Utilisation du logiciel R pour le traitement statistique des données biologiques récoltées sur les pâquerettes (1ère partie)

L'objectif de ces TP est de réaliser, à l'aide d'un logiciel de traitement de données, les analyses statistiques abordées en TD de Traitement des Données (S3) afin de répondre à des questions relatives aux données récoltées dans l'UE Démarche scientifique appliquée à l'écologie évolutive. ***N'oubliez pas d'apporter votre polycopié de cours du S3 ainsi qu'une clef USB à chaque TP.***

Dans quelques semaines, vous allez mesurer tout un ensemble de caractéristiques (définies lors du TD1) sur des pâquerettes que vous allez échantillonner dans plusieurs sites localisés dans Toulouse et ses alentours. Un travail similaire a été effectué durant l'été 2021 par une étudiante de Licence ainsi que par les étudiants de L2 BBE en mars 2023. Nous utiliserons les données récoltées en 2021 et 2023 pour les quatre premiers TP de biostatistiques.

Démarrage : connexion aux ordinateurs et récupération des données

⇨ Connectez-vous sur l'ordinateur : démarrez l'ordinateur sous Windows, puis renseignez votre identifiant UT3 (ex : brd2926a) ainsi que votre mot de passe.

⇨ Une fois connecté, commencez par créer un dossier nommé 'TP_biostats' sur votre clef USB.

⇨ Les fichiers utilisés sont disponibles sur Moodle (section TP biostats). Le répertoire compressé (.zip) nommé '*2024-Documents-TPbiostats.zip*' contient les fichiers '*paquerettes.ods*' et '*data.csv*', un script R '*script_etudiants.R*' à compléter, ainsi que la version PDF de ce polycopié.

⇨ Enregistrez puis décompressez ce fichier .zip dans le répertoire 'TP_biostats' que vous venez de créer sur votre clef USB, sans modifier le nom ni le format des fichiers contenus et sans chercher à les ouvrir (pour l'instant). Pour cela, téléchargez le fichier sur Moodle, choisissez "enregistrer le fichier sur" et sélectionnez le répertoire 'TP_biostats' créé sur la clef USB. Ouvrez ensuite ce répertoire, faites un clic droit sur le fichier .zip, sélectionnez le logiciel 7Z (archivage) puis "extraire ici". Tous les fichiers contenus dans le répertoire compressé sont normalement visibles dans le répertoire 'TP_biostats' sur la clef USB.

Prise en main des données

⇨ Ouvrez le fichier '*paquerettes.ods*' que vous venez d'enregistrer, en utilisant le logiciel Classeur de Libre Office (équivalent de Excel mais libre et gratuit).

Le tableau contient ___ lignes et ___ colonnes
(ligne contenant les noms des colonnes non comprise)

Q. Que représentent les lignes du tableau ?

Les colonnes fournissent les informations suivantes :

- *Colonne A* : Année d'échantillonnage
- *Colonne B* : Groupe TP qui a échantillonné la pâquerette
- *Colonne C* : Identifiant individuel de chaque pâquerette
- *Colonne D* : Milieu échantillonné

- *Colonne E* : Code du site échantillonné
- *Colonne F* : Pourcentage de surfaces végétales dans un rayon de 500 m autour du site échantillonné
- *Colonne G* : Date d'échantillonnage
- *Colonne H* : Date de dissection
- *Colonne I* : Nombre d'inflorescences
- *Colonne J* : Nombre de feuilles
- *Colonne K* : Hauteur de la tige (en mm)
- *Colonne L* : Diamètre du capitule (en mm)
- *Colonne M* : Diamètre du disque central (en mm)
- *Colonne N* : Degré de coloration par les anthocyanes de la face supérieure
- *Colonne O* : Degré de coloration par les anthocyanes de la face inférieure
- *Colonne P* : Nombre de fleurs ligulées (blanches)
- *Colonne Q* : Nombre de fleurs tubulées (jaunes)
- *Colonne R* : Classe d'abondance en individus autour de la pâquerette échantillonnée
- *Colonne S* : Classe d'agrégation des individus autour de la pâquerette échantillonnée

Source : Wikipedia (janvier 2024)

