sigma/n^0.5)**

**t.seuil.sup<-qt(alpha/2, n-1, lower.tail=FALSE)**

**t.seuil.inf<-qt(alpha/2, n-1)**

**puissance<-pt(t.seuil.sup, n-1, ncp= ncp.test, lower.tail=FALSE) + pt(t.seuil.inf, n-1, ncp= ncp.test)**

**puissance**

**power.t.test(n=20, delta=2, sd=7, sig.level=0.05, power=NULL, type="one.sample", alternative="two.sided")**

**puissance<-pt(t.seuil.sup, n-1, ncp= ncp.test, lower.tail=FALSE)**

**puissance**

**m.theo<-165**

**m.testee<-169**

**n<-10**

**t<-NULL**

**for(i in 1:500){Echantillon<-rnorm(10,m.testee,7) ; m<-mean(Echantillon) ; s<-sd(Echantillon) ; t.calc<-(m-m.theo)/(s/n^0.5) ; t<-c(t, t.calc)}**

**par(xpd=FALSE)**

**hist(t, freq=FALSE, main="Fluctuation de la VT t", ylim=range(0, 0.5), xlab= "Valeurs de la VT t", ylab="Densité de la distribution", cex.lab=1.5, cex.axis=1.5, cex.main = 1.8)**

**VT.t<-seq(-5,10,0.01)**

**lines(VT.t,dt(VT.t,n-1), lwd=2)**

**sigma<-7**

**ncp.test = (m.testee-m.theo)/(sigma/n^0.5)**

**lines(VT.t, dt(VT.t, n-1, ncp= ncp.test), type="l", lwd=2, col="grey")**

**par(xpd=TRUE)**

**legend(x=1, y=0.52, c("A cause de l'échantillonnage", expression(paste("Sous ", H[0] ," (", mu[h], " = 165 cm)")), expression(paste("Sous ", H[1] ," (", mu[h], " = 169 cm)"))), lty=c(NA,1,1), lwd=c(NA,2,2), col=c("black", "black", "grey"), pch=c(22,NA,NA), pt.cex=c(3,NA,NA), pt.bg=c("white",NA,NA), bty="n", cex=1.5, y.intersp=1.5)**

### 4.6.10. Comment conclure finalement ?

## 4.7. Récapitulation

## 4.8. Exercices

## 4.9. Correction des exercices

**nb.eleves<-122:130**

**nb.colleges<-c(1,4,24,48,59,108,92,52,12)**

**proportion.colleges<- nb.colleges / sum(nb.colleges)**

**effectif<-rep(nb.eleves, nb.colleges)**

**hist(effectif,breaks=c(121.5:130.5), freq=FALSE, main=NULL, xlab="Nombre d'élèves par collège", ylab="Proportion de collèges", cex.lab=1.5, cex.axis=1.5)**

**plot(nb.eleves, proportion.colleges, type = "b", pch=16, cex=1.5, xlab="Nombre d'élèves par collège", ylab="Proportion de collèges", cex.lab=1.5, cex.axis=1.5, bty="l")**

**barplot(proportion.colleges, names.arg=nb.eleves, xlab="Nombre d'élèves par collège", ylab="Proportion de collèges", cex=1.5, cex.lab=1.5, cex.axis=1.5, bty="l")**

**plot(nb.eleves, proportion.colleges, type = "h", pch=16, cex=1.5, xlab="Nombre d'élèves par collège", ylab="Proportion de collèges", cex.lab=1.5, cex.axis=1.5, bty="l")**

**dhyper(10,12,8,10)**

**nb.males<-0:20**

**nb.femelles<-rev(nb.males)**

**proba<-dhyper(8, nb.males, nb.femelles, 10)**

**cbind(nb.males, proba)**

**phyper(6,12,8,10, lower.tail=FALSE)**

**nb.males<-7:10**

**sum(dhyper(nb.males, 12, 8, 10))**

**dgeom(3,(12/20))**

**mean<-43**

**sigma<-2.3**

**pnorm(42.5, mean, sigma, lower.tail=FALSE)-pnorm(43.5, mean, sigma, lower.tail=FALSE)**

**pnorm(43.5, mean, sigma)-pnorm(42.5, mean, sigma)**

**pnorm(43.05, mean, sigma)-pnorm(42.95, mean, sigma)**

**(43.5-42.5)\*dnorm(43, mean, sigma)**

**dnorm(170,165,7)**

**xj<-170**

**xk<-170**

**intervalle<-NULL**

**F<-NULL**

**D<-NULL**

**for (i in 1:500){xj<-(xj-0.1) ; xk<-(xk+0.1) ; intervalle<-c(intervalle, (xk-xj)) ; F<-c(F, pnorm(xk, 165, 7)- pnorm(xj, 165, 7)) ; D<-c(D, (xk-xj) \* dnorm(170, 165, 7))}**

**par(mar=c(5,5,5,4), xpd=FALSE)**

**plot(intervalle, F, type="l", lwd = 2, xlab="Distance xk-xj en cm centrée sur 170 cm", ylab="Probabilité P(X = 170 cm)", cex.lab=1.7, cex.axis=1.7, bty="l")**

**lines(intervalle, D, type="l", lty="44", lwd=2)**

**legend(x=30, y=0.4, c("Fonction de répartition","Densité"), lty=c(1,2), lwd=c(2,2), bty="n", cex=1.7, y.intersp=1.5)**

**pnorm(180,165,7)- pnorm(150,165,7)**

**xj<-seq(140, 188, by = 2)**

**xk<-seq(142, 190, by = 2)**

**xi<-(xk+xj)/2**

**frequences <- pnorm(xk,165,7)- pnorm(xj,165,7)**

**densites <- frequences / (xk-xj)**

**barplot(densites, space=0, col="white", xlab="Taille des femmes", ylab="Densité de la distribution", cex.lab=1.5, cex.axis=1.5, bty="l")**

**axis(side=1, at=0.5:(length(xi)-0.5), labels=xi, cex.axis=1.5)**

**x<-seq(140, 190, 0.1)**

**lines((x-140)/((190-140)/length(xi)), dnorm(x, 165, 7), type="l")**

**runif(1,20,30)**

**sample(20:30, 1)**

**k=0:4; n1=2; n2=2**

**plot(k,dwilcox(k,n1,n2), pch=16, cex=1.5, ylim=range(0, 0.4), xlab="Valeur de U", ylab="Probabilité f(U)", cex.lab=1.5, cex.axis=1.5)**

**n1=2; n2=2**

**U.calc=4**

**U.seuil=3**

**alpha.seuil<-pwilcox(U.seuil-1, n1, n2, lower.tail=FALSE)**

**alpha.seuil**

**p.value<-pwilcox(U.calc-1, n1, n2, lower.tail=FALSE)**

**p.value**

**U.seuil=0:5; n1=2; n2=2**

**F.U.droite.moins.1 <- pwilcox(U.seuil-1, n1, n2, lower.tail=FALSE)**

**cbind(U.seuil, F.U.droite.moins.1)**

**qwilcox(0.05, n1, n2, lower.tail=FALSE) + 1**

**qwilcox(0.4, n1, n2, lower.tail=FALSE) + 1**

**min(pwilcox(3-1, n1, n2, lower.tail=FALSE), pwilcox(3,n1, n2))\*2**

**n<-10**

**F<-rnorm(n/2,165,7)**

**H<-rnorm(n/2,169,7)**

**obs1<-data.frame(c(F,H),c(rep("Femme",n/2),rep("Homme",n/2)))**

**names(obs1)<-c("Taille", "Sexe")**

**obs1**

**library(gplots)**

**m<-sapply(split(obs1[, 1], obs1[, 2]), mean)**

**s<- sapply(split(obs1[, 1], obs1[, 2]), sd)**

**barplot2(m, xlim=c(0, 4), ylim= range(120, 200), space=0.6, names.arg=levels(obs1[, 2]), plot.ci=TRUE, ci.l=(m-s), ci.u= (m+s), xpd=FALSE, xlab=names(obs1)[2], ylab=paste("Moyenne +/- écart type de", names(obs1)[1]), cex.lab=1.5, cex.axis=1.5,cex.names=1.5, ci.lwd=2)**

**t.test(split(obs1[, 1], obs1[, 2])[[2]], split(obs1[, 1], obs1[, 2])[[1]],var.equal=TRUE, alternative="greater")**

**n<-10**

**par(mar=c(5,5,10,4))**

**t<-NULL**

**for(i in 1:5000){Femme<-rnorm(n/2,165,7) ; mf<-mean(Femme) ; sf<-sd(Femme) ; Homme<-rnorm(n/2,169,7) ; mh<-mean(Homme) ; sh<-sd(Homme) ; s2<-((n/2-1)\*sf^2+(n/2-1)\*sh^2)/(n-2) ; t.calc<-(mh-mf)/(s2\*(1/(n/2)+1/(n/2)))^0.5 ; t<-c(t, t.calc)}**

**hist(t, freq=FALSE, xlim=range(-4, 8), ylim=range(0, 0.4), xlab="Valeurs de la VT t ",ylab="Densité de la distribution", main =NULL, cex.lab=1.7, cex.axis=1.7, cex.main=2.5)**

**VT.t<-seq(-5,5,0.01)**

**lines(VT.t, dt(VT.t,n-2), type="l", lwd=2)**

**par(xpd=TRUE)**

**legend(x=-3, y=0.53, c("Par l'échantillonnage (distribution réelle)", expression(paste("Prévue sous ", H[0]))), lty=c(NA,1), lwd=c(NA,1.5), pch=c(22,NA), pt.cex=c(3,NA), pt.bg=c("white",NA), bty="n", cex=1.5, y.intersp=1.5)**

**text(x=1, y=0.55, paste("n = ",n), cex=2.5)**

**n <- 200**

**k <- 4**

**proba.theo <- rep(1/k,k)**

**X2.obs<-NULL**

**for(i in 1:500){obs <- rmultinom(1, n, prob=proba.theo) ; X2.obs <-c(X2.obs, sum((obs - proba.theo \* n)^2 / (proba.theo \* n)))}**

**hist(X2.obs, freq=FALSE, main = expression(paste("Fluctuation de la VT ", chi^2)), ylim=range(0, 0.3), xlab = expression(paste("Valeurs de la VT ", chi^2)), ylab="Densité de la distribution", cex.lab=1.5, cex.axis=1.5, cex.main = 1.8)**

**VT.X2=seq(-0.1,40,0.01)**

**ddl=k-1**

**lines(VT.X2, dchisq(VT.X2,ddl))**

**legend("topright", c("A cause de l'échantillonnage","Prévue par le test"), lty=c(NA,1), lwd=c(NA,2), pch=c(22,NA), pt.cex=c(3,NA), pt.bg=c("white",NA), bty="n", cex=1.5, y.intersp=1.5)**

**G.obs<-NULL**

**for(i in 1:500){obs <- rmultinom(1, n, prob=proba.theo) ; G.obs <-c(G.obs, 2 \* sum(obs\*log(obs / (proba.theo \* n))))}**

**hist(G.obs, freq=FALSE, main = "Fluctuation de la VT G", ylim=range(0, 0.3), xlab = "Valeurs de la VT G", ylab="Densité de la distribution", cex.lab=1.5, cex.axis=1.5, cex.main = 1.8)**

**proba.popu <- c(0.21, 0.23, 0.27, 0.29)**

**alpha<-0.05**

**X2.seuil<-qchisq(alpha,ddl,lower.tail=FALSE)**

**X2.seuil**

**puissance <- length(which(X2.obs >= X2.seuil))/length(X2.obs)**

**puissance**

**nf=11**

**nh=9**

**n<-nf + nh**

**mu.f=165**

**mu.h=165**

**Uh<-NULL**

**for(i in 1:5000){Echantillon <- c(rnorm(nf, mu.f, 7), rnorm(nh, mu.h, 7)) ; Wh.calc <- sum(rank(Echantillon)[(nf+1):n]) ; Uh.calc <- Wh.calc-0.5 \* nh \* (nh+1) ; Uh<-c(Uh, Uh.calc)}**

**hist(Uh, freq=FALSE, main = expression(paste("Fluctuation de la VT ", U[h])), ylim=range(0, max(hist(Uh, plot = FALSE)$density) \* 1.3), xlab= expression(paste("Valeurs de la VT ", U[h])), ylab="Densité de la distribution", cex.lab=1.5, cex.axis=1.5, cex.main = 1.8)**

**VT.U<-0:nf\*nh**

**lines(VT.U, dwilcox(VT.U, nh, nf), lwd=2)**

**legend("topright", c("A cause de l'échantillonnage","Prévue par le test de Mann - Whitney"), lty=c(NA,1), lwd=c(NA,2), pch=c(22,NA), pt.cex=c(3,NA), pt.bg=c("white",NA), bty="n", cex=1.5, y.intersp=1.5)**

**alpha<-0.05**

**Uh.seuil<-qwilcox(alpha, nf, nh, lower.tail=FALSE)**

**Uh.seuil**

**puissance <- length(which(Uh >= Uh.seuil))/length(Uh)**

**puissance**

**n<-30**

**p1<-seq(0, 1, 0.01)**

**p2<-NULL**

**compteur<-NULL**

**j<-0**

**for(i in p1){j<-j+1 ; try<-try(power.prop.test(n=n, p1=i, p2=NULL, sig.level=0.05, power=0.8, alternative="two.sided", strict=TRUE)$p2, silent=TRUE) ; if(is.numeric(try)==TRUE){p2<-c(p2, try) ; compteur <- c(compteur, j)}}**

**p1<-p1[compteur]**

**par(mar=c(6,5,6,2))**

**plot(p1, p2, type="l", lwd=2, xlim = range(0,1), ylim = range(0,1), xlab = expression(p["attaqué / vertical"]), ylab = expression(p["attaqué / versé "]), main =expression(paste(alpha[seuil], " = 5% et 1 - ", beta, " = 80%")), cex.lab=1.9, cex.axis=1.5, cex.main=2, bty="l")**

**rug(seq(0,1,0.05), -0.01, side=1) ; rug(seq(0,1,0.05), -0.01, side=2)**

**k=1**

**for (n in c(50,100,200,500,1000)){k<-k+1 ; p1<-seq(0, 1, 0.01) ; p2<-NULL ; compteur<-NULL ; j<-0 ; for(i in p1){j<-j+1 ; try<-try(power.prop.test(n=n, p1=i, p2=NULL, sig.level=0.05, power=0.8, alternative="two.sided", strict=TRUE)$p2, silent=TRUE) ; if(is.numeric(try)==TRUE){p2<-c(p2, try) ; compteur <- c(compteur, j)}} ; p1<-p1[compteur] ; lines(p1, p2, lty=k, lwd=2)}**

**legend("bottomright", c("n=30", "n=50", "n=100", "n=200", "n=500", "n=1000"), lty=c(1:6), lwd=2, cex=1.5, bty="n", y.intersp=1.5)**

**points(x=0.368, y=0.571, pch="X", cex=2)**

**CHAPITRE 5**

**LES TESTS STATISTIQUES**

## 5.1. A lire absolument avant d'utiliser un test

## 5.2. Quel test appliquer et quelle fonction de R utiliser ?

## 5.3. χ2 de conformité

### 5.3.1. Méthode

### 5.3.2. Exemple avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-data.frame(mais$Couleur)**

**# obs1 <- as.data.frame(matrix(rep(c("Jaune", "Jaune.rouge", "Rouge"), times=c(48, 22, 29)), ncol=1))**

**names(obs1)<-c("Couleur")**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na <- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**obs2[, 1]<-factor(obs2[, 1])**

**obs3<- table(obs2, dnn = names(obs1))**

**addmargins(obs3)**

**graphics.off()**

**pi.theo<- c(0.5,0.3,0.2)**

**pi<- obs3/ margin.table(obs3)**

**par(mar=c(5.1, 5.1, 4.1, 2.1))**

**barplot(pi, xlab=names(obs2)[1], ylab="Proportion", ylim=range(0, max(pi, pi.theo) + 0.15), cex.lab=2, cex.axis=1.8, cex=1.8)**

**points(barplot(1:length(obs3), plot=FALSE), pi.theo, pch="-", cex=5)**

**pi.theo<- c(0.5, 0.3, 0.2)**

**chisq.test(obs3, p=pi.theo)**

**k<-dim(obs3)**

**par<-1**

**nu=k-par**

**nu**

**obs3theo<-suppressWarnings(chisq.test(obs3, p=pi.theo)$expected)**

**addmargins(as.table(obs3theo))**

**n.cases.theo<-length(obs3theo)**

**n.cases.inf.1<-length(which(obs3theo<1))**

**n.cases.inf.5<-length(which(obs3theo<5))**

**n.cases.entre.1et5<- length(which(obs3theo<=5 & obs3theo>=1))**

**prop.Cochran<- length(which(obs3theo>5))/n.cases.theo**

**if(n.cases.inf.1>0){cat("La règle de Cochran n'est pas respectée\nLes n° de cases suivants sont d'effectif théorique strictement inférieur à 1 :\n") ; print(which(obs3theo<1, arr.ind=TRUE)) ; cat("\n")}**

**if(n.cases.inf.1==0 & n.cases.inf.5>0 & prop.Cochran<0.8){cat("La règle de Cochran n'est pas respectée\nLes n° de cases suivants sont d'effectif théorique entre 1 et 5 :\n") ; print(which(obs3theo<=5 & obs3theo>=1, arr.ind=TRUE)) ; cat("Et la proportion de cases d'effectif théorique strictement supérieur à 5 est : ") ; cat(prop.Cochran, "\n")}**

**if(n.cases.inf.1==0 & n.cases.inf.5>0 & prop.Cochran>=0.8){cat("La règle de Cochran est respectée\nLes n° de cases suivants sont d'effectif théorique entre 1 et 5 :\n") ; print(which(obs3theo<=5 & obs3theo>=1, arr.ind=TRUE)) ; cat("Et la proportion de cases d'effectif théorique strictement supérieur à 5 est : ") ; cat(prop.Cochran, "\n")}**

**if(n.cases.inf.5==0){cat("La règle de Cochran est respectée\nIl n'y a pas de cases d'effectif théorique strictement inférieur à 5\n")}**

**pi<- obs3/ margin.table(obs3)**

**pi**

**ni<-obs3**

**n <- margin.table(obs3)**

**ti <- pi.theo\*n**

**chi2.calc<-sum((ni-ti)^2/ti)**

**chi2.calc**

**pchisq(chi2.calc, nu, lower.tail=FALSE)**

**obs3<- as.table(matrix(c(4,0,0,0), nrow=1))**

**nc<-ncol(obs3)**

**n<-margin.table(obs3)**

**pi.theo<- c(0.1, 0.2, 0.3, 0.4)**

**tij<-pi.theo\*n**

**vecteur.ini<-NULL**

**for(i in 1:nc){vecteur.ini<-c(vecteur.ini, rep(LETTERS[i], n))}**

**library(gplots)**

**table.diff<-as.data.frame(unique(combinations(nc\*n, n, vecteur.ini, set=FALSE)))**

**tempo<-as.data.frame(t(table.diff))**

**obs3.diff<-vector("list", nrow(table.diff))**

**serie.VT<-NULL**

**serie.prob<-NULL**

**for(i in 1:ncol(tempo)){ tempo[,i] <- factor(tempo[,i], levels=LETTERS[1:nc]); nij.tempo<-matrix(table(tempo[,i]), ncol=nc); obs3.diff[[i]] <- nij.tempo; chi2 <- round(sum((nij.tempo - tij)^2 / tij), 8); serie.VT<-c(serie.VT, chi2); serie.prob<-c(serie.prob, dmultinom(as.vector(nij.tempo), prob=pi.theo))}**

**# obs3.diff**

**table.diff.VT.prob<-cbind(table.diff, serie.VT, serie.prob)**

**table.diff.VT.prob**

**VT<-round(sort(unique(serie.VT)), 8)**

**prob.VT<-sapply(split(serie.prob, serie.VT), sum)**

**prob.VT**

**plot(VT, prob.VT, type="l", xlab="Valeur de chi2", ylab="Probabilité f(chi2)", cex.lab=1.5, cex.axis=1.5)**

**VT.calc <-round(sum((obs3 - tij)^2 / tij), 8)**

**VT.calc**

**p.value<-sum(prob.VT[VT>=VT.calc])**

**p.value**

**comb<-unique(permutations(n, n, as.matrix(table.diff[1,]), set=FALSE))**

**for(i in 2:nrow(table.diff)){comb<-rbind(comb, unique(permutations(n, n, as.matrix(table.diff[i,]), set=FALSE)))}**

**tempo<-as.data.frame(t(comb))**

**for(i in 1:length(tempo)){tempo[,i] <- factor(tempo[,i], levels=LETTERS[1:nc])}**

**serie.VT.comb<-NULL**

**for(i in 1:ncol(tempo)){nij.tempo<-matrix(table(tempo[,i]), nrow=1) ; serie.VT.comb <- c(serie.VT.comb, round(sum((nij.tempo - tij)^2 / tij), 8))}**

**tempo2<-comb**

**for(i in 1:nc){tempo2[which(tempo2 == LETTERS[i])] <- pi.theo[i]}**

**tempo2 <- matrix(as.numeric(tempo2), ncol=n)**

**prob <- apply(tempo2, 1, prod)**

**comb.VT.prob<-cbind(as.data.frame(comb), serie.VT.comb, prob)**

**# comb.VT.prob**

**VT<-sort(unique(serie.VT.comb))**

**prob.VT<-sapply(split(comb.VT.prob[, ncol(comb.VT.prob)], comb.VT.prob[, ncol(comb.VT.prob)-1]), sum)**

**prob.VT**

### 5.3.3. Tests de comparaisons deux à deux

**cat("Vérifier que le tableau obs3 correspond bien à celui du test dont on souhaite les comparaisons multiples\n")**

**k<-length(obs3)**

**n<-margin.table(obs3)**

**pi.theo=c(0.5, 0.3, 0.2)**

**classe<-NULL; p.value<-NULL**

**for(i in 1:k){classe<- c(classe, names(obs3)[i]) ; obs<- data.frame(obs3[i], n- obs3[i]) ; obstheo<- chisq.test(obs, p=c(pi.theo[i], 1-pi.theo[i]))$expected ; n.cases.inf.5<-length(which(obstheo<5)) ; if(n.cases.inf.5 == 0){ p.value<-c(p.value, round(chisq.test(c(obs3[i], n- obs3[i]), p=c(pi.theo[i], (1- pi.theo[i])))$p.value, 9))} else {p.value<- c(p.value, NA)}}**

**p.mult<-data.frame(classe, p.value)**

**p.mult**

**p.mult2<-transform(p.mult, Holm=round(p.adjust(p.mult$p.value, method="holm"), 6))**

**p.mult2**

## 5.4. χ2 d'homogénéité

### 5.4.1. Méthode

### 5.4.2. Exemples avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[,c(6,8)]**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**for(i in 1:length(obs2)){obs2[,i]<-factor(obs2[,i])}**

**obs3 <- table(obs2)**

**# tempo <- list(c("Jaune", "Jaune.rouge", "Rouge"), c("Faible", "Fort", "Moyen", "Tres.fort")) ; names(tempo) <- c("Couleur", "Enracinement"); obs3 <- as.table(matrix(c(13, 2, 3, 6, 7, 13, 17, 3, 8, 12, 10, 5), ncol = 4, dimnames=tempo))**

**addmargins(obs3)**

**graphics.off()**

**par(cex.lab=1.5, xpd=NA, font=2)**

**mosaicplot(t(obs3), main=NULL, cex.axis=1.1)**

**chisq.test(obs3)**

**obs3theo<-suppressWarnings(chisq.test(obs3)$expected)**

**addmargins(as.table(obs3theo))**

**nij<-obs3**

**tij<-obs3theo**

**chi2.calc<-sum((nij-tij)^2/tij)**

**chi2.calc**

**k<-dim(obs3)[1]**

**c<-dim(obs3)[2]**

**nu=(k-1)\*(c-1)**

**nu**

**pchisq(chi2.calc, nu, lower.tail=FALSE)**

**obs3<- as.table(matrix(c(4,0,0,0), ncol=2))**

**k<-dim(obs3)[1]**

**c<-dim(obs3)[2]**

**n<-margin.table(obs3)**

**nc<-k\*c**

**vecteur.ini<-NULL**

**for(i in 1:nc){vecteur.ini<-c(vecteur.ini, rep(LETTERS[i], n))}**

**library(gplots)**

**library(MASS)**

**table.diff<-as.data.frame(unique(combinations(nc\*n, n, vecteur.ini, set=FALSE)))**

**tempo<-as.data.frame(t(table.diff))**

**obs3.diff<-vector("list", nrow(table.diff))**

**serie.VT<-NULL**

**serie.prob<-NULL**

**for(i in 1:ncol(tempo)){ tempo[,i] <- factor(tempo[,i], levels=LETTERS[1:nc]); nij.tempo<-matrix(table(tempo[,i]), ncol=c); obs3.diff[[i]] <- nij.tempo; chi2.res <- suppressWarnings(chisq.test(nij.tempo, correct=FALSE)$residuals); chi2.res[which(chi2.res == "NaN")]<-0; serie.VT<-c(serie.VT, round(sum(chi2.res^2), 9)); serie.prob<-c(serie.prob, dmultinom(as.vector(nij.tempo), prob=rep(1/nc, nc)))}**

**# obs3.diff**

**table.diff.VT.prob<-cbind(table.diff, serie.VT, serie.prob)**

**table.diff.VT.prob**

**VT<-round(sort(unique(serie.VT)), 8)**

**prob.VT<-sapply(split(serie.prob, serie.VT), sum)**

**prob.VT**

**plot(VT, prob.VT, type="l", xlab="Valeur de chi2", ylab="Probabilité f(chi2)", cex.lab=1.5, cex.axis=1.5)**

**chi2.res <-suppressWarnings(chisq.test(obs3, correct=FALSE)$residuals); chi2.res[which(chi2.res == "NaN")]<-0; VT.calc<-round(sum(chi2.res^2), 8)**

**VT.calc**

**p.value<-sum(prob.VT[VT>=VT.calc])**

**p.value**

**comb<-unique(permutations(n, n, as.matrix(table.diff[1,]), set=FALSE))**

**for(i in 2:nrow(table.diff)){comb<-rbind(comb, unique(permutations(n, n, as.matrix(table.diff[i,]), set=FALSE)))}**

**tempo<-as.data.frame(t(comb))**

**for(i in 1:length(tempo)){tempo[,i] <- factor(tempo[,i], levels=LETTERS[1:nc])}**

**serie.VT.comb<-NULL ; obs3.diff<-vector("list", nrow(comb)) ; compteur <- 0**

**for(i in 1:ncol(tempo)){nij.tempo<-matrix(table(tempo[,i]), ncol=c) ; chi2.res <- suppressWarnings(chisq.test(nij.tempo)$residuals); chi2.res[which(chi2.res == "NaN")]<-0; serie.VT.comb<-c(serie.VT.comb, round(sum(chi2.res^2), 9))}**

**comb.VT<-cbind(as.data.frame(comb), serie.VT.comb)**

**# comb.VT**

**VT<-sort(unique(serie.VT.comb))**

**prob.VT<-table(serie.VT.comb)/length(serie.VT.comb)**

**prob.VT**

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[,c(9,10)]**

**addmargins(obs3)**

**chisq.test(obs3, correct = FALSE)**

**nij<-obs3**

**tij <- chisq.test(obs3)$expected**

**chi2.calc<-sum((nij-tij)^2/tij)**

**chi2.calc**

**k<-dim(obs3)[1]**

**c<-dim(obs3)[2]**

**nu=(k-1)\*(c-1)**

**nu**

**pchisq(chi2.calc, nu, lower.tail=FALSE)**

**n11 <- obs3[1,1]**

**n1. <- margin.table(obs3, 1)[1]**

**n2. <- margin.table(obs3, 1)[2]**

**n.1 <- margin.table(obs3, 2)[1]**

**n.2 <- margin.table(obs3, 2)[2]**

**n <- margin.table(obs3)**

**chi2.calc <- (n11-n1.\*n.1/n)^2/(n.1\*n.2\*n1.\*n2./(n^3))**

**chi2.calc**

**chisq.test(obs3)**

**chi2.corr<-sum((abs(nij-tij)-0.5)^2/tij)**

**chi2.corr**

**pchisq(chi2.corr, nu, lower.tail=FALSE)**

### 5.4.3. Tests de comparaisons deux à deux

**cat("Vérifier que le tableau obs3 correspond bien à celui du test dont on souhaite les comparaisons multiples\n")**

**k<-nrow(obs3)**

**c<-ncol(obs3)**

**classe1.G<-NULL; classe2.G<-NULL; classe1.F<-NULL; classe2.F<-NULL; test<-NULL; p.value<-NULL**

**for(col1 in 1:(c-1)){for(col2 in (col1+1):c){for(row1 in 1:(k-1)){for(row2 in (row1+1):k){classe1.G<-c(classe1.G, dimnames(obs3)[[2]][col1]) ; classe2.G<-c(classe2.G, dimnames(obs3)[[2]][col2]) ; classe1.F<-c(classe1.F, dimnames(obs3)[[1]][row1]) ; classe2.F<-c(classe2.F, dimnames(obs3)[[1]][row2]) ; obs<- obs3[c(row1, row2), c(col1, col2)] ; obstheo<-suppressWarnings(chisq.test(obs)$expected) ; n.cases.inf.5<-length(which(obstheo<5)) ; if(n.cases.inf.5 == 0){test<-c(test, "Chi2") ; p.value<-c(p.value, round(suppressWarnings(chisq.test(obs, correct=TRUE)$p.value), 9))} else { test<-c(test, "Fisher.exact") ; p.value<- c(p.value, round(fisher.test(obs)$p.value, 9))}}}}}**

