Programmation avec R – Polycopié d'exercices

Leslie REGAD ; Gaëlle LELANDAIS

leslie.regad@univ-paris-diderot.fr ; gaelle.lelandais@univ-paris-diderot.fr

Ce polycopié rassemble différents exercices vous permettant d'apprendre à utiliser le langage de programmation R. Ces exercices sont de difficultés variables. Il est important de réaliser les exercices les plus simples avant de débuter les plus difficiles. Nous vous conseillons également de lire attentivement le document « bonnes pratiques en R » (disponible sur le plateforme Didel) afin d'écrire un code lisible et d'acquérir de bonnes habitudes de programmation en R.

Remarque : N'hésitez pas à solliciter les enseignants présents dans la salle de cours (si vous rencontrez des difficultés ou que vous souhaitez obtenir des conseils et ainsi découvrir des « trucs et astuces »).

Séance 1 d'exercices

Objectifs : Sauvegarder un ensemble de commandes R dans un fichier texte, les exécuter individuellement ou successivement dans une console R.

Exercice 1

En utilisant un éditeur de texte de votre choix (*kate*, *gedit*, etc.), créer un fichier nommé « *Commandes.R* ». Dans ce fichier, vous écrirez ligne par ligne les commandes R permettant de résoudre les exercices de ce polycopié.

Sauvegarder le fichier *Commandes.R* dans le répertoire de votre choix. Lancer une console R à partir de ce même répertoire. Ecrire les commandes R permettant de calculer les cosinus de 75, 25 et 90 ; imprimer la phrase suivante à l'écran « un bon bioinformaticien ne peut pas travailler sans connaitre le langage R » et la mettre en majuscules ; compter et afficher le nombre de caractères dans cette phrase.

Exécuter les commandes R de votre script ligne par ligne (copier/coller) et successivement (automatiquement).

Commandes R : cos( ) ; print( ) ; toupper( ) ; nchar( ) ; etc.

Remarques : N'oubliez pas qu'il est nécessaire de commenter votre code soigneusement. Il est possible « d'ignorer » certaines lignes en utilisant le symbole « # ». Sauvegarder votre fichier Commandes.R *régulièrement.*

Exercice 2

Effectuer l'opération suivante : 4850 / 26. Afficher le résultat avec seulement 3 décimales. Comment est réalisé l'arrondi ? Trouver plusieurs fonctions R permettant de réaliser une réduction de décimales.

Commandes R : round() ; help( ) ; etc.

Exercice 3

Dans R, la valeur de la variable du nombre π est pré calculée et nommée « *pi* ». Afficher cette valeur à l'écran avec une précision de 3 puis de 5 décimales. Ajouter 12 à la variable *pi*, puis affecter à cette même variable la valeur 9. Imprimer successivement les nouvelles valeurs de *pi* à l'écran.

Remarque : Notez ici qu'une variable interne au logiciel R (ici pi*) peut être modifiée. Il est conseillé de ne pas utiliser les noms des variables internes de R dans un programme que vous écrivez (ceci afin d'éviter des confusions). Les variables internes sont réinitialisées après fermeture et réouverture de la session R.*

Séance 2 d'exercices

Objectifs : Créer et manipuler les objets R (vecteurs, tableaux et listes).

Exercice 4

Créer le vecteur « *vec1* » contenant la suite des entiers de 1 à 12. Ajouter à la fin de ce vecteur les valeurs 16, 17, 18.

Commandes R : « : » ; c( ) ; etc.

Exercice 5

Créer le vecteur « *vec2* » contenant les valeurs suivantes : 0, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5.0. Réaliser cette action de différentes manières.

Remarque : Il est important de savoir créer un vecteur de plusieurs manières, afin d'optimiser votre code et d'éviter d'écrire « à la main » un trop grand nombre de valeurs.

Commandes R : seq( ) ; etc.

Exercice 6

Créer le vecteur « *vec3* » contenant tous les multiples de 2 compris entre 1 et 50.