Q. Complétez le tableau suivant et cochez les cases correspondantes :

Nom de la variable	**Choisi a priori / observé a posteriori**	**Type**	**Nombre de modalités si qualitative** **Unités si quantitative**
	□ choisi a priori par l'expérimentateur □ observé a posteriori	□ qualitative □ quantitative	
	□ choisi a priori par l'expérimentateur □ observé a posteriori	□ qualitative □ quantitative	
	□ choisi a priori par l'expérimentateur □ observé a posteriori	□ qualitative □ quantitative	
	□ choisi a priori par l'expérimentateur □ observé a posteriori	□ qualitative □ quantitative	
	□ choisi a priori par l'expérimentateur □ observé a posteriori	□ qualitative □ quantitative	
	□ choisi a priori par l'expérimentateur □ observé a posteriori	□ qualitative □ quantitative	
	□ choisi a priori par l'expérimentateur □ observé a posteriori	□ qualitative □ quantitative	
	□ choisi a priori par l'expérimentateur □ observé a posteriori	□ qualitative □ quantitative	
	□ choisi a priori par l'expérimentateur □ observé a posteriori	□ qualitative □ quantitative	
	□ choisi a priori par l'expérimentateur □ observé a posteriori	□ qualitative □ quantitative	
	□ choisi a priori par l'expérimentateur □ observé a posteriori	□ qualitative □ quantitative	
	□ choisi a priori par l'expérimentateur □ observé a posteriori	□ qualitative □ quantitative	
	□ choisi a priori par l'expérimentateur □ observé a posteriori	□ qualitative □ quantitative	
	□ choisi a priori par l'expérimentateur □ observé a posteriori	□ qualitative □ quantitative	
	□ choisi a priori par l'expérimentateur □ observé a posteriori	□ qualitative □ quantitative	
	□ choisi a priori par l'expérimentateur □ observé a posteriori	□ qualitative □ quantitative	
	□ choisi a priori par l'expérimentateur □ observé a posteriori	□ qualitative □ quantitative	
	□ choisi a priori par l'expérimentateur □ observé a posteriori	□ qualitative □ quantitative	
	□ choisi a priori par l'expérimentateur □ observé a posteriori	□ qualitative □ quantitative	

Q. En vous basant sur les hypothèses relatives aux adaptations des pâquerettes élaborées lors du TD1, quelles questions biologiques pourriez-vous formuler à partir de ces données ? Pour chaque question, formulez explicitement la réponse attendue.

1.
2.
3.
4.
5.

Q. Pour chacune des questions :

a. Quelle(s) est(sont) la (les) variable(s) en jeu ?
b. Quel graphique serait le plus pertinent pour illustrer la question ?
c. Laquelle (lesquelles) des analyses statistiques vues au S3 (rappel ci-dessous) pourriez-vous mettre en œuvre pour répondre à la question ?

Comparaison de moyennes de 2 échantillons
Comparaison de moyennes de > 2 échantillons
Comparaison des variances de 2 échantillons
Régression linéaire simple
Corrélation
Chi2 d'indépendance

Le logiciel R et l'interface RStudio

R est un langage de programmation utilisé pour le traitement de données et l'analyse statistique. C'est un logiciel gratuit du type « open source » dont certaines fonctionnalités sont disponibles dans des bibliothèques (*library* en anglais). R fonctionne en mode lignes de commande, mais des interfaces permettent une utilisation plus conviviale. C'est le cas de l'interface RStudio.
R et RStudio sont installés sur les ordinateurs des salles informatiques, mais vous devez les installer sur votre ordinateur personnel (voir encadré page suivante).

Description de l'interface RStudio

⇨ Ouvrez RStudio.

Cette interface est composée de trois ou quatre fenêtres selon la configuration de RStudio sur votre ordinateur :

- *Fenêtre de script* (en haut à gauche mais il est possible que cette fenêtre ne soit pas active lors de la première ouverture de RStudio) : elle contient les scripts de commandes R. Nous reviendrons sur cette notion de script de commandes juste après.
- *Console* (en bas à gauche ou à gauche selon la présence ou non de la fenêtre d'édition) : il s'agit du logiciel R dans lequel toutes les commandes du script sont envoyées pour être exécutées.

- *Fenêtre Environment / History / Connections / Tutorial* (en haut à droite) : elle contient (entre autres) les objets en mémoire ainsi que l'historique des commandes exécutées.
- *Fenêtre Files / Plots / Packages / Help / Viewer / Presentation* (en bas à droite) : nous utiliserons l'onglet Plots qui contient les graphiques tracés, l'onglet Packages qui montre les bibliothèques installées et actuellement chargées et l'onglet Help qui contient la documentation d'aide sur les fonctions et bibliothèques.

Installation de R et de RStudio sur votre ordinateur personnel

Il faut commencer par installer le logiciel R **puis** l'interface RStudio.

1) Rendez vous à l'adresse suivante : http://www.r-project.org/
2) Cliquez sur « CRAN » (à gauche de l'écran, dans la rubrique « Download »). Vous êtes alors dirigé sur une page listant les différents sites « miroir ».
3) Choisissez un site « miroir » (n'importe lequel). Une nouvelle page s'ouvre.
4) Sélectionnez le système d'exploitation de votre ordinateur dans la rubrique « Download and install R » (3 possibilités : Linux, MacOS X, Windows). Une nouvelle page s'ouvre.