**p.mult<-data.frame(classe1.G, classe2.G, classe1.F, classe2.F, test, p.value)**

**p.mult**

**cat("Vérifier que le tableau obs3 correspond bien à celui du test dont on souhaite les comparaisons multiples\n")**

**k<-nrow(obs3)**

**c<-ncol(obs3)**

**classe1.G<-NULL; classe2.G<-NULL; classe1.F<-NULL; classe2.F<-NULL; test<-NULL; p.value<-NULL**

**for(col1 in 1:(c-1)){for(col2 in (col1+1):c){for(row1 in 1:(k-1)){for(row2 in (row1+1):k){classe1.G<-c(classe1.G, dimnames(obs3)[[2]][col1]) ; classe2.G<-c(classe2.G, dimnames(obs3)[[2]][col2]) ; classe1.F<-c(classe1.F, dimnames(obs3)[[1]][row1]) ; classe2.F<-c(classe2.F, dimnames(obs3)[[1]][row2]) ; obs<- obs3[c(row1, row2), c(col1,col2)] ; test<-c(test,"Fisher.exact") ; p.value<- c(p.value, round(fisher.test(obs)$p.value,9))}}}}**

**p.mult<-data.frame(classe1.G, classe2.G, classe1.F, classe2.F, test, p.value)**

**p.mult**

**p.mult2<-transform(p.mult,Holm=round(p.adjust(p.mult$p.value,method="holm"),6))**

**p.mult2**

## 5.5. Test G

### 5.5.1. Méthode

### 5.5.2. Exemples avec R

**library(MASS)**

**loglm(~1+2,obs3)**

**nij<-obs3**

**tij<-obs3theo**

**Gcalc<-2\*(sum(nij\*log(nij/tij)))**

**Gcalc**

**k<-dim(obs3)[1]**

**c<-dim(obs3)[2]**

**nu=(k-1)\*(c-1)**

**nu**

**pchisq(Gcalc, nu, lower.tail=FALSE)**

**nij<-obs3**

**tij<-obs3theo**

**Gcalc<-2\*(sum(nij\*log(nij/tij)))**

**ni.<-margin.table(obs3,1)**

**n.j<-margin.table(obs3,2)**

**n<-margin.table(obs3)**

**k<-dim(obs3)[1]**

**c<-dim(obs3)[2]**

**Gcorr<-Gcalc / (1+(n\*sum(1/ni.)-1)\*(n\*sum(1/n.j)-1)/(6\*n\*(k-1)\*(c-1)))**

**Gcorr**

**nu=(k-1)\*(c-1)**

**nu**

**pchisq(Gcorr,nu,lower.tail=FALSE)**

### 5.5.3. Tests de comparaisons deux à deux

**library(MASS)**

**cat("Vérifier que le tableau obs3 correspond bien à celui du test dont on souhaite les comparaisons multiples\n")**

**k<-nrow(obs3)**

**c<-ncol(obs3)**

**classe1.G<-NULL; classe2.G<-NULL; classe1.F<-NULL; classe2.F<-NULL; test<-NULL; p.value<-NULL**

**for(col1 in 1:(c-1)){for(col2 in (col1+1):c){for(row1 in 1:(k-1)){for(row2 in (row1+1):k){classe1.G<-c(classe1.G, dimnames(obs3)[[2]][col1]) ; classe2.G<-c(classe2.G, dimnames(obs3)[[2]][col2]) ; classe1.F<-c(classe1.F, dimnames(obs3)[[1]][row1]) ; classe2.F<-c(classe2.F, dimnames(obs3)[[1]][row2]) ; obs<- obs3[c(row1, row2), c(col1,col2)] ; obstheo<-suppressWarnings(chisq.test(obs)$expected) ; n.cases.inf.5<-length(which(obstheo<5)) ; if(n.cases.inf.5 == 0){test<-c(test, "G") ; Gcalc<-loglm(~1+2, obs)$lrt ; p.value<-c(p.value, round(pchisq(Gcalc, 1, lower.tail=FALSE),9))} else { test<-c(test,"Fisher.exact") ; p.value<- c(p.value, round(fisher.test(obs)$p.value,9))}}}}}**

**p.mult<-data.frame(classe1.G, classe2.G, classe1.F, classe2.F, test, p.value)**

**p.mult**

**p.mult2<-transform(p.mult,Holm=round(p.adjust(p.mult$p.value,method="holm"),6))**

**p.mult2**

## 5.6. Test exact de Fisher

### 5.6.1. Tableau de contingence 2 × 2

#### 5.6.1.1. Méthode

#### 5.6.1.2. Exemples avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[,c(7,9)]**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**for(i in 1:length(obs2)){obs2[,i]<-factor(obs2[,i])}**

**obs3<- table(obs2)**

**# obs3<-t(obs3)**

**# obs4<-obs3 ; obs4[,1]<-obs3[,2] ; obs4[,2]<-obs3[,1] ; dimnames(obs4)[[2]][1] <- dimnames(obs3)[[2]][2] ; dimnames(obs4)[[2]][2] <- dimnames(obs3)[[2]][1] ; obs3<-obs4**

**# tempo <- list(c("Non", "Oui"), c("Non", "Oui")) ; names(tempo) <- c("Germination.epi", "Verse"); obs3 <- as.table(matrix(c(53, 4, 37, 5), ncol = 2, dimnames=tempo))**

**addmargins(obs3)**

**cat("Si vous comparez des proportions,\nla colonne de gauche correspond-elle aux proportions testées ?\nSi non : utiliser à convenance les trois lignes ci-dessus en enlevant # en début de ligne\n")**

**graphics.off()**

**pG1.Fi<- obs3[, 1]/ margin.table(obs3, 1)**

**par(mar=c(5.1, 5.1, 4.1, 2.1))**

**barplot(pG1.Fi, xlab = paste(labels(dimnames(obs3))[2], " / ", labels(dimnames(obs3))[1]), xaxt = "n", ylab="Proportion", ylim=range(0, max(pG1.Fi)+ 0.15), cex.lab=2, cex.axis=1.8)**

**position.labels <- barplot(pG1.Fi, plot = FALSE)[]**

**axis(side=1, at = position.labels, labels = c(paste(colnames(obs3)[1], " / ", rownames(obs3)[1]), paste(colnames(obs3)[1], " / ", rownames(obs3)[2])), cex.axis=1.8)**

**windows()**

**par(cex.lab=2, xpd=NA, font=2)**

**mosaicplot(t(obs3), main=NULL, cex.axis=1.5)**

**n11<-obs3[1, 1]**

**n1.<-margin.table(obs3, 1)[1]**

**n21<-obs3[2, 1]**

**n2.<-margin.table(obs3, 1)[2]**

**pG1.F1<-n11/n1.**

**pG1.F2<-n21/n2.**

**t(data.frame(pG1.F1, pG1.F2))**

**n12<-obs3[1,2]**

**n22<-obs3[2,2]**

**n.1<-margin.table(obs3,2)[1]**

**n.2<-margin.table(obs3,2)[2]**

**pG2.F1<-n12/n1.**

**pG2.F2<-n22/n2.**

**pF1.G1<-n11/n.1**

**pF1.G2<-n12/n.2**

**pF2.G1<-n21/n.1**

**pF2.G2<-n22/n.2**

**t(data.frame(pG2.F1, pG2.F2, pF1.G1, pF1.G2, pF2.G1, pF2.G2))**

**fisher.test(obs3)**

**n11<-obs3[1,1]**

**NFE.calc<-n11**

**NFE.calc**

**n1.<-margin.table(obs3,1)[1]**

**n<- margin.table(obs3)**

**n.1<-margin.table(obs3,2)[1]**

**p.droite<-phyper(NFE.calc-1,n1.,n-n1.,n.1,lower.tail=FALSE)**

**p.gauche<-phyper(NFE.calc,n1.,n-n1.,n.1)**

**p.droite**

**p.gauche**

**if(p.droite < p.gauche){p.value1<-p.droite ; NFE.gauche<-NFE.calc ; d.NFE.calc<-round(dhyper(NFE.calc, n1. ,n-n1., n.1),12) ; d.NFE.gauche<-Inf ; while(NFE.gauche >= 0 & d.NFE.gauche > d.NFE.calc){ NFE.gauche<-NFE.gauche - 1 ; d.NFE.gauche<- round(dhyper(NFE.gauche,n1.,n-n1.,n.1),12)} ; if(d.NFE.gauche > d.NFE.calc){p.value2<-0} else{p.value2<- phyper(NFE.gauche,n1.,n-n1.,n.1)}} else{p.value1<-p.gauche ; NFE.droite<-NFE.calc ; d.NFE.calc<-round(dhyper(NFE.calc, n1. ,n-n1., n.1),12) ; d.NFE.droite<-Inf ; while(d.NFE.droite > d.NFE.calc){ NFE.droite<-NFE.droite + 1 ; d.NFE.droite<- round(dhyper(NFE.droite,n1.,n-n1.,n.1),12)} ; p.value2<- phyper(NFE.droite-1,n1.,n-n1.,n.1,lower.tail=FALSE)}**

**p.value<-p.value1+p.value2**

**p.value**

**Pn11<-choose(n1.,n11)\*choose(n-n1.,n.1-n11)/choose(n,n.1)**

**Pn11**

**dhyper(n11,n1.,n-n1.,n.1)**

**n11<-obs3[1,1]**

**n21<-obs3[2,1]**

**n12<-obs3[1,2]**

**n22<-obs3[2,2]**

**odd<-n11\*n22/(n12\*n21)**

**odd**

**obs3<- as.table(matrix(c(5,3,4,2), ncol=2))**

**n11<-obs3[1,1]**

**n1.<- margin.table(obs3, 1)[1]**

**n2.<- margin.table(obs3, 1)[2]**

**n.1<- margin.table(obs3, 2)[1]**

**n<-margin.table(obs3)**

**mat.sup<- matrix(c(1,-1,-1,1), ncol=2)**

**mat.inf<- matrix(c(-1,1,1,-1), ncol=2)**

**obs3.diff<-vector("list", max(obs3)+n11+2)**

**obs3.diff[[1]] <- obs3**

**obs3.tempo <- obs3**

**i=2**

**while(all(obs3.tempo != 0)){obs3.tempo <- obs3.tempo + mat.inf; obs3.diff[[i]] <- obs3.tempo; i=i+1}**

**obs3.tempo<-obs3**

**while(all(obs3.tempo != 0)){obs3.tempo <- obs3.tempo + mat.sup; obs3.diff[[i]] <- obs3.tempo; i=i+1}**

**obs3.diff <- obs3.diff[-(i : (max(obs3)+n11+2))]**

**# obs3.diff**

**vecteur.ini <- c(rep("A", n1.), rep("B", n2.))**

**library(gplots)**

**comb <- as.data.frame(combinations(n, n.1, vecteur.ini, set=FALSE))**

**tempo<-as.data.frame(t(comb))**

**serie.VT.comb<-NULL**

**for(i in 1:length(tempo)){tempo[,i] <- factor(tempo[,i], levels=LETTERS[1:2]); serie.VT.comb <- c(serie.VT.comb, table(tempo[,i])[1])}**

**VT<-sort(unique(serie.VT.comb))**

**prob.VT<-table(serie.VT.comb)/length(serie.VT.comb)**

**prob.VT**

**table.diff <- unique(comb)**

**row.names(table.diff) = NULL**

**tempo<-as.data.frame(t(table.diff))**

**serie.VT<-NULL**

**serie.prob<-NULL**

**for(i in 1:length(tempo)){tempo[,i] <- factor(tempo[,i], levels=LETTERS[1:2]); n11.tempo <- table(tempo[,i])[1]; serie.VT <- c(serie.VT, n11.tempo); serie.prob <- c(serie.prob, dhyper(n11.tempo, n1., n2., n.1))}**

**table.diff.VT.prob<-cbind(table.diff, serie.VT, serie.prob)**

**table.diff.VT.prob**

**fisher.test(obs3,alternative="greater")**

**phyper(NFE.calc-1, n1., n-n1., n.1, lower.tail=FALSE)**

**fisher.test(obs3,alternative="less")**

**phyper(NFE.calc, n1., n-n1., n.1)**

### 5.6.2. Tableau de contingence c × k

#### 5.6.2.1. Méthode

#### 5.6.2.2. Exemple avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[,c(7,8)]**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**for(i in 1:length(obs2)){obs2[,i]<-factor(obs2[,i])}**

**obs3<- table(obs2)**

**# tempo <- list(c("Non", "Oui"), c("Faible", "Fort", "Moyen", "Tres.fort")) ; names(tempo) <- c("Germination.epi", "Enracinement"); obs3 <- as.table(matrix(c(13, 5, 23, 3, 28, 0, 26, 1), ncol = 4, dimnames=tempo))**

**addmargins(obs3)**

**graphics.off()**

**par(cex.lab=2, xpd=NA, font=2)**

**mosaicplot(t(obs3), main=NULL, cex.axis=1.5)**

**fisher.test(obs3)**

#### 5.6.2.3. Tests de comparaisons deux à deux

**p.mult**

**p.mult2<-transform(p.mult,Holm=round(p.adjust(p.mult$p.value,method="holm"),6))**

**p.mult2**

## 5.7. Test de Mantel-Haenszel

### 5.7.1. Méthode

### 5.7.2. Exemples avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[,c(9,10,6)]**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**for(i in 1:length(obs2)){obs2[, i]<-factor(obs2[, i])}**

**obs3<- table(obs2)**

**# obs4<-obs3 ; obs4[1,,]<-obs3[2,,] ; obs4[2,,]<-obs3[1,,] ; dimnames(obs4)[[1]][1] <- dimnames(obs3)[[1]][2] ; dimnames(obs4)[[1]][2] <- dimnames(obs3)[[1]][1] ; obs3<-obs4**

**# obs4<-obs3 ; obs4[, 1,]<-obs3[, 2,] ; obs4[, 2,]<-obs3[, 1,] ; dimnames(obs4)[[2]][1] <- dimnames(obs3)[[2]][2] ; dimnames(obs4)[[2]][2] <- dimnames(obs3)[[2]][1] ; obs3<-obs4**

**# tempo <- list(c("Non", "Oui"), c("Oui", "Non"), c("Jaune", "Jaune.rouge", "Rouge")) ; names(tempo) <- c("Verse", "Attaque", "Couleur"); obs3 <- as.table(array(c(7, 16, 13, 12, 3, 5, 12, 2, 11, 3, 11, 4), dim = c(2, 2, 3), dimnames=tempo))**

**addmargins(obs3)**

**cat("La colonne de gauche correspond-elle aux proportions testées ?\nSi non : utiliser à convenance les trois lignes ci-dessus en enlevant # en début de ligne\n")**

**graphics.off()**

**par(xpd=NA)**

**h<-dim(obs3)[3]**

**row=1 ; col=1 ; while(h > (row \* col)){if(row < col){row<-row+1} else{col<-col+1}} ; par(mfrow=c(row, col))**

**for(i in 1:h){pG1.Fi<- obs3[, 1, i]/ (obs3[, 1, i] + obs3[, 2, i]) ; barplot(pG1.Fi, xlab = paste(labels(dimnames(obs3))[2], " / ", labels(dimnames(obs3))[1]), xaxt = "n", ylab="Proportion", ylim=range(0, max(pG1.Fi)+ 0.15), main=paste(names(dimnames(obs3))[3], "=", dimnames(obs3)[[3]][i]), cex.lab=1.2, cex.axis=1.2, cex=1.2) ; position.labels <- barplot(pG1.Fi, plot = FALSE)[] ; axis(side=1, at = position.labels, labels = c(paste(colnames(obs3)[1], " / ", rownames(obs3)[1]), paste(colnames(obs3)[1], " / ", rownames(obs3)[2])), cex.axis=1.2)}**

**windows()**

**par(xpd=NA, mfrow=c(row, col), cex.lab=1)**

**for(i in 1:h){mosaicplot(t(obs3[, , i]), main=paste(names(dimnames(obs3))[3], "=", dimnames(obs3)[[3]][i]), cex.axis=1)}**

**mantelhaen.test(obs3, exact=FALSE, correct=FALSE)**

**n1.z <- margin.table(obs3[1,,],2)**

**n.1z <- margin.table(obs3[,1,],2)**

**n.2z<- margin.table(obs3[,2,],2)**

**nz <- margin.table(obs3,3)**

**t11z <- n1.z\*n.1z/nz**

**if((sum(t11z) - sum(pmax(0, (n1.z-n.2z)))) >= 5 & (sum(pmin(n1.z, n.1z)) - sum(t11z)) >= 5){cat("Les conditions d'application du test sont remplies\n")} else{cat("Les conditions d'application du test ne sont pas remplies\n")}**

**n11z<-obs3[1,1,]**

**n21z<-obs3[2,1,]**

**n1.z<- margin.table(obs3[1,,],2)**

**n2.z<- margin.table(obs3[2,,],2)**

**pzG1.F1<-n11z/n1.z**

**pzG1.F2<-n21z/n2.z**

**Classes.K<- dimnames(obs3)[[3]]**

**param<-data.frame(Classes.K, pzG1.F1, pzG1.F2)**

**rownames(param)<-NULL**

**param**

**Zcalc<-sum(n11z - t11z) / (sum(n1.z \* n2.z \* n.1z \* n.2z / (nz^2 \* (nz-1))))^0.5**

**Zcalc^2**

**pchisq(Zcalc^2, df=1, lower.tail=FALSE)**

**Znf.calc<- sum(n11z - t11z) /(sum(n1.z \* n2.z \* n.1z \* n.2z / (nz^3)))^0.5**

**Znf.calc^2**

**n11z<-obs3[1, 1, ]**

**n12z<-obs3[1, 2, ]**

**n21z<-obs3[2, 1, ]**

**n22z<-obs3[2, 2, ]**

**oz<-(n11z\*n22z)/(n12z\*n21z)**

**oz**

**odd<-sum(n11z\*n22z/nz)/sum(n12z\*n21z/nz)**

**odd**

**mantelhaen.test(obs3, exact=FALSE, correct=TRUE)**

**if(abs(sum(n11z - t11z))>=0.5){cat("Les conditions sont remplies pour l'utilisation de la correction\n")} else{cat("Attention : les conditions ne sont pas remplies pour l'utilisation de la correction\nLa fonction utilise la VT Z classique\n")}**

**Zcorr <-(abs(sum(n11z - t11z))-0.5) / (sum(n1.z \* n2.z \* n.1z \* n.2z / (nz^2 \* (nz-1))))^0.5**

**Zcorr^2**

**pchisq(Zcorr^2,df=1,lower.tail=FALSE)**

### 5.7.3. Tests de comparaisons deux à deux

**cat("Vérifier que le tableau obs3 correspond bien à celui du test dont on souhaite les comparaisons multiples\n")**

**h<-dim(obs3)[3]**

**h**

**classe<-NULL; test<-NULL; p.value<-NULL**

**for(i in 1:h){classe<- c(classe, dimnames(obs3)[[3]][i]) ; if(length(which(suppressWarnings(chisq.test(obs3[, , i])$expected) >= 5)) == 4){test<-c(test, "Chi2") ; p.value<-c(p.value, round(chisq.test(obs3[, , i], correct=TRUE)$p.value, 9))} else { test<-c(test, "Fisher.exact") ; p.value<- c(p.value, round(fisher.test(obs3[, , i])$p.value, 9))}}**

**# library(MASS) ; classe<-NULL ; test<-NULL ; p.value<-NULL ; for(i in 1:h){classe <- c(classe, dimnames(obs3)[[3]][i]) ; if(length(which(suppressWarnings(chisq.test(obs3[, , i])$expected) >= 5)) == 4){test<-c(test, "G") ; Gcalc<-loglm(~1+2, obs3[, , i])$lrt ; nu<- loglm(~1+2, obs3[, , i])$df ; p.value<-c(p.value, round(pchisq(Gcalc, nu, lower.tail=FALSE), 9))} else { test<-c(test, "Fisher.exact") ; p.value<- c(p.value, round(fisher.test(obs3[, , i])$p.value, 9))}}**

**p.mult<-data.frame(classe, test, p.value)**

**names(p.mult)<-c("classe", "test", "p.value")**

**p.mult**

**cat("Vérifier que le tableau obs3 correspond bien à celui du test dont on souhaite les comparaisons multiples\n")**

**h<-dim(obs3)[3]**

**h**

**classe<-NULL; test<-NULL; p.value<-NULL**

**for(i in 1:h){classe<- c(classe, dimnames(obs3)[[3]][i]) ; test<-c(test,"Fisher.exact") ; p.value<- c(p.value, round(fisher.test(obs3[,,i])$p.value,9))}**

**p.mult<-data.frame(classe,test,p.value)**

**names(p.mult)<-c("classe","test","p.value")**

**p.mult**

**p.mult2<-transform(p.mult, Holm=round(p.adjust(p.mult$p.value,method="holm"),6))**

**p.mult2**

## 5.8. Comparaison d'une proportion observée à une proportion théorique

### 5.8.1. Méthode

### 5.8.2. Exemples avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-data.frame(mais$Attaque)**

**# obs1 <- as.data.frame(matrix(rep(c("Non", "Oui"), times=c(54, 46)), ncol=1))**

**names(obs1)<-c("Attaque")**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**obs2[,1]<-factor(obs2[,1])**

**obs3<- table(obs2, dnn = names(obs1))**

**# obs4<-obs3 ; obs4[1]<-obs3[2] ; obs4[2]<-obs3[1] ; names(obs4)[1]<-names(obs3)[2] ; names(obs4)[2]<-names(obs3)[1] ; obs3<-obs4**

**addmargins(obs3)**

**cat("Votre effectif observé, sur lequel la proportion théorique est fondée, est-il à gauche dans le tableau ?\nSi non : enlever # au début de la ligne ci-dessus\n")**

**graphics.off()**

**p.theo<-0.6**

**pi<- obs3/ margin.table(obs3)**

**par(mar=c(5.1, 5.1, 4.1, 2.1))**

**barplot(pi[1], xlab=names(obs2), ylab="Proportion", ylim=range(0, max(pi, p.theo)+ 0.15), cex.lab=2, cex.axis=1.8, cex=1.8)**

**points(barplot(1,plot=FALSE), p.theo, pch="-", cex=5)**

**p.theo<-0.6**

**n1<-obs3[1]**

**n<-margin.table(obs3)**

**binom.test(n1, n, p.theo)**

**p1<-n1/n**

**p1**

**NB.calc<-n1**

**NB.calc**

**p.droite<-pbinom(NB.calc-1, n, p.theo, lower.tail=FALSE)**

**p.gauche<-pbinom(NB.calc, n, p.theo)**

**p.droite**

**p.gauche**

**if(p.droite < p.gauche){p.value1<-p.droite ; NB.gauche<-NB.calc ; d.NB.calc<-round(dbinom(NB.calc, n, p.theo),12) ; d.NB.gauche <- Inf ; while(NB.gauche >= 0 & d.NB.gauche > d.NB.calc){ NB.gauche <- NB.gauche - 1 ; d.NB.gauche<- round(dbinom(NB.gauche, n, p.theo),12)} ; if(d.NB.gauche > d.NB.calc){p.value2<-0} else{p.value2<- pbinom(NB.gauche, n, p.theo)}} else{p.value1<-p.gauche ; NB.droite<-NB.calc ; d.NB.calc<-round(dbinom(NB.calc, n,p.theo),12) ; d.NB.droite<-Inf ; while(d.NB.droite > d.NB.calc){NB.droite<-NB.droite + 1 ; d.NB.droite <- round(dbinom(NB.droite, n, p.theo), 12)} ; p.value2 <- pbinom(NB.droite-1, n, p.theo, lower.tail=FALSE)}**

**p.value<-p.value1 + p.value2**

**p.value**

**p.theo<-0.6**

**n1<-obs3[1]**

**n<-margin.table(obs3)**

**prop.test(n1, n, p.theo, correct=FALSE)**

**obs3theo<- suppressWarnings(as.table(chisq.test(obs3, p=c(p.theo, (1 - p.theo)))$expected))**

**addmargins(obs3theo)**

**n.cases.inf.5<-length(which(obs3theo<5))**

**if(n.cases.inf.5==0){cat("La condition d'application est remplie\nIl n'y a pas de cases d'effectif théorique strictement inférieur à 5\n")} else{cat("La condition d'application n'est pas remplie\nCertaines cases sont d'effectif théorique strictement inférieur à 5\n")}**

**Zcalc<-(p1-p.theo)/(p.theo\*(1-p.theo)/n)^0.5**

**Zcalc**

**Zcalc^2**

**min(pnorm(Zcalc, lower.tail=FALSE), pnorm(Zcalc))\*2**

**nu=1**

**pchisq(Zcalc^2, nu, lower.tail=FALSE)**

**prop.test(n1, n, p.theo, correct=TRUE)**

**Zcorr<-(abs(p1-p.theo)-0.5/n)/(p.theo\*(1-p.theo)/n)^0.5**

**if(p1<p.theo){Zcorr <- -Zcorr}**

**if(abs(p1-p.theo)<0.5/n){Zcorr<-0}**

**Zcorr**

**nu=1**

**min(pnorm(Zcorr,lower.tail=FALSE), pnorm(Zcorr))\*2**

**pchisq(Zcorr^2,nu,lower.tail=FALSE)**

**p.theo<-0.6**

**n1<-obs3[1]**

**n<-margin.table(obs3)**

**binom.test(n1, n, p.theo, alternative="greater")**

**pbinom(NB.calc-1, n, p.theo, lower.tail=FALSE)**

**p.theo<-0.6**

**n1<-obs3[1]**

**n<-margin.table(obs3)**

**prop.test(n1,n, p.theo, alternative="greater", correct=FALSE)**

**pnorm(Zcalc, lower.tail=FALSE)**

**nu=1**

**if (p1 > p.theo){pchisq(Zcalc^2, nu, lower.tail=FALSE)/2} else {1-pchisq(Zcalc^2, nu, lower.tail=FALSE)/2}**

**p.theo<-0.6**

**n1<-obs3[1]**

**n<-margin.table(obs3)**

**binom.test(n1,n, p.theo,alternative="less")**

**pbinom(NB.calc, n, p.theo)**

**p.theo<-0.6**

**n1<-obs3[1]**

**n<-margin.table(obs3)**

**prop.test(n1,n, p.theo, alternative="less", correct=FALSE)**

**pnorm(Zcalc)**

**nu=1**

**if (p1 < p.theo){pchisq(Zcalc^2, nu, lower.tail=FALSE)/2} else {1-pchisq(Zcalc^2, nu, lower.tail=FALSE)/2}**

## 5.9. Comparaison de deux proportions observées

### 5.9.1. Méthode

### 5.9.2. Exemples avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[,c(9,10)]**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**for(i in 1:length(obs2)){obs2[,i]<-factor(obs2[,i])}**

**obs3<- table(obs2)**

**# obs3<-t(obs3)**

**# obs4<-obs3 ; obs4[,1]<-obs3[,2] ; obs4[,2]<-obs3[,1] ; dimnames(obs4)[[2]][1] <- dimnames(obs3)[[2]][2] ; dimnames(obs4)[[2]][2] <- dimnames(obs3)[[2]][1] ; obs3<-obs4**

**# tempo <- list(c("Non", "Oui"), c("Oui", "Non")) ; names(tempo) <- c("Verse", "Attaque"); obs3 <- as.table(matrix(c(21, 24, 36, 18), ncol = 2, dimnames=tempo))**

**addmargins(obs3)**

**cat("La colonne de gauche correspond-elle aux proportions testées ?\nSi non : utiliser à convenance les trois lignes ci-dessus en enlevant # en début de ligne\n")**

**prop.test(obs3, correct=FALSE)**

**obs3theo<- suppressWarnings(as.table(chisq.test(obs3, p=c(p.theo, (1-p.theo)))$expected))**

**addmargins(obs3theo)**

**n.cases.inf.5<-length(which(obs3theo<5))**

**if(n.cases.inf.5==0){cat("La condition d'application est remplie\nIl n'y a pas de cases d'effectif théorique strictement inférieur à 5\n")} else{cat("La condition d'application n'est pas remplie\nCertaines cases sont d'effectif théorique strictement inférieur à 5\n")}**

**n<-margin.table(obs3)**

**n11<- obs3[1, 1]**

**n21<- obs3[2, 1]**

**n1.<-margin.table(obs3, 1)[1]**

**n2.<-margin.table(obs3, 1)[2]**

**pG1.F1<-n11/n1.**

**pG1.F2<-n21/n2.**

**t(data.frame(pG1.F1, pG1.F2))**

**n12<-obs3[1, 2]**

**n22<-obs3[2, 2]**

**n.1<-margin.table(obs3, 2)[1]**

**n.2<-margin.table(obs3, 2)[2]**

**pG2.F1<-n12/n1.**

**pG2.F2<-n22/n2.**

**pF1.G1<-n11/n.1**

**pF1.G2<-n12/n.2**

**pF2.G1<-n21/n.1**

**pF2.G2<-n22/n.2**

**t(data.frame(pG2.F1, pG2.F2, pF1.G1, pF1.G2, pF2.G1, pF2.G2))**

**p.1<-(n11+n21)/n**

**Zcalc <- (pG1.F1 - pG1.F2) / (p.1 \* (1-p.1) \* (1/n1. + 1/n2.))^0.5**

**Zcalc**

**Zcalc^2**

**min(pnorm(Zcalc, lower.tail=FALSE), pnorm(Zcalc))\*2**

**nu=1**

**pchisq(Zcalc^2, nu, lower.tail=FALSE)**

**prop.test(obs3, correct=TRUE)**

**Zcorr<-(abs(pG1.F1- pG1.F2)-0.5\*(1/n1.+1/n2.))/(p.1\*(1-p.1)\*(1/n1.+1/n2.))^0.5**

**if(pG1.F1< pG1.F2){Zcorr <- -Zcorr}**

**if(abs(pG1.F1- pG1.F2) < 0.5\*(1/n1.+1/n2.)){Zcorr<-0}**

**Zcorr^2**

**min(pnorm(Zcorr, lower.tail=FALSE), pnorm(Zcorr))\*2**

**nu=1**

**pchisq(Zcorr^2, nu, lower.tail=FALSE)**

## 5.10. Comparaison de deux proportions en séries appariées (test de Mac Nemar)

### 5.10.1. Méthode

### 5.10.2. Exemples avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[,c(9,13)]**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**for(i in 1:length(obs2)){obs2[,i]<-factor(obs2[,i])}**

**cat("Mesure1 =", names(obs2)[1], "\n") ; cat("Mesure2 =", names(obs2)[2], "\n") ; cat("Est-ce correct ? Si non : inverser les colonnes à la création de obs1\n")**

