Aide - mémoire de statistique appliquée à la biologie

– Construire son étude et analyser les résultats à l'aide du logiciel **R** *–*

Maxime HERVE

3ème version
2011
(1ère version 2010)

Avant-propos

Lors de mon stage de Master 2, j'ai réalisé une chose importante : en biologie, les statistiques sont essentielles. J'ai aussi réalisé une autre chose importante : je n'y connaissais rien. Et par dessus tout, j'en avais horreur. Mais étant obligé d'en passer par là, je m'y suis mis. Et j'ai alors réalisé une chose encore plus importante : il n'y a pas besoin d'être statisticien pour analyser ses données. Il faut simplement savoir se poser un peu, réfléchir à l'étude qu'on a mené (ou mieux, à l'étude qu'on va mener), et être rigoureux. Pour le reste, tout est disponible dans les livres ou sur internet.

J'ai donc décidé de me former dans mon coin à la statistique appliquée à la biologie. Je me suis alors confronté à un problème qui m'a fait perdre beaucoup de temps, un temps que la plupart des stagiaires n'ont pas : il existe de très nombreux et très bons documents sur le sujet, mais très peu qui regroupent les analyses de base, celles que l'on rencontre le plus souvent. Afin de ne pas oublier tout ce que j'avais appris par - ci, par - là, j'ai donc voulu me rédiger un petit document de synthèse. Finalement, j'ai décidé d'en faire un véritable aide - mémoire et de le mettre à la disposition des autres étudiants. L'objectif (ambitieux) de cet ouvrage est donc d'être pour vous le guide que j'aurais aimé avoir lors de mon stage.

Utiliser cet aide - mémoire ne demande que très peu de connaissances en statistiques. Savoir ce que sont une moyenne, une variance, une médiane ou un intervalle de confiance est suffisant. Par contre, il exige une chose : si vous voulez qu'il vous prenne par la main, il faut que vous en ayez envie. J'entends par là qu'il est indispensable de se poser des questions : à quelle question mon étude doit - elle répondre ? Quel dispositif vais - je mettre en place pour y répondre ? Que vais - je contrôler, que vais - je observer dans l'étude ? Comment vais - je utiliser mes résultats ? Si l'on prend le temps de se poser ces questions, et surtout le temps d'y apporter une réponse, analyser ses données n'est pas compliqué. Vous verrez même qu'il est très agréable de comprendre ce que l'on fait, et pourquoi on le fait. Peut - être même que comme moi, vous prendrez goût aux statistiques. Mais si par contre vous ne vous voulez pas prendre le temps de réfléchir à votre travail, malgré toute la bonne volonté que j'ai mise à écrire cet ouvrage le plus simplement possible, je ne peux rien pour vous.

Pour réaliser l'analyse des résultats, j'ai choisi d'utiliser **R**, *qui est à la fois un langage informatique et un logiciel. J'ai fait ce choix car il est gratuit et libre, ce qui vous permet de l'utiliser absolument partout. De plus, il est extrêmement puissant et son caractère libre fait que de nombreux utilisateurs s'investissent pour l'améliorer et l'enrichir en permanence. Enfin, passé le dégoût éventuel d'avoir à écrire soi - même des lignes de commande, vous verrez que* **R** *est simple à utiliser et que mieux, il permet (car il l'oblige) de comprendre ce que l'on fait.*

Comme pour la théorie statistique, utiliser cet aide - mémoire n'exige que très peu de connaissances sur **R**. *Il nécessite seulement de savoir créer les objets de base du langage (vecteur, tableau, matrice) et de savoir effectuer des manipulations simples sur ces objets. Si ces bases ne sont pas acquises, vous pouvez vous référer à certains documents d'introduction à* **R** *cités dans la bibliographie. N'oubliez pas également qu'à chaque fonction dans* **R** *est associée une page d'aide, que l'on appelle par la syntaxe* `?fonction`.

*Il est très important pour moi d'être en contact avec les utilisateurs de cet aide - mémoire, car c'est grâce à cela que je peux l'améliorer. Je remercie donc toutes les personnes qui m'écrivent pour me poser des questions ou pour rectifier des erreurs. Ce sont elles qui me donnent envie de le clarifier, de l'enrichir et de le corriger. Je vous invite donc sincèrement à m'envoyer un e - mail (*mx.herve@gmail.com*) si vous trouvez qu'un point n'est pas clair, qu'un autre mériterait d'être ajouté ou approfondi, ou encore qu'il subsiste des erreurs dans le document.*

Certaines des fonctions présentées dans cet ouvrage nécessitent d'installer des packages qui ne sont pas fournis avec la distribution de base de **R**. *Parmi ceux - ci se trouve le package* `RVAideMemoire`, *qui contient des fonctions que j'ai écrites spécialement pour accompagner cet aide - mémoire. Son développement est donc intimement lié à celui de ce document, et là encore je vous encourage à me faire part de vos remarques, suggestions, critiques et / ou corrections.*

Pour finir, même si vous n'avez rien à dire je vous invite à m'envoyer un petit mail quand même. Cela me permettra de créer une liste de diffusion afin d'informer un maximum d'utilisateurs de la sortie de nouvelles versions de l'aide - mémoire et/ou du package `RVAideMemoire`.

J'espère sincèrement que ce livre comblera vos attentes et qu'il vous permettra de répondre à vos questions.

Le 13 Juillet 2011

Maxime Hervé

Sommaire

L'ouvrage est divisé en quatre parties :

La préparation de l'étude : souvent trop peu d'importance y est attachée. Pourtant, cette phase est au moins aussi cruciale que l'analyse des résultats puisqu'elle détermine la façon dont ceux - ci vont pouvoir être analysés. Une étude bien préparée facilite grandement l'exploitation des résultats, tandis qu'une étude mal préparée entraîne généralement des complications au moment de l'analyse et de l'interprétation.

La préparation et l'importation des données : cette étape apparemment simple peut poser problème par manque d'expérience. Elle est pourtant cruciale, puisque des données mal structurées ou mal importées dans R peuvent conduire à une analyse complètement faussée.

L'analyse descriptive des résultats : ce type d'analyse est toujours indispensable, et selon l'objectif de l'étude il peut être suffisant. L'analyse descriptive est souvent négligée pour « foncer sur les tests », ce qui conduit à oublier la réalité des données (et par conséquent à compliquer voire fausser l'interprétation des résultats).

L'analyse inférentielle des résultats : ce type d'analyse regroupe la détermination des intervalles de confiance et la réalisation des tests statistiques. L'analyse inférentielle est la seule phase de l'étude qui est facultative. Dans tous les cas elle doit passer après l'analyse descriptive.

3.2. Statistique bivariée

13. Nuages de points : la fonction `plot()`
14. La réduction des données à deux dimensions

3.3. Statistique multivariée

> Choisir son analyse multivariée
>
> Ce choix dépend de la nature des variables étudiées :
> – toutes quantitatives : ACP
> – toutes qualitatives :
> – deux variables : AFC
> – plus de deux variables : ACM
> – à la fois quantitatives et qualitatives : Analyse mixte.

15. L'Analyse en Composantes Principales (ACP)
16. L'Analyse Factorielle des Correspondances (AFC)
17. L'Analyse des Correspondances Multiples (ACM)
18. L'Analyse mixte de Hill et Smith

4. ANALYSE INFERENTIELLE DES RESULTATS

4.1. Quelques bases théoriques

4.1.1. Lois de probabilité

4.1.1.1. Lois de probabilité discontinues

19. Lois de probabilité discontinues – généralités
20. La loi binomiale
21. La loi de Poisson
22. La loi binomiale négative

4.1.1.2. Lois de probabilité continues

23. Lois de probabilité continues – généralités
24. La loi normale
25. La loi exponentielle
26. La loi de χ^2
27. La loi de Fisher - Snedecor
28. La loi de Student

4.1.2. Risques et puissance associés aux tests statistiques

29. Principe des tests statistiques et risques associés à la conclusion
30. Le risque ou seuil de rejet α
31. La correction du seuil de rejet α
32. Le risque β et la puissance du test

4.2. Identification des données aberrantes

33. L'identification des données aberrantes

4.3. Intervalles de confiance et erreur standard

34. Intervalle de confiance et erreur standard
35. ℹ Tracer un diagramme en barres avec barres d'erreur

4.4. Tests d'hypothèses

36. Les différents types de test statistique

4.4.1. Conditions préalables à l'utilisation des tests

Ces conditions ne sont pas toujours à remplir, cela dépend du test que l'on souhaite utiliser.

37. Caractère aléatoire et simple d'une série de données
38. Ajustement à une distribution théorique
39. Egalité des variances de plusieurs séries de données
40. Les transformations de variable

4.4.2. Réalisation des tests

Souvent, plusieurs tests peuvent être utilisés pour répondre à la même question. Les conditions de leur emploi sont cependant plus ou moins restrictives, et leur puissance plus ou moins grande (un test plus restrictif étant généralement plus puissant). Lorsque plusieurs tests sont disponibles ils sont présentés du plus au moins restrictif, du plus « pointu » au plus « passe - partout ».

4.4.2.1. Statistique univariée

Tests sur des probabilités de réponse (variables binaires 0 / 1)

Le test de conformité d'une ou de plusieurs probabilité(s) de réponse avec une ou plusieurs valeur(s) théorique(s) est une démarche identique à celle du test de conformité de proportion(s).

41. Comparaison de plusieurs probabilités de réponse – un facteur
42. Comparaison de plusieurs probabilités de réponse – deux facteurs

Tests sur des effectifs

43. Conformité de plusieurs effectifs avec des valeurs théoriques
44. Comparaison de plusieurs effectifs – sans facteur (effectifs bruts)
45. Comparaison de plusieurs effectifs – un facteur
46. Comparaison de plusieurs effectifs – deux facteurs

Tests sur des proportions

47. Conformité d'une proportion avec une valeur théorique
48. Conformité de plusieurs proportions avec des valeurs théoriques
49. Comparaison de deux proportions – sans répétition

50. Comparaison de plusieurs proportions – sans répétition
51. Comparaison de plusieurs proportions – avec répétitions et un facteur
52. Comparaison de plusieurs proportions – avec répétitions et deux facteurs

Régression, analyse de variance / déviance ou analyse de la covariance ?

Dans tous les cas la variable à expliquer est unique et quantitative. Le choix dépend de la nature des variables explicatives :
– toutes quantitatives : régression
– toutes qualitatives : analyse de variance / déviance
– à la fois quantitatives et qualitatives : analyse de la covariance.
Le cas des variables à expliquer qualitatives n'est abordé ici que pour des variables binaires.

Tests sur des moyennes
53. Conformité d'une moyenne avec une valeur théorique
54. Comparaison de deux moyennes
55. Comparaison de plusieurs moyennes – un facteur
56. Comparaison de plusieurs moyennes – deux facteurs

Tests sur des temps de survie
Ces tests sont traditionnellement utilisés pour comparer des temps de survie, mais ils peuvent être appliqués à n'importe quelle variable représentant un temps avant la survenue d'un évènement.
57. Comparaison de plusieurs temps de survie
58. ℹ Tracer des courbes de survie

4.4.2.2. Statistique bivariée

Tests autour de la liaison entre deux variables
59. Indépendance de deux variables qualitatives
60. Corrélation entre deux variables
61. Conformité d'un coefficient de corrélation linéaire avec une valeur théorique
62. Comparaison de plusieurs coefficients de corrélation linéaire

Tests autour de la régression
63. La régression linéaire simple au sens des moindres carrés
64. La régression linéaire simple au sens des moindres rectangles
65. Comparaison de plusieurs droites de régression linéaire simple
66. La régression logistique binaire simple
67. La régression non linéaire simple
68. ℹ Tracer une droite ou une courbe de régression simple

1. Les différents types de variable

Une variable est dite aléatoire si l'on ne peut pas prédire à coup sûr la valeur que prendra un individu. Il existe deux types de variable aléatoire :

1. quantitatives : elles ont en général une infinité de valeurs numériques possibles et peuvent être :
   - continues (ex : masse, temps, distance, volume)
   - discrètes (ex : dénombrement)
2. qualitatives : elles sont en général non numériques (mais pas toujours) et sont appelées facteurs. Leur valeur est appelée classe, niveau ou modalité. Ces variables peuvent être :
   - ordinales, lorsque les classes peuvent être ordonnées (ex : classement)
   - nominales, lorsque les classes ne peuvent pas être ordonnées (ex : sexe).

Les classes d'une variable qualitative sont dites *exclusives* si un individu ne peut pas appartenir à plusieurs classes en même temps. Beaucoup de tests statistiques exigent que les classes soient exclusives. Dans tous les cas, le caractère d'exclusivité doit être déterminé avant toute analyse statistique.

Il existe deux types de facteur :

- fixe : un facteur est fixe si ses classes ont été délibérément choisies, et si le but de l'étude est de les comparer. Par exemple, si l'on veut comparer la taille des individus entre trois espèces, le facteur « espèce » est fixe (à trois classes)
- aléatoire : un facteur est aléatoire si ses classes ont été choisies parmi un grand nombre de classes possibles, et si le but de l'étude n'est pas de les comparer mais simplement de prendre en compte la variabilité qu'il existe entre elles. Par exemple, si les mesures de taille des trois espèces sont réalisées par deux personnes différentes (qui ont chacune mesuré la moitié des individus), on peut considérer un facteur « expérimentateur », aléatoire. L'objectif ici n'est en effet pas de comparer les mesures réalisées par les deux personnes, mais de prendre en compte le fait que la façon de réaliser les mesures peut varier entre les deux.

Il y a deux choses à bien garder à l'esprit : (i) la décision de déclarer un facteur comme fixe ou aléatoire est *fondamentale* pour l'analyse des données, car ce ne sont pas les mêmes analyses qui sont réalisées dans les deux cas ; (ii) cette décision doit être prise *selon l'objectif de l'étude*, *i.e.* la question à laquelle l'étude doit répondre. Il est donc indispensable de bien se poser la question avant de déclarer un facteur fixe ou aléatoire, car aucune décision de peut être prise dans l'absolu.

Que ce soit pour des variables qualitatives ou quantitatives, si certaines mesures ne sont pas indépendantes entre elles, elles constituent des *séries*

appariées. Le cas le plus simple est celui où plusieurs mesures sont réalisées sur un même individu (par exemple avant et après un traitement). Mais d'autres cas plus subtils peuvent se présenter : si des mesures sont réalisées sur des individus apparentés (ces mesures ne sont pas indépendantes car il existe une corrélation d'origine génétique entre elles), si des séries de mesures sont réalisées à des localisations différentes (ces mesures ne sont pas indépendantes car chaque série est influencée par l'environnement local) ou encore si des séries de mesures sont réalisées à des temps différents (ces mesures ne sont pas indépendantes car chaque série est influencée par ce qu'il a pu se passer avant). Il est très important d'identifier les séries appariées lorsqu'elles existent, car ce ne sont pas les mêmes analyses statistiques qui doivent alors être utilisées.

Dans les modèles statistiques, les séries appariées sont identifiées par l'introduction d'un facteur aléatoire. Pour les exemples précédents, on a donc respectivement un facteur « individu », un facteur « famille », un facteur « localisation » et un facteur « moment ».

2. Le plan d'échantillonnage

On utilise un plan d'échantillonnage lorsque l'on réalise une étude par enquête, *i.e.* lorsque l'on collecte des informations sur un groupe d'individus dans leur milieu habituel, mais que tous les individus ne sont pas accessibles (par choix ou par contrainte).

Les principales méthodes d'échantillonnage peuvent être regroupées en deux ensembles :

1. l'échantillonnage aléatoire : tous les individus (au sens statistique) ont la même probabilité d'être choisis, et le choix de l'un n'influence pas celui des autres. Différentes méthodes d'échantillonnage aléatoire existent :
   - l'échantillonnage aléatoire et simple : le choix se fait parmi tous les individus de la population (au sens statistique), qui ne forme qu'un grand ensemble
   - l'échantillonnage stratifié : si la population est très hétérogène, elle peut être divisée en sous - ensembles exclusifs (ou strates). Au sein de ces strates l'échantillonnage est ensuite aléatoire et simple
   - l'échantillonnage en grappes : si les strates sont très nombreuses, on en choisit certaines au hasard (les grappes). Au sein de ces grappes l'échantillonnage est ensuite aléatoire et simple
   - l'échantillonnage par degrés : il est une généralisation de l'échantillonnage en grappes (qui est en fait un échantillonnage du premier degré). Au sein de la population on choisit des grappes « primaires », puis à l'intérieur de celles-ci des grappes « secondaires » (toujours au hasard), et ainsi du suite... Au dernier niveau l'échantillonnage est aléatoire et simple
2. l'échantillonnage systématique : un premier individu est choisi aléatoirement, puis les autres sont choisis de façon régulière à partir du précédent (dans le temps ou l'espace). L'analyse de ce type d'échantillonnage, qui fait appel à la statistique spatiale ou à l'analyse des séries chronologiques, n'est pas abordée dans cet ouvrage.

Il est important d'identifier la méthode mise en œuvre car les analyses statistiques doivent être adaptées. Seule l'analyse de plans d'échantillonnage aléatoires est abordée dans cet ouvrage.

3. Le plan d'expérience

On utilise un plan d'expérience lorsque l'on réaliste une étude par expérimentation, *i.e.* lorsque l'on provoque volontairement les faits à étudier. Le plan d'expérience comprend notamment le(s) facteur(s) à faire varier, le nombre de répétitions à réaliser et le dispositif expérimental à mettre en place. L'association des classes de plusieurs facteurs constitue un traitement.

Il existe de nombreux types de dispositif expérimental, dont les principaux sont :

- le plan d'expérience complètement aléatoire : chaque individu (au sens statistique) est affecté à un traitement aléatoirement
- le plan d'expérience en blocs aléatoires complets : s'il y a (ou s'il peut y avoir) une grande hétérogénéité entre les individus, ils sont réunis en groupes aussi homogènes que possibles (ou blocs). Au sein de ces blocs chaque individu est ensuite affecté aléatoirement à un traitement, de manière à ce que tous les traitements soient présents dans chacun des blocs
- le plan d'expérience en blocs aléatoires incomplets : dans ce cas tous les traitements ne sont pas présents dans chacun des blocs
- le plan d'expérience en split - plot : le principe du split - plot est le plus souvent associé à celui des blocs aléatoires complets. Dans ce cas, dans chacun des blocs sont créés autant de sous - blocs qu'il y a de classes au premier facteur étudié. A chacun de ces sous - blocs est associée une classe. Puis chaque sous - bloc est divisé en autant d'unités qu'il y a de classes au second facteur étudié. A chacun de ces « sous - sous - blocs » est associée une classe. Pour plus de deux facteurs, la situation est plus complexe.

Quelle que soit la méthode employée, elle doit être clairement définie car elle doit être prise en compte dans les analyses statistiques.

4. La détermination de la taille de l'échantillon

Il existe un lien entre le seuil de rejet α du test statistique utilisé (voir fiches **29** et **30**), la puissance de ce test (voir fiche **32**), la différence entre les échantillons pour le paramètre mesuré et la taille des échantillons. Déterminer la taille de l'échantillon à constituer passe donc par fixer les autres paramètres. Ceci implique deux choses importantes :

- choisir avant de démarrer l'étude les types de test qui vont être utilisés (ce qui oblige à bien identifier les questions auxquelles l'étude doit répondre) et leur précision
- avoir une idée de la variabilité naturelle du paramètre mesuré et / ou de la différence minimale à détecter. Ceci passe soit par une étude de la bibliographie, soit par la consultation de spécialistes, soit par la réalisation d'un pré - échantillonnage ou d'une pré - expérience.

Dans **R**, les fonctions `power()` et `pwr()` (la seconde étant contenue dans le package `pwr`) déterminent le paramètre souhaité quand les autres sont fixés, pour plusieurs tests.

Toutes les fonctions décrites sont basées sur le même principe : le paramètre à déterminer doit avoir comme valeur `NULL` tandis que tous les autres doivent être fixés.

Comparaison de deux moyennes (test *t* de Student)

`power.t.test(n,delta,sd,sig.level,power,type)`

avec :
`n` : effectif (identique pour les deux échantillons)
`delta` : différence minimale à détecter entre les deux moyennes
`sd` : écart-type (identique pour les deux échantillons)
`sig.level` : seuil de rejet α (généralement 0,05)
`power` : puissance minimale du test (généralement 80 ou 90 %)
`type` : type de test (`"two.sample"` pour deux moyennes observées, `"one.sample"` pour une moyenne observée à comparer avec une théorique, `"paired"` pour deux moyennes observées en séries appariées).

`pwr.t.test(n,d,sig.level,power,type)`

avec :
`d` : $\frac{\mu_A - \mu_B}{\sigma}$ (différence des moyennes sur écart - type).

Utiliser `pwr.t2n.test(n1,n2,d,sig.level,power)` pour deux échantillons de taille différente (la fonction ne gère pas les séries appariées).

Comparaison de plus de deux moyennes (ANOVA)

`power.anova.test(groups,n,between.var,within.var,sig.level,power)`

avec :

`groups` : nombre de modalités à comparer

`between.var` : variance intergroupe minimale à détecter

`within.var` : variance intragroupe (identique pour toutes les modalités).

La fonction ne gère pas les séries appariées.

`pwr.anova.test(k,n,f,sig.level,power)`

avec :

`k` : nombre de modalités à comparer

`f` : taille minimale de l'effet à détecter.

La fonction ne gère pas les séries appariées.

Comparaison de deux proportions

`power.prop.test(n,p1,p2,sig.level,power)`

avec `p1`, `p2` : proportion observée dans chaque échantillon.

`pwr.2p.test(h,n,sig.level,power)`

avec `h` : taille minimale de l'effet à détecter (en proportion).

Utiliser `pwr.2p2n.test(h,n1,n2,sig.level,power)` pour deux échantillons de taille différente.

Corrélation linéaire entre deux séries de données

`pwr.r.test(n,r,sig.level,power)`

avec `r` : coefficient de corrélation linéaire de Pearson minimum à mettre en évidence.

5. Construction du tableau de données

La construction d'un tableau de données correctement structuré est une étape importante de l'étude, car si elle est mal réalisée elle peut mener à des résultats faux, ou le plus souvent à des erreurs une fois dans R.

Cette construction nécessite de se poser une question essentielle : quelles sont les variables prises en compte dans l'étude ? Y répondre implique d'identifier les variables quantitatives et les facteurs, ainsi que les classes des facteurs. Si les choses sont claires, l'analyse statistique le sera également.

D'une manière générale, il est conseillé de toujours construire son tableau de données dans un tableur. Cela permet d'enregistrer le jeu de données dans un fichier externe à R, et donc de toujours pouvoir y revenir puisque R ne modifie pas les fichiers externes (sauf si on lui demande explicitement).

Une fois dans le tableur, la règle est simple : les individus doivent être placés en *lignes* et les variables en *colonnes*.

Il est conseillé de donner un titre à chaque colonne, qui deviendra le nom de la variable dans R. Il est indispensable cependant de respecter certaines règles : les noms de variable ne doivent contenir ni espace, ni caractère accentué, ni symbole (ceci est une règle pour tous les noms d'objet dans R). Si un nom de variable doit contenir deux mots, ils peuvent être séparés par un point (.) ou un tiret bas (_). Mieux vaut également privilégier les noms courts mais clairs, car une fois dans R taper sans cesse des noms de variable longs est vite fastidieux.

Le tableau de données doit absolument obéir à une autre règle : *aucune case ne doit être vide*. S'il manque une donnée pour un individu, il faut se demander d'où elle vient :

- si c'est une donnée inutilisable (mesure ratée, mal retranscrite...), c'est normal. On dit alors qu'on a une « donnée manquante », que l'on doit noter `NA` (pour *Not Available*, *i.e.* donnée manquante). Le tableur comme R reconnaissent le `NA`, qu'ils interprètent correctement
- si la situation est autre, c'est que le tableau est mal construit et qu'en particulier les variables n'ont pas été bien définies. La réflexion s'impose donc pour identifier les variables et reconstruire un tableau de données.

Si des analyses dans R doivent se faire uniquement sur un sous - ensemble du tableau de données, ou si pour certaines analyses le tableau de données serait plus facile à utiliser s'il était construit autrement, il est conseillé de construire plusieurs tableaux de données. Il est toujours possible de manipuler le tableau initial dans R pour en extraire une partie ou pour le transformer, mais il est clairement plus facile (et surtout moins source d'erreur) de le faire en amont, dans le tableur.

6. Importation du tableau de données dans R

Il existe de nombreuses méthodes pour importer ses données dans R. Une seule est présentée ici, qui est à la fois très simple, fonctionne dans la plupart des situations et peut être utilisée sur toutes les plates - formes.

La procédure se fait en trois étapes :

1. dans le tableur, sélectionner toutes les cases constituant le tableau de données
2. copier ce tableau dans le bloc - notes et enregistrer le fichier en `.txt`
3. dans R, charger le tableau de données grâce à la fonction `read.table()` et le stocker dans un objet : `tableau<-read.table(fichier,dec=",")` où `fichier` est le nom du fichier texte (et éventuellement du chemin qui y mène), entre guillemets.

R étant un logiciel anglo - saxon, le séparateur décimal qu'il utilise est le point. Or dans les tableurs français (et donc dans le fichier texte) le séparateur décimal est la virgule. Il est donc nécessaire de préciser à R qu'il interprète la virgule comme séparateur décimal, d'où l'argument `dec=","`.

Si les colonnes du tableau de données ont un titre, qui doit donc être interprété comme le nom de la variable, ajouter l'argument `header=TRUE`.

Une fois le tableau importé, il est indispensable de vérifier qu'il n'y a pas eu d'erreur pendant son chargement. Pour cela appeler le résumé du tableau *via* `summary(tableau)`. R renvoie un résumé de chaque variable :

- pour une variable numérique, R donne des indications sur sa distribution : minimum, 1^{er} quartile, médiane, moyenne, $3^{ème}$ quartile et maximum
- pour un facteur, R donne le nombre d'individus par classe.

Si un facteur est codé numériquement (par exemple un facteur binaire ou un facteur ordinal), R l'interprète comme une variable numérique. Pour transformer la variable en facteur : `tableau$variable<-factor(tableau$variable)` où `variable` est le nom de la variable.

7. ℹ Installer et charger un package

Installer un package

Il est nécessaire d'être connecté à internet pour installer un package, car celui - ci doit être téléchargé. L'installation ne se fait qu'une seule fois.

Si R est utilisé depuis la console R, utiliser : `install.packages("package")` où `package` est le nom du package désiré, entre guillemets. Il est demandé ensuite de choisir un serveur, Lyon1 est bien réputé en France.

Si R est utilisé depuis la console système, la procédure se fait en deux étapes :

1. télécharger les sources du package à partir de son site de dépôt, le site principal étant le CRAN (*the Comprehensive R Archive Network*) : *http ://cran.r-project.org* rubrique *Packages*
2. installer le package en tapant `R CMD INSTALL package` où `package` est le nom du fichier `tar.gz` contenant les sources.

La procédure expliquée ici est la plus simple, mais il existe de nombreuses variantes. Voir la R FAQ pour plus d'informations : *http ://cran.r-project.org/doc/manuals/R-admin.html#Installing-packages*.

Charger un package

Le chargement d'un package doit se faire à chaque session où il doit être utilisé. La commande est simple : `library(package)` où `package` est le nom du package, sans guillemets.

Mettre à jours les packages installés

Pour mettre à jour automatiquement tous les packages installés (chargés ou non), utiliser : `update.packages()`. R demande une confirmation pour chaque package dont une mise à jour est disponible, puis télécharge toutes les mises à jour demandées.

8. ℹ Citer R et ses packages

Lors de l'écriture d'un document scientifique, il est une évidence de citer ses sources bibliographiques. Il doit également en être une de citer les logiciels utilisés lors de la réalisation de l'étude. R est certes gratuit, mais il n'en reste pas moins que des dizaines de personnes s'impliquent dans son développement, et qu'il est normal de faire honneur à leur travail en les citant.

R doit être cité dès lors qu'il est utilisé. Pour savoir comment le citer, il suffit de taper `citation()` et de recopier ce qui figure après `To cite R in publications use:`.

Concernant les packages, la règle est de citer tous ceux qui ne sont pas chargés au démarrage de R. Cela comprend les packages installés avec R mais non chargés automatiquement, ainsi que ceux installés par l'utilisateur. Pour savoir comment les citer, utiliser : `citation("package")` où `package` est le nom du package, entre guillemets. Recopier ce qui figure après `To cite the xxx package in publications use:`.

9. Graphiques de dispersion : la fonction `stripchart()`

Le graphique tracé représente toutes les données individuelles d'un vecteur, d'une matrice ou d'un tableau. Il permet donc d'avoir un aperçu de la variabilité des données et d'identifier les observations aberrantes.

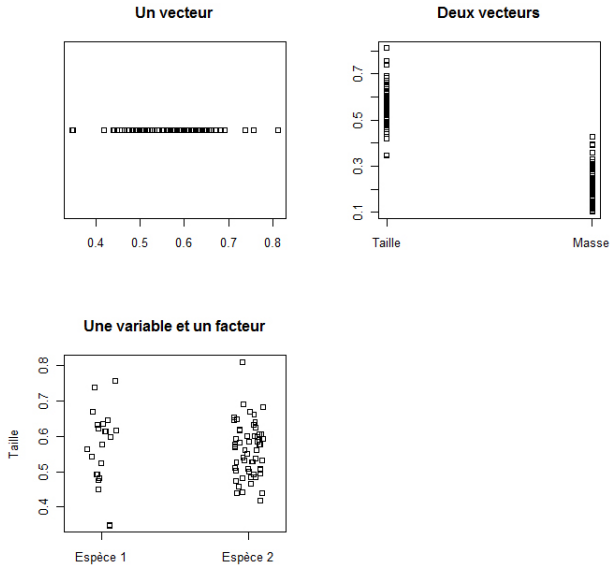
Pour représenter un vecteur : `stripchart(vecteur)`.