Si vous avez choisi Windows comme système d'exploitation :

5) Cliquez sur « base ». Une nouvelle page s'ouvre.
6) Téléchargez le fichier d'installation du logiciel R (fichier .exe) en cliquant sur « Download R-4.3.2 for Windows » (ou une version ultérieure si vous installez R dans quelques mois). NB : il faut se souvenir du répertoire dans lequel a été téléchargé le fichier .exe !
7) Installez le logiciel « classiquement » en suivant les instructions données par votre système d'exploitation. R est maintenant installé sur votre ordinateur.
8) Allez maintenant sur le site suivant pour télécharger RStudio : https://posit.co/download/rstudio-desktop/
9) Passez l'étape 1 puisque vous avez déjà installé R.
10) A l'étape 2, si le système d'exploitation de votre ordinateur est Windows, cliquez sur « Download RStudio Desktop for Windows ». Sinon, parcourez la liste des différentes versions disponibles et choisissez celle correspondant au système d'exploitation de votre ordinateur.
11) Téléchargez et enregistrez le fichier d'installation de RStudio (fichier .exe).
12) Installez RStudio « classiquement » en suivant les instructions données par votre système d'exploitation. RStudio est maintenant installé sur votre ordinateur.
13) Ouvrez l'interface RStudio (icône sur le bureau ou dans la liste des programmes installés).
14) Bravo, vous êtes (normalement) prêt à utiliser R via l'interface RStudio !!!

Création et exécution d'un script de commandes

Pour simplifier le dialogue avec R, nous allons écrire les commandes dans un script. Ces commandes seront ensuite lues et exécutées par R qui affichera les résultats numériques dans la console (fenêtre en bas à gauche) et, s'il y a lieu, les graphiques dans la fenêtre en bas à droite. Ce système est très pratique car il permet de :

- sauvegarder toutes les lignes de commande, le travail réalisé et ainsi refaire les analyses facilement. *C'est votre "cahier de TP", vous devez y sauvegarder votre travail !*
- annoter ce script (symbole # devant le commentaire pour préciser à R que ce n'est pas une commande à exécuter)
- découper les commandes en sections comportant des titres (exemple : `# Titre 1 -----`) ; ces sections peuvent éventuellement être masquées (flèche).

⇨ Ouvrez le script '*script_etudiants.R*' en utilisant le menu File/Open File… puis en sélectionnant le script enregistré sur votre clef USB au début du TP.
NB : il est aussi possible de créer un nouveau script vide en cliquant sur le bouton en haut à gauche (sous File, page blanche avec le symbole + en vert) puis en choisissant « R script ».

Une fenêtre s'ouvre en haut à gauche : il s'agit d'un script de commandes R que vous allez devoir compléter dans la suite des TP. C'est dans cette fenêtre que toutes les commandes permettant d'importer les données et de les analyser seront écrites, puis envoyées vers la console R où elles seront exécutées.

⇨ Pensez à enregistrer régulièrement ce script au fur et à mesure des modifications apportées en cliquant sur l'icône représentant une disquette dans la fenêtre de script.

Nous allons maintenant apprendre à exécuter une commande et obtenir son résultat à partir de ce script. Pour démarrer simplement, nous allons utiliser R comme une calculatrice.

L'opération `3 + 4` est la première commande de votre script. Tout ce qui figure avant (en vert après le symbole `#`) correspond à des commentaires qui ne sont pas des commandes exécutables par R.

⇨ Exécutez cette commande en positionnant le curseur de la souris sur cette ligne puis en cliquant sur le bouton « Run » (situé en haut à droite de la fenêtre de script)[1]. Que se passe-t-il ?

⇨ Tapez une autre opération de votre choix et affichez son résultat.

Les objets dans R

Le résultat de l'opération précédente peut être stocké dans un objet. Le symbole `=` (équivalent au symbole `<-`) permet d'assigner un contenu dans un objet dont le nom est défini par l'utilisateur. Ce nom doit débuter par une lettre, la suite peut comporter des lettres (majuscules ou minuscules), des chiffres, des points et des tirets, mais il ne doit pas y avoir d'espace dans le nom.

⇨ Exécutez la commande `objet1 = 3 + 4` (avec ou sans espace) de votre script. Que se passe-t-il ?

Q. Que doit-on faire pour afficher le contenu de l'objet nommé `objet1` dans la console ?

Q. Tapez maintenant l'opération arithmétique de votre choix permettant d'obtenir la valeur 10 comme résultat à partir d'`objet1`. Il y a plein de solutions possibles, plus ou moins complexes ! Stockez ce résultat dans un objet nommé `objet2`.

[1]Pour exécuter la commande, on peut aussi utiliser le raccourci clavier « Ctrl+Entrée ». Pour exécuter plusieurs lignes de script consécutives, il suffit de toutes les sélectionner puis de cliquer sur « Run ».

Importation d'un tableau de données dans R (*dataframe*)

Après cette prise en main des objets, nous allons importer un tableau de données dans R (objet de type *dataframe*). Le tableau doit répondre à un certain nombre de critères : il ne doit y avoir ni accent, ni symbole, ni espace, les noms des colonnes doivent être simples et les éventuelles valeurs manquantes doivent être remplacées par des **NA** (pour *Non Available*, c'est-à-dire des données non renseignées). Le format préférentiel pour l'importation d'un tableau de données dans R est « .csv » (ou « .txt »).