**names(obs2)<-c(paste("Mesure1 (", names(obs2)[1], ")", sep=""), paste("Mesure2 (", names(obs2)[2], ")", sep=""))**

**obs3<- table(obs2)**

**# obs3<-t(obs3)**

**# obs4<-obs3 ; obs4[, 1]<-obs3[, 2] ; obs4[, 2]<-obs3[, 1] ; dimnames(obs4)[[2]][1] <- dimnames(obs3)[[2]][2] ; dimnames(obs4)[[2]][2] <- dimnames(obs3)[[2]][1] ; obs3<-obs4**

**# tempo <- list(c("Non", "Oui"), c("Non", "Oui")) ; names(tempo) <- c("Mesure1 (Verse)", "Mesure2 (Verse.Traitement)"); obs3 <- as.table(matrix(c(39, 5, 18, 37), ncol = 2, dimnames=tempo))**

**addmargins(obs3)**

**cat("Le tableau de contingence est-il présenté comme suit :\n(1) La première mesure forme les lignes ?\n(2) (f1.f1) en haut à gauche et (f2.f2) en bas à droite ?\nSi non : utiliser les trois lignes précédentes à convenance en enlevant # en début de ligne\n")**

**graphics.off()**

**n11<-obs3[1, 1] ; n12<- obs3[1, 2] ; n21<- obs3[2, 1] ; n<-margin.table(obs3) ; pF1.G1<-(n11+n12)/n ; pF1.G2<-(n11+n21)/n**

**par(mar=c(5.1, 5.1, 4.1, 2.1))**

**barplot(c(pF1.G1, pF1.G2), xlab = paste(names(dimnames(obs3))[1], "et", names(dimnames(obs3))[2]), xaxt = "n", ylab="Proportion", ylim=range(0, max(pF1.G1, pF1.G2)+ 0.15), cex.lab=2, cex.axis=1.8)**

**position.labels <- barplot(c(pF1.G1, pF1.G2), plot = FALSE)[]**

**axis(side=1, at = position.labels, labels = c(paste(rownames(obs3)[1], " / Mesure1"), paste(rownames(obs3)[1], " / Mesure2")), cex.axis=1.8)**

**windows()**

**par(cex.lab=2, xpd=NA, font=2)**

**mosaicplot(t(obs3), main=NULL, cex.axis=1.4)**

**windows()**

**ns<-n12+n21 ; pf1.f2<-n12/ns ; pf2.f1<-n21/ns**

**par(mar=c(5.1, 5.1, 4.1, 2.1))**

**barplot(c(pf1.f2, pf2.f1), xlab ="", xaxt = "n", ylab="Proportion", ylim=range(0, max(pf1.f2, pf2.f1)+ 0.15), cex.lab=2, cex.axis=1.8)**

**position.labels <- barplot(c(pf1.f2, pf2.f1), plot = FALSE)[]**

**axis(side=1, at = position.labels, labels = c("pf1.f2", "pf2.f1"), cex.axis=1.8)**

**n12<- obs3[1, 2]**

**n21<- obs3[2, 1]**

**ns<-n12+n21**

**if(ns > 20){cat("ns = ", ns, ". La fonction mcnemar.test() peut être utilisée\n", sep="")} else{cat("ns = ", ns, ". La fonction mcnemar.test() ne peut pas être utilisée\n", sep="")}**

**mcnemar.test(obs3, correct=FALSE)**

**n11<-obs3[1, 1]**

**n<-margin.table(obs3)**

**pF1.G1<-(n11+n12)/n**

**pF1.G2<-(n11+n21)/n**

**t(data.frame(pF1.G1, pF1.G2))**

**pf1.f2<-n12/ns**

**pf2.f1<-n21/ns**

**t(data.frame(pf1.f2, pf2.f1))**

**Z2.calc<-(n12-n21)^2/ns**

**Z2.calc**

**nu=1**

**pchisq(Z2.calc, nu, lower.tail=FALSE)**

**Zcalc<-(n12-ns/2)/(ns/4)^0.5**

**min(pnorm(Zcalc,lower.tail=FALSE), pnorm(Zcalc))\*2**

**mcnemar.test(obs3, correct=TRUE)**

**Z2.corr<-(abs(n12-n21)-1)^2/(n12+n21)**

**Z2.corr**

**nu=1**

**pchisq(Z2.corr, nu, lower.tail=FALSE)**

**Zcorr<-(abs(n12-ns/2)-0.5)/(ns/4)^0.5**

**min(pnorm(Zcorr, lower.tail=FALSE), pnorm(Zcorr))\*2**

**binom.test(n12,ns, 0.5)**

**NB.calc<-n12**

**NB.calc**

**p.droite<-pbinom(NB.calc-1,ns,0.5,lower.tail=FALSE)**

**p.gauche<-pbinom(NB.calc,ns,0.5)**

**p.droite**

**p.gauche**

**if(p.droite < p.gauche){p.value1<-p.droite ; NB.gauche<-NB.calc ; d.NB.calc<-round(dbinom(NB.calc, ns,0.5),12) ; d.NB.gauche<-Inf ; while(NB.gauche >= 0 & d.NB.gauche > d.NB.calc){ NB.gauche<-NB.gauche - 1 ; d.NB.gauche<- round(dbinom(NB.gauche,ns,0.5),12)} ; if(d.NB.gauche > d.NB.calc){p.value2<-0} else{p.value2<- pbinom(NB.gauche,ns,0.5)}} else{p.value1<-p.gauche ; NB.droite<-NB.calc ; d.NB.calc<-round(dbinom(NB.calc, ns,0.5),12) ; d.NB.droite<-Inf ; while(d.NB.droite > d.NB.calc){ NB.droite<-NB.droite + 1 ; d.NB.droite<- round(dbinom(NB.droite,ns,0.5),12)} ; p.value2<- pbinom(NB.droite-1,ns,0.5,lower.tail=FALSE)}**

**p.value<-p.value1+p.value2**

**p.value**

## 5.11. Comparaison de plusieurs proportions observées

### 5.11.1. Méthode

### 5.11.2. Exemple avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[,c(6,9)]**

**# obs1[,1]<-ordered(obs1[,1], levels=c("Faible", "Moyen", "Fort", "Tres.fort"))**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**for(i in 1:length(obs2)){obs2[,i]<-factor(obs2[,i])}**

**obs3<- table(obs2)**

**# obs3<-t(obs3)**

**# obs4<-obs3 ; obs4[,1]<-obs3[,2] ; obs4[,2]<-obs3[,1] ; dimnames(obs4)[[2]][1] <- dimnames(obs3)[[2]][2] ; dimnames(obs4)[[2]][2] <- dimnames(obs3)[[2]][1] ; obs3<-obs4**

**# obs3<-as.table(matrix(c(5, 10, 50, 20, 10, 20, 100, 40), ncol=2))**

**# tempo <- list(c("Jaune", "Jaune.rouge", "Rouge"), c("Non", "Oui")) ; names(tempo) <- c("Couleur", "Verse"); obs3 <- as.table(matrix(c(20, 15, 22, 28, 7, 7), ncol = 2, dimnames=tempo))**

**addmargins(obs3)**

**cat("La colonne de gauche correspond-elle aux proportions testées ?\nSi non : utiliser à convenance les trois lignes de modification du tableau de contingence ci-dessus en enlevant # en début de ligne\n")**

**graphics.off()**

**pG1.Fi <- obs3[, 1]/ margin.table(obs3, 1)**

**par(mar=c(5.1, 5.1, 4.1, 2.1))**

**barplot(pG1.Fi, xlab = paste(labels(dimnames(obs3))[2], " / ", labels(dimnames(obs3))[1]), xaxt = "n", ylab="Proportion", ylim=range(0, max(pG1.Fi)+ 0.15), cex.lab=2, cex.axis=1.7)**

**position.labels <- barplot(pG1.Fi, plot = FALSE)[]**

**noms.proportions <- NULL**

**for(i in 1:dim(obs3)[1]){noms.proportions <- c(noms.proportions, paste(colnames(obs3)[1], " / ", rownames(obs3)[i]))}**

**axis(side=1, at = position.labels, labels = noms.proportions, cex.axis=1.7)**

**windows()**

**par(cex.lab=2, xpd=NA, font=2)**

**mosaicplot(t(obs3), main=NULL, cex.axis=1.5)**

**prop.test(obs3)**

**n<- margin.table(obs3)**

**ni1<- obs3[,1]**

**ni.<- margin.table(obs3,1)**

**pG1.Fi<- ni1/ni.**

**print(pG1.Fi)**

**pi.<- ni./n**

**p.1<- sum(pG1.Fi\*pi.)**

**chi2.calc <- n\*sum(pi.\*(pG1.Fi-p.1)^2/(p.1\*(1-p.1)))**

**chi2.calc**

**k<-dim(obs3)[1]**

**nu<-k-1**

**nu**

**pchisq(chi2.calc,nu,lower.tail=FALSE)**

**pFi.G1 <- obs3[, 1]/ margin.table(obs3, 2)[1]**

**print(pFi.G1)**

**pFi.G2 <- obs3[, 2]/ margin.table(obs3, 2)[2]**

**print(pFi.G2)**

**sum(pFi.G1)**

**sum(pFi.G2)**

**windows()**

**par(mar=c(5.1, 5.1, 8, 2.1))**

**barplot(t(matrix(c(pFi.G1, pFi.G2), ncol=2)), beside=TRUE, col=c("grey", "black"), xlab =names(dimnames(obs3))[1], xaxt = "n", ylab="Proportion", cex.lab=2, cex.axis=1.6)**

**position.labels <- barplot(t(matrix(c(pFi.G1, pFi.G2), ncol=2)), beside=TRUE, plot = FALSE)[] ; position.labels <- margin.table(position.labels, 2)/2**

**axis(side=1, at = position.labels, labels = dimnames(obs3)[[1]], cex.axis=1.6)**

**legend("topright", dimnames(obs3)[[2]], title=names(dimnames(obs3))[2], inset=c(0.2, -0.35), pch=15, pt.cex=3, col=c("grey", "black"), bty="n", cex=1.8, y.intersp=1.1, xpd=NA)**

### 5.11.3. Tests de comparaisons deux à deux

**cat("Vérifier que le tableau obs3 correspond bien à celui du test dont on souhaite les comparaisons multiples\n")**

**pairwise.prop.test(obs3, p.adjust="none")**

**k<-dim(obs3)[1]**

**proportion.1<-NULL; proportion.2<-NULL; p.value<-NULL**

**for(row1 in 1:(k-1)){for(row2 in (row1+1):k){ proportion.1<-c(proportion.1, dimnames(obs3)[[1]][row1]) ; proportion.2<-c(proportion.2, dimnames(obs3)[[1]][row2]) ; obs<- obs3[c(row1, row2),] ; p.value<-c(p.value, round(chisq.test(obs, correct=TRUE)$p.value, 9))}}**

**p.mult<-data.frame(proportion.1, proportion.2, p.value)**

**p.mult**

**k<-dim(obs3)[1]**

**proportion.1<-NULL; proportion.2<-NULL; test<-NULL; p.value<-NULL**

**for(row1 in 1:(k-1)){for(row2 in (row1+1):k){ proportion.1<-c(proportion.1, dimnames(obs3)[[1]][row1]) ; proportion.2<-c(proportion.2, dimnames(obs3)[[1]][row2]) ; obs<- obs3[c(row1, row2), ] ; suppressWarnings(obstheo<- chisq.test(obs)$expected) ; n.cases.inf.5<-length(which(obstheo<5)) ; if(n.cases.inf.5 == 0){test<-c(test, "Chi2") ; p.value<-c(p.value, round(chisq.test(obs, correct=TRUE)$p.value, 9))} else { test<-c(test, "Fisher.exact") ; p.value<- c(p.value, round(fisher.test(obs)$p.value, 9))}}}**

**p.mult<-data.frame(proportion.1, proportion.2, test, p.value)**

**p.mult**

**k<-dim(obs3)[1]**

**proportion.1<-NULL; proportion.2<-NULL; test<-NULL; p.value<-NULL**

**for(row1 in 1:(k-1)){for(row2 in (row1+1):k){ proportion.1<-c(proportion.1, dimnames(obs3)[[1]][row1]) ; proportion.2<-c(proportion.2, dimnames(obs3)[[1]][row2]) ; obs<- obs3[c(row1, row2),] ; test<-c(test,"Fisher.exact") ; p.value<- c(p.value, round(fisher.test(obs)$p.value,9))}}**

**p.mult<-data.frame(proportion.1, proportion.2, test, p.value)**

**p.mult**

**pairwise.prop.test(obs3, p.adjust="holm")**

**k<-nrow(p.mult)**

**k**

**tempo<-sort(p.mult$p.value)**

**tempo2<-NULL**

**for(i in 1:k){tempo2[i]<-tempo[i]\*(k+1-i)}**

**for(i in 2:k){if(tempo2[i] <= tempo2[i-1]){ tempo2[i]<-tempo2[i-1]}}**

**for(i in 1:k){if(tempo2[i] > 1){ tempo2[i]<-1}}**

**p.mult2<-transform(p.mult,Holm=tempo2[rank(p.mult$p.value)])**

**p.mult2**

**pairwise.prop.test(obs3, p.adjust="bonferroni")**

**k<-nrow(p.mult2)**

**k**

**Bonferroni<-p.mult$p.value \* k**

**Bonferroni[which(Bonferroni > 1)]=1**

**p.mult3<-data.frame(p.mult2,Bonferroni)**

**p.mult3**

## 5.12. Comparaison de plusieurs proportions observées à plusieurs proportions théoriques

### 5.12.1. Méthode

### 5.12.2. Exemple avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[,c(6,9)]**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**for(i in 1:length(obs2)){obs2[,i]<-factor(obs2[,i])}**

**obs3<- table(obs2)**

**# obs3<-t(obs3)**

**# obs4<-obs3 ; obs4[,1]<-obs3[,2] ; obs4[,2]<-obs3[,1] ; dimnames(obs4)[[2]][1] <- dimnames(obs3)[[2]][2] ; dimnames(obs4)[[2]][2] <- dimnames(obs3)[[2]][1] ; obs3<-obs4**

**# tempo <- list(c("Jaune", "Jaune.rouge", "Rouge"), c("Non", "Oui")) ; names(tempo) <- c("Couleur", "Verse"); obs3 <- as.table(matrix(c(20, 15, 22, 28, 7, 7), ncol = 2, dimnames=tempo))**

**addmargins(obs3)**

**cat("La colonne de gauche correspond-elle aux proportions testées ?\nSi non : utiliser à convenance les trois lignes ci-dessus en enlevant # en début de ligne\n")**

**graphics.off()**

**pG1.Fi<- obs3[, 1]/ margin.table(obs3, 1)**

**pG1.Fitheo<-c(0.5, 0.6, 0.4)**

**par(mar=c(5.1, 5.1, 4.1, 2.1))**

**barplot(t(matrix(c(pG1.Fi, pG1.Fitheo), ncol=2)), beside=TRUE, col=c("grey", "black"), xlab = paste(labels(dimnames(obs3))[2], " / ", labels(dimnames(obs3))[1]), xaxt = "n", ylab=" ", ylim=range(0, max(pG1.Fi, pG1.Fitheo) + 0.3), cex.lab=2, cex.axis=1.8)**

**position.labels <- barplot(t(matrix(c(pG1.Fi, pG1.Fitheo), ncol=2)), beside=TRUE, plot = FALSE)[] ; position.labels <- margin.table(position.labels, 2)/2**

**noms.proportions <- NULL**

**for(i in 1:dim(obs3)[1]){noms.proportions <- c(noms.proportions, paste(colnames(obs3)[1], " / ", rownames(obs3)[i]))}**

**axis(side=1, at = position.labels, labels = noms.proportions, cex.axis=1.8)**

**legend("topright", c("Proportions observées", "Proportions théoriques"), pch=15, pt.cex=3, col=c("grey", "black"), bty="n", cex=2, y.intersp=1.5)**

**windows()**

**par(mfrow=c(1, 2), cex.main=2, cex.lab=2, xpd=NA, font=2)**

**mosaicplot(t(obs3), main="Observé", cex.axis=1.5)**

**ni.<-margin.table(obs3, 1)**

**pG1.Fitheo<-c(0.5, 0.6, 0.4)**

**obs3theo<-obs3**

**obs3theo[, 1]<- pG1.Fitheo \* ni.**

**obs3theo[, 2]<- (1-pG1.Fitheo) \* ni.**

**mosaicplot(t(obs3theo), main="Théorique", cex.axis=1.5)**

**pG1.Fitheo<-c(0.5,0.6,0.4)**

**prop.test(obs3, p=pG1.Fitheo)**

**obs3theo<- suppressWarnings(chisq.test(obs3, p= pG1.Fitheo)$expected)**

**addmargins(as.table(obs3theo))**

**ni1<- obs3[,1]**

**ni.<- margin.table(obs3,1)**

**pG1.Fi<- ni1/ni.**

**print(pG1.Fi)**

**n<- margin.table(obs3)**

**pi.<- ni./n**

**chi2.calc <- n\*sum(pi.\*(pG1.Fi - pG1.Fitheo)^2/(pG1.Fitheo \* (1 - pG1.Fitheo)))**

**chi2.calc**

**k<-dim(obs3)[1]**

**nu<-k**

**nu**

**pchisq(chi2.calc, nu, lower.tail=FALSE)**

### 5.12.3. Tests de comparaisons deux à deux

**cat("Vérifier que le tableau obs3 correspond bien à celui du test dont on souhaite les comparaisons multiples\n")**

**k<-dim(obs3)[1]**

**pG1.Fitheo<-c(0.5,0.6,0.4)**

**ni.<- margin.table(obs3,1)**

**classe<-NULL; p.value<-NULL**

**for(i in 1:k){classe<- c(classe, dimnames(obs3)[[1]][i]) ; obs<- obs3[i, ] ; obstheo<- suppressWarnings(chisq.test(obs, p=c(pG1.Fitheo[i], 1-pG1.Fitheo[i]))$expected) ; n.cases.inf.5<-length(which(obstheo<5)) ; if(n.cases.inf.5 == 0){ p.value<-c(p.value, round(chisq.test(obs, p=c(pG1.Fitheo[i], 1-pG1.Fitheo[i]))$p.value, 9))} else {p.value<- c(p.value, NA)}}**

**p.mult<-data.frame(classe, p.value)**

**p.mult**

**p.mult2<-transform(p.mult, Holm=round(p.adjust(p.mult$p.value,method="holm"),6))**

**p.mult2**

## 5.13. Le test t de Student de comparaison de moyennes

### 5.13.1. Comparaison d'une moyenne observée à une valeur théorique

#### 5.13.1.1. Méthode

#### 5.13.1.2. Exemples avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-data.frame(mais[which(mais$Parcelle=="Est"), "Hauteur"])**

**names(obs1)<-c("Hauteur")**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**obs2**

**graphics.off()**

**zone<-matrix(c(1,1,1:4), ncol=2) ; layout(zone, widths=c(1,1.5), heights=c(2,1,1)) ; par(cex=5, lwd=10) ; layout.show(max(zone))**

**windows()**

**layout(zone, widths=c(1,1.5), heights=c(2,1,1))**

**par(mar=c(5.1, 6, 4.1, 2.1), mgp=c(3.5,1,0))**

**m.theo<-265**

**library(gplots)**

**barplot2(mean(obs2[, 1]), xlim=c(0, 1.5), ylim= range(0, m.theo, obs2[, 1], max(m.theo, obs2[, 1])\*1.2), names.arg="", plot.ci=TRUE, ci.l= (mean(obs2[, 1])-sd(obs2[, 1])), ci.u= (mean(obs2[,1])+sd(obs2[,1])), xpd=FALSE, ylab=paste("Moyenne +/- écart type de ", names(obs1)[1]), cex.lab=2.2, cex.axis=2.2, ci.lwd=2)**

**abline(h=m.theo, lty="77", lwd=2)**

**hist<-hist(obs2[, 1], freq=FALSE, main=NULL, xlab= names(obs1), ylab="Densité de la distribution", cex.lab=1.8, cex.axis=1.8)**

**par(cex.lab=1.8, cex.axis=1.8, bty="n")**

**stripchart(obs2[,1], method="jitter", jitter=1, vertical=FALSE, xlab=names(obs1)[1], pch=16, cex=1.2, xlim=range(hist$breaks))**

**abline(v=mean(obs2[,1]), lwd=2, col=gray(0.5))**

**par(bty="n")**

**boxplot(obs2[,1], horizontal=TRUE, xlab=names(obs1)[1], pch=16, cex=1.2, cex.lab=1.8, cex.axis=1.8, bty="l", ylim=range(hist$breaks))**

**abline(v=mean(obs2[,1]), lwd=2, col=gray(0.5))**

**m.theo<-265**

**t.test(obs1,mu=m.theo)**

**n<-length(obs2[, 1])**

**m<- mean(obs2[, 1])**

**s<- sd(obs2[, 1])**

**param <- data.frame(n, m, s)**

**names(param) <- c("Effectif", "Moyenne", "Ecart.type")**

**param**

**m.theo<-265**

**t.calc<- (m-m.theo)/(s/n^0.5)**

**t.calc**

**min(pt(t.calc, n-1, lower.tail=FALSE), pt(t.calc, n-1))\*2**

**m+qt(0.025,n-1)\*s/n^0.5**

**m+qt(0.025,n-1,lower.tail=FALSE)\*s/n^0.5**

**m.theo<-265**

**t.test(obs1,mu=m.theo, alternative="greater")**

**pt(t.calc, n-1, lower.tail=FALSE)**

**m+qt(0.05,n-1)\*s/n^0.5**

**Mcalc<-m**

**Mcalc**

**pnorm(Mcalc, m.theo, (s/n^0.5), lower.tail=FALSE)**

**m.theo<-265**

**Zcalc<- (m-m.theo)/(s/n^0.5)**

**Zcalc**

**pnorm(Zcalc, lower.tail=FALSE)**

**m.theo<-265**

**t.test(obs1,mu=m.theo, alternative="less")**

**pt(t.calc, n-1)**

**m+qt(0.05,n-1,lower.tail=FALSE)\*s/n^0.5**

### 5.13.2. Comparaison de deux moyennes observées

#### 5.13.2.1. Méthode

#### 5.13.2.2. Exemple avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[which(mais$Parcelle=="Nord" | mais$Parcelle=="Sud"), c(2,11)]**

**obs3**

**t.test(obs3[,1], obs3[,2], var.equal=TRUE)**

**param**

**s2<-((n1-1)\*s.1^2+(n2-1)\*s.2^2)/(n1+n2-2)**

**t.calc<-(m1-m2)/(s2\*(1/n1+1/n2))^0.5**

**t.calc**

**nu<-n1+n2-2**

**nu**

**min(pt(t.calc, nu, lower.tail=FALSE), pt(t.calc, nu))\*2**

### 5.13.3. Comparaison de deux moyennes observées avec variances différentes (test de Welch)

#### 5.13.3.1. Méthode

#### 5.13.3.2. Exemples avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[which(mais$Parcelle=="Nord" | mais$Parcelle=="Sud"), c(3,11)]**

**obs1[,2]<-factor(obs1[,2])**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**nc.max<-max(table(obs2[,2]))**

**nb.na<-nc.max- table(obs2[,2])**

**tempo<-split(obs2[,1], obs2[,2])**

**for(i in 1:length(tempo)){tempo[[i]]<-append(tempo[[i]],rep(NA,nb.na[i]))}**

**obs3<-data.frame(tempo)**

**obs3**

**graphics.off()**

**zone<-matrix(1:4, ncol=2) ; layout(zone) ; par(cex=5, lwd=10) ; layout.show(max(zone))**

**windows()**

**layout(zone)**

**par(mar=c(5.1, 6, 3, 2.1), mgp=c(3.5,1,0))**

**hist(split(obs2[, 1],obs2[, 2])[[1]], freq=FALSE, main=NULL, xlab= levels(obs2[, 2])[1], ylab="Densité de la distribution", cex.lab=1.5, cex.axis=1.5)**

**hist(split(obs2[, 1],obs2[, 2])[[2]], freq=FALSE, main=NULL, xlab= levels(obs2[, 2])[2], ylab="Densité de la distribution", cex.lab=1.5, cex.axis=1.5)**

**par(cex.lab=1.5, cex.axis=1.5, bty="l")**

**stripchart(obs2[,1]~obs2[,2], method="jitter", jitter=0.1, vertical=FALSE, ylim=c(0.5,2.5), group.names=levels(obs2[,2]), xlab=names(obs2)[1], ylab=names(obs2)[2], pch=16, cex=1.2)**

**m<-sapply(split(obs2[, 1], obs2[, 2]), mean)**

**segments(m[1], 0.6, m[1], 1.4, lwd=3, col=gray(0.5))**

**segments(m[2], 1.6, m[2], 2.4, lwd=3, col=gray(0.5))**

**par(cex.lab=1.5, cex.axis=1.5, bty="l")**

**boxplot(obs2[,1] ~ obs2[,2], horizontal=TRUE, xlab=names(obs2)[1], ylab=names(obs2)[2], pch=16, cex=1.2, boxwex = 0.5)**

**m<-sapply(split(obs2[, 1], obs2[, 2]), mean)**

**segments(m[1], 0.6, m[1], 1.4, lwd=3, col=gray(0.5))**

**segments(m[2], 1.6, m[2], 2.4, lwd=3, col=gray(0.5))**

**windows()**

**par(mar=c(5.1, 6, 4.1, 2.1))**

**library(gplots)**

**m<-sapply(split(obs2[, 1], obs2[, 2]), mean)**

**s<- sapply(split(obs2[, 1], obs2[, 2]), sd)**

**barplot2(m, xlim=c(0, 4), ylim= range(0, obs2[, 1], max(obs2[, 1])\*1.2), space=0.6, names.arg=levels(obs2[, 2]), plot.ci=TRUE, ci.l=(m-s), ci.u= (m+s), xpd=FALSE, xlab=names(obs2)[2], ylab=paste("Moyenne +/- écart type de", names(obs2)[1]), cex.lab=1.5, cex.axis=1.5,cex.names=1.5, ci.lwd=2)**

**t.test(obs3[, 1], obs3[, 2], var.equal=FALSE)**

**n1<-sapply(split(obs2[, 1], obs2[, 2]), length)[1]**

**n2<-sapply(split(obs2[, 1], obs2[, 2]), length)[2]**

**m1<-sapply(split(obs2[, 1], obs2[, 2]), mean)[1]**

**m2<-sapply(split(obs2[, 1], obs2[, 2]), mean)[2]**

**s.1<- sapply(split(obs2[, 1], obs2[, 2]), sd)[1]**

**s.2<- sapply(split(obs2[, 1], obs2[, 2]), sd)[2]**

**param <- data.frame(c(n1, n2), c(m1, m2), c(s.1, s.2))**

**names(param) <- c("Effectif", "Moyenne", "Ecart.type")**

**row.names(param) <- levels(obs2[, 2])**

**param**

**t.calc<-(m1-m2)/(s.1^2/n1+s.2^2/n2)^0.5**

**t.calc**

**nu<-(s.1^2/n1+s.2^2/n2)^2/((s.1^2/n1)^2/(n1-1)+(s.2^2/n2)^2/(n2-1))**

**nu**

**min(pt(t.calc, nu, lower.tail=FALSE), pt(t.calc, nu))\*2**

**t.test(obs3[,1], obs3[,2], var.equal=FALSE, alternative="greater")**

**pt(t.calc, nu, lower.tail=FALSE)**

**t.calc.inv<-(m2-m1)/(s.1^2/n1+s.2^2/n2)^0.5**

**t.calc.inv**

**pt(t.calc.inv,nu)**

**t.test(obs3[, 2], obs3[,1], var.equal=FALSE, alternative="less")$p.value**

**t.test(obs3[,1], obs3[,2], var.equal=FALSE, alternative="less")**

**pt(t.calc,nu)**

**t.calc.inv<-(m2-m1)/(s.1^2/n1+s.2^2/n2)^0.5**

**t.calc.inv**

**pt(t.calc.inv,nu,lower.tail=FALSE)**

**t.test(obs3[,2], obs3[,1],var.equal=FALSE, alternative="greater")$p.value**

### 5.13.4. Comparaison de deux moyennes observées en séries appariées

#### 5.13.4.1. Méthode

#### 5.13.4.2. Exemple avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[which(mais$Parcelle=="Nord"), c(2,12)]**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**obs3<-transform(obs2,difference= obs2[,1]- obs2[,2])**

**obs3**

**obs4<-data.frame(c(obs3[,1], obs3[,2]), c(rep(names(obs3)[1], length(obs3[, 1])), rep(names(obs3)[2], length(obs3[, 2]))))**

**names(obs4)<-c("var.quanti","var.quali")**

**graphics.off()**

**zone<-matrix(1:4, ncol=2) ; layout(zone) ; par(cex=5, lwd=10) ; layout.show(max(zone))**

**windows()**

**layout(zone)**

**par(mar=c(5.1, 6, 3, 2.1), mgp=c(3.5, 1, 0))**

**hist(split(obs4[, 1], obs4[, 2])[[1]], freq=FALSE, main=NULL, xlab= levels(obs4[, 2])[1], ylab="Densité de la distribution", cex.lab=1.5, cex.axis=1.5)**

**hist(split(obs4[, 1], obs4[, 2])[[2]], freq=FALSE, main=NULL, xlab= levels(obs4[, 2])[2], ylab="Densité de la distribution", cex.lab=1.5, cex.axis=1.5)**

**par(cex.lab=1.5, cex.axis=1.5, bty="l")**

**stripchart(obs4[, 1]~obs4[, 2], method="jitter", jitter=0.1, vertical=FALSE, ylim=c(0.5, 2.5), group.names=levels(obs4[, 2]), xlab="", pch=16, cex=1.2)**