Pour représenter plusieurs vecteurs : `stripchart(list(vecteur1,vecteur2,...))`.
Pour donner un nom aux vecteurs sur le graphe, ajouter l'argument `group.names=c("Nom1","Nom2",...)`.

Pour représenter des données en fonction d'un facteur : `stripchart(donnees~facteur)` où les deux objets sont des vecteurs contenant la valeur de chaque individu (dans le même ordre).

Pour représenter les données verticalement, ajouter l'argument `vertical=TRUE`.

Pour que les valeurs identiques ne se superposent pas, ajouter l'argument `method="jitter"` (par défaut `method="overplot"`).



10. Histogrammes : la fonction `hist()`

Le graphique tracé divise les données contenues dans un vecteur en classes, et représente chaque classe en effectif ou densité. Il permet donc d'avoir un aperçu de la distribution des données.

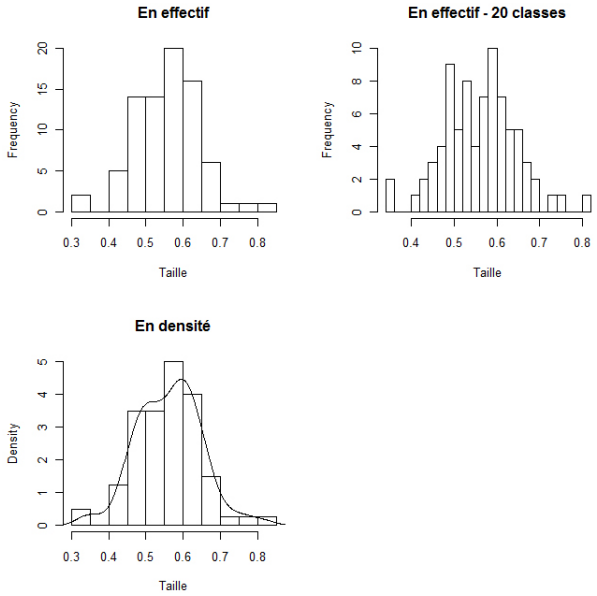
Pour représenter les classes en effectifs : `hist(vecteur)`.
Pour représenter les classes en densités : `hist(vecteur,freq=FALSE)` (`freq=TRUE` par défaut, ce qui représente les effectifs).

Pour ajouter une courbe de densité : `lines(density(vecteur))`.

Pour ajouter une courbe de distribution : `lines(seq2(vecteur),dloi(seq2(vecteur),par))` où `loi` est la loi de probabilité choisie et `par` ses paramètres séparés par une virgule (calculés à partir des données ; voir fiches **19** à **28**). La fonction `seq2()` est contenue dans le package `RVAideMemoire`.

Pour modifier le nombre de classes, ajouter l'argument `breaks=n` où `n` est le nombre de coupures souhaitées (il y a donc `n` + 1 classes).

La fonction considère par défaut qu'une valeur égale à la borne inférieure d'une classe appartient à la classe précédente, et qu'une valeur égale à la borne supérieure appartient à cette classe. Pour inverser cette exclusivité ajouter l'argument `right=FALSE` (par défaut `right=TRUE`).



11. Boîtes à moustaches : la fonction `boxplot()`

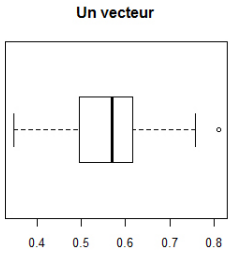
Le graphique tracé représente de façon simplifiée la dispersion des données contenues dans un vecteur. Il permet donc d'avoir un aperçu de la distribution et de la variabilité des données, et d'identifier les observations aberrantes.

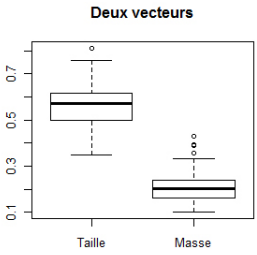
Pour représenter un vecteur : `boxplot(vecteur)`. Le trait épais représente la médiane, la boîte est formée par les valeurs des 1er et 3ème quartiles, et les moustaches mesurent maximum 1,5 fois la longueur de l'interquartile (3ème - 1er). Les valeurs au - delà des moustaches sont représentées individuellement.

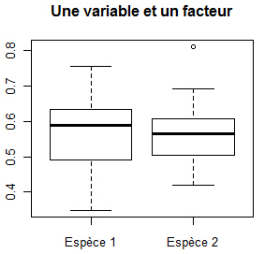
Pour représenter plusieurs vecteurs : `boxplot(list(vecteur1,vecteur2,...))`.
Pour donner un nom aux boîtes, ajouter l'argument `names=c("Nom1","Nom2",...)`.

Pour représenter des données en fonction d'un facteur : `boxplot(donnees~facteur)` où les deux objets sont des vecteurs contenant la valeur de chaque individu (dans le même ordre).

Pour représenter les boîtes horizontalement, ajouter l'argument `horizontal=TRUE`.







12. La réduction des données à une dimension

Les paramètres suivants permettent de réduire une série de données à quelques valeurs apportant une information générale.

Paramètres de position

Ils permettent de donner un ordre de grandeur des données.
Moyenne : `mean(serie)` où `serie` est un vecteur contenant la valeur de chaque individu.
Médiane : `median(serie)`.
Mode : utiliser la fonction `mod()` du package `RVAideMemoire` : `mod(serie)`.

Si les vecteurs contiennent des données manquantes (`NA`), ajouter l'argument `na.rm=TRUE` aux fonctions `mean()` et `median()`. La fonction `mod()` gère par défaut les données manquantes.

Paramètres de dispersion

Ils permettent d'estimer la variabilité des données autour des paramètres de position.
Variance : `var(serie)`.
Ecart - type (*standard deviation*) : `sd(serie)`.
Coefficient de variation : utiliser la fonction `cv()` du package `RVAideMemoire` : `cv(serie)`. Le coefficient est par défaut exprimé en valeur absolue et en pourcentage.

Si les vecteurs contiennent des données manquantes (`NA`), ajouter l'argument `na.rm=TRUE` aux fonctions `var()` et `sd()`. La fonction `cv()` gère par défaut les données manquantes.

Les fonctions `var()` et `sd()` calculent la variance et l'écart - type non biaisés (sur la base de $n-1$ et non n, si n est l'effectif de l'échantillon).

13. Nuages de points : la fonction `plot()`

La fonction `plot()` permet de représenter les valeurs de deux variables numériques pour chaque individu, dans un graphe du type $y = f(x)$. Elle permet donc d'avoir un aperçu de la liaison qu'il peut exister entre ces variables. Elle peut être utilisée avec deux vecteurs, une matrice à deux colonnes ou un tableau.

Pour représenter deux vecteurs `x` et `y` contenant la valeur de chaque individu pour les deux variables (dans le même ordre) : `plot(x,y)` ou `plot(y~x)`.

Pour ajouter une droite du type $y = ax + b$: `abline(b,a)`.

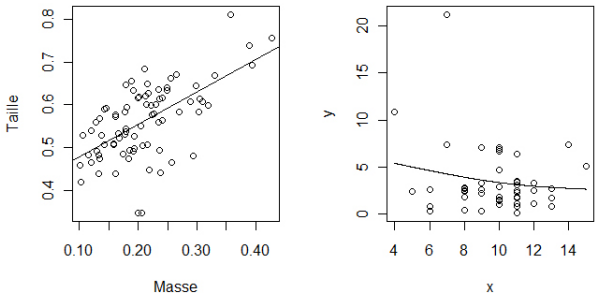
Pour ajouter une droite de régression linéaire au sens des moindres carrés (voir fiche **63**) : `abline(lm(y~x))`.

Pour ajouter une droite de régression linéaire au sens des moindres rectangles (voir fiche **64**), utiliser la fonction `least.rect()` du package `RVAideMemoire` : `abline(least.rect(x,y))`.

Pour ajouter une courbe de tendance du nuage de points : `panel.smooth(x, y)`

Pour ajouter une droite horizontale : `abline(h=ordonnee)`.

Pour ajouter une droite verticale : `abline(v=abscisse)`.



14. La réduction des données à deux dimensions

Les paramètres suivants permettent de réduire deux séries de données à quelques valeurs apportant une information générale sur la liaison qui peut les unir.

Pour deux vecteurs `x` et `y` contenant la valeur de chaque individu pour chaque série (dans le même ordre) :
Covariance : `cov(x,y)`.
Coefficient de corrélation linéaire de Pearson (voir fiche **60**) : `cor(x,y)`.

Les vecteurs doivent avoir la même taille (*i.e.* contenir autant de valeurs). S'ils contiennent des données manquantes (`NA`), ajouter l'argument `use="complete.obs"` (qui ne considère que les couples de données complets) aux fonctions `cov()` et `cor()`.

La régression linéaire au sens des moindres carrés (voir fiche **63**) s'utilise dans le cas où la variable `y` (dite dépendante ou à expliquer) varie en fonction de la variable `x` (dite indépendante ou explicative). Pour récupérer les paramètres de la droite : `lm(y~x)$coefficients`. La première valeur (`Intercept`) correspond à l'ordonnée à l'origine, la seconde au coefficient directeur de la droite. La fonction `lm()` construit un modèle linéaire reliant `x` et `y` au sens des moindres carrés.

La régression linéaire au sens des moindres rectangles (voir fiche **64**) s'utilise dans le cas où les deux variables sont considérées sur un pied d'égalité (elles sont dites interdépendantes). Pour récupérer les paramètres de la droite, utiliser la fonction `least.rect()` du package `RVAideMemoire` : `least.rect(x,y)$coefficients`. La première valeur (`Intercept`) correspond à l'ordonnée à l'origine, la seconde au coefficient directeur de la droite. La fonction `least.rect()` construit un modèle linéaire reliant `x` et `y` au sens des moindres rectangles.

Les vecteurs `x` et `y` doivent avoir la même taille pour les deux fonctions `lm()` et `least.rect()`. Ils peuvent contenir des données manquantes (`NA`).

15. L'Analyse en Composantes Principales (ACP)

Conditions : les variables doivent être quantitatives ou ordinales.
Les données doivent être contenues dans un tableau avec en colonnes les variables (avec un titre) et en ligne les individus. La 1ère case du tableau (en haut à gauche) doit être vide.
Le tableau ne doit pas contenir de données manquantes (`NA`). Pour supprimer les lignes qui en possèdent : `tableau2<-na.omit(tableau)`, où `tableau` est le tableau de départ.

L'ACP se prête mieux à l'analyse de relations linéaires et de variables ayant une distribution symétrique. Pour observer ces caractères, utiliser `pairs(tableau2,panel=panel.smooth)` d'une part et `hist()` (voir fiche **10**) avec chacune des variables d'autre part. Si besoin transformer les variables (voir fiche **40**).

Pour visualiser la matrice des corrélations linéaires entre les variables : `cor(tableau2)` (utilisée dans le cas de l'ACP réduite).
Pour visualiser la matrice des variances - covariances : `cov(tableau2)` (utilisée dans le cas de l'ACP non réduite).

L'ACP est réalisée grâce à la fonction `dudi.pca()` du package `ade4`. Par défaut elle est centrée - réduite, pour changer ces paramètres ajouter les arguments `center=FALSE` (pour ne pas centrer) et/ou `scale=FALSE` (pour ne pas réduire). En général, l'ACP est réduite quand les variables n'ont pas le même ordre de grandeur (ou unité de mesure), non réduite dans le cas contraire (elles sont donc directement comparables).

Lorsque la commande `acp<-dudi.pca(tableau2)` est passée, **R** renvoie le graphe des valeurs propres (ou pouvoirs de synthèse) associées à chaque variable de synthèse (ou axe ou composante principale), et demande le nombre d'axes à sélectionner. Le choix peut se faire sur la base du critère de Kaiser (*i.e.* ne choisir que les axes dont la valeur propre est supérieure à 1) ou sur un critère fixé avant l'analyse, à savoir le nombre de composantes principales expliquant un minimum de x % de l'information (ou inertie totale) contenue dans le tableau initial. Ce pourcentage d'inertie est calculé en divisant la valeur propre d'un axe par le nombre d'axes possibles (égal au nombre de variables du tableau de départ).

Pour visualiser la corrélation entre chaque variable et une composante principale : `score.pca(acp,xax=num)` où `num` est le numéro de l'axe choisi.

Le diagnostic de l'ACP se fait grâce à la fonction `inertia.dudi(acp)`, qui renvoie le tableau `$TOT`. Celui - ci contient la contribution à l'inertie de chaque composante principale (proportions cumulées dans la colonne `ratio`, multiplier par 100 pour le pourcentage). En ajoutant l'argument `col.inertia=TRUE` à la fonction on obtient trois tableaux supplémentaires :

- `$col.abs` : donne la contribution à l'inertie de chaque variable du tableau de départ, *i.e.* leur importance respective dans la construction des axes (ce qui aide à l'interprétation de ces axes). Diviser par 100 pour obtenir les pourcentages. La somme de chaque colonne est égale à 100
- `$col.rel` : donne la part d'information apportée par chaque axe pour chaque variable (diviser la valeur absolue par 100 pour obtenir les pourcentages). Ne pas tenir compte de la 3ème colonne
- `$col.cum` : valeurs absolues de `$col.rel` cumulées (diviser par 100 pour obtenir les pourcentages). Donne donc la part totale d'information apportée par tous les axes retenus pour chaque variable, autrement dit la qualité de la représentation de chaque variable.

Ajouter l'argument `row.inertia=TRUE` à la fonction `inertia.dudi()` pour obtenir le diagnostic pour chaque individu (`$row.abs`, `$row.rel` et `$row.cum`).

Pour visualiser graphiquement les relations entre :

- les individus : `s.label(acp$li)`, le tableau `$li` de l'ACP donnant les coordonnées des individus dans les différents plans factoriels. Pour sélectionner un plan factoriel, ajouter les arguments `xax=num1` et `yax=num2` où `num1` est le numéro du 1er axe choisi et `num2` celui du 2nd (il faut avoir sélectionné au moins deux axes au départ de l'ACP). Par convention on choisit pour l'axe horizontal celui des deux ayant la meilleure contribution à l'inertie totale. Des individus éloignés sur le graphe le sont dans le tableau initial (mais faire attention aux contributions relatives `$row.rel` du diagnostic).
  Pour ajouter comme information supplémentaire une variable qualitative définissant des groupes d'individus, utiliser `s.class(dfxy=acp$li,fac=facteur)` où `facteur` est un vecteur contenant la modalité de chaque individu (dans le même ordre que les autres variables). Pour donner une couleur à chaque groupe ajouter l'argument `col=couleur` où `couleur` est un vecteur contenant la couleur de chaque modalité, dans l'ordre alphabétique des modalités. Préciser le plan factoriel grâce aux arguments `xax` et `yax`
- les variables :
  - ACP réduite : `s.corcircle(acp$co)` trace le cercle des corrélations (le tableau `$co` de l'ACP donne les coordonnées des variables dans les différents plans factoriels), où la longueur des flèches indique la part de leur information représentée par les deux axes (contributions

relatives cumulées `$col.cum` du diagnostic). L'angle entre deux flèches représente la corrélation qui les lie : angle aigu = positive ; angle droit = nulle ; angle obtus = négative

- ACP non réduite : `s.arrow(acp$co)` où la longueur des flèches représente la contribution à l'inertie de chaque variable (contributions absolues `$col.abs` du diagnostic). Les relations entre variables s'interprètent de la même façon que pour l'ACP réduite, mais cette fois en terme de covariances et non de corrélations.

Pour représenter à la fois les individus et les variables dans un plan factoriel, utiliser `scatter(acp)`. Pour ajouter comme information supplémentaire une variable qualitative définissant des groupes d'individus, utiliser la procédure suivante :

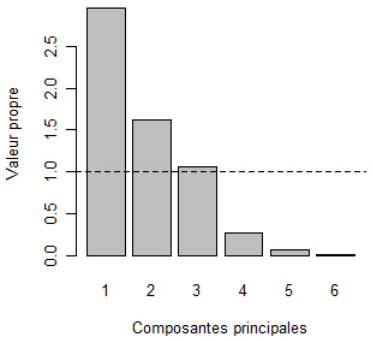
```
> scatter(acp,clab.row=0,posieig="none")
> s.class(dfxy=acp$li,fac=facteur,col=couleur,add.plot=TRUE)
```

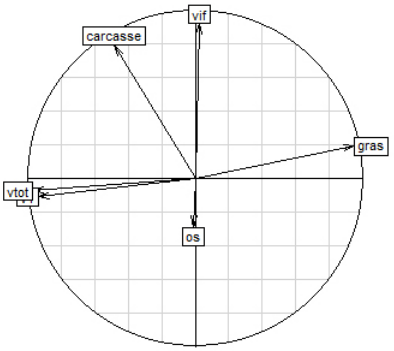
Préciser le plan factoriel grâce aux arguments `xax` et `yax`.

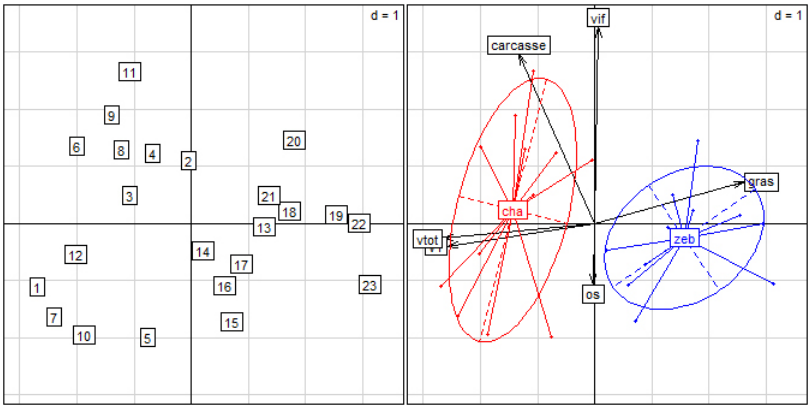
L'échelle des flèches sur la double représentation individus - variables est arbitraire, elle peut être changée sans que cela ne change l'interprétation.

Pour l'interprétation, n'utiliser que les individus les plus éloignés du centre du nuage de points et les flèches les plus longues pour les variables, car ce sont eux qui sont le mieux représentés par les axes.

Un 1er axe très corrélé de façon positive avec toutes les variables est souvent le signe d'un « effet taille ». Il convient dans ce cas de se placer dans des plans factoriels ne comprenant pas le 1er axe pour l'interprétation.


Valeur propre
0.0
0.5
1.0
1.5
2.0
2.5
1
2
3
4
5
6
Composantes principales


vif
carcasse
gras
vtot
os


d = 1
11
9
6
8
4
2
20
3
21
18
19
22
13
12
14
17
16
23
1
7
15
10
5
d = 1
vif
carcasse
gras
cha
vtot
zeb
os

16. L'Analyse Factorielle des Correspondances (AFC)

Conditions : les variables doivent être au nombre de deux et qualitatives.

Les données doivent être en effectifs et organisées dans un tableau de contingence, du type :

		Variable B		
		Classe 1	...	Classe c
Variable A	Classe 1			
	...			
	Classe k			

Ce tableau est obtenu de la manière suivante : `tableau<-table(variableA,variableB)` où `variableA` et `variableB` sont des vecteurs contenant la valeur de chaque individu pour la chaque variable (dans le même ordre).

L'AFC est sensible aux effectifs faibles, aussi regrouper les classes quand cela est nécessaire.

Commencer par calculer la valeur du χ^2 du test d'indépendance des deux variables (voir fiche **59**), réalisé à partir du tableau initial : `chisq.test(tableau)$statistic`.

L'AFC est réalisée grâce à la fonction `dudi.coa()` du package `ade4`. Lorsque la commande `afc<-dudi.coa(tableau)` est passée, R renvoie le graphe des valeurs propres (ou pouvoirs de synthèse) associées à chaque variable de synthèse (ou axe) et demande le nombre d'axes à sélectionner. La part de l'inertie (ou information totale contenue dans le tableau initial) expliquée par chaque axe se calcule simplement par :

$$\frac{\text{valeur propre} \times \text{effectif total}}{\chi^2}$$

On peut choisir le nombre d'axes expliquant ensemble x % de l'inertie, x étant choisi à l'avance, ou un nombre d'axes déterminé à l'avance.

Le diagnostic de l'AFC se fait grâce à la fonction `inertia.dudi(afc)`, qui renvoie le tableau `$TOT`. Celui - ci contient la contribution à l'inertie de chaque axe (proportions cumulées dans la colonne `ratio`, multiplier par 100 pour le pourcentage). En ajoutant l'argument `col.inertia=TRUE` à la fonction on obtient trois tableaux supplémentaires :

- `$col.abs` : donne la contribution à l'inertie de chaque colonne du tableau de départ, *i.e.* leur importance respective dans la construction des axes (ce qui aide à l'interprétation de ces axes). Diviser par 100 pour obtenir les pourcentages. La somme de chaque colonne est égale à 100 %
- `$col.rel` : donne la part d'information apportée par chaque axe pour chaque colonne (diviser la valeur absolue par 100 pour obtenir les pourcentages). Ne pas tenir compte de la 3$^{\text{ème}}$ colonne

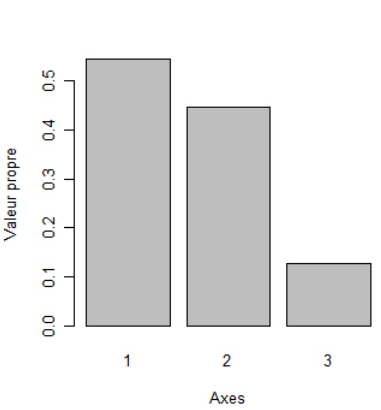
– `$col.cum` : valeurs absolues de `$col.rel` cumulées (diviser par 100 pour obtenir les pourcentages). Donne donc la part totale d'information apportée par tous les axes retenus pour chaque colonne, autrement dit la qualité de la représentation de chaque colonne.

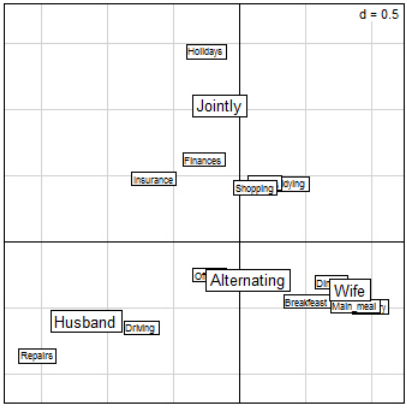
Ajouter l'argument `row.inertia=TRUE` à la fonction `inertia.dudi()` pour obtenir le diagnostic pour chaque ligne (`$row.abs`, `$row.rel` et `$row.cum`).

Pour visualiser graphiquement le résultat de l'AFC (donc la structure du tableau), utiliser `scatter(afc,posieig="none")`. Pour sélectionner un plan factoriel, ajouter les arguments `xax=num1` et `yax=num2` où `num1` est le numéro du 1er axe choisi et `num2` celui du 2nd (il faut avoir sélectionné au moins deux axes au départ de l'AFC). Par convention on choisit pour l'axe horizontal celui des deux ayant la meilleure contribution à l'inertie totale. La proximité des modalités représente leur liaison plus ou moins forte dans le tableau initial (mais faire attention aux contributions relatives `$col.rel` et `$row.rel` du diagnostic).

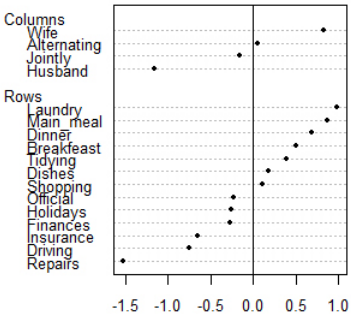
La contribution des lignes et des colonnes à la construction des axes ne peut pas se lire sur le graphique (leur éloignement de l'origine n'est pas représentatif de leur contribution à l'inertie des axes). Il est donc indispensable de lire en détail le diagnostic.

Pour interpréter les axes il peut être utile d'utiliser `score.coa(afc,xax=num,dotchart=TRUE)` où `num` est le numéro de l'axe à représenter. Le graphique montre la répartition des modalités sur l'axe choisi. Utiliser `abline(v=0)` pour ajouter une ligne marquant l'origine de l'axe.





Répartition des modalités sur l'axe 1



17. L'Analyse en Composantes Multiples (ACM)

Conditions : les variables doivent être qualitatives (ordinales ou nominales).

Les données doivent être contenues dans un tableau avec en colonnes les variables (avec un titre) et en ligne les individus. La 1ère case du tableau (en haut à gauche) doit être vide. Pour intégrer des variables quantitatives, regrouper les valeurs en classes (les variables deviennent donc qualitatives).

Le tableau ne doit pas contenir de données manquantes (`NA`). Pour supprimer les lignes qui en possèdent : `tableau2<-na.omit(tableau)`, où `tableau` est le tableau de départ.

Il est indispensable de réaliser une analyse préliminaire de chaque variable, afin de voir si toutes les classes sont aussi bien représentées ou s'il existe un déséquilibre. Pour cela utiliser `plot(variable)` où `variable` est la variable choisie. L'ACM est sensible aux effectifs faibles, aussi regrouper les classes quand cela est nécessaire.

De même, si les variables ne sont pas trop nombreuses, réaliser un test du χ^2 d'indépendance deux - à - deux est une première approche : `chisq.test(table(variable1,variable2))` (voir fiche **59**).

L'ACM est réalisée grâce à la fonction `dudi.acm()` du package `ade4`. Lorsque la commande `acm<-dudi.acm(tableau)` est passée, **R** renvoie le graphe des valeurs propres (ou pouvoirs de synthèse) associées à chaque variable de synthèse (ou axe) et demande le nombre d'axes à sélectionner. Il est difficile d'établir un critère de choix pour l'ACM, ne pas multiplier les axes est en tout cas bénéfique pour l'interprétation (ce qui est aussi valable pour les autres analyses multivariées).

Le diagnostic de l'ACM se fait grâce à la fonction `inertia.dudi(acm)`, qui renvoie le tableau `$TOT`. Celui - ci contient la contribution à l'inertie (ou information totale) de chaque axe (proportions cumulées dans la colonne `ratio`, multiplier par 100 pour le pourcentage).

Pour représenter les résultats de l'ACM, utiliser `scatter(acm)`. Les résultats sont séparés pour chaque variable, toutes représentées sur le même plan factoriel. Pour sélectionner un plan factoriel, ajouter les arguments `xax=num1` et `yax=num2` où `num1` est le numéro du 1er axe choisi et `num2` celui du 2nd (il faut avoir sélectionné au moins deux axes au départ de l'ACM). Par convention on choisit pour l'axe horizontal celui des deux ayant la meilleure contribution à l'inertie totale. Les modalités de chaque variable sont représentées par des ellipses portant leur nom. Pour rendre le graphique plus clair et donner une couleur à chaque modalité, ajouter l'argument `col=couleur` où `couleur` est un vecteur contenant autant de noms (ou numéros) de couleurs qu'il y a de modalités possibles.

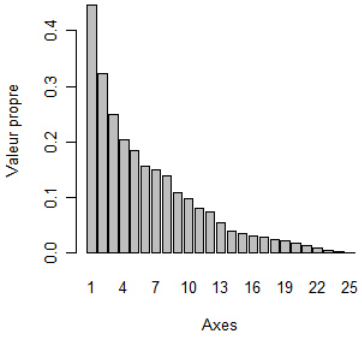
Pour ne représenter qu'une seule variable, le même résultat est obtenu par `s.class(dfxy=acm$li,fac=variable,col=couleur,cstar=0,sub="nom")` où `variable` est la variable choisie et `nom` son nom (entre guillemets). Choisir le plan factoriel à l'aide des arguments `xax` et `yax`.

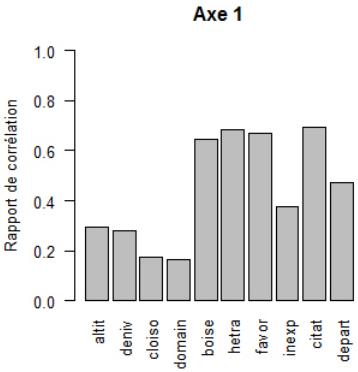
Le tableau `acm$cr` contient les rapports de corrélation (variant de 0 à 1) entre les variables et les axes choisis au départ de l'ACM. Pour représenter graphiquement ces rapports, utiliser `barplot(acm$cr[,num],names.arg=row.names(acm$cr),las=2)` où `num` est le numéro de l'axe à représenter. Pour l'interprétation des axes, se concentrer sur les variables les plus structurantes, *i.e.* dont le rapport de corrélation est le plus proche de 1.

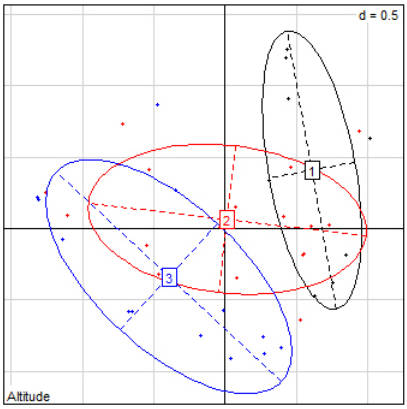
Une aide à l'interprétation est fournie par la fonction `score.acm(acm,xax=num)` où `num` est le numéro de l'axe à représenter. Le graphique montre la répartition des modalités de chaque variable sur l'axe choisi. Pour sélectionner les variables à représenter, ajouter l'argument `which.var=variables` où `variables` est un vecteur contenant le numéro des variables choisies, *i.e.* le numéro des colonnes correspondantes dans le tableau initial.