Le tableau '*data.csv'* que vous avez téléchargé sur Moodle au début de la séance répond à tous ces critères. N'ouvrez surtout pas ce fichier ailleurs que dans RStudio ! Nous allons voir comment ci-dessous.

⇨ Dans l'onglet *Environment* de la fenêtre en haut à droite, cliquez sur « Import Dataset » puis choisissez « From Text (base)… ». Sélectionnez le fichier '*data.csv'* enregistré sur votre clef USB.

⇨ Dans la fenêtre qui s'ouvre, modifiez les champs suivants :
- Name : remplacez *data* par *data_paq*.
- Heading : cochez la valeur *Yes*.
- Separator : sélectionnez *Semicolon* (point-virgule en anglais).
- Decimal : sélectionnez *Period* (point en anglais).
- na.strings : assurez-vous que la valeur NA est bien indiquée.
- Strings as factors : cochez cette case.

⇨ Cliquez sur Import. Que se passe-t-il ?

Q. Expliquez précisément le rôle de chacun des champs complétés ci-dessus.

Le principe des fonctions dans R

Sans le savoir, vous venez d'utiliser deux fonctions du logiciel R ! R est en effet basé sur des fonctions prédéfinies correspondant chacune à une tâche ou une analyse bien précise. Chaque fonction porte un nom et est suivie de parenthèses entre lesquelles des arguments sont indiqués. Pour réaliser vos analyses, il suffira donc de connaître le nom de quelques fonctions basiques[2].

[2] La liste complète des fonctions (et de leurs arguments) utilisées au cours des séances de TP est fournie à la fin de ce polycopié.

Q. Quels sont les noms des deux fonctions qui ont été utilisées précédemment ?

⇨ Copiez les deux lignes de commande apparues dans la console lorsque vous avez importé le jeu de données (sans le symbole >) et collez-les dans votre script à la suite des commandes déjà tapées. Exécutez-les.

Q. Quel est l'intérêt de cette manipulation pour les prochaines séances de TP ?

Description du tableau de données

Nous allons maintenant décrire quantitativement le tableau de données.

⇨ Exécutez la commande `summary(data_paq)`. Que se passe-t-il ? Pour chacune des variables, expliquez à quoi correspond chacun des termes affichés.

Enregistrement du script de commandes

Puisque vous allez travailler sur ces données au cours des prochaines séances, il est recommandé de garder une trace des commandes réalisées.

⇨ Pour cela, il vous suffit de sauvegarder le script sur votre clef USB en cliquant sur l'icône représentant une disquette dans la fenêtre de script.

⇨ Vous pouvez ensuite fermer RStudio. Répondez « Don't save » à la question « Save workspace image to ~/.RData ».

Réouverture de RStudio

Lors de la prochaine séance de TP, il sera nécessaire de repartir du script de commandes qui a été complété et sauvegardé aujourd'hui afin de ne pas avoir à retaper toutes les lignes de commande.

⇨ Commencez par ouvrir RStudio. Selon la configuration des ordinateurs, il est possible que le dernier script ouvert soit affiché automatiquement dans la fenêtre de script. Si ce n'est pas le cas, cliquez sur « File » puis « Open file » et sélectionnez dans l'arborescence Windows le script R enregistré sur votre clef USB à la fin de la séance précédente (*script_etudiants.R*).

⇨ Le script s'ouvre mais les objets créés lors de la séance précédente (*dataframe*, vecteurs) n'ont pas été sauvegardés (espace de travail vide) : il faut donc ré-exécuter **certaines lignes du script de commandes**, notamment celle qui permet d'importer le tableau de données.

Exercice 1 : Le nombre de fleurs ligulées et le nombre de fleurs tubulées des pâquerettes sont-ils liés ?

Nous allons nous intéresser à la relation entre le nombre de fleurs ligulées (blanches) et le nombre de fleurs tubulées (jaunes) pour les pâquerettes prélevées en 2023 par la promotion précédente.

Pour des raisons pratiques, nous allons créer un sous-tableau de notre jeu de données ne contenant que les pâquerettes prélevées en 2023. La fonction `subset()` permet de créer un sous-ensemble.

⇨ Exécutez la commande : `data_2023 = subset(data_paq, Annee=="2023")`
Visualisez ce nouveau tableau.

Q. Comment se nomme ce nouveau tableau et combien de lignes et de colonnes possède-t-il ?

Pour commencer, nous allons représenter graphiquement cette relation.

Q. Si vous deviez l'imaginer, quel type de graphique dessineriez-vous pour représenter la relation entre ces deux variables ?

⇨ Exécutez la commande : `plot(Nb_ligu ~ Nb_tubu, data = data_2023)`
Le symbole ~ (tilde) signifie « en fonction de ». On l'obtient en tapant simultanément sur les touches « Alt Gr » et « 2 » du clavier, suivi d'un espace (ou de tout autre caractère).
Le graphique s'affiche dans la fenêtre en bas à droite (onglet Plots).