**m<-sapply(split(obs4[, 1], obs4[, 2]), mean)**

**segments(m[1], 0.6, m[1], 1.4, lwd=3, col=gray(0.5))**

**segments(m[2], 1.6, m[2], 2.4, lwd=3, col=gray(0.5))**

**par(cex.lab=1.5, cex.axis=1.5, bty="l")**

**boxplot(obs4[, 1] ~ obs4[, 2], horizontal=TRUE, pch=16, cex=1.2, boxwex = 0.5)**

**m<-sapply(split(obs4[, 1], obs4[, 2]), mean)**

**segments(m[1], 0.6, m[1], 1.4, lwd=3, col=gray(0.5))**

**segments(m[2], 1.6, m[2], 2.4, lwd=3, col=gray(0.5))**

**windows()**

**library(gplots)**

**m<-sapply(split(obs4[, 1], obs4[, 2]), mean)**

**s<- sapply(split(obs4[, 1], obs4[, 2]), sd)**

**barplot2(m, xlim=c(0, 4), ylim= range(0, obs4[, 1], max(obs4[, 1])\*1.2), space=0.6, names.arg=levels(obs4[, 2]), plot.ci=TRUE, ci.l=(m-s), ci.u= (m+s), xpd=FALSE, ylab="Moyenne +/- écart type", cex.lab=1.5, cex.axis=1.5, cex.names=1.5, ci.lwd=2)**

**windows()**

**zone<-matrix(1:2, ncol=1) ; layout(zone, heights=c(3, 1)) ; par(cex=5, lwd=10) ; layout.show(max(zone))**

**windows()**

**layout(zone, heights=c(3, 1))**

**par(mar=c(5.1, 6, 3, 2.1))**

**hist<-hist(obs3[, 3], freq=FALSE, main=NULL, xlab="Différences", ylab="Densité", cex.lab=1.5, cex.axis=1.5)**

**par(cex.lab=1.5, cex.axis=1.5, bty="l", xpd=NA)**

**stripchart(obs3[, 3], method="stack", pch=16, cex=1.2, xlim=range(obs3[, 3]), ylim=c(0, 3), xlab="Différences")**

**md<-mean(obs3[, 3])**

**segments(md, 0, md, 10, lwd=4, col=gray(0.5))**

**t.test(obs1[,1], obs1[,2], paired=TRUE)**

**n <- length(obs2[, 1])**

**m1 <- colMeans(obs2)[1]**

**m2 <- colMeans(obs2)[2]**

**param <- data.frame(c(n, n), c(m1, m2))**

**names(param) <- c("Effectif", "Moyenne")**

**row.names(param) <- levels(obs2[, 2])**

**param**

**di<-obs3[,3]**

**md<-mean(di)**

**sd<- sd(di)**

**t.calc<- md/(sd/n^0.5)**

**t.calc**

**min(pt(t.calc, n-1, lower.tail=FALSE), pt(t.calc, n-1))\*2**

## 5.14. Comparaison d'au moins deux moyennes observées

### 5.14.1. Anova (analyse de variances à un facteur)

#### 5.14.1.1. Méthode

#### 5.14.1.2. Exemple avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[, c(2,11)]**

**obs3**

**oneway.test(obs1[,1] ~ obs1[,2], var.equal=TRUE)**

**anova(lm(obs1[,1] ~ obs1[,2]))**

**param**

**k<-nlevels(obs2[,2])**

**xi<- obs2[,1]**

**m<-mean(xi)**

**SCT<-sum((xi-m)^2)**

**SCT**

**SCI<-sum(nc\*(mc-m)^2)**

**SCI**

**SCE<-0**

**for(i in 1:k){SCE<-SCE+sum((na.omit(obs3[i])-mc[i])^2)}**

**SCE**

**SCI+SCE**

**n<-length(obs2[,1])**

**Fcalc<-(SCI/(k-1))/(SCE/(n-k))**

**Fcalc**

**pf(Fcalc, (k-1), (n-k), lower.tail=FALSE)**

### 5.14.2. Anova avec variances différentes (correction de Welch)

#### 5.14.2.1. Méthode

#### 5.14.2.2. Exemple avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[, c(3,11)]**

**obs1[,2]<-factor(obs1[,2])**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**nc.max<-max(table(obs2[,2]))**

**nb.na<-nc.max- table(obs2[,2])**

**tempo<-split(obs2[,1], obs2[,2])**

**for(i in 1:length(tempo)){tempo[[i]]<-append(tempo[[i]],rep(NA,nb.na[i]))}**

**obs3<-data.frame(tempo)**

**obs3**

**graphics.off()**

**k<-nlevels(obs2[, 2])**

**row=1 ; col=1 ; while(k > (row \* col)){if(row < col){row<-row+1} else{col<-col+1}} ; par(mfrow=c(row, col)) ; for(i in 1:k){hist(split(obs2[, 1],obs2[, 2])[[i]], freq=FALSE, main=NULL, xlab= levels(obs2[, 2])[i], ylab="Densité", cex.lab=1.5, cex.axis=1.5)}**

**windows()**

**par(mar=c(5.1, 6, 3, 2.1), cex.lab=1.5, cex.axis=1.5, bty="l")**

**stripchart(obs2[, 1]~obs2[, 2], method="jitter", jitter=0.1, vertical=FALSE, ylim=range(0.5, k+0.5), group.names=levels(obs2[, 2]), xlab= names(obs2)[1], ylab=names(obs2)[2], pch=16, cex=1.2)**

**mc<-sapply(split(obs2[, 1], obs2[, 2]), mean)**

**for(i in 1:k){segments(mc[i], i-0.25, mc[i], i+0.25, lwd=3, col=gray(0.5))}**

**windows()**

**par(mar=c(5.1, 6, 3, 2.1), cex.lab=1.5, cex.axis=1.5, bty="l")**

**boxplot(obs2[, 1] ~ obs2[, 2], horizontal=TRUE, xlab= names(obs2)[1], ylab=names(obs2)[2], pch=16, cex=1.2, boxwex = 0.5)**

**for(i in 1:k){segments(mc[i], i-0.3, mc[i], i+0.3, lwd=3, col=gray(0.5))}**

**windows()**

**library(gplots)**

**mc<-sapply(split(obs2[, 1], obs2[, 2]), mean)**

**sc<- sapply(split(obs2[, 1], obs2[, 2]), sd)**

**barplot2(mc, xlim=range(0, k\*1.8), ylim= range(0, obs2[, 1], max(obs2[, 1])\*1.2), space=0.6, names.arg=levels(obs2[, 2]), plot.ci=TRUE, ci.l=(mc-sc), ci.u= (mc+sc), xpd=FALSE, xlab= names(obs2)[2], ylab=paste("Moyenne +/- écart type de", names(obs2)[1]), cex.lab=1.5, cex.axis=1.5,cex.names=1.5, ci.lwd=2)**

**oneway.test(obs1[,1] ~ obs1[,2], var.equal=FALSE)**

**nc<-sapply(split(obs2[, 1], obs2[, 2]), length)**

**mc<-sapply(split(obs2[, 1], obs2[, 2]), mean)**

**sc<-sapply(split(obs2[, 1], obs2[, 2]), sd)**

**param <- data.frame(nc, mc, sc)**

**names(param) <- c("Effectif", "Moyenne", "Ecart.type")**

**row.names(param) <- levels(obs2[, 2])**

**param**

**k<-nlevels(obs2[,2])**

**s2.c<-sc^2**

**nc.corr<- nc/s2.c**

**m.corr<-sum(nc.corr\*mc)/ sum(nc.corr)**

**F.corr<-(sum(nc.corr\*(mc-m.corr)^2)/(k-1))/(1+(2\*(k-2)/(k^2-1))\*sum(1/(nc-1)\*(1-nc.corr/sum(nc.corr))^2))**

**F.corr**

**nu2.corr<-(k^2-1)/(3\*sum((1/(nc-1))\*(1-nc.corr/sum(nc.corr))^2))**

**nu2.corr**

**pf(F.corr, (k-1), nu2.corr, lower.tail=FALSE)**

### 5.14.3. Tests de comparaisons deux à deux

**cat("Vérifier que le tableau obs1 correspond bien à celui du test dont on souhaite les comparaisons multiples\n")**

**pairwise.t.test(obs1[,1], obs1[,2],p.adjust.method="none", pool.sd=FALSE)**

**k<-length(obs3)**

**moyenne.1<-NULL; moyenne.2<-NULL; p.value<-NULL**

**for(col1 in 1:(k-1)){for(col2 in (col1+1):k){ moyenne.1<-c(moyenne.1, names(obs3)[col1]) ; moyenne.2<-c(moyenne.2, names(obs3)[col2]) ; p.value<-c(p.value,round(t.test(obs3 [,col1], obs3 [,col2], var.equal=FALSE)$p.value, 9))}}**

**p.mult<-data.frame(moyenne.1, moyenne.2, p.value)**

**p.mult**

**k<-nlevels(obs2[,2])**

**mc<-sapply(split(obs2[, 1], obs2[, 2]), mean)**

**nc<-sapply(split(obs2[, 1], obs2[, 2]), length)**

**s2.c<-sapply(split(obs2[, 1], obs2[, 2]), var)**

**pooled.s2<-sum((nc-1)\*s2.c)/(sum(nc)-k)**

**pooled.nu<-sum(nc)-k**

**moyenne.1<-NULL; moyenne.2<-NULL; p.value<-NULL**

**for(col1 in 1:(k-1)){for(col2 in (col1+1):k){ moyenne.1<-c(moyenne.1, names(obs3)[col1]) ; moyenne.2<-c(moyenne.2, names(obs3)[col2]) ; t.calc<-(mc[col1]-mc[col2])/(pooled.s2\*(1/nc[col1]+1/nc[col2]))^0.5 ; p.value<-c(p.value, round(2\*min(pt(t.calc, pooled.nu, lower.tail=FALSE), pt(t.calc, pooled.nu)),9))}}**

**p.mult<-data.frame(moyenne.1, moyenne.2, p.value)**

**p.mult**

**pairwise.t.test(obs1[,1], obs1[,2],p.adjust.method="holm", pool.sd=FALSE)**

## 5.15. Comparaison d'une médiane observée à une valeur théorique (test des signes de Wilcoxon)

### 5.15.1. Méthode

### 5.15.2. Exemples avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-data.frame(mais[which(mais$Parcelle=="Sud"), "Masse.grains"])**

**names(obs1)<-c("Masse.grains")**

**me.theo<-80**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**xi<-obs2[, 1]**

**di<-xi-me.theo**

**signe.di<-factor(sign(di), levels=-1:1, labels=c("negatif", "nul", "positif"))**

**obs3<-data.frame(xi, di, signe.di)**

**di0<-which(obs3$di==0)**

**if(any(di==0)){cat("Certaines différences sont nulles (leur numéro de ligne dans obs3 figure dessous) :\n") ; cat(di0, "\n")} else {cat("Pas de différence nulle\n")}**

**if(any(di==0)){obs4<-obs3[-(which(obs3$di==0)), ]} else{obs4<-obs3}**

**dr<-sort(abs(obs4[, 2]))**

**if(any(table(dr)>1)){cat("Certaines différences dr sont identiques (le nombre de répétitions figure dessous) :\n") ; print(table(as.double(round(dr, 9)))[which(table(dr)>1)])} else {cat("Pas de différences dr identiques\n")}**

**cr<-factor(obs4$signe.di[order(abs(obs4[, 2]))])**

**r<-1:length(cr)**

**obs5<-data.frame(r, dr, cr)**

**names(obs5)<- c("Rang", "dr", "Signe de dr")**

**obs5**

**graphics.off()**

**zone<-matrix(1:3, ncol=1) ; layout(zone, heights=c(2,1,1)) ; par(cex=5, lwd=10) ; layout.show(max(zone))**

**windows()**

**layout(zone, heights=c(2,1,1))**

**par(mar=c(5.1, 6, 4.1, 2.1), mgp=c(3.5,1,0))**

**hist<-hist(obs2[, 1], freq=FALSE, main=NULL, xlab= names(obs1), ylab="Densité de la distribution", cex.lab=1.8, cex.axis=1.8)**

**par(cex.lab=1.8, cex.axis=1.8, bty="n")**

**stripchart(obs2[,1], method="jitter", jitter=1, vertical=FALSE, xlab=names(obs1)[1], pch=16, cex=1.2, xlim=range(hist$breaks))**

**abline(v=median(obs2[,1]), lwd=3)**

**abline(v=mean(obs2[,1]), lwd=3, col=gray(0.5))**

**par(bty="n")**

**boxplot(obs2[,1], horizontal=TRUE, xlab=names(obs1)[1], pch=16, cex=1.2, cex.lab=1.8, cex.axis=1.8, bty="l", ylim=range(hist$breaks))**

**abline(v=me.theo, lty="33", lwd=3)**

**windows()**

**zone<-matrix(1:3, ncol=1) ; layout(zone, heights=c(2, 1, 2)) ; par(cex=5, lwd=10) ; layout.show(max(zone))**

**windows()**

**layout(zone, heights= c(2, 1, 2))**

**par(mar=c(5.1, 6, 4.1, 2.1), mgp=c(3.5,1,0))**

**hist<-hist(di, freq=FALSE, main=NULL, xlab="Différences", ylab="Densité", cex.lab=1.8, cex.axis=1.8)**

**par(cex.lab=1.8, cex.axis=1.8, bty="l", xpd=NA)**

**stripchart(di, method="stack", pch=16, cex=1.2, xlim=range(hist$breaks), ylim=c(0,3), xlab="Différences")**

**me.d<-median(di)**

**segments(me.d, 0, me.d, 5, lwd=4)**

**md<-mean(di)**

**segments(md, 0, md, 5, lwd=4, col=gray(0.5))**

**par(cex.lab=1.8, cex.axis=1.8, bty="l")**

**stripchart(obs5[,1]~obs5[,3], method="stack", vertical=FALSE, ylim=c(0.5,2.5), group.names=levels(obs5[,3]), xlab="Rang des différences dr", pch=16, cex=1.2)**

**wilcox.test(obs1[,1],mu=me.theo)**

**n<-length(obs2[, 1])**

**n**

**summary(obs2)**

**Vpos.calc<-sum(subset(obs5[,1], obs5[,3]=="positif"))**

**Vpos.calc**

**n.pos<-length(which(obs5[,3]=="positif"))**

**n.neg<- length(which(obs5[,3]=="negatif"))**

**n<- n.pos+n.neg**

**min(psignrank((Vpos.calc - 1), n, lower.tail=FALSE), psignrank(Vpos.calc,n))\*2**

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-data.frame(mais[which(mais$Parcelle=="Ouest"), "Masse.grains"])**

**me.theo<-110**

**obs5**

**wilcox.test(obs1[,1],mu=me.theo, correct=FALSE)**

**n<-length(obs2[, 1])**

**n**

**summary(obs2)**

**n.pos<-length(which(obs5[,3]=="positif"))**

**n.neg<- length(which(obs5[,3]=="negatif"))**

**Vpos.calc<-sum(subset(obs5[,1], obs5[,3]=="positif"))**

**Vpos.calc**

**n.ss0<-n.pos + n.neg**

**mV<-n.ss0\*(n.ss0+1)/4**

**ti<-table(obs5[, 2])**

**sV<-(n.ss0\*(n.ss0+1)\*(2\*n.ss0+1)/24 - sum(ti^3-ti)/48)^0.5**

**Zvpos.calc<-(Vpos.calc - mV) / sV**

**Zvpos.calc**

**min(pnorm(Zvpos.calc, lower.tail=FALSE), pnorm(Zvpos.calc))\*2**

**wilcox.test(obs2[,1], mu=me.theo, correct=TRUE)**

**Zvpos.corr<-(abs(Vpos.calc - mV)-0.5) / sV**

**Zvpos.corr**

**min(pnorm(Zvpos.corr, lower.tail=FALSE), pnorm(Zvpos.corr))\*2**

## 5.16. Comparaison de deux médianes observées (test de Mann-Whitney-Wilcoxon)

### 5.16.1. Méthode

### 5.16.2. Exemples avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[which(mais$Parcelle=="Nord" | mais$Parcelle=="Sud"), c(5,11)]**

**obs1[, 2]<-factor(obs1[, 2])**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**if(any(table(obs2[, 1])>1)){cat("Certaines valeurs de", names(obs2)[1], "sont identiques (le nombre de répétitions figure dessous) :\n") ; print(table(obs2[, 1])[which(table(obs2[, 1])>1)])} else {cat("Pas de valeurs identiques dans", names(obs2)[1], "\n")}**

**nc.max<-max(table(obs2[, 2]))**

**nb.na<-nc.max- table(obs2[, 2])**

**tempo<-split(obs2[, 1], obs2[, 2])**

**for(i in 1:length(tempo)){tempo[[i]]<-append(tempo[[i]],rep(NA,nb.na[i]))}**

**obs3<-data.frame(tempo)**

**xr<-sort(obs2[, 1])**

**ci<-obs2[, 2]**

**cr<-ci[order(obs2[, 1])]**

**r<-rank(xr,ties.method="average")**

**obs4<-data.frame(r,xr,cr)**

**names(obs4)<- c("Rang", paste(names(obs2)[1], "en rang croissant"), names(obs2)[2])**

**obs4**

**graphics.off()**

**zone<-matrix(c(1, 2, 0, 3), ncol=2) ; layout(zone) ; par(cex=5, lwd=10) ; layout.show(max(zone))**

**windows()**

**layout(zone)**

**par(mar=c(5.1, 6, 3, 2.1), mgp=c(3.5,1,0))**

**hist1<-hist(split(obs2[, 1],obs2[, 2])[[1]], freq=FALSE, main=NULL, xlab= levels(obs2[, 2])[1], ylab="Densité de la distribution", cex.lab=1.5, cex.axis=1.5)**

**hist2<-hist(split(obs2[, 1],obs2[, 2])[[2]], freq=FALSE, main=NULL, xlab= levels(obs2[, 2])[2], ylab="Densité de la distribution", cex.lab=1.5, cex.axis=1.5)**

**plot(hist1$mids, hist1$density, type="l", lwd=2, xlim=range(hist1$breaks, hist2$breaks), ylim=range(hist1$density, hist2$density), xlab= names(obs2)[1], ylab="Densité de la distribution", cex.lab=1.5, cex.axis=1.5, bty="l")**

**lines(hist2$mids, hist2$density, type="l", lwd=2, col=gray(0.7))**

**par(xpd=NA)**

**legend("topright", inset=c(0, -0.5), legend=levels(obs2[, 2]), lty=c(1,1), col=c("black", gray(0.7)), lwd=2, cex=1.5, bty="n", y.intersp=1.5)**

**windows()**

**zone<-matrix(1:4, ncol=2) ; layout(zone) ; par(cex=5, lwd=10) ; layout.show(max(zone))**

**windows()**

**layout(zone)**

**par(mar=c(5.1, 6, 3, 2.1), mgp=c(3.5,1,0))**

**par(cex.lab=1.5, cex.axis=1.5, bty="l")**

**stripchart(obs2[, 1]~obs2[, 2], method="jitter", jitter=0.1, vertical=FALSE, ylim=c(0.5,2.5), group.names=levels(obs2[, 2]), xlab=names(obs2)[1], ylab=names(obs2)[2], pch=16, cex=1.2)**

**m<-sapply(split(obs2[, 1], obs2[, 2]), mean)**

**segments(m[1], 0.6, m[1], 1.4, lwd=3, col=gray(0.5))**

**segments(m[2], 1.6, m[2], 2.4, lwd=3, col=gray(0.5))**

**par(cex.lab=1.5, cex.axis=1.5, bty="l")**

**boxplot(obs2[, 1] ~ obs2[, 2], horizontal=TRUE, xlab=names(obs2)[1], ylab=names(obs2)[2], pch=16, cex=1.2, boxwex = 0.5)**

**m<-sapply(split(obs2[, 1], obs2[, 2]), mean)**

**segments(m[1], 0.6, m[1], 1.4, lwd=3, col=gray(0.5))**

**segments(m[2], 1.6, m[2], 2.4, lwd=3, col=gray(0.5))**

**par(cex.lab=1.5, cex.axis=1.5, bty="l")**

**stripchart(obs4[, 1]~obs4[, 3], method="stack", vertical=FALSE, ylim=c(0.5, 2.5), group.names=levels(obs4[, 3]), xlab=paste("Rang de", names(obs2)[1]), ylab=names(obs2)[2], pch=16, cex=1.2)**

**library(gplots)**

**m<-sapply(split(obs2[, 1], obs2[, 2]), mean)**

**s<- sapply(split(obs2[, 1], obs2[, 2]), sd)**

**barplot2(m, xlim=c(0, 4), ylim= range(0, obs2[, 1], max(obs2[, 1])\*1.2), space=0.6, names.arg=levels(obs2[, 2]), plot.ci=TRUE, ci.l=(m-s), ci.u= (m+s), xpd=FALSE, xlab=names(obs2)[2], ylab=paste("Moyenne +/- écart type \n de ", names(obs2)[1]), cex.lab=1.5, cex.axis=1.5, cex.names=1.5, ci.lwd=2)**

**cat("La VT U1 est calculée avec les données de", names(obs3)[1], "\n")**

**wilcox.test(obs3[, 1], obs3[, 2])**

**n1<-sapply(split(obs2[, 1], obs2[, 2]), length)[1]**

**n2<-sapply(split(obs2[, 1], obs2[, 2]), length)[2]**

**m1<- sapply(split(obs2[, 1], obs2[, 2]), mean)[1]**

**m2<- sapply(split(obs2[, 1], obs2[, 2]), mean)[2]**

**s1<-sapply(split(obs2[, 1], obs2[, 2]), sd)[1]**

**s2<-sapply(split(obs2[, 1], obs2[, 2]), sd)[2]**

**me1<- sapply(split(obs2[, 1], obs2[, 2]), median)[1]**

**me2<- sapply(split(obs2[, 1], obs2[, 2]), median)[2]**

**min1<- sapply(split(obs2[, 1], obs2[, 2]), min)[1]**

**min2<- sapply(split(obs2[, 1], obs2[, 2]), min)[2]**

**max1<- sapply(split(obs2[, 1], obs2[, 2]), max)[1]**

**max2<- sapply(split(obs2[, 1], obs2[, 2]), max)[2]**

**param <- data.frame(c(n1, n2), c(m1, m2), c(s1, s2), c(me1, me2), c(min1, min2), c(max1, max2))**

**names(param) <- c("Effectif", "Moyenne", "Ecart.type", "Mediane", "Minimum", "Maximum")**

**row.names(param) <- levels(obs2[, 2])**

**param**

**W1calc<-sum(subset(obs4[, 1], obs4[, 3]== names(obs3)[1]))**

**U1calc<-W1calc-0.5\*n1\*(1+n1)**

**U1calc**

**min(pwilcox((U1calc-1), n1, n2, lower.tail=FALSE), pwilcox(U1calc, n1, n2))\*2**

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[which(mais$Parcelle=="Est" | mais$Parcelle=="Ouest"), c(5,11)]**

**obs4**

**cat("La VT U1 est calculée avec les données de", names(obs3)[1], "\n")**

**wilcox.test(obs3[, 1], obs3[, 2], correct=FALSE)**

**param**

**W1calc<-sum(subset(obs4[, 1], obs4[, 3]== names(obs3)[1]))**

**U1calc<-W1calc-0.5\*n1\*(1+n1)**

**U1calc**

**mU<-n1\*n2/2**

**ti<-table(obs4[, 2])**

**sU<-((n1\*n2/12) \* ((n1+n2+1) - sum(ti^3-ti) / ((n1 + n2) \* (n1 + n2 - 1))))^0.5**

**ZU1.calc<-(U1calc-mU)/sU**

**ZU1.calc**

**min(pnorm(ZU1.calc, lower.tail=FALSE), pnorm(ZU1.calc))\*2**

**wilcox.test(obs3[, 1], obs3[, 2],correct=TRUE)**

**ZU1.corr<-(abs(U1calc-mU)-0.5)/sU**

**ZU1.corr**

**min(pnorm(ZU1.corr,lower.tail=FALSE), pnorm(ZU1.corr))\*2**

## 5.17. Comparaison de deux médianes observées en séries appariées (test des signes de Wilcoxon)

### 5.17.1. Méthode

### 5.17.2. Exemples avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[which(mais$Parcelle=="Sud"), c(2,12)]**

**# obs1<-as.data.frame(cbind(c(100, 90, 80, 70, 60, 51, 41, 31, 21, 11), c(90, 81, 72, 63, 54, 52, 43, 34, 25, 16)))**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**di<-obs2[, 1]- obs2[, 2]**

**signe.di<-factor(sign(di), levels=-1:1, labels=c("negatif", "nul", "positif"))**

**obs3<-data.frame(obs2[, 1], obs2[, 2], di, signe.di)**

**names(obs3)[1:2]<-names(obs2)**

**di0<-which(obs3$di==0)**

**if(any(di==0)){cat("Certaines différences sont nulles (leur numéro de ligne dans obs3 figure dessous) :\n") ; cat(di0, "\n")} else {cat("Pas de différence nulle\n")}**

**if(any(di==0)){obs4<-obs3[-(which(obs3$di==0)),]} else{obs4<-obs3}**

**dr<-sort(abs(obs4$di))**

**if(any(table(dr)>1)){cat("Certaines différences dr sont identiques (le nombre de répétitions figure dessous) :\n") ; print(table(as.double(round(dr, 9)))[which(table(dr)>1)])} else {cat("Pas de différences dr identiques\n")}**

**cr<-factor(obs4$signe.di[order(abs(obs4$di))])**

**r<-rank(dr, ties.method = "average")**

**obs5<-data.frame(r,dr,cr)**

**names(obs5)<- c("Rang", "dr", "Signe de dr")**

**obs5**

**obs6<-data.frame(c(obs2[,1], obs2[,2]), c(rep(names(obs2)[1], length(obs2[, 1])), rep(names(obs2)[2], length(obs2[, 2]))))**

**names(obs6)<-c("var.quanti","var.quali")**

**wilcox.test(obs1[,1], obs1[,2], paired=TRUE)**

**n <- length(obs2[, 1])**

**m1 <- mean(obs2[, 1])**

**m2 <- mean(obs2[, 2])**

**me1 <- median(obs2[, 1])**

**me2 <- median(obs2[, 2])**

**param <- data.frame(c(n, n), c(me1, me2), c(m1, m2))**

**names(param) <- c("Effectif", "Mediane", "Moyenne")**

**row.names(param) <- levels(obs6[, 2])**

**param**

**Vpos.calc<-sum(subset(obs5[,1], obs5[,3]=="positif"))**

**Vpos.calc**

**n.pos<-length(which(obs5[,3]=="positif"))**

**n.neg<- length(which(obs5[,3]=="negatif"))**

**n<- n.pos+n.neg**

**min(psignrank((Vpos.calc - 1), n, lower.tail=FALSE), psignrank(Vpos.calc,n))\*2**

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[which(mais$Parcelle=="Ouest"), c(2,12)]**

**obs5**

**wilcox.test(obs1[,1], obs1[,2], paired=TRUE, correct=FALSE)**

**param**

**Vpos.calc<-sum(subset(obs5[,1], obs5[,3]=="positif"))**

**Vpos.calc**

**n.pos<-length(which(obs5[, 3]=="positif"))**

**n.neg<- length(which(obs5[, 3]=="negatif"))**

**n.ss0<-n.pos + n.neg**

**mV<-n.ss0\*(n.ss0+1)/4**

**ti<-table(obs5[, 2])**

**sV<-(n.ss0\*(n.ss0+1)\*(2\*n.ss0+1)/24 - sum(ti^3-ti)/48)^0.5**

**Zvpos.calc<-(Vpos.calc - mV) / sV**

**Zvpos.calc**

**min(pnorm(Zvpos.calc, lower.tail=FALSE), pnorm(Zvpos.calc))\*2**

**wilcox.test(obs1[,1], obs1[,2], paired=TRUE, correct=TRUE)**

**Zvpos.corr<-(abs(Vpos.calc - mV)-0.5) / sV**

**Zvpos.corr**

**min(pnorm(Zvpos.corr,lower.tail=FALSE), pnorm(Zvpos.corr))\*2**

## 5.18. Comparaison d'au moins deux médianes observées

### 5.18.1. Test de Kruskal-Wallis

#### 5.18.1.1. Méthode

#### 5.18.1.2. Exemple avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[, c(5,11)]**

**obs1[,2]<-factor(obs1[,2])**

**# obs1[, 2] <- ordered(obs1[, 2], levels=c("Nord", "Sud", "Est", "Ouest"), labels=c("A", "B", "C", "D"))**

**# obs1 <- cbind(c(100, 200, 300, 400, 500), as.data.frame(c("A", "A", "B", "B", "C")))**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**if(any(table(obs2[, 1])>1)){cat("Certaines valeurs de", names(obs2)[1], "sont identiques (le nombre de répétitions figure dessous) :\n") ; print(table(obs2[, 1])[which(table(obs2[, 1])>1)])} else {cat("Pas de valeurs identiques dans", names(obs2)[1], "\n")}**

**ri<-rank(obs2[,1] ,ties.method="average")**

**obs3<-data.frame(obs2,ri)**

**names(obs3)[3]<- "Rang"**

**obs3**

**nc.max<-max(table(obs2[,2]))**

**nb.na<-nc.max- table(obs2[,2])**

**tempo<-split(obs2[,1], obs2[,2])**

**for(i in 1:length(tempo)){tempo[[i]]<-append(tempo[[i]],rep(NA,nb.na[i]))}**

**obs4<-data.frame(tempo)**

**graphics.off()**

**k<-nlevels(obs2[, 2])**

**row=1 ; col=1 ; while(k > (row \* col)){if(row < col){row<-row+1} else{col<-col+1}} ; par(mfrow=c(row, col)) ; xlim.fen6<-NULL ; ylim.fen6<-NULL ; for(i in 1:k){nom.hist<-paste("hist", i, sep="") ; assign(nom.hist, hist(split(obs2[, 1],obs2[, 2])[[i]], freq=FALSE, main=NULL, xlab= levels(obs2[, 2])[i], ylab="Densité", cex.lab=1.5, cex.axis=1.5)) ; xlim.fen6 <- c(xlim.fen6, get(paste("hist", i, sep=""))$breaks) ; ylim.fen6 <- c(ylim.fen6, get(paste("hist", i, sep=""))$density)}**

**windows()**

**layout(matrix(c(0, 1), ncol=1))**

**plot(hist1$mids, hist1$density, type="l", lwd=2, xlim=range(xlim.fen6), ylim=range(ylim.fen6), xlab= names(obs2)[1], ylab="Densité", cex.lab=1.5, cex.axis=1.5, bty="l")**

