Analyses statistiques de base avec **R** et `Rcmdr` comme interface graphique

Christian Jost - Biologie Santé - UE 3M7BS15M - 2010/11

Ce polycopié a été développé en 2009 pour une introduction aux statistiques de base (Chi-2, tests à un, deux ou plusieurs échantillons) en 4 TP de 2h, à l'aide de **R** 2.9.2 et R-commander 1.5-3. Attention, R-commander évolue rapidement, des versions antérieures ne correspondent pas toujours aux indications dans ce polycopié.
Pour l'organisation des données les logiciels type « tableur »
(Excel, Openoffice, **Libreoffice**) sont très utiles. Dans ces TP
on va se servir de **Libreoffice** qui est la suite de OpenOffice.

Veuillez noter que le vide dans les encadrés de ce polycopié n'est pas une erreur, c'est votre espace à vous pour prendre des notes.

Tout commentaire bienvenu (jost@cict.fr)

16 février 2012

Table des matières

Chapitre 1

Organisation de données, description et graphisme

Dans le traitement statistique (graphisme, estimations des tendances moyennes, tests d'hypothèses, ...) on distingue entre deux grands types de données : les données continues (tel que le poids d'un homme, la durée de vie d'un animal, la concentration d'une molécule dans le sang, ...) et les données dites catégoriques (couleur des yeux, sexe d'un individu, ...). Dans ce TP nous allons apprendre pour ces deux types de données

- comment les organiser afin de les utiliser dans un logiciel statistique,
- comment les importer ensuite dans un tel logiciel,
- et comment faire les graphismes et statistiques descriptifs les plus communs.

1.1 Le poids des moutons en fonction de saison et sexe

Dans le cadre d'une étude sur l'influence de la testostérone sur le comportement grégaire des moutons les chercheurs ont travaillé à deux reprises, en hiver et en été, avec des moutons mâles (♂) et femelles (♀). Le poids des moutons est une covariable importante, par exemple un dosage hormonal doit être calibré selon ce poids. Nous allons explorer pour le moment simplement ce poids en fonctions du sexe et de la saison.

Le fichier `moutonsPoidsOrig.xls` contient les données brutes telles qu'un stagiaire les noterait intuitivement. Il se peut que vous ne voyez pas l'extension `.xls` dans votre version de Windows, Micro$oft aime bien cacher certaines choses des utilisateurs, mais dans notre cas cela ne sera pas grave. Ouvrez ce fichier dans le tableur de LibreOffice (lancer le logiciel, menus Fichier - Ouvrir). Combien de moutons mâles et femelles y avait-t-il en hiver et en été ?

Un tableur (tel qu'Excel ou OpenOffice) est un très bon outil pour faire une première exploration des données. Calculons d'abord moyenne et écart type de chaque groupe de moutons. Sélectionnez une cellule sous la colonne de données que vous voulez décrire, cliquez sur le menu `Insertion-Fonction...` et choisissez la fonction `moyenne`, vous pouvez ensuite vous laisser guider par l'interface du tableur pour sélectionner la plage de données sur laquelle vous voulez calculer la moyenne. Faites la même manoeuvre pour calculer dans la cellule au-dessous l'écart type (fonction `ecartype`). Notez moyennes et écart types pour les quatre groupes de mouton :

Vous avez vu dans la liste des fonctions que votre tableur offre énormément d'outils pour faire des calculs sur vos données, mais veuillez noter que ces tableurs ne sont pas des logiciels spécialisés dans le traitement statistique, plusieurs bugs et erreurs se cachent par exemple dans les fonctions d'Excel sans que Micro$oft les corrigerait. C'est donc un bon outil d'exploration, mais pas de calculs statistiques à des

fins scientifiques. Il faut passer à un logiciel reconnu dans le monde scientifique, tel que SPSS, Minitab, Systat, Splus, SAS, **R** ou autre. Le fonctionnement de tous ces logiciels est très similaire, mais il faut leur préparer les données dans un format bien défini. La façon dont notre stagiaire a noté dans son tableur les poids des moutons serait complètement inaccessible pour un logiciel statistique.

Le principe de base est de donner à chaque mesure une ligne, qui contiendra cette mesure et également toutes les informations qui lui sont propres, telles que le sexe et la saison dans le cas de nos moutons. Il faut donc réorganiser ces données (dans un nouveau fichier du tableur) en trois colonnes, la première contenant les poids, la seconde le sexe et la troisième la saison. Ici se pose une première question importante : comment « encoder » les valeurs des variables catégoriques sexe et saison ? On pourrait écrire `mâle` ou `Mâle` quand c'est un mâle et `été 2007` si la mesure a été prise en été 2007, tout lecteur humain comprendrait. Seulement, en faisant cela sans réfléchir, on risque que le logiciel statistique ne comprenne pas nos encodages et fasse n'importe quoi. Par exemple, est-ce que `été` et `2007` sont deux noms différents ou pas ? On ne peut jamais non plus être sur que le logiciel sache correctement traiter les lettres accentuées (vos collègues informatitiens peuvent vous raconter des histoires là-dessus). Et enfin, même si le logiciel maîtrise les accents, est-ce que `mâle` est la même chose que `Mâle` ? Les règles suivantes sont donc toujours de rigueur pour une utilisation ultérieure dans **R** sans problèmes :

- toujours commencer le nom par une lettre (pas un chiffre),
- ne jamais mettre un espace dans un nom,
- ne jamais utiliser des lettres accentuées,
- soigneusement distinguer entre lettres majuscules et minuscules,
- éviter des signes qui ont une signification pour le calcul mathématique (+, -, :, ...)

Par contre, vous pouvez utiliser des lettres telles que le point (.) ou le sous-ligné (_). Encodez donc sexe et saison dans votre tableau excel ou Libreoffice par les symboles `M` et `F` pour le sexe et `ete` ou `hiver` pour la saison. Enregistrez votre fichier sous le nom `moutonsPoids.xls` (format qui est utilisable par excel) ou sous le nom `moutonsPoids.ods` (format qui est utilisable par LibreOffice) sur le bureau dans un dossier nommé `BMCEM-2012` (créez-le s'il n'existe pas encore).

On a presque fini. Nos données sont maintenant organisées en trois colonnes, il est donc d'usage de donner un nom spécifique à chaque colonne (d'après les même règles données ci-dessus pour les modalités d'un facteur) et de noter ces noms directement au-dessus de chaque colonne. Dans le tableur, sélectionnez donc la première cellule, insérez une ligne vide et ajoutez au-dessus des trois colonnes leurs noms (par exemple `poids`, `sexe` et `saison`).

On va encore s'entraîner un peu dans l'utilisation du tableur. Le poids étant une grandeur positive il est souvent utile pour les calculs statistiques de le transformer d'abord en logarithme (une grandeur positive sera ensuite inclut entre $-\infty$ et ∞). A l'aide des fonctions du tableau, ajoutez donc une quatrième colonne au nom de `poids.log` contenant le logarithme népérien du poids. Notez maintenant les 5 premières lignes de votre fichier :

Il ne nous reste qu'à passer ces données à un logiciel statistique. Certains peuvent lire des fichiers LibreOffice ou excel directement, mais pas tous. Par contre, tous comprennent des fichiers sous format texte. Après avoir sauvé votre fichier sous le format `.ods` ou `.xls`, enregistrez-le ensuite sous le nom `moutonsPoids.csv` et au format `Text (.CSV)`. Cochez la case `Editer les paramètres du filtre`. Cette case va ouvrir à l'étape suivante une nouvelle fenêtre qui vous laisse préciser trois choses : l'encodage du texte (laissez-le sur utf-8, même si vous ne comprenez pas ce que cela veut dire) ; le caractère qui permet de séparer deux données (laisser le choix par défaut, la virgule) ; comment séparer des mots l'un de l'autre (laisser le choix par défaut, cela permettra à **R** de lire les chiffres avec la virgule comme séparateur des décimales).

Ouvrez ce nouveau fichier dans `Blocnotes` ou `Notepad+` et vérifiez bien qu'il contient un tableau avec quatre colonnes et autant de lignes que de mesures disponibles (plus une ligne en haut pour les noms des colonnes), mais rien d'autre. Est-ce que les décimaux sont bien indiqués par une virgule ?

1.2 R, un logiciel statistique libre de droit (open source)

Le logiciel qu'on utilisera, **R**, n'est pas seulement gratuitement disponible sur internet pour tout le monde ; il est du type « open source », ce qui veut dire que même les codes de programmation du logiciel sont disponibles librement et peuvent être repris et améliorés par n'importe qui (par exemple vous). Ceci assure, entre outre, que le logiciel ne se perd pas : si les programmeurs actuels arrêtent de travailler là-dessus, d'autres peuvent prendre le relais. Tant qu'il y a des utilisateurs du logiciel il y aura toujours des programmeurs qui consacrent une partie de leur temps à sa maintenance. Apprendre **R** est donc en quelques sorte un apprentissage à très long terme (ce qui n'est pas toujours le cas des logiciels commerciaux, voir Word qui vous force à changer vos habitudes avec chaque nouvelle version). Mais il a le désavantage de ne pas toujours être d'une utilisation intuitive, et des fois on a même besoins de faire un peu de programmation pour s'en servir. Ceci dit, ce n'est pas sorcier pour un étudiant M1.

Sur les ordinateurs de la salle informatique tout est déjà installé, mais on va vite discuter comment l'installer sur votre propre ordinateur (on le fait sous l'hypothèse que vous avez un accès internet, pour les autres je mettrai à disposition un CD). Allez sur le site
`http://www.r-project.org/`
qui est la 'homepage' de **R**. Vous voyez que tout est par défaut en anglais, c'est normal pour un logiciel qui est utilisé partout dans le monde, mais il y a souvent des adaptations aux diverses langues du monde. Pour le moment, cliquez à gauche sur `CRAN` (**C**omprehensive **R** **A**rchive **N**etwork) qui vous laisse choisir le miroir/serveur [1] par lequel vous voulez passer. Prenez par exemple un des serveurs à Lyon.

Maintenant vous pouvez directement trouver l'installateur pour votre système d'exploitation (Linux, MacOS ou Windows). En cliquant sur `Windows` vous verrez un nouveau lien `base` qui contient l'installateur pour Windows. Il suffit de le télécharger sur votre ordinateur et de suivre les instructions. On se servira également de « packages » supplémentaires. Pour les installer il suffit de démarrer **R** et de taper sur la ligne de commande

```
> install.packages("Rcmdr",dependencies=T)
```

R va peut-être vous redemander le miroir à utiliser et ensuite télécharger tout ce qui est nécessaire. Ce package s'appuie sur plusieurs autres packages disponibles pour **R**, l'argument `dependencies=T` dit à **R** de les installer en même temps que `Rcmdr`. Ceci va prendre quelques temps en fonction de votre connexion internet, et en cas de problèmes de téléchargement il suffira de relancer la commande. Pour tester à la fin si toute l'installation a réussi vous pouvez taper la commande

```
> library(Rcmdr)
```

1.3 Premiers pas dans R et Rcmdr

Démarrez **R** et lancez Rcmdr [2] par la commande

```
> library(Rcmdr)
```

Vous verrez s'afficher une fenêtre similaire à celle de la Fig 1.1. Nous voulons maintenant travailler avec les données des moutons. La première chose à faire est donc de dire a **R** dans quel répertoire se trouve le fichier des données que vous avez préparé ci-dessus. Cliquez dans `Rcmdr` sur le menu `Fichier -> Changer le répertoire de travail` et sélectionnez ce répertoire.