Créer le vecteur « *vec4* » contenant 3 fois chacun des 10 chiffres (soit 0, 0, 0 ; 1, 1, 1 ; 2, 2, 2 ; 3, 3, 3 ; etc.).

Créer le vecteur « *vec5* » contenant une fois la lettre A, deux fois la lettre B, trois fois la lettre C ... et 26 fois la lettre Z. Quelle est la longueur de cette suite ?

Commandes R : LETTERS ; length( ) ; etc.

Exercice 7

Créer le vecteur « *vec6* » contenant les noms suivants : « individu1 », « individu2 », ... , « individu100 ».

Commandes R : paste( ) ; etc.

Exercice 8

Créer la matrice identité « *matIdentite* » de dimension 10 lignes x 10 colonnes (chiffre 1 sur la diagonale et chiffre 0 ailleurs).

Créer une matrice « *matAleatoire* » contenant des valeurs tirées aléatoirement de dimension 10 lignes x 10 colonnes, dont les éléments suivent une loi normale de moyenne 0 et de variance 5.

Commandes R : matrix( ) ; rnorm() ; etc.

Remarque : Certaines fonctions seront détaillées ultérieurement dans le cours. Consulter l'aide de R pour les utiliser.

Exercice 9

Afficher la date du jour. Sauvegarder le résultat dans une variable « *dateJour* ». Créer les variables « *m1* » et « *m2* » telles que m1 = 3 et m2 = « promotions assistent au cours de R du ».

Combiner les variables *dateJour, m1* et *m2* afin d'afficher à l'écran la phrase suivante « 3 promotions assistent au cours de R qui a lieu le Thu sep 04 2012 ».

Commandes R : date( ) ; diag( ) ; etc.

Exercice 10

Créer le vecteur d'entiers « *vecX* » avec les valeurs 1 à 20. Calculer le carré des 5 premiers éléments de *vecX*. Calculer la somme de *vecX*.

Créer le vecteur "*sinusX*" qui contient les valeurs de sinus du vecteur "*vecX*". Créer le vecteur « *xNeg* » des indices des valeurs négatives de *sinusX*. Donner le nombre de valeurs négatives et remplacer ces valeurs par 0. Créer les vecteurs « *xEven* » avec les éléments d'indice pair de *sinusX* et « *xOdd* » les éléments de *sinusX* d'indices impairs.

Combiner les vecteurs *xOdd* et *xEven* en une unique matrice « *matCombine* » à deux colonnes.

Commandes R : sum( ) ; which( ) ; cbind( ) ; etc.

Exercice 11

Importer dans votre session R le jeu de données « *precip* ». Ce jeu de données est pré-enregistré dans le logiciel R et regroupe des données de précipitations dans différentes villes américaines.

Donner la liste des villes pour lesquelles des mesures sont disponibles, combien y en a-t-il ? Quel est le niveau de précipitation des villes suivantes : Philadelphia, Columbia, Baltimore, Sacramento ?

Commandes R : data( ) ; names( ) ; etc.

Exercice 12

Créer deux vecteurs aléatoires nommés « *x1* » et « *x2* », contenant chacun 100 valeurs aléatoires compatibles 1) avec une distribution de loi normale centrée réduite et 2) avec une distribution de loi uniforme définie sur l'intervalle [0 ; 10].

Créer une matrice « *m1* » qui contient les 10 premières valeurs de *x1* (colonne 1 de *m1*) et les 10 dernières valeurs de *x2* (colonne 2 de *m1*).

Créer une matrice « *m2* » qui contient les 16ème, 51ème, 79ème, 31ème et 27ème valeurs de *x1* (colonne 1 de *m2*) et les 30ème, 70ème, 12ème, 49ème et 45ème de *x2* (colonne 2 de *m2*).

Concaténer à la suite les matrices *m1* et *m2* en une matrice *m12*. Quelles sont les dimensions (nombre de lignes et de colonnes) de *m12* ?