**for(i in 2:k){lines(get(paste("hist", i, sep=""))$mids, get(paste("hist", i, sep=""))$density, lty=i, lwd=2)}**

**par(xpd=NA)**

**legend("topright", inset=c(0, -1), legend=levels(obs2[, 2]), lty=c(1:k), lwd=2, cex=1.5, bty="n", y.intersp=1.5)**

**windows()**

**par(mar=c(5.1, 6, 3, 2.1), cex.lab=1.5, cex.axis=1.5, bty="l")**

**stripchart(obs3[, 3]~obs3[, 2], method="stack", vertical=FALSE, ylim=range(0.5, k+0.5), group.names=levels(obs2[, 2]), xlab=paste("Rang de", names(obs2)[1]), ylab=names(obs2)[2], pch=16, cex=1.2)**

**kruskal.test(obs1[,1], obs1[,2])**

**nc<-sapply(split(obs2[, 1], obs2[, 2]), length)**

**me.c<-sapply(split(obs2[,1], obs2[,2]), median)**

**min.c<- sapply(split(obs2[,1], obs2[,2]), min)**

**max.c<- sapply(split(obs2[,1], obs2[,2]), max)**

**mc<-sapply(split(obs2[, 1], obs2[, 2]), mean)**

**sc<-sapply(split(obs2[, 1], obs2[, 2]), sd)**

**param <- data.frame(nc, me.c, min.c, max.c, mc, sc)**

**names(param) <- c("Effectif", "Mediane", "Minimum", "Maximum", "Moyenne", "Ecart.type")**

**row.names(param) <- levels(obs2[, 2])**

**param**

**n<-length(obs3$Rang)**

**Rc<-tapply(obs3$Rang, obs3[, 2], "sum")**

**K.calc<-12\*sum(Rc^2/nc)/(n\*(n+1))-3\*(n+1)**

**K.calc**

**ti<-table(obs3[,1])**

**Kcorr<-K.calc/(1-sum(ti^3-ti)/(n^3-n))**

**Kcorr**

**k<-length(nc)**

**pchisq(Kcorr, k-1, lower.tail=FALSE)**

**nc<-c(3, 2, 2)**

**k<-length(nc)**

**n<-sum(nc)**

**vecteur.ini<-NULL**

**for(i in 1:k){vecteur.ini<-c(vecteur.ini, rep(LETTERS[i], nc[i]))}**

**rang<-1:n**

**library(gtools)**

**perm<-as.data.frame(t(unique(permutations(n, n, vecteur.ini, set=FALSE))))**

**# perm**

**serie.K<-NULL**

**for(i in 1:length(perm)){Rc<-tapply(rang, perm[, i], "sum") ; serie.K<-c(serie.K, round(12\*sum(Rc^2/nc)/(n\*(n+1))-3\*(n+1), 9))}**

**K<-sort(unique(serie.K))**

**prob.K<-table(serie.K)/length(perm)**

**round(cumsum(rev(table(serie.K)/length(perm))),3)**

**plot(K, as.vector(prob.K), type="l", xlab="Valeur de K", ylab="Probabilité f(K)", cex.lab=1.5, cex.axis=1.5)**

**p.value<-sum(prob.K[K >= round(K.calc, 9)])**

**p.value**

#### 5.18.1.3. Tests de comparaisons deux à deux

**cat("Vérifier que le tableau obs1 correspond bien à celui du test dont on souhaite les comparaisons multiples\n")**

**suppressWarnings(pairwise.wilcox.test(obs1[, 1], obs1[, 2], p.adjust.method="none"))**

**cat("Vérifier que le tableau obs4 correspond bien à celui du test dont on souhaite les comparaisons multiples\n")**

**k<-length(obs4)**

**classe.1<-NULL; classe.2<-NULL; p.value<-NULL**

**for(col1 in 1:(k-1)){for(col2 in (col1+1):k){ classe.1<-c(classe.1, names(obs4)[col1]) ; classe.2<-c(classe.2, names(obs4)[col2]) ; p.value<-c(p.value, round(suppressWarnings(wilcox.test(obs4 [, col1], obs4 [, col2])$p.value), 9))}}**

**p.mult<-data.frame(classe.1, classe.2, p.value)**

**p.mult**

**suppressWarnings(pairwise.wilcox.test(obs1[, 1], obs1[, 2], p.adjust.method="holm"))**

### 5.18.2. Test des médianes

#### 5.18.2.1. Méthode

#### 5.18.2.2. Exemple avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[, c(5,11)]**

**obs1[,2]<-factor(obs1[,2])**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**me<-median(obs2[,1])**

**me**

**cat(paste("Nombre de valeurs de", names(obs2)[1], "identiques à me :"), length(which(obs2[,1] == me)), "\n")**

**obs3<-transform(obs2, Sup.me= (obs2[,1] > me))**

**obs4<-table(obs3[,2], obs3[,3], dnn=c(names(obs3)[2], "Sup.me"))**

**addmargins(obs4)**

**windows()**

**par(cex.lab=2, xpd=NA, font=2)**

**mosaicplot(t(obs4), main=NULL, cex.axis=1.5)**

**nc<-sapply(split(obs2[, 1], obs2[, 2]), length)**

**me.c<-sapply(split(obs2[,1], obs2[,2]), median)**

**min.c<- sapply(split(obs2[,1], obs2[,2]), min)**

**max.c<- sapply(split(obs2[,1], obs2[,2]), max)**

**mc<-sapply(split(obs2[, 1], obs2[, 2]), mean)**

**param <- data.frame(nc, me.c, min.c, max.c, mc)**

**names(param) <- c("Effectif", "Mediane", "Minimum", "Maximum", "Moyenne")**

**row.names(param) <- levels(obs2[, 2])**

**param**

**chisq.test(obs4)**

**obs4theo<-suppressWarnings(chisq.test(obs4)$expected)**

**obs4theo<-suppressWarnings(chisq.test(obs4)$expected)**

**chi2.calc<-sum((obs4-obs4theo)^2/obs4theo)**

**chi2.calc**

**k<-nlevels(obs2[,2])**

**pchisq(chi2.calc, (k-1), lower.tail=FALSE)**

**n.1<-as.double(margin.table(obs4,2)[1])**

**n.2<-as.double(margin.table(obs4,2)[2])**

**ni<-margin.table(obs4,1)**

**n<-margin.table(obs4)**

**ni1<-obs4[,1]**

**ni2<-obs4[,2]**

**ti1<-ni\*n.1/n**

**n^2/(n.1\*n.2)\*sum((ni1-ti1)^2/ni)**

**4\*sum((ni1-ti1)^2/ni)**

**sum((ni1-ni2)^2/ni)**

#### 5.18.2.3. Tests de comparaisons deux à deux

**cat("Vérifier que les tableaux obs2 et obs4 correspondent bien à ceux du test dont on souhaite les comparaisons multiples\n")**

**k<-nlevels(obs2[,2])**

**classe.1<-NULL; classe.2<-NULL; me<-NULL ; test<-NULL ; p.value<-NULL**

**for(col1 in 1:(k-1)){for(col2 in (col1+1):k){ classe.1<-c(classe.1, levels(obs2[,2])[col1]) ; classe.2<-c(classe.2, levels(obs2[,2])[col2]) ; obs <- obs2[which(obs2[,2] == levels(obs2[,2])[col1] | obs2[,2] == levels(obs2[,2])[col2]), ] ; obs[,2]<-factor(obs[,2]) ; me.loop<-median(obs[,1]) ; me <- c(me, me.loop) ; obs3<-transform(obs, Sup.me= (obs[,1] > me.loop)) ; obs4<-table(obs3[,2], obs3[,3], dnn=c(names(obs3)[2], "Sup.me")) ; suppressWarnings(obs4theo<- chisq.test(obs4)$expected) ; n.cases.inf.5<-length(which(obs4theo<5)) ; if(n.cases.inf.5 == 0){test<-c(test, "Chi2") ; p.value<-c(p.value, round(chisq.test(obs4, correct=TRUE)$p.value, 9))} else { test<-c(test, "Fisher.exact") ; p.value<- c(p.value, round(fisher.test(obs4)$p.value, 9))}}}**

**p.mult<-data.frame(classe.1, classe.2, me, test, p.value)**

**p.mult**

**k<-nlevels(obs2[,2])**

**classe.1<-NULL; classe.2<-NULL; me<-NULL ; test<-NULL ; p.value<-NULL**

**for(col1 in 1:(k-1)){for(col2 in (col1+1):k){ classe.1<-c(classe.1, levels(obs2[,2])[col1]) ; classe.2<-c(classe.2, levels(obs2[,2])[col2]) ; obs <- obs2[which(obs2[,2] == levels(obs2[,2])[col1] | obs2[,2] == levels(obs2[,2])[col2]), ] ; obs[,2]<-factor(obs[,2]) ; me.loop<-median(obs[,1]) ; me <- c(me, me.loop) ; obs3<-transform(obs, Sup.me= (obs[,1] > me.loop)) ; obs4<-table(obs3[,2], obs3[,3], dnn=c(names(obs3)[2], "Sup.me")) ; test<-c(test, "Fisher.exact") ; p.value<- c(p.value, round(fisher.test(obs4)$p.value, 9))}}**

**p.mult<-data.frame(classe.1, classe.2, me, test, p.value)**

**p.mult**

**p.mult2<-transform(p.mult, Holm=round(p.adjust(p.mult$p.value,method="holm"),9))**

**p.mult2**

### 5.19.1. Test de Fisher-Snedecor

#### 5.19.1.1. Méthode

#### 5.19.1.2. Exemple avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[which(mais$Parcelle=="Nord" | mais$Parcelle=="Sud"), c(2,11)]**

**obs1[,2]<-factor(obs1[,2])**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**nc.max<-max(table(obs2[,2]))**

**nb.na<-nc.max- table(obs2[,2])**

**tempo<-split(obs2[,1], obs2[,2])**

**for(i in 1:length(tempo)){tempo[[i]]<-append(tempo[[i]],rep(NA,nb.na[i]))}**

**obs3<-data.frame(tempo)**

**obs3**

**mc<-sapply(split(obs2[,1], obs2[,2]),mean)**

**obs4<-obs2**

**obs4[which(obs2[, 2] == levels(obs2[, 2])[1]), 1] <- obs4[which(obs2[, 2] == levels(obs2[, 2])[1]), 1] - mc[1]**

**obs4[which(obs2[, 2] == levels(obs2[, 2])[2]), 1] <- obs4[which(obs2[, 2] == levels(obs2[, 2])[2]), 1] - mc[2]**

**graphics.off()**

**zone<-matrix(1:4, ncol=2) ; layout(zone) ; par(cex=5, lwd=10) ; layout.show(max(zone))**

**windows()**

**layout(zone)**

**par(mar=c(5.1, 6, 3, 2.1))**

**hist(split(obs2[, 1],obs2[, 2])[[1]], freq=FALSE, main=NULL, xlab= levels(obs2[, 2])[1], ylab="Densité de la distribution", cex.lab=1.5, cex.axis=1.5)**

**hist(split(obs2[, 1],obs2[, 2])[[2]], freq=FALSE, main=NULL, xlab= levels(obs2[, 2])[2], ylab="Densité de la distribution", cex.lab=1.5, cex.axis=1.5)**

**par(cex.lab=1.5, cex.axis=1.5, bty="l", mgp=c(4,1,0))**

**stripchart(obs4[, 1]~obs4[, 2], method="jitter", jitter=0.1, vertical=FALSE, ylim=c(0.5, 2.5), group.names=levels(obs2[, 2]), xlab= paste("Valeurs de", names(obs2)[1], "\n centrées sur la moyenne"), ylab=names(obs2)[2], pch=16, cex=1.2)**

**m0c<-sapply(split(obs4[, 1], obs4[, 2]), mean)**

**s<-sapply(split(obs2[, 1], obs2[, 2]), sd)**

**segments(m0c[1], 0.6, m0c[1], 0.8, lwd=3, col=gray(0.5))**

**segments(-s[1], 0.7, s[1], 0.7, lwd=3) ; segments(-s[1], 0.6, -s[1], 0.8, lwd=3) ; segments(s[1], 0.6, s[1], 0.8, lwd=3)**

**segments(m0c[2], 1.6, m0c[2], 1.8, lwd=3, col=gray(0.5))**

**segments(-s[2], 1.7, s[2], 1.7, lwd=3) ; segments(-s[2], 1.6, -s[2], 1.8, lwd=3) ; segments(s[2], 1.6, s[2], 1.8, lwd=3)**

**par(cex.lab=1.5, cex.axis=1.5, bty="l", mgp=c(3,1,0))**

**boxplot(obs2[, 1] ~ obs2[, 2], horizontal=TRUE, xlab=names(obs2)[1], ylab=names(obs2)[2], pch=16, cex=1.2, boxwex = 0.5)**

**m<-sapply(split(obs2[, 1], obs2[, 2]), mean)**

**segments(m[1], 0.6, m[1], 1.4, lwd=3, col=gray(0.5))**

**segments(m[2], 1.6, m[2], 2.4, lwd=3, col=gray(0.5))**

**windows()**

**par(mar=c(5.1, 6, 3, 2.1), mgp=c(3.5,1,0))**

**library(gplots)**

**barplot2(m, xlim=c(0, 4), ylim= range(0, obs2[, 1], max(obs2[, 1])\*1.2), space=0.6, names.arg=levels(obs2[, 2]), plot.ci=TRUE, ci.l=(m-s), ci.u= (m+s), xpd=FALSE, xlab=names(obs2)[2], ylab=paste("Moyenne +/- écart type de", names(obs2)[1]), cex.lab=1.5, cex.axis=1.5,cex.names=1.5, ci.lwd=2)**

**var.test(obs3[, 1], obs3[, 2])**

**n1<-sapply(split(obs2[, 1], obs2[, 2]), length)[1]**

**n2<-sapply(split(obs2[, 1], obs2[, 2]), length)[2]**

**s2.1<-sapply(split(obs2[, 1], obs2[, 2]), var)[1]**

**s2.2<-sapply(split(obs2[, 1], obs2[, 2]), var)[2]**

**m1<- sapply(split(obs2[, 1], obs2[, 2]), mean)[1]**

**m2<- sapply(split(obs2[, 1], obs2[, 2]), mean)[2]**

**param <- data.frame(c(n1, n2), c(s2.1, s2.2), c(s2.1^0.5, s2.2^0.5), c(m1, m2))**

**names(param) <- c("Effectif", "Variance", "Ecart.type", "Moyenne")**

**row.names(param) <- levels(obs2[, 2])**

**param**

**F.calc<-s2.1/s2.2**

**F.calc**

**nu.1<-n1-1**

**nu.2<-n2-1**

**min(pf(F.calc, nu.1, nu.2, lower.tail=FALSE), pf(F.calc, nu.1, nu.2))\*2**

**F.calc/qf(0.025, nu.1, nu.2, lower.tail=FALSE)**

**F.calc/qf(0.025, nu.1, nu.2)**

### 3.0233985.19.2. Test d'Ansari-Bradley

#### 5.19.2.1. Méthode

#### 5.19.2.2. Exemples avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[which(mais$Parcelle=="Nord" | mais$Parcelle=="Sud"), c(5,11)]**

**obs1[, 2]<-factor(obs1[,2])**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**if(any(table(obs2[, 1])>1)){cat("Certaines valeurs de", names(obs2)[1], "sont identiques (le nombre de répétitions figure dessous) :\n") ; print(table(obs2[, 1])[which(table(obs2[, 1])>1)])} else {cat("Pas de valeurs identiques dans", names(obs2)[1], "\n")}**

**nc.max<-max(table(obs2[,2]))**

**nb.na<-nc.max- table(obs2[,2])**

**tempo<-split(obs2[,1], obs2[,2])**

**for(i in 1:length(tempo)){tempo[[i]]<-append(tempo[[i]],rep(NA,nb.na[i]))}**

**obs3<-data.frame(tempo)**

**xr<-sort(obs2[,1])**

**ci<-obs2[,2]**

**cr<-ci[order(obs2[,1])]**

**n<-length(obs2[,1])**

**if(n%%2==0){rAB<-c(1:(n/2),(n/2):1)} else{rAB<-c(1:((n-1)/2), (n+1)/2, ((n-1)/2):1)}**

**obs4<-data.frame(rAB,xr,cr)**

**ties<-as.numeric(names(which((table(obs2[,1])>1)==TRUE)))**

**if(length(ties) > 0){cat("Les rangs de obs4 sont modifiés car il y a des valeurs xi de la variable quantitative identiques\n") ; for(i in ties){obs4$rAB[(which(obs4$xr == i))] <- mean(obs4$rAB[(which(obs4$xr == i))])}}**

**obs4**

**mc<-sapply(split(obs2[,1], obs2[,2]),mean)**

**obs5<-obs2**

**obs5[which(obs2[, 2] == levels(obs2[, 2])[1]), 1] <- obs5[which(obs2[, 2] == levels(obs2[, 2])[1]), 1] - mc[1]**

**obs5[which(obs2[, 2] == levels(obs2[, 2])[2]), 1] <- obs5[which(obs2[, 2] == levels(obs2[, 2])[2]), 1] - mc[2]**

**windows()**

**zone<-matrix(1:2, ncol=1) ; layout(zone) ; par(cex=5, lwd=10) ; layout.show(max(zone))**

**windows()**

**layout(zone)**

**par(mar=c(5.1, 6, 3, 2.1))**

**par(xaxt="n", cex.lab=1.3, cex.axis=1.3, bty="l", mgp=c(4,1,0))**

**r<-rank(obs4[ , 2], ties.method="average")**

**stripchart(r~obs4[, 3], method="stack", vertical=FALSE, ylim=c(0.5, 2.5), group.names=levels(obs4[, 3]), xlab=paste("Rang d'Ansari-Bradley \n de ", names(obs2)[1]), ylab=names(obs2)[2], pch=16, cex=1.2)**

**me.r<-sapply(split(r, obs4[, 3]), median)**

**segments(me.r[1], 0.6, me.r[1], 1.4, lwd=3)**

**segments(me.r[2], 1.6, me.r[2], 2.4, lwd=3)**

**i=1; pos<-rep(1, 100000); while(length(pos)>10){pos<-(1:(max(r)/i))\*i;i<-i+1}**

**par(xaxt="s") ; axis(side=1, at=pos, labels = obs4[, 1][pos], cex.axis=1.3)**

**library(gplots)**

**par(mar=c(5.1, 6, 3, 2.1), mgp=c(3,1,0))**

**barplot2(m, xlim=c(0, 4), ylim= range(0, obs2[, 1], max(obs2[, 1])\*1.2), space=0.6, names.arg=levels(obs2[, 2]), plot.ci=TRUE, ci.l=(m-s), ci.u= (m+s), xpd=FALSE, xlab=names(obs2)[2], ylab=paste("Moyenne +/- écart type \n de ", names(obs2)[1]), cex.lab=1.3, cex.axis=1.3, cex.names=1.3, ci.lwd=2)**

**cat("La VT AB1 est calculée avec les données de", names(obs3)[1], "\n")**

**ansari.test(obs3[,1], obs3[,2])**

**n1<-sapply(split(obs2[, 1], obs2[, 2]), length)[1]**

**n2<-sapply(split(obs2[, 1], obs2[, 2]), length)[2]**

**s2.1<-sapply(split(obs2[, 1], obs2[, 2]), var)[1]**

**s2.2<-sapply(split(obs2[, 1], obs2[, 2]), var)[2]**

**m1<- sapply(split(obs2[, 1], obs2[, 2]), mean)[1]**

**m2<- sapply(split(obs2[, 1], obs2[, 2]), mean)[2]**

**me1<- sapply(split(obs2[, 1], obs2[, 2]), median)[1]**

**me2<- sapply(split(obs2[, 1], obs2[, 2]), median)[2]**

**param <- data.frame(c(n1, n2), c(s2.1, s2.2), c(s2.1^0.5, s2.2^0.5), c(m1, m2), c(me1, me2))**

**names(param) <- c("Effectif", "Variance", "Ecart.type", "Moyenne", "Médiane")**

**row.names(param) <- levels(obs2[, 2])**

**param**

**n<-length(obs2[,1])**

**n**

**AB1.calc<-sum(subset(obs4[,1],obs4[,3]== names(obs3)[1]))**

**AB1.calc**

**cat("La VT AB1 est calculée avec les données de", names(obs3)[1], "\n")**

**ansari.test(obs3[,1], obs3[,2], exact=FALSE)**

**n<-length(obs2[,1])**

**n**

**AB1.calc<-sum(subset(obs4[,1],obs4[,3]== names(obs3)[1]))**

**AB1.calc**

**n1<-sapply(split(obs2[, 1], obs2[, 2]), length)[1]**

**n2<-sapply(split(obs2[, 1], obs2[, 2]), length)[2]**

**n<-length(obs2[,1])**

**if(n%%2 == 0){mAB1<-n1\*(n+2)/4} else { mAB1<-n1\*(n+1)^2/(4\*n)}**

**if(n%%2 == 0){s2.AB<-(n1\*n2\*(n+2)\*(n-2))/(48\*(n-1))} else{ s2.AB<-(n1\*n2\*(n+1)\*(3+n^2))/(48\*n^2)}**

**ZAB1.calc<-(AB1.calc-mAB1)/ (s2.AB)^0.5**

**ZAB1.calc**

**min(pnorm(ZAB1.calc,lower.tail=FALSE), pnorm(ZAB1.calc))\*2**

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[which(mais$Parcelle=="Est" | mais$Parcelle=="Ouest"), c(5,11)]**

**obs4**

**cat("La VT ZAB1 est calculée avec les données de", names(obs3)[1], "\n")**

**library(coin)**

**ansari\_test(Masse.grains~Parcelle,obs2, ties.method = "average-scores")**

**param**

**n<-length(obs2[,1])**

**n**

**AB1.calc<-sum(subset(obs4[,1],obs4[,3]== names(obs3)[1]))**

**AB1.calc**

**n1<-sapply(split(obs2[, 1], obs2[, 2]), length)[1]**

**n2<-sapply(split(obs2[, 1], obs2[, 2]), length)[2]**

**n<-length(obs2[,1])**

**if(n%%2 == 0){mAB1<-n1\*(n+2)/4} else { mAB1<-n1\*(n+1)^2/(4\*n)}**

**ti<- table(obs4$xr)**

**rABi<- obs4$rAB[c(which(diff(obs4$xr)!=0),n)]**

**if(n%%2 == 0){s2.AB<-(n1\*n2\*(16\*sum(ti\*rABi^2)-n\*(n+2)^2))/(16\*n\*(n-1))} else{ s2.AB<-(n1\*n2\*(16\*n\*sum(ti\*rABi^2)-(n+1)^4))/(16\*n^2\*(n-1))}**

**ZAB1.calc<-(AB1.calc-mAB1)/ (s2.AB)^0.5**

**ZAB1.calc**

**min(pnorm(ZAB1.calc, lower.tail=FALSE), pnorm(ZAB1.calc))\*2**

**xr<-sort(obs2[,1])**

**ci<-obs2[,2]**

**cr<-ci[order(obs2[,1])]**

**r<-rank(xr,ties.method="average")**

**obs4<-data.frame(r,xr,cr)**

**obs4**

**cat("La VT ZAB1 est calculée avec les données de", names(obs3)[1], "\n")**

**ansari.test(obs3[,1], obs3[,2], exact=FALSE)**

**n<-length(obs2[,1])**

**n**

**rj<-split(obs4[,1],obs4[,3])[[1]]**

**AB1.calc<-sum(pmin(rj,n-rj+1))**

**AB1.calc**

**n1<-sapply(split(obs2[, 1], obs2[, 2]), length)[1]**

**n2<-sapply(split(obs2[, 1], obs2[, 2]), length)[2]**

**n<-length(obs2[,1])**

**if(n%%2 == 0){mAB1<-n1\*(n+2)/4} else { mAB1<-n1\*(n+1)^2/(4\*n)}**

**ti<- table(pmin(r,n-r+1))**

**ri<- as.numeric(names(ti))**

**if(n%%2 == 0){s2.AB<-(n1\*n2\*(16\*sum(ti\*ri^2)-n\*(n+2)^2))/(16\*n\*(n-1))} else{ s2.AB<-(n1\*n2\*(16\*n\*sum(ti\*ri^2)-(n+1)^4))/(16\*n^2\*(n-1))}**

**ZAB1.calc<-(AB1.calc-mAB1)/ (s2.AB)^0.5**

**ZAB1.calc**

**min(pnorm(ZAB1.calc, lower.tail=FALSE), pnorm(ZAB1.calc))\*2**

## 5.20. Comparaison d'au moins deux variances observées

### 5.20.1. Test de Bartlett

#### 5.20.1.1. Méthode

#### 5.20.1.2. Exemple avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[, c(3,11)]**

**obs1[,2]<-factor(obs1[,2])**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**nc.max<-max(table(obs2[,2]))**

**nb.na<-nc.max- table(obs2[,2])**

**tempo<-split(obs2[,1], obs2[,2])**

**for(i in 1:length(tempo)){tempo[[i]]<-append(tempo[[i]],rep(NA,nb.na[i]))}**

**obs3<-data.frame(tempo)**

**obs3**

**mc<-sapply(split(obs2[, 1], obs2[, 2]),mean)**

**obs4<-obs2**

**for(i in 1:nlevels(obs2[, 2])){obs4[which(obs2[, 2] == levels(obs2[, 2])[i]), 1] <- obs4[which(obs2[, 2] == levels(obs2[, 2])[i]), 1] - mc[i]}**

**graphics.off()**

**k<-nlevels(obs2[, 2])**

**row=1 ; col=1 ; while(k > (row \* col)){if(row < col){row<-row+1} else{col<-col+1}} ; par(mfrow=c(row, col)) ; for(i in 1:k){hist(split(obs2[, 1],obs2[, 2])[[i]], freq=FALSE, main=NULL, xlab= levels(obs2[, 2])[i], ylab="Densité", cex.lab=1.5, cex.axis=1.5)}**

**windows()**

**par(mar=c(5.1, 6, 3, 2.1), cex.lab=1.5, cex.axis=1.5, bty="l")**

**stripchart(obs4[, 1]~obs4[, 2], method="jitter", jitter=0.1, vertical=FALSE, ylim=range(0.5, k+0.5), group.names=levels(obs2[, 2]), xlab= names(obs2)[1], ylab=names(obs2)[2], pch=16, cex=1.2)**

**mc0<-sapply(split(obs4[, 1], obs4[, 2]), mean)**

**sc<-sapply(split(obs2[, 1], obs2[, 2]), sd)**

**for(i in 1:k){segments(mc0[i], i-0.45, mc0[i], i-0.25, lwd=3, col=gray(0.5))}**

**for(i in 1:k){ segments(-sc[i], i-0.35, sc[i], i-0.35, lwd=3) ; segments(-sc[i], i-0.45, -sc[i], i-0.25, lwd=3) ; segments(sc[i], i-0.45, sc[i], i-0.25, lwd=3)}**

**windows()**

**par(mar=c(5.1, 6, 3, 2.1), cex.lab=1.5, cex.axis=1.5, bty="l")**

**boxplot(obs2[, 1] ~ obs2[, 2], horizontal=TRUE, xlab= names(obs2)[1], ylab=names(obs2)[2], pch=16, cex=1.2, boxwex = 0.5)**

**for(i in 1:k){segments(mc[i], i-0.3, mc[i], i+0.3, lwd=3, col=gray(0.5))}**

**windows()**

**library(gplots)**

**mc<-sapply(split(obs2[, 1], obs2[, 2]), mean)**

**sc<- sapply(split(obs2[, 1], obs2[, 2]), sd)**

**barplot2(mc, xlim=range(0, k\*1.8), ylim= range(0, obs2[, 1], max(obs2[, 1])\*1.2), space=0.6, names.arg=levels(obs2[, 2]), plot.ci=TRUE, ci.l=(mc-sc), ci.u= (mc+sc), xpd=FALSE, xlab= names(obs2)[2], ylab=paste("Moyenne +/- écart type de", names(obs2)[1]), cex.lab=1.5, cex.axis=1.5,cex.names=1.5, ci.lwd=2)**

**bartlett.test(obs3)**

**nc<-sapply(split(obs2[, 1], obs2[, 2]), length)**

**s2c<-sapply(split(obs2[,1], obs2[,2]), var)**

**mc<-sapply(split(obs2[, 1], obs2[, 2]), mean)**

**param <- data.frame(nc, s2c, s2c^0.5, mc)**

**names(param) <- c("Effectif", "Variance", "Ecart.type", "Moyenne")**

**row.names(param) <- levels(obs2[, 2])**

**param**

**k<-nlevels(obs2[, 2])**

**n<-sum(nc)**

**sigma2<-sum((nc-1)\*s2c)/(n-k)**

**Bcalc<-sum((nc - 1)\*log(sigma2/s2c)) / (1 + 1/(3\*(k-1)) \* (sum(1 / (nc - 1)) - 1/sum(nc-1)))**

**Bcalc**

**pchisq(Bcalc, (k-1), lower.tail=FALSE)**

### 5.20.2. Test de Fligner- Killeen

#### 5.20.2.1. Méthode

#### 5.20.2.2. Exemple avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[, c(5,11)]**

**obs1[,2]<-factor(obs1[,2])**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**xi<-obs2[,1]**

**ci<-obs2[,2]**

**Me.c<-tapply(xi,ci,median)**

**Me.c**

**zi<-xi-as.numeric(Me.c[ci])**

**ri<-rank(abs(zi))**

**n<-length(xi)**

**Fi<-0.5\*(1+ri/(n+1))**

**qi<-qnorm(Fi)**

**obs4<-data.frame(xi,ci,zi,ri,Fi,qi)**

**obs4**

**nc.max<-max(table(ci))**

**nb.na<-nc.max- table(ci)**

**tempo<-split(xi,ci)**

**for(i in 1:length(tempo)){tempo[[i]]<-append(tempo[[i]],rep(NA,nb.na[i]))}**

**obs3<-data.frame(tempo)**

**obs3**

**mc<-sapply(split(obs2[, 1], obs2[, 2]),mean)**

**obs5<-obs2**

**for(i in 1:nlevels(obs2[, 2])){obs5[which(obs2[, 2] == levels(obs2[, 2])[i]), 1] <- obs5[which(obs2[, 2] == levels(obs2[, 2])[i]), 1] - mc[i]}**