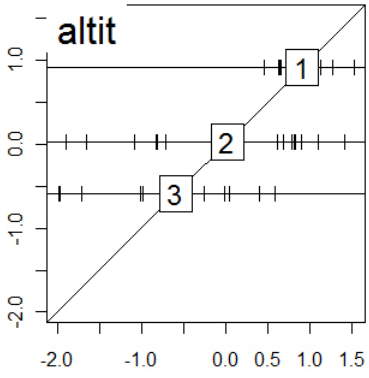
L'ACM est clairement une analyse plus difficile à interpréter que l'ACP ou l'AFC. Aussi, il est bon de limiter la difficulté en ne considérant pas des dizaines de variables mais en se limitant aux questions essentielles.

De plus, l'ACM est sensible aux modalités contenant un faible effectif et aux variables contenant un grand nombre de modalités. Mieux vaut donc regrouper ces modalités en classes plus larges lorsque l'un de ces deux cas se présente.


Valeur propre
0.0
0.1
0.2
0.3
0.4
1
4
7
10
13
16
19
22
25
Axes


Axe 1
Rapport de corrélation
0.0
0.2
0.4
0.6
0.8
1.0
altit
deniv
cloiso
domain
boise
hetra
favor
inexp
citat
depart


d = 0.5
1
2
3
Altitude


altit
1
2
3
1.0
0.0
-1.0
-2.0
-2.0
-1.0
0.0
0.5
1.0
1.5

18. L'Analyse mixte de Hill et Smith

Conditions : les variables peuvent être quantitatives ou qualitatives.

Les données doivent être contenues dans un tableau avec en colonnes les variables (avec un titre) et en ligne les individus. La 1ère case du tableau (en haut à gauche) doit être vide.

Le tableau ne doit pas contenir de données manquantes (`NA`). Pour supprimer les lignes qui en possèdent : `tableau2<-na.omit(tableau)`, où `tableau` est le tableau de départ.

Il est indispensable de réaliser une analyse préliminaire de chaque variable afin d'observer la distribution des valeurs (variables quantitatives) ou leur répartition entre les différentes classes (variables qualitatives). Utiliser pour cela les fonctions `hist()`, `plot()` et / ou `boxplot()`, selon les variables (voir fiches **10**, **11** et **13**).

L'analyse mixte est réalisée grâce à la fonction `dudi.mix()` du package `ade4`. Lorsque la commande `amix<-dudi.mix(tableau2)` est passée, **R** renvoie le graphe des valeurs propres (ou pouvoirs de synthèse) associées à chaque variable de synthèse (ou axe) et demande le nombre d'axes à sélectionner. Il est difficile d'établir un critère de choix pour l'analyse mixte, ne pas multiplier les axes est en tout cas bénéfique pour l'interprétation (ce qui est aussi valable pour les autres analyses multivariées). Le pourcentage d'inertie (ou de l'information totale) expliqué par chaque axe est calculé en divisant la valeur propre d'un axe par le nombre d'axes possibles.

Le diagnostic de l'ACM se fait grâce à la fonction `inertia.dudi(amix)`, qui renvoie le tableau `$TOT`. Celui - ci contient la contribution à l'inertie de chaque axe (proportions cumulées dans la colonne `ratio`, multiplier par 100 pour le pourcentage).

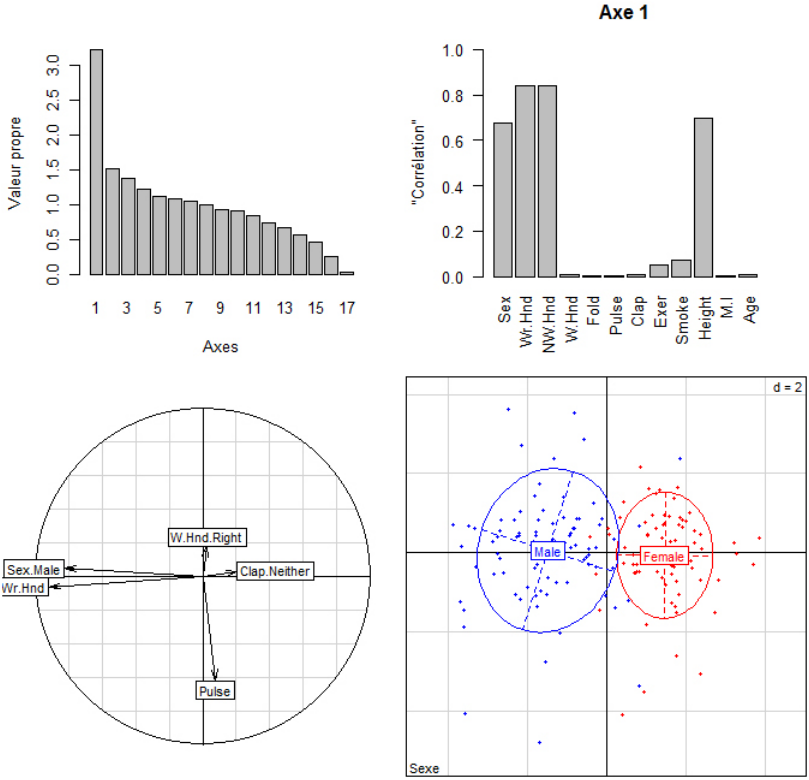
Pour représenter les résultats de l'analyse mixte, la fonction `scatter(amix)` peut être utilisée. Elle représente dans le plan factoriel les individus, les variables quantitatives comme dans une ACP (voir fiche **15**) et les variables qualitatives comme dans une ACM (voir fiche **17**). Pour sélectionner un plan factoriel, ajouter les arguments `xax=num1` et `yax=num2` où `num1` est le numéro du 1er axe choisi et `num2` celui du 2nd (il faut avoir sélectionné au moins deux axes au départ de l'analyse). Par convention on choisit pour l'axe horizontal celui des deux ayant la meilleure contribution à l'inertie totale.

Cette représentation peut cependant être illisible. Il vaut mieux utiliser les fonctions `scat.mix.numeric()` et `scat.mix.categorical()` du package `RVAideMemoire`. La fonction `scat.mix.numeric(amix)` représente les variables quantitatives sur un cercle des corrélation (comme en ACP, voir fiche **15**), tandis que la fonction `scat.mix.categorical(amix)` représente les

variables qualitatives à la manière d'une ACM (voir fiche **17**).

Le tableau `amix$cr` contient les valeurs des « corrélations » (de 0 à 1) qui lient les variables de synthèse aux variables initiales. Pour les représenter graphiquement, utiliser `barplot(amix$cr[,num],names.arg=row.names(amix$cr),las=2)` où `num` est le numéro de l'axe à représenter. Pour l'interprétation des axes, se concentrer sur les variables les plus structurantes, *i.e.* dont le rapport de corrélation est le plus proche de 1.

Une aide à l'interprétation est fournie par la fonction `score(amix,xax=num)` où `num` est le numéro de l'axe à représenter. Là encore la représentation est de type ACP ou ACM selon la nature des variables. Pour sélectionner les variables à représenter, ajouter l'argument `which.var=variables` où `variables` est un vecteur contenant le numéro des variables choisies, *i.e.* le numéro des colonnes correspondantes dans le tableau initial.

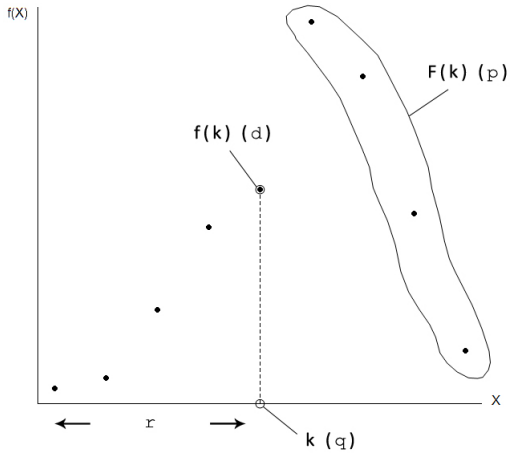


19. Lois de probabilité discontinues – généralités

Ces lois s'appliquent à des variables quantitatives discrètes.

Paramètres :
- k : chaque valeur possible rencontrée dans la population par la variable discrète X. Egalement appelée quantile
- $f(k)$: fréquence, ou probabilité, associée à chaque valeur de la variable discrète X. Egalement appelée distribution de probabilité de X ou fonction de masse de X. Comprise entre 0 et 1
- $F(k)$: somme des probabilités $f(k)$ situées à droite ou à gauche de k, suivant la situation. Egalement appelée fonction de répartition de X. On note $F(k)_{\text{droite}} = P(X > k)$ et $F(k)_{\text{gauche}} = P(X \leq k)$. Comprise entre 0 et 1.

Dans **R** :
- `dY()` : donne la probabilité $f(k)$ pour une distribution de type `Y`
- `pY()` : donne la fonction de répartition $F(k)$ pour une distribution de type `Y`. **R** considère par défaut la répartition à gauche, préciser `lower.tail=FALSE` pour la répartition à droite
- `qY()` : donne la valeur k de la variable X correspondant à une valeur de $F(k)$ pour une distribution de type `Y`. **R** considère par défaut la répartition à gauche de k, préciser `lower.tail=FALSE` pour la répartition à droite
- `rY()` : donne une série de valeurs aléatoires de la variable X pour une distribution de type `Y`.

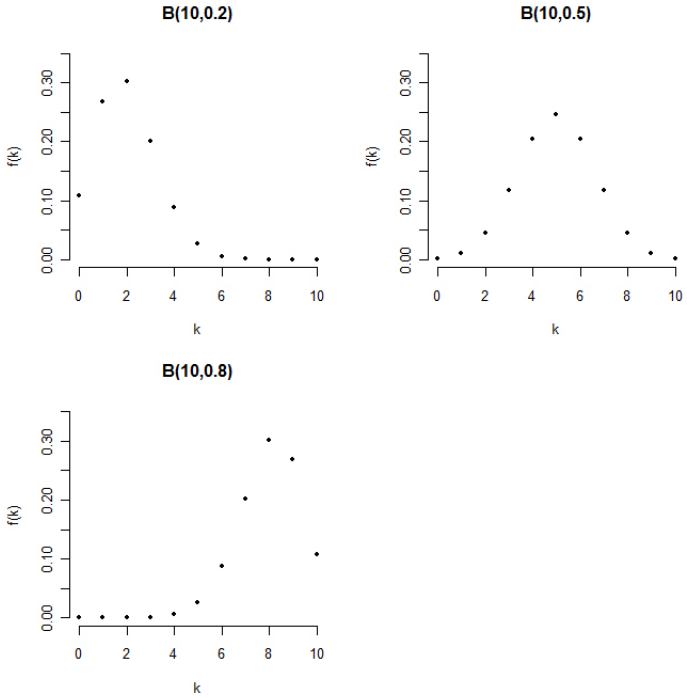


20. La loi binomiale

La loi binomiale est la loi suivie par les résultats de tirages aléatoires lorsqu'il n'y a que deux possibilités mutuellement exclusives de résultat et que la probabilité d'obtenir chaque possibilité est constante au cours de l'expérience (population infinie ou tirages avec remise). La loi donne la probabilité d'obtenir k fois le résultat A quand n tirages sont réalisés.

Ecriture : $B(n, p)$
avec :
n : nombre de tirages
p : probabilité associée au résultat A

Dans **R** :
`dbinom(k,n,p)`
`pbinom(k,n,p)`
`qbinom(F(k),n,p)`
`rbinom(x,n,p)` avec `x` : nombre de valeurs à générer



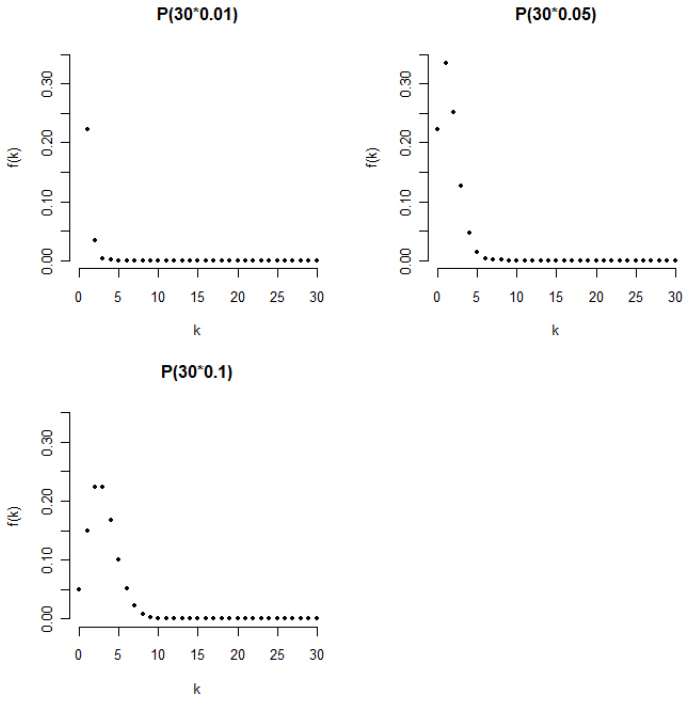
21. La loi de Poisson

La loi de Poisson est une limite de la loi binomiale (voir fiche **20**) quand p tend vers 0 et n vers l'infini (« loi des évènements rares »). L'approximation de la loi binomiale par la loi de Poisson est possible quand quand $p < 0,1$ et $n > 30$.

Sa moyenne est égale à sa variance et vaut np (ou λ).

Ecriture : $P(np)$ ou $P(\lambda)$
avec :
n : nombre de tirages
p : probabilité associée au résultat rare

Dans **R** :
`dpois(k,n*p)` ou `dpois(k,lambda)`
`ppois(k,n*p)` ou `ppois(k,lambda)`
`qpois(F(k),n*p)` ou `qpois(F(k),lambda)`
`rpois(x,n*p)` ou `rpois(x,lambda)` avec `x` : nombre de valeurs à générer

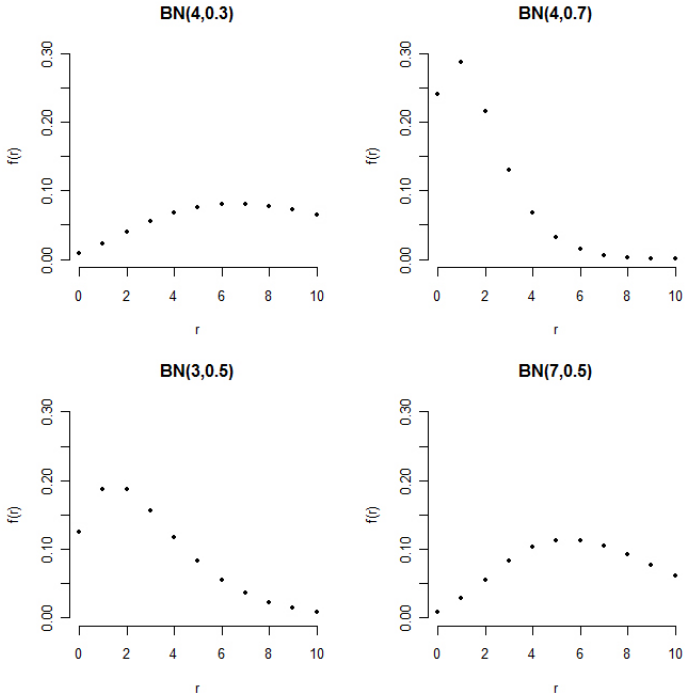


22. La loi binomiale négative

La loi binomiale négative correspond à la même situation que la loi binomiale (voir fiche **20**), mais elle donne la probabilité d'obtenir r résultats B avant d'obtenir k résultats A (approche par l'échec).

Ecriture : $BN(k, p)$
avec :
k : nombre de résultats A désirés
p : probabilité associée au résultat A

Dans **R** :
`dnbinom(r,k,p)`
`pnbinom(r,k,p)`
`qnbinom(F(r),k,p)`
`rnbinom(x,k,p)` avec `x` : nombre de valeurs à générer

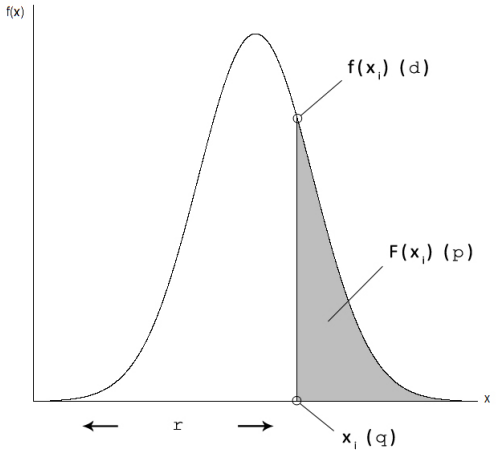


23. Lois de probabilité continues – généralités

Ces lois s'appliquent à des variables quantitatives continues.

Paramètres :
- x_i : chaque valeur possible de la variable continue x. Egalement appelée quantile
- $f(x_i)$: distribution de probabilité de la valeur x_i. Egalement appelée densité de probabilité de x_i. Comprise entre 0 et 1
- $F(x_i)$: aire sous la courbe située à droite ou à gauche de x_i, suivant la situation. Egalement appelée fonction de répartition de x_i. On note $F(x_i)_{\text{droite}} = P(x > x_i)$ et $F(x_i)_{\text{gauche}} = P(x \leq x_i)$. Comprise entre 0 et 1.

Dans **R** :
- `dY()` : donne la densité de probabilité $f(x_i)$ pour une distribution de type `Y`
- `pY()` : donne la fonction de répartition $F(x_i)$ pour une distribution de type `Y`. **R** considère par défaut la répartition à gauche, préciser `lower.tail=FALSE` pour la répartition à droite
- `qY()` : donne la valeur x_i de la variable x correspondant à une valeur de $F(x_i)$ pour une distribution de type `Y`. **R** considère par défaut la répartition à gauche de k, préciser `lower.tail=FALSE` pour la répartition à droite
- `rY()` : donne une série de valeurs aléatoires de la variable x pour une distribution de type `Y`.

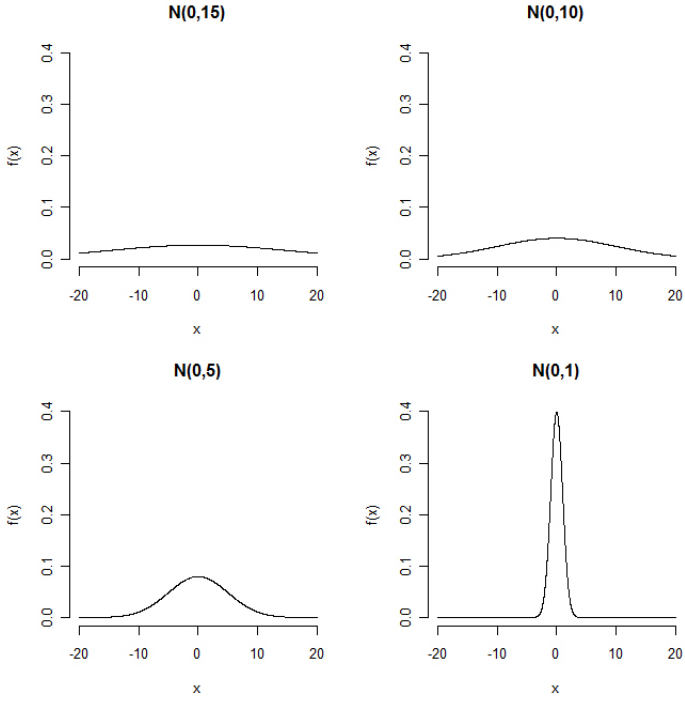


24. La loi normale

Ecriture : $N(\mu, \sigma)$
avec :
μ : moyenne de la variable x
σ : écart-type de la variable x

Cas particulier, la loi normale centrée - réduite : $N(0, 1)$

Dans **R** :
`dnorm(xi,mu,sigma)`
`pnorm(xi,mu,sigma)`
`qnorm(F(xi),mu,sigma)`
`rnorm(z,mu,sigma)` avec `z` : nombre de valeurs à générer

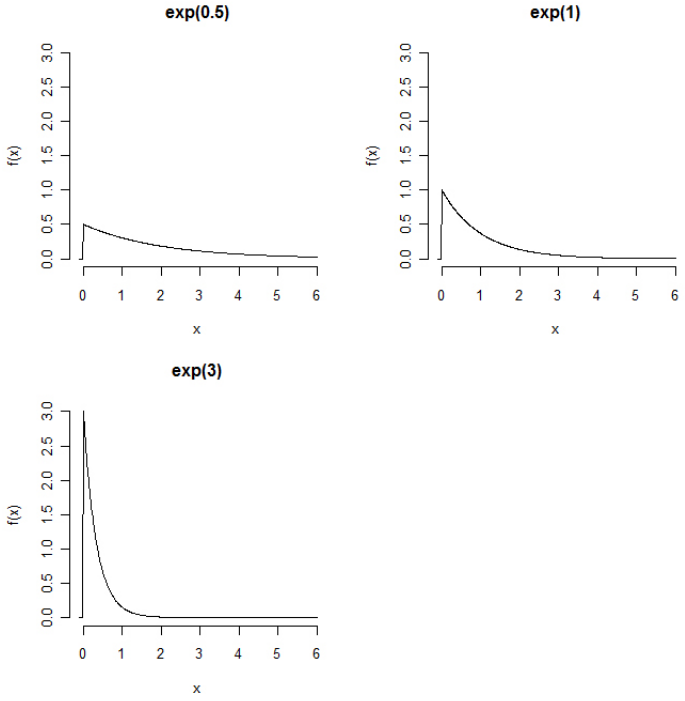


25. La loi exponentielle

La loi exponentielle correspond souvent à des évènements dont la probabilité de survenue diminue avec le temps. Elle est également utilisée pour modéliser des durées de vie.

Ecriture : $exp(\lambda)$
avec λ : paramètre de la loi $(0 < \lambda < +\infty)$

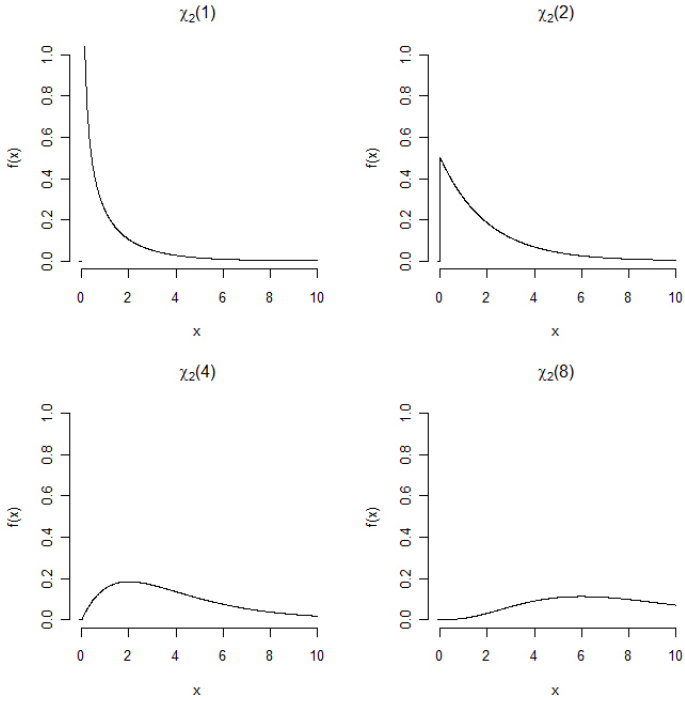
Dans **R** :
`dexp(xi,lambda)`
`pexp(xi,lambda)`
`qexp(F(xi),lambda)`
`rexp(z,lambda)` avec `z` : nombre de valeurs à générer



26. La loi de χ^2

Ecriture : $\chi^2(\nu)$
avec ν : nombre de degrés de liberté (ddl), *i.e.* de paramètres indépendants impliqués dans la loi $(0 < \nu < +\infty)$

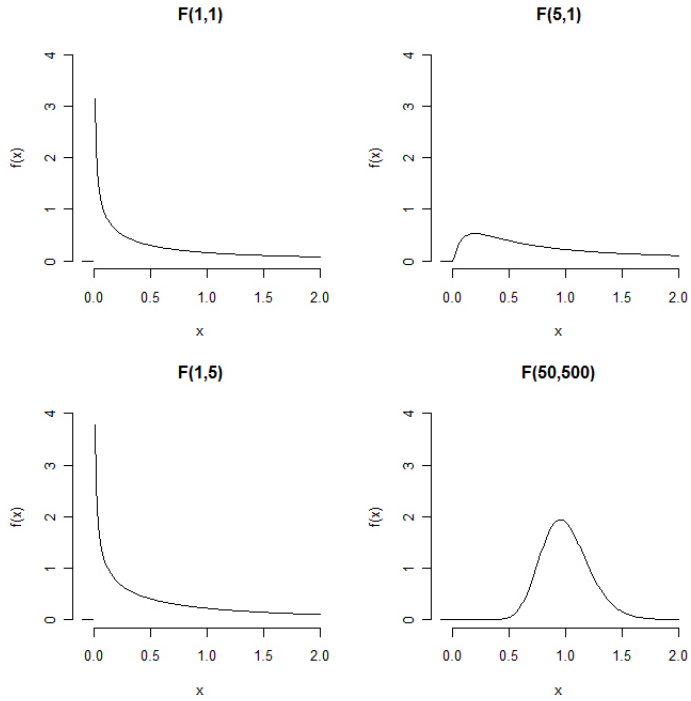
Dans **R** :
`dchisq(xi,ddl)`
`pchisq(xi,ddl)`
`qchisq(F(xi),ddl)`
`rchisq(z,ddl)` avec `z` : nombre de valeurs à générer



27. La loi de Fisher - Snedecor

Ecriture : $F(\nu_1, \nu_2)$
avec :
ν_1 : 1^{er} nombre de degrés de liberté (ddl) $(0 < \nu_1 < +\infty)$
ν_2 : $2^{\text{ème}}$ nombre de ddl $(0 < \nu_2 < +\infty)$

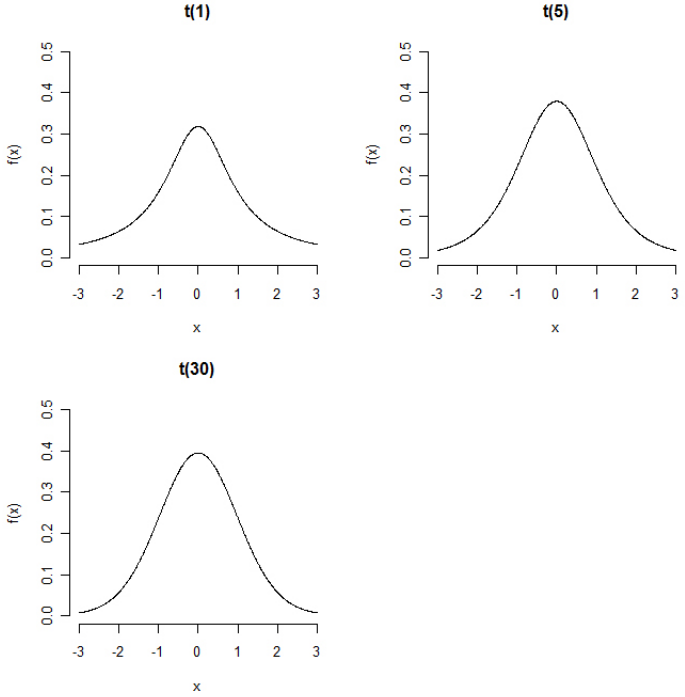
Dans **R** :
`df(xi,ddl1,ddl2)`
`pf(xi,ddl1,ddl2)`
`qf(F(xi),ddl1,ddl2)`
`rf(z,ddl1,ddl2)` avec `z` : nombre de valeurs à générer



28. La loi de Student

Ecriture : $t(\nu)$
avec ν : nombre de degrés de liberté (ddl) $(0 < \nu < +\infty)$

Dans **R** :
`dt(xi,ddl)`
`pt(xi,ddl)`
`qt(F(xi),ddl)`
`rt(z,ddl)` avec `z` : nombre de valeurs à générer



29. Principe des tests statistiques et risques associés à la conclusion

Principe de réalisation des tests statistiques :

1. poser une hypothèse nulle H_0, de type « rien à signaler » (ex : les moyennes μ_A et μ_B sont égales) ou valeur ponctuelle (ex : $\mu = 10$, $\mu = 50$ %)
2. poser une hypothèse H_1, de telle manière que H_0 et H_1 soient exclusives (ex : les moyennes μ_A et μ_B sont différentes)
3. calculer la valeur de la Variable de Test (VT)
4. utiliser la valeur de la VT calculée pour déterminer une p - *value*, *i.e.* une probabilité d'obtenir la valeur mesurée (moyenne, pourcentage...) si H_0 est vraie
5. conclure sur les deux hypothèses posées grâce à cette p - *value* :
   - si la p - *value* est supérieure au seuil α fixé avant le test (5 % en général, voir fiche **30**), ne pas rejeter H_0 (donc rejeter H_1)
   - si la p - *value* est inférieure au seuil α, rejeter H_0 (donc accepter H_1).

Conclure sur les deux hypothèses présente deux risques :
- le risque de 1ère espèce ou risque α : risque de rejeter de H_0 si celle - ci est vraie
- le risque de 2ème espèce ou risque β : risque de ne pas rejeter de H_0 si celle - ci est fausse.