Commandes R : runif( ) ; rbind( ) ; dim( ) ; etc.

Séance 3 d'exercices

Objectifs : Manipuler les boucles et réaliser des tests.

Exercice 13

Créer une boucle qui affiche l'indice « *i* » de l'itération en cours (10 itérations). Calculer la somme cumulée « *sumCumul* » des indices.

Commandes R : for( ) ; etc.

Exercice 14

Créer une matrice « *matAlea* » de dimension 10 lignes x 10 colonnes contenant des valeurs choisies aléatoirement.

La somme de toutes les valeurs de la matrice est-elle supérieure à 15 ? Si oui, afficher le message « la somme est supérieure à 15 » sinon, afficher « la somme n'est pas supérieure à 15 ».

Commandes R : if( ) ; else( ) ; etc.

Exercice 15

Créer le vecteur « *vecPasMultiples* » contenant tous les nombres de 1 à 100 qui ne sont pas des multiples de 5.

Commandes R : « %% » ; etc.

Exercice 16

Créer une matrice « *A* » de dimension 10 lignes x 10 colonnes telle que : A[i, i] = 2 ; A[i, i+1] = -1 ; A[i+1, i] = 1 ; le reste des valeurs = 0.

Commandes R : etc.

Exercice 17

Créer un vecteur « *vecAlea* » de 100 valeurs tirées aléatoirement selon une loi normale de moyenne 4 et d'écart type 5.

Donner les indices des valeurs supérieures à 3, puis récupérer ces valeurs dans le vecteur « *vecSup3* ». Tester si la somme « *valSomme* » de ces valeurs est supérieure à 40, 30 et 20. Afficher un message adéquat.

Commandes R : else if( ) ; etc.

Exercice 18

Créer une matrice « *matrice* » contenant des valeurs tirées aléatoirement de dimension 10 lignes x 10 colonnes et dont les éléments de la colonne j suivent une loi normale de moyenne nulle et de variance j^2 (faire une boucle). Afficher nombre d'éléments positifs et négatifs de la matrice. Remplacer les nombre négatifs par 0.

Afficher les « marges » de *matrice* (sommes des valeurs en lignes et en colonnes).

Commandes R : apply( ) ; etc.

Exercice 19

Importer dans votre session R les données nommées « *WorldPhones* » (pré-existantes dans R). Afficher le contenu de la variable *WorldPhones*, quel est sont type ?

Calculer le nombre total de numéros de téléphone attribués : 1) au cours des différentes années (vecteur « *nbrTelAn* »); 2) pour chaque continent (vecteur « *nbrTelCont* »).

Quelle est le continent qui a le plus / moins de numéros attribués ? Dans combien de continents y a-t-il plus de : 20 000 numéros de téléphone attribués ? 50 000 numéros de téléphone attribués ? 200 000 numéros de téléphone attribués ?

Commandes R : data.class( ) ; etc.

Exercice 20

Simuler 100 lancers d'une pièce de monnaie (résultats possibles « pile »/ « face »). Pour cela, réaliser un tirage (avec remise) dans un ensemble de deux valeurs possibles (« pile » et « face »).

Réaliser la même simulation que précédemment, mais en biaisant la pièce de monnaie (la probabilité d'obtenir « pile » doit seulement être de 0,3). Sauver le résultat dans un vecteur nommé « *vecSampleBiais* ».

Ecrire une commande R qui permet de dénombrer la proportion de « pile » obtenue aux questions précédentes. Ces valeurs sont-elles cohérentes avec les proportions théoriques souhaitées ?

Commandes R : sample( ) ; etc.

Exercice 21

On suppose que la taille et le poids des individus en France se distribuent selon des lois normales de paramètres suivants : 1) la taille des femmes est en moyenne de 165 cm avec un écart-type de 6 cm, 2) la taille des hommes est en moyenne de 175 cm avec un écart-type de 7 cm, 3) le poids des femmes est en moyenne de 60 kg avec un écart-type de 2 kg, 4) le poids des hommes est en moyenne de 75 kg avec un écart-type de 4 kg.