**windows()**

**par(mar=c(5.1, 6, 3, 2.1), cex.lab=1.5, cex.axis=1.5, bty="l")**

**stripchart(obs4[, 6]~obs4[, 2], method="stack", vertical=FALSE, ylim=range(0.5, k+0.5), group.names=levels(obs2[, 2]), xlab= paste("Quantiles qi provenant de", names(obs2)[1]), ylab=names(obs2)[2], pch=16, cex=1.2)**

**mc.qi<-sapply(split(obs4[, 6], obs4[, 2]), mean)**

**for(i in 1:k){segments(mc.qi[i], i-0.25, mc.qi[i], i+0.25, lwd=3, col=gray(0.5))}**

**fligner.test(obs1[,1], obs1[,2])**

**param**

**xi<-obs2[,1]**

**ci<-obs2[,2]**

**Me.c<-tapply(xi,ci,median)**

**zi<-xi-as.numeric(Me.c[ci])**

**ri<-rank(abs(zi))**

**n<-sum(nc)**

**Fi<-0.5\*(1+ri/(n+1))**

**qi<-qnorm(Fi)**

**Q<-sum(tapply(qi,ci,"sum")^2/nc)**

**mq<-mean(qi)**

**s2q<-var(qi)**

**FK.calc<-(Q-n\*mq^2)/s2q**

**FK.calc**

**k<-length(nc)**

**pchisq(FK.calc,k-1,lower.tail=FALSE)**

### 5.20.3. Tests de comparaisons deux à deux

**cat("Vérifier que le tableau obs3 correspond bien à celui du test dont on souhaite les comparaisons multiples\n")**

**k<-length(obs3)**

**variance.1<-NULL; variance.2<-NULL; test<-NULL; p.value<-NULL**

**for(col1 in 1:(k-1)){for(col2 in (col1+1):k){ variance.1<-c(variance.1, names(obs3)[col1]) ; variance.2<-c(variance.2, names(obs3)[col2]) ; test<-c(test, "Fisher") ; p.value<-c(p.value,round(var.test(obs3 [,col1], obs3 [,col2], na.rm=TRUE)$p.value, 9))}}**

**p.mult<-data.frame(variance.1, variance.2, test, p.value)**

**p.mult**

**p.mult2<-transform(p.mult, Holm=round(p.adjust(p.mult$p.value,method="holm"),6))**

**p.mult2**

5.21. Test du coefficient de corrélation linéaire de Pearson

### 5.21.1. Méthode

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[which(mais$Parcelle=="Est"), c(2,3)]**

**mx<-mean(obs1$Hauteur, na.rm = TRUE)**

**my<-mean(obs1$Masse, na.rm = TRUE)**

**sx2<-var(obs1$Hauteur, na.rm = TRUE)**

**sy2<-var(obs1$Masse, na.rm = TRUE)**

**sx<-sd(obs1$Hauteur, na.rm = TRUE)**

**sy<-sd(obs1$Masse, na.rm = TRUE)**

**sxy<-var(obs1$Hauteur, obs1$Masse, na.rm = TRUE)**

**rxy<-cor(obs1$Hauteur, obs1$Masse, use= "complete.obs")**

**xi<-seq((mx-4\*sx), (mx+4\*sx), length.out=50)**

**yi<-seq((my-4\*sy), (my+4\*sy), length.out=50)**

**rm(pi)**

**f<-function(xi,yi){ 1/(2 \* pi \* (sx2 \* sy2 \* (1- rxy^2))^0.5) \* exp(-1/(2 \* (1-rxy^2))\*((xi-mx)^2 / sx2 + (yi-my)^2/sy2 - 2 \* rxy \* (xi-mx) \* (yi-my)/((sx2)^0.5 \* (sy2)^0.5)))}**

**density<-outer(xi,yi,f)**

**persp(xi,yi,density, zlim = range(0, 0.0001, na.rm = TRUE), phi = 50, ,box=FALSE)**

### 5.21.2. Exemple avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[which(mais$Parcelle=="Est"), c(2,3)]**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**obs2**

**graphics.off()**

**zone<-matrix(1:6, ncol=2) ; layout(zone, heights=c(2,1,1)) ; par(cex=5, lwd=10) ; layout.show(max(zone))**

**windows()**

**layout(zone, heights=c(2,1,1))**

**par(mar=c(5.1, 6, 4.1, 2.1), mgp=c(3.5,1,0))**

**hist<-hist(obs2[, 1], freq=FALSE, main=NULL, xlab= names(obs2)[1], ylab="Densité de la distribution", cex.lab=1.8, cex.axis=1.8)**

**par(cex.lab=1.8, cex.axis=1.8, bty="n")**

**stripchart(obs2[, 1], method="jitter", jitter=1, vertical=FALSE, xlab=names(obs2)[1], pch=16, cex=1.2, xlim=range(hist$breaks))**

**abline(v=mean(obs2[, 1]), lwd=2, col=gray(0.5))**

**par(bty="n")**

**boxplot(obs2[, 1], horizontal=TRUE, xlab=names(obs2)[1], pch=16, cex=1.2, cex.lab=1.8, cex.axis=1.8, bty="l", ylim=range(hist$breaks))**

**abline(v=mean(obs2[, 1]), lwd=2, col=gray(0.5))**

**hist<-hist(obs2[, 2], freq=FALSE, main=NULL, xlab= names(obs2)[2], ylab="Densité de la distribution", cex.lab=1.8, cex.axis=1.8)**

**par(cex.lab=1.8, cex.axis=1.8, bty="n")**

**stripchart(obs2[, 2], method="jitter", jitter=1, vertical=FALSE, xlab=names(obs2)[2], pch=16, cex=1.2, xlim=range(hist$breaks))**

**abline(v=mean(obs2[, 2]), lwd=2, col=gray(0.5))**

**par(bty="n")**

**boxplot(obs2[, 2], horizontal=TRUE, xlab=names(obs2)[2], pch=16, cex=1.2, cex.lab=1.8, cex.axis=1.8, bty="l", ylim=range(hist$breaks))**

**abline(v=mean(obs2[, 2]), lwd=2, col=gray(0.5))**

**windows()**

**plot(obs2, pch=16, cex=1.2, cex.lab=1.5, cex.axis=1.5, bty="l")**

**# x<-obs2[, 1] ; y<-obs2[, 2] ; a=cov(x,y)/(sd(x)^2) ; b=mean(y)-a\*mean(x) ; abline(b,a) ; a.prime=cov(x,y)/(sd(y)^2) ; b.prime=mean(x)-a.prime\*mean(y) ; abline(-b.prime/a.prime, 1/a.prime) ; matrix(c(a, b, a.prime, b.prime), ncol=2, dimnames=list(c("a", "b"), c("y en x", "x en y"))) ; a.sec=ifelse(a>=0, sd(y)/sd(x), -sd(y)/sd(x)) ; b.sec=mean(y)-a.sec\*mean(x) ; abline(b.sec, a.sec, col="red")**

**cor.test(obs1[,1], obs1[,2])**

**n<-length(obs2[,1])**

**n**

**r.xy<-cor(obs2[,1], obs2[,2])**

**r.xy**

**tr.calc<-r.xy\*(n-2)^0.5/(1-r.xy^2)^0.5**

**tr.calc**

**min(pt(tr.calc,(n-2),lower.tail=FALSE), pt(tr.calc,(n-2)))\*2**

**pf(tr.calc^2,1, (n-2), lower.tail=FALSE)**

## 5.22. Test du coefficient de corrélation de Spearman

### 5.22.1. Méthode

### 5.22.2. Exemples avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[which(mais$Parcelle=="Est"), c(2,5)]**

**# obs1 <- as.data.frame(matrix(c(c(1, 2, 3, 4), c(1, 2, 4, 3)), ncol=2))**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**if(any(table(obs2[, 1])>1)){cat("Certaines valeurs de", names(obs2)[1], "sont identiques (le nombre de répétitions figure dessous) :\n") ; print(table(obs2[, 1])[which(table(obs2[, 1])>1)])} else {cat("Pas de valeurs identiques dans", names(obs2)[1], "\n")}**

**if(any(table(obs2[, 2])>1)){cat("Certaines valeurs de", names(obs2)[2], "sont identiques (le nombre de répétitions figure dessous) :\n") ; print(table(obs2[, 2])[which(table(obs2[, 2])>1)])} else {cat("Pas de valeurs identiques dans", names(obs2)[2], "\n")}**

**rxi<-rank(obs2[,1],ties.method = "average")**

**ryi<-rank(obs2[,2],ties.method = "average")**

**obs3<-data.frame(rxi,ryi)**

**names(obs3)<-c(paste("rang de", names(obs2)[1]), paste("rang de", names(obs2)[2]))**

**obs3**

**windows()**

**plot(obs3, pch=16, cex=1.2, cex.lab=1.5, cex.axis=1.5, bty="l")**

**# a=cov(rxi,ryi)/(sd(rxi)^2) ; b=mean(ryi)-a\*mean(rxi) ; abline(b,a) ; a.prime=cov(rxi,ryi)/(sd(ryi)^2) ; b.prime=mean(rxi)-a.prime\*mean(ryi) ; abline(-b.prime/a.prime, 1/a.prime) ; matrix(c(a, b, a.prime, b.prime), ncol=2, dimnames=list(c("a", "b"), c("yr en xr", "xr en yr"))) ; a.sec=ifelse(a>=0, sd(ryi)/sd(rxi), -sd(ryi)/sd(rxi)) ; b.sec=mean(ryi)-a.sec\*mean(rxi) ; abline(b.sec, a.sec, col="red")**

**cor.test(obs1[,1], obs1[,2], method="spearman")**

**n<-length(obs2[, 1])**

**n**

**rho.xy<-(sum(rxi\*ryi) - n\*((n+1)/2)^2) / ((sum(rxi^2) - n\*((n+1)/2)^2)^0.5 \* (sum(ryi^2) -n\*((n+1)/2)^2)^0.5)**

**rho.xy**

**cor(rank(obs2[,1]), rank(obs2[,2]))**

**Scalc<-n\*(n^2-1)\*(1-rho.xy)/6**

**Scalc**

**sum((ryi-rxi)^2)**

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[which(mais$Parcelle=="Ouest"), c(2,5)]**

**obs3**

**cor.test(obs1[,1], obs1[,2], method="spearman")**

**n**

**rho.xy**

**Scalc<-n\*(n^2-1)\*(1-rho.xy)/6**

**Scalc**

**trho.calc<- rho.xy / ((1 - rho.xy^2)/(n-2))^0.5**

**trho.calc**

**min(pt(trho.calc, (n-2), lower.tail= FALSE), pt(trho.calc, (n-2)))\*2**

**Zrho.calc<- rho.xy\*(n-1)^0.5**

**min(pnorm(Zrho.calc, lower.tail= FALSE), pnorm(Zrho.calc))\*2**

**n<-4**

**library(gplots)**

**perm <- permutations(n, n)**

**rxi <- 1:n**

**serie.rho.xy <- as.vector(cor(rxi, t(perm)))**

**serie.VT <- n\*(n^2-1)\*(1-serie.rho.xy)/6**

**perm.VT <- cbind(as.data.frame(perm), serie.VT, serie.rho.xy)**

**# perm.VT**

**VT <- round(sort(unique(serie.VT)), 9)**

**prob.VT <- table(serie.VT)/length(serie.VT)**

**VT**

**prob.VT**

**plot(VT, as.vector(prob.VT), type="l", xlab="Valeur de S", ylab="Probabilité f(S)", cex.lab=1.5, cex.axis=1.5)**

**windows()**

**plot(-sort(unique(serie.rho.xy)), as.vector(table(serie.rho.xy) / length(serie.rho.xy)), type="l", xlab="Valeur de -rho.xy", ylab="Probabilité f(-rho.xy)", cex.lab=1.5, cex.axis=1.5)**

**p.value <- min(sum(prob.VT[VT >= round(Scalc, 9)]), sum(prob.VT[VT <= round(Scalc, 9)]))\*2**

**p.value**

## 5.23. Test du coefficient de corrélation de Kendall

### 5.23.1. Méthode

### 5.23.2. Exemples avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[which(mais$Parcelle=="Est"), c(2,5)]**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**if(any(table(obs2[, 1])>1)){cat("Certaines valeurs de", names(obs2)[1], "sont identiques (le nombre de répétitions figure dessous) :\n") ; print(table(obs2[, 1])[which(table(obs2[, 1])>1)])} else {cat("Pas de valeurs identiques dans", names(obs2)[1], "\n")}**

**if(any(table(obs2[, 2])>1)){cat("Certaines valeurs de", names(obs2)[2], "sont identiques (le nombre de répétitions figure dessous) :\n") ; print(table(obs2[, 2])[which(table(obs2[, 2])>1)])} else {cat("Pas de valeurs identiques dans", names(obs2)[2], "\n")}**

**n<-length(obs2[,1])**

**obs3<-obs2[order(obs2[,1]),]**

**concordant<-NULL; discordant<-NULL; tempo.c<-0; tempo.d<-0**

**for(i in 1:n){if(i == n){ concordant<-c(concordant,0) ; discordant<-c(discordant,0)} else{for (j in (i+1):n){if(obs3[i,1] == obs3[j,1]){} ; if(obs3[i,2] == obs3[j,2]){ tempo.c<-tempo.c+0.5 ; tempo.d<-tempo.d+0.5} ; if(all(obs3[i,] != obs3[j,])){if(all(obs3[i,] < obs3[j,]) | all(obs3[i,] > obs3[j,])){tempo.c<-tempo.c+1} else{tempo.d<-tempo.d+1}}} ; concordant<-c(concordant,tempo.c) ; discordant<-c(discordant,tempo.d);tempo.c<-0 ; tempo.d<-0}}**

**somme<-concordant + discordant**

**obs3<-cbind(obs3,concordant,discordant,somme)**

**obs3**

**windows()**

**plot(obs3[,3], obs3[,4], pch=16, cex=1.2, xlab="Concordance", ylab=" Discordance", xlim=range(0,n), ylim=range(0,n), cex.lab=1.5, cex.axis=1.5, bty="l")**

**cor.test(obs1[,1], obs1[,2], method="kendall")**

**n<-length(obs2[, 1])**

**n**

**tau.xy<-(sum(concordant)-sum(discordant))/(sum(concordant)+sum(discordant))**

**tau.xy**

**Tcalc<- n\*(n-1)\*(tau.xy+1)/4**

**Tcalc**

**obs3**

**cor.test(obs1[,1], obs1[,2], method="kendall")**

**n<-length(obs2[, 1])**

**n**

**txi<-table(obs2[, 1])**

**tyj<-table(obs2[, 2])**

**tau.xy<-(sum(concordant)-sum(discordant))/((n\*(n-1)/2-sum(txi\*(txi-1)/2))\* (n\*(n-1)/2- sum(tyj\*(tyj-1)/2)))^0.5**

**tau.xy**

**A<-(n\*(n-1)\*(2\*n+5) - sum(txi\*(txi-1)\*(2\*txi+5)) - sum(tyj\*(tyj-1)\*(2\*tyj+5)))/18**

**B<- sum(txi\*(txi-1)) \* sum(tyj\*(tyj-1))/(2\*n\*(n-1))**

**C<-sum(txi\*(txi-1)\*(txi-2)) \* sum(tyj\*(tyj-1)\*(tyj-2))/(9\*n\*(n-1)\*(n-2))**

**Ztau.calc<-(sum(concordant)-sum(discordant))/(A+B+C)^0.5**

**Ztau.calc**

**min(pnorm(Ztau.calc, lower.tail= FALSE), pnorm(Ztau.calc))\*2**

## 5.24. Test de χ2

### 5.24.1. Méthode

### 5.24.2. Exemple avec R

**c<-nrow(obs3)**

**k<-ncol(obs3)**

**n<-margin.table(obs3)**

**chi2.calc<-as.double(suppressWarnings(chisq.test(obs3)$statistic))**

**phi.FG<-(chi2.calc/n)^0.5**

**C.FG<-(chi2.calc/(chi2.calc+n))^0.5**

**T.FG<-(chi2.calc/(n\*((k-1)\*(c-1))^0.5))^0.5**

**V.FG<-(chi2.calc/(n\*min((k-1),(c-1))))^0.5**

**sumk.max.nij<-sum(apply(obs3,1,max))**

**sumc.max.nij<-sum(apply(obs3,2,max))**

**maxni.<-max(margin.table(obs3,1))**

**maxn.j<-max(margin.table(obs3,2))**

**lambda.FG<-(sumk.max.nij + sumc.max.nij - maxni. - maxn.j)/(2\*n - maxni. - maxn.j)**

**resultat<-data.frame(phi.FG, C.FG, T.FG, V.FG, lambda.FG)**

**resultat**

## 5.25. Tests de corrélations multiples

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[which(mais$Parcelle=="Est"), c(2,3,4)]**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs1**

**obs2<-na.omit(obs1)**

**cor(obs1, method="spearman", use= "pairwise.complete.obs")**

**graphics.off()**

**windows()**

**plot(obs1, pch=16, cex.axis=2)**

**# windows()**

**# library(ellipse) ; plotcorr(cor(obs2, method="pearson", use= "pairwise.complete.obs"))**

**c<-length(obs1)**

**variable.1<-NULL; variable.2<-NULL; test<-NULL; coef.correlation<-NULL; p.value<-NULL**

**for(col1 in 1:(c-1)){for(col2 in (col1+1):c){ variable.1<-c(variable.1, names(obs1)[col1]) ; variable.2<-c(variable.2, names(obs1)[col2]) ; test<-c(test,"Spearman") ; coef.correlation <- c(coef.correlation, cor(obs1[, col1], obs1[, col2], method="spearman", use= "complete.obs")) ; p.value<-c(p.value,round(suppressWarnings(cor.test(obs1[,col1], obs1[,col2], method="spearman")$p.value), 9))}}**

**p.mult<-data.frame(variable.1, variable.2, test, coef.correlation, p.value)**

**p.mult**

**p.mult2<-transform(p.mult, Holm=round(p.adjust(p.mult$p.value,method="holm"),8))**

**p.mult2**

5.26. Ajustement d'une distribution observée à une distribution théorique

### 5.26.1. Introduction

### 5.26.2. Test de χ2 de conformité

#### 5.26.2.1. Méthode

#### 5.26.2.2. Exemple avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-data.frame(mais[which(mais$Parcelle=="Est"), 2])**

**names(obs1)<-c("Hauteur")**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**obs3<-hist(obs2[,1], plot=FALSE)$counts**

**bornes<- hist(obs2[,1], plot=FALSE)$breaks**

**# obs3<-c(1, 2, 4, 6, 7, 5, 3, 3, 1) ; bornes<-c(180, 200, 220, 240, 260, 280, 300, 320, 340, 360)**

**bornes[1] <- -Inf**

**bornes[length(bornes)] <- Inf**

**names(obs3)<- paste("[",bornes[-length(bornes)],",", bornes[-1], "[", sep = "")**

**obs3**

**mu<-270 ; sigma<-25**

**n<-length(obs2[,1])**

**ptheo<- pnorm(bornes[-1],mu, sigma) - pnorm(bornes[-length(bornes)], mu, sigma)**

**ptheo**

**sum(ptheo)**

**obs3theo<-ptheo\*n**

**names(obs3theo)<-names(obs3)**

**obs3theo**

**last.name<- names(obs3theo)[length(names(obs3theo))]**

**last.name <-sub("[","",last.name,fixed = TRUE)**

**last.name <-sub("[","",last.name,fixed = TRUE)**

**last.name <-strsplit(last.name,split=",")[[1]][2]**

**first.name<- names(obs3theo)[length(names(obs3theo))-1]**

**first.name <-sub("[","",first.name,fixed = TRUE)**

**first.name <-sub("[","",first.name,fixed = TRUE)**

**first.name <-strsplit(first.name,split=",")[[1]][1]**

**obs3theo[length(obs3theo)-1] <- obs3theo[length(obs3theo)] + obs3theo[length(obs3theo)-1]**

**obs3theo<- obs3theo[-length(obs3theo)]**

**names(obs3theo)[length(obs3theo)]<- paste("[", first.name, ",", last.name, "[", sep = "")**

**obs3 [length(obs3)-1] <- obs3 [length(obs3)] + obs3 [length(obs3)-1]**

**obs3<- obs3[-length(obs3)]**

**names(obs3)[length(obs3)]<- paste("[", first.name, ",", last.name, "[", sep = "")**

**obs3theo**

**obs3**

**first.name<- names(obs3theo)[1]**

**first.name <-sub("[","",first.name,fixed = TRUE)**

**first.name <-sub("[","",first.name,fixed = TRUE)**

**first.name <-strsplit(first.name,split=",")[[1]][1]**

**last.name<- names(obs3theo)[2]**

**last.name <-sub("[","",last.name,fixed = TRUE)**

**last.name <-sub("[","",last.name,fixed = TRUE)**

**last.name <-strsplit(last.name,split=",")[[1]][2]**

**obs3theo[2] <- obs3theo[1] + obs3theo[2]**

**obs3theo<- obs3theo[-1]**

**names(obs3theo)[1]<- paste("[", first.name, ",", last.name, "[", sep = "")**

**obs3 [2]<-obs3[1] + obs3[2]**

**obs3<-obs3 [-1]**

**names(obs3)[1]<- paste("[", first.name, ",", last.name, "[", sep = "")**

**obs3theo**

**obs3**

**obs3theo**

**obs3**

**graphics.off()**

**mu<-270**

**sigma<-25**

**histo<-hist(obs1[, 1], main=NULL, freq=FALSE, ylim=range(0, max(dnorm(mu, mu, sigma), max(hist(obs1[, 1])$density))), xlab="Distribution de la hauteur (cm)",ylab="Densité de la distribution", cex.lab=1.5, cex.axis=1.5)**

**x<-seq(min(histo$breaks), max(histo$breaks),0.01)**

**lines(x, dnorm(x,mu,sigma))**

**windows()**

**barplot(t(matrix(c(obs3, obs3theo),ncol=2)), beside=TRUE, ylim=range(0, max(obs3,obs3theo)\*1.3), names.arg=names(obs3), xlab=names(obs2), col=c("grey","black"), cex=1.5, cex.axis=1.5, cex.lab=1.5)**

**legend("topright", c("Effectifs observés", "Effectifs théoriques"), pch=15, pt.cex=3, col=c("grey", "black"), bty="n", cex=1.5, y.intersp=1.5)**

**n<-length(obs2[,1])**

**chisq.test(obs3, p=obs3theo/n)**

**n<-length(obs2[, 1])**

**n**

**ni<-obs3**

**ti<- obs3theo**

**chi2.calc<-sum((ni-ti)^2/ti)**

**chi2.calc**

**k<-length(obs3)**

**par<-1**

**nu<-k-par**

**nu**

**pchisq(chi2.calc,nu,lower.tail=FALSE)**

### 5.26.3. Test de Kolmogorov-Smirnov

#### 5.26.3.1. Méthode

#### 5.26.3.2. Exemple avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-data.frame(mais[which(mais$Parcelle=="Est"), 2])**

**names(obs1)<-c("Hauteur")**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**if(any(table(obs2[, 1])>1)){cat("Attention car certaines valeurs de", names(obs2)[1], "sont identiques (le nombre de répétitions figure dessous) :\n") ; print(table(obs2[, 1])[which(table(obs2[, 1])>1)])} else {cat("Pas de valeurs identiques dans", names(obs2)[1], "\n")}**

**n<-length(obs2[,1])**

**pr<-rep(1/n,n)**

**Fobs1<-cumsum(pr)-1/n**

**Fobs2<-cumsum(pr)**

**Ftheo<-pnorm(sort(obs2[,1]),270,25)**

**Dr1 <- Ftheo - Fobs1**

**Dr2<- Fobs2 - Ftheo**

**obs3<-data.frame(c(1:n),sort(obs2[,1]),Ftheo,pr,Fobs1,Fobs2,Dr1,Dr2)**

**names(obs3)[1:2]<-c("Rang",names(obs2))**

**obs3**

**graphics.off()**

**mu<-270**

**sigma<-25**

**histo<-hist(obs1[, 1], main=NULL, freq=FALSE, ylim=range(0, max(dnorm(mu, mu, sigma), max(hist(obs1[, 1])$density))), xlab="Distribution de la hauteur (cm)",ylab="Densité de la distribution", cex.lab=1.5, cex.axis=1.5)**

**x<-seq(min(histo$breaks), max(histo$breaks),0.01)**

**lines(x, dnorm(x,mu,sigma))**

**windows()**

**xr<-sort(obs2[, 1])**

**plot(xr, Ftheo, type="s", lwd=3, ylim=c(0,1), xlab=names(obs2), ylab="Probabilité", bty="l", cex.lab=1.5, cex.axis=1.5)**

**lines(xr, Fobs1, type="s", lty=1)**

**lines(xr, Fobs2, type="s", lty=5)**

**arrows(xr[ifelse(max(abs(Dr1)) > max(abs(Dr2)), order(abs(Dr1))[length(xr)], order(abs(Dr2))[length(xr)])], 0, xr[ifelse(max(abs(Dr1)) > max(abs(Dr2)), order(abs(Dr1))[length(xr)], order(abs(Dr2))[length(xr)])], 0.1, lwd=3)**

**legend("bottomright", c("Fthéo", "Fobs1", "Fobs2"), lty=c(1,1,5), lwd=c(3,1,1), bty="n", cex=1.5,y.intersp=1.5)**

**windows()**

**plot(xr, Ftheo, type="o", lwd=3, cex=1.2, ylim=c(0,1), xlab=names(obs2), ylab="Probabilité", bty="l", cex.lab=1.5, cex.axis=1.5)**

**lines(xr, Fobs1, type="o", cex=1.2, lty=1)**

**lines(xr, Fobs2, type="o", cex=1.2, lty=5)**

**arrows(xr[ifelse(max(abs(Dr1)) > max(abs(Dr2)), order(abs(Dr1))[length(xr)], order(abs(Dr2))[length(xr)])], 0, xr[ifelse(max(abs(Dr1)) > max(abs(Dr2)), order(abs(Dr1))[length(xr)], order(abs(Dr2))[length(xr)])], 0.1, lwd=3)**

**legend("bottomright", c("Fthéo", "Fobs1", "Fobs2"), lty=c(1,1,5), lwd=c(3,1,1), bty="n", cex=1.5,y.intersp=1.5)**

**ks.test(obs1[,1],"pnorm",270,25)**

**n<-length(obs2[,1])**

**n**

**D<-max(abs(Dr1), abs(Dr2))**

**D**

**tempo<-NULL**

**for(j in 0:floor(n\*(1-D))){tempo<-c(tempo, choose(n,j)\*(1-D-j/n)^(n-j)\*(D+j/n)^(j-1))}**

**p.value<-2\*D\*sum(tempo)**

**p.value**

**Fobs1<-1-cumsum(pr)+1/n**

**Fobs2<-1-cumsum(pr)**

**Ftheo<-pnorm(sort(obs2[,1]),270,25,lower.tail=FALSE)**

**Dr1 <- Fobs1 - Ftheo**

**Dr2<- Ftheo - Fobs2**

**windows()**

**qqplot(obs2[, 1], qnorm(ppoints(obs2[, 1]),270, 25), pch=16, cex=1.2, xlab=names(obs2), ylab = "Quantiles théoriques", bty="l", cex.main=1.5, cex.lab=1.5, cex.axis=1.5)**

**quantiles.obs<-sort(obs2[, 1])**

**n<-length(obs2[, 1])**

**n**

**if(n<=10){a=3/8} else{a=1/2}**

**Fobs<-(1:n-a)/(n+1-2\*a)**

**quantiles.theo<-qnorm(Fobs, 270, 25)**

**plot(quantiles.obs, quantiles.theo, pch=16, cex=1.2, xlab=names(obs2), ylab = "Quantiles théoriques", bty="l", cex.main=1.5, cex.lab=1.5, cex.axis=1.5)**

**ppoints(obs2[,1])**

**qqnorm(obs2[, 1], datax=TRUE, pch=16, cex=1.2, ylab=names(obs2), bty="l", cex.main=1.5, cex.lab=1.5, cex.axis=1.5)**

**qqline(obs2[,1], datax=TRUE)**

**q.line.obs<-quantile(quantiles.obs, probs=c(0.25, 0.75))**

**q.line.theo<- qnorm(c(0.25, 0.75))**

**a<-diff(q.line.theo) / diff(q.line.obs)**

**b<- q.line.theo[1]-a \* q.line.obs[1]**

**abline(b, a)**

### 5.26.4. Test de Shapiro-Wilk

#### 5.26.4.1. Méthode

#### 5.26.4.2. Exemple avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-data.frame(mais[which(mais$Parcelle=="Est"), 5])**

**names(obs1)<-c("Masse.grains")**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**obs2**

**graphics.off()**

**m<-mean(obs2[, 1])**

**s<-sd(obs2[, 1])**

**histo<-hist(obs1[, 1], main=NULL, freq=FALSE, ylim=range(0, max(dnorm(m, m, s), max(hist(obs1[, 1])$density))), xlab="Distribution de la hauteur (cm)", ylab="Densité de la distribution", cex.lab=1.5, cex.axis=1.5)**

**x<-seq(min(histo$breaks), max(histo$breaks), 0.01)**

**lines(x, dnorm(x, m, s))**

**windows()**

**qqnorm(obs2[, 1], datax=TRUE, pch=16, cex=1.2, ylab=names(obs2), bty="l", cex.main=1.5, cex.lab=1.5, cex.axis=1.5)**

**qqline(obs2[, 1], datax=TRUE)**

**n<-length(obs2[, 1])**

**m<- mean(obs2[, 1])**

**s<- sd(obs2[, 1])**

**param <- data.frame(n, m, s)**

**names(param) <- c("Effectif", "Moyenne", "Ecart.type")**

**param**

**shapiro.test(obs1[,1])**

**xr<-sort(obs2[, 1])**

**n<-length(xr)**

**k<-100000**

**obs3<-matrix(,nrow=k, ncol=n)**

**for(j in 1:k){q.jr<-sort(rnorm(n)) ; mj<-mean(q.jr) ; sj<-sd(q.jr) ; u.jr<-(q.jr-mj) / (sj\*(n-1)^0.5) ; obs3[j,]<-u.jr}**