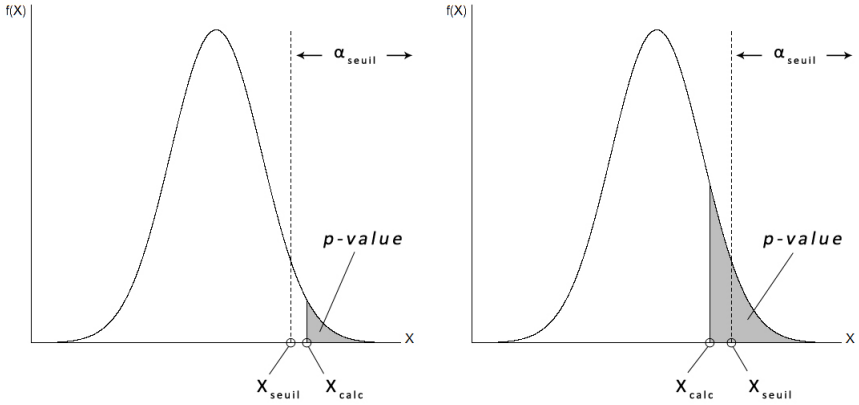
	Réalité (inconnue le plus souvent)	
Décision	H_0 vraie	H_0 fausse
H_0 non rejetée	Bonne décision	Erreur β
H_0 rejetée	Erreur α	Bonne décision

La probabilité associée au fait de rejeter H_0 si celle - ci est fausse (soit $1-\beta$) est appelée puissance du test.

30. Le risque ou seuil de rejet α

Il existe deux définitions du risque α :

- le seuil de rejet (ou seuil de signification) de l'hypothèse H_0 : si la p - *value* du test réalisé est inférieure à ce seuil, l'hypothèse H_0 est rejetée (la p - *value* est dite significative), dans le cas contraire H_0 n'est pas rejetée (la p - *value* est dite non significative). Habituellement fixé à 5 %, dans tous les cas il doit être fixé avant la réalisation du test statistique
- la p - *value* du test statistique : le risque α est égal à la p - *value*, *i.e.* en rejetant H_0 on prend le risque p de se tromper.



X_{seuil} : valeur de la Variable de Test (VT) X qui donne une fonction de répartition à droite égale au seuil α (test unilatéral droit).

X_{calc} : valeur de la VT X calculée à partir de l'échantillon testé.

A gauche l'hypothèse H_0 est rejetée, à droite elle ne l'est pas.

31. La correction du seuil de rejet α

Si une série de tests statistiques est réalisée, avec à chaque fois α pour seuil de rejet de H_0 (voir fiche **30**), le risque global de rejeter H_0 si celle - ci est vraie augmente. En effet, plus on effectue de tests, plus on a de chance de tomber sur un échantillon peu représentatif de la population dont il provient (donnant une p - *value* inférieure au α_{seuil}).

Il est donc nécessaire de corriger le seuil de rejet α de chaque test lorsque plusieurs sont réalisés, afin que le risque global soit égal au α souhaité.

Cette situation se présente :

- lorsque les tests vont permettre de prendre une décision unique, dès que l'un d'eux au moins permet le rejet de H_0
- lorsque sont réalisés une série de tests deux - à - deux, soit directement soit après une analyse globale (ANOVA, analyse de déviance, test du χ^2 d'homogénéité...).

Plusieurs méthodes de correction existent, dont les trois suivantes :

La technique de Bonferroni

Si k tests sont effectués, la technique consiste simplement à diviser le seuil de rejet global α par k, donc considérer pour chaque test le seuil de rejet $\frac{\alpha}{k}$.

La technique séquentielle de Holm

La procédure se réalise en plusieurs étapes :

1. classer les p - *values* de tous les tests réalisés par ordre croissant ($p_1 <...< p_k$), k étant le nombre de tests effectués
2. rejeter H_0 pour les tests dont la p - *value* satisfait la condition :

$$p_i \leq \frac{\alpha_{\text{seuil}}}{k+1-i}$$

où i est le rang de la p - *value* après classement.

La technique du False Discovery Rate (FDR) de Benjamini et Hochberg

La procédure se réalise en plusieurs étapes :

1. classer les p - *values* de tous les tests réalisés par ordre croissant ($p_1 <...< p_k$), k étant le nombre de tests effectués
2. rejeter H_0 pour les tests dont la p - *value* satisfait la condition :

$$p_i \leq \alpha_{\text{seuil}} \times \frac{i}{k}$$

où i est le rang de la p - *value* après classement.

La technique la plus stricte est celle de Bonferroni, la moins stricte celle du FDR. Cette dernière peut être appliquée par défaut. Dans tous les cas la méthode de correction du seuil de rejet de H_0 doit être décidée avant de réaliser les tests.

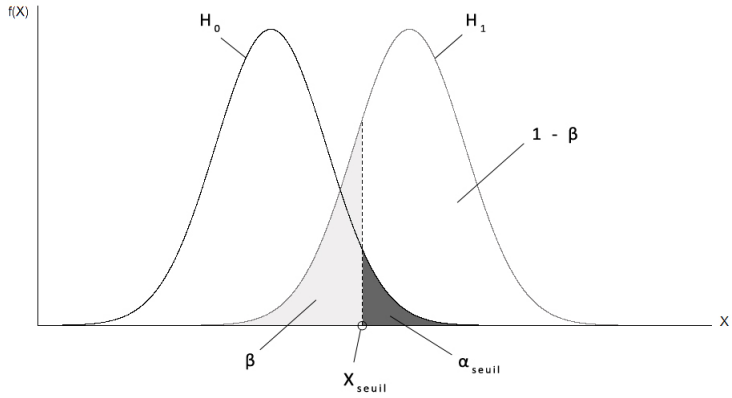
Dans **R**, si `p` est le vecteur contenant les p - *values* non corrigées, utiliser la fonction `p.adjust()` pour récupérer un vecteur avec les p - *values* corrigées (dans le même ordre) :
`p.adjust(p,method="bonferroni")` pour la correction de Bonferroni
`p.adjust(p,method="holm")` pour la correction de Holm
`p.adjust(p,method="BH")` ou `p.adjust(p,method="fdr")` pour la correction de Benjamini et Hochberg (FDR).

32. Le risque β et la puissance du test

Le risque β est le risque de considérer l'hypothèse H_0 comme acceptable (*i.e.* ne pas la rejeter) si celle - ci est fausse. Contrairement au risque α, le risque β ne peut pas être fixé. En effet, si α dépend de la distribution de la Variable de Test (VT) sous H_0 (voir fiche **30**), β dépend de sa distribution sous H_1. Or cette distribution est inconnue, puisque l'hypothèse H_1 regroupe une infinité de distributions (ex : si l'hypothèse H_1 est $\mu_A \neq \mu_B$, les deux moyennes peuvent différer d'une infinité de façons).

La puissance d'un test représente la probabilité de rejeter H_0 si celle - ci est fausse (*i.e.* de faire le bon choix). Elle équivaut à $1 - \beta$, et est donc également une variable dépendant de la distribution de la VT sous H_1, inconnue.

Le risque β et la puissance du test dépendent du seuil α fixé :



X_{seuil} : valeur de la VT X qui donne une fonction de répartition à droite égale au seuil α pour la distribution sous H_0 (test unilatéral droit).

La puissance d'un test augmente :
- quand augmente le seuil α
- quand augmente l'effectif de l'échantillon testé (ce qui diminue l'étalement de la distribution de la VT ou éloigne les distributions de la VT sous H_0 et H_1, selon le test)
- quand augmente l'écart réel entre les paramètres (moyennes, proportions...) testés.

33. L'identification des données aberrantes

L'identification des valeurs aberrantes doit se faire avant tout visuellement (voir fiches **9** et **11**), cependant certains tests peuvent apporter une aide. L'élimination d'une valeur d'une série de données doit toutefois se faire sur des considérations techniques (erreur de mesure ou de retranscription de la mesure) ou biologiques (valeur improbable pour la variable mesurée). Dans tous les cas, l'élimination des données aberrantes doit se faire avant tout autre test.

Les fonctions suivantes sont toutes contenues dans le package `outliers`.

Identification d'une valeur au sein d'une distribution

Test du χ^2 : `chisq.out.test(serie)` où `serie` est un vecteur contenant une série de données. La fonction teste la valeur extrême de la série la plus éloignée de la moyenne. Pour tester l'autre valeur extrême, ajouter l'argument `opposite=TRUE`.

Test de Dixon : la fonction `dixon.test()` a la même syntaxe que la précédente, avec le même argument `opposite`.

Identification d'une moyenne au sein d'un groupe de moyennes

Test de Grubbs : `grubbs.test(moyennes)` où `moyennes` est un vecteur contenant la moyenne de chaque série de données. L'argument `opposite` peut aussi être utilisé. Pour travailler directement avec un tableau de données, utiliser `grubbs.test(tapply(variable,facteur,mean))`. La fonction `tapply()` calcule la moyenne de la variable `variable` par modalité du facteur `facteur`.

Identification d'une variance au sein d'un groupe de variances

Test de Cochran : `cochran.test(variable~facteur)`. La fonction teste la variance la plus élevée, pour tester la plus faible ajouter l'argument `inlying=TRUE`. Les deux vecteurs ne doivent pas contenir de données manquantes (`NA`).

34. Intervalle de confiance et erreur standard

Moyennes

Intervalle de confiance :

- petit effectif (< 30 individus) : utiliser le module « intervalle de confiance » du test de Mann - Whitney - Wilcoxon (test non paramétrique qui calcule en fait l'intervalle de confiance de la *médiane*) : `wilcox.test(serie,conf.int=TRUE)$conf.int` où `serie` est un vecteur contenant la série de données. Si les conditions du test de Mann - Whitney - Wilcoxon sont réunies (voir fiche **54**), médiane et moyenne sont proches
- grand effectif (> 30 individus) : utiliser le module « intervalle de confiance » du test t de Student (test paramétrique) : `t.test(serie)$conf.int`.

Erreur standard : utiliser la fonction `se()` du package `RVAideMemoire` : `se(serie)`.

Pourcentages

Quel que soit l'effectif, utiliser le module « intervalle de confiance » du test binomial exact : `binom.test(a,b)$conf.int` où `a` est le nombre d'individus de la catégorie d'intérêt et `b` l'effectif total (ex : 9 femelles sur 25 individus).

N'importe quoi

Utiliser la fonction `bootstrap()` du package `RVAideMemoire`, qui est basée sur la technique du bootstrap : `bootstrap(serie,function(x,i) mean(x[i]))`. Dans cet exemple une moyenne est calculée, mais la fonction gère des expressions bien plus complexes. La syntaxe particulière de cette fonction doit obéir à deux règles :

- l'argument utilisé dans l'expression à calculer (ici cette expression est `mean(x[i])`) doit être le même que le 1er déclaré à `function()`. Ici cet argument est `x`
- l'argument utilisé dans l'expression à calculer doit toujours être suivi du second déclaré à `function()` placé entre crochets. Celui - ci est par défaut `i`, voir l'aide de la fonction `boot()` pour plus d'informations.

Pour toutes ces fonctions, la précision de l'intervalle de confiance peut être modifiée grâce à l'argument `conf.level` (par défaut `conf.level=0.95`).

35. ℹ Tracer un diagramme en barres avec barres d'erreur

L'exemple présenté ici traite de moyennes et de barres d'erreur représentant des erreurs standards. Il peut bien sûr être adapté à n'importe quelles valeurs.

Un facteur

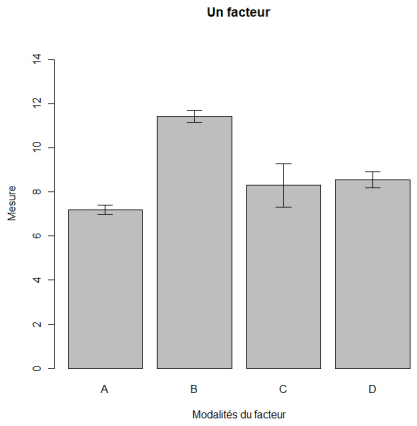
L'étape préliminaire est de rassembler les moyennes (une par modalité du facteur) dans un vecteur `moyennes` (contenant les valeurs dans l'ordre du graphe, de gauche à droite) et les erreurs standards (voir fiche **34**) dans un vecteur `erreurs` (avec les valeurs dans le même ordre que les moyennes).
La procédure est ensuite la suivante :

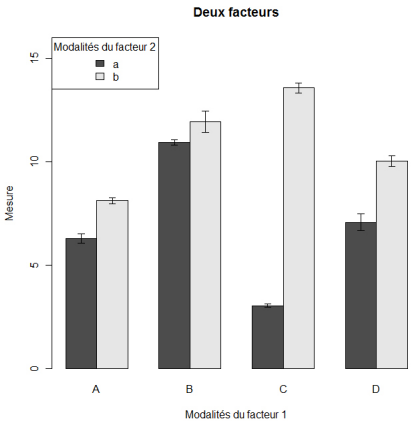
```
> abscisses<-barplot(moyennes)
> segments(abscisses,moyennes-erreurs,abscisses,moyennes+erreurs)
> segments(abscisses-0.1,moyennes-erreurs,abscisses+0.1,moyennes-erreurs)
> segments(abscisses-0.1,moyennes+erreurs,abscisses+0.1,moyennes+erreurs)
```

Deux facteurs

Moyennes et erreurs standards doivent être contenues dans des matrices. Ces matrices doivent avoir en lignes les modalités du 2nd facteur et en colonnes celles du 1er facteur. La procédure est ensuite identique, il faut seulement ajouter l'argument `beside=TRUE` à la fonction `barplot()`.

L'argument `names.arg=noms` de la fonction `barplot()` ajoute ou modifie le nom des barres (`noms` étant un vecteur contenant les noms de gauche à droite), tandis que `legend=TRUE` ajoute une légende dans le cas de deux facteurs.





36. Les différents types de test statistique

Il existe quatre grandes familles de tests statistiques, qui se différencient par leur objectif :

- les tests d'homogénéité : ils permettent de tester l'égalité d'un paramètre entre plusieurs populations (ex : égalité des moyennes entre n populations)
- les tests de conformité : ils permettent de tester l'égalité d'un paramètres d'une (ou plusieurs) population(s) à une (ou plusieurs) valeur(s) théorique(s) (ex : sex - ratio équilibré 1 : 1)
- les tests d'indépendance : ils permettent de tester si deux variables mesurées sont indépendantes l'une par rapport à l'autre
- les tests d'ajustement : ils permettent de tester si une distribution de valeurs observée est conforme à une distribution théorique ou si deux distributions observées sont identiques.

Ces quatre catégories ne sont pas exclusives, un même test pouvant avoir plusieurs objectifs (ex : le test du χ^2 de conformité est aussi un test d'ajustement).

Le choix d'un test statistique repose sur plusieurs critères :

- son objectif : à quel question doit - il permettre de répondre ?
- le type de données :
  - comment ont - elles été récoltées (voir fiches **2** et **3**) ?
  - combien de variables ont été mesurées ?
  - quelle est la nature de ces variables (voir fiche **1**) ?
  - comment ces variables sont - elles distribuées ?
- la quantité de données : combien d'individus ont été comptés / mesurés ?
- le nombre de populations à comparer.

La réponse à toutes ces questions peut généralement être apportée avant même de récolter les données, dès la préparation de l'étude. Une bonne démarche consiste ainsi à choisir dès le départ les tests qui vont être utilisés. Ce choix ne pose pas de problèmes si les questions auxquelles l'étude doit répondre sont clairement identifiées.

37. Caractère aléatoire et simple d'une série de données

Cette condition est l'une des plus fondamentales d'un grand nombre de tests statistiques. On peut la remplir dès la mise en place du plan d'échantillonnage (voir fiche **2**) ou du plan d'expérience (voir fiche **3**). On peut également la tester grâce aux deux tests non paramétriques suivants, dont les fonctions sont contenues dans le package `lawstat` :

Test des runs : `runs.test(serie)` où `serie` est un vecteur contenant des données numériques, dans l'ordre où elles ont été récoltées.

Test de Bartels : `bartels.test(serie)` où `serie` est un vecteur contenant des données numériques ou non (ex : modalités d'un facteur), dans l'ordre où elles ont été récoltées.

Une p - *value* significative indique qu'il existe une corrélation entre les valeurs de la série, négative ou positive (par défaut les deux tests sont bilatéraux). Pour connaître le signe de cette corrélation, réaliser un test unilatéral en ajoutant l'argument `alternative="positive.correlated"` ou `alternative="negative.correlated"`. Dans ce cas l'hypothèse alternative H_1 n'est plus « il y a une corrélation entre les valeurs » mais « il y a une corrélation positive » ou « il y a une corrélation négative » entre les valeurs.

38. Ajustement à une distribution théorique

Ce test peut être une fin en soi mais est souvent préliminaire à d'autres tests. La condition qu'une distribution observée suive une loi normale est en particulier très fréquemment requise.

Avant de réaliser le test, la première analyse doit être graphique et passe par l'observation du graphe quantile - quantile. Il est tracé de la manière suivante : `qqplot(serie,qloi(ppoints(serie),par))` où `serie` est un vecteur contenant la série de données, `loi` est la loi théorique choisie et `par` ses paramètres séparés par une virgule (voir fiches **19** à **28**). La distribution de la série suit la loi théorique choisie si les points du graphe sont alignés sur une droite. Toute autre structuration des points (courbures, nombreux points éloignés...) indique le contraire.

Dans le cas d'une loi normale, le même résultat est obtenu directement *via* la fonction `qqnorm(serie)`.

Ajustement à une loi normale – Test de Shapiro - Wilk

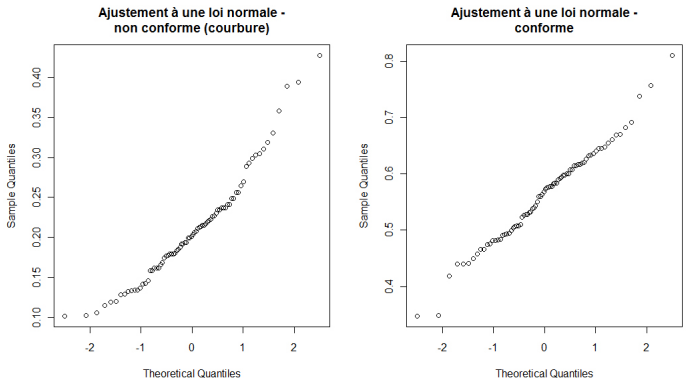
Conditions : l'échantillonnage doit être aléatoire et simple ; la variable testée doit être quantitative.

Pour réaliser le test : `shapiro.test(serie)`.

Ajustement à une loi autre que normale – Test de Kolmogorov - Smirnov

Conditions : l'échantillonnage doit être aléatoire et simple ; la variable testée doit être quantitative ; les paramètres de la loi théorique ne doivent pas provenir de l'échantillon testé.

Pour réaliser le test : `ks.test(serie,ploi,par)` où `loi` est la loi choisie et `par` ses paramètres séparés par une virgule. Ce test peut être utilisé pour une loi normale mais il est moins puissant que celui de Shapiro - Wilk.



39. Egalité des variances de plusieurs séries de données

Ce test peut être une fin en soi mais est souvent préliminaire à d'autres tests. L'égalité des variances (ou homoscédasticité) doit en effet au minimum être contrôlée, au maximum avérée dans de nombreux tests usuels.

Comparaison des variances de deux populations

Test de Fisher (paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple ; les individus ne doivent pas pouvoir être dans les deux échantillons à la fois ; la distribution des valeurs dans chaque échantillon doit être normale.

Pour réaliser le test : `var.test(serie1,serie2)` où `serie1` et `serie2` sont des vecteurs contenant les deux séries de données à comparer. Si les deux séries correspondent aux deux modalités d'un facteur, la syntaxe peut être : `var.test(variable~facteur)` où `facteur` est un vecteur contenant la valeur de chaque individu (dans le même ordre que `variable`).

Ce test est très sensible à la condition de normalité. De plus prendre garde aux individus extrêmes qui ont une grande influence sur la variance.

Test de Fligner - Killeen (non paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple ; les individus ne doivent pas pouvoir être dans les deux échantillons à la fois.

Pour réaliser le test : `fligner.test(variable,facteur)` ou `fligner.test(variable~facteur)`.

Comparaison des variances de plus de deux populations

Test de Bartlett (paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple ; les individus ne doivent pas pouvoir être dans plusieurs échantillons à la fois ; la distribution des valeurs dans chaque échantillon doit être normale ; l'effectif de chaque échantillon doit être au moins égal à 4.

Pour réaliser le test : `bartlett.test(variable,facteur)` ou `bartlett.test(variable~facteur)`.

Ce test est très sensible aux conditions de normalité et d'effectif minimum. De plus prendre garde aux individus extrêmes qui ont une grande influence sur la variance.

Test de Fligner - Killeen (non paramétrique)

La procédure est identique au cas de deux populations.

40. Les transformations de variable

Les tests paramétriques sont en général plus puissants que leurs homologues non paramétriques. Cependant ils ont des conditions d'application plus rigoureuses (notamment normalité des distributions et homoscédasticité). Lorsque celles - ci ne sont pas remplies, il y a deux possibilités : (i) utiliser un test homologue non paramétrique moins puissant ou (ii) transformer la variable pour satisfaire la ou les conditions non remplie(s) et utiliser le test paramétrique. La conclusion d'un test sur des données transformées peut en effet s'appliquer aux données initiales.

Les transformations usuelles sont les suivantes :

- logarithmique : `variable2<-log10(variable)`. A utiliser lorsque la moyenne des échantillons (si plusieurs sont disponibles) est proportionnelle à leur écart - type. En histogramme cela donne un fort biais à gauche de la distribution. Utilisée fréquemment lorsque des processus de croissance ou de multiplication sont en jeu : masse, rendement, décompte d'organismes vivants...
- racine carrée : `variable2<-sqrt(variable)`. A utiliser lorsque la moyenne des échantillons est proportionnelle à leur variance (ce qui est exactement le cas pour les distributions de Poisson). En histogramme cela donne un léger biais à gauche
- inverse : `variable2<-1/variable`. A utiliser lorsque la variable est distribuée « en i », *i.e.* lorsque les valeurs les plus faibles sont les plus fréquentes
- angulaire : `variable2<-asin(sqrt(variable))`. A utiliser lorsque la variable suit une distribution binomiale (proportions).

Il existe un très grand nombre de transformations. Elles peuvent être utilisées sur des bases théoriques ou de façon empirique. Dans tous les cas, chercher une transformation complexe qui permettra de satisfaire une condition particulière d'un test n'est pas conseillé. Si une transformation simple ne fonctionne pas, c'est que les données sont vraiment « particulières ». Il convient alors (i) de chercher d'où vient cette particularité, ce qui peut être biologiquement intéressant et (ii) d'utiliser un test non paramétrique.

Une fois qu'une transformation a été appliquée, il est nécessaire de repasser par un examen graphique des données (voir fiches **10**, **11** et **13**) et par le(s) test(s) qui avai(en)t conduit à les transformer (le plus souvent ajustement à une distribution théorique et /ou homoscédasticité, voir fiches **38** et **39**).

41. Comparaison de plusieurs probabilités de réponse – un facteur

Séries non appariées

*Analyse de dévience en Modèle Linéaire Généralisé (*GLM *; paramétrique)*

Conditions : l'échantillonnage doit être aléatoire et simple ; les classes du facteur doivent être exclusives.

Commencer par écrire le modèle censé représenter les données : `modele<-glm(formule,family=binomial)` où `formule` est la formule contenant la variable à expliquer et le facteur (voir fiche **71**). La loi binomiale est celle à utiliser lorsque la variable à expliquer est binaire (0 / 1).

Réaliser ensuite l'analyse de déviance *via* `anova(modele,test="Chi")`. Le tableau renvoyé donne l'effet du facteur et la p - *value* associée. Si cette p - *value* est significative, cela indique qu'au moins deux classes du facteur ont un effet différent sur la variable à expliquer (sans préciser lesquelles). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les classes en question. Ces comparaisons passent par l'utilisation de la méthode des contrastes (voir fiche **74**).

Test du χ^2 d'homogénéité (ou d'indépendance ; non paramétrique)

La démarche est exactement la même que dans le cas de comparaison de plusieurs proportions sans répétition (voir fiche **50**). Les conditions d'application du test sont donc identiques.

Commencer par créer le tableau de contingence sur lequel le test doit s'appliquer : `tab.cont<-table(facteur,relevel(reponse,ref="1"))` où `reponse` est un vecteur contenant la réponse binaire de chaque individu et `facteur` un vecteur contenant la classe de chaque individu (dans le même ordre que `reponse`).

Réaliser ensuite le test : `prop.test(tab.cont)`.

Une p - *value* significative indique qu'au moins deux classes du facteur ont un effet différent sur la variable à expliquer (sans préciser lesquelles). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les classes en question. Utiliser pour cela la fonction `pairwise.prop.test(tab.cont,p.adjust.method=methode)` (voir fiche **31** pour choisir la méthode de correction du seuil de rejet α).

Séries appariées

*Test du rapport des vraisemblances en Modèle Linéaire Généralisé Mixte (*GLMM *; paramétrique)*

Conditions : l'échantillonnage doit être aléatoire et simple au sein des classes du facteur aléatoire ; les classes du facteur fixe et du facteur aléatoire doivent être exclusives.

Commencer par écrire le modèle censé représenter les données, *via* la fonction `glmer()` du package `lme4` : `modele<-glmer(formule,family=binomial)`

où `formule` est la formule contenant la variable à expliquer, le facteur fixe et le facteur aléatoire (voir fiche **71**). La loi binomiale est celle à utiliser lorsque la variable à expliquer est binaire (0 / 1).

Créer un second modèle, dit *nul*, appelé `modele.nul` (voir fiche **71**).

Il n'est pas possible de réaliser une analyse de déviance comme avec les modèles à facteur fixe. La démarche avec les modèles mixtes est la comparaison de modèles par le test du rapport des vraisemblances. Le principe est de comparer le modèle contenant le facteur d'intérêt avec un modèle identique en tout point sauf qu'il ne comprend pas ce facteur. La *p - value* du test correspond à celle du facteur qui a été enlevé dans le second modèle. Pour réaliser le test : `anova(modele,modele.nul)`.

Une *p - value* significative indique qu'au moins deux classes du facteur fixe ont un effet différent sur la variable à expliquer (sans préciser lesquelles). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les classes en question. Ces comparaisons passent par l'utilisation de la méthode des contrastes (voir fiche **74**).

Test Q de Cochran (non paramétrique)

Conditions : : le plan d'expérience doit être en blocs aléatoires complets, avec une seule observation par modalité du facteur au sein de chaque bloc (*i.e.* de chaque classe du facteur aléatoire) ; l'échantillonnage doit être aléatoire et simple au sein des classes du facteur aléatoire ; les classes du facteur fixe et du facteur aléatoire doivent être exclusives.

Pour réaliser le test, utiliser la fonction `cochran.qtest()` du package `RVAideMemoire` : `cochran.qtest(reponse,facteur.fixe,facteur.aleatoire)`.

Si la *p - value* du test est significative, cela indique qu'au moins deux classes du facteur fixe ont un effet différent sur la variable à expliquer (sans préciser lesquelles). La fonction réalise alors automatiquement toutes les comparaisons deux - à - deux possibles par le test des signes de Wilcoxon.

42. Comparaison de plusieurs probabilités de réponse – deux facteurs

Séries non appariées

*Analyse de déviance en Modèle Linéaire Généralisé (*GLM *; paramétrique)*

Conditions : l'échantillonnage doit être aléatoire et simple ; les classes des facteurs doivent être exclusives.

Commencer par écrire le modèle censé représenter les données : `modele<-glm(formule,family=binomial)` où `formule` est la formule contenant la variable à expliquer et les deux facteurs (voir fiche **71**). La loi binomiale est celle à utiliser lorsque la variable à expliquer est binaire (0 / 1).

Réaliser ensuite l'analyse de déviance *via* `anova(modele,test="Chi")`. Le tableau renvoyé donne l'effet de chaque facteur (et de leur interaction si elle est prise en compte) et la *p - value* associée. Si une *p - value* est significative, cela indique qu'au moins deux classes du facteur en question (ou au moins deux combinaisons de classes des deux facteurs si c'est l'effet de l'interaction qui est significatif) ont un effet différent sur la variable à expliquer (sans préciser lesquelles). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les classes (ou combinaisons de classes) en question. Ces comparaisons passent par l'utilisation de la méthode des contrastes (voir fiche **74**).

Séries appariées

*Test du rapport des vraisemblances en Modèle Linéaire Généralisé Mixte (*GLMM *; paramétrique)*

Conditions : l'échantillonnage doit être aléatoire et simple au sein des classes du facteur aléatoire ; les classes des facteurs fixes et du facteur aléatoire doivent être exclusives.

Commencer par écrire le modèle censé représenter les données, *via* la fonction `glmer()` du package `lme4` : `modele<-glmer(formule,family=binomial)` où `formule` est la formule contenant la variable à expliquer, les deux facteurs fixes et le facteur aléatoire (voir fiche **71**). La loi binomiale est celle à utiliser lorsque la variable à expliquer est binaire (0 / 1).

Créer ensuite deux modèles dits *réduits* : l'un sans le premier facteur (mais avec l'interaction si elle est prise en compte) et l'autre sans le second facteur (mais avec l'interaction si elle est prise en compte). Si l'interaction est prise en compte dans le modèle initial, créer également un modèle additif, avec les deux facteurs mais sans leur interaction.

Il n'est pas possible de réaliser une analyse de déviance comme avec les modèles à facteurs fixes. La démarche avec les modèles mixtes est la comparaison de modèles par le test du rapport des vraisemblances. Le principe est de comparer le modèle contenant le facteur d'intérêt (ou l'interaction d'intérêt) avec un modèle identique en tout point sauf qu'il ne comprend pas ce facteur (ou cette interaction). La *p - value* du test correspond à celle du facteur

(ou de l'interaction) qui a été enlevé(e) dans le second modèle. Réaliser donc une série de tests de la forme : `anova(modele,modele.reduit)`, en comparant chaque fois le modèle complet avec l'un des deux ou trois modèles réduits.

Une p - *value* significative indique qu'au moins deux classes du facteur fixe testé (ou au moins deux combinaisons de classes des deux facteurs si c'est l'interaction qui est testée) ont un effet différent sur la variable à expliquer (sans préciser lesquelles). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les classes (ou combinaisons de classes) en question. Ces comparaisons passent par l'utilisation de la méthode des contrastes (voir fiche **74**).