Créer les vecteurs « *tailleF* », « *tailleH* », « *poidsF* », « *poidsH* » contenant la taille et le poids de n = 257 femmes et m = 312 hommes tirés aléatoirement selon les lois ci-dessus (on supposera que poids et taille sont deux variables indépendantes, ce qui est bien entendu faux !). Ne garder que la partie entière des valeurs.

Créer un tableau « *combineData* » à $n + m$ lignes et 3 colonnes telles que : la première colonne contienne la variable taille, la deuxième colonne contienne la variable poids, la troisième colonne soit un facteur indiquant le sexe de l'individu. Nommer les colonnes de ce tableau de manière adéquate.

A partir du tableau *combineData*, déterminer la plus petite taille chez les femmes et la plus petite taille chez les hommes. Réaliser la même chose pour le poids.

Y a-t-il des individus de plus de 1,70 m et de moins de 75 kilos ? Si oui, combien ? Combien sont des femmes, combien sont des hommes ?

Commandes R : « & » ; data.frame() ; etc.

Exercice 22

Créer une séquence d'ADN aléatoire « *S* » de 1000 nucléotides. Combien de fois est présent chaque nucléotide?

Créer le vecteur indicateur « *SA* » tel que SA[i] = 1 si S[i] = « A » et 0 sinon. Réaliser la même opération pour les autres nucléotides.

Commandes R : etc.

Exercice 23

Ecrire une suite de commandes R comparant les mots « parisien» et « aspirine» en tant qu'anagrammes.

Commandes R : unlist( ) ; strsplit( ) ; is.element() ou setequal() ; etc.

Séance 4 d'exercices

Objectifs : Créer et utiliser ses propres fonctions R.

Exercice 24

Créer une fonction nommée « *somme* » qui calcule la somme de deux variables « *x* » et « *y* » passées en arguments.

Commandes R : function( ) ; return( ) ; etc.

Exercice 25

Ecrire une fonction « *calculTarif* » qui prend pour argument une valeur d'âge et qui affiche : « demi-tarif » si l'âge est inférieur à 12 ans, « tarif sénior » si l'âge est supérieur à 60 ans et « plein tarif » sinon.

Tester votre fonction pour des personnes de 5, 65, 85, 41, 23, 47 ans.

Commandes R : etc.

Exercice 26

Ecrire une fonction « *resMention* » qui prend pour argument une note (de 0 à 20) et qui affiche la mention correspondante. Faite en sorte que votre fonction affiche un message d'erreur approprié si la note n'est pas correcte (par exemple une valeur négative ou supérieure à 20).

Commandes R : « || » ; etc.

Exercice 27

La relation qui relie la variable *Y* à la variable *X* est la suivante : $Y = \frac{4X^3}{18} + \frac{8.34X}{5}$

Créer une fonction « *formule* » qui prend en argument une valeur de *X* et qui donne en sortie la valeur de *Y* associée.

Calculer le vecteur « *vecY* » des valeurs de *Y* associées aux valeurs de *X* = 15, 89, 56, 78, 152, 66, 48, 77, 2, 96.

Commandes R : « NULL » ; etc.

Exercice 28

Ecrire une fonction qui prend en entrée un entier *N* positif et qui retourne VRAI si ce nombre est premier, FAUX sinon.

Trouver tous les nombres premiers compris entre 1 et 100. Les conserver dans le vecteur « *vecNbrPrem* »

Commandes R : etc.

Exercice 29

Créer une fonction qui identifie les mots ou phrases palindromiques (mots ou phrases qui se lisent dans les deux sens, par exemple RADAR). Cette fonction prendra comme argument le mot « *mot* » et retournera les phrases : « mot est un palindrome » si le *mot* = palindrome et « mot n'est pas un palindrome », sinon.