⇨ Pour ajouter un titre au graphique, il faut ajouter l'argument `main = "mon titre"` dans la liste des arguments de la commande (à la suite de `data = data_2023`, en le séparant par une virgule).

⇨ Vous pouvez également ajouter les arguments `xlab = "ma variable X"` et `ylab = "ma variable Y"` pour ajouter respectivement un titre à l'axe des abscisses et un à celui des ordonnées.

⇨ Pour mettre les points en couleur, ajoutez en plus l'argument `col = "couleur de mon choix en anglais"`.

⇨ RStudio permet de naviguer facilement entre des graphiques tracés successivement grâce aux flèches droite-gauche de l'onglet Plots.

⇨ Si vous souhaitez enregistrer un graphique pour l'utiliser dans un rapport par exemple, cliquez sur le bouton « Export » de la fenêtre graphique, puis choisissez « Save as image ». Dans la fenêtre qui s'ouvre, précisez le format d'enregistrement[3], choisissez le répertoire dans lequel sera sauvegardé le graphique (après avoir cliqué sur le bouton « Directory ») et nommez ce fichier dans « File name » (ex : fig1). Vous pouvez également simplement le copier afin de le coller dans un document.

[3] Le format PNG est préférable lorsque les graphiques ont des lignes. Le format JPEG est plus adapté aux photos dans lesquelles il y a des plages de couleurs.

Q. Que pensez-vous de la relation entre le nombre de fleurs tubulées et le nombre de fleurs ligulées pour les pâquerettes prélevées en 2023 ?

Nous allons maintenant tester la significativité de cette relation.

Q. En vous aidant de la clef pour choisir un test statistique figurant dans le polycopié du S3 (p. 16), quel test statistique paramétrique permettrait de tester cette relation ?

Q. Quelles sont les conditions de validité de ce test ?

Nous devons donc commencer par étudier la distribution de ces deux variables.

Q. Quelle est la représentation graphique la plus adaptée pour évaluer l'allure de ces distributions ?

⇨ Exécutez la commande `data_2023$Nb_ligu`.

Q. Qu'obtenez-vous ? A quoi sert le symbole `$` ?

⇨ Appliquez la fonction `hist()` afin de tracer le graphique approprié pour chaque variable.

Q. Que vous apprennent ces graphiques concernant les caractéristiques de la distribution du nombre de fleurs ligulées et du nombre de fleurs tubulées pour les pâquerettes prélevées en 2023 ?

⇨ Testez la normalité des variables `Nb_ligu` et `Nb_tubu` à l'aide d'un test de Shapiro effectué avec la fonction `shapiro.test()`.

Q. Quelles sont les hypothèses H_0 et H_1 du test de Shapiro ? Que concluez-vous ?

Q. Quelles solutions faut-il envisager pour pouvoir poursuivre l'analyse ?

Dans R, la fonction `log` permet d'obtenir le logarithme népérien d'une valeur. On peut par exemple afficher le logarithme du nombre de fleurs ligulées : `log(data_2023$Nb_ligu)`

⇨ Tracez la distribution et testez de nouveau la normalité du nombre de fleurs ligulées, mais cette fois sur la variable transformée par le logarithme. Pour gagner du temps, ré-utilisez les lignes de commande tapées précédemment en les modifiant où cela est nécessaire !

Q. Qu'en concluez-vous ? Quelle sera la prochaine étape ?

La fonction `cor.test()` permet de réaliser les tests de corrélation les plus classiques.

⇨ Affichez l'aide de cette fonction à l'aide du « ? » suivi du nom de la fonction : `?cor.test`

⇨ En utilisant l'aide, complétez la commande de votre script pour réaliser le test de corrélation de Pearson entre le nombre de fleurs tubulées et le nombre de fleurs ligulées transformé en logarithme. Il est à noter que l'ordre de saisie des variables dans la commande n'a pas d'importance ici car les deux variables ont un rôle symétrique.

Q. Quelles sont les hypothèses H_0 et H_1 du test ? Que pouvez-vous conclure ?

⇨ Ajoutez l'argument `alternative = "greater"` dans la commande précédente.

Q. A quoi sert-il ? Utilisez l'aide pour répondre à cette question.

Ici, pour des raisons pédagogiques, nous avons réalisé un test de corrélation de Pearson bien que la transformation logarithme n'ait pas permis de résoudre complètement le problème de normalité du nombre de fleurs ligulées. Un test de corrélation des rangs (non-paramétrique) aurait été plus approprié, notamment car la relation suspectée semble monotone. Le test de corrélation de Spearman est l'un de ces tests.

⇨ Modifiez la commande précédente pour réaliser le test de corrélation de Spearman.

Q. Après avoir posé les hypothèses H_0 et H_1 de ce test, concluez. Le résultat est-il cohérent avec celui trouvé précédemment ?

Q. Le résultat serait-il le même que l'on transforme ou pas le nombre de fleurs ligulées en logarithme ?

<u>Exercice 2</u> : Le nombre de fleurs tubulées des pâquerettes de milieu semi-urbain est-il le même durant l'été 2021 et au printemps 2023 ?