Maintenant on est prêt à charger les données. Cliquez sur `Données -> Importer des données -> depuis un fichier texte ...`. Vous verrez la fenêtre Fig 1.2 : donnez un nom au tableau à importer (par exemple `moutons`) et vérifiez que la case `Noms de variables dans le fichier` est coché (c'est comme ça qu'on a préparé les données, la première ligne contient les noms des colonnes). Un aspect très important est la case `Séparateur décimal - Virgule` : votre tableur fonctionne avec des virgules décimales (comme il est d'usage en France et dans quelques autres pays), mais un logiciel international utilise le standard international qui et le point décimal. Il faut signaler à **R** que dans votre fichier la virgule est utilisée afin d'assurer la conversion en point décimal compréhensible pour **R**. Un autre aspect est le `Séparateur de champs` : dans nos exportations depuis LibreOffice nous avons choisi la virgule,

1. Un miroir est simplement un serveur web quelques part dans le monde qui contient une exacte copie du contenu de `CRAN`. Ceci permet à plus d'utilisateurs d'accéder à ces ressources au même moment.
2. Si au milieu du TP vous devez quitter `Rcmdr` sans quitter **R**, vous pouvez redémarrer `Rcmdr` par la commande `Commander()`

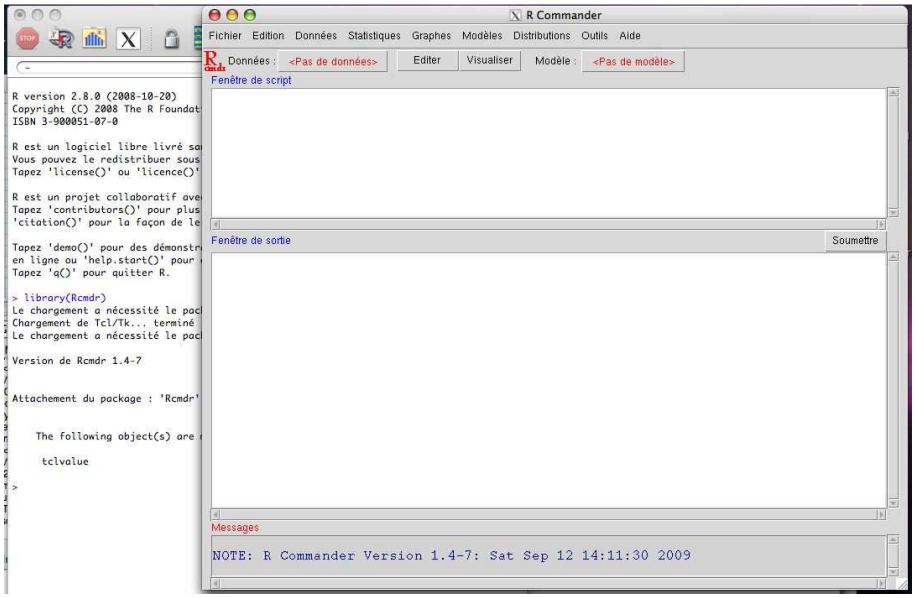


FIGURE 1.1 – La fenêtre d'ouverture de `Rcmdr`, ici sous Mac OS, mais sous Windows elle est très similaire. La `Fenêtre de script` va afficher les commandes que `Rcmdr` exécute selon vos clicks dans les menus, et la `Fenêtre de sortie` affichera les réponses de **R** et contiendra en particulier les résultats statistiques.

mais selon l'origine de vos données d'autres séparateurs peuvent exister. Cliquez sur `Ok` et sélectionnez votre fichier. En cliquant dans `Rcmdr` sur `Visualiser` vous pouvez vous assurer que vos données ont été correctement lu (le séparateur des décimaux devrait maintenant être un point, **R** a fait la conversion). Après vérification refermez cette fenêtre, des fois une fenêtre ouverte bloque `Rcmdr`, en particulier les fenêtres de dialogue, c'est donc plus prudent de les fermer systématiquement après usage.

Vous avez du remarquer que le choix des menus à déclenché l'affichage des commandes que **R** a effectuées dans la `Fenêtre de script`. Pour le moment vous n'avez pas besoin de comprendre les détails des commandes, mais à fur et à mesure des TP nous passerons à ce langage nous-même.

Regardons d'abord les statistiques descriptives globales par la commande `Statistiques -> Résumé -> Jeu de donnée active`. Les commandes effectuées apparaitront de nouveau dans la `Fenêtre de script`, mais en plus les résultats en fonction de ces commandes s'affichent dans la `Fenêtre de sortie`. Notez le résultat et ce que veut dire chaque ligne :

Il serait également intéressant de regarder moyenne et écart type selon sexe et saison. Sélectionnez `Statistiques -> Résumé -> Tableau de statistiques`, surlignez `saison` et `sexe` (en restant appuyé sur la touche majuscule) et choisissez le paramètre statistique que vous voulez (moyenne et écart type). Notez les résultats et décrivez les différences du poids moyen selon sexe et saison :

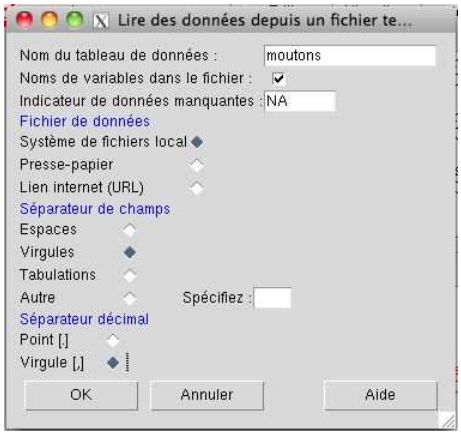


FIGURE 1.2 – Fenêtre d'importation de données

1.3.1 Le graphisme de données continues

Une première chose à faire avec des nouvelles données est d'essayer de les visualiser (cela vous permettra en outre de détecter des données aberrantes dues à des fautes de frappe ou d'autres causes). Faisons par exemple un histogramme global des poids par le menu `Graphes -> Histogramme ...`. Vous rappelez-vous ce que veut dire représentation en `Fréquence`, `Pourcentage` ou `Densité` ? Répondez dans la case (et jouez avec les trois options afin de vous rafraîchir la mémoire) :

Remarque : vous avez peut-être déjà vu que dans excel il y a aussi un type de graphique appelé « histogramme », mais il ne correspond pas du tout à l'histogramme dans le sens statistique. Malheureusement Micro$oft ne respecte pas du tout les notations établies et essaye plutôt d'imposer sa propre vision (arbitraire) : il y a un risque qu'il y arrive par sa position de quasi monopole sur le marché des PC.

Une autre représentation très utile sont les boîtes à moustache ou boîtes de dispersion, `Graphes -> Boîte de dispersion`. Faites le `Graphe par groupe` des saisons. Qu'est-ce que représentent les divers lignes et boîtes dans une boîte à moustache ? Qu'est-ce que vous pouvez dire sur le poids des moutons selon la saison ?

Le graphique indique que l'ordonnée contient des `poids` (le nom de la colonne dans le tableau ou `data.frame` de données des moutons), mais on aimerait bien ajouter l'information que le poids a été mesuré en kg. Regardez la `fenêtre de script`, en particulier la dernière ligne
`boxplot(poids~saison, ylab="poids", xlab="saison", data=moutons)`
en fait, `ylab` est le label de l'ordonnée, vous pouvez donc facilement modifier ce label
`boxplot(poids~saison, ylab="poids (kg)", xlab="saison", data=moutons, main="Moutons")`
sélectionnez cette ligne et cliquez sur `soumettre`. Enfin, essayez la commande
`boxplot(poids~saison, ylab="poids (kg)", xlab="saison", data=moutons,`
`main="Poids des moutons", notch=T)`
Comment interprétez-vous ce nouveau graphique ?

Ce graphique perfectionné peut être enregistré pour servir dans un rapport : cliquez sur `Graphes -> Sauver le graphe ...` et vous verrez que, sous Windows, vous avez deux possibilités

bitmap : qui l'enregistre en un nombre de pixel fixe (`png` est mieux que `jpg` pour les graphiques statistiques), c'est-à-dire la qualité de l'image sera défini une fois pour tout, si vous l'agrandissez ensuite trop vous verrez les pixels,

comme pdf/postscript : qui l'enregistre en format vectoriel (pdf ou postscript), si vous changez donc la taille la qualité restera toujours la même.

Selon le logiciel avec lequel vous écrivez vos rapports vous pouvez choisir l'un ou l'autre. Enregistrez-le sous le format `png` dans le répertoire sur le bureau et vérifiez le résultat.

Enfin, un autre graphe utile est celui des moyennes avec leurs erreurs standards. Vous pouvez le faire par le menu `Graphes -> Graphe des moyennes ....` et en surlignant `saison` et `sexe` afin d'indiquer que vous voulez voir les quatre différentes moyennes. Vous pouvez améliorer les label/étiquettes comme indiqué ci-dessus, par exemple

```
plotMeans(moutons$poids, moutons$saison, moutons$sexe, error.bars="se",
xlab="Saison", ylab="Poids (kg)", main="Poids des moutons")
```

Question de rappel : qu'est-ce que représente l'erreur standard ?

1.4 Un exemple de données catégoriques : le phénotype

Passons à un exemple de données catégoriques, par exemple les phénotypes. Vous connaissez probablement tous l'anémie falciforme, une maladie héréditaire due à une mutation récessive d'un gêne particulier. Si on « croise » des parents hétérozygotes (Rr × Rr) on obtient 3 génotypes (RR, Rr et rr) en proportion 1 : 2 : 1, donc les deux phénotypes 'normal' et 'atteint' en proportion 3 : 1.

En général, regarder les fréquences de phénotypes dans des expériences de croisement permet d'inférer si tel ou tel phénotype est codé au niveau génétique comme l'anémie falciforme ou si d'autres règles d'hérédité doivent être évoquées pour expliquer ce phénotype. Charger dans `Rcmdr` les données du fichier `phenoTypeYeuxEtud.xls` (après exportation correcte du tableur) et visualisez les données. Il contient la couleur des yeux pour 60 individus. Comment est-ce que ce type de données a été encodé pour le logiciel statistique ?

A quelles fréquences est-ce qu'on observe les deux phénotypes (`Stastiques -> Résumé -> Distributions de fréquences ....`) ?

En passant par la boîte de dialogue du menu précédent vous avez probablement vu la case a cocher `Test d'ajustement ....` Qui est-ce qui n'a jamais entendu parler du test du Chi-deux (χ^2) ? Petit rappel des stats (de licence) : on pose une hypothèse nulle (H0, par exemple les deux phénotypes existent en proportion 3 : 1). Un test statistique nous donne la probabilité p d'observer nos données si H0 est vraie. Si, pour un α fixé d'avance, on observe $p < \alpha$, on conclue que c'est trop peu probable et on rejette H0. Faites ce test en cochant la case et rapportez/interprétez le résultat.

Quand vous faites des tests statistiques, quelles sont les erreurs que vous pouvez commettre ?

On reviendra là-dessus en cours de statistiques. Passons maintenant à la représentation graphique de ce type de données. La plus répandue est le diagramme en bâton qui représente pour chaque catégorie un bâton dont la hauteur est la fréquence de cette catégorie. Faites ce graphique avec le menu `Graphes -> Graphe en barre`, ensuite modifiez la commande afin de donner des étiquettes plus adaptées et sauvez le graphique en format `png` (notez la commande modifiée) :

1.4.1 Comment garder une trace de ses analyses ?

Une fois que vous avez fini vos analyses statistiques il faut sauver cette analyse pour en garder une trace (soit pour écrire un rapport, soit simplement pour pouvoir retrouver ces résultats plus tard en cas de besoin). Pour enregistrer les résultats eux-mêmes vous pouvez d'abord éditer la `Fenêtre de sortie` en effaçant ce qui ne vous intéresse pas et en gardant uniquement les commandes qui ont calculées les résultats importants tels que les statistiques descriptives de base (moyennes, écart types, fréquences, tailles des échantillons, ...) et les tests d'hypothèses. Ensuite vous sauvez ce qui reste en format texte en passant par le menu `Fichier -> Sauver les sorties`.

Mais souvent il se trouve que vous découvrez une erreur dans vos données et que vous devez refaire l'analyse. Evidemment, vous pouvez refaire toutes les manipulations ci-dessus, mais cela prend du temps. Une solution plus « professionnelle » est d'éditer la `Fenêtre de script` en ne gardant que les commandes des analyses que vous voulez garder (par exemple la lecture des données, l'analyse en χ^2 et le graphique associé, le graphique des moyennes) et en l'enregistrant. A une prochaine séance **R** vous pouvez simplement démarrer `Rcmdr`, ouvrir le script, sélectionner toutes les commandes et cliquer sur `soumettre` (il faut peut-être d'abord redire à `Rcmdr` où se trouve le répertoire de travail).