Appliquer votre fonction sur les mots « radar », « hannah », « sept », « kayak », « la mariée ira mal », « capitaine », « engage le jeu que je le gagne ».

Commandes R : etc.

Exercice 30

Générer aléatoirement 3 séquences d'ARN de 50, 60 et 79 nucléotides. Créer une fonction « *compteNucleo( )* » qui permet de déterminer le nombre de fois où chaque nucléotides est présent dans la séquence et qui retourne celui qui est le plus fréquent ainsi que le nombre d'occurrences de ce nucléotide dans la séquence.

Utiliser cette fonction pour déterminer le nucléotide le plus fréquent pour chacune de vos trois séquences.

Commandes R : list( ) ; etc.

Exercice 31

Pour estimer le poids moléculaire (PM) d'une protéine à partir de son nombre de résidus (t) la formule suivante est utilisée : $PM = (128 \times t) - \left[(t-1) \times 18\right]$.

Créer une fonction qui prend en entrée le nombre de résidus de la protéine et qui renvoie la valeur estimée de son poids moléculaire.

Estimer le poids moléculaire d'une protéine contenant 25, 50, 178, 89, 345, 879 résidus.

Commandes R : etc.

Exercice 32

La formule du calcul de l'ICM (Indice de Masse Corporelle) est la suivante : $ICM = \frac{poids(kg)}{taille(m)^2}$. L'IMC est un moyen d'évaluer les risques liés à un surpoids chez l'adulte (voir tableau ci-dessous).

Créer une fonction qui prend en entrée le poids et la taille d'une personne et qui donne en sortie l'ICM de la personne.

Calculer l'ICM pour une personne : mesurant 164 cm et pesant 64 kg, mesurant 161 cm et pesant 56 kg, mesurant 1,72 m et pesant 102 kg, mesurant 1,65 m et pesant 51 kg.

Créer une seconde fonction qui prend en arguments le poids et la taille d'une personne et qui donne en sortie sa classification sur son poids. Quelle est la classification des quatre personnes citées précédemment ?

ICM (kg/m2)	Classification	Risque
<18.5	Poids insuffisant	Accru
18.5 à 24.9	Poids normal	Moindre
25 à 29.9	Surpoids ou pré-obésité	Accru
> 30	Obésité	Accru voir très élevé

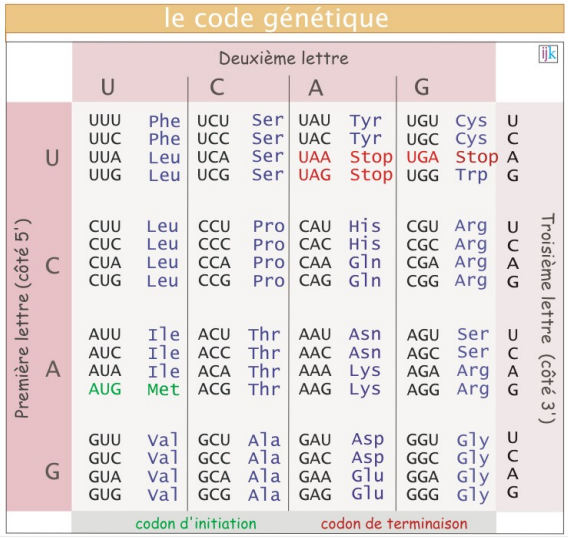
Commandes R : etc.

Exercice 33

Créer une fonction qui prend argument d'entrée une séquence d'ARNm et qui renvoie en sortie la séquence primaire de la protéine correspondante (voir code génétique tableau ci-dessous). Traduire les séquences d'ARNm suivantes :

ARNm = AUGUCUAGGCUGCACUCAGUUAGUAGGUAG

ARNm = AUGCUUACUAACAGGGGUGACUAUCCCUAAAGUGGC



Commandes R : substring() ; etc.

