

Introduction à R

Florence Vallée-Dubois (cours POL 2809)

Table des matières

Bienvenue à votre introduction à R!	1
R Studio Cloud	2
Pour commencer, enregistrez votre éditeur R (<i>R script</i>).	3
Commandes de base	3
Les banques de données	5
Importer une banque de données	5
Explorer vos données	5
Exercices (I)	7
Statistiques descriptives (univariées et bivariées)	8
Régression linéaire	13
Exercices (II)	17
Pour vos projets futurs	17

Bienvenue à votre introduction à R!

R a vu le jour en 1993. Ce langage de programmation est de plus en plus utilisé en analyse statistique. Il permet, entre autres, de visualiser des données quantitatives et de faire des analyses statistiques sur ces données. Étant donné que R est un logiciel libre, de nombreux contributeurs et de nombreuses contributrices travaillent à la création de nouvelles fonctions, qui sont rendues disponibles à tous les utilisateurs de R. R permet des analyses simples et sophistiquées, allant de l'algèbre à l'intelligence artificielle.

Ces notes de cours sont adaptées au contexte du cours POL 2809, où l'apprentissage de R est restreint à la plate-forme R Studio Cloud. Certaines fonctions sont donc omises de ces notes de cours. Les personnes qui seraient intéressées à utiliser R à travers d'autres interfaces (comme R Studio tout simplement) peuvent consulter mon [site web](#) pour des notes d'atelier plus complètes.

Il existe de nombreuses ressources internet pour l'apprentissage de R.

Sites en français :

- Le site web d'[Alexandre Blanchet](#) offre une introduction très complète à l'inférence statistique avec R.
- Même si la plupart des forums d'aide sur R sont en anglais, il existe tout de même des blogs francophones qui permettent de démystifier certaines fonctionnalités. Entre autres, on retrouve [ElementR](#) et [R-atique](#).
- Le [Groupe des utilisateurs du logiciel R](#) est un forum francophone sur R.
- Le site [FUN Mooc](#) est une plateforme internet gratuite d'enseignement supérieur. On y offre le cours "Introduction à la statistique avec R".
- L'université Lyon 2 offre le cours "Programmation sous R". Le contenu du cours est disponible sur ce [site](#).

En anglais :

- Vous tomberez probablement assez vite sur [Stack Overflow](#) en cherchant des réponses concernant R sur Google. La communauté d'utilisateurs du langage R est très active sur ce site, qui est organisé sous forme de questions-réponses.
- Je consulte régulièrement le site [R-bloggers](#).

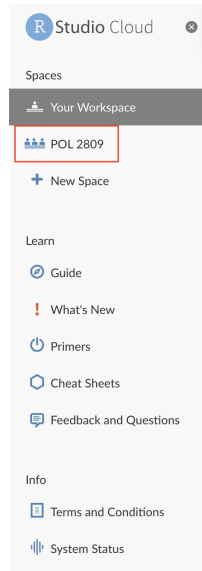



FIGURE 1 – L'espace du cours


- Pour suivre des formations en ligne, il existe le site [Coursera](#). Les cours ont le format de cours universitaires, mais avec la version gratuite : pas besoin de suivre l'entièreté des plans de cours (ni de remettre les travaux).

R Studio Cloud

R Studio Cloud est un environnement permettant de travailler avec le langage R de façon collaborative, ou de tenir des formations sur R. Quand vous ouvrez R Studio Cloud, l'espace du cours POL2809 se trouve à gauche de la page, sous "Spaces".

Dans l'espace de notre cours, vous trouverez différents "projets" : un nommé "Cours du 16 octobre", un nommé "Devoir 3" et un nommé "Travail final" (je vais les créer au fur et à mesure, donc il se peut que vous ne les voyiez pas tous encore). Cliquez sur le projet "Cours du 16 octobre" pour l'ouvrir.

Une fois entré.e dans le projet, vous verrez une interface avec 4 fenêtres (si elle en contient seulement 3, cliquez sur l'icône  en haut à droite de la fenêtre "console" pour ouvrir la 4e fenêtre).