Q. Pourquoi ne s'intéresser aux pâquerettes que d'un seul milieu (semi-urbain) ?

Comme dans l'exercice n°1, il est préférable de créer un sous-ensemble de données ne contenant que les pâquerettes prélevées en milieu « Semi-urbain ».

⇨ Créez ce sous-tableau et nommez-le `data_su`.

Nous allons commencer par représenter le nombre de fleurs tubulées en fonction de l'année pour les pâquerettes du milieu semi-urbain.

⇨ Complétez puis exécutez la commande : `boxplot(Nb_tubu ~ Annee, data = ???)`

⇨ Utilisez les arguments vus dans l'exercice n°1 pour y ajouter un titre, légender les axes et mettre les boîtes en couleur.

Q. Comment s'appelle le graphique obtenu ?

Q. Comment pouvez-vous interpréter ce graphique ?

Nous cherchons maintenant à confirmer cette observation graphique en comparant statistiquement le nombre de fleurs tubulées des pâquerettes prélevées en 2021 et de celles prélevées en 2023 dans le milieu semi-urbain.

⇨ Commencez par construire deux nouveaux vecteurs, l'un contenant le nombre de fleurs tubulées pour les pâquerettes prélevées en 2021 (nommé `Nb_tubu_2021`) et l'autre contenant le nombre de fleurs tubulées pour les pâquerettes prélevées en 2023 (nommé `Nb_tubu_2023`), à l'aide de la fonction `subset()`.

⇨ Utilisez la fonction `mean()` pour calculer le nombre moyen de fleurs tubulées pour chacune des deux années.

Q. Qu'en déduisez-vous ? Que pouvez-vous en conclure ?

Q. Proposez la démarche à suivre pour identifier le test statistique approprié.

La première étape est de s'intéresser à la distribution du nombre de fleurs tubulées pour chacune des deux années.

⇨ Tracez l'histogramme pour chacune des deux années (cf. exercice n°1).

⇨ Testez ensuite la normalité du nombre de fleurs tubulées pour chacune des deux années (cf. exercice n°1).

Q. Que vous indiquent ces graphiques et les résultats des tests effectués ? Quelle sera la prochaine étape ?

⇨ Pour tester l'égalité des variances, nous allons utiliser le test de Fisher de comparaison de variances : `var.test(Nb_tubu ~ Annee, data=data_su)`

Q. Quelles sont les hypothèses H_0 et H_1 du test de Fisher ? Que concluez-vous ? Quel test de comparaison de moyennes sera le plus approprié ?

La fonction `t.test` qui a une syntaxe identique à la fonction `var.test` utilisée à la question précédente permet d'effectuer un test de Student avec ou sans correction de Welch selon les arguments utilisés.

⇨ En vous servant de l'aide de la fonction `t.test()`, complétez la commande du script fourni afin de réaliser le test le plus approprié à vos données.

Q. A quoi servent les arguments `var.equal` et `alternative` ?

Q. Comment auriez-vous pu modifier cette commande pour réaliser un test de Student (sans correction de Welch) ?

Q. Interprétez la sortie du test le plus approprié après avoir formulé ses hypothèses H_0 et H_1.

Dans notre situation, nous avons pu réaliser un test paramétrique sans même devoir transformer les données (en logarithme par exemple) pour se rapprocher de la normalité. Ce n'est pas toujours possible et il faut donc garder cette alternative en tête.

Dans le cas où même une transformation ne permettrait pas de vérifier les conditions de validité des tests paramétriques, une solution consisterait en l'utilisation du test de Wilcoxon qui est une alternative non-paramétrique au test de Student basé sur les rangs des valeurs et non les valeurs brutes. La fonction `wilcox.test()` permet de réaliser ce test.

⇨ Exécutez la commande suivante :

```
wilcox.test(Nb_tubu ~ Annee, data=data_su, alternative = "two.sided")
```

Q. Posez les hypothèses H_0 et H_1 de ce test et concluez. Ce résultat est-il cohérent avec celui du test paramétrique réalisé précédemment ?

<u>Exercice 3</u> : Le nombre de fleurs ligulées des pâquerettes prélevées en 2023 dépend-il du milieu ?

Q. En vous appuyant sur vos réflexions du TD1, quelle(s) hypothèse(s) biologique(s) pourrai(en)t être formulée(s) ici ?

⇨ Ré-exécutez la commande permettant de créer un sous-tableau de données ne contenant que les pâquerettes prélevées en 2023 (cf. exercice n°1).

⇨ Tracez le graphique permettant de représenter le nombre de fleurs ligulées en fonction des trois milieux.

Q. Que suggère ce graphique ?

Q. En vous aidant de la clef pour choisir un test statistique figurant dans le polycopié du S3 (p. 16), quel test statistique permettrait de confirmer votre intuition ? Quelles sont les hypothèses H_0 et H_1 ?

Pour répondre à cette question, nous allons ajuster un modèle linéaire à nos données. Ce modèle devra prédire le nombre de fleurs ligulées pour chacun des trois milieux.