Exercice 34

Créer une fonction qui permet de convertir les séquences protéiques (code 3 lettres) en séquences protéiques (code 1 lettre). Le tableau ci-dessous donne les correspondances entre le code 1 et 3 lettres (fichier de données « *ConversionCode_1_3.dat* »).

Convertir les séquences protéiques suivantes : SER-CYS-THR-ARG-SER-ILE-PRO-PRO-GLN-CYS ; ILE-VAL-VAL-SER-SER-ALA-SER ; SER-SER-ASN-ASP-ASN-ILE-GLU-LEU-VAL-PHE-MET-ILE

Acides aminés	Code 3 lettres	Code 1 lettre
Alanine	Ala	A
Arginine	Arg	R
Acide Aspartique	Asp	D
Asparagine	Asn	N
Cystéine	Cys	C
Glutamine	Gln	Q
Acide Glutamique	Glu	E
Glycine	Gly	G

Acides aminés	Code 3 lettres	Code 1 lettre
Leucine	Leu	L
Lysine	Lys	K
Méthionine	Met	M
Phénylalanine	Phe	F
Proline	Pro	P
Sérine	Ser	S
Thréonine	Thr	T
Tryptophane	Trp	W

Histidine	His	H
Isoleucine	Ile	I

Tyrosine	Tyr	Y
Valine	Val	V

Commandes R : etc.

Exercice 35

Créer une fonction « *compSeq* » qui prend en entrée deux séquences protéiques et qui retourne un vecteur des positions mutées. Comparer les séquences suivantes :

SER-CYS-THR-ARG-SER-ILE-PRO-PRO-GLN-CYS et SER-CYS-THR-THR-SER-ILE-PRO-PRO-GLN-MET

SER-SER-ASN-ASP-ASN-LEU-GLU-ALA-VAL-PHE-MET-ILE et SER-SER-ASN-ASP-ASN-ILE-GLU-LEU-VAL-PHE-MET

Commandes R : etc.

Séance d'exercices 5

Objectifs : Comprendre et utiliser les fonctions statistiques de R.

Exercice 36

Charger le jeu de données « *airquality* ». Quelle est la classe de l'objet R obtenu ? Quelles informations composent cet objet ?

Donner le nombre de valeurs manquantes (indiquées « NA ») pour la concentration d'ozone. Calculer la concentration d'ozone moyenne et sa variance pour les mois de mai, juillet et septembre.

Commandes R : « $ » ; is.na( ) etc.

Exercice 37

Tracer la fonction sinus entre $-\pi$ et $+\pi$. Exporter le graphique au graphique pdf.

Remarque : N'oubliez pas que tout graphique comporte un titre et des axes nommés.

Commandes R : pdf( ) ; dev.off( ) ; etc.

Exercice 38

Superposer sur le même graphique les fonctions sin(x) et cos(x) pour x défini sur $[-\pi, \pi]$.

Commandes R : par(new = T) ; lines().

Exercice 39

Créer un vecteur *N* dont les 1000 éléments suivent une loi normale centrée réduite. Calculer la moyenne et la variance. Représenter l'histogramme des éléments de ce vecteur et superposer sur ce graphique la fonction de densité de la loi normale centrée réduite.

Commandes R : par( ) ; etc.

Exercice 40

Générer aléatoirement une séquence de 100 nucléotides. Sauvegarder cette séquence dans un fichier « *test.out* ».

Créer une matrice de 5 lignes et 5 colonnes. Sauvegarder cette matrice dans un fichier texte « *test2.out* ». Ajouter des arguments à la commande précédente pour retirer les noms des lignes et des colonnes, ainsi que les guillemets.

Commandes R : write.table( ) ; etc.

Exercice 41

A l'aide du logiciel Excel (ou de sont équivalent OpenOffice), créer un tableau de 5 colonnes et 6 lignes. Donner des noms aux colonnes (ex : « colonne 1 », « colonne 2 », …, « colonne 5 ») et aux lignes (ex : « ligne 1 », « ligne 2 », …, « ligne 6 »). Remplir les cases du tableau comme vous le souhaitez.