43. Conformité de plusieurs effectifs avec des valeurs théoriques

Test du χ^2 de conformité (non paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple ; les individus ne doivent pas pouvoir être décomptés dans plusieurs effectifs à la fois ; au moins 80 % des effectifs théoriques doivent être non nuls et ≥ 5 (voir ci - dessous pour obtenir les effectifs théoriques).

Pour réaliser le test : `chisq.test(effectifs,p=prop.theo)` où `effectifs` est un vecteur contenant les effectifs et `prop.theo` un vecteur contenant les proportions théoriques (et *non pas* les effectifs théoriques), dans le même ordre qu'`effectifs`. La somme de ces proportions doit être égale à 1.

Les effectifs théoriques sont donnés par la fonction `chisq.test(effectifs,p=prop.theo)$expected`.

Une *p - value* significative indique qu'au moins un effectif diffère de sa valeur théorique, sans préciser le(s)quel(s). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier l'(les) effectif(s) en question. Utiliser pour cela la fonction `chisq.gof.multcomp()` du package `RVAideMemoire` : `chisq.gof.multcomp(effectifs,p=prop.theo)`.

Il peut arriver que les comparaisons deux - à - deux n'indiquent aucune différence significative, contrairement au test global. Dans ce cas, la solution la plus prudente est de considérer qu'on ne peut pas savoir quel effectif est responsable du rejet de l'hypothèse nulle dans le test global.

44. Comparaison de plusieurs effectifs – sans facteur (effectifs bruts)

Test du χ^2 de conformité (non paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple ; les individus ne doivent pas pouvoir être décomptés dans plusieurs effectifs à la fois ; au moins 80 % des effectifs théoriques doivent être non nuls et ≥ 5 (voir ci - dessous pour obtenir les effectifs théoriques).

Pour réaliser le test : `chisq.test(effectifs)` où `effectifs` est un vecteur contenant les effectifs.

Les effectifs théoriques sont donnés par la fonction `chisq.test(effectifs)$expected`.

Une *p - value* significative indique qu'au moins deux effectifs diffèrent l'un de l'autre, sans préciser lesquels. Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les effectifs en question. Utiliser pour cela la fonction `chisq.gof.multcomp()` du package `RVAideMemoire` : `chisq.gof.multcomp(effectifs)`.

Il peut arriver que les comparaisons deux - à - deux n'indiquent aucune différence significative, contrairement au test global. Dans ce cas, la solution la plus prudente est de considérer qu'on ne peut pas savoir quels effectifs sont responsables du rejet de l'hypothèse nulle dans le test global.

45. Comparaison de plusieurs effectifs – un facteur

Séries non appariées

*Analyse de déviance en Modèle Linéaire Généralisé (*GLM *; paramétrique)*

Conditions : l'échantillonnage doit être aléatoire et simple ; les classes du facteur doivent être exclusives.

Commencer par écrire le modèle censé représenter les données : `modele<-glm(formule,family=poisson)` où `formule` est la formule contenant la variable à expliquer et le facteur (voir fiche **71**). La loi de Poisson est la plus fréquemment utilisée lorsque la variable à expliquer représente des données de comptage (*i.e.* des valeurs entières et positives).

Appeler ensuite le résumé du modèle *via* `summary(modele)` et comparer la valeur de la déviance résiduelle (`Residual deviance`) avec celle des degrés de liberté résiduels (`degrees of freedom`, sur la même ligne que la déviance résiduelle). Si la déviance résiduelle est inférieure à ces degrés de liberté (ddl), l'analyse peut continuer. Dans le cas inverse on dit qu'il y a surdispersion. Cela peut vouloir dire :

- qu'un ou plusieurs facteurs importants n'ont pas été intégrés dans le modèle
- que la loi du modèle (qui représente en fait la loi de distribution des *erreurs* du modèle) n'est pas adaptée
- dans le pire des cas, les deux !

Dans le premier cas (où la déviance résiduelle est inférieure aux ddl résiduels), réaliser l'analyse de déviance *via* `anova(modele,test="Chi")`. Le tableau renvoyé donne l'effet du facteur et la *p - value* associée. Si cette *p - value* est significative, cela indique qu'au moins deux classes du facteur ont un effet différent sur la variable à expliquer (sans préciser lesquelles). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les classes en question. Ces comparaisons passent par l'utilisation de la méthode des contrastes (voir fiche **74**).

Dans le second cas (où la déviance résiduelle est supérieure aux ddl résiduels), la loi de Poisson peut être remplacée par :

- une loi quasi - Poisson. Le modèle s'écrit alors `modele2<-glm(formule,family=quasipoisson)`. L'analyse de déviance est réalisée *via* `anova(modele2,test="F")` et les comparaisons deux - à - deux éventuelles se font par la méthode des contrastes (voir fiche **74**)
- une loi binomiale négative. Il faut alors utiliser la fonction `glm.nb()` du package `MASS` et récrire le modèle : `modele2<-glm.nb(formule)`. L'analyse de déviance est réalisée *via* `anova(modele2,test="Chi")` et les comparaisons deux - à - deux éventuelles se font par la méthode des contrastes (voir fiche **74**).

Dans tous les cas, quelle que soit la loi utilisée (Poisson, quasi - Poisson ou binomiale négative), il est nécessaire de vérifier que le modèle s'ajuste bien aux données (voir fiche **73**).

Séries appariées

*Test du rapport des vraisemblances en Modèle Linéaire Généralisé Mixte (*GLMM *; paramétrique)*

Conditions : l'échantillonnage doit être aléatoire et simple au sein des classes du facteur aléatoire ; les classes du facteur fixe et du facteur aléatoire doivent être exclusives.

Commencer par écrire le modèle censé représenter les données, *via* la fonction `glmer()` du package `lme4` : `modele<-glmer(formule,family=poisson)` où `formule` est la formule contenant la variable à expliquer, le facteur fixe et le facteur aléatoire (voir fiche **71**). La loi de Poisson est la plus fréquemment utilisée lorsque la variable à expliquer représente des données de comptage (*i.e.* des valeurs entières et positives).

Vérifier que le modèle s'ajuste bien aux données (voir fiche **73**). Il n'est pas possible d'utiliser une loi quasi - Poisson ou binomiale négative avec un modèle mixte. Si le modèle s'ajuste mal aux données, demander de l'aide à un statisticien.

Si le modèle s'ajuste bien aux données, créer un second modèle, dit *nul*, appelé `modele.nul` (voir fiche **71**).

Il n'est pas possible de réaliser une analyse de déviance comme avec les modèles à facteur fixe. La démarche avec les modèles mixtes est la comparaison de modèles par le test du rapport des vraisemblances. Le principe est de comparer le modèle contenant le facteur d'intérêt avec un modèle identique en tout point sauf qu'il ne comprend pas ce facteur. La *p - value* du test correspond à celle du facteur qui a été enlevé dans le second modèle. Pour réaliser le test : `anova(modele,modele.nul)`.

Une *p - value* significative indique qu'au moins deux classes du facteur fixe ont un effet différent sur la variable à expliquer (sans préciser lesquelles). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les classes en question. Ces comparaisons passent par l'utilisation de la méthode des contrastes (voir fiche **74**).

46. Comparaison de plusieurs effectifs – deux facteurs

Séries non appariées

*Analyse de déviance en Modèle Linéaire Généralisé (*GLM ; paramétrique*)*

Conditions : l'échantillonnage doit être aléatoire et simple ; les classes des facteurs doivent être exclusives.

Commencer par écrire le modèle censé représenter les données : `modele<-glm(formule,family=poisson)` où `formule` est la formule contenant la variable à expliquer et les deux facteurs (voir fiche **71**). La loi de Poisson est la plus fréquemment utilisée lorsque la variable à expliquer représente des données de comptage (*i.e.* des valeurs entières et positives).

Appeler ensuite le résumé du modèle *via* `summary(modele)` et comparer la valeur de la déviance résiduelle (`Residual deviance`) avec celle des degrés de liberté résiduels (`degrees of freedom`, sur la même ligne que la déviance résiduelle). Si la déviance résiduelle est inférieure à ces degrés de liberté (ddl), l'analyse peut continuer. Dans le cas inverse on dit qu'il y a surdispersion. Cela peut vouloir dire :

- qu'un ou plusieurs facteurs importants n'ont pas été intégrés dans le modèle
- que la loi du modèle (qui représente en fait la loi de distribution des *erreurs* du modèle) n'est pas adaptée
- dans le pire des cas, les deux !

Dans le premier cas (où la déviance résiduelle est inférieure aux ddl résiduels), réaliser l'analyse de déviance *via* `anova(modele,test="Chi")`. Le tableau renvoyé donne l'effet de chaque facteur (et de leur interaction si elle est prise en compte) et la p - *value* associée. Si une p - *value* est significative, cela indique qu'au moins deux classes du facteur en question (ou au moins deux combinaisons de classes des deux facteurs si c'est l'effet de l'interaction qui est significatif) ont un effet différent sur la variable à expliquer (sans préciser lesquelles). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les classes (ou combinaisons de classes) en question. Ces comparaisons passent par l'utilisation de la méthode des contrastes (voir fiche **74**).

Dans le second cas (où la déviance résiduelle est supérieure aux ddl résiduels), la loi de Poisson peut être remplacée par :

- une loi quasi - Poisson. Le modèle s'écrit alors `modele2<-glm(formule,family=quasipoisson)`. L'analyse de déviance est réalisée *via* `anova(modele2,test="F")` et les comparaisons deux - à - deux éventuelles passent par la méthode des contrastes (voir fiche **74**)
- une loi binomiale négative. Il faut alors utiliser la fonction `glm.nb()` du package `MASS` et récrire le modèle : `modele2<-glm.nb(formule)`. L'analyse de déviance est réalisée *via* `anova(modele2,test="Chi")` et les

comparaisons deux - à - deux éventuelles passent par la méthode des contrastes (voir fiche **74**).

Dans tous les cas, quelle que soit la loi utilisée (Poisson, quasi - Poisson ou binomiale négative), il est nécessaire de vérifier que le modèle s'ajuste bien aux données (voir fiche **73**).

Séries appariées

*Test du rapport des vraisemblances en Modèle Linéaire Généralisé Mixte (*GLMM *; paramétrique)*

Conditions : l'échantillonnage doit être aléatoire et simple au sein des classes du facteur aléatoire ; les classes des facteurs fixes et du facteur aléatoire doivent être exclusives.

Commencer par écrire le modèle censé représenter les données, *via* la fonction `glmer()` du package `lme4` : `modele<-glmer(formule,family=poisson)` où `formule` est la formule contenant la variable à expliquer, les deux facteurs fixes et le facteur aléatoire (voir fiche **71**). La loi de Poisson est la plus fréquemment utilisée lorsque la variable à expliquer représente des données de comptage (*i.e.* des valeurs entières et positives).

Vérifier que le modèle s'ajuste bien aux données (voir fiche **73**). Il n'est pas possible d'utiliser une loi quasi - Poisson ou binomiale négative avec un modèle mixte. Si le modèle s'ajuste mal aux données, demander de l'aide à un statisticien.

Si le modèle s'ajuste bien aux données, créer ensuite deux modèles dits *réduits* : l'un sans le premier facteur (mais avec l'interaction si elle est prise en compte) et l'autre sans le second facteur (mais avec l'interaction si elle est prise en compte). Si l'interaction est prise en compte dans le modèle initial, créer également un modèle additif, avec les deux facteurs mais sans leur interaction.

Il n'est pas possible de réaliser une analyse de déviance comme avec les modèles à facteurs fixes. La démarche avec les modèles mixtes est la comparaison de modèles par le test du rapport des vraisemblances. Le principe est de comparer le modèle contenant le facteur d'intérêt (ou l'interaction d'intérêt) avec un modèle identique en tout point sauf qu'il ne comprend pas ce facteur (ou cette interaction). La *p - value* du test correspond à celle du facteur (ou de l'interaction) qui a été enlevé(e) dans le second modèle. Réaliser donc une série de tests de la forme : `anova(modele,modele.reduit)`, en comparant chaque fois le modèle complet avec l'un des deux ou trois modèles réduits.

Une *p - value* significative indique qu'au moins deux classes du facteur fixe testé (ou au moins deux combinaisons de classes des deux facteurs si c'est l'interaction qui est testée) ont un effet différent sur la variable à expliquer (sans préciser lesquelles). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les classes (ou combinaisons de classes) en question. Ces comparaisons passent par l'utilisation de la méthode des contrastes (voir fiche **74**).

47. Conformité d'une proportion avec une valeur théorique

Test binomial exact (non paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple.

Pour réaliser le test : `binom.test(n1,n,p)` où `n1` est le nombre d'individus de la catégorie d'intérêt, `n` l'effectif total et `p` la proportion théorique de la catégorie d'intérêt (ex : pour comparer le sex - ratio d'un échantillon de 20 individus contenant 8 femelles à un sex - ratio équilibré, la syntaxe est `binom.test(8,20,0.5)`).

48. Conformité de plusieurs proportions avec des valeurs théoriques

Les données doivent être présentées sous la forme de deux variables qualitatives : l'une définissant les k classes à tester, l'autre définissant les deux groupes à l'intérieur de chaque classe. Grâce à ces deux variables, les données peuvent (et doivent) être organisées en un tableau de contingence du type :

		Variable B (mesure)	
		Classe 1	Classe 2
Variable A (classes à tester)	Classe 1		
	...		
	Classe k		

où chaque case contient le nombre d'individus possédant à la fois le caractère de la variable A et celui de la variable B.

Ce tableau est obtenu de la manière suivante : `tab.cont<-table(variableA,variableB)` où `variableA` et `variableB` sont des vecteurs contenant la valeur de chaque individu pour chaque variable (dans le même ordre). Une représentation graphique du tableau de contingence peut être obtenue à l'aide de la fonction `mosaicplot(tab.cont)`.

Les proportions comparées sont celles de la première colonne du tableau de contingence (Classe 1 de la variable B).

Test du χ^2 (non paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple ; les classes des deux variables doivent être exclusives ; chaque case du tableau de contingence doit présenter un effectif théorique non nul et au moins 80 % des effectifs théoriques doivent être ≥ 5 (voir ci - dessous pour obtenir les effectifs théoriques).

Pour réaliser le test : `prop.test(tab.cont,p=prop.theo)` où `prop.theo` est un vecteur contenant les proportions théoriques de chaque classe (de 1 à k).

Pour obtenir les effectifs théoriques utiliser la fonction `chisq.exp()` du package `RVAideMemoire` : `chisq.exp(tab.cont,prop.theo)`. L'argument facultatif `graph=TRUE` permet d'obtenir une représentation graphique des effectifs théoriques.

Une p - *value* significative indique qu'au moins une proportion diffère de sa valeur théorique, sans préciser la(les)quelle(s). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier la (les) proportion(s) en question. Utiliser pour cela la fonction `prop.multcomp` du package `RVAideMemoire` : `prop.multcomp(tab.cont,prop.theo)`

Il peut arriver que les comparaisons deux - à - deux n'indiquent aucune différence significative, contrairement au test global. Dans ce cas, la solution la plus prudente est de considérer qu'on ne peut pas savoir quelle classe est responsable du rejet de l'hypothèse nulle dans le test global.

49. Comparaison de deux proportions – sans répétition

On dit qu'il n'y a pas de répétition lorsque l'on a une valeur de la proportion en question par population (*i.e.* un échantillon par population).

Les données doivent être présentées sous la forme de deux variables qualitatives : l'une définissant les deux classes à comparer, l'autre définissant les deux groupes à l'intérieur de chaque classe. Grâce à ces deux variables, les données peuvent (et doivent) être organisées en un tableau de contingence du type :

		Variable B (mesure)	
		Classe 1	Classe 2
Variable A (classes à comparer)	Classe 1		
	Classe 2		

où chaque case contient le nombre d'individus possédant à la fois le caractère de la variable A et celui de la variable B.

Ce tableau est obtenu de la manière suivante : `tab.cont<-table(variableA,variableB)` où `variableA` et `variableB` sont des vecteurs contenant la valeur de chaque individu pour chaque variable (dans le même ordre). Une représentation graphique du tableau de contingence peut être obtenue à l'aide de la fonction `mosaicplot(tab.cont)`.

Séries non appariées

Dans ces tests, les proportions comparées sont celles de la première colonne du tableau de contingence (Classe 1 de la variable B).

Test du χ^2 d'homogénéité (ou d'indépendance ; non paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple ; les classes des deux variables doivent être exclusives ; chaque case du tableau de contingence doit présenter un effectif théorique non nul et ≥ 5 (voir ci - dessous pour obtenir les effectifs théoriques).

Pour réaliser le test : `prop.test(tab.cont)`.

Les effectifs théoriques sont donnés par la fonction `chisq.test(tab.cont)$expected`.

Test exact de Fisher (non paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple ; les classes des deux variables doivent être exclusives.

Pour réaliser le test : `fisher.test(tab.cont)`.

Séries appariées

Dans le cas de séries appariées, l'individu est soit une entité mesurée deux fois, soit une paire d'entités reliées entre elles et sur qui la même mesure a été réalisée. Le tableau de contingence est donc différent :

		Variable B (2ème mesure)	
		Classe 1	Classe 2
Variable A (1ère mesure)	Classe 1		
	Classe 2		

Test binomial exact (non paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple ; les classes des deux variables doivent être exclusives ; les individus doivent pouvoir changer de classe lors de la 2ème mesure.

Pour réaliser le test : `binom.test(n1.2,ns,0.5)` où `n1.2` est le nombre d'individus qui sont de Classe 1 lors de la première mesure et de Classe 2 lors de la seconde (case en haut à droite), et `ns` est le nombre d'individus qui ont changé de classe entre les deux mesures (quel que soit le sens de ce changement : case en bas à gauche + case en haut à droite).

50. Comparaison de plusieurs proportions – sans répétition

On dit qu'il n'y a pas de répétition lorsque l'on a une valeur de la proportion en question par population (*i.e.* un échantillon par population).

Les données doivent être présentées sous la forme de deux variables qualitatives : l'une définissant les k classes à comparer, l'autre définissant les deux groupes à l'intérieur de chaque classe. Grâce à ces deux variables, les données peuvent (et doivent) être organisées en un tableau de contingence du type :

		Variable B (mesure)	
		Classe 1	Classe 2
Variable A (classes à comparer)	Classe 1		
	...		
	Classe k		

où chaque case contient le nombre d'individus possédant à la fois le caractère de la variable A et celui de la variable B.

Ce tableau est obtenu de la manière suivante : `tab.cont<-table(variableA,variableB)` où `variableA` et `variableB` sont des vecteurs contenant la valeur de chaque individu pour chaque variable (dans le même ordre). Une représentation graphique du tableau de contingence peut être obtenue à l'aide de la fonction `mosaicplot(tab.cont)`.

Les proportions comparées sont celles de la première colonne du tableau de contingence (Classe 1 de la variable B).

Test du χ^2 d'homogénéité (ou d'indépendance ; non paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple ; les classes des deux variables doivent être exclusives ; chaque case du tableau de contingence doit présenter un effectif théorique non nul et ≥ 5 (voir ci - dessous pour obtenir les effectifs théoriques).

Pour réaliser le test : `prop.test(tab.cont)`.

Les effectifs théoriques sont donnés par la fonction `chisq.test(tab.cont)$expected`.

Une p - *value* significative indique qu'au moins deux proportions diffèrent l'une de l'autre, sans préciser lesquelles. Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les proportions en question. Utiliser pour cela la fonction `pairwise.prop.test(tab.cont,p.adjust.method=methode)` (voir fiche **31** pour choisir la méthode de correction du seuil de rejet α).

51. Comparaison de plusieurs proportions – avec répétitions et un facteur

On dit qu'il y a répétitions lorsque l'on a plusieurs valeurs de la proportion en question par population (*i.e.* plusieurs échantillons par population).

Avant de réaliser l'analyse, il est nécessaire de mettre la variable à expliquer au bon format. Celle - ci ne doit en effet pas être un vecteur mais un tableau à deux colonnes du type :

```
     Groupe 1  Groupe 2
[1,]       10        16
[2,]        8        24
[3,]       14        11
...
```

Dans ce tableau, chaque ligne correspond à un échantillon et chaque colonne à un groupe à l'intérieur des échantillons (les deux colonnes définissent donc la proportion étudiée ; ex : mâles et femelles). Les données dans ce tableau sont des effectifs, *i.e.* chaque case contient le nombre d'individus appartenant à la fois à l'échantillon `[x,]` (où `x` est le numéro de la ligne) et au groupe (colonne) correspondant. Au sens statistique, un individu est représenté par une *ligne* du tableau.

Ce tableau peut être obtenu *via* `proportions<-cbind(groupe1,groupe2)` où `groupe1` et `groupe2` sont des vecteurs correspondant aux deux colonnes (la 1ère valeur correspondant à la 1ère ligne du tableau et les deux vecteurs étant dans le même ordre).

Séries non appariées

*Analyse de déviance en Modèle Linéaire Généralisé (*GLM *; paramétrique)*

Conditions : l'échantillonnage doit être aléatoire et simple ; les classes du facteur doivent être exclusives.

Commencer par écrire le modèle censé représenter les données : `modele<-glm(formule,family=binomial)` où `formule` est la formule contenant la variable à expliquer et le facteur (voir fiche **71**). La loi binomiale est la plus fréquemment utilisée lorsque la variable à expliquer représente des proportions.

Appeler ensuite le résumé du modèle *via* `summary(modele)` et comparer la valeur de la déviance résiduelle (`Residual deviance`) avec celle des degrés de liberté résiduels (`degrees of freedom`, sur la même ligne que la déviance résiduelle). Si la déviance résiduelle est inférieure à ces degrés de liberté (ddl), l'analyse peut continuer. Dans le cas inverse on dit qu'il y a surdispersion. Cela peut vouloir dire :

- qu'un ou plusieurs facteurs importants n'ont pas été intégrés dans le modèle

– que la loi du modèle (qui représente en fait la loi de distribution des *erreurs* du modèle) n'est pas adaptée
– dans le pire des cas, les deux !

Dans le premier cas (où la déviance résiduelle est inférieure aux ddl résiduels), réaliser l'analyse de déviance *via* `anova(modele,test="Chi")`. Le tableau renvoyé donne l'effet du facteur et la *p - value* associée. Si cette *p - value* est significative, cela indique qu'au moins deux classe du facteur ont un effet différent sur la variable à expliquer (sans préciser lesquelles). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les classes en question. Ces comparaisons passent par l'utilisation de la méthode des contrastes (voir fiche **74**).

Dans le second cas (où la déviance résiduelle est supérieure aux ddl résiduels), la loi binomiale peut être remplacée par une loi quasi - binomiale. Le modèle s'écrit alors `modele2<-glm(formule,family=quasibinomial)`. L'analyse de déviance est réalisée *via* `anova(modele2,test="F")` et les comparaisons deux - à - deux passent par la méthode des contrastes (voir fiche **74**).

Quelle que soit la loi utilisée (binomiale ou quasi - binomiale), il est nécessaire de vérifier que le modèle s'ajuste bien aux données (voir fiche **73**).

Séries appariées

*Test du rapport des vraisemblances en Modèle Linéaire Généralisé Mixte (*GLMM *; paramétrique)*

Conditions : l'échantillonnage doit être aléatoire et simple au sein des classes du facteur aléatoire ; les classes du facteur fixe et du facteur aléatoire doivent être exclusives.

Commencer par écrire le modèle censé représenter les données, *via* la fonction `glmer()` du package `lme4` : `modele<-glmer(formule,family=binomial)` où `formule` est la formule contenant la variable à expliquer, le facteur fixe et le facteur aléatoire (voir fiche **71**). La loi binomiale est la plus fréquemment utilisée lorsque la variable à expliquer représente des proportions.

Vérifier que le modèle s'ajuste bien aux données (voir fiche **73**). Il n'est pas possible d'utiliser une loi quasi - binomiale avec un modèle mixte. Si le modèle s'ajuste mal aux données, demander de l'aide à un statisticien.

Si le modèle s'ajuste bien aux données, créer un second modèle, dit *nul*, appelé `modele.nul` (voir fiche **71**).

Il n'est pas possible de réaliser une analyse de déviance comme avec les modèles à facteur fixe. La démarche avec les modèles mixtes est la comparaison de modèles par le test du rapport des vraisemblances. Le principe est de comparer le modèle contenant le facteur d'intérêt avec un modèle identique en tout point sauf qu'il ne comprend pas ce facteur. La *p - value* du test correspond à celle du facteur qui a été enlevé dans le second modèle. Pour

réaliser le test : `anova(modele,modele.nul)`.

Une p - *value* significative indique qu'au moins deux classes du facteur fixe ont un effet différent sur la variable à expliquer (sans préciser lesquelles). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les classes en question. Ces comparaisons passent par l'utilisation de la méthode des contrastes (voir fiche **74**).

52. Comparaison de plusieurs proportions – avec répétitions et deux facteurs

On dit qu'il y a répétitions lorsque l'on a plusieurs valeurs de la proportion en question par population (*i.e.* plusieurs échantillons par population).

Avant de réaliser l'analyse, il est nécessaire de mettre la variable à expliquer au bon format. Celle - ci ne doit en effet pas être un vecteur mais un tableau à deux colonnes du type :

```
     Groupe 1  Groupe 2
[1,]       10        16
[2,]        8        24
[3,]       14        11
...
```

Dans ce tableau, chaque ligne correspond à un échantillon et chaque colonne à un groupe à l'intérieur des échantillons (les deux colonnes définissent donc la proportion étudiée ; ex : mâles et femelles). Les données dans ce tableau sont des effectifs, *i.e.* chaque case contient le nombre d'individus appartenant à la fois à l'échantillon `[x,]` (où `x` est le numéro de la ligne) et au groupe (colonne) correspondant. Au sens statistique, un individu est représenté par une *ligne* du tableau.

Ce tableau peut être obtenu *via* `proportions<-cbind(groupe1,groupe2)` où `groupe1` et `groupe2` sont des vecteurs correspondant aux deux colonnes (la 1ère valeur correspondant à la 1ère ligne du tableau et les deux vecteurs étant dans le même ordre).

Séries non appariées

*Analyse de déviance en Modèle Linéaire Généralisé (*GLM ; paramétrique*)*

Conditions : l'échantillonnage doit être aléatoire et simple ; les classes des facteurs doivent être exclusives.

Commencer par écrire le modèle censé représenter les données : `modele<-glm(formule,family=binomial)` où `formule` est la formule contenant la variable à expliquer et les deux facteurs (voir fiche **71**). La loi binomiale est la plus fréquemment utilisée lorsque la variable à expliquer représente des proportions.

Appeler ensuite le résumé du modèle *via* `summary(modele)` et comparer la valeur de la déviance résiduelle (`Residual deviance`) avec celle des degrés de liberté résiduels (`degrees of freedom`, sur la même ligne que la déviance résiduelle). Si la déviance résiduelle est inférieure à ces degrés de liberté (ddl), l'analyse peut continuer. Dans le cas inverse on dit qu'il y a surdispersion. Cela peut vouloir dire :

- qu'un ou plusieurs facteurs importants n'ont pas été intégrés dans le modèle

– que la loi du modèle (qui représente en fait la loi de distribution des *erreurs* du modèle) n'est pas adaptée
– dans le pire des cas, les deux !

Dans le premier cas (où la déviance résiduelle est inférieure aux ddl résiduels), l'analyse de déviance est réalisée *via* `anova(modele,test="Chi")`. Le tableau renvoyé donne l'effet de chaque facteur (et de leur interaction si elle est prise en compte) et la p - *value* associée. Si une p - *value* est significative, cela indique qu'au moins deux classes du facteur en question (ou au moins deux combinaisons de classes des deux facteurs si c'est l'effet de l'interaction qui est significatif) ont un effet différent sur la variable à expliquer (sans préciser lesquelles). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les classes (ou combinaisons de classes) en question. Ces comparaisons passent par l'utilisation de la méthode des contrastes (voir fiche **74**).

Dans le second cas (où la déviance résiduelle est supérieure aux ddl résiduels), la loi binomiale peut être remplacée par une loi quasi - binomiale. Le modèle s'écrit alors `modele2<-glm(formule,family=quasibinomial)`. L'analyse de déviance est réalisée *via* `anova(modele2,test="F")` et les comparaisons deux - à - deux passent par la méthode des contrastes (voir fiche **74**).

Quelle que soit la loi utilisée (binomiale ou quasi - binomiale), il est nécessaire de vérifier que le modèle s'ajuste bien aux données (voir fiche **73**).

Séries appariées

*Test du rapport des vraisemblances en Modèle Linéaire Généralisé Mixte (*GLMM *; paramétrique)*

Conditions : l'échantillonnage doit être aléatoire et simple au sein des classes du facteur aléatoire ; les classes des facteurs fixes et du facteur aléatoire doivent être exclusives.

Commencer par écrire le modèle censé représenter les données, *via* la fonction `glmer()` du package `lme4` : `modele<-glmer(formule,family=binomial)` où `formule` est la formule contenant la variable à expliquer, les deux facteurs fixes et le facteur aléatoire (voir fiche **71**). La loi binomiale est la plus fréquemment utilisée lorsque la variable à expliquer représente des proportions.

Vérifier que le modèle s'ajuste bien aux données (voir fiche **73**). Il n'est pas possible d'utiliser une loi quasi - binomiale avec un modèle mixte. Si le modèle s'ajuste mal aux données, demander de l'aide à un statisticien.

Si le modèle s'ajuste bien aux données, créer ensuite deux modèles dits *réduits* : l'un sans le premier facteur (mais avec l'interaction si elle est prise en compte) et l'autre sans le second facteur (mais avec l'interaction si elle est prise en compte). Si l'interaction est prise en compte dans le modèle initial, créer également un modèle additif, avec les deux facteurs mais sans leur interaction.