⇨ Pour créer ce modèle, exécutez la commande suivante :

```
mod_ligu = lm(Nb_ligu ~ Milieu, data = data_2023)
```

Q. Dans ce modèle, quelle est la variable dépendante et quelle est la variable explicative ?

⇨ Affichez les paramètres du modèle ajusté en exécutant `mod_ligu`.

Q. Que signifie `intercept` ? Que représente chacun des autres paramètres du modèle ?

Pour s'assurer que les conditions permettant l'ajustement d'un modèle linéaire sont bien remplies, nous allons analyser les résidus du modèle en utilisant des outils graphiques ainsi que des tests statistiques.

⇨ Affichez les résidus du modèle ajusté en exécutant `mod_ligu$residuals`.

Q. Dans le cas de ce modèle, qu'est-ce qu'un résidu ?

Q. Rappelez les conditions de validité du modèle linéaire.

⇨ Commencez par tracer l'histogramme des résidus du modèle.

⇨ Tracez également le graphique *Normal Q-Q* qui représente les quantiles observés pour les résidus en fonction de ceux d'une loi normale théorique. Ce graphique permet de diagnostiquer d'éventuels écarts à la normalité des résidus d'un modèle.

```
qqnorm(mod_ligu$residuals)
qqline(mod_ligu$residuals)
```

Q. Si le modèle est correctement ajusté, quelle doit être la relation entre ces deux quantités ? Cette relation vous paraît-elle satisfaisante ?

⇨ Réalisez un test de normalité sur les résidus du modèle.

Q. Que pouvez-vous en conclure ?

Bien que ce résultat suggère d'ores et déjà que les conditions de validité du modèle linéaire ne sont pas toutes remplies, nous allons tout de même explorer l'hypothèse de constance de la variance des résidus (homoscédasticité).

⇨ Tracez le graphique des résidus en fonction des valeurs prédites par le modèle :

```
plot(mod_ligu$residuals ~ mod_ligu$fitted.values)
abline(a=0, b=0)
```

Q. Si le modèle est correctement ajusté, quelle relation attendez-vous entre les valeurs prédites par le modèle (`mod_ligu$fitted.values`) et les résidus (`mod_ligu$residuals`) ? Cet attendu vous paraît-il satisfait ?

Nous allons maintenant tester l'homoscédasticité des résidus en utilisant le test de Breusch-Pagan. Ce test n'est pas disponible parmi les fonctions de base de R mais est inclus dans la bibliothèque[4] `lmtest`.

⇨ Réalisez ce test à l'aide des commandes suivantes et concluez.

```
library(lmtest)
bptest(mod_ligu)
```

[4] Une bibliothèque (ou *package*) est un ensemble de fonctions R, généralement dévolues à des méthodes particulières ou à un domaine d'application spécifique. Une bibliothèque doit être installée puis chargée (fonction `library()`) avant de pouvoir utiliser ses fonctions.

Comme cela a été vu précédemment, il est souvent utile d'appliquer une transformation logarithmique à la variable dépendante afin de résoudre le(s) problème(s) identifié(s) ci-dessus.

⇨ Ajustez un nouveau modèle (nommé `mod_ligu_log`) dans lequel la variable dépendante devient le nombre de fleurs ligulées transformé en logarithme et la variable explicative reste le milieu. Effectuez ensuite toutes les analyses précédentes (inspection graphique des résidus, tests de normalité et d'homoscédasticité des résidus) sur le nouveau modèle. Pour vous simplifier le travail, pensez à copier les lignes de commande que vous avez déjà tapées dans votre script et à les modifier partout où cela est nécessaire !

Q. Concluez : la transformation logarithmique permet-elle de régler le(s) problème(s) identifié(s) dans notre premier modèle ?

Nous allons maintenant tester la significativité des différences du nombre de fleurs ligulées entre les milieux.

⇨ Exécutez la commande : `anova(mod_ligu_log)`

Q. Pourquoi choisit-on ici de réaliser l'analyse sur `mod_ligu_log` et pas sur `mod_ligu` ?

Q. Que représente chaque colonne de la table d'ANOVA ? Comment pouvez-vous conclure à partir de ce résultat ? Comment rapporter ce résultat dans un compte-rendu scientifique ?

Puisque la transformation logarithmique a permis de vérifier les conditions de validité du modèle linéaire, nous avons pu utiliser une analyse de variance pour tester les différences dans le nombre de fleurs ligulées produites par les pâquerettes dans les différents milieux. Il est préférable de l'utiliser lorsque cela est possible car elle est plus puissante qu'un test non-paramétrique pour détecter un effet significatif lorsqu'il existe réellement.

Toutefois, une transformation de la variable dépendante ne permet pas toujours de vérifier les conditions de validité du modèle linéaire : dans ce cas, il est recommandé d'utiliser une alternative non-paramétrique à l'analyse de variance, à savoir le test de Kruskal-Wallis. Ce test peut toujours être utilisé car il est basé sur les rangs des observations et est moins contraignant dans ses conditions d'application que le modèle linéaire. Ce test est mentionné dans le polycopié du S3 mais son calcul n'a pas été détaillé. Le logiciel R va le réaliser pour nous.