Sauvegarder ce tableau sous la forme d'un fichier texte (ou fichier CSV), en utilisant des tabulations comme caractères de séparation. Importer le fichier texte obtenu sous R. Afficher le vecteur des noms de lignes. Sauvegarder ce vecteur dans un fichier texte. Enfin, ouvrir ce dernier fichier texte avec Excel.

Commandes R : etc.

Exercice 42

A partir des données *WorldPhones* (exercice 19), partitionner votre feuille graphique en 2 et représenter : (à gauche) le nombre total de numéros de téléphone attribués au cours des différentes années ; (à droite) le nombre moyen de numéros de téléphone attribués au cours des différentes années.

Représenter pour chaque année le nombre total de numéros attribués pour les différentes villes.

Commandes R : barplot( ) ; etc.

Exercice 43

Tirer aléatoirement un ensemble de 100 nombres compatibles avec une distribution normale de moyenne 10 et de variance 25. Tracer la représentation histogramme des valeurs obtenues. Changer les nombre de « barres » de l'histogramme : 5, 50 et 100.

Commandes R : etc.

Exercice 44

Représenter trois points en positions (1, 2), (2, 1) et (3,2). En utilisant l'argument « cex », augmenter la taille des points. En utilisant l'argument « pch », changer la forme des points. En utilisant les arguments « xlim » et « ylim », fixer les échelles des axes. En utilisant l'argument « col », changer la couleur des points. En utilisant l'argument « main », donner un titre au graphique. En utilisant les arguments « xlab » et « ylab », donner des noms aux axes. Sauvegarder le résultat sous la forme d'un document PDF et d'une image JPEG.

Commandes R : jpeg( ) ; dev.off( ) ; etc.

Séance 6 d'exercices : Pour aller plus loin...

Remarque : Ces exercices sont d'un niveau avancé. Ils sont facultatifs. Il est préférable de terminer l'ensemble des exercices précédents avant de travailler à la résolution des exercices de cette section.

Exercice 45

Un sondage est réalisé auprès de 100 individus pour savoir où va leur préférence parmi un panel représentatifs de marques de bière. Les résultats obtenus se trouvent dans le fichier « *bieres.csv* » (disponible sur Didel).

Lire le fichier de données sous forme de « *data.frame* ». Combien de marques sont considérées ? Quelles sont-elles ?

Compter les occurrences de chacune des marques de bières. Les représenter sous la forme de graphe en barres. Représenter cette distribution sous forme de camembert en choisissant les couleurs vous même. Utiliser une seule fenêtre graphique pour les deux figures.

Commandes R : table( ) ; barplot( ) ; pie( ) ; par( ) etc.

Exercice 46

Ouvrir le fichier « *note2010.csv* » (disponible sur Didel) qui contient des notes d'étudiants obtenues au cours de l'année 2010. Quel est le type des variables des colonnes 2 et 4 ?

Sélectionner uniquement les noms des étudiants (colonne 2) et les notes des exercices (colonnes 4-9). Sauvegarder la data.frame obtenue dans le fichier « *notes_ex_2010.dat* » (disponible dur Didel).

Représenter l'histogramme des notes. Calculer la moyenne, la médiane et l'écart type des notes. Combien d'étudiants ont une note en dessous de la moyenne ?

En utilisant la fonction que vous avez écrite précédemment, déterminer la mention de chaque étudiant à ce module.

Commandes R : etc.

Exercice 47

Proposez une procédure permettant d'illustrer le théorème central limite (toute somme de n variables aléatoires indépendantes et identiquement distribuées tend vers une certaine variable aléatoire dont la loi converge vers la loi normale quand n croit indéfiniment).

Commandes R : etc.