Il n'est pas possible de réaliser une analyse de déviance comme avec les modèles à facteurs fixes. La démarche avec les modèles mixtes est la comparaison de modèles par le test du rapport des vraisemblances. Le principe est de comparer le modèle contenant le facteur d'intérêt (ou l'interaction d'intérêt) avec un modèle identique en tout point sauf qu'il ne comprend pas ce facteur (ou cette interaction). La *p - value* du test correspond à celle du facteur (ou de l'interaction) qui a été enlevé(e) dans le second modèle. Réaliser donc une série de tests de la forme : `anova(modele,modele.reduit)`, en comparant chaque fois le modèle complet avec l'un des deux ou trois modèles réduits.

Une *p - value* significative indique qu'au moins deux classes du facteur fixe testé (ou au moins deux combinaisons de classes des deux facteurs si c'est l'interaction qui est testée) ont un effet différent sur la variable à expliquer (sans préciser lesquelles). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les classes (ou combinaisons de classes) en question. Ces comparaisons passent par l'utilisation de la méthode des contrastes (voir fiche **74**).

53. Conformité d'une moyenne avec une valeur théorique

Test t de Student (paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple ; la distribution des données doit être normale.

Pour réaliser le test : `t.test(serie,mu=m.theo)` où `serie` est un vecteur contenant la série de données et `m.theo` la moyenne théorique à comparer.

Ce test étant assez robuste, il peut être utilisé lorsque la distribution des données ne suit pas une loi normale, à condition qu'elle ne s'en éloigne pas trop et que l'échantillon soit de grande taille (> 30 individus). Prendre garde aux individus extrêmes qui ont une grande influence sur la moyenne.

Test des rangs signés de Wilcoxon (non paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple ; l'échantillon doit contenir au moins 8 individus ; la distribution des données doit être symétrique (uni- ou polymodale mais symétrique).

Pour réaliser le test : `wilcox.test(serie,mu=m.theo)`.

Ce test compare en fait la *médiane* de l'échantillon avec la valeur théorique. Mais si la distribution est symétrique, médiane et moyenne sont très proches. Examiner cette condition à l'aide d'un graphique de type `boxplot()` (voir fiche **11**). La médiane n'est pas sensible aux individus extrêmes.

Test des signes de Wilcoxon (non paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple.

Pour réaliser le test, utiliser la fonction `wilcox.sign.test()` du package `RVAideMemoire` : `wilcox.sign.test(serie,mu=m.theo)`.

Ce test est à utiliser lorsque les conditions du test des rangs signés de Wilcoxon ne sont pas réunies. Il compare en fait la *médiane* de l'échantillon avec la valeur théorique.

54. Comparaison de deux moyennes

Séries non appariées

Test t *de Student (paramétrique)*

Conditions : l'échantillonnage doit être aléatoire et simple ; les individus ne doivent pas pouvoir être décomptés dans les deux échantillons à la fois ; la distribution des données dans chaque échantillon doit être normale ; la variance des deux échantillons doit être égale.

Pour réaliser le test : `t.test(serie1,serie2,var.equal=TRUE)` où `serie1` et `serie2` sont les deux vecteurs contenant les séries de données à comparer. Si les deux séries correspondent aux deux classes d'un facteur, la syntaxe peut être `t.test(variable~facteur)`, où `variable` est un vecteur contenant les valeurs de la variable mesurée et `facteur` un vecteur contenant la classe de chaque individu (dans le même ordre que `variable`).

Ce test étant assez robuste, il peut être utilisé lorsque la distribution des données ne suit pas une loi normale, à condition qu'elle ne s'en éloigne pas trop et que l'échantillon soit de grande taille (> 30 individus). Il est cependant très sensible à la condition d'homoscédasticité. De plus prendre garde aux individus extrêmes qui ont une grande influence sur la moyenne.

Si les variances sont inégales, appliquer la correction de Welch en précisant `var.equal=FALSE` (ce qui est le cas par défaut si `var.equal` n'est pas précisé).

Test de Mann - Whitney - Wilcoxon (non paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple ; les individus ne doivent pas pouvoir être décomptés dans les deux échantillons à la fois ; les deux échantillons doivent contenir au moins 8 individus ; la distribution des données dans les deux échantillons doit avoir la même forme (peu importe celle - ci).

Pour réaliser le test : `wilcox.test(serie1,serie2)` ou `wilcox.test(variable~facteur)`.

Ce test compare en fait la *médiane* des deux échantillons, et est donc très sensible à la similitude de leur distribution. Examiner cette condition à l'aide d'un graphique de type `hist()` (voir fiche **10**). Il faut de fait toujours rester prudent quant à son interprétation car cette condition est difficile à remplir.

Test exact de Fisher (non paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple ; les individus ne doivent pas pouvoir être décomptés dans les deux échantillons à la fois.

Pour réaliser le test, utiliser la fonction `fisher.medtest` du package `RVAideMemoire` : `fisher.medtest(serie1,serie2)`.

Ce test est à utiliser lorsque les conditions du test de Mann - Whitney - Wilcoxon ne sont pas réunies. Il compare en fait la *médiane* des deux échantillons.

Séries appariées

Test t de Student pour séries appariées (paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple ; la distribution des données dans chaque échantillon doit être normale.

Pour réaliser le test : `t.test(serie1,serie2,paired=TRUE)`.

Ce test étant assez robuste, il peut être utilisé lorsque la distribution des données ne suit pas une loi normale, à condition qu'elle ne s'en éloigne pas trop et que l'échantillon soit de grande taille (> 30 individus). Prendre garde aux individus extrêmes qui ont une grande influence sur la moyenne.

Test des rangs signés de Wilcoxon (non paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple ; les deux échantillons doivent contenir au moins 8 individus ; la distribution des différences entre les valeurs appariées doit être symétriques (voir ci - dessous pour vérifier cette condition).

Pour réaliser le test : `wilcox.test(serie1,serie2,paired=TRUE)`.

Ce test compare en fait la *médiane* des deux échantillons. Mais si la distribution des différences entre les valeurs appariées est symétrique, médiane et moyenne sont très proches. Examiner cette condition à l'aide d'un graphique de type `boxplot(serie1-serie2)` (voir fiche **11**). La médiane n'est pas sensible aux individus extrêmes.

Test des signes de Wilcoxon (non paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple.

Pour réaliser le test, utiliser la fonction `wilcox.sign.test()` du package `RVAideMemoire` : `wilcox.sign.test(serie1,serie2)`.

Ce test est à utiliser lorsque les conditions du test des rangs signés de Wilcoxon ne sont pas réunies. Il compare en fait la *médiane* des deux échantillons.

55. Comparaison de plusieurs moyennes – un facteur

Séries non appariées

Analyse de variance (ANOVA) en Modèle Linéaire (LM) (paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple ; les classes du facteur doivent être exclusives ; la distribution des données doit être normale dans chaque classe du facteur ; la variance des données doit être égale entre toutes les classes du facteur ; la distribution des résidus du modèle doit être normale (voir fiche **73** pour tester cette hypothèse).

Commencer par écrire le modèle censé représenter les données : `modele<-lm(formule)`, où `formule` est la formule contenant la variable à expliquer et le facteur (voir fiche **71**).

L'ANOVA est réalisée grâce à la fonction `anova(modele)`. Le tableau renvoyé donne l'effet du facteur et la p - *value* associée. Si cette p - *value* est significative, cela indique qu'au moins deux classes du facteur ont un effet différent sur la variable à expliquer (sans préciser lesquelles). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les classes en question. Utiliser pour cela la fonction `pairwise.t.test(variable,facteur,p.adjust.method=methode)` (voir fiche **31** pour choisir la méthode de correction du seuil de rejet α).

Ce test étant assez robuste, il peut être utilisé lorsque la distribution des données ne suit pas une loi normale, à condition qu'elle ne s'en éloigne pas trop et que l'échantillon soit de grande taille (> 30 individus). Il est cependant très sensible à la condition d'homoscédasticité. De plus prendre garde aux individus extrêmes qui ont une grande influence sur la moyenne.

Test de Kruskal - Wallis (non paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple ; les classes du facteur doivent être exclusives ; la distribution des données doit avoir la même forme dans toutes les classes du facteur (peu importe celle - ci).

Pour réaliser le test : `kruskal.test(formule)`.

Si la p - *value* est significative, les comparaisons deux - à - deux sont réalisées grâce à la fonction `pairwise.wilcox.test(variable,facteur,p.adjust.method=methode)` (voir fiche **31** pour choisir la méthode de correction du seuil de rejet α).

Ce test compare en fait la *médiane* des différents échantillons, et est donc très sensible à la similitude de leur distribution. Examiner cette condition à l'aide d'un graphique de type `hist()` (voir fiche **10**). Il faut de fait toujours rester prudent quant à son interprétation car cette condition est difficile à remplir.

Séries appariées

*Test du rapport des vraisemblances en Modèle Linéaire Mixte (*LMM *; paramétrique)*

Conditions : l'échantillonnage doit être aléatoire et simple au sein des classes du facteur aléatoire ; les classes du facteur fixe et du facteur aléatoire doivent être exclusives ; la distribution des données doit être normale dans chaque classe du facteur fixe ; la variance des données doit être égale entre toutes les classes du facteur fixe ; la distribution des résidus du modèle doit être normale (voir fiche **73** pour tester cette hypothèse).

Commencer par écrire le modèle censé représenter les données, *via* la fonction `lmer()` du package `lme4` : `modele<-lmer(formule,REML=FALSE)` où `formule` est la formule contenant la variable à expliquer, le facteur fixe et le facteur aléatoire (voir fiche **71**).

Créer un second modèle, dit *nul*, appelé `modele.nul` (voir fiche **71**).

Il n'est pas possible de réaliser une analyse de variance comme avec les modèles à facteur fixe. La démarche avec les modèles mixtes est la comparaison de modèles par le test du rapport des vraisemblances. Le principe est de comparer le modèle contenant le facteur d'intérêt avec un modèle identique en tout point sauf qu'il ne comprend pas ce facteur. La *p - value* du test correspond à celle du facteur qui a été enlevé dans le second modèle. Pour réaliser le test : `anova(modele,modele.nul)`.

Une *p - value* significative indique qu'au moins deux classes du facteur fixe ont un effet différent sur la variable à expliquer (sans préciser lesquelles). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les classes en question. Pour cela, commencer par récrire le modèle en remplaçant `REML=FALSE` par `REML=TRUE`. Utiliser ensuite la méthode des contrastes pour les comparaisons (voir fiche **74**).

Test de Friedman (non paramétrique)

Conditions : le plan d'expérience doit être en blocs aléatoire complets, avec une seule observation par modalité du facteur au sein de chaque bloc (*i.e.* de chaque classe du facteur aléatoire) ; l'échantillonnage doit être aléatoire et simple au sein des classes du facteur aléatoire ; les classes du facteur fixe et du facteur aléatoire doivent être exclusives.

Pour réaliser le test : `friedman.test(a.expliquer~fact.fixe|fact.aleatoire)`.

Si la *p - value* du test est significative, cela indique qu'au moins deux classes du facteur fixe ont un effet différent sur la variable à expliquer (sans préciser lesquelles). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les classes en question. Utiliser pour cela la fonction `wilcox.paired.multcomp()` du package `RVAideMemoire` : `wilcox.paired.multcomp(a.expliquer,fact.fixe,fact.aleatoire)`.

56. Comparaison de plusieurs moyennes – deux facteurs

Séries non appariées

*Analyse de variance (*ANOVA*) en Modèle Linéaire (*LM*) (paramétrique)*

Conditions : l'échantillonnage doit être aléatoire et simple ; les classes des facteurs doivent être exclusives ; la distribution des données doit être normale dans chaque classe des deux facteurs ; la variance des données doit être égale entre toutes les classes d'un même facteur ; la distribution des résidus du modèle doit être normale (voir fiche **73** pour tester cette hypothèse).

Commencer par écrire le modèle censé représenter les données : `modele<-lm(formule)` où `formule` est la formule contenant la variable à expliquer et les deux facteurs (voir fiche **71**).

L'ANOVA est réalisée grâce à la fonction `anova(modele)`. Le tableau renvoyé donne l'effet de chaque facteur (et de leur interaction si elle est prise en compte) et la *p - value* associée. Si une *p - value* est significative, cela indique qu'au moins deux classes du facteur en question (ou au moins deux combinaisons de classes des deux facteurs si c'est l'effet de l'interaction qui est significatif) ont un effet différent des autres sur la variable à expliquer (sans préciser lesquelles). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les classes (ou combinaisons de classes) en question. Ces comparaisons passent par l'utilisation de la méthode des contrastes (voir fiche **74**).

Ce test étant assez robuste, il peut être utilisé lorsque la distribution des données ne suit pas une loi normale, à condition qu'elle ne s'en éloigne pas trop et que l'échantillon soit de grande taille (> 30 individus). Il est cependant très sensible à la condition d'homoscédasticité. De plus prendre garde aux individus extrêmes qui ont une grande influence sur la moyenne.

Séries appariées

*Test du rapport des vraisemblances en Modèle Linéaire Mixte (*LMM *; paramétrique)*

Conditions : l'échantillonnage doit être aléatoire et simple au sein des classes du facteur aléatoire ; les classes des facteurs fixes et du facteur aléatoire doivent être exclusives ; la distribution des données doit être normale dans chaque classe des facteurs fixes ; la variance des données doit être égale entre toutes les classes d'un même facteur fixe ; la distribution des résidus du modèle doit être normale (voir fiche **73** pour tester cette hypothèse).

Commencer par écrire le modèle censé représenter les données, *via* la fonction `lmer()` du package `lme4` : `modele<-lmer(formule,REML=FALSE)` où `formule` est la formule contenant la variable à expliquer, les deux facteurs fixes et le facteur aléatoire (voir fiche **71**).

Créer deux modèles dits *réduits* : l'un sans le premier facteur (mais avec l'interaction si elle est prise en compte) et l'autre sans le second facteur (mais avec l'interaction si elle est prise en compte). Si l'interaction est prise en

compte dans le modèle initial, créer également un modèle additif, avec les deux facteurs mais sans leur interaction.

Il n'est pas possible de réaliser une analyse de déviance comme avec les modèles à facteurs fixes. La démarche avec les modèles mixtes est la comparaison de modèles par le test du rapport des vraisemblances. Le principe est de comparer le modèle contenant le facteur d'intérêt (ou l'interaction d'intérêt) avec un modèle identique en tout point sauf qu'il ne comprend pas ce facteur (ou cette interaction). La *p - value* du test correspond à celle du facteur (ou de l'interaction) qui a été enlevé(e) dans le second modèle. Réaliser donc une série de tests de la forme : `anova(modele,modele.reduit)`, en comparant chaque fois le modèle complet avec l'un des deux ou trois modèles réduits.

Une *p - value* significative indique qu'au moins deux classes du facteur fixe testé (ou au moins deux combinaisons de classes des deux facteurs si c'est l'interaction qui est testée) ont un effet différent sur la variable à expliquer (sans préciser lesquelles). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les classes (ou combinaisons de classes) en question. Pour cela, commencer par récrire le modèle en remplaçant `REML=FALSE` par `REML=TRUE`. Utiliser ensuite la méthode des contrastes pour les comparaisons (voir fiche **74**).

57. Comparaison de plusieurs temps de survie

Avant d'analyser des temps de survie, il est indispensable d'avoir bien compris les notions suivantes :

- *censure* : un individu est dit censuré lorsque sa mort est survenue avant le début de l'étude (censure *à gauche*) ou quelle n'a pas été observée avant la fin de l'étude (parce que l'étude s'est arrêtée ou parce que l'individu en est sorti ; censure *à droite*). Cette fiche ne traite pas des censures à gauche
- *risque instantané* : ce risque est celui de mourir à l'instant t, sachant que la mort n'est pas survenue avant.

Il y a globalement 3 situations différentes lorsque l'on analyse des temps de survie :

- le risque instantané est constant quel que soit l'âge des individus et aucune donnée n'est censurée
- le risque instantané est constant quel que soit l'âge des individus et il existe des données censurées
- le risque instantané n'est pas constant, *i.e.* il augmente ou diminue avec l'âge des individus.

Pour savoir si le risque instantané est constant, tracer la courbe de survie des individus grâce à la fonction `plotsurvivors()` du package `RVAideMemoire` : `plotsurvivors(mort,censure)` où `mort` est un vecteur contenant le temps de mort de chaque individu et `censure` un vecteur indiquant si l'individu est censuré ou non (0 si censuré ou 1 si non censuré, *i.e.* 0 si la mort n'a pas été observée ou 1 si elle l'a été), dans le même ordre que `mort`. Le risque instantané est constant si la courbe de survie est une droite.

Risque instantané constant et absence de données censurées

*Analyse de déviance en Modèle Linéaire Généralisé (*GLM *; paramétrique)*

Conditions : l'échantillonnage doit être aléatoire et simple ; si certaines variables explicatives sont des facteurs, leurs classes doivent être exclusives.

Commencer par écrire le modèle censé représenter les données : `modele<-glm(formule,family=Gamma)` où `formule` est la formule contenant la variable à expliquer et les variables explicatives (voir fiche **71**). La loi Gamma sert à modéliser la constance du risque instantané.

Réaliser ensuite l'analyse de déviance *via* `anova(modele,test="Chi")`. Le tableau renvoyé donne l'effet de chaque variable explicative (et de leurs interactions si elles sont prises en compte) et la p - *value* associée. Si une p - *value* est significative, cela indique :

- si la variable explicative en question est quantitative, qu'elle a un effet sur le temps de survie
- si la variable explicative en question est un facteur, qu'au moins deux classes de ce facteur ont un effet différent sur le temps de survie (sans pré-

ciser lesquelles). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les classes en question. Ces comparaisons passent par l'utilisation de la méthode des contrastes (voir fiche **74**)
- si la *p - value* correspond à l'interaction entre deux variables, qu'au moins deux combinaisons des deux variables ont un effet différent sur le temps de survie (sans préciser lesquelles). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les combinaisons en question. Ces comparaisons passent par l'utilisation de la méthode des contrastes (voir fiche **74**).

Comme pour tout modèle, il est nécessaire de vérifier qu'il s'ajuste bien aux données (voir fiche **73**).

Risque instantané constant et présence de données censurées

Analyse de déviance en régression de survie (paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple ; si certaines variables explicatives sont des facteurs, leurs classes doivent être exclusives ; les censures doivent être indépendantes des conditions d'expérience.

Commencer par créer la variable à expliquer, qui doit être un *objet de survie*. Utiliser pour cela la fonction `Surv()` du package `survival` :
- si tous les individus sont observés pendant toute la durée de l'étude : `a.expliquer<-Surv(mort,censure)`
- si tous les individus ne sont pas observés pendant toute la durée de l'étude : `a.expliquer<-Surv(start,stop,censure)` où `start` est un vecteur contenant le moment où chaque individu rentre dans l'étude et `stop` un vecteur contenant le moment où il en sort (dans le même ordre que `start`).

Créer ensuite le modèle censé représenter les données, grâce à la fonction `survreg()` du package `survival` : `modele<-survreg(formule,dist="exponential")` où `formule` est la formule contenant la variable à expliquer et les variables explicatives (voir fiche **71**). La loi exponentielle sert à modéliser la constance du risque instantané.

Réaliser ensuite l'analyse de déviance *via* `anova(modele)`. Le tableau renvoyé donne l'effet de chaque variable explicative (et de leurs interactions si elles sont prises en compte) et la *p - value* associée. Si une *p - value* est significative, cela indique :
- si la variable explicative en question est quantitative, qu'elle a un effet sur le temps de survie
- si la variable explicative en question est un facteur, qu'au moins deux classes de ce facteur ont un effet différent sur le temps de survie (sans préciser lesquelles). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les classes en question. Utiliser pour cela la fonction `surv.multcomp()` du package `RVAideMemoire` :

surv.multcomp(a.expliquer,facteur,matrice,type="survreg",distribution="exponential"), où facteur est le facteur dont on veut comparer les classes et matrice est la matrice des comparaisons, construite de la même façon que pour la méthode des contrastes (voir fiche **74**)

– si la p - *value* correspond à l'interaction entre deux variables, qu'au moins deux combinaisons des deux variables ont un effet différent sur le temps de survie (sans préciser lesquelles). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les combinaisons en question. Utiliser pour cela la fonction surv.multcomp() du package RVAideMemoire : surv.multcomp(a.expliquer,interaction,matrice,type="survreg",distribution="exponential") (voir fiche **74** pour créer le nouveau facteur interaction et la matrice des comparaisons).

Risque instantané dépendant de l'âge des individus

Analyse de déviance en régression de survie (paramétrique)

La procédure est identique à celle mise en œuvre lorsque le risque est constant, excepté le fait que le modèle de régression de survie doit être défini avec dist="weibull". La loi de Weibull est en effet la plus utilisée lorsque le risque instantané n'est pas constant.

On peut appeler le résumé du modèle *via* summary(modele). La valeur du paramètre Scale indique l'évolution de ce risque :

– < 1 : le risque diminue avec l'âge des individus

– > 1 : le risque augmente avec l'âge des individus.

Pour les comparaisons deux - à - deux, il faut préciser distribution="weibull" à la fonction surv.multcomp().

Analyse de déviance en modèle de Cox (semi - paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple ; si certaines variables explicatives sont des facteurs, leurs classes doivent être exclusives ; les censures doivent être indépendantes des conditions d'expérience ; la relation entre chaque variable explicative quantitative (ou covariable) et le risque instantané doit être log - linéaire ; le rapport des risques instantanés de deux individus doit être indépendant du temps (voir ci - dessous pour tester ces deux hypothèses, dont la seconde est dite des *risques proportionnels*).

Créer le modèle censé représenter les données, grâce à la fonction coxph() du package survival : modele<-coxph(formule) où formule est la formule contenant la variable à expliquer (*i.e.* l'objet de survie créé *via* la fonction Surv()) et les variables explicatives (voir fiche **71**).

Pour tester l'hypothèse de log - linéarité entre les covariables et le risque instantané, utiliser la fonction cox.resid() du package RVAideMemoire : cox.resid(modele,list(variable1=variable1,variable2=variable2...)) où le 2nd argument est une liste contenant chaque covariable. La fonction trace un graphe par covariable. Sur ces graphes, la ligne rouge représente la tendance du nuage de point. On accepte l'hypothèse de log - linéarité pour une

covariable si la ligne rouge correspondante est à peu près horizontale. Dans le cas contraire il vaut mieux alors la transformer en facteur en la découpant en classes, puis la réintégrer au modèle.

Pour tester l'hypothèse des risques proportionnels, utiliser la fonction `cox.zph()` du package `survival` : `cox.zph(modele)`. La fonction teste l'hypothèse pour chaque variable explicative, ainsi que pour le modèle global. Si une *p - value* est significative, cela indique que l'hypothèse n'est pas respectée pour la variable explicative en question, qui est dite *dépendante du temps*. Il vaut mieux alors l'intégrer au modèle en temps que *strate* et non variable explicative (pour les variables explicatives quantitatives, cela passe par un découpage en classes et une transformation en facteur). L'effet de la variable ne sera plus calculé, mais pris en compte à travers la définition de risques instantanés *de base* différents selon les strates. Pour intégrer une strate dans la formule du modèle, ajouter `+strata(variable)` après les variables explicatives (et retirer la variable désormais stratifiée des variables explicatives).

Une fois toutes ces vérifications faites, réaliser l'analyse de déviance *via* `anova(modele)`. Le tableau renvoyé donne l'effet de chaque variable explicative (et de leurs interactions si elles sont prises en compte) et la *p - value* associée. Si une *p - value* est significative, cela indique :

- si la variable explicative en question est quantitative, qu'elle a un effet sur le temps de survie
- si la variable explicative en question est un facteur, qu'au moins deux classes de ce facteur ont un effet différent sur le temps de survie (sans préciser lesquelles). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les classes en question. Utiliser pour cela la fonction `surv.multcomp()` du package `RVAideMemoire` : `surv.multcomp(a.expliquer,facteur,matrice,type="coxph")`, où `facteur` est le facteur dont on veut comparer les classes et `matrice` est la matrice des comparaisons, construite de la même façon que pour la méthode des contrastes (voir fiche **74**). Ajouter l'argument `strata=variable` si une variable explicative est stratifiée dans `modele`
- si la *p - value* correspond à l'interaction entre deux variables, qu'au moins deux combinaisons des deux variables ont un effet différent sur le temps de survie (sans préciser lesquelles). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les combinaisons en question. Utiliser pour cela la fonction `surv.multcomp()` du package `RVAideMemoire` : `surv.multcomp(a.expliquer,interaction,matrice,type="coxph")` (voir fiche **74** pour créer le nouveau facteur `interaction` et la matrice des comparaisons). Ajouter l'argument `strata=variable` si une variable explicative est stratifiée dans `modele`.

58. ℹ Tracer des courbes de survie

La première étape est de créer les données des courbes de survie. Utiliser pour cela la fonction `survfit()` du package `survival`, mais de façon différente selon l'objectif désiré :

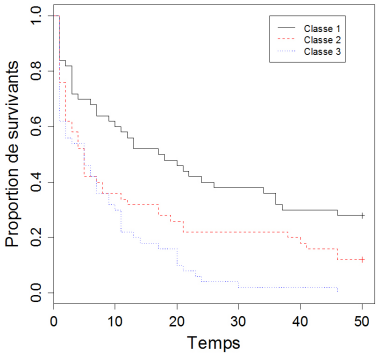
- pour représenter une courbe par niveau d'un facteur : `courbes<-survfit(formule)` où `formule` est la formule contenant la variable à expliquer (*i.e.* l'objet de survie, voir fiche **57**) et le facteur (voir fiche **71**). Il est possible d'ajouter une strate dans la formule (voir fiche **57**) pour tracer des courbes par niveau du facteur et par strate. Pour tracer une courbe par combinaison des classes de deux facteurs (si c'est l'effet de l'interaction entre ces deux facteurs qui doit être représenté), commencer par créer un nouveau facteur : `interaction<-factor(paste(facteur1,facteur2,sep=":"))` puis utiliser ce facteur comme variable explicative dans `formule`
- pour représenter une courbe moyenne après ajustement d'un modèle de Cox : `courbes<-survfit(modele)` (voir fiche **57** pour la création d'un modèle de Cox).

Tracer ensuite ces courbes de survie, simplement *via* `plot(courbes)`. Les individus censurés sont représentés par une croix (+).

Comme pour tout graphique, R offre de nombreuses possibilités de personnalisation grâce aux arguments facultatifs de `plot()` :

- pour tracer les intervalles de confiance (à 95 %) des courbes de survie, utiliser l'argument `conf.int` (`TRUE` ou `FALSE`)
- pour que le tracé des lignes (pleine, pointillée...) diffère si plusieurs sont représentées, utiliser l'argument `lty` (voir l'aide de la fonction `par()`)
- pour que la couleur des lignes diffère si plusieurs sont représentées, utiliser l'argument `col` (voir l'aide de la fonction `par()`).

Enfin, pour ajouter une légende, utiliser la fonction `legend()` (voir l'aide de cette fonction, qui possède de nombreux arguments facultatifs).



59. Indépendance de deux variables qualitatives

Les données doivent être organisées en un tableau de contingence du type :

		Variable B		
		Classe 1	...	Classe c
Variable A	Classe 1			
	...			
	Classe k			

où chaque case contient le nombre d'individus possédant à la fois le caractère de la variable A et celui de la variable B.

Ce tableau est obtenu de la manière suivante : `tab.cont<-table(variableA,variableB)` où `variableA` et `variableB` sont des vecteurs contenant la valeur de chaque individu pour chaque variable (dans le même ordre). Une représentation graphique du tableau de contingence peut être obtenue à l'aide de la fonction `mosaicplot(tab.cont)`.

Test du χ^2 d'homogénéité (ou d'indépendance ; non paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple ; les classes des deux variables doivent être exclusives ; chaque case du tableau de contingence doit présenter un effectif théorique non nul et ≥ 5 (voir ci - dessous pour obtenir les effectifs théorique).

Pour réaliser le test : `chisq.test(tab.cont)`.

Les effectifs théoriques sont donnés par la fonction `chisq.test(tab.cont)$expected`)

Test exact de Fisher (non paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple ; les classes des deux variables doivent être exclusives.

Pour réaliser le test : `fisher.test(tab.cont)`.

Si le message d'avertissement `out of workspace` apparaît, augmenter la valeur de l'argument `workspace` (par défaut `workspace=200000`). Si un autre message d'avertissement apparaît, cela peut être à cause d'un tableau trop complexe à analyser.

Quel que soit le test utilisé, une p - *value* significative indique que les deux variables ne sont pas indépendantes, sans préciser les classes qui sont à l'origine de cette liaison. Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les classes en question. Utiliser pour cela la fonction `fisher.multcomp()` du package `RVAideMemoire` : `fisher.multcomp(tab.cont)`. La fonction réalise un test exact de Fisher sur chaque tableau de contingence 2 x 2 possible à partir de `tab.cont`, et renvoie tous les résultats dont la p - *value* est inférieure à 0,1. Il est nécessaire d'interpréter ces résultats pour repérer les classes qui apparaissent systématiquement

dans les tests qui donnent un p - *value* significative. Ce sont ces classes qui sont liées.

Il peut arriver que les comparaisons deux - à - deux n'indiquent aucune liaison significative, contrairement au test global. Dans ce cas, la solution la plus prudente est de considérer qu'on ne peut pas savoir quelles classes sont responsables du rejet de l'hypothèse nulle dans le test global.

60. Corrélation entre deux variables

Coefficient de corrélation linéaire de Pearson (paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple ; les deux variables doivent être quantitatives ; chaque individu doit posséder une valeur pour les deux variables ; chaque variable doit avoir une distribution normale ; pour chaque valeur d'une variable, la distribution des valeurs possibles de la seconde doit suivre une loi normale (voir ci - dessous pour tester cette hypothèse, dite de *binormalité*) ; la relation entre les deux variables doit être linéaire.