Enfin, si vous avez fait un login personnalisé (votre numéro de carte d'étudiant), vous disposez sur le disque dure `Z:` d'un espace sur le serveur commun. Si vous copiez les dossier créé sur le bureau dans votre espace vous le retrouverez quand vous vous connectez la prochaine fois.

1.5 Entrainement

1. Prenons l'exemple du rendement en fonction d'un engrais (4 types d'engrais, on mesure le rendement sur plusieurs réplications). Vos données sont probablement saisies dans un tableur (genre excel) avec une colonne contenant les rendements pour un type d'engrais, donc 4 colonnes :

eng1	eng2	eng3	eng4
45	31	32	39
42	34	33	38
39	40	38	41
41		33	44
			48

   Réorganisez ces données dans LibreOffice afin de pouvoir les importer dans **R** et de faire un graphique « parfait » représentant le rendement en fonction de l'engrais en boîtes à moustache. Enregistrez le graphique sous format `png` ou `pdf`. Faites aussi le graphique des moyennes avec les erreurs standards et écrivez une légende de ce graphique.

2. Le fichier `phenotypes.txt` contient les résultats d'un croisement avec deux phénotypes différents à génotype dominant-récessif, A ou a pour le premier phénotype et B ou b pour le second (lettre majuscule = phénotype dominant). Préparez ce jeu de données dans LibreOffice, importez-le ensuite dans `Rcmdr`, calculez les fréquences des 4 phénotypes et faites un graphiques pour représenter ces données. Cherchez (dans votre mémoire ou sur internet) les fréquences attendues et testez si

les fréquences observées sont compatibles avec ces fréquences attendues. Quelle est le résultat ? Comment pouvez-vous l'interpréter ?

3. Aller sur le site de **R**, `http://www.r-project.org/`, et cliquez à gauche sur le lien `Documentation - Manuals`. Ne vous laissez pas décourager par le fait que tout semble être en anglais ; en bas de la page il y a un lien `contributed documentation` où vous trouverez des manuels et tutoriaux dans divers langues, en particulier aussi en français.

Chapitre 2

Quand les interfaces graphiques ne suffisent plus : l'exemple du bootstrap

Un rôle important des méthodes statistiques est l'estimation des paramètres d'un système dynamique. Dans ce chapitre on va explorer ce rôle à l'aide de l'exemple du taux de départ d'un compartiment (voir les exemples du chapitre 4 du polycopié modélisation). Plus particulièrement, le compartiment en question est l'action d'une fourmi de transporter le cadavre d'un congénère (on suppose que cela se fait pour des raisons d'hygiène, il faut éloigner tous les déchets du nid et les agréger quelque part pour diminuer le risque de les rencontrer). Une fourmi rejoint ce compartiment en prenant un cadavre, et elle le quitte en déposant ce cadavre. C'est cette seconde partie, dépôt de cadavre, qu'on veut modéliser/estimer. [1]

2.1 Les courbes de survie et les modèles à compartiments

Le fichier `depotSpont16.xls` contient les mesures du temps (en secondes) qu'une fourmi granivore *Messor sanctus* passe à transporter un cadavre avant de le déposer. Ouvrez ce fichier dans excel ou OpenOffice. Combien de mesures y a-t-il dans ce jeu de données ? Et combien de mesures y a-t-il entre 0 et 20 s, entre 21 et 40s, ..., entre 181 et 200s (servez vous de la fonction `Données - Trier` pour faire ce comptage) ? Quelle tendance observez-vous ?

Vous voyez dans ces comptages qu'il y a beaucoup de petits temps et quelques temps très long, mais on ne voit pas très bien la structure détaillée de ces durées.

La représentation graphique de durées d'évènements a été bien étudiée dans le contexte des études cliniques (temps avant guérison, temps avant rechute, temps avant la mort) et un des outils qui a émergé est l'analyse de survie. Dans une telle analyse chaque temps de transport représente un évènement. Pour tracer une **courbe de survie** on les fait tous commencer au même moment et ensuite on trace le nombre d'événements qui durent encore au cours du temps. Dans notre exemple on trace le nombre de fourmis qui transportent encore au cours de temps, et à chaque fois qu'une fourmi dépose son cadavre ce nombre est diminué par un. Par exemple, le temps le plus court est 9, donc jusqu'à $t = 9$ il y a 71 fourmis qui transportent, mais immédiatement après seulement 70, et cet effectif dure jusqu'au prochain événement

1. Vous pouvez remplacer la fourmi transporteuse par d'autres processus biologique : une substance chimique qui va se dégrader de façon spontané, une enzyme A qui change de configuration s'il rencontre au hasard l'enzyme B, une ARNm qui rencontre au hasard un ribosome pour être transcrit. La modélisation du temps avant que cet événement arrive sera très similaire à la modélisation dans ce TP.

de dépôt (à 10s). [2] Mettez donc dans votre tableur ces temps en ordre croissant et ajoutez dans la colonne à droite le nombre de fourmis qui transportent encore :

```
tempsDep  rang
9           71
10          70
11          69
12          68
12          67
13          66
13          65
15          64
...         ...
```

formant ainsi un tableau avec deux colonnes. Tracez maintenant la courbe de survie, c'est-à-dire le `rang` en fonction du `tempsDep` (choisissez le `nuage de points` comme type de graphique). Quelle forme de courbe observez-vous ? Quelle forme observez-vous après avoir changé l'échelle de l'ordonnée en logarithme (double-clicker sur l'ordonnées et ensuite cochez la case `Echelle logarithmique`) ? Recopiez les graphiques et commentez.

Importez maintenant ces données dans `R-cmdr` (soit en copiant le tableau et en les y important directement depuis le presse-papier, soit en passant par l'enregistrement dans un fichier texte comme dans le premier TP) et donnez leur le nom `depot16`.

2.2 Estimation d'un taux de départ d'un compartiment

La courbe de survie, par sa définition, décroit de façon monotone. Le modèle générale décrivant cette décroissance est

$$\frac{dN(t)}{dt} = -\lambda(N(t), t)N(t)$$

où $N(t)$ est le nombre de fourmis qui transportent encore et $\lambda(N(t), t)$ est le taux de décroissement qui peut dépendre aussi bien de $N(t)$ comme de t, et $\frac{dN(t)}{dt}$ et la variation du nombre de fourmis qui transportent encore. Le but est d'estimer ce $\lambda(N(t), t)$ à partir des données mesurées.

Retracez d'abord la courbe de survie avec `Rcmdr` (`Graphes - Nuage de points`, avec les options décochées comme dans la Fig 2.1) en mettant l'ordonnée en échelle logarithmique (vous pouvez ajouter au tableau de données importées une colonne contenant le logarithme néperien du nombre d'individus qui transportent encore (voir le dernier TP) ; ou alternativement vous servir des cases à cocher dans le dialogue). Servez-vous aussi de cette boîte de dialogue pour donner des noms « explicatifs » aux axes.

Le fait qu'en échelle log-linéaire la courbe de survie devienne quasiment une droite veut dire que $\lambda(N(t), t)$ ne dépend pas de N et ne change pas au cours du temps, donc $\lambda(N(t), t) = \lambda$ (λ est constant). Ce λ est facile à estimer, il correspond à la valeur absolue de la pente de cette droite. Il suffit donc d'estimer cette pente par une régression linéaire.

2. Dans ce jeu de données, il n'y a pas de temps inférieurs à 9s, ce qui peut paraître étonnant. En fait, dans les mesures de durées d'événements il est souvent impossible de mesurer des temps inférieurs à un certain seuil qui est nécessaire expérimentalement pour s'assurer que l'événement ait effectivement commencé. Il manque donc dans le jeu de données des événements qui ont durés moins que ce seuil. Mais les méthodes d'analyse de survie savent prendre en compte cette contrainte.

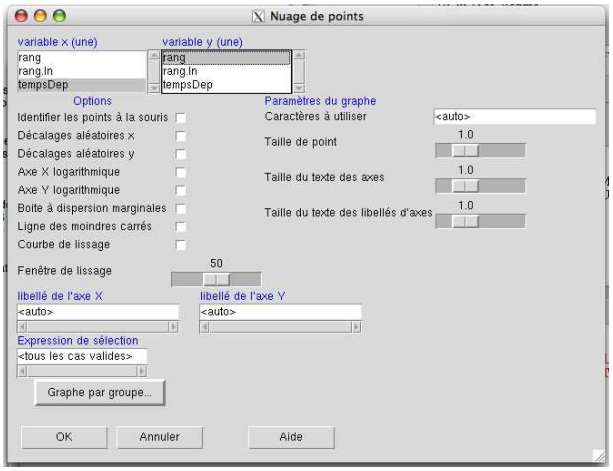


FIGURE 2.1 – Les options pour tracer un nuage de points, $x \sim y$. Ces options vous permettent notamment d'ajuster une ligne droite à votre nuage de points et à donner des légendes aux axes.

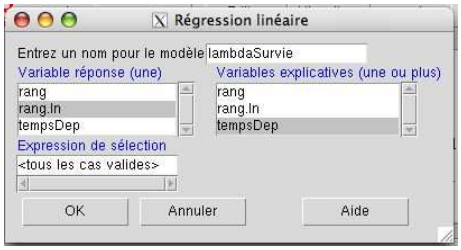


FIGURE 2.2 – Boîte de dialogue pour faire une régression linéaire.

2.2.1 Principe de la régression linéaire

On a un nuage de points $(x_i, y_i)_{1 \leq i \leq n}$ et on veut trouver une droite $y = a + bx$ qui s'ajuste bien dans ce nuage de points. L'idée est de trouver des coefficients a et b tel que les valeurs prédites, $y_{i,pred} = a + bx_i$ sont le plus proche possible des valeurs observées y_i. Le critère le plus souvent utilisé est de trouver a et b tel que la somme des carrés des écarts,

$$E = \sum_{i=1}^{n} (y_{i,pred} - y_i)^2 = \sum_{i=1}^{n} (a + bx_i - y_i)^2$$

soit minimale. Il y a des formules qui permettent de faire cela à la main, mais nous le feront ici avec `Rcmdr`. Pour cela il faut maintenant ajouter une colonne (au nom de `rang.ln`) au jeu de données qui contient le logarithme néperien du nombre d'individus qui transportent encore (`Données - gérer les variables dans le jeu de données actif - Calculer une nouvelle variable ...`). Choisissez ensuite le menu `Statistiques - Ajustement de modèle - Régression linéaire ...` et remplissez le dialogue tel qu'indiqué dans la Fig 2.2. En cliquant sur `OK` **R** fait les calculs pour vous. Regardons d'abord les commandes que **R** a fait (Fenêtre de script)

```
> lambdaSurvie <- lm(rang.ln ~ tempsDep, data=depot16)
> summary(lambdaSurvie)
```

La première ligne fait la régression linéaire (lm = linear model) de `rang.ln` (le logarithme du nombre d'individus encore transportant, vous avez peut-être donné un autre nom) en fonction du temps de transport) et met le résultat dans la variable `lambdaSurvie`. La seconde ligne affiche les résultats de bases (voir `Fenêtre de sortie`) dont les plus importants pour nous sont

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  4.4164924  0.0151824   290.9   <2e-16 ***
```