Éditeur : C'est dans l'éditeur que vous écrirez toutes vos lignes de code. Vous pourrez ensuite enregistrer le contenu de votre éditeur en cliquant sur l'icône .

Console : C'est dans la console que vos lignes de codes s'effectueront. Je donne plus de détails plus bas. Mais en bref : c'est là que vous trouverez les résultats de vos analyses statistiques (tableaux de régression, etc.).

Dossiers, graphiques, visionneur, etc. Dans cette fenêtre, vous trouverez vos fichiers enregistrés (par exemple, vos codes enregistrés). J'y ai déjà enregistré pour vous les banques de données que nous utiliserons dans le cours ("CSES.csv" et "quality_governance.csv"). Ces notes de cours y sont aussi enregistrées.

Environnement : Pour utiliser vos banques de données, vous devrez d'abord les faire passer de vos dossiers à votre Environnement. Je vous montre comment faire plus bas.

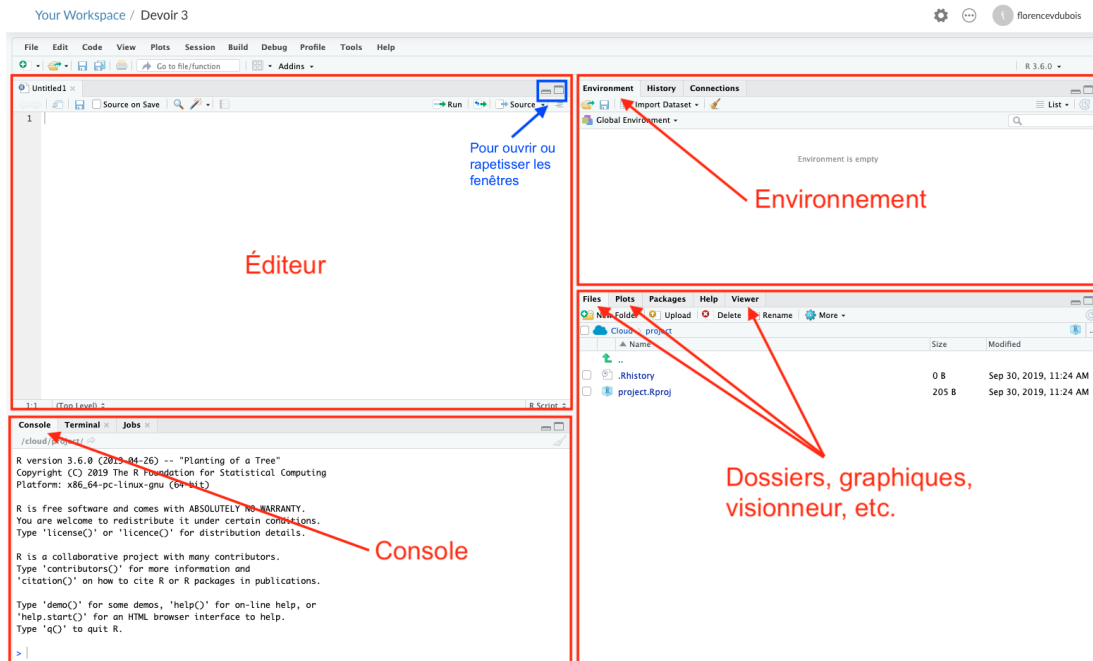


FIGURE 2 – L'interface R Studio Cloud

Pour commencer, enregistrez votre éditeur R (*R script*).

Avant d'aller plus loin, il est important d'enregistrer votre éditeur. Vous entrerez toutes vos commandes dans l'éditeur. Si vous l'enregistrez au fur et à mesure, vous pourrez retrouver votre travail la prochaine fois.