Avant de réaliser le test, il est indispensable de vérifier que la relation entre les deux variables est linéaire. Examiner cette condition à l'aide d'un graphique de type `plot()` (voir fiche **13**).

Pour réaliser le test : `cor.test(variable1,variable2,method="pearson")` où `variable1` et `variable2` sont des vecteurs contenant la valeur de chaque individu pour chaque variable (dans le même ordre). **R** renvoie la valeur du coefficient de corrélation et son intervalle de confiance à 95 %.

Pour tester l'hypothèse de binormalité, tester la normalité des résidus de la régression de la 1ère variable sur la 2ème et inversement, en utilisant les fonctions `qqnorm()` et `shapiro.test()` (voir fiches **38** et **73**) sur `lm(variable1~variable2)$resid` et `lm(variable2~variable1)$resid`.

Ce test étant assez robuste, la condition de binormalité est peu contraignante lorsque l'échantillon est de grande taille (> 20 individus). Il peut dans ces conditions être aussi utilisé pour des variables qualitatives ordinales codées sous forme numérique. Prendre garde aux individus extrêmes qui ont une grande influence sur le coefficient de corrélation.

Coefficient de corrélation de Spearman (non paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple ; les deux variables doivent être quantitatives ou qualitatives ordinales (codées numériquement) ; chaque individu doit posséder une valeur pour les deux variables ; la relation entre les deux variables doit être monotone (ascendante ou descendante, mais pas forcément linéaire).

Avant de réaliser le test, il est indispensable de vérifier que la relation entre les deux variables est monotone. Examiner cette condition à l'aide d'un graphique de type `plot()` (voir fiche **13**).

Pour réaliser le test : `cor.test(variable1,variable2,method="spearman")`. **R** ne calcule pas l'intervalle de confiance du coefficient de corrélation, pour l'obtenir utiliser la fonction `spearman.ci()` du package `RVAideMemoire`, qui le calcule par bootstrap : `spearman.ci(variable1,variable2)`. La fonction renvoie par défaut l'intervalle de confiance à 95 %.

Coefficient d'association de Cramer (non paramétrique)

Conditions : l'échantillonnage doit être aléatoire et simple ; les deux variables doivent être qualitatives (ordinales ou nominales) ; chaque individu doit posséder une valeur pour les deux variables ; chaque classe des deux variables doit contenir au moins 5 % du nombre total d'individus.

Pour réaliser le test, utiliser la fonction `cramer.cor()` du package `RVAideMemoire` : `cramer.cor(variable1,variable2)`. La fonction renvoie l'intervalle de confiance à 95 % du coefficient d'association, calculé par bootstrap.

Contrairement aux coefficients de Pearson et Spearman, le coefficient d'association de Cramer n'est pas un coefficient de corrélation (encore moins de corrélation linéaire) car les variables auxquelles il s'intéresse ne sont pas quantitatives. Il représente simplement l'intensité de la liaison entre deux variables qualitatives.

61. Conformité d'un coefficient de corrélation linéaire avec une valeur théorique

Dans la pratique, un coefficient de corrélation est le plus souvent comparé à la valeur nulle, ce qui permet de conclure s'il y a corrélation ou pas entre les deux variables.

La fonction `cor.test()` réalise systématiquement ce test et en revoie la *p - value* lorsqu'elle calcule un coefficient de corrélation de Pearson ou de Spearman (voir fiche **60**).

Pour comparer un coefficient de corrélation de Pearson à une valeur quelconque, utiliser la fonction `cor.conf()` du package `RVAideMemoire` : `cor.conf(variable1,variable2,theo=valeur)` où `variable1` et `variable2` sont des vecteurs contenant la valeur de chaque individu pour les deux variables (dans le même ordre), et `valeur` la valeur théorique à comparer (entre -1 et 1). Les conditions d'utilisation de ce test sont identiques à celles du calcul du coefficient de corrélation linéaire de Pearson (voir fiche **60**).

De façon plus générale, on peut tester la conformité d'un coefficient de corrélation (ou d'association) avec une valeur théorique quelconque simplement en regardant si celle - ci est contenue dans l'intervalle de confiance du coefficient (le niveau de précision de cet intervalle étant le plus souvent de 95 %, mais cette valeur est toujours modifiable).

62. Comparaison de plusieurs coefficients de corrélation linéaire

Ces tests ne s'appliquent qu'aux coefficients de corrélation linéaire de Pearson (voir fiche **60**).

Leurs conditions d'utilisation sont identiques à celles du calcul du coefficient de corrélation linéaire de Pearson (voir fiche **60**).

Comparaison de deux coefficients

Pour réaliser le test, utiliser selon la situation l'une des deux fonctions suivantes, contenues dans le package `RVAideMemoire` :

- `cor.2comp(variable1,variable2,variable3,variable4)` où `variable1` et `variable2` sont des vecteurs contenant la valeur de chaque individu pour les deux variables définissant la 1ère corrélation (dans le même ordre), tandis que `variable3` et `variable4` sont des vecteurs contenant la valeur de chaque individu pour les deux variables définissant la 2nde corrélation (dans le même ordre)
- `cor.multcomp(variable1,variable2,facteur)` où `variable1` et `variable2` sont des vecteurs contenant la valeur de chaque individu pour les deux variables à tester (dans le même ordre), et `facteur` un vecteur contenant la valeur de chaque individu pour le facteur définissant les deux groupes à comparer.

Si les deux coefficients de corrélation ne sont significativement pas différents, les deux fonctions renvoient la valeur du coefficient de corrélation commun, son intervalle de confiance à 95 % et le résultat du test de conformité de ce coefficient avec la valeur nulle (cette valeur théorique peut être modifiée grâce à l'argument `theo=valeur`).

Comparaison de plus de deux coefficients

Pour réaliser le test, utiliser la fonction `cor.multcomp()` du package `RVAideMemoire` : `cor.multcomp(variable1,variable2,facteur)` où `variable1` et `variable2` sont des vecteurs contenant la valeur de chaque individu pour les deux variables à tester (dans le même ordre), et `facteur` un vecteur contenant la valeur de chaque individu pour le facteur définissant les groupes à comparer (dans le même ordre que les deux premiers vecteurs).

Si les coefficients de corrélation ne sont significativement pas différents, la fonction renvoie la valeur du coefficient de corrélation commun, son intervalle de confiance à 95 % et le résultat du test de conformité de ce coefficient avec la valeur nulle (cette valeur théorique peut être modifiée grâce à l'argument `theo=valeur`).

Si la p - *value* du test est significative, cela indique qu'au moins deux coefficients diffèrent l'un de l'autre, sans préciser lesquels. La fonction effectue alors toutes les comparaisons deux - à - deux possibles.

63. La régression linéaire simple au sens des moindres carrés

Ce type de régression est utilisé lorsque l'on a une variable à expliquer et une variable explicative, la relation entre les deux étant linéaire.

Il est indispensable d'observer graphiquement la relation entre les deux variables avant d'envisager une régression linéaire (voir fiche **13**), ne serait - ce que pour contrôler si cette relation est effectivement linéaire.

Conditions : l'échantillonnage doit être aléatoire et simple ; chaque individu doit posséder une valeur pour les deux variables ; la relation entre les deux variables doit être linéaire ; pour chaque valeur de la variable explicative la distribution des résidus de la régression doit suivre une loi normale, de plus toutes ces lois normales doivent avoir la même variance (hypothèse dite d'*équivariance*) ; les résidus de la régression doivent être *indépendants* des deux variables (voir fiche **73** pour tester les deux conditions précédentes).

La régression s'écrit `regression<-lm(formule)`, où `lm()` construit un modèle linéaire, tandis que `formule` est la formule contenant la variable à expliquer et la variable explicative (voir fiche **71**).

Paramètres de la régression

Valeurs, erreurs standards et intervalles de confiance

La plupart des informations sur la régression sont données par la fonction `summary(regression)`. En particulier, la ligne `(Intercept)` du tableau `Coefficients` donne la valeur (`Estimate`) et l'erreur standard (`Std. Error`) de l'ordonnée à l'origine de la droite de régression, tandis que la ligne ayant le nom de la variable explicative donne la valeur et l'erreur standard du coefficient directeur (ou pente) de la droite.

Les intervalles de confiance de l'ordonnée à l'origine et du coefficient directeur sont obtenus grâce à la fonction `confint(regression)`.

Tests de conformité avec la valeur nulle

Lorsque la fonction `summary(regression)` est appelée, **R** réalise automatiquement un test de conformité des paramètres de la droite par rapport à zéro. Les *p - values* de ces tests sont données dans le tableau `Coefficients`, sur la ligne `(Intercept)` pour l'ordonnée à l'origine et sur la ligne ayant le nom de la variable explicative pour le coefficient directeur.

Analyse des contributions individuelles

La régression linéaire est très sensible aux individus extrêmes. Ceux - ci peuvent en effet avoir une grande influence sur ses paramètres. Il est donc indispensable de vérifier que la valeur des paramètres n'est pas due en grande partie à seulement un ou quelques individus. Pour cela utiliser la fonction `ind.contrib()` du package `RAideMemoire` : `ind.contrib(regression)`. La

fonction calcule la valeur des paramètres de la droite de régression en enlevant à tour de rôle chaque individu. Elle renvoie la différence entre les paramètres de la régression complète et ceux calculés, exprimée en proportion des paramètres de la régression complète. Par défaut elle trace un graphique pour représenter les résultats, pour obtenir seulement le tableau des résultats utiliser les arguments `graph=FALSE` et `print.diff=TRUE`.

Pouvoir explicatif de la régression

Le coefficient de détermination R^2 représente la part de la variance de la variable à expliquer qui est expliquée par la variable explicative. Il varie entre 0 (*i.e.* la variable explicative n'apporte aucune information) et 1 (*i.e.* les valeurs prises par la variable à expliquer sont totalement expliquées par la variable explicative). Graphiquement, plus R^2 est élevé et plus les points sont proches de la droite de régression.

Dans le cas de la régression linéaire simple (*i.e.* avec une seule variable explicative), le coefficient de détermination est égal au carré du coefficient de corrélation linéaire de Pearson (voir fiche **60**). La valeur de R^2 est donnée lorsque la fonction `summary(regression)` est appelée. Il se nomme `Multiple R-Squared`.

Prédiction à partir de la régression

Le but d'une régression linéaire est souvent de déterminer l'équation de la droite, qui sert ensuite à prédire les valeurs prises par la variable à expliquer à partir de valeurs connues de la variable explicative.

Cette prédiction peut être réalisée grâce à la fonction `predict(regression,list(explicative=valeur))` où `valeur` est soit un nombre, soit un vecteur de nombres correspondant aux valeurs de la variable explicative pour lesquelles on souhaite obtenir la valeur de la variable à expliquer.

Il faut toutefois être vigilant car l'équation de la droite de régression est établie à partir d'une certaine gamme de valeurs de la variable explicative. Il est donc d'autant plus hasardeux de prédire une valeur de la variable à expliquer à partir d'une valeur de la variable explicative que celle - ci est éloignée de cette gamme. Rien ne dit par exemple que la relation entre les deux variables est toujours linéaire en dehors de la gamme de valeurs qui a servi à définir la régression.

64. La régression linéaire simple au sens des moindres rectangles

Ce type de régression est utilisé lorsque l'on a deux variables considérées sur un pied d'égalité, aucune n'étant expliquée par l'autre. On dit ces variables interdépendantes.

En pratique la régression linéaire au sens des moindres rectangles est surtout utilisée en allométrie, où sont comparées plusieurs caractéristiques physiques d'un même organe ou organisme (en général l'une des deux variables est la taille ou la masse, ce qui permet d'utiliser ces mesures simples comme reflet de mesures plus difficiles à réaliser).

Il est indispensable d'observer graphiquement la relation entre les deux variables avant d'envisager une régression linéaire (voir fiche **13**), ne serait - ce que pour contrôler si cette relation est effectivement linéaire.

Conditions : l'échantillonnage doit être aléatoire et simple ; chaque individu doit posséder une valeur pour les deux variables ; la relation entre les deux variables doit être linéaire ; chaque variable doit avoir une distribution normale ; pour chaque valeur d'une variable, la distribution des valeurs possibles de la seconde doit suivre une loi normale (voir ci - dessous pour tester cette hypothèse, dite de *binormalité*).

La régression s'écrit grâce à la fonction `least.rect()` du package `RVAideMemoire` : `regression<-least.rect(variable.x,variable.y)`, où `variable.x` et `variable.y` sont des vecteurs contenant la valeur de chaque individu pour chaque variable (dans le même ordre). Les noms `variable.x` et `variable.y` n'ont qu'une valeur graphique : `variable.x` est destiné à être tracé en abscisses et `variable.y` en ordonnées.

Pour tester l'hypothèse de binormalité, tester la normalité des résidus de la régression de la 1ère variable sur la 2ème et inversement, en utilisant les fonctions `qqnorm()` et `shapiro.test()` (voir fiches **38** et **73**) sur `lm(variable.x~variable.y)$resid` et `lm(variable.y~variable.x)$resid`.

Paramètres de la régression

Valeurs et intervalles de confiance

Toutes les informations importantes sur la régression sont données en appelant `regression`. En particulier, la fonction renvoie l'équation de la droite de régression et la valeur de ses paramètres : `(Intercept)` correspond à l'ordonnée à l'origine et le coefficient `variable.x` correspond au coefficient directeur de la droite. La fonction renvoie également l'intervalle de confiance à 95 % de ces paramètres.

Test de conformité du coefficient directeur avec une valeur théorique

Le coefficient directeur est généralement comparé à la valeur 1, qui correspond en allométrie à une relation d'isométrie entre les deux caractéristiques

comparées. Par défaut la fonction `least.rect()` utilise cette valeur 1 pour réaliser le test, dont le résultat est renvoyé en appelant `regression`. La valeur théorique peut être modifiée grâce à l'argument `theo=valeur`.

Analyse des contributions individuelles

La régression linéaire est très sensible aux individus extrêmes. Ceux - ci peuvent en effet avoir une grande influence sur ses paramètres. Il est donc indispensable de vérifier que la valeur des paramètres n'est pas due en grande partie à seulement un ou quelques individus. Pour cela utiliser la fonction `ind.contrib()` du package `RAideMemoire` : `ind.contrib(regression)`. La fonction calcule la valeur des paramètres de la droite de régression en enlevant à tour de rôle chaque individu. Elle renvoie la différence entre les paramètres de la régression complète et ceux calculés, exprimée en proportion des paramètres de la régression complète. Par défaut elle trace un graphique pour représenter les résultats, pour obtenir seulement le tableau des résultats utiliser les arguments `graph=FALSE` et `print.diff=TRUE`.

Précision de la régression

En comparaison avec la méthode des moindres carrés et son coefficient de détermination R^2 (voir fiche **63**), il n'y a pas pour la régression linéaire au sens des moindres rectangles de quantification de cette précision. Cependant on peut utiliser le coefficient de corrélation linéaire de Pearson (voir fiche **60**) pour s'en faire une idée : plus ce coefficient est proche de 1 ou -1, plus la précision est grande. Un appel à `regression` renvoie ce coefficient de corrélation.

Prédiction à partir de la régression

Le but d'une régression au sens des moindres rectangles n'est généralement pas de prédire les valeurs d'une variable, puisqu'aucune des deux variables n'est explicative (et donc contrôlée). Cependant, s'il en est besoin, il suffit simplement d'utiliser l'équation de la droite renvoyée en appelant `regression`.

Il faut toutefois être vigilant car l'équation de la droite de régression est établie à partir d'une certaine gamme de valeurs de `variable.x`. Il est donc d'autant plus hasardeux de prédire une valeur de `variable.y` à partir d'une valeur de `variable.x` que celle - ci est éloignée de cette gamme. Rien ne dit par exemple que la relation entre les deux variables est toujours linéaire en dehors de la gamme de valeurs qui a servi à définir la régression.

65. Comparaison de plusieurs droites de régression linéaire simple

Droites de régression au sens des moindres carrés

L'analyse réalisée est en fait une analyse de la covariance (*ANCOVA*), puisqu'elle fait intervenir une variable à expliquer quantitative, une variable explicative quantitative (la covariable) et une variable explicative qualitative (le facteur, qui définit les droites à comparer).

Conditions : identiques à celles de l'analyse de la variance (voir fiche **55**), de plus la relation entre la variable à expliquer et la covariable doit être linéaire.

Commencer par écrire le modèle `modele<-lm(a.expliquer~covariable*facteur)` où `a.expliquer`, `covariable` et `facteur` sont des vecteurs contenant la valeur de chaque individu pour chacune des trois variables (dans le même ordre).

L'ANCOVA est réalisée grâce la fonction `anova(modele)`. Le tableau renvoyé donne l'effet de la covariable, celui du facteur et celui de leur interaction.

Si la *p - value* associée à la covariable est significative, cela indique que les coefficients directeurs des droites de régression sont différents de la valeur nulle.

Si la *p - value* associée à l'interaction facteur - covariable est significative, cela indique qu'au moins deux coefficients directeurs sont différents l'un de l'autre, sans préciser lesquels. Pour identifier ces coefficients, utiliser la fonction `reg.slpcomp()` du package `RVAideMemoire` : `reg.slpcomp(a.expliquer,covariable,facteur)`. Cette fonction renvoie la valeur et l'intervalle de confiance de chaque coefficient directeur. Elle effectue également toutes les comparaisons deux - à - deux possibles.

Dans le cas de coefficients directeurs différents (*i.e.* de droites non parallèles), les régressions doivent être étudiées séparément (voir fiche **63**).

Si la *p - value* associée à l'interaction facteur - covariable n'est pas significative, cela indique que les coefficients directeurs ne sont pas différents (*i.e.* les droites de régression sont parallèles). On peut dans ce cas calculer la valeur du coefficient commun et comparer les ordonnées à l'origine des différentes droites de régression. Utiliser pour cela la fonction `reg.intcomp()` du package `RVAideMemoire` : `reg.intcomp(a.expliquer,covariable,facteur)`. Cette fonction renvoie la valeur et l'intervalle de confiance du coefficient directeur commun à toutes les régressions, l'ordonnée à l'origine de chaque régression ainsi que son intervalle de confiance et le résultat du test de conformité avec la valeur nulle (cette valeur théorique peut être modifiée par l'argument `theo=valeur`, où `valeur` est un vecteur contenant l'ordonnée à l'origine théorique de chaque régression, dans l'ordre alphabétique des modalités du facteur). Elle effectue également toutes les comparaisons deux - à - deux possibles entre les ordonnées à l'orgine.

Finalement, les régressions qui ne diffèrent ni par leur coefficient directeur ni par leur ordonnée à l'origine peuvent être regroupées (*i.e.* les classes correspondantes du facteur peuvent être regroupées).

Droites de régression au sens des moindres rectangles

Il n'existe pas de test statistique simple permettant de comparer des droites de régression au sens des moindres rectangles. On peut cependant utiliser la fonction `lr.multcomp()` du package `RVAideMemoire`, qui compare simplement les *intervalles de confiance* des paramètres des droites de régression.

Ses conditions d'utilisation sont identiques à celles de la régression linéaire simple au sens des moindres rectangles (voir fiche **64**).

Pour réaliser les comparaisons : `lr.multcomp(variable.x,variable.y,facteur)` où `variable.x`, `variable.y` et `facteur` sont des vecteurs contenant la valeur de chaque individu pour chacune des trois variables (dans le même ordre). Les différentes droites de régression sont définies par rapport aux classes du facteur. Les noms `variable.x` et `variable.y` n'ont qu'une valeur graphique : `variable.x` est tracé en abscisses, `variable.y` en ordonnées.

66. La régression logistique binaire simple

Ce type de régression est utilisé lorsque l'on a une variable à expliquer binaire (0 = pas de réponse, 1 = réponse ; ex : mort ou vivant) et une variable explicative quantitative continue. La variable étudiée est donc une probabilité de réponse.

Conditions : l'échantillonnage doit être aléatoire et simple ; les classes de la variable à expliquer doivent être exclusives.

Effet de la variable explicative sur la variable à expliquer

Choix du modèle

Commencer par construire le Modèle Linéaire Généralisé (*GLM*) censé représenter les données : `modele1<-glm(formule,family=binomial(link="logit"))` où `formule` est la formule contenant la variable à expliquer et la variable explicative (voir fiche **71**). La loi binomiale (qui représente en fait la loi de distribution des *erreurs* du modèle) est à utiliser lorsque la variable à expliquer est binaire. La précision `link="logit"` décrit la fonction de lien qui rend la relation entre les deux variables linéaires (ce n'est pas vraiment *directement* la relation entre les variables qui est rendue linéaire, mais peu importe ici).

Appeler ensuite le résumé du modèle *via* `summary(modele1)`. Parmi les informations renvoyées, noter la valeur de la déviance résiduelle (`Residual deviance`). Cette valeur représente de façon globale l'écart entre les valeurs prédites par le modèle et les données réelles. Elle doit donc être la plus faible possible. Comparer cette déviance résiduelle avec celle du second modèle : `modele2<-glm(formule,family=binomial(link="cloglog"))`. Choisir celui des deux modèles qui possède la plus petite déviance résiduelle pour continuer l'analyse. On le nommera simplement `modele.lineaire`.

Effet de la variable explicative

Appeler le résumé du modèle *via* `summary(modele.lineaire)`. Parmi les informations renvoyées, le tableau `Coefficients` donne la valeur (`Estimate`) du coefficient directeur de la droite de régression (cette droite de régression *ne relie pas directement* la variable à expliquer avec la variable explicative), ainsi que le résultat du test de conformité de ce coefficient avec la valeur nulle. Ce qu'il faut observer ici est le résultat du test de conformité. Si la p - *value* est significative, cela veut dire que la variable explicative a réellement un effet sur la probabilité de réponse et l'analyse peut continuer. Dans le cas contraire l'analyse s'arrête car la variable explicative n'a pas d'effet.

Paramètres de la courbe de régression

Valeurs, erreurs standards et intervalles de confiance

Si la variable explicative a un effet sur la probabilité de réponse, la courbe

de régression (celle qui *relie directement* les deux variables) est logistique (*i.e.* sigmoïde). Il faut construire le modèle de cette régression pour en obtenir les paramètres (ici un modèle à trois paramètres est utilisé, voir fiche **67** pour un modèle à quatre paramètres). Utiliser pour cela la procédure suivante, qui fait appel à la fonction `logis.noise()` du package `RVAideMemoire` :

```
> y<-logis.noise(modele.lineaire)
> modele.logistique<-nls(y~SSlogis(explicative,Asymp,mid,scale))
```

Appeler ensuite le résumé du modèle *via* `summary(modele.logistique)`. Parmi les informations renvoyées, le tableau `Coefficients` donne la valeur (`Estimate`) et l'erreur standard (`Std. Error`) des trois paramètres de la courbe : l'asymptote (`Asymp`), l'abscisse du point d'inflexion (`mid`) et l'échelle (`scale`).

Les intervalles de confiance de ces paramètres sont obtenus grâce à la fonction `confint(modele.logistique)`.

Tests de conformité avec la valeur nulle

Lorsque la fonction `summary(modele.logistique)` est appelée, R réalise automatiquement un test de conformité des paramètres de la courbe par rapport à la valeur nulle. Les *p - values* de ces tests sont données dans le tableau `Coefficients`, sur la ligne de chaque paramètre.

Ajustement du modèle aux données

Il est toujours difficile dans le cas d'une régression logistique binaire de savoir si le modèle est bien ajusté aux données. On peut s'en faire une idée graphiquement grâce à la fonction `logis.fit()` du package `RVAideMemoire`. Il faut au préalable avoir tracé la courbe de régression (voir fiche **68**), puis utiliser : `logis.fit(modele.lineaire)`.

La fonction découpe en fait les valeurs de la variable explicative en un certains nombre d'intervalles (par défaut 5, pour changer cette valeur utiliser l'argument `int`), et calcule la probabilité de réponse et son erreur standard pour chaque intervalle. Les points et barres d'erreurs correspondants sont ensuite ajoutés sur la courbe de régression.

Il faut garder en tête que cette procédure comporte une grande part d'arbitraire (surtout dans le choix du nombre d'intervalles) et n'a rien à voir avec un quelconque test.

Prédiction à partir de la régression

Le but de la régression logistique est souvent de prédire la probabilité de réponse selon une valeur donnée de la variable explicative. Cette prédiction peut être réalisée grâce à la fonction `predict(modele.lineaire,list(explicative=valeur),type="response")` où `valeur` est soit un nombre, soit un vecteur de nombres correspondant aux valeurs de la variable explicative pour lesquelles on souhaite obtenir la probabilité de réponse.

67. La régression non linéaire simple

Ce type de régression est utilisé lorsque que la relation entre une variable à expliquer et une variable explicative n'est pas linéaire, mais que l'on connaît le type d'équation qui peut la représenter. L'observation graphique de cette relation est donc primordiale pour choisir le type d'équation (voir fiche **13**).

Paramètres de la régression

Régression quadratique (ou polynomiale du 2nd degré)

Elle s'écrit `regression<-lm(a.expliquer~explicative+I(explicative^2))` où `a.expliquer` et `explicative` sont des vecteurs contenant la valeur de chaque individu pour les deux variables (dans le même ordre), et `I()` une fonction indiquant que ce qui est situé entre ses parenthèses est une formule mathématique.

La régression est analysée grâce à `summary(regression)` de la même façon qu'une régression linéaire simple au sens des moindres carrés (valeur, erreur standard et conformité des paramètres avec la valeur nulle, pouvoir explicatif et prédiction ; voir fiche **63**).

Elle peut être étendue à une régression polynomiale de degré supérieur en rajoutant les autres termes *via* la fonction `I()` dans la formule.

Autres régressions non linéaires

Leur syntaxe est toujours de la forme `regression<-nls(a.expliquer~equation)` où `nls()` est la fonction construisant la régression et `equation` une fonction dépendante de l'équation choisie, la plupart du temps parmi les suivantes :

Type de régression	Equation	`equation`
Asymptotique		
Michaelis - Menten	$y = \frac{ax}{1+bx}$	`SSmicmen(explicative,a,b)`
Exponentielle à 2 paramètres	$y = a(1 - e^{-bx})$	`SSasympOrig(explicative,a,b`
Exponentielle à 3 paramètres	$y = a - be^{-cx}$	`SSasymp(explicative,a,b,c)`
Sigmoïde		
Logistique à 3 paramètres	$y = \frac{a}{1+be^{-cx}}$	`SSlogis(explicative,a,b,c)`
Logistique à 4 paramètres	$y = a + \frac{b-a}{1+e^{\frac{c-x}{d}}}$	`SSfpl(explicative,a,b,c,d)`
Weibull	$y = a - be^{-(cx^d)}$	`SSweibull(explicative,a,b,c,d)`
Gompertz	$y = ae^{-be^{-cx}}$	`SSgompertz(explicative,a,b,c)`
En cloche		
Biexponentielle	$y = ae^{bx} - ce^{-dx}$	`SSbiexp(explicative,a,b,c,d)`

La valeur, l'erreur standard et le résultat du test de conformité avec la valeur nulle des paramètres de l'équation sont donnés par la fonction `summary(regression)`. L'intervalle de confiance à 95 % des paramètres est donné par la fonction `confint(regression)`.

Prédiction à partir de la régression

Quelle que soit la régression, la syntaxe est toujours de la forme `predict(regression,list(explicative=valeur))` où `valeur` est soit un nombre, soit un vecteur de nombres correspondant aux valeurs de la variable explicative pour lesquelles on souhaite obtenir la valeur de la variable à expliquer.

Il faut toutefois être vigilant car l'équation de la droite de régression est établie à partir d'une certaine gamme de valeurs de la variable explicative. Il est donc d'autant plus hasardeux de prédire une valeur de la variable à expliquer à partir d'une valeur de la variable explicative que celle - ci est éloignée de cette gamme. Rien ne dit par exemple que la relation entre les deux variables est toujours de la même forme en dehors de la gamme de valeurs qui a servi à définir la régression.

68. ℹ Tracer une droite ou une courbe de régression simple

La première étape est identique pour tous les types de régression : représenter le nuage de points. Ceci est obtenu simplement *via* `plot(explicative,a.expliquer)` où `explicative` et `a.expliquer` sont des vecteurs contenant la valeur de chaque individu pour chaque variable. Dans le cas de la régression linéaire au sens des moindres rectangles (où les deux variables sont interdépendantes), la syntaxe est `plot(variable.x,variable.y)`.

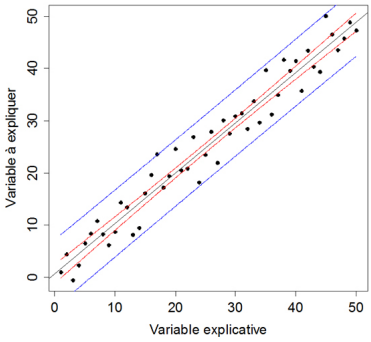
Régression linéaire au sens des moindres carrés

La droite s'obtient simplement *via* `abline(regression)` (voir fiche **63** pour plus d'informations sur la régression linéaire simple au sens des moindres carrés).

Il est possible de tracer un intervalle de confiance autour de la droite de régression. On s'intéresse généralement :

- soit à l'intervalle de confiance des points de la droite, *i.e.* l'intervalle de confiance de la *moyenne* de la variable à expliquer pour une valeur donnée de la variable explicative
- soit à l'intervalle de confiance des valeurs individuelles, *i.e.* l'intervalle de confiance des *valeurs individuelles* prises par la la variable à expliquer pour une valeur donnée de la variable explicative.