```
tempsDep    -0.0186276  0.0001978   -94.2   <2e-16 ***
```

Par rapport à notre modèle $y = a + bx$ on trouve $a = 4.416$ (l'ordonnée à l'origine ou intercept) et $b = -0.018$ (pente), la valeur absolue de la pente est donc $\lambda = 0.018$. Sachant que les temps ont été mesurés en secondes, quelle est l'unité de λ ?

Dans les résultats affichés il y avait aussi une grandeur appelé `Multiple R-squared` : $R^2 = 0.9923$. Cette valeur indique la proportion de la variabilité totale dans y qui est expliquée par le modèle $y = a+bx$ et s'appelle de coefficient de détermination, c'est donc une mesure pour dire si le modèle linéaire explique bien nos données. Ici r^2 est très proche de 1, c'est donc très bien. Pour information : les `t value` et `Pr` (p-value) correspondent aux tests statistiques de l'hypothèse nulle que l'intercept et la pente soient égales à 0, mais dans notre contexte cela ne nous intéresse pas.

Dans cette sortie on trouve également une estimation des erreurs standards de a et b (Std. Error). Qu'est-ce que représente l'erreur standard dans ce contexte ?

Cependant, les formules utilisés dans **R** pour les calculer exigent que les valeurs de l'abscisse et de l'ordonnée ont été mesurées indépendemment, ce qui n'est pas le cas dans les courbes de survie où on construit l'ordonnée à partir des valeurs de l'abscisse. Cette reconstruction de y à partir de x rend aussi caduque les tests statistiques mentionnés ci-dessus. Il nous faut donc une autre méthode pour estimer l'erreur standard de la pente (taux de déposer le cadavre). Cette méthode est le **bootstrap**. Mais, comme elle n'est pas déjà programmé dans `Rcmdr`, il faudra la programmer nous-même via la ligne de commande et notre propre script. Vous pouvez donc quitter `Rcmdr`, mais sans quitter **R** (répondez 'non' aux questions si vous voulez enregistrer le script ou le fichier de sortie, sauf si vous voulez les garder pour vous ; ce sont des fichiers du type texte que vous pouvez regarder dans BlocNotes ou dans n'importe quel autre éditeur de texte).

2.3 Programmer dans R

R de base utilise les commandes que vous avez vues dans `Rcmdr` dans la `Fenêtre de script`. Toute l'interaction avec le logiciel se fait par une ligne de commande où vous tapez une commande et **R** vous rend les résultats correspondants à votre demande. Le seul problème est d'habitude de trouver la bonne syntaxe et d'éviter des fautes de frappe, mais dans notre cas cela sera assez simple. Tapez par exemple

```
> ls()                    # ls = 'lister' les variables actuellement disponible dans R
```

et vous verrez au moins les variables `depot16` (nos données) et `lambdaSurvie` (le résultat de notre régression linéaire). Vous pouvez afficher les valeurs de ces variables en simplement tapant leur nom,

```
> depot16                                                  # affiche les trois colonnes
> lambdaSurvie                                    # les résultats brutes, assez succincts
```

Avec la ligne de commande vous pouvez extraire des informations détaillées des variables, par exemple

```
> lambdaSurvie$coefficients[[2]]
```

affiche la valeur de la pente de la régression. Vous pouvez attribuer cette valeur a une nouvelle variable `penteOriginale` par la commande

```
> penteOriginale = lambdaSurvie$coefficients[[2]]
> penteOriginale
```

Les noms des variables doivent respecter les mêmes règles que les noms des colonnes dans les tableaux de données (voir premier TP).

2.3.1 Les vecteurs et leurs manipulations

Il nous faut d'abord une structure dans laquelle nous pouvons stocker plusieurs valeurs : un vecteur. Un vecteur peut se créer directement rempli avec les bonnes valeurs, par exemple la commande

```
> vec1 = c(5,4,1,4)                                        # c = combiner ou connecter
```

crée le vecteur `vec1` rempli des valeurs 5, 4, 1 et 4,

```
> vec1                                                    # afficher les valeurs de vec1
```
Vous pouvez afficher uniquement la 3ème valeur
```
> vec1[3]
```
ou même donner une nouvelle valeur au troisième élément
```
> vec1[3] = 2
> vec1
```
Un vecteur contient donc plusieurs chiffres et on y accède via la position (le nombre entre les crochets) de ces chiffres dans le vecteur, sachant que le comptage des positions commence à 1 dans la syntaxe de **R** (dans l'autres langues de programmation cela peut commencer à 0).

On peut également préparer un vecteur rempli de 0 pour l'utiliser plus tard,
```
> vec2 = rep(0,10)                                        # rep veut dire répéter
> vec2
```
qui rempli `vec2` de dix fois 0. Une autre commande permet de remplir un vecteur avec les chiffres de 1 à 20
```
> vec3 = 1:20                                  # remplir avec les chiffres 1, 2, 3, ...
> vec3
```

2.3.2 Les boucles

Les boucles permettent de faire la même chose (ou une chose similaire) plusieurs fois. Par exemple, la commande
```
> for (k in 1:10) { print(k) }
```
fait varier la variable k de 1 à 10 et l'affiche (`print`) chaque fois sur l'écran.

A partir de maintenant les commandes vont devenir plus longues et cela commence à être fastidieux de tout taper sur la ligne de commande. On va donc créer notre premier script qui contiendra les commandes et que nous exécuterons via un copier/coller dans la « console » de **R** (la fenêtre qui contient la ligne de commande). Ouvrez donc un nouveau script (`Fichier - nouveau script`), copiez là-dedans la définition de `vec2` et sauvez le script dans notre répertoire de travail habituel sur le bureau.

Ajoutez maintenant dans ce script les lignes suivantes :

```
for (k in 1:10) {
vec2[k] = k^2 # calculer k au carré
}
vec2
```

Qu'est-ce que font ces lignes ?

Vérifiez en faisant un copier/coller des commandes dans la console (sélectionner les commandes et appuyer sur `Ctrl - R`).

2.3.3 Une expérience virtuelle

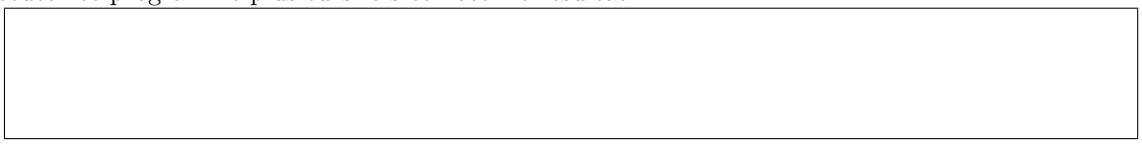
Pour nous familiariser avec les boucles on va faire une petite expérience virtuelle. Supposons que l'enzyme A à une probabilité de 0.4 de changer de configuration quand elle rencontre l'enzyme B. On veut explorer combien d'enzymes A ont changé de configuration après 20 rencontres entre une enzyme A et une enzyme B.

Pour décider à chaque rencontre si le changement de configuration à lieu ou pas on tire d'abord un chiffre aléatoire qui est distribué uniformément entre 0 e 1 et ensuite on regarde si ce chiffre aléatoire est < 0.4 (changement à lieu) ou > 0.4 (changement n'a pas lieu). La commande pour créer un tel chiffre aléatoire dans **R** est
```
> runif(1)                                                # générer 1 chiffre aléatoire
```
Créez plusieurs fois un chiffre aléatoire - est-il toujours entre 0 et 1 ? Pour visualiser qu'il a une distribution bien uniforme on va créer 10'000 chiffres et tracer leur histogramme,
```
> hist(runif(10000))
```

Maintenant on peut passer à notre expérience virtuelle. Recopiez les commandes suivantes dans votre script