Exercice 48

Les données « *old faithful geyser* » ont été collectées dans le cadre d'une étude du temps d'attente entre deux éruptions et la durée des éruptions au sein du parc National de Yellowstone (USA). Ce jeu de données est disponible sous R et est nommé « *faithful* » (package *datasets*). Le seuil critique d'attente au delà duquel la probabilité que la prochaine éruption soit longue et forte est fixé à 63.

Télécharger et visualiser le jeu de données (fonction « *plot( )* »). Construire un histogramme de la durée d'éruption. Représenter l'histogramme en terme de fréquence plutôt qu'en termes d'effectifs (axe Y). Ajouter un titre, nommer les axes et colorer les barres de l'histogramme en vert et les traits de l'histogramme en rouge. Augmenter la taille du pas de l'histogramme à 20 et ajuster une loi normale sur cet histogramme. Que remarquez-vous ?

Partitionner votre graphique en deux dans le sens horizontal. Dans le premier cadre de la partition, tracer les boîtes à moustache (*boxplot*) associées aux durées de l'éruption conditionnellement aux temps d'attente entre deux éruptions. Dans le second cadre de la partition, utiliser le seuil défini précédemment pour construire deux boîtes à moustache associées aux durées d'éruption conditionnellement aux temps d'attente entre 2 éruptions. L'une codant les éruptions courtes et l'autre les éruptions longues.

Commandes R : etc.

Exercice 49

Deux fragments protéiques de même séquence en acides aminés peuvent avoir des géométries différentes. A partir d'une banque de structures protéiques, 36 motifs de 7 acides aminés ont été extraits. Ces motifs sont nommés par des mots de 4 lettres. Un motif correspond à un ensemble de fragments de 7 acides aminés présentant des séquences en acides aminés identiques. Le but de ce travail est d'identifier les motifs les plus similaires en termes de géométrie. Pour cela, le fichier « *distance_fragments.dat* » (disponible sur Didel) qui contient les distances géométriques 2 à 2 entre les différents motifs. Plus la distance géométrique entre deux motifs est petite, plus ces deux motifs ont une géométrie similaire. Cette distance est aussi calculée entre les fragments d'un même motif, nommée distance intra, permettant ainsi de quantifier la variabilité structurale d'un motif donné.

Importer le fichier *distance_fragments .dat ».* Formater les données de ce fichier en une matrice 36 x 36 avec les distances géométriques entre 2 fragments. Les noms de lignes et de colonnes correspondent aux noms des fragments.

BBQK	BBQP	BCDS	…
BBQK	0.53	0.84	3.27
BBQP	0.84	0.64	2 .98
BCDS	3.27	2.98	0.41

Rechercher les motifs qui sont peu variables en termes de géométrie, c'est-à-dire qu'il présente une distance intra < 0.5A et ceux qui sont fortement variables en termes de géométrie (distance intra >= 0.5A). Pour chaque motif calculer la distance moyenne entre motifs et sa variance. Combien de fragments ont au moins une distance inter inférieure à leur distance intra ? Représenter sur le même graphique la distance intra et distance moyenne de chaque fragment. Représenter l'image de la matrice de distance. Extraire pour chaque motif, le motif avec la géométrie la plus similaire (distance la plus petite). Exporter cet objet dans un fichier nommé « *Res.out* ».

On voudrait savoir si certains motifs peuvent être regroupés en termes de structures. Pour cela, réaliser une classification hiérarchique des 36 motifs à l'aide de la matrice de

distance : Remplacer la distance intra par 0. Transformer cette nouvelle matrice en objet de type « *dist* » et réaliser une classification hiérarchique. Combien de groupe faut-il faire ?

Commandes R : etc.

Exercice 50

Un ami vous propose le jeu suivant. On lance un dé. Si le résultat est 5 ou 6, on gagne 3 €, si le résultat est 4 on gagne 1 € et si c'est 3 ou moins on perd 2.5€. Avant d'accepter la partie, vous essayez de simuler ce jeu, pour voir si vous avez des chances de vous enrichir. Conclusion ?

Commandes R : etc.