Pour ajouter l'un de ces intervalles de confiance sur la droite, utiliser la fonction `reg.ci()` du package `RVAideMemoire` : `reg.ci(regression,type=type.ci)`, où `type.ci` vaut `"mean"` pour l'intervalle de confiance des moyennes ou `"ind"` pour l'intervalle de confiance des valeurs individuelles.



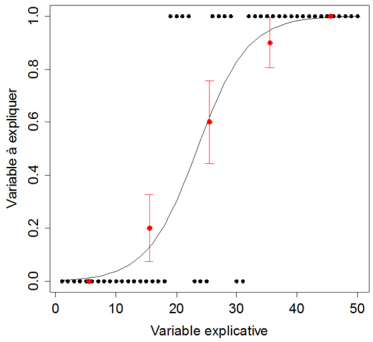
Régression linéaire au sens des moindres rectangles

La droite s'obtient simplement *via* `abline(regression)` (voir fiche **64** pour plus d'informations sur la régression linéaire simple au sens des moindres rectangles).

Régression logistique binaire

La courbe s'obtient simplement *via* `lines(explicative,modele.lineaire$fitted.values)` (voir fiche **66** pour plus d'informations sur la régression logistique binaire simple).

On peut se faire une idée graphique de l'ajustement du modèle aux données en utilisant la fonction `logis.fit()` du package `RVAideMemoire` : `logis.fit(modele.lineaire)` (voir fiche **66**).

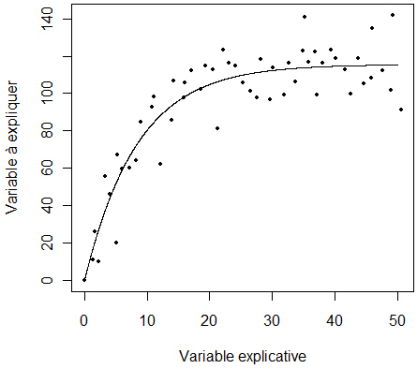


Autres régressions non linéaires

La procédure est légèrement plus longue dans ce cas. Elle fait appel à la fonction `seq2()` du package `RVAideMemoire` :

```
> x<-seq2(explicative)
> y<-predict(regression,list(explicative=x))
> lines(x,y)
```

Voir fiche **67** pour plus d'informations sur la régression non linéaire simple.



69. L'analyse de la covariance – un facteur

Ce type d'analyse est utilisé lorsque l'on a une variable à expliquer quantitative et deux variables explicatives, l'une quantitative (la covariable) et l'autre qualitative (le facteur).

L'analyse de la covariance (*ANCOVA*) peut être employée pour des données telles que effectifs, des proportions, des probabilités de réponse, des moyennes ou encore pour comparer des droites de régression linéaire au sens des moindres carrés (sur ce dernier point, voir fiche **63**).

Cette fiche ne traite pas de l'ANCOVA mixte (*i.e.* avec facteur aléatoire). Pour plus d'informations, voir **Bates** (2010).

Moyennes

Dans ce cadre, l'ANCOVA est utilisée pour comparer des moyennes en fonction des modalités d'un facteur, tout en tenant compte d'une variable auxiliaire (la covariable). Le but est d'éliminer l'influence de la covariable, qui peut être une variable différente de la variable à expliquer ou une mesure antérieure (ex : en début d'expérience) de cette même variable à expliquer.

La covariable ne doit pas être influencée par le facteur. Si elle l'est, il convient d'utiliser l'analyse de la covariance avec beaucoup de prudence (surtout au moment de conclure).

Conditions : identiques à celles de l'analyse de la variance (voir fiche **55**), de plus la relation entre la variable à expliquer et la covariable doit être linéaire.

Commencer par écrire le modèle censé représenter les données : `modele<-lm(a.expliquer~covariable+facteur)` où `a.expliquer`, `covariable` et `facteur` sont des vecteurs contenant la valeur de chaque individu pour chacune des trois variables (dans le même ordre). L'analyse est ensuite réalisée *via* à la fonction `anova(modele)`.

Le tableau renvoyé donne l'effet de la covariable et celui du facteur. Si la *p - value* associée au facteur est significative, cela indique qu'au moins deux moyennes diffèrent l'une de l'autre (sans préciser lesquelles). Il est dans ce cas nécessaire de réaliser des comparaisons deux - à - deux pour identifier les moyennes en question. Utiliser pour cela la fonction `scheffe.test()` du package `agricolae` : `scheffe.test(modele,trt="facteur")` (le facteur doit être entre guillemets).

Effectifs, proportions et probabilités de réponse

Dans ce cadre, l'ANCOVA est utilisée :

– soit pour comparer les valeurs de la variable à expliquer en fonction des modalités du facteur, tout en tenant compte d'une variable auxiliaire (la covariable). Le but est ici d'éliminer l'influence de la covariable, qui peut être une variable différente de la variable à expliquer ou une mesure antérieure (ex : en début d'expérience) de cette même variable à expliquer

– soit pour comparer la relation entre la variable et la covariable en fonction des modalités du facteur.

Quel que soit l'objectif, la covariable ne doit pas être influencée par le facteur. Si elle l'est, il convient d'utiliser l'analyse de la covariance avec beaucoup de prudence (surtout au moment de conclure).

Conditions : l'échantillonnage doit être aléatoire et simple ; les classes du facteur doivent être exclusives.

Commencer par écrire le modèle censé représenter les données : `modele<-glm(formule,family=loi)` où `formule` est la formule contenant la variable à expliquer, le facteur et la covariable (voir fiche **71**), et `loi` dépend du type de variable à expliquer (voir fiche **41** pour des probabilités de réponse, **45** pour des effectifs et **51** pour des proportions). Attention, selon l'objectif de l'analyse l'ordre des variables explicatives n'est pas forcément le même (voir fiche **71**).

Si la variable à expliquer représente des effectifs ou des proportions, il est indispensable de vérifier s'il n'y a pas surdispersion des résidus. Si c'est le cas il est nécessaire de remplacer la loi du modèle par `quasipoisson` (voir fiche **45**) ou `quasibinomial` (voir fiche **51**), ou d'utiliser la fonction `glm.nb()` du package `MASS` (voir fiche **45**).

L'analyse est ensuite réalisée *via* la fonction `anova(modele)`, en précisant `test="F"` si la loi est `quasipoisson` ou `quasibinomial`, `test="Chi"` dans tous les autres cas . Le tableau renvoyé donne l'effet de la covariable, celui du facteur et celui de leur interaction.

Si l'interaction facteur - covariable a été prise en compte dans le modèle et que la *p - value* associée est significative, cela indique qu'il y a une relation entre la variable à expliquer et la covariable, mais qu'elle est différente selon la modalité du facteur.

Si la *p - value* associée à la covariable est significative et que l'interaction ne l'est pas (ou qu'elle n'a pas été prise en compte dans le modèle), cela indique qu'il y a une relation entre la variable à expliquer et la covariable, indépendamment de la modalité du facteur. L'analyse peut donc être réduite à une analyse de la régression entre la variable à expliquer et la covariable (voir fiche **66** pour des probabilités de réponse, non développé dans cet ouvrage pour les effectifs et les proportions ; cependant la procédure est la même que l'ANCOVA, sans le facteur).

Si la *p - value* associée au facteur est significative et que l'interaction ne l'est pas (ou qu'elle n'a pas été prise en compte dans le modèle), cela signifie que les valeurs de la variable à expliquer sont différentes selon la modalité du facteur, indépendamment de la covariable. L'analyse peut donc être réduite à une comparaison de probabilités de réponse (voir fiche **41**), d'effectifs (voir fiche **45**) ou de proportions (voir fiche **51**).

Quelle que soit la loi utilisée dans le modèle, il est nécessaire de vérifier que celui - ci s'ajuste bien aux données (voir fiche **73**).

70. La régression linéaire multiple

Une régression linéaire multiple est une régression linéaire au sens des moindres carrés (voir fiche **63**) où le nombre de variables explicatives est au moins égal à deux.

Conditions : l'échantillonnage doit être aléatoire et simple ; chaque individu doit posséder une valeur pour chaque variable ; la relation entre la variable à expliquer et les variables explicatives doit être linéaire ; pour chaque valeur des variables explicatives la distribution des résidus de la régression doit suivre une loi normale, de plus toutes ces lois normales doivent avoir la même variance (hypothèse d'*équivariance*) ; les résidus de la régression doivent être *indépendants* des variables explicatives (voir fiche **73** pour tester les deux conditions précédentes).

La régression s'écrit `regression<-lm(formule` où `lm()` construit un modèle linéaire, tandis que `formule` est la formule est la formule contenant la variable à expliquer et les variables explicatives (voir fiche **71**).

Paramètres de la régression

Valeurs, erreurs standards et intervalles de confiances

La plupart des informations sur la régression sont données par la fonction `summary(regression)`. En particulier, la ligne `(Intercept)` du tableau `Coefficients` donne la valeur (`Estimate`) et l'erreur standard (`Std. Error`) de l'ordonnée à l'origine de la droite de régression, tandis que la ligne associée à chaque variable explicative donne la valeur et l'erreur standard de son coefficient dans l'équation de la droite.

Les intervalles de confiance de l'ordonnée à l'origine et des coefficients sont obtenus grâce à la fonction `confint(regression)`.

Tests de conformité avec la valeur nulle

Lorsque la fonction `summary(regression)` est appelée, **R** réalise automatiquement un test de conformité des paramètres de la droite par rapport à zéro. Les p - *values* de ces tests sont données dans le tableau `Coefficients`, sur la ligne `(Intercept)` pour l'ordonnée à l'origine et sur la ligne correspondant à chaque variable explicative pour son coefficient associé.

Analyse des contributions individuelles

La régression linéaire est très sensible aux individus extrêmes. Ceux - ci peuvent en effet avoir une grande influence sur ses paramètres. Il est donc indispensable de vérifier que la valeur des paramètres n'est pas due en grande partie à seulement un ou quelques individus. Pour cela utiliser la fonction `ind.contrib()` du package `RAideMemoire` : `ind.contrib(regression)`. La fonction calcule la valeur des paramètres de la droite de régression en enlevant à tour de rôle chaque individu. Elle renvoie la différence entre les paramètres de

la régression complète et ceux calculés, exprimée en proportion des paramètres de la régression complète. Par défaut elle trace un graphique pour représenter les résultats, pour obtenir seulement le tableau des résultats utiliser les arguments `graph=FALSE` et `print.diff=TRUE`.

Pouvoir explicatif de la régression

Le coefficient de détermination R^2 représente la part de la variance de la variable à expliquer qui est expliquée par la régression. Il varie entre 0 (*i.e.* la régression n'apporte aucune information) et 1 (*i.e.* les valeurs prises par la variable à expliquer sont totalement expliquées par la régression). Graphiquement, plus R^2 est élevé et plus les points sont proches de la droite de régression.

Sa valeur (ajustée dans le cas de la régression multiple) est donnée lorsque la fonction `summary(regression)` est appelée. Il se nomme `Adjusted R-Squared`.

Prédiction à partir de la régression

Le but d'une régression linéaire est souvent de déterminer l'équation de la droite, qui sert ensuite à prédire les valeurs prises par la variable à expliquer à partir de valeurs connues de la variable explicative.

Cette prédiction peut être réalisée grâce à la fonction `predict(regression,list(explicative1=valeur1,explicative2=valeur2,...))` où `valeur1`, `valeur2`, ... sont soit des nombres, soit des vecteurs de nombres correspondant aux valeurs des variables explicatives pour lesquelles on souhaite obtenir la valeur de la variable à expliquer.

Il faut toutefois être vigilant car l'équation de la droite de régression est établie à partir d'une certaine gamme de valeurs des variables explicatives. Il est donc d'autant plus hasardeux de prédire une valeur de la variable à expliquer à partir de valeurs des variables explicatives que celles - ci sont éloignées de cette gamme. Rien ne dit par exemple que la relation entre les variables est toujours linéaire en dehors de la gamme de valeurs qui a servi à définir la régression.

71. Construction de la formule d'un modèle

La construction de modèles est un élément essentiel dans l'analyse statistique des résultats d'une étude. Elle peut être relativement simple si les notions suivantes sont bien comprises :

- variable à expliquer et variable explicative
- variable quantitative et facteur (voir fiche **1**)
- facteur fixe et facteur aléatoire (voir fiche **1**)
- plan d'échantillonnage (voir fiche **2**) et plan d'expérience (voir fiche **3**).

Nous nous arrêtons dans cet ouvrage aux modèles à deux variables explicatives. Le principe est cependant le même au - delà de deux variables.

Formules à une variable explicative

Formules à facteur fixe ou variable explicative quantitative

Ces formules sont les plus simples à construire. Elles s'écrivent simplement `a.expliquer~explicative`, où `a.expliquer` et `explicative` sont des vecteurs contenant la valeur de chaque individu pour chaque variable (dans le même ordre). Le symbole `~` signifie « expliqué par ».

Si la variable explicative est quantitative cette formule s'utilise dans une régression simple, tandis que si c'est un facteur elle s'utilise (le plus souvent) dans un modèle linéaire ou linéaire généralisé.

Formules mixtes

On appelle *mixte* une formule qui contient au moins un facteur aléatoire. C'est le cas notamment lorsque l'on veut prendre en compte des séries appariées (voir fiche **1**), des grappes (voir fiche **2**) ou des blocs (voir fiche **3**).

Les modèles mixtes sont gérés par les packages `nlme` et `lme4`. Les formules sont écrites différemment selon le package utilisé. Dans cet ouvrage, l'utilisation du package `lme4` a été privilégiée.

La formule s'écrit `a.expliquer~explicative+(1|aleatoire)`, où `aleatoire` est un vecteur contenant la valeur de chaque individu pour le facteur aléatoire (dans le même ordre qu'`a.expliquer` et `explicative`). La syntaxe `(1|x)` indique que le facteur `x` est aléatoire.

Si la variable explicative est quantitative cette formule s'utilise dans une régression simple mixte, tandis que si c'est un facteur elle s'utilise dans un modèle linéaire mixte ou linéaire généralisé mixte.

Formules à deux variables explicatives

Attention, si le nombre de répétitions n'est pas identique pour les deux variables, leur ordre d'entrée dans le modèle a une importance sur le calcul de leur effet. En effet, **R** calcule l'effet de chaque variable de façon *séquentielle*, *i.e.* il calcule celui de la 1ère variable puis celui de la 2nde *sachant la 1ère*. L'effet

de la 2nde variable est donc calculé sur la variation de la variable à expliquer qui reste *après avoir retiré* la variation due à la 1ère variable explicative.

Il faut donc bien réfléchir au sens biologique des variables explicatives :

- si l'objectif est d'éliminer l'influence d'une variable avant de calculer l'effet d'une autre, placer celle dont on veut éliminer l'influence en premier
- si l'on connaît *a priori* l'importance relative des deux variables dans le système biologique, placer celle qui a le plus d'importance en premier
- si l'on a aucune idée *a priori*, tester les deux modèles (en inversant l'ordre des variables dans le deuxième), comparer les résultats puis interpréter leurs différences éventuelles en terme biologique pour retenir le modèle le plus pertinent.

Formules à facteur(s) fixe(s) et / ou variable(s) explicative(s) continue(s)

A partir de deux variables explicatives, il est nécessaire de se poser la question de la relation qui lie ces variables. Deux cas de figures sont ainsi possibles :

- les variables explicatives sont *croisées* : c'est le cas lorsque tous les croisements entre les modalités de la 1ère variable et celles de la 2nde sont représentés par au moins un individu.
  Pour écrire la formule, il faut se poser une autre question : l'interaction entre les deux variable (*i.e.* le fait que l'effet de l'une puisse dépendre de la valeur prise par la seconde) doit - elle être prise en compte ?
  - si non, le modèle est *additif*. La formule s'écrit alors :
    `a.expliquer~explicative1+explicative2`
  - si oui, le modèle est *multiplicatif*. La formule s'écrit alors :
    `a.expliquer~explicative1*explicative2`
    La partie droite de cette formule est identique à : `explicative1+explicative2+explicative1:explicative2`, *i.e.* « 1ère variable » + « 2nde variable » + « interaction entre les deux »

  Si les deux variables explicatives sont quantitatives la formule s'utilise dans une régression multiple, si ce sont deux facteurs elle s'utilise (le plus souvent) dans un modèle linéaire ou linéaire généralisé, enfin si l'une est quantitative et l'autre est un facteur la formule s'utilise dans une analyse de la covariance
- les variables explicatives sont *hiérarchisées* : dans ce cas une variable est subordonnée à l'autre (ex : des populations subordonnées à des régions). On ne peut donc pas croiser toutes les modalités de la 1ère variable avec celles de la 2nde. La formule s'écrit :
  `a.expliquer~explicative1/explicative2`, où le slash (`/`) signifie que la variable de droite est subordonnée à celle de gauche.
  La partie droite de cette formule est identique à : `explicative1+explicative1:explicative2`, *i.e.* « 1ère variable » + « interaction entre les deux ».

Formules mixtes

Si écrire une formule à deux variables explicatives peut être compliqué, ajouter un facteur aléatoire est relativement simple :

- pour un modèle croisé :
  - additif : `a.expliquer~explicative1+explicative2+(1|aleatoire)`
  - multiplicatif : `a.expliquer~explicative1*explicative2+(1|aleatoire)`
- pour un modèle hiérarchisé : `a.expliquer~explicative1/explicative2+(1|aleatoire)`

Formules des modèles nuls

On appelle *nul* un modèle qui ne contient aucune variable explicative. Sa formule est donc simple :

- sans facteur aléatoire : `a.expliquer~1`
- avec facteur aléatoire : `a.expliquer~1+(1|aleatoire)`

72. Sélection de modèle

Les modèles présentés dans cet ouvrage sont toujours relativement simples, ne contenant qu'une ou deux variables explicatives. Cependant il arrive fréquemment que l'analyse débute avec un grand nombre de facteurs et / ou de covariables. L'objectif devant toujours être d'être le plus parcimonieux possible dans l'explication du phénomène observé, l'étape de sélection du modèle le plus simple *et* pertinent est très importante.

Cette procédure de sélection peut se faire de deux manières, à la main ou automatiquement.

Sélection manuelle

Deux étapes sont nécessaires :

1. appeler `summary(modele1)` où `modele1` est le modèle le plus complexe (dit *saturé*), intégrant toutes les variables explicatives et leurs interactions. Noter le terme le moins significatif. Créer ensuite un 2ème modèle identique au premier mais en retirant ce terme, en commençant toujours par les interactions de plus grand ordre (une interaction entre deux variables s'écrit `variable1:variable2`). Ce 2ème modèle peut être construit très simplement *via* `modele2<-update(modele1,~.-terme)`. La fonction reprend le `modele1` avec toutes ses variables explicatives (ce que signifie le symbole `~.`), mais ôte le terme désiré (qui doit être précédé du signe `-`)
2. comparer les deux modèles *via* `anova(modele1,modele2)`. Si les modèles ont été créés *via* la fonction `glm()`, préciser `test="F"` pour une loi `quasipoisson` ou `quasibinomial`, `test="Chi"` dans tous les autres cas. La *p - value* renvoyée par le test indique l'effet du terme ôté dans le 2ème modèle. Si celle - ci est significative, le terme en question ne peut être enlevé du modèle, dans le cas contraire celui - ci peut être simplifié.

Répéter la procédure en testant chaque terme un à un et en simplifiant ainsi le modèle saturé petit à petit.

*Note sur les Modèles Linéaires Mixtes (*LMM*)* : tous les modèles à comparer doivent avoir été créés avec l'option `REML=FALSE` (par défaut `REML=TRUE`). Une fois le meilleur modèle déterminé, il doit être récrit avec l'option `REML=TRUE` pour être analysé.

Sélection automatique

La sélection est basée sur une valeur représentant l'ajustement du modèle aux données, le Critère d'Information d'Akaike (*AIC*). Plus l'AIC est faible, meilleure est l'adéquation du modèle.

Utiliser la fonction `dredge()` du package `MuMIn` : `dredge(modele1)`. Cette fonction calcule l'AIC de tous les modèles possibles à partir du modèle saturé

et renvoie un tableau classant tous ces modèles. Parmi les arguments facultatifs de la fonction, deux sont particulièrement intéressants :

- `m.max=valeur`, où `valeur` est le nombre maximal de variables explicatives à intégrer dans les modèles à tester
- `fixed=variables`, où `variables` est un vecteur contenant les variables explicatives à intégrer dans tous les modèles à tester (entre guillemets).

Attention, si $\frac{n}{k} < 40$, où n est le nombre d'individus et k le nombre de paramètres estimés par modèle (renvoyé par le fonction `dredge()`), il faut utiliser l'AICc (AIC corrigé) et non pas l'AIC. Dans le cas de modèles avec une loi `quasipoisson` ou `quasibinomial`, le critère utilisé est le QAICc, dérivé de l'AIC pour les distributions « quasi ».

*Note sur les Modèles Linéaires Mixtes (*LMM*)* : le modèle saturé (`modele1`) doit avoir été créé avec l'option `REML=FALSE` (par défaut `REML=TRUE`). Une fois le meilleur modèle déterminé, il doit être récrit avec l'option `REML=TRUE` pour être analysé.

Il faut être conscient d'une chose lorsque l'on utilise une procédure de sélection automatique : le modèle avec l'AIC le plus faible n'est pas forcément celui qui *biologiquement* a le plus de sens. Il ne faut donc pas utiliser cette procédure aveuglément mais toujours réfléchir aux variables et interactions qui sont retenues. Il est ainsi possible d'enlever des termes manuellement si ceux - ci ne sont pas pertinents ou trop complexes à interpréter (comme une interaction d'ordre trois ou quatre).

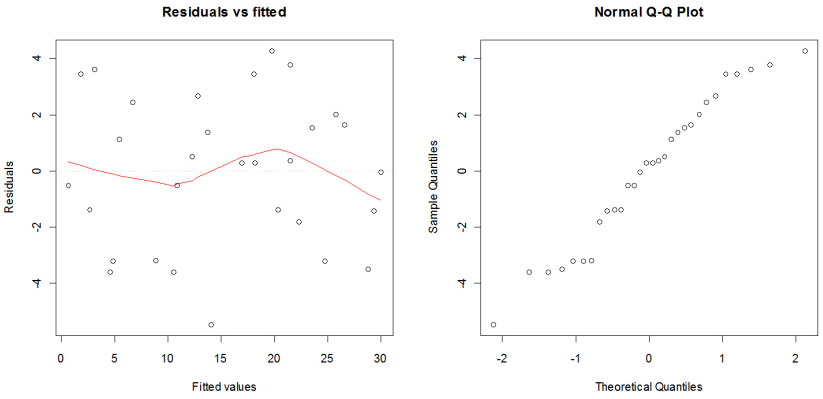
73. Vérification de la validité d'un modèle

Ecrire un modèle pour analyser des données est une chose, mais il est très important de vérifier que ce modèle s'ajuste bien aux données. Si ce n'est pas le cas, toute analyse découlant de ce modèle ne serait pas valide.

Les vérifications à effectuer se font essentiellement graphiquement et tournent globalement autour de trois points : la *variance*, l'*indépendance* et la *normalité* des résidus du modèle (les résidus étant les écarts entre les valeurs réellement observées et celles prédites par le modèle, nommées *fitted values*).

Ces trois vérifications essentielles sont réalisées grâce à la fonction `plotresid()` du package `RVAideMemoire`. Selon l'analyse, utiliser `plotresid(modele)` ou `plotresid(regression)`. Cette fonction renvoie deux graphes et le résultat d'un test statistique :

- le graphe de gauche sert à tester l'équivariance et l'indépendance des résidus. Sur ce graphe, la ligne rouge représente la tendance du nuage de points. Les hypothèses d'équivariance et d'indépendance sont acceptées lorsque cette ligne ne s'éloigne pas trop de l'horizontale. Plus précisément, l'hypothèse d'équivariance est acceptée lorsque la dispersion verticale des points est à peu près constante sur toute la longueur de l'axe des abscisses. L'hypothèse d'indépendance est acceptée lorsque l'orientation du nuage de points est horizontale
- le graphe de droite sert à tester la normalité des résidus. L'hypothèse de normalité est acceptée lorsque les points sont à peu près alignés sur une droite (voir fiche **38**)
- le test statistique réalisé est un test de Shapiro - Wilk appliqué aux résidus, qui teste lui aussi leur normalité (voir fiche **38**).



74. La méthode des contrastes

L'analyse de déviance en modèle linéaire généralisé (*GLM*) est, comme l'analyse de variance (*ANOVA*) ou le test du χ^2 d'homogénéité, un test global qui calcule l'effet d'un facteur sur une variable à expliquer. Pour cela il compare les valeurs de la variable à expliquer pour les différentes classes du facteur étudié. Comme tout test global, il indique si au moins deux classes du facteur donnent des valeurs de la variable à expliquer différentes, mais sans préciser lesquelles. Une *p - value* significative doit donc entraîner la réalisation de comparaisons deux - à - deux pour identifier les classes en question. C'est par la méthode des contrastes que sont réalisées ces comparaisons dans le cas d'un GLM ou d'un modèle linéaire mixte (LMM ou GLMM).

La procédure se fait en trois étapes :

1. créer la matrice des contrastes, *i.e.* la matrice des comparaisons à réaliser. Celle - ci est de la forme :

	Classe 1	Classe 2	Classe 3
Contraste 1	1	-1	0
Contraste 2	0	1	-1
Contraste 3	2	-1	-1

   Dans cette matrice, les comparaisons (ou contrastes) doivent être représentées en lignes, tandis que les classes du facteur sont en colonne. Les conventions d'écriture des contrastes sont les suivantes :
   - les classes n'intervenant pas dans la comparaison doivent avoir une valeur nulle
   - les classes à comparer doivent avoir une valeur non nulle et un signe opposé
   - il est possible d'effectuer des regroupements de classes
   - la somme des valeurs positives et négatives d'un contraste doit être nulle.

   Attention, **R** considère les classes du facteur dans l'ordre alphabétique, *i.e.* la 1ère colonne correspond à la 1ère classe dans l'ordre alphabétique, et ainsi de suite.

   Pour créer la matrice : `contrastes<-rbind(ligne1,ligne2,...)` où `ligne1` est un vecteur contenant les valeurs de la 1ère ligne, de gauche à droite (dans notre exemple `c(1,-1,0)`). Pour ne pas se tromper dans l'interprétation des comparaisons, on peut utiliser `colnames(contrastes)<-levels(facteur)` qui donne aux colonnes le nom de chaque classe du facteur, dans l'ordre alphabétique

2. créer un nouveau modèle ne contenant que le facteur étudié :
   - modèle linéaire mixte : `modele2<-lmer(variable~facteur-1+(1|aleatoire))` où `variable`, `facteur` et `aleatoire` dépendent du modèle initial

- GLM : `modele2<-glm(variable~facteur-1,family=loi)` où `loi` dépend du modèle initial
- GLM mixte : `modele2<-glmer(variable~facteur-1+(1|aleatoire),family=loi)`

Si c'est l'interaction entre deux facteurs qui est étudiée, créer d'abord un nouveau facteur : `interaction<-factor(paste(facteur1,facteur2,sep=":"))`. Puis créer le `modele2` en remplaçant `facteur` par `interaction`.

3. réaliser les comparaisons grâce à la fonction `adjust.esticon()` du package `RVAideMemoire` : `adjust.esticon(modele2,contrastes)`. La fonction renvoie un tableau avec une ligne par contraste (dans le même ordre que la matrice) et sur chaque ligne le résultat du test de comparaison des classes correspondantes.

Index des packages externes

Ces packages nécessitent d'être installés pour l'utilisation de certaines fonctions présentées dans cet ouvrage (voir fiche **7**) :

Bibliographie et ouvrages / documents / liens recommandés

Bates D. (2010) lme4 :Mixed-effects modeling with R. [en ligne : *http ://lme4.r-forge.r-project.org*]

Champely S. (2005) Introduction à l'analyse multivariée (factorielle) sous R. [pdf en ligne]

Crawley M.J. (2007) The R Book. Editions John Wiley & Sons, inc.

Dagnelie P. (2003) Principes d'expérimentation : planification des expériences et analyse de leurs résultats. Les presses agronomiques de Gembloux. [en ligne : *http ://www.dagnelie.be*]

Dagnelie P. (2006a) Statistique théorique et appliquée. 1. Statistique descriptive et base de l'inférence statistique. 2ème édition. Editions De Boeck.

Dagnelie P. (2006b) Statistique théorique et appliquée. 2. Inférence statistique à une et à deux dimensions. 2ème édition. Editions De Boeck.

Fox J. (2002) Cox Proportional-Hazards Regression fot Survival Data. [en ligne : *http ://cran.r-project.org/*, rubrique *Contributed* : *Web Appendix to the book "An R and S-PLUS Companion to Applied Regression"*]

Millot G. (2008) Comprendre et réaliser les tests statistiques à l'aide de R. Editions De Boeck.

Paradis E. (2002) R pour les débutants. [en ligne : *http ://cran.r-project.org/*, rubrique *Contributed*]

Poinsot D. (2004) Statistiques pour statophobes. [en ligne : *http ://perso.univ-rennes1.fr/denis.poinsot*]

Poinsot D. (2005) R pour les statophobes. [en ligne : *http ://perso.univ-rennes1.fr/denis.poinsot*]

R **Development Core Team** (2011). R : A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL *http ://www.R-project.org.*

Les nombreux cours en ligne de **D. Chessel**, **A.B. Dufour**, **J.R. Lobry** et **M. Royer** : *http ://pbil.univ-lyon1.fr/R/enseignement.html*

Les cours en ligne de **M.-L. Delignette-Muller** : *http ://www2.vet-lyon.fr/ens/biostat/accueil.html*

Le forum du groupe des utilisateurs du logiciel **R** : *http ://forums.cirad.fr/logiciel-R/index.php*

Semin-**R**, un autre groupe d'utilisateurs de **R** : *http ://rug.mnhn.fr/semin-r*