```
nbReaction = 0
for (n in 1:20) {
  if (runif(1) < 0.4) {
     nbReaction = nbReaction + 1
  }
}
nbReaction
```

Comprenez-vous ce que font ces commandes ? Quelle chiffre y aura-t-il à la fin dans la variable `nbReaction` ? Exécutez ce programme plusieurs fois et notez le résultat

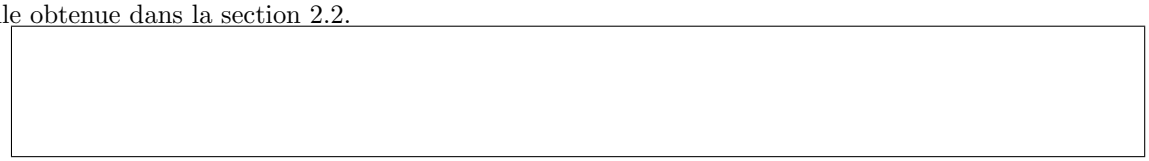


Voilà, c'est tout ce que nous devons savoir sur la programmation pour estimer l'erreur standard de λ par la méthode du bootstrap.

2.4 Le boostrap : une méthode générale pour estimer l'erreur standard

Un jeu de données bootstrap est un vecteur qui contient autant de valeurs que le jeu de données original et qu'on a tiré au hasard là-dedans (avec remise). Par exemple,
`> sample(c(2,5,1,3),replace=T)  # T veut dire TRUE, càd échantillonner avec remise`
tire au hasard avec remise 4 valeurs du jeu de données `(2,5,1,3)` (répétez la commande plusieurs fois : comprenez-vous ce qu'elle fait ?).

Pour créer un jeu de données bootstrap des temps avant dépôt il faut d'abord pouvoir accéder au vecteur de ces temps (qui se trouve être la première colonne du jeu de données `depot16` que vous avez déjà chargé dans **R**). Cette colonne s'appelle `tempsDep` (sauf si vous avez changé ce nom dans LibreOffice au début du TP), et on y accède par la commande
`> depot16$tempsDep`
Créez maintenant un jeu de données boostrap de ces temps par la commande
`> tempsDepBS = sample(depot16$tempsDep,replace=T)`
`> tempsDepBS`
Pour tracer la courbe de survie de ce jeu de données boostrap il faut encore les remettre en ordre croissant :
`> sort(tempsDepBS)`
Les valeurs de l'abscisse associées sont les mêmes que ceux que vous avez déjà dans `depot16`. Selon le nom que vous avez donné au logarithme néperien du nombre de fourmis qui transportent encore vous pouvez accéder à ces valeurs par une commande du type
`> depot16$rang.ln`
et on est prêt à faire la régression linéaire pour estimer la valeur de la pente
`> lambdaSurvieBS = lm(depot16$rang.ln ~ sort(tempsDepBS))`
`> summary(lambdaSurvieBS)`
et vous pouvez mettre la valeur absolue de la pente dans une variable nommée `penteBS`
`> penteBS = - lambdaSurvieBS$coefficients[[2]]`
Répétez cette procédure plusieurs fois, notez la valeur de la pente boostrap et comparez-la chaque fois à celle obtenue dans la section 2.2.



Pour estimer l'erreur standard par la méthode du bootstrap il suffit de répéter cette estimation du taux de dépôt pour un jeu de données boostrap 300 fois, de stocker les pentes (ou n'importe quel autre paramètre statistique dont on veut estimer l'erreur standard) dans un vecteur de taille 300 et de calculer

ensuite leur écart type (`sd(...)`). Le **théorème fondamental du bootstrap** dit que cet écart type est une estimation de l'erreur standard. Vous savez maintenant tout ce qu'il faut, programmez donc cette procédure dans **R** (dans un script) et notez les commandes et le résultat :

Calculez également l'intervalle de confiance à 95% de λ (notez commande et résultat. Connaissez-vous les deux méthodes principales pour calculer cette intervalle de confiance, l'une paramétrique et l'autre non-paramétrique ?)

2.5 Entrainement

Le fichier `depotSpont30.xls` contient le temps avant dépôt de cadavres des fourmis de la même espèce mais à une température de 30°C. Estimez le taux de dépôt λ et son erreur standard. Comparez à la valeur obtenue pour 16°C.

Chapitre 3

L'ANOVA appliquée sur un ou deux échantillons

Dans le dernier TP, vous avez mesuré vos propres temps de réaction sous différentes conditions expérimentales. On apprendra aujourd'hui comment analyser ce type de données dans **R**. On s'appuiera sur les données de 2009 qui sont déjà correctement organisées dans le fichier `tempsReaction2009.txt`, chargez-le dans `Rcmdr` (appelez de tableau de donnée « reaction ») et affichez les données pour vérifier que l'importation a correctement marché (noms des colonnes, type de variables). Comment s'appellent les colonnes et qu'est-ce qu'elles contiennent ?

Vous pouvez faire un résumé de chaque colonne par le menu `Statistiques -> Résumés -> Jeu de données actif`. Qu'est-ce qu'affiche ce résumé pour les colonnes du type « facteur » et pour les colonnes du type « chiffre » ? Comment est-ce que ces informations vous aident à vérifier si les chiffres avec décimaux ont correctement été importés ?

3.1 Est-ce que la distraction change votre temps de réaction ?

Regardons d'abord si la distraction change le temps de réaction (colonnes `simple.sans.dist` et `simple.avec.dist`). Il s'agit d'un test apparié, notez pourquoi :

Ceux qui ont compris l'introduction aux traitements statistiques du premier semestre savent déjà qu'on peut répondre avec un test de Student à données appariées (dans `R-cmdr` ces tests se trouvent dans le menu `Statistiques -> Moyennes`), et comme dans Minitab vous pourriez probablement correctement appliquer ce test. Mais dans cette UE on veut comprendre plus en détail ce qu'on fait.

Comme ce sont des données appariées, la valeur qui nous intéresse est la différence entre les deux temps de réaction (pour ensuite regarder si cette différence est significativement différente de 0). Calculez donc d'abord une nouvelle colonne au nom `simple.diff` qui représente la différence entre `simple.avec.dist` et `simple.sans.dist` (menu `Données -> Gérer les variables ... -> Calculer une nouvelle variable ...`, voir Fig 3.1). Faites maintenant une boite à moustaches de cette nouvelle variable : qu'est-ce que vous pensez, est-ce que la distraction a une effet significatif sur le temps de réaction ?

FIGURE 3.1 – Boîte à dialogue pour coder une nouvelle variable.

La prochaine étape est de « modéliser » ces différences par leur moyenne. Pour cela, passez par le dialogue `Statistiques -> Ajustement de modèles -> Modele linéaire...`, appelez le modèle `reaction-Distraction` et formulez le modèle `simp.diff ~ 1` (vous pouvez double cliquer sur une variable pour la copier dans la fenêtre active). Veuillez noter que le nom du modèle apparaît en haut à droite, c'est maintenant notre modèle active.

Pour continuer de façon paramétrique il faut vérifier si les résidus (c'est-à-dire la différence entre moyenne et valeurs observées) sont distribués de façon normale. Pour cela, copiez ces résidus dans une nouvelle colonne par le menu `Modèles -> Ajouter les statistiques des observations au jeu de données ...` et en cochant uniquement la case `Résidus`. Comment s'appelle la nouvelle colonne créée ?

On peut maintenant tester la normalité de cette colonne par un test de Shapiro-Wilk (menu `Statistiques -> Test de normalité de Shapiro-Wilk`). Quel est le résultat, a-t-on le droit de faire un test paramétrique ? Comment le rapportez-vous dans un rapport ?

Pour terminer l'analyse on peut maintenant déterminer la moyenne estimée et regarder si elle est significativement différente de 0. Regardez la sortie du menu `Modèles -> Résumer le modèle` : quelle est la valeur de la moyenne estimée ? et de l'erreur standard (es) ? et le nombre de degrés de liberté (dl) ?

Pour tester si la moyenne est significativement différente de 0 on utilise la moyenne, es et dl avec un test de Student (comparaison d'une moyenne à une moyenne théorique, ici 0). Ceci est également fait dans le résumé précédent. Rapportez correctement le résultat de ce test.

Calculez maintenant l'intervalle de confiance à 95% de la moyenne par le menu `Modèles -> Intervalles de confiance ...`. Il calcule les intervalles de confiance de tous les paramètres estimés, ici seulement la moyenne. Quel est le résultat ? Est-il cohérent avec le résultat du test ci-dessus ?

Remarque : Si on n'avait pas eu normalité des résidus on aurait pu essayer de transformer la variable dépendante (par exemple par une transformation log, voir section 3.5) et refaire les manipulations ci-

dessus avec la variable transformée. Enfin, si une non-normalité persiste, on peut tester notre hypothèse par un test non-paramétrique de Wilcoxon, menu `Statstiques -> Tests non paramétriques -> Test Wilcoxon apparié`.

3.2 Y a-t-il une différence entre les temps de réaction des hommes et des femmes ?

Posons maintenant la question si nos données indiquent une différence entre le temps de réaction (on prendra les données avec distraction) des ♀ et des ♂. Dans ce cas on veut comparer entre 2 échantillons indépendants. La première étape est de visualiser ces données : faites une boîte de dispersion en fonction du sexe. Est-ce que vous pensez qu'il y a une différence ?

D'après le cours du premier semestre ceci pourrait se faire soit par un test de Student (paramétrique) ou par un test de Mann-Whitney (non-paramétrique). Mais on passera de nouveau par un ajustement de modèle pour mieux comprendre ce qu'on fait. On modélisera en particulier le temps de réaction avec distraction en fonction du sexe de l'individu (`Statistiques -> Ajustement de modèles -> Modèle linéaire...`, appelez le modèle `reactionHF`). Ajoutez les résidus du modèle au jeu de données et testez leur normalité : quel est le résultat ? Si on n'a pas normalité, transformez la variable dépendante en log népérien, refaites le modèle et testez à nouveau la normalité. Allez-vous continuer avec les données originales ou les données transformées ?

Comme on compare maintenant entre deux échantillons il faut également s'assurer que les variances des deux échantillons sont égaux. Ceci se fait par le menu `Statistiques -> Variances -> Test de Bartlett` (on pourrait également prendre le test de Levene ou, comme on a deux échantillons, le test F de deux variances ...). Est-ce qu'on peut supposer que les deux variances sont égaux ? Rapportez correctement le résultat.

Après vérification de normalité et homoscédasticité on peut passer à une analyse paramétrique. Quelles sont les valeurs des deux moyennes et les erreurs standards associées (menu `Modèles -> Résumer le modèle`) ?

Vous pouvez vérifier votre lecture de la sortie en faisant un graphique des moyennes en fonction du sexe (menu `Graphes -> Graphe des moyennes...`, choisir votre variable dépendante et cocher `erreur standard`). Retrouvez-vous dans le graphique les valeurs des moyennes et erreurs standards ? Recopiez qualitativement ce graphique.

Demander si le sexe a une influence significatif sur le temps de réaction est équivalent à demander si

le facteur sexe explique une partie significatif de la variabilité totale. Autrement dit, la variabilité globale entre toutes les mesures est décomposée en variabilité qui peut être expliquée par le facteur sexe et une variabilité résiduelle qui n'est pas expliquée. C'est l'analyse de variance. On peut la faire avec le menu `Modèles -> Tests d'hypothèses -> Table de l'ANOVA` (assurez-vous que le modèle active est bien `reactionHF`). Laissez « Type II » coché (les nuances entre ces types dépasse le cadre de cette UE) et dites `OK`. Recopiez le résultat :

Reconnaissez-vous les éléments discutés en cours (somme des carrés des écarts, somme des carrés des écarts moyenne, degrés de libertés `Sexe` et `Residuals` ? Comment est-ce que la valeur de F est calculée ? Est-ce que le sexe explique une partie significatif de la variabilité globale ? Rapportez correctement ce résultat.

3.3 Quoi faire si on n'a pas normalité des résidus ou homoscédasticité ?

Essayez maintenant de faire exactement la même démarche pour regarder si le sexe influence le temps de réaction sans distraction. Est-ce que les résidus sont normales ?

Si les résidus ne le sont pas, refaites la même analyse avec la transformé en log népérien du temps de réaction sans distraction. Est-ce que c'est mieux ?

Avec ces données, rien à faire, on n'obtient pas la normalité des résidus. Une alternative à l'ANOVA à un facteur à deux modalités qu'on a fait ci-dessus est le test non-paramétrique de Wilcoxon-Mann-Whitney. Vous le trouvez par le menu `Statistiques -> Tests non paramétriques -> Test Wilcoxon bivarié...` : sélectionnez `simple.sans.dist` comme variable réponse, laissez toutes les autres options par défaut et cliquez sur `OK`. Qu'est-ce que vous en concluez ? Rapportez le résultat

et comparez-le avec le graphique des moyennes.

3.4 Entraînement

Prenez les temps de réaction de votre promo [1] et testez

1. si la distraction influence le temps de réaction,
2. si le sexe influence le temps de réaction sans ou avec distraction.

Tous les tests sont à faire avec $\alpha = 0.05$ et les résultats doivent être rapportés correctement comme dans ce TP. Essayez divers transformations avant d'abandonner l'idée de faire un test paramétrique. Mettez-vous en binôme et faites un rapport technique (2 pages maximum) de cette analyse. Ce rapport est à rendre dans une semaine.

Autres entrainements :

1. Testez dans le jeu de données « Temps de réaction » si le temps de réaction est différent selon la nature du stimulus complexe (couleur, colonne `complexe.coul.sans.dist`, ou forme, colonne `complexe.forme.sans.dist`). Ensuite, on dit souvent que les femmes ont une meilleure perception des couleurs, testez donc si pour le traitement `complex.coul.sans.dist` on trouve une différence du temps de réaction entre hommes et femmes.
2. **Comparaison à une moyenne connue.** Nous reprenons l'exemple du cours qui voulait tester si la température des crabes correspond la température ambiante ou si elle est différente. Les mesures dans les crabes ont été pris à une température ambiante de 24.3°C. La question est : est-elle significativement différente de 24.3°C ? Les données sont dans le fichier `crabes.txt`. Lisez-le et mettez les données dans le vecteur 'crabes'. Calculez d'abord les statistiques de base (moyenne, écart type, erreur standard) et visualisez-les sous forme d'un histogramme. Testez la normalité des données et choisissez le test de comparaison (avec 24.3 °C) adapté.
3. **L'effet d'un nouveau engrais sur la croissance des plantes** Dans l'exemple suivant, on a mesuré la hauteur de plantes cultivées avec un engrais habituel et la hauteur de plantes cultivées avec un nouveau super engrais.
   `engraisActuel : 48.2, 54.6, 58.3, 47.8, 51.4, 52.0, 55.2, 49.1, 49.9, 52.6`
   `engraisNeuf : 52.3, 57.4, 55.6, 53.2, 61.3, 58.0, 59.8, 54.8`
   Préparez ces données dans un tableur et passez-les à Rcmdr pour tester si le super engrais fait pousser plus haut. Faites un graphique en boîte à moustache où on voit la qualité de l'estimation de la moyenne. Après avoir vérifié les hypothèses de base, est-ce que le super engrais est vraiment super ? Résumer comment vous avez procédé et le résultat du test statistique.
4. **Comparaison entre traffic de fourmis sur un pont de 10mm et sur un pont de 3mm.** Les jeux de données suivants représentent des mesures de flux de fourmis sur un pont de largeur 10mm ou de largeur 3mm. On veut tester s'ils sont significativement différents.
   `pont10 : 54.1, 88.1, 140.25, 88.05, 63.25, 72.0, 56.65, 41.85, 123.15, 82.85, 70.15, 47.3, 58.7, 97.25, 60.6`
   `pont6 : 51.4, 69.6, 122.8, 76.75, 76.25, 105.8, 71.8, 78.6, 54.5, 43.5, 51.25, 74.85, 33.7, 66.15, 51.8`
   Testez la normalité et l'hétéroscédasticité de ces deux jeux de données. Est-ce que vous avez le droit d'appliquer un test paramétrique ? Est-ce que les flux sont différents entre les deux ponts ?

3.5 Informations supplémentaires : transformation de données

Remarque : au lieu d'utiliser un test non-paramétrique, on essaye souvent de transformer les données pour les rendre normales (et pouvoir ainsi appliquer un `t.test()` sur les données transformées). Les transformations habituelles sont : $\log(X+1)$ (données lognormales, par exemple estimations d'effectifs), $\sqrt{X+0.5}$ (données Poissonniennes) ou $\arcsin(\sqrt{X})$ (pour des proportions, ne s'applique donc pas à nos données). Essayez si une de ces transformations rend le jeu de données du temps avant dépôt normales.

Pourquoi est-ce qu'on ne se simplifie pas la vie en faisant uniquement des tests non-paramétriques ?

1. Veuillez notez que j'ai enlevé une valeur atypique qui était très différente des autres valeurs. L'idée d'enlever une valeur n'est pas de tricher, mais c'est un choix entre décrire 100% des mesures mais avec des très grandes variances qui risquent de masquer tout facteur avec influence, ou de n'expliquer que $21/22 = 95.5\%$ des données mais celles-là avec une meilleure précision. Tant que c'est correctement rapporté il n'y a pas de problème.

Chapitre 4

Les analyses de la variance (ANOVA)

Les ANOVA (ou **AN**alysis **O**f **VA**riance) généralisent le test t de Student aux cas où il y a plus de 2 moyennes à comparer (au lieu de 2 dans le t-test). Dans ce cas là, il n'est pas légitime d'effectuer plusieurs tests t (par exemple pour 3 moyennes A, B et C, comparer la moyenne A à la moyenne B, puis la B à la C et la A à la C) car on se donne plusieurs fois la possibilité de commettre une erreur de type I (la probabilité de rejeter H0 à tort). Le risque global de commettre au moins une fois cette erreur est donc bien plus important que le $\alpha = 5\%$ que l'on fixe ordinairement (dans notre exemple avec trois échantillons : $1 - (1 - 0.05)^3 = 0.14$). Avez-vous bien vu l'ampleur de ce problème ? Dessinez le tableau fondamental des erreurs qu'on peut commettre avec les tests d'hypothèses :

L'ANOVA permet de contourner ce problème en faisant un test global. En situation non paramétrique (c'est à dire sur des données quantitatives absolument pas gaussiennes, non transformables et hétéroscédastiques) il existe un équivalent à l'ANOVA : le test de Kruskal-Wallis.

Pendant le TP, vous devrez remplir toutes les cases dans ce poly. Pour chaque test d'hypothèse veuillez notez en particulier l'hypothèse nulle (H0), la valeur de p (avec les statistiques associées) que vous rend le logiciel, et l'interprétation. Tout comme le test de Student, l'ANOVA suppose la normalité de vos échantillons. Plus important encore, il faut que les variances soient égales (test de Bartlett ou de Levene). N'oubliez pas lors de ce TP de tester ces hypothèses (voir le 2nd TP) et de mentionner le résultat dans les cases. A l'avenir, n'oubliez pas de les faire lorsque vous travaillerez avec vos propres données !

4.1 Ouvrir ou sauver un espace de travail

Démarrez **R** mais sans lancer `Rcmdr` (on le fera plus tard). Les données pour ce TP sont déjà préparées dans ce qu'on appelle un « espace de travail ». Avant de le charger vérifiez que votre espace de travail actuel est vide par la commande (à entrer dans la console de **R**)