**M<-matrix(apply(obs3, 2, mean), ncol=1)**

**V<-cov(obs3)**

**val.critique<-which(svd(V)$d<10^-8)**

**Diag.mod<-diag(1/svd(V)$d)**

**for(i in val.critique){Diag.mod[i, i]<-0}**

**V.inv<-svd(V)$v%\*%Diag.mod%\*% t(svd(V)$u)**

**ar<-as.vector(t(M)%\*%V.inv)/as.numeric(t(M)%\*%V.inv%\*%V.inv%\*%M)^0.5**

**mean(ar)**

**sum((ar - mean(ar))^2)**

**W<- sum(ar\*xr)^2/sum((xr-m)^2)**

**W**

**cor(ar,xr)^2**

**summary(lm(ar~xr))$r.squared**

**ar.approx<- sort(qqnorm(obs2[, 1], plot=FALSE, datax=TRUE, pch=16, cex=1.2, ylab=names(obs2), bty="l", cex.main=1.5, cex.lab=1.5, cex.axis=1.5)$y)**

**ar.approx<- (ar.approx -mean(ar.approx)) /(sd(ar.approx)\*(n-1)^0.5)**

**cor(ar.approx, xr)^2**

**abline(lm(sort(qqnorm(obs2[, 1], plot=FALSE)$x)~sort(obs2[, 1])), lwd=1.5, col="red")**

## 5.27. Comparaison de deux distributions observées (test de Kolmogorov-Smirnov)

### 5.27.1. Méthode

### 5.27.2. Exemple avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[which(mais$Parcelle=="Nord" | mais$Parcelle=="Sud"), c(5,11)]**

**# obs1 <- cbind(c(100, 200, 300, 400, 500), as.data.frame(c("O", "A", "O", "A", "A")))**

**obs1[,2]<-factor(obs1[,2])**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**if(any(table(obs2[, 1])>1)){cat("Attention car certaines valeurs de", names(obs2)[1], "sont identiques (le nombre de répétitions figure dessous) :\n") ; print(table(obs2[, 1])[which(table(obs2[, 1])>1)])} else {cat("Pas de valeurs identiques dans", names(obs2)[1], "\n")}**

**nc.max<-max(table(obs2[,2]))**

**nb.na<-nc.max- table(obs2[,2])**

**tempo<-split(obs2[,1], obs2[,2])**

**for(i in 1:length(tempo)){tempo[[i]]<-append(tempo[[i]],rep(NA,nb.na[i]))}**

**obs3<-data.frame(tempo)**

**xr<-sort(obs2[,1])**

**ci<-obs2[,2]**

**cr<-ci[order(obs2[,1])]**

**r<-1:length(cr)**

**obs4<-data.frame(r,xr,cr)**

**names(obs4)<- c("rang", names(obs2)[1], names(obs2)[2])**

**n1<-sapply(split(obs4[,2],obs4[,3]),length)[1]**

**n2<-sapply(split(obs4[,2],obs4[,3]),length)[2]**

**pr<-ifelse(obs4[,3]==names(obs3)[1],1/n1,1/n2)**

**F1.c1<-cumsum(rep(1/n1,n1))**

**F2.c2<-cumsum(rep(1/n2,n2))**

**F.xr<-NULL**

**F.xr[which(obs4[,3]==names(obs3)[1])]<-F1.c1**

**F.xr[which(obs4[,3]==names(obs3)[2])]<-F2.c2**

**F1<-NULL**

**F1[1:length(obs4[,2])]<-NA**

**F1[which(obs4[,3]==names(obs3)[1])]<- F.xr[which(obs4[,3]==names(obs3)[1])]**

**F2<-NULL**

**F2[1: length(obs4[,2])]<-NA**

**F2[which(obs4[,3]==names(obs3)[2])]<- F.xr[which(obs4[,3]==names(obs3)[2])]**

**dr<-NULL**

**for(i in 1:nrow(obs4)){dr<-c(dr, round(max(0, F1[1:i], na.rm=TRUE) - max(0, F2[1:i], na.rm=TRUE) ,9))}**

**obs5<-data.frame(obs4,pr,F.xr,F1,F2,dr)**

**if(length(unique(obs2[, 1]))!=length(obs2[, 1])){cat("Les lignes suivantes du tableau obs5 initial ont été enlevées arbitrairement :\n") ; cat(which(diff(obs5[, 2])==0), "\n") ; cat("Les individus appartenaient aux classes suivantes :\n") ; cat(as.character(obs5[which(diff(obs5[, 2])==0), 3]), "\n")}**

**obs5<-obs5[c(which(diff(obs5[,2])!=0), nrow(obs4)),]**

**obs5**

**windows()**

**plot(c(min(xr), obs5[which(obs5[, 3]==names(obs3)[1]), 2], max(xr)), c(0, obs5[which(obs5[, 3]==names(obs3)[1]), 6], 1), type="s", lwd=2, ylim=c(0,1), xlab=names(obs2)[1], ylab="Probabilité", bty="l", cex.lab=1.5, cex.axis=1.5)**

**lines(c(min(xr), obs5[which(obs5[,3]==names(obs3)[2]),2], max(xr)), c(0, obs5[which(obs5[,3]==names(obs3)[2]),7], 1), type="s", lty=5,lwd=2)**

**arrows(xr[order(abs(dr))[length(xr)]], 1, xr[order(abs(dr))[length(xr)]], 0.9, lwd=3)**

**legend("bottomright", c(paste("F1(xr)gauche", names(obs3)[1]), paste("F2(xr)gauche", names(obs3)[2])), lty=c(1, 5), lwd=c(2, 2), bty="n", cex=1.5, y.intersp=1.5)**

**ks.test(obs3[,1],obs3[,2])**

**n<-sapply(split(obs2[, 1], obs2[, 2]), length)**

**n**

**D.calc<-max(abs(obs5$dr))**

**D.calc**

**F1.c1<- (((n1-1):0)\*1/n1)**

**F2.c2<- (((n2-1):0)\*1/n2)**

**F.xr<-NULL**

**F.xr[which(obs4[,3]==names(obs3)[1])]<-F1.c1**

**F.xr[which(obs4[,3]==names(obs3)[2])]<-F2.c2**

**F1<-NULL**

**F1[1:length(obs4[,2])]<-NA**

**F1[which(obs4[,3]==names(obs3)[1])]<- F.xr[which(obs4[,3]==names(obs3)[1])]**

**F2<-NULL**

**F2[1: length(obs4[,2])]<-NA**

**F2[which(obs4[,3]==names(obs3)[2])]<- F.xr[which(obs4[,3]==names(obs3)[2])]**

**dr<-NULL**

**for(i in 1:nrow(obs4)){dr<-c(dr, round(min(1, F2[1:i], na.rm=TRUE)- min(1, F1[1:i], na.rm=TRUE) ,9))}**

**obs5<-data.frame(obs4,pr,F.xr,F1,F2,dr)**

**if(length(unique(obs2[, 1]))!=length(obs2[, 1])){cat("Les lignes suivantes du tableau obs5 initial ont été enlevées arbitrairement :\n") ; cat(which(diff(obs5[, 2])==0), "\n") ; cat("Les individus appartenaient aux classes suivantes :\n") ; cat(as.character(obs5[which(diff(obs5[, 2])==0), 3]), "\n")}**

**obs5<-obs5[c(which(diff(obs5[,2])!=0), nrow(obs4)),]**

**obs5**

**n1<-3**

**n2<-2**

**n<-n1+n2**

**vecteur.ini<-c(rep("A", n1), rep("O", n2))**

**library(gtools)**

**comb<-as.data.frame(unique(permutations(n, n, vecteur.ini, set=FALSE)))**

**tempo<-t(comb)**

**F1.c1<-cumsum(rep(1/n1,n1))**

**F2.c2<-cumsum(rep(1/n2,n2))**

**serie.D<-NULL**

**for(i in 1:ncol(tempo)){F1<-NULL ; F1[1:n]<-NA ; F1[which(tempo[,i]=="A")]<- F1.c1 ; F2<-NULL ; F2[1:n]<-NA ; F2[which(tempo[,i]=="O")]<- F2.c2**

**dr<-NULL ; for(j in 1:n){dr<-c(dr, round(max(0, F1[1:j], na.rm=TRUE) - max(0, F2[1:j], na.rm=TRUE) ,9))} ; serie.D <- c(serie.D, max(abs(dr)))}**

**comb.D<-cbind(comb, serie.D)**

**# comb.D**

**D<-round(sort(unique(serie.D)), 9)**

**prob.D<-table(serie.D)/length(serie.D)**

**prob.D**

**plot(D, as.vector(prob.D), pch=16, xlab="Valeur de D", ylab="Probabilité f(D)", cex.lab=1.5, cex.axis=1.5)**

**p.value <- sum(prob.D[D >= round(D.calc, 9)])**

**p.value**

5.28. Tests autour de la régression

### 5.28.1. Introduction

### 5.28.2. Principe de la régression linéaire simple

### 5.28.3. Comparaison d'une régression observée à une régression nulle

#### 5.28.3.1. Méthode

#### 5.28.3.2. Exemple avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[which(mais$Parcelle=="Est"), c(2,3)]**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**obs2**

**windows()**

**par(mar=c(5, 5, 6, 2))**

**plot(obs2[, 2], obs2[, 1], pch=16, cex=1.2, xlab=paste("Variable explicative :", names(obs2)[2]), ylab=paste("Variable expliquée :", names(obs2)[1]), cex.lab=1.5, cex.axis=1.5, bty="l")**

**abline(lm(obs2[, 1]~obs2[, 2]), lwd=1.5)**

**par(xpd=TRUE)**

**xi<-obs2[, 2] ; yi<-obs2[, 1]**

**legend("top", adj=c(0.5, -2.5), legend = as.expression(substitute(italic(R)^2 == r, list(r=summary(lm(yi~xi))$r.squared))), cex=1.7, bty="n")**

**yi<-obs2[,1]**

**xi<-obs2[,2]**

**summary(lm(yi~xi))**

**n<-length(yi)**

**n**

**my<-mean(yi)**

**my**

**ydi<-fitted(lm(yi~xi))**

**SCT<-sum((yi-my)^2)**

**SCR<-sum((ydi-my)^2)**

**SCE<-sum((yi-ydi)^2)**

**SCT**

**SCR+SCE**

**R2<-SCR/SCT**

**R2**

**R2^0.5**

**k<-length(obs2)-1**

**R2.ajust<-1-(1-R2)\*(n-k)/(n-k-1)**

**R2.ajust**

**Fcalc<-SCR/(SCE/(n-2))**

**Fcalc**

**pf(Fcalc,1, (n-2), lower.tail=FALSE)**

**windows()**

**par(mfrow=c(2,2))**

**n<-length(yi)**

**plot(ls.diag(lsfit(xi, yi))$hat, pch=16, cex=1.2, ylim = range(0, max(ls.diag(lsfit(xi, yi))$hat\*1.125, (4/n\*1.125))), xlab="Numéro de l'individu (i)", ylab="Leverage hi", cex.lab=1.2, cex.axis=1.2, cex.main=1.5)**

**abline(h=4/n, lty=2)**

**legend("topleft","Seuil d'influence (hi > 4/n)", lty=2, cex=0.9, bty="n")**

**plot(ls.diag(lsfit(xi, yi))$dfits, pch=16, cex=1.2, ylim = range(min(ls.diag(lsfit(xi, yi))$dfits, (-2\*(2/n)^0.5)), max(ls.diag(lsfit(xi, yi))$dfits\*1.4,(2\*(2/n)^0.5\*1.4))), xlab="Numéro de l'individu (i)", ylab="Dffits i", cex.lab=1.2, cex.axis=1.2, cex.main=1.5)**

**abline(h=(8/n)^0.5, lty=2)**

**abline(h=-(8/n)^0.5, lty=2)**

**legend("topleft", inset = c(-0.02,0), x.intersp = 0.5, "Seuil d'influence (Dffits > +/-(8/n)^0.5)", lty=2, cex=0.9, bty="n")**

**seuil.cook<-qf(0.5,2,(n-2))**

**plot(ls.diag(lsfit(xi, yi))$cooks, pch=16, cex=1.2, ylim=range(0, max(ls.diag(lsfit(xi, yi))$cooks\*1.125,(seuil.cook\*1.125))), xlab="Numéro de l'individu (i)", ylab="DCook i", cex.lab=1.2, cex.axis=1.2, cex.main=1.5)**

**abline(h=seuil.cook, lty=2)**

**legend("topleft","Seuil d'influence (D > F(0.5,2,n-2))", lty=2, cex=0.9, bty="n")**

**windows()**

**par(mfrow=c(2,2))**

**plot(lm(yi~xi), pch=16, cex=1.2, cex.lab=1.2, cex.axis=1.2, cex.main=1.5)**

**n<-length(yi)**

**mx<-mean(xi)**

**sx2<-var(xi)**

**hi<-1/n+(xi-mx)^2/((n-1)\*sx2)**

**hi**

**n<-length(yi)**

**coef.sans.i<-matrix(nrow=n, ncol=2, dimnames=dimnames(lm.influence(lm(yi ~ xi))$coefficients))**

**for(i in 1:n){coef.sans.i[i,]<- coef<-coef(lm(yi~xi)) - coef(lm(yi[-i]~xi[-i]))}**

**coef.sans.i**

**n<-length(yi)**

**sres.sans.i<-NULL; ydi.sans.i<-NULL; SCE.sans.i<-NULL**

**for(i in 1:n){ydi.sans.i<-fitted(lm(yi[-i]~xi[-i])) ; SCE.sans.i<-sum((yi[-i]-ydi.sans.i)^2) ; sres.sans.i<-c(sres.sans.i,(SCE.sans.i/(n-3))^0.5)}**

**sres.sans.i**

**n<-length(yi)**

**ei<-resid(lm(yi~xi))**

**ydi<-fitted(lm(yi~xi))**

**SCE<-sum((yi-ydi)^2)**

**sres<-(SCE/(n-2))^0.5**

**ei.std<-ei/(sres\*(1-hi)^0.5)**

**ei.std**

**ei<-resid(lm(yi~xi))**

**ei.stud<-ei/(sres.sans.i\*(1-hi)^0.5)**

**ei.stud**

**Dcook.i<- ei.std^2\*hi/(2\*(1-hi))**

**Dcook.i**

**dffits.i<- ei\*hi^0.5/(sres.sans.i\*(1-hi))**

**dffits.i**

**sx<-sd(xi)**

**sa<-sres/((n-1)^0.5\*sx)**

**sa**

**sb<-sres\*(sum(xi^2))^0.5/((n\*(n-1))^0.5\*sx)**

**sb**

**p <- lsfit(xi, yi)$qr$rank**

**qr <- as.matrix(lsfit(xi, yi)$qr$qr[1:p, 1:p])**

**qr[row(qr) > col(qr)] <- 0**

**qrinv <- solve(qr)**

**cov.unscaled <- qrinv %\*% t(qrinv)**

**cov.unscaled**

**cov.scaled<-cov.unscaled\*sres^2**

**cov.scaled**

**corr<-cov.scaled/matrix(c(sb^2,sa\*sb,sa\*sb,sa^2),ncol=2)**

**corr**

### 5.28.4. Comparaison d'une régression observée à une régression théorique

#### 5.28.4.1. Méthode

#### 5.28.4.2. Exemples avec R

**windows()**

**a<-cov(obs2[, 1], obs2[, 2])/var(obs2[, 2])**

**b<-mean(obs2[, 1])-a\*mean(obs2[, 2])**

**par(mar=c(5,5,6,2))**

**plot(obs2[, 2], obs2[, 1], pch=16, cex=1.2, xlim=range(0, obs2[, 2]), ylim=range(b,obs2[, 1]), xlab=paste("Variable explicative :", names(obs2)[2]), ylab=paste("Variable expliquée :", names(obs2)[1]), cex.lab=1.5, cex.axis=1.5, bty="l", xaxs="i", yaxs="i")**

**abline(lm(obs2[, 1]~obs2[, 2]), lwd=1.5)**

**par(xpd=TRUE)**

**text(x=0, y=max(obs2[, 1]), adj=c(-0.5,-4), paste("a = ", round(a,4)), cex=1.7)**

**text(x=0, y=max(obs2[, 1]), adj=c(-0.5,-2), paste("b = ", round(b,4)), cex=1.7)**

**yi<-obs2[,1]**

**xi<-obs2[,2]**

**summary(lm(yi~xi))**

**n<-length(yi)**

**n**

**sxy <- cov(xi, yi)**

**sx2 <- var(xi)**

**a<- sxy/sx2**

**a**

**mx<-mean(xi); my<-mean(yi)**

**b<-mean(yi)-a\*mean(xi)**

**b**

**ydi<-a\*xi+b**

**SCE<-sum((yi-ydi)^2)**

**sa<-(SCE/((n-2)\*(n-1)\*sx2))^0.5**

**sa**

**ta.calc<-a/sa**

**ta.calc**

**ta.calc^2**

**sres2<-SCE/(n-2)**

**sb<-(sres2\*sum(xi^2)/(n\*(n-1)\*sx2))^0.5**

**sb**

**tb.calc<-b/sb**

**tb.calc**

**min(pt(ta.calc,(n-2),lower.tail=FALSE), pt(ta.calc,(n-2)))\*2**

**min(pt(tb.calc,(n-2),lower.tail=FALSE), pt(tb.calc,(n-2)))\*2**

**yi<-obs2[,1]**

**xi<-obs2[,2]**

**a.theo<-0.2**

**sxy <- cov(xi, yi)**

**sx2 <- var(xi)**

**a<- sxy/sx2**

**a**

**n<-length(yi)**

**ydi<-a\*xi+b**

**SCE<-sum((yi-ydi)^2)**

**sa<-(SCE/((n-2)\*(n-1)\*sx2))^0.5**

**ta.calc<-(a-a.theo)/sa**

**ta.calc**

**min(pt(ta.calc,(n-2),lower.tail=FALSE), pt(ta.calc,(n-2)))\*2**

**b.theo<-50**

**mx<-mean(xi); my<-mean(yi)**

**b<-mean(yi)-a\*mean(xi)**

**b**

**sres2<-SCE/(n-2)**

**sb<-(sres2\*sum(xi^2)/(n\*(n-1)\*sx2))^0.5**

**tb.calc<-(b-b.theo)/sb**

**tb.calc**

**min(pt(tb.calc,(n-2),lower.tail=FALSE), pt(tb.calc,(n-2)))\*2**

## 5.29. Test autour de la survie

### 5.29.1. Introduction

### 5.29.2. Comparaison de deux courbes de survie (test du logrank)

#### 5.29.2.1. Méthode

#### 5.29.2.2. Exemple avec R

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[which(mais$Parcelle=="Est" | mais$Parcelle=="Ouest"), c(14,15,11)]**

**obs1[,3]<-factor(obs1[,3])**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**# obs2<-rbind(obs2[1:32,],obs2[1:32,]) ; obs2[33:64,3]<-"Ouest" ; obs2[,2]<-1**

**# obs2<-rbind(obs2[1:32,],obs2[1:32,]) ; obs2[33:64,3]<-"Ouest" ; obs2[2:32,2]<-0 ; obs2[33:64,2]<-1**

**# obs2[33:67,1]<-c(seq(12, 60, length.out=5), seq(61, 70, length.out=25), seq(71, 130, length.out=5)) ; obs2[33:67,2]<-1**

**library(survival)**

**obs3<-summary(survfit(Surv(obs2[,1], obs2[,2])~ obs2[,3], error="greenwood", conf.type="plain"))**

**obs3**

**tr<- unique(sort(obs3$time))**

**time.1<- split(obs3$time,obs3$strata)[[1]]**

**time.2<- split(obs3$time,obs3$strata)[[2]]**

**np.r1.tempo<-split(obs3$n.risk,obs3$strata)[[1]]**

**n.r1.tempo <-split(obs3$n.event,obs3$strata)[[1]]**

**np.r2.tempo <-split(obs3$n.risk,obs3$strata)[[2]]**

**n.r2.tempo <-split(obs3$n.event,obs3$strata)[[2]]**

**n1<-table(obs2[,3])[1]**

**n2<-table(obs2[,3])[2]**

**n.r1<-NULL; count<-0**

**for (i in 1:length(tr)){if(any(tr[i]==time.1)){n.r1 <- c(n.r1, n.r1.tempo[i-count])} else {n.r1<-c(n.r1, 0) ; count<-count+1}}**

**n.r2<-NULL; count<-0**

**for(i in 1:length(tr)){if(any(tr[i]==time.2)){n.r2 <- c(n.r2, n.r2.tempo[i-count])} else {n.r2<-c(n.r2, 0) ; count<-count+1}}**

**time.cs<-split(obs2[which(obs2[, 2]==0), 1], obs2[which(obs2[, 2]==0), 3])**

**na.r1.tempo<-c(0,n.r1[-length(n.r1)])**

**if(length(time.cs[[1]]) > 0){for(i in 1:length(time.cs[[1]])){if(time.cs[[1]][i] >= max(tr)){} else{if(time.cs[[1]][i] < min(tr)){ na.r1.tempo[1]<-na.r1.tempo[1] + as.double(table(time.cs[[1]][i]))} else{ coord<-max(which(tr <= time.cs[[1]][i])) ; na.r1.tempo[coord+1]<-na.r1.tempo[coord+1] + as.double(table(time.cs[[1]][i]))}}}}**

**na.r1<-cumsum(na.r1.tempo)**

**na.r2.tempo<-c(0,n.r2[-length(n.r2)])**

**if(length(time.cs[[2]]) > 0){for(i in 1:length(time.cs[[2]])){if(time.cs[[2]][i] >= max(tr)){} else{if(time.cs[[2]][i] < min(tr)){ na.r2.tempo[1]<-na.r2.tempo[1] + as.double(table(time.cs[[2]][i]))} else{ coord<-max(which(tr <= time.cs[[2]][i])) ; na.r2.tempo[coord+1]<-na.r2.tempo[coord+1] + as.double(table(time.cs[[2]][i]))}}}}**

**na.r2<-cumsum(na.r2.tempo)**

**np.r1<-n1-na.r1**

**np.r2<-n2-na.r2**

**np.r<-np.r1+np.r2**

**n.r<-n.r1+n.r2**

**obs4<-data.frame(tr, na.r1, np.r1, n.r1, na.r2, np.r2, n.r2, np.r, n.r)**

**cat("La classe 1 est", levels(obs2[, 3])[1], "et la classe 2", levels(obs2[, 3])[2], "\n")**

**obs4**

**graphics.off()**

**library(survival)**

**windows()**

**plot(survfit(Surv(obs2[, 1], obs2[, 2])~obs2[,3]), conf.int=FALSE, lty=c(1,2), lwd=2, mark="/", cex=1.5, firstx=min(tr)\*0.8, xlab="Durée (nombre de jours)", ylab="Proportion d'individus sans statut final", cex.lab=1.5, cex.axis=1.5, bty="l")**

**legend("bottomleft", c(paste(names(obs2)[3], levels(obs2[, 3]))), lty=c(1, 2), lwd=c(2, 2), bty="n", cex=1.5, y.intersp=1.5)**

**windows()**

**y1 <- log(-log(unlist(split(obs3$surv, obs3$strata)[1])))**

**y2 <- log(-log(unlist(split(obs3$surv, obs3$strata)[2])))**

**x1 <- unlist(split(obs3$time, obs3$strata)[1])[is.finite(y1)]**

**x2 <- unlist(split(obs3$time, obs3$strata)[2])[is.finite(y2)]**

**plot(x1, y1[is.finite(y1)], xlim=range(obs3$time), pch=16, cex=1.5, xlab="Durée (nombre de jours)", ylab="ln[-ln(Pr)]", ylim=range(y1[is.finite(y1)], y2[is.finite(y2)]), cex.lab=1.5, cex.axis=1.5, bty="l")**

**points(x2, y2[is.finite(y2)], pch=21, cex=1.5)**

**legend("bottomright", c(paste(names(obs2)[3], levels(obs2[, 3]))), pch=c(16, 21), bty="n", cex=1.5, pt.cex=2, y.intersp=1.5)**

**windows()**

**plot(c(min(obs3$time), unlist(split(obs3$time, obs3$strata)[1])), c(1, unlist(split(obs3$surv, obs3$strata)[1])), ylim=c(0, 1), xlim=range(obs3$time), type="s", lty=1, lwd=2, cex=1.5, xlab="Durée (nombre de jours)", ylab="Proportion d'individus sans statut final", cex.lab=1.5, cex.axis=1.5, bty="l")**

**lines(c(min(obs3$time), unlist(split(obs3$time, obs3$strata)[2])), c(1, unlist(split(obs3$surv, obs3$strata)[2])), ylim=c(0, 1), type="s", lty=2, lwd=2)**

**library(survival)**

**cat("La VT LR1 est calculée avec les données de la classe", levels(obs2[, 3])[1], "\n")**

**survdiff(Surv(obs2[,1], obs2[,2])~obs2[,3])**

**nc<- table(obs2[,3])**

**nc**

**table(obs2[,2], obs2[,3])[1,]**

**survfit(Surv(obs2[,1], obs2[,2])~obs2[,3])**

**O1<-sum(n.r1)**

**O1**

**E1<-sum(np.r1\*n.r/np.r)**

**E1**

**m<-length(np.r)**

**if (np.r[m]==1){mss0<-m-1} else{ mss0<-m}**

**s2.O1E1<-sum((np.r1\*np.r2\*(np.r-n.r)\*n.r/(np.r^2\*(np.r-1)))[1:mss0])**

**s2.O1E1**

**LR1.calc<-(O1-E1)^2/s2.O1E1**

**LR1.calc**

**pchisq(LR1.calc, 1,lower.tail=FALSE)**

**O2<-sum(n.r2)**

**O2**

**E2<-sum(np.r2\*n.r/np.r)**

**E2**

**s2.O2E2<- s2.O1E1**

**LR2.calc<-(O2-E2)^2/s2.O2E2**

**LR2.calc**

**pchisq(LR2.calc, 1,lower.tail=FALSE)**

**O1+O2**

**E1+E2**

**abs(O1-E1)**

**abs(O2-E2)**

**RR <- (O1/E1)/(O2/E2)**

**RR**

**obs5<-array(rep(NA, 4\* nrow(obs4)), c(2,2, nrow(obs4)), dimnames=list(levels(obs2[, 3]), c("Avec.statut.fin", "Sans.statut.fin"), paste("t", tr, sep="")))**

**for(i in 1:nrow(obs4)){obs5[,,i]<-matrix(c(obs4[i,4], obs4[i,7], (obs4[i,3]- obs4[i,4]), (obs4[i,6]- obs4[i,7])), nrow=2, byrow=FALSE)}**

**obs5<-as.table(obs5)**

**if((E1 - sum(pmax(0, (np.r1 - (np.r - n.r))))) >= 5 & (sum(pmin(np.r1, n.r)) - E1) >= 5){cat("Les conditions d'application du test sont remplies\n")} else{cat("Les conditions d'application du test ne sont pas remplies\n")}**

**mantelhaen.test(obs5, exact=FALSE, correct=FALSE)**

**m1<-table(obs3$strata)[1]**

**m2<-table(obs3$strata)[2]**

**q.j1<-(obs3$n.event/obs3$n.risk)[1:m1]**

**q.j2<-(obs3$n.event/obs3$n.risk)[(m1+1):(m1+m2)]**

**Pr1<-cumprod(1-q.j1)**

**Pr2<-cumprod(1-q.j2)**

**s.Pr1<-Pr1 \* (cumsum(q.j1/(obs3$n.risk[1:m1] \* (1-q.j1))))^0.5**

**s.Pr1**

**s.Pr2<-Pr2 \* (cumsum(q.j2/(obs3$n.risk[(m1+1):(m1+m2)] \* (1-q.j2))))^0.5**

**s.Pr2**

**Lower.CI.1<-Pr1 +qnorm(0.025) \* s.Pr1**

**Lower.CI.1**

**Lower.CI.2<-Pr2 +qnorm(0.025) \* s.Pr2**

**Lower.CI.2**

**Upper.CI.1<-Pr1 +qnorm(0.025, lower.tail=FALSE) \* s.Pr1**

**Upper.CI.1**

**Upper.CI.2<-Pr2 +qnorm(0.025, lower.tail=FALSE) \* s.Pr2**

**Upper.CI.2**

# ANNEXES

### 1. Formule développée de la variance et de la covariance

### 2. L'estimateur

### 3. Distribution normale de variables mesurées et théorème central limite

**hist(rnorm(10,165,7), main=NULL, freq=FALSE, xlab="Variable continue x", ylab="Densité f(x)", col="grey", cex.lab=1.5, cex.axis=1.5)**

### 4. Rappel des moyennes et variances des distributions de probabilité

### 5. Rappel sur les combinaisons

**n=4**

**k=2**

**choose(n,k)**

**factorial(n)/( factorial(k)\*factorial(n-k))**

### 6. Passage du χ2 au Z2 dans le cas de la comparaison d'une proportion observée à une proportion théorique

### 7. Passage du χ2 au Z2 dans le cas de la comparaison de deux proportions observées

### 8. Retrouver la formule de la VT à partir de la formule du χ2 dans le cas de la comparaison de plusieurs proportions observées

### 9. Estimation de la fluctuation de la VT χ2 avec correction de continuité de Yates

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[,c(9,10)]**

**obs2<-na.omit(obs1)**

**for(i in 1:length(obs2)){obs2[,i]<-factor(obs2[,i])}**

**obs3<- table(obs2)**

**theo<-chisq.test(obs3)$expected**

**theo**

**n<-margin.table(theo)**

**p.non.verse<-margin.table(theo,1)[1]/n**

**p.non.attaque<-margin.table(theo,2)[1]/n**

**chi2<-NULL; yates<-NULL**

**for(i in 1:5000){non.verse<-rbinom(1, n, p.non.verse) ; non.attaque<-rbinom(1, n, p.non.attaque) ; obs<-outer(c(non.verse, n-non.verse), c(non.attaque, n-non.attaque))/n ; chi2<-c(chi2, sum((obs - theo)^2/ theo)) ; yates<-c(yates,sum((abs(obs -theo)-0.5)^2/ theo))}**

**h0<-hist(chi2, plot=FALSE)**

**hY<- hist(yates, plot=FALSE)**

**plot(h0$mids,h0$density, type="l", lwd=2, xlim=c(0,20), ylim=c(0,0.4), xlab = expression(paste("Valeurs de ", chi^2, " calculées")), ylab="Densité de la distribution", cex.lab=1.5, cex.axis=1.5, bty="l")**