⇨ Réalisez ce test à l'aide de la commande suivante :

```
kruskal.test(Nb_ligu ~ Milieu, data = data_2023)
```

Q. Quelles sont les hypothèses H_0 et H_1 de ce test ? Son résultat est-il cohérent avec celui obtenu par l'ANOVA ?

Principales fonctions R vues pendant les TPs

Pour plus d'informations, voir l'aide des fonctions dans R : `? nom de la fonction`

Importation des données

`read.csv``(file = '…', header = …, sep = '…', dec = '…', stringsAsFactors = …)`

`file` : le nom du fichier

`header` : `TRUE` si la 1ère ligne du tableau correspond aux noms des colonnes du tableau, `FALSE` sinon

`sep` : le séparateur de colonnes (par exemple, si c'est le point-virgule, `sep = ';'`)

`dec` : le séparateur de décimales (une virgule en français ou un point pour un anglo-saxon)

`stringsAsFactors` : pour convertir les chaînes de caractères en facteurs

Visualisation des données

`View``(x = …)`

`x` : le tableau de données (dataframe) à visualiser

`summary``(object = …)`

`object` : l'objet (vecteur, tableau, modèle) dont on veut avoir une vue globale

`mean``(x = …, na.rm = …)`

`x` : vecteur dont on veut calculer la moyenne

`na.rm` : `TRUE` si les NA ne doivent pas être pris en compte dans le calcul de la moyenne, `FALSE` sinon

`plot``(formula = …, col = '…', xlab = '…', ylab = '…', main = '…' )`

`formula` : variable y (axe des ordonnées) en fonction de x (axe des abscisses), du type `y ~ x`

`col` : la couleur en anglais et entre guillemets

`xlab` et `ylab` : les légendes des axes X et Y entre guillemets

`main` : le titre du graphique

`hist``(x = …, xlab = '…', main = '…' )`

`x` : les données dont on veut représenter la distribution

`abline``(a = …, b = …)`

`a, b` : valeurs de l'ordonnée et de la pente de la droite

`barplot``(height, xlab = '…', main = '…' )`

`height` : matrice de valeurs décrivant la hauteur des barres (par exemple, table de contingence)

Manipulation des données

`subset``(x = …, subset = …)`

`x` : l'objet dont on veut extraire un sous-ensemble

`subset` : le(s) critère(s) selon le(s)quel(s) on découpe l'objet `x` pour obtenir le sous-ensemble

`log``(x = …)`

`x` : le vecteur sur lequel on souhaite appliquer le logarithme népérien

`table``(…)`

`…` : un (ou plusieurs) vecteur(s) qualitatif(s) pour le(s)quel(s) on veut obtenir la table de contingence

`tapply``(x =…, index = …, function = …)`

`x` : vecteur sur lequel on souhaite appliquer une fonction

`index` : vecteur de type facteur

`function` : fonction qui doit être appliquée au vecteur pour chaque modalité du facteur

Tests statistiques

`shapiro.test``(x = …)`

`x` : la variable (un vecteur) dont on veut tester la normalité de la distribution

`cor.test``(x = …, y = …, method = '…', alternative = '…')`

`x` : la première variable

`y` : la seconde variable

`method` : `'spearman'` pour un test de Spearman, `'pearson'` pour un test de Pearson

`alternative` : hypothèse alternative qui doit être testée (`'two.sided'`, `'less'`, `'greater'`)

`var.test``(formula = …, alternative = '…')`

`formula` : variable y en fonction de x `(y ~ x)`

`alternative` : hypothèse alternative qui doit être testée (`'two.sided'`, `'less'`, `'greater'`)

`t.test``(formula = …, alternative = '…', var.equal = …)`

`formula` : variable y en fonction de x `(y ~ x)`

`alternative` : hypothèse alternative qui doit être testée (`'two.sided'`, `'less'`, `'greater'`)

`var.equal` : pour préciser si homoscédasticité (`TRUE`) ou non (`FALSE`)

`wilcox.test``(formula = …, alternative = '…')`

`formula` : variable y en fonction de x `(y ~ x)`

`alternative` : hypothèse alternative qui doit être testée (`'two.sided'`, `'less'`, `'greater'`)

`lm``(formula = …, data = …)`

`formula` : variable y en fonction de x `(y ~ x)`

`data` : le tableau qui contient les variables `x` et `y`

`bptest``(formula = …)`

`formula` : variable y en fonction de x `(y ~ x)` ou le nom d'un modèle

`anova``(object = …)`

`object` : le modèle dont on veut obtenir la table d'analyse de variance

`chisq.test``(x = …)`

`x` : la table de contingence

Divers

`library``(package = …)`

`package` : la bibliothèque que l'on souhaite charger dans l'espace de travail