```
> ls()                    # afficher toutes les variables dans votre espace de travail
```

Si vous voyez pleins de noms apparaître cela veut dire que vos prédécesseurs n'ont pas fait du ménage. Faites-le par la commande

```
> rm(list=ls())                              # vider l'espace de travail, rm = remove
> ls()                                             # plus rien ne devrait s'afficher
```

Télécharger de mon site web le fichier `tp-anova-M1.rda` et chargez-le ensuite dans **R** (par `Rcmdr`) par le menu `Fichier - Charger l'environnement de travail ...`. Vérifiez par la commande `ls()` si vous disposez bien des objets `engrais, moutons, plasmaData, sourisA` et `sourisB`. Ce fichier, `tp-anova-M1.rda`, contient les données en format binaire facilement chargeable par **R**, et les espaces

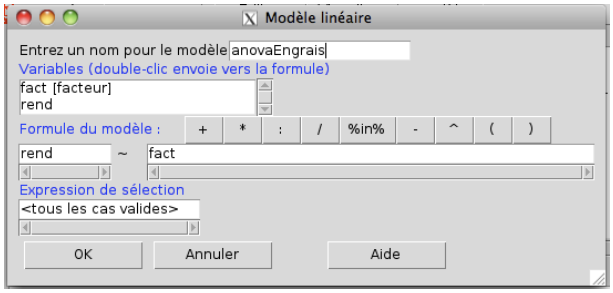


FIGURE 4.1 – Boîte de dialogue pour faire une ANOVA à un facteur en passant par un modèle linéaire.

ou environnements de travail sont un moyen simple pour sauvegarder des données ou analyses dans **R**. Si après un certain temps de travail sous **R** ou `Rcmdr` vous voulez sauver votre espace de travail (donc avec tous les fichiers que vous avez pu lire et toutes les variables que vous avez peut-être créées) vous pouvez le faire par le menu `Fichier - Sauver l'environnement de travail ...`. Il est suggéré de donner l'extension `.rda` ou `.RData` au nom du fichier sauvé afin d'indiquer qu'il s'agit d'un espace de travail de **R**. Ceci vous permet de transférer vos données sur votre ordinateur personnel ou de les envoyer à un collègue.

Lancez maintenant `Rcmdr` par la commande habituelle.

4.2 ANOVA à un facteur

Avant de commencer, veuillez noter que les données sont organisées par variable (comme on l'a appris au premier TP) : ce n'est pas intuitif (vous ne noteriez pas vos données comme cela dans un tableur) mais les logiciels de statistique fonctionnent de cette façon. Ainsi, le jeux de données `engrais` (activez-le dans `Rcmdr` en cliquant sur la fenêtre en rouge qui affiche au début `Pas de données` et en le sélectionnant, cliquez ensuite sur le bouton `visualiser`) contient 2 colonnes, une première `rend` qui contient le rendement des parcelles, et une colonne `fact` (facteur) qui indique l'engrais utilisé (pour **R**, le premier est du type 'numeric' et le deuxième du type 'factor', on reviendra sur cette subtilité). Faites un graphique de ces données (rendement en fonction du type d'engrais) en `Boîte de dispersion...` ou `Graphe de moyennes...`. Commentez l'effet de l'engrais sur le rendement :

Pour commencer, nous allons chercher à savoir si le rendement `rend` varie effectivement en fonction de l'engrais `fact` ou si la variabilité observée peut simplement être expliquée par le hasard qui joue à chaque mesure. Il y a 4 engrais, donc 4 groupes à comparer : c'est bien une situation d'ANOVA à 1 facteur. Rappelons que les hypothèses de base pour l'utilisation d'une ANOVA sont l'homoscédasticité (égalité des variances) et la normalité des données. Nous testerons la première tout de suite et la seconde à posteriori. Vérifiez l'homoscédasticité par le menu `Statistiques - Variances - Test de Levene`. Quelle est votre conclusion ?

Nous pouvons passer maintenant aux choses sérieuses et faire notre ANOVA. On procédera exactement comme dans le dernier TP en faisant un modèle, `Statistiques -> Ajustement de modèle -> modèle linéaire...`. Appelez le modèle `anovaEngrais`.

Regardons d'abord les commandes éxécutées par **R** :

```
> anovaEngrais <- lm(rend   fact, data=engrais)
> summary(anovaEngrais)
```

La première fait les calculs de base selon le modèle statistiques « rendement en fonction du type d'engrais » (le caractère ~ (qui se lit tilde) sert juste à indiquer cette relation, « en fonction de ... ») et met le résultat dans la variable `anovaEngrais`. La seconde commande fait le résumé typique d'un modèle

linéaire, avec `Intercept` l'estimation de la moyenne du rendement pour l'engrais dans le nom est premier dans l'alphabet (`e1`) et les chiffres suivants la différence entre la moyenne estimée pour chaque type d'engrais et `e1`. Ce n'est pas exactement ce qui nous intéresse, mais on tout cas il faut d'abord vérifier la normalité des résidus. Procédez comme au dernier TP et rapportez le résultat. Est-ce qu'on a le droit de procéder avec une ANOVA ? Faites également un histogramme de ces résidus.

Les calculs pour l'ANOVA se font pas le menu `Modèles -> Tests d'hypothèses -> Table de l'ANOVA...` (choisissez les options par défaut, type II).

```
Response: rend
          Sum Sq Df F value   Pr(>F)
fact      234.75  3  6.9556 0.005755 **
Residuals 135.00 12
```

Quelle est l'interprétation de ces chiffres ? Comment est-ce que la `F value` est calculée ?

4.3 Tests post-hoc

Pour regarder entre lesquelles des types d'engrais il y a une différence significative **R** sait faire le test « honestly significant difference » de Tukey. Il suffit de faire une ANOVA en passant par le menu `Statistiques -> Moyennes -> ANOVA à un facteur...` et de cocher la case `comparaisons multiples des moyennes` (Fig 4.2). Le résultat,

```
Linear Hypotheses:
             Estimate lwr      upr
e2 - e1 == 0  -8.0000 -15.2634  -0.7366
e3 - e1 == 0  -9.0000 -15.6718  -2.3282
e4 - e1 == 0  -2.5000  -9.1718   4.1718
e3 - e2 == 0  -1.0000  -8.5962   6.5962
e4 - e2 == 0   5.5000  -2.0962  13.0962
e4 - e3 == 0   6.5000  -0.5327  13.5327
```

est donné en forme de l'intervalle de confiance de la différence entre chaque couple de moyennes, ce qui est suffisant pour conclure (si l'IC n'inclut pas 0, la différence est significative). Entre lesquelles des moyennes est-ce qu'il y a une différence significative ? Retracez la Boîte de dispersion et indiquez les populations statistiques par des lettres.

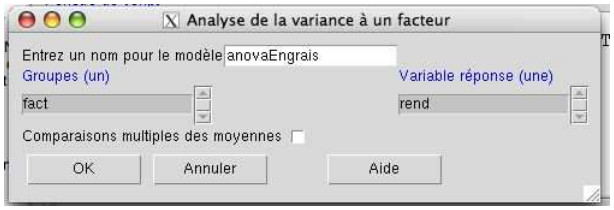


FIGURE 4.2 – Boîte de dialogue pour faire une ANOVA à un facteur avec un post-hoc.

4.4 Alternative non-paramétrique à l'ANOVA

Dans le cas de l'ANOVA à un facteur, si on a un problème avec la normalité des résidus ou l'homoscédasticité et si aucune transformation des données ne permet de remédier au problème, on a une alternative non-paramétrique : le test de Kruskal-Wallis. Effectuez-le sur le jeu de données des engrais (menu `Statistiques -> Tests non paramétriques -> Test Kruskal-Wallis`). Quel est le résultat ?

Comparez le p estimé avec celui estimé par l'ANOVA. Qu'est-ce que vous pouvez dire sur la puissance du test de Kruskal-Wallis dans ce cas où on avait le droit de faire un test paramétrique ?

4.5 ANOVA à deux facteurs

Le jeux de données `moutons` (voir premier TP) contient les poids de moutons mérinos d'Arles. Un premier lot de mâles et de femelles a été mesuré en hiver, un second en été. Activez-le dans `Rcmdr` et visualisez-le : il y a trois colonnes, `poids`, `sexe` et `saison`.

On veut d'abord tester l'homoscédasticité des échantillons. Malheureusement `Rcmdr` n'a pas prévu le cas d'une ANOVA à 2 facteurs, mais il suffit de faire le test de Levene d'après un seul facteur (par exemple `sexe`), de modifier la commande dans la fenêtre de script

```
> levene.test(poids ~ sexe*saison)
```

et de resoumettre cette commande (on passe de nouveau par une formule, le poids on fonction des facteurs croisés `saison` et `sexe`). Quel est le résultat ?

A-t-on le droit de procéder avec une ANOVA ? Sinon, créez une nouvelle variable avec le poids transformé log (appelez-la `log.poids`) et refaites le test avec cette variable. Résultat ?

On veut maintenant tester l'effet des deux facteurs `sexe` et `saison` et leur interaction sur le poids des moutons. Passez par le menu `Statistiques -> Ajustement de modèles -> Modèle linéaire...`, appelez le modèle `anovaMoutons` et modélisez `log.poids ~ saison*sexe` (Figure 4.3). Testez ensuite si les résidus sont distribués de façon normale. Quelle est le résultat ? A-t-on le droit de procéder ?

Passez maintenant à l'analyse ANOVA par le menu `Modèles -> Tests d'hypothèses -> Table de l'ANOVA` (type II, comme dans le dernier TP).

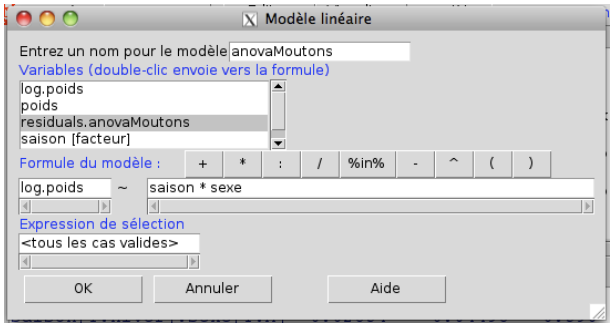


FIGURE 4.3 – Boîte de dialogue pour faire le modèle d'une ANOVA à deux facteurs

```
Response: log.poids
             Sum Sq  Df F value    Pr(>F)
saison      0.12748   1  8.0169  0.005422 **
sexe        1.37369   1 86.3900 7.256e-16 ***
saison:sexe 0.00554   1  0.3481  0.556282
Residuals   1.93993 122
```

Notez et interprétez ces résultats. Comment est-ce que les valeurs de F se calculent ? Y a-t-il un effet `sexe` ? Effet `saison` ? Interaction entre `sexe` et `saison` ?

4.6 ANOVA avec mesures répétées (ANOVA appariée)

Prenons maintenant les données `sourisA`. Il s'agit d'un jeu de données dans lequel 16 souris ont été traitées avec NaCl (7) ou AP5 (9), et ensuite soumises à 4 séances d'entraînement successives. Regardons maintenant ce jeux de données (visualiser).