**lines(hY$mids,hY$density, lty=2, lwd=2)**

**x=seq(-0.1,40,0.01)**

**nu=1**

**lines(x, dchisq(x, nu),lty=3, lwd=2)**

**legend("topright",c("Sans correction de Yates","Avec correction de Yates", expression(paste("Loi de ", chi^2, " à 1 ddl"))), lty=c(1:3), lwd=2, cex=1.5, bty="n", y.intersp=1.5)**

### 10. Comment se comportent les différents couples de proportions (pG1/F1 , pG1/F2), (pG2/F1 , pG2/F2), (pF1/G1 , pF1/G2) et (pF2/G1 , pF2/G2) lors d'un test exact de Fisher sur tableau de contingence 2 × 2

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt", header=TRUE)**

**obs1<-mais[, c(7, 9)]**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<-length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**for(i in 1:length(obs2)){obs2[,i]<-factor(obs2[,i])}**

**obs3<- table(obs2)**

**addmargins(obs3)**

**n11<-obs3[1, 1]**

**n1.<-margin.table(obs3, 1)[1]**

**n21<-obs3[2, 1]**

**n2.<-margin.table(obs3, 1)[2]**

**pG1.F1<-n11/n1.**

**pG1.F1**

**pG1.F2<-n21/n2.**

**pG1.F2**

**fisher.test(obs3)$p**

**fisher.test(obs3, alternative="greater")$p**

**fisher.test(obs3, alternative="less")$p**

**obs3<- table(obs2)**

**tempo<-obs3[, 1]**

**obs3[, 1]<-obs3[, 2]**

**obs3[, 2]<-tempo**

**dimnames(obs3)[[2]][1]<-names(tempo)[2]**

**dimnames(obs3)[[2]][2]<-names(tempo)[1]**

**addmargins(obs3)**

**pG1.F1**

**pG1.F2**

**fisher.test(obs3)$p**

**fisher.test(obs3, alternative="greater")$p**

**fisher.test(obs3, alternative="less")$p**

**obs3<- table(obs2)**

**obs3<-t(obs3)**

**addmargins(obs3)**

**pG1.F1**

**pG1.F2**

**fisher.test(obs3)$p**

**fisher.test(obs3, alternative="greater")$p**

**fisher.test(obs3, alternative="less")$p**

**obs3<- table(obs2)**

**tempo<-obs3[1, ]**

**obs3[1, ]<-obs3[2, ]**

**obs3[2, ]<-tempo**

**obs3<-t(obs3)**

**dimnames(obs3)[[2]][1]<-names(tempo)[2]**

**dimnames(obs3)[[2]][2]<-names(tempo)[1]**

**addmargins(obs3)**

**pG1.F1**

**pG1.F2**

**fisher.test(obs3)$p**

**fisher.test(obs3, alternative="greater")$p**

**fisher.test(obs3, alternative="less")$p**

### 11. Anova et régression linéaire sont liées

**mais<-read.table("C:/Users/Gael/Desktop/mais.txt",header=TRUE)**

**obs1<-mais[, c(2,11)]**

**obs1[,2]<-factor(obs1[,2])**

**if(any(is.na(obs1) == TRUE)){nombre.lig.na<- length(table(which(is.na(obs1) == TRUE, arr.ind = TRUE)[,1]))} else{nombre.lig.na<-0}**

**cat("Nombre de lignes avec NA :", nombre.lig.na, "\n")**

**obs2<-na.omit(obs1)**

**n<-length(obs2[,1])**

**k<-nlevels(obs2[,2])**

**obs3<-as.data.frame(matrix(c(rep(1, n), rep(0, n\*(k-1))), byrow=FALSE, nrow=n, dimnames=list(NULL, levels(obs2[, 2]))))**

**for(i in 1:n){obs3[i,as.character(obs2[i,2])]<-1}**

**obs4<-cbind(obs2,obs3)**

**obs4**

**xi<-obs2[,2]**

**yi<-obs2[,1]**

**anova(lm(yi ~ xi))**

**oneway.test(yi ~ xi, var.equal=TRUE)**

**X<-as.matrix(obs3)**

**Y<-as.matrix(obs2[,1])**

**coef<-solve(t(X)%\*%X)%\*%t(X)%\*%Y**

**SCE<-(t(Y)%\*%Y)-t(coef)%\*%t(X)%\*%Y**

**SCE**

**Xred<-as.matrix(obs3[,1])**

**coef.red<-solve(t(Xred)%\*%Xred)%\*%t(Xred)%\*%Y**

**SCT<-(t(Y)%\*%Y)-t(coef.red)%\*%t(Xred)%\*%Y**

**SCI<-SCT-SCE**

**SCI**

**lapply(split(yi,xi),mean)**

**b<-coef[1,1]**

**m1<-data.frame(b,row.names=rownames(coef)[1])**

**m1**

**mc<-coef[-1,1]+b**

**mc**

**e2<-resid(lm(yi ~ xi))[2]**

**coef<-as.vector(coef)**

**y2<-sum(coef\*obs3[2,])+e2**

**y2**

### 12. Procédure lorsque la fonction solve() n'est pas utilisable

**obs1<-matrix(rep(0,300),nrow=50)**

**obs1[,1]<-rep(1,50)**

**for(i in 2:6){obs1[(10\*(i-2)+(1:10)),i]<-1}**

**X<-t(obs1)%\*%obs1**

**solve(X)**

**val.critique<-which(svd(X)$d<10^-8)**

**Diag.mod<-diag(1/svd(X)$d)**

**for(i in val.critique){Diag.mod[i,i]<-0}**

**svd(X)$v%\*%Diag.mod%\*% t(svd(X)$u)**

### 13. Exemples graphiques avec R

**obs1<-matrix(c(12, 15, 25, 6),ncol=2)**

**graphics.off()**

**par(mar=c(5, 5, 7, 2))**

**par(yaxs="i")**

**library(gplots)**

**barplot2(obs1, beside=TRUE, col=c("white","black"), ylim=c(0,40), ylab="Effectif", cex.lab=2, cex.axis=1.8)**

**rug(seq(par("yaxp")[1], par("yaxp")[2], length.out=par("yaxp")[3]\*2+1), -0.015, side=2)**

**par(xpd=FALSE)**

**abline(h=0, lwd=1.5)**

**par(xpd=TRUE)**

**text(x=c(2,5), y=-4, c("Sud", "Nord"), srt=45, cex=2)**

**text(x=c(2.5,5.5), y=c(20,10), c("NS", "\*"), font=c(3,1), cex=c(2,3))**

**legend(x=4, y=50, c("Contrôle", "Traitement"), pch=22, pt.bg=c("white","black"), pt.cex=3, bty="o", cex=2,y.intersp=1.5)**

**segments(1.5, 33, 4.5, 33, lwd=2)**

**segments(1.5, 33, 1.5, 31, lwd=2)**

**segments(4.5, 33, 4.5, 31, lwd=2)**

**text(x=3, y=35, c("p < 0.001"), font=3, cex=2)**

**obs1<-matrix(c(5, 15, 25, 16, 26, 6, 28,8,18),ncol=3)**

**graphics.off()**

**par(xaxt="n")**

**par(mar=c(6,5,5,2))**

**y.label<- expression(paste("Effectif (",P >= 3, ")"))**

**barplot(obs1, beside=TRUE, density=c(2, -1, 3), angle=c(45,0,-60), col=c("black","grey","red"), ylim=c(0,35), ylab=y.label, cex.lab=1.8, cex.axis=1.7)**

**classe<-expression(paste(Omega, " et 37°C"), paste(delta," ",Delta), bar(x) == sum(over(hat(x)[i], y^2), i==1, n))**

**par(xpd=TRUE)**

**text(x=c(2.5,6.5,10.5), y=rep(-4.5, 3), classe, srt=30, cex=1.8)**

**effect<-expression(paste(H[2],O[2]), paste(chi^2 != 10^-6, " {çéüse}"), sqrt(x,3))**

**legend(x=2, y=40, effect, density=c(10, -1, 15), angle=c(45,0,-60), fill=c("black","grey","red"), bty="n", cex=1.8,y.intersp=1.5)**

**Heure<-c(11.75,19.25,21.75,24.25,26.75,29.25,31.75,35.25)**

**Malade.Placebo<-c(0.0,0.5,1.2,2.4,3.5,3.7,3.9,4.0)**

**SD.Malade.Placebo<-c(0.0,0.0,0.1,0.2,0.1,0.2,0.1,0.1)**

**Sain.Placebo<-c(0.0,0.6,1.4,2.5,3.6,3.8,3.9,4.0)**

**SD.Sain.Placebo<-c(0.0,0.2,0.3,0.3,0.2,0.2,0.1,0.3)**

**Malade.Traitement<-c(0.0,0.1,0.2,0.3,0.5,0.5,0.7,1.0)**

**SD.Malade.Traitement<-c(0.0,0.1,0.1,0.2,0.3,0.3,0.9,0.7)**

**Sain.Traitement<-c(0.0,0.3,0.6,1.3,2.7,4.2,4.9,5.5)**

**SD.Sain.Traitement<-c(0.0,0.0,0.0,0.0,0.0,0.1,0.1,0.1)**

**obs1<-data.frame(Heure, Malade.Placebo, SD.Malade.Placebo, Sain.Placebo, SD.Sain.Placebo, Malade.Traitement, SD.Malade.Traitement, Sain.Traitement, SD.Sain.Traitement)**

**graphics.off()**

**par(mar=c(5,6,4,2))**

**y.label<-expression(paste("Nombre de cellules (x", 10^6,")"))**

**plot(obs1[,1], obs1[,2], type="l", col="white", ylim=c(0,6), xlab="Temps (heures)", ylab=y.label, bty="l", cex.lab=1.8, cex.axis=1.8)**

**for(i in c(2,4,6,8)){suppressWarnings(arrows(obs1[,1], obs1[,i]-obs1[,i+1], obs1[,1], obs1[,i]+obs1[,i+1], angle=90, code=3, length=0.08, lwd=1))}**

**lines(obs1[,1], obs1[,2], type="b", pch=22, bg="black", lwd=2, cex=2.5)**

**lines(obs1[,1], obs1[,4], type="b", pch=24, bg="black", lwd=2, cex=2.5)**

**lines(obs1[,1], obs1[,6], type="b", pch=22, bg="white", lwd=2, cex=2.5)**

**lines(obs1[,1], obs1[,8], type="b", pch=24, bg="white", lwd=2, cex=2.5)**

**par(xpd=TRUE)**

**legend(x=12.5, y=7, names(obs1)[c(2, 4, 6, 8)], lty=c(1, 1, 1, 1), pch=c(22, 24, 22, 24), pt.cex=c(2.3, 2.3, 2.3, 2.3), pt.bg=c("black", "black", "white", "white"), lwd=c(2, 2, 2, 2), bty="n", cex=1.8,y.intersp=1.6)**

**activite<-c(2, 5, 45, 28, 12, 20, 17, 28, 55, 60, 81, 42, 67, 42, 42, 85)**

**maladie<-c(rep("Hépatite", 8), rep("Cirrhose", 8))**

**operation<-c("Oui", "Oui", rep("Non", 12), "Oui", "Non")**

**obs<-data.frame(activite, maladie, operation)**

**graphics.off()**

**par(mar=c(5, 7, 5, 7), mfrow=c(1, 1), cex.lab=1.7, cex.axis=1.7, bty="u", xaxt="n", xpd=NA, xaxs="i", yaxs="i", lwd=2)**

**stripchart(obs[, 1]~obs[, 2], vertical=TRUE, method="stack", offset=0.5, pch=1, cex=3, ylab="Activité \n enzymatique (%)", ylim=c(-2.2, 102), xlim=c(0.5, 2.5))**

**polygon(x=c(0.5, 2.5, 2.5, 0.5, 0.5), y=c(50, 50, 102, 102, 50), density=-1, col="grey90", lty=0)**

**segments(0.5, 50, 0.5, 102)**

**segments(2.5, 50, 2.5, 102)**

**stripchart(obs[, 1]~obs[, 2], vertical=TRUE, method="stack", offset=0.5, pch=1, cex=3, add=TRUE)**

**obs2<-obs[obs[, 3]=="Oui", ]**

**stripchart(obs2[, 1]~obs2[, 2], vertical=TRUE, method="stack", offset=0.5, pch=16, cex=3, add=TRUE)**

**par(xaxt="s")**

**axis(side=1, at=c(1, 2), labels = levels(obs[, 2]))**

**medianes<-unlist(lapply(split(activite, maladie), median))**

**segments(0.8, medianes[1], 1.2, medianes[1], lwd=5)**

**segments(1.8, medianes[2], 2.2, medianes[2], lwd=5)**

**par(usr=c(par("usr")[1:2], par("usr")[3:4]/100))**

**axis(side=4)**

**mtext("Activité \n enzymatique (proportion)", side = 4, line=5, cex=1.7)**

### 14. Exécution des codes du chapitre 5 depuis un fichier de type "texte" et exportation des résultats

**source("C:/Users/Gael/Desktop/test de Welch.txt", print.eval=TRUE)**

**t.calc<-(m1-m2)/(s.1^2/n1+s.2^2/n2)^0.5**

**print("Valeur de la VT t sur notre échantillon")**

**t.calc**

**cat("\n")**

**chem <- "C:/Users/Gael/Desktop/dossier résultat"**

**dir.create(chem)**

**dput(t.calc, file=paste(chem, "/valeur de tcalc.txt", sep=""))**

**list.files(chem)**

**chem <- "C:/Users/Gael/Desktop/dossier résultat"**

**dir.create(chem)**

**name.file <- "resultats.txt"**

**export.data <- function(data){if(any(grepl(name.file, list.files(chem)))){append.log <- TRUE} else{append.log <- FALSE}**

**if(is.vector(data) & class(data) != "list" & length(data) == 1){cat(data, file= paste(chem, "\\", name.file, sep=""), append= append.log)} else{capture.output(data, file=paste(chem, "\\", name.file, sep=""), append = append.log)}**

**write("\n\n", file= paste(chem, "\\", name.file, sep=""), append= TRUE)}**

**export.data("\n\n########## Résultats de l'analyse ##########")**

**matrice.2<-matrix((1:15)\*100,nrow=3, byrow=TRUE)**

**export.data("matrice.2 :")**

**export.data(matrice.2)**

**export.data("dataframe.1 :")**

**dataframe.1<-data.frame(matrice.2)**

**export.data(dataframe.1)**

### 15. Edition des graphiques du chapitre 5 dans un fichier de type "pdf"

**graphics.off()**

**pdf("C:/Users/Gael/Desktop/Graphiques.pdf")**

**pG1.Fi<- obs3[, 1]/ margin.table(obs3, 1)**

**barplot(pG1.Fi, xlab = paste(labels(dimnames(obs3))[2], " / ", labels(dimnames(obs3))[1]), xaxt = "n", ylab="Proportion", ylim=range(0, max(pG1.Fi)+ 0.15), cex.lab=1.5, cex.axis=1.5, cex=1.5)**

**position.labels <- barplot(pG1.Fi, plot = FALSE)[]**

**axis(side=1, at = position.labels, labels = c(paste(colnames(obs3)[1], " / ", rownames(obs3)[1]), paste(colnames(obs3)[1], " / ", rownames(obs3)[2])), cex.axis=1.5)**

**par(cex.lab=2, xpd=NA, font=2)**

**mosaicplot(t(obs3), main=NULL, cex.axis=1.5)**

**dev.off()**

**if(length(dev.list()) != 0){for(i in length(names(dev.list())) : 1){if(names(dev.list())[i] == "pdf"){dev.off(dev.list()[i])}}}**

### 16. Différences entre les fonctions sort(), rank() et order()

### 17. Précisions sur la médiane et autres quantiles

**x = (1:10)\*10**

**x**

**n = length(x)**

**p = 0.333**

**r = (n-1)\*p + 1**

**r**

**Qp = x[floor(r)]+(r-floor(r))\*(x[floor(r)+1] - x[floor(r)])**

**Qp**

**quantile(x, probs = p)**

**x = (1:10)\*10**

**x**

**p = 0.333**

**technique <- 1:9**

**quantile.res <- NULL**

**for(i in technique){quantile.res <- c(quantile.res, quantile(x, probs = p, type = i))}**

**data.frame(technique, quantile.res)**

### 18. Création de fonctions

**# rm(list = ls())** *# attention, ceci supprime l'ensemble des objets de l'environnement de travail. Effectuer une sauvegarde de cet environnement si besoin avant de lancer l'instruction*

**function(){x <- 4}**

**fonction.1 <- function(){x <- 4}**

**ls()**

**fonction.1**

**fonction.1()**

**ls()**

**x <- 2**

**ls()**

**fonction.1()**

**x**

**fonction.2 <- function(){x <- 4 ; x}**

**fonction.2()**

**fonction.3 <- function(){x <- 4 ; z}**

**fonction.3()**

**z <- 10**

**ls()**

**fonction.3()**

**rm(z)**

**ls()**

**x**

**fonction.4 <- function(){x <<- 4 ; z <<-9}**

**fonction.4()**

**ls()**

**x**

**z**

**rm(x, z)**

**ls()**

**fonction.5 <- function(){x <-1 ; x <<- 4 ; y <- x+2 ; y}**

**fonction.5()**

**x**

**rm(x)**

**fonction.6 <- function(){x <- "test" ; if(is.numeric(x) == FALSE){stop("x n'est pas 'numérique'")} ; y <<- x+5}**

**fonction.6()**

**ls()**

**options(show.error.messages = FALSE)**

**fonction.6()**

**options(show.error.messages = TRUE)**

**fonction.2 <- function(){**

**x <- 4**

**z**

**}**

**fonction.1 <- function()x <- 4**

**fonction.2 <- function(){x <- 4 ; x}**

**fonction.2()**

**fonction.7 <- function(){x <- 4 ; y <- 1 ; x ; y}**

**fonction.7()**

**fonction.8 <- function(){x <- 4 ; y <- 1 ; return(x) ; y}**

**fonction.8()**

**fonction.9 <- function(){x <- 4 ; y <- 1 ; print(x) ; print(y)}**

**fonction.9()**

**z <- 9**

**fonction.10 <- function(){print(z) ; z <- 5 ; print(z)}**

**fonction.10()**

**fonction.11 <- function(){x <- 4 ; y <- 1 ; list(res1 = x, res2 = y)}**

**fonction.11()**

**fonction.11()$res2**

**# rm(list = ls())** *# attention, ceci supprime l'ensemble des objets de l'environnement de travail. Effectuer une sauvegarde de cet environnement si besoin avant de lancer l'instruction*

**fonction.12 <- function(nb, exposant){nb^exposant}**

**fonction.12(2, 3)**

**ls()**

**fonction.12(3, 2)**

**fonction.12(exposant = 3, nb = 2)**

**nb <- 5**

**fonction.12(nb = nb\*2, exposant = nb)**

**rm(nb)**

**fonction.13 <- function(nb, exposant = 2, param1 = TRUE){if(is.numeric(nb) == param1){nb^exposant} else{print("Utiliser un chiffre")}}**

**fonction.13(3)**

**fonction.13(2, 3, FALSE)**

**fonction.13(exposant = 3, 2)**

**fonction.13(1, exposant = 3, nb = 2)**

**fonction.14 <- function(nb, exposant = c(1, 2, 3)){nb^exposant}**

**fonction.14(3)**

**fonction.14(0, nb = 3)**

**fonction.15 <- function(nb, char = c("a", "b", "c")){char <- match.arg(char) ; paste(char, nb, sep=".")}**

**fonction.15(3)**

**fonction.15(char ="b", nb = 3)**

**fonction.15(3, "e")**

**fonction.16 <- function(...){sum(...)}**

**fonction.16(3, 2, 5)**

**fonction.17 <- function(){sum(...)}**

**fonction.17(3, 2, 5)**

**fonction.18 <- function(nb1, nb2, ...){cor(nb1, nb2, ...)}**

**fonction.18(nb1 = 1:5, nb2 = c(1, 5, 1, 3, 40), method="pearson")**

**fonction.18(nb1 = 1:5, nb2 = c(1, 5, 1, 3, 40), method="spearman")**

**fonction.18(nb1 = 1:5, nb2 = c(1, 5, 1, 3, 40), method="Albert.Einstein")**

**fonction.19 <- function(nb1, nb2){cor(nb1, nb2, ...)}**

**fonction.19(nb1 = 1:5, nb2 = c(1, 5, 1, 3, 40), method="pearson")**

**fonction.20 <- function(nb1, nb2, ...){cor(nb1, nb2)}**

**fonction.20(nb1 = 1:5, nb2 = c(1, 5, 1, 3, 40), method="spearman")**

**fonction.21 <- function(nb1, nb2, ...){cor1 <- cor(nb1, nb2, ...) ; sum1 <- sum(nb1, nb2) ; list(cor1, sum1)}**

**fonction.21(1:5, c(1, 5, 1, 3, 40))**

**fonction.22 <- function(nb1, nb2, ...){cor1 <- cor(nb1, nb2, ...) ; sum1 <- sum(...) ; list(cor1, sum1)}**

**fonction.22(1:5, c(1, 5, 1, 3, 40))**

**fonction.23 <- function(nb1, nb2, ...){cor1 <- cor(nb1, nb2) ; sum1 <- sum(...) ; list(cor1, sum1)}**

**fonction.23(1:5, c(1, 5, 1, 3, 40), 5, 4)**

**fonction.24 <- function(nb1, nb2, ...){cor1 <- cor(nb1, nb2, ...) ; sum1 <- sum(nb1, nb2, ...) ; list(cor1, sum1)}**

**fonction.24(1:5, c(1, 5, 1, 3, 40), na.rm=TRUE)**

**fonction.24(1:5, c(1, 5, 1, 3, 40), method="spearman")**

**fonction.25 <- function(nb1, nb2, param1 = TRUE, param2= c("pearson", "kendall", "spearman")){cor1 <- cor(nb1, nb2, method = param2) ; sum1 <- sum(nb1, nb2, na.rm = param1) ; list(cor1, sum1)}**

**fonction.25(1:5, c(1, 5, 1, 3, 40), param1 = FALSE, param2 = "spearman")**

**fonction.26 <- function(nb1, fun1){fun1(nb1)}**

**fonction.26(1:5, sum)**

**fonction.26(1:5, mean)**

**fonction.27 <- function(nb1, fun1, ...){fun1(nb1, ...)}**

**fonction.27(c(1:5, NA), sum, na.rm=TRUE)**

**fonction.28 <- function(nb1, fun1=c("sum", "mean"), ...){fun1 <- match.arg(fun1) ; get(fun1)(nb1, ...)}**

**fonction.28(c(1:5, NA), "sum", na.rm=TRUE)**

**fonction.28(1:5, "var")**

**"%!!%" <- function(nb1, fun1){fun1(nb1)}**

**1:5 %!!% sum**

**"!!" <- function(nb1, fun1){fun1(nb1)}**

**1:5 !! sum**

**"%fun\%" <- function(nb1, fun1){fun1(nb1)}**

**.GlobalEnv**

**is.environment(.GlobalEnv)**

**environmentName(.GlobalEnv)**

**typeof(.GlobalEnv)**

**emptyenv()**

**baseenv()**

*# Pour obtenir exactement les mêmes résultats que ceux affichés, quitter et rouvrir R*

**search()**

**searchpaths()**

**as.environment(1)**

**as.environment(2)**

**as.environment(10)**

**env.sup <- parent.env(.GlobalEnv)**

**env.sup**

**ls(env.sup, all.names = TRUE)**

*# Pour obtenir exactement les mêmes résultats que ceux affichés, quitter et rouvrir R*

**# rm(list = ls( all.names = TRUE))** *# attention, ceci supprime l'ensemble des objets de l'environnement de travail. Effectuer une sauvegarde de cet environnement si besoin avant de lancer l'instruction*

**nb.objets <- length(ls(.GlobalEnv, all.names = TRUE))**

**name <- "R\_GlobalEnv"**

**num <- 1**

**for(i in 1:9){env.sup <- parent.env(as.environment(i)) ; num <- c(num, i+1) ; name <- c(name, environmentName(env.sup)) ; nb.objets <- c(nb.objets, length(ls(env.sup, all.names = TRUE)))}**

**data.frame(nom.env=name, numero.env=num, nombre.objets = nb.objets)**

*# Pour obtenir exactement les mêmes résultats que ceux affichés, quitter et rouvrir R*

**library(gdata)**

**search()**

**parent.env(.GlobalEnv)**

*# Pour obtenir exactement les mêmes résultats que ceux affichés, quitter et rouvrir R*

**library(gdata)**

**library(splines)**

**search()**

*# Pour obtenir exactement les mêmes résultats que ceux affichés, quitter et rouvrir R*

**# rm(list = ls( all.names = TRUE))** *# attention, ceci supprime l'ensemble des objets de l'environnement de travail. Effectuer une sauvegarde de cet environnement si besoin avant de lancer l'instruction*

**env.1 <- new.env()**

**ls()**

**find("env.1")**

**typeof(env.1)**

**search()**

**parent.env(env.1)**

*# Pour obtenir exactement les mêmes résultats que ceux affichés, quitter et rouvrir R*

**# rm(list = ls( all.names = TRUE))** *# attention, ceci supprime l'ensemble des objets de l'environnement de travail. Effectuer une sauvegarde de cet environnement si besoin avant de lancer l'instruction*

**env.1 <- new.env()**

**ls()**

**parent.env(env.1)**

**assign("fonction.29", function(){print(x)}, envir = env.1)**

**fonction.29()**

**ls()**

**ls(envir = env.1)**

**assign("x", 3, envir = env.1)**

**get("fonction.29", envir=env.1)()**

**ls(envir = env.1)**

**rm(x, envir = env.1)**

**x <- 3**

**get("fonction.29", envir=env.1)()**

**ls(envir = env.1)**

**ls()**

**rm(x)**

**assign("x", 3, envir = baseenv())**

**get("fonction.29", envir=env.1)()**

**ls()**

**find("x")**

**x <- "Einstein"**

**get("fonction.29", envir=env.1)()**

**find("x")**

**assign("fonction.30", function(){mean(x)}, envir = env.1)**

**get("fonction.30", envir=env.1)()**

**rm(x)**

**mean <- "Bohr"**

**get("fonction.30", envir=env.1)()**

**find("mean")**

**assign("fonction.31", function(){x <- 10 ; fonction.32 <- function(){2 \* x} ; fonction.32()}, envir = env.1)**

**get("fonction.31", envir=env.1)()**

**environment(t.test)**

**find("mean", numeric = TRUE)**

**apropos("t.test", where = TRUE)**

**x <- 4**

**fonction.33 <- function(nb1, fun1=c("sum", "mean")){fun1 <- match.arg(fun1) ; get(fun1)(nb1) ; mean(x) ; y <- 2 ; ls()}**

**fonction.33(1:5, "sum")**

*# Pour obtenir exactement les mêmes résultats que ceux affichés, quitter et rouvrir R*

**# rm(list = ls( all.names = TRUE))** *# attention, ceci supprime l'ensemble des objets de l'environnement de travail. Effectuer une sauvegarde de cet environnement si besoin avant de lancer l'instruction*

**fonction.34 <- function(){Bob <<- 20}**

**fonction.34()**

**Bob**

**find("Bob")**

**assign("x", 3, envir = baseenv())**

**fonction.35 <- function(){x <<- 10}**

**fonction.35()**

**x**

**x <- "Einstein"**

**assign("x", 3, envir = baseenv())**

**fonction.35()**

**x**

**base::x**

**fonction.36 <- function(){mean <<- 15}**

**fonction.36()**

**fonction.37 <- function(){assign("mean", 15, envir=.GlobalEnv)}**

**fonction.37()**

**mean**

**find("mean")**

**mean(2)**

**fonction.38 <- function(){assign("Albert", 15)}**

**fonction.38()**

**find("Albert")**

**y <- 20**

**fonction.39 <- function(){y <- 10 ; fonction.40 <- function(){y <<- 3} ; fonction.40() ; print(y)}**

**fonction.39()**

**y**

*# Pour obtenir exactement les mêmes résultats que ceux affichés, il est conseillé de quitter et rouvrir R*

**# rm(list = ls( all.names = TRUE))** *# il est préférable d'effectuer cette instruction. Attention, ceci supprime l'ensemble des objets de l'environnement de travail. Effectuer une sauvegarde de cet environnement si besoin avant de lancer l'instruction*

**ls("package:base", all = TRUE)[sapply(sapply(ls("package:base", all = TRUE), get), is.primitive)]**

**class(sum)**

**is.primitive(sum)**

**sum**

**typeof(sum)**

**environment(sum)**

**class(get("<-"))**

**is.primitive(get("<-"))**

**get("<-")**

**typeof(get("<-"))**

**class(sample)**

**is.primitive(sample)**

**sample**

**typeof(sample)**

**formals(sample)**

**ks.test**

*# Pour obtenir exactement les mêmes résultats que ceux affichés dans cette section, il est conseillé de quitter et rouvrir R*

**# rm(list = ls( all.names = TRUE))** *# il est préférable d'effectuer cette instruction. Attention, ceci supprime l'ensemble des objets de l'environnement de travail. Effectuer une sauvegarde de cet environnement si besoin avant de lancer l'instruction*

**t.test**

**methods("t.test")**

**getAnywhere("t.test.default")**

**typeof(show)**

**class(show)**

**mode(show)**

**isS4(show)**

**show**

**showMethods("show")**

**findMethods("show")**

**source("C:\\Users\\Gael\\Desktop\\t.test.R")**

**ls()**

**t.test**

**environment(t.test)**

**# rm(list = ls())** *# Suppression de l'ensemble des objets de l'environnement de travail*

**t.test**

**sum**

**{"sum", do\_summary, 0, 1, -1, {PP\_FUNCALL, PREC\_FN, 0}},**

**mean**

**methods("mean")**

**getAnywhere("mean.default")**

**{"mean", do\_summary, 1, 11, 1, {PP\_FUNCALL, PREC\_FN, 0}},**

**"+"(1, 2)**

**z <- c(10, 20, 30)**

**"["(z, 2)**

**loop <-NULL**

**"for"(i, 1:3, {loop <- c(loop, i)})**

**loop**

### 19. Principaux attributs de R