```
trmt  sujet   seance  explore
NaCl  1       s1      65
   ...     ...       ...
NaCl  7       s1      83
AP5   8       s1      75
   ...     ...       ...
AP5   16       s1      74
NaCl  1        s2      30
   ...     ...       ...
NaCl  16       s4      28
   ...     ...       ...
AP5   16       s4      52
```

Il y a eu en total 64 mesures, mais comment interpréter la représentation ? Explication : on va comparer les performances des animaux sur les séances `s1`, `s2`, `s3` et `s4`. On introduit donc un facteur `seance` qui contient simplement le numéro de la séance pour chaque performance. De plus, comme chaque souris est testée à chaque séance (donc 4 fois), il faut ajouter un facteur `sujet` qui identifie de quelle souris il s'agit. Enfin, le facteur `trmt` contient le traitement des souris. La variable dépendante que l'on teste sera `explore`, qui contient le temps que chaque souris passe à explorer un nouvel objet. Chaque souris

apparaîtra donc 4 fois dans le facteur `trmt`, car on a pour chacune d'elle 4 valeurs correspondantes dans `seance` (puisque chacune est testée dans chaque séance).

Essayez maintenant de faire l'ANOVA à deux facteurs comme ci-dessus en appelant le résultat `anovaSourisA1`. Les commandes pertinentes sont

```
anovaSourisA2 <- (lm(explore ~ seance*trmt, data=sourisA))
Anova(anovaSourisA1)
```

mais rien là-dedans indique que ce sont des mesures répétées. Effectivement, le cas de l'ANOVA à mesures répétées n'a pas été prévu dans `Rcmdr`. Cependant, on peut tester la normalité des résidus : procédez comme ci-dessus et notez le résultat et si on a le droit de procéder.

faite une copie du modèle et modifiez la copie comme ci-dessous

```
anovaSourisA2 <- aov(explore ~ seance*trmt + Error(sujet), data=sourisA)
summary(anovaSourisA2)
```

La sous-fonction `Error` est utilisée dans le calcul pour faire comprendre à **R** qu'on est en situation appariée, mieux vaut la traiter comme une « boîte noire ». Soumettez ces commandes et interprétez les résultats (effet `trmt` ? `seance` (apprentissage) ? Interaction entre les deux ?)

Faites un graphique des moyennes (avec les erreurs standards). Quel est le lien entre ce graphique et les effets détectés ?

Quid de l'homoscédasticité qu'on avait toujours testé dans les ANOVA a 1 et 2 facteurs ? Dans une ANOVA à mesures répétés il faut avoir homoscédasticité entre les différences des sessions deux par deux. Le test pour cela et le test de Mauchly. Pour l'appliquer il faut réorganiser nos données en forme d'une matrice où les 4 mesures d'un même individu se trouvent sur une ligne. La commande pour cela est la suivante (vous pouvez la saisir dans la fenêtre de script ou dans la console de **R**) :

```
> sourisMat = matrix(sourisA$explore,nrow=16,byrow=F)                  # réorganisation
> sourisMat                                        # afficher la matrice pour la vérifier
> mauchly.test(lm(sourisMat ~ 1))                                                  # test
```

Quel est le résultat ? Est-ce qu'on a eu le droit de faire une ANOVA à mesures répétées ?

4.7 Entrainement

Le jeu de données `plasmaData.rda`[1] contient les concentrations de Ca^{++} dans le plasma (colonne `plasma`) en fonction d'un traitement hormonal (colonne `treat`, NH = pas de traitement, HT = traitement avec l'hormone) et du sexe des oiseaux étudiés (colonne `sexe`). Testez s'il y a un effet du traitement hormonal, du sexe ou une interaction entre ces deux facteurs.

`sourisB` contient de nouveau des séances d'exploration de souris, comme `sourisA`. Faites une ANOVA à mesures répétées. Quel est le résultat ? Faites aussi le graphique des moyennes pour interpréter les effets.

1. Ce jeu de données est extrait de l'exemple 12.1 dans « Zar, J. H. (1999). Biostatistical analysis. Prentice Hall, New Jersey, 4th edition ».

Annexe A

Petit résumé de l'Analyse de variance

Une ANOVA teste l'hypothèse nulle de l'égalité entre g moyennes

$$\text{H0} : \mu_1 = \mu_2 = \ldots = \mu_g$$

Pour cela on regarde la variabilité dans les données. Comment la quantifier ? Par exemple, entre la moyenne globale $\bar{x}$ et les mesures $x_i,\ i = 1, \ldots, n$ (voir Fig. A.2) ? On pourrait regarder la différence $x_i - \bar{x}$, mais cette différence étant tantôt négative, tantôt positive, la somme serait proche de 0. On regarde donc plutôt la somme des différences au carré (ou la somme des carrés des écarts, notée $SS =$ « sum of squares ») $(x_i - \bar{x})^2$.

Cette somme des carrés des écarts ($SS = \sum (x_i - \bar{x})^2$) est d'autant plus grande qu'il y a plus de données. Pour quantifier la variabilité ajoutée en moyenne par chaque donnée supplémentaire il faut donc la diviser par le nombre de données. Cependant, comme on a calculé la moyenne à partir des n mesures on n'a en réalité que $n-1$ écarts indépendants. On appelle $df = n-1$ les degrés de liberté, et on calcule la moyenne des carrés des écarts,

$$MS = \frac{\sum_{i=1}^{n} (x_i - \bar{x})^2}{n-1}$$

avec $MS =$ « mean squares ». Dans une ANOVA on garde toujours les deux informations, SS et df.

Regardons un exemple (Fig. A.2), le rendement des pommes de terre en fonction de $g = 4$ différents types d'engrais. Pour chaque type d'engrais on a fait entre 3 et 5 replications, récuperant au total $n = 16$ mesures de rendement. Qu'est-ce qui cause la variabilité entre ces mesures ? Cela peu être due au type d'engrais, mais aussi bien dues aux charactéristiques des sols, les taux d'humidité ou d'autres facteurs environnementaux, et c'est justement grâce à la réplication qu'on peut séparer la variabilité due à l'engrais de toutes les autres sources de variabilité. On divise donc la variabilité entre

- variabilité due au type d'engrais
- variabilité due à tous les autres facteurs

On peut voir cette division dans la Fig. A.2. En haut il y a la variabilité totale (somme des carrés des écarts totale, SST). Au milieu on voit la variabilité due à tous les autres facteurs (après avoir enlevé la variabilité due à l'engrais), appelée somme des carrés des erreurs ou résiduelle (SSE). Enfin, en bas on voit la variabilité due à l'engrais, la somme des carrés des écarts inter-groupe, SSG). Entre les trois il y a la relation

$$SST = SSG + SSE.$$

Comment calculer les divers carrés des écarts moyens ? On avait déja vu qu'au total, on a $df_T = n - 1$ degrés de liberté, donc la somme totale des carrés des écarts (TMS=« total mean squares ») se calcule comme :

$$TMS = \frac{SST}{n-1}.$$

Au niveau groupe on compare quatre moyennes avec une moyenne globale, on a donc $df_G = 3 = g - 1$ degrés de liberté, et la somme des carrées des écarts entre la moyenne de chaque groupe et la moyenne

totale (GMP=« group mean squares ») est :

$$GMS = \frac{SSG}{g-1}.$$

Enfin, la moyenne des carrés des écarts entre chaque mesure et la moyenne de son groupe (SSE=« error mean squares ») a $df_E = 12 = n - g$ degrés de liberté et elle est :

$$EMS = \frac{SSE}{n-g}.$$

S'il n'y a pas d'effet engrais on devrait trouver $GMS = EMS$, ou $F = GMS/EMS = 1$. Si $H0$ est fausse, on aura $GMS > EMS$ ou $F = GMS/EMS > 1$. On appelle F le **F-ratio**, plus F est grand, plus les inégalités entre groupes sont grandes par rapport aux inégalités entre les réplications au sein d'un groupe. Tout dépend donc si $F = GMS/EMS$ est significativement plus grand que 1. Si $H0$ est vraie, le rapport GMS/EMS est distribué selon une loi de Fisher avec $3 = g - 1$ et $12 = n - g$ degrés de liberté (voir Fig. A.1) [1]. La valeur seuil $F_{3,12,0.95}$ telle que seulement 0.05% de F-ratio dépassent ce seuil si $H0$ est vraie est de 3.49. Si le F calculé avec les données dépasse ce seuil on peut rejeter $H0$ au risque $\alpha = 0.05$ de se tromper, c'est-à-dire avec 5% de risque de conclure que l'engrais a un effet alors qu'en réalité il n'en a pas.

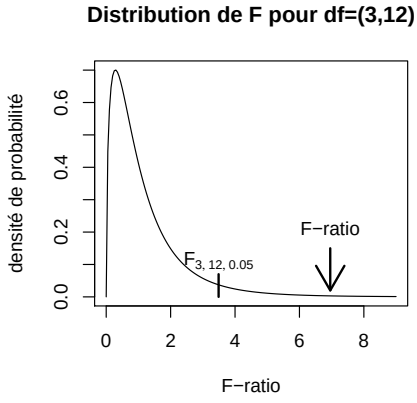


FIGURE A.1 – La distribution de Fisher pour 3 et 12 degrés de liberté. $F_{3,12,0.05}$ indique la valeur seuil de significativité, F-ratio la valeur calculée pour l'exemple du rendement en fonction du type d'engrais.

Dans notre exemple, on trouve (calculé par **R**)

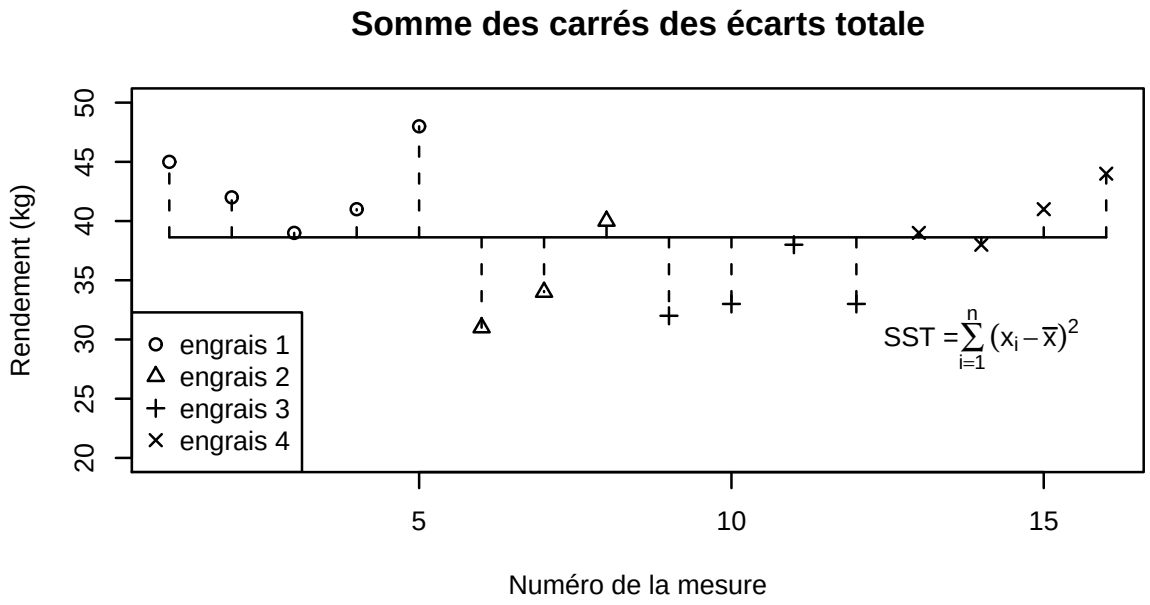
```
            Df Sum Sq Mean Sq F value   Pr(>F)
fact         3 234.75   78.25  6.9556 0.005755 **
Residuals   12 135.00   11.25
```

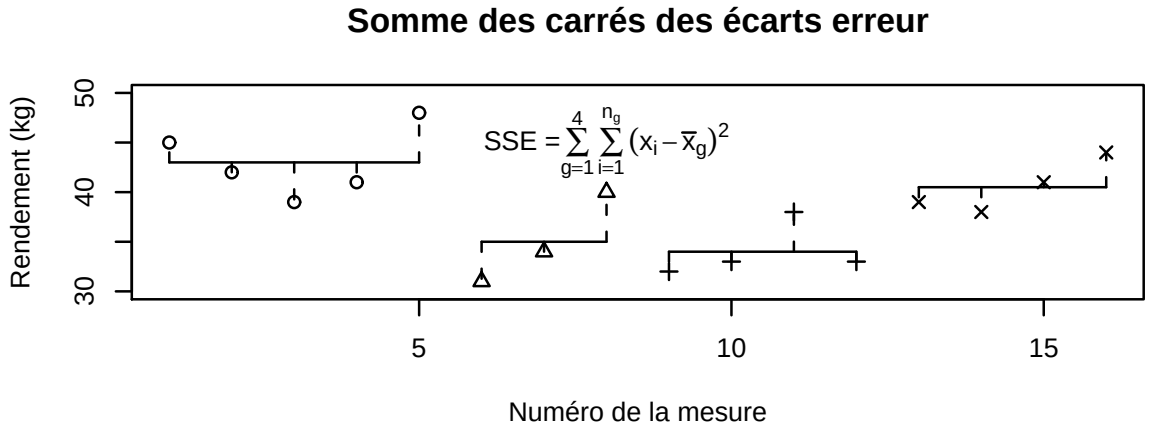
Ce tableau résume sur la première ligne les calculs par rapport à la variabilité due à l'engrais (inter-groupe) avec `Df` degrés de liberté, somme des carrés des écarts SSG (`Sum Sq`) et la moyenne des carrés des écarts GMS (`Mean Sq`). La seconde ligne donne les même informations pour la variabilité due à toutes les autres sources que l'analyse considère comme du bruit, SSE et EMS. La "`F value`" représente le F-ratio (voir Fig. A.1) et le "`Pr(>F)`" la valeur de p (probabilité de se tromper si on rejette $H0$). Dans le cas présent, on a F-ratio $> F_{3,12,0.95}$ ce qui est équivalent à $p < \alpha$; on peut donc rejeter $H0$ et dire que l'engrais a effectivement un effet sur le rendement.

On peut aussi dire que $\frac{\text{SSE}}{\text{SST}}$ est la variabilité non-expliquée par le facteur `engrais`, ou, de façon équivalente, que $R^2 = 1 - \frac{\text{SSE}}{\text{SST}}$ est la variabilité expliquée. On appelle R^2 le coefficient de détermination.

Pour mettre ce résultat dans un rapport on dit « Un test ANOVA nous a permis de mettre en évidence un effet significatif de l'engrais sur le rendement des pommes de terre ($F = 6.956, df = (3, 12), p = 0.0058$). Le facteur engrais explique $R^2 = 0.63$ (63 %) de la variabilité globale des rendements ».

1. Attention, pour que cela soit vrai il faut que les résidus (carrés des écarts) soient distribués selon une Gaussienne et que les variances entre les groupes soient égales (homoscédasticité), sinon les F seuils calculés risquent d'être faux. Cependant, l'ANOVA est assez robuste vis à vis des problèmes de non-normalité ou d'hétéroscédasticité, c'est-à-dire dans le cas de résultats hautement significatifs une légère non-normalité ou hétéroscédasticité ne sera pas trop grave.





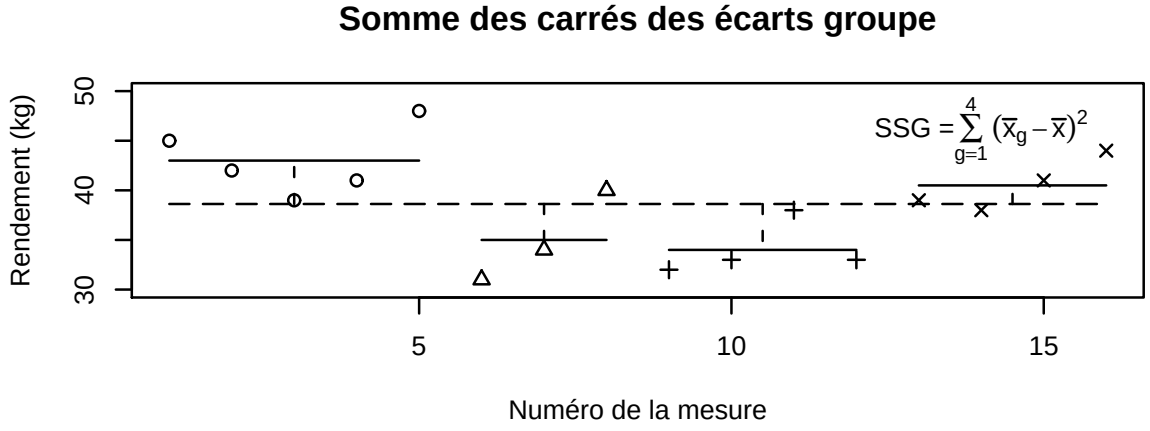


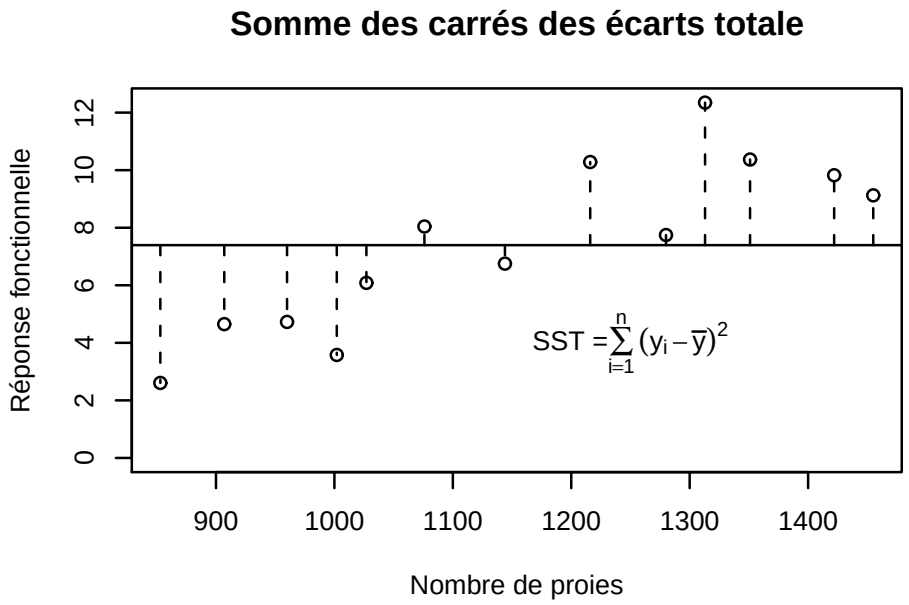
FIGURE A.2 – Les trois types de *somme des carrés des écarts* d'une ANOVA (totale, erreur et groupe) pour l'exemple du rendement en fonction du type d'engrais. Le modèle explique $R^2 = 1 - \frac{SSE}{SST} = 0.63$ de la variance totale des données.

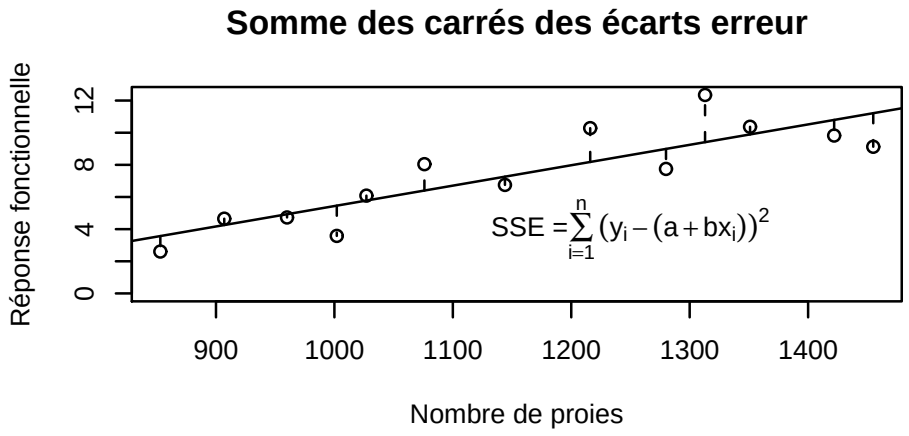
A.1 Le lien avec la régression linéaire

Il est intéressant de noter que le même formalisme peut s'appliquer à la régression linéaire. Prenons l'exemple de l'interaction proie-prédateur qui existe entre les loups et les orignaux sur l'Ile Royale dans le lac Michigan. Des chercheurs ont mesuré chaque année la densité d'orignaux, la densité de loups, et leur réponse fonctionnelle (nombre d'orignaux mangés par loup et par an). Ils ont essayé de modéliser cette réponse fonctionnelle (y) comme une fonction linéaire des proies disponibles (x). La Fig. A.3 montre ces données (pour simplifier le graphique, seulement environ deux tiers des données ont été utilisées) par rapport à la moyenne globale de la réponse fonctionnelle (en haut, SST), par rapport au modèle linéaire $y = a + bx$ (au milieu, SSE) et la variabilité totale expliquée par le modèle (en bas, SSG).

Pour tester si cette relation est significative on calcule classiquement la pentre b avec son erreur standard et on teste par un test de Student si la pente est significativement différente de 0 (ici on a $b = 0.012 \pm 0.002$, $t = 5.54$, $df = 11$ et $p = 0.00017$). Ceci est équivalent à faire une analyse ANOVA et tester si le facteur `proie` explique une variabilité significative de la réponse fonctionnelle (ici, $F = 77.58$, $df = (1, 11)$, $p = 0.00017$).

Cette équivalence du formalisme entre ANOVA (variable dépendante VD continue en fonction d'une variable indépendante VI catégorielle) et régression linéaire (VD continue en fonction d'une VI continue) permet de faire des analyses plus complexes où on essaye d'expliquer une VD continue en fonction d'un mélange de VD catégorielles ou continues, les **G**eneral **L**inear **M**odels (**GLM**).





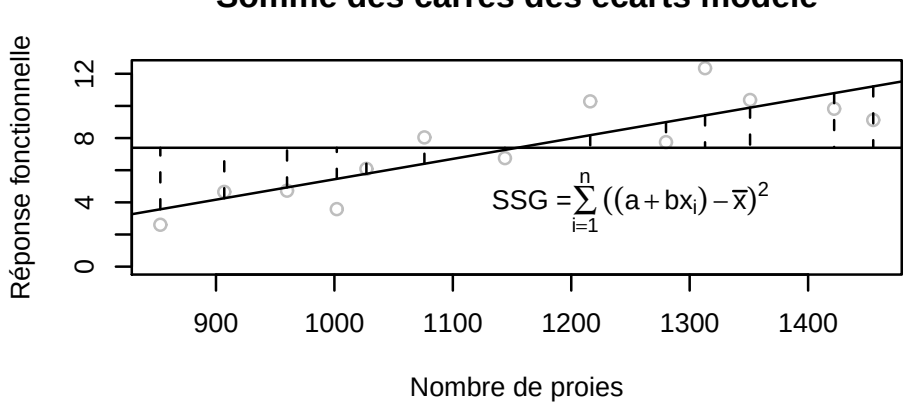


FIGURE A.3 – Les trois types de *somme des carrés des écarts* d'une régression linéaire (totale, erreur et groupe) pour l'exemple de la prédation (réponse fonctionnelle) des loups sur l'Ile Royale en fonction de la densité de proies (orignaux) disponibles. Le modèle explique $R^2 = 1 - \frac{SSE}{SST} = 0.74$ de la variance totale.