Commandes de base

Faire des commentaires/prendre des notes

À tout moment, vous pouvez prendre des notes pour vous-même dans votre éditeur. Les notes sont tout ce qui est précédé d'un ou plusieurs signes de dièse (#). Ces lignes ne seront pas "lues" par R au moment d'exécuter les commandes.

```
> # Ceci est une note -- elle ne sera pas exécutée par R. Annoter votre
> # éditeur vous permet de revenir à votre code plus tard, et de comprendre ce
> # que vous avez fait. Un code bien annoté vous fera gagner du temps! Il vous
> # permettra aussi de facilement partager vos analyses.
```

Exécution des commandes

Pour exécuter une commande, placez votre curseur sur la ligne que vous souhaitez exécuter, puis appuyez sur CMD + ENTER (en mac) ou sur CTRL + ENTER (en windows). Vous pouvez aussi sélectionner une (ou plusieurs) lignes et cliquer sur le bouton  (Run) en haut de votre éditeur.

Explorer une commande

Lorsque que vous n'êtes pas certain de savoir à quoi sert une commande ou comment elle doit être utilisée, la première étape est de l'explorer dans l'assistant R à l'aide du `?`.

```
> ?mean
```

Une fois la ligne de code exécutée, le résultat apparaîtra dans l'onglet *Help*.

Commandes mathématiques

On peut se servir de R comme d'une calculatrice. Par exemple, on peut faire des additions (+), des soustractions (-), des divisions (/) et des multiplications (*). De nombreuses autres opérations mathématiques peuvent y être réalisées.

```
> 4 + 2
## [1] 6
>
> 4 * 2
## [1] 8
>
> # Attention, R est 'anglophone'. Les virgules sont donc des points. pour
> # dire 2,4; on écrit 2.4
> 2.4 + 2
## [1] 4.4
>
> # Moyenne: je demande à R de calculer la moyenne des chiffres 4 à 10 (le
> # symbole ':' signifie 'à')
> mean(4:10)
## [1] 7
>
> # Médiane
> median(20:27)
## [1] 23.5
```

Attention, R est sensible aux majuscules!

```
> # Si j'écris 'Median', R me renvoie un code d'erreur.
> Median(20:27)
```

Assigner des valeurs à des objets

R est un langage "orienté objet" (*object oriented*). En R, on crée des "objets" et on leur assigne des données. Prenons un exemple simple : nous créons deux variables, nommées `allo` et `coucou`, et leur assignons à chacun une valeur. On assigne une valeur à un objet à l'aide du symbole `<-` ou `=`.

```
> allo <- 2
> coucou = 3
```

Vous observerez que les deux objets que nous venons de définir, `allo` et `coucou`, se trouvent maintenant dans notre Environnement! On peut s'en servir pour faire toutes sortes de manipulations.

```
> # Premièrement, en tapant le nom de l'objet, R nous renvoie son contenu.
> allo
## [1] 2
```

Voyez que la valeur que nous avons assignée à `allo` s'affiche dans la Console.

```
> # On peut ensuite faire des opérations mathématiques sur ces objets.
> allo - coucou
## [1] -1
>
> # On peut même créer un objet qui sera le résultat d'une opération.
> operation = allo * coucou
>
> # Et faire apparaître le résultat.
> operation
## [1] 6
```

Les banques de données

Grâce à R, il est possible de lire, de manipuler et de faire des analyses statistiques à partir de banques de données. C'est justement ce que nous voulons ! Une banque de données est une façon d'organiser l'information sous forme de rangées et de colonnes, comme dans un tableau Excel. Chaque colonne d'une banque de données correspond à une variable, tandis que chaque rangée correspond à une observation.

La Figure 3 montre un aperçu de la banque de données CSES, à laquelle vous avez accès pour le travail final. Chaque rangée correspond à un répondant au sondage, et chaque colonne est une variable. La première rangée correspond à quelqu'un né en 1960, qui vient de l'Autriche, vit dans un village, etc.

Importer une banque de données

Pour manipuler une banque de données, il faut d'abord l'importer dans notre environnement. Les banques de données que je vous ai fournies (et qui sont enregistrées dans vos fichiers sur **R Studio Cloud**) sont toutes deux en format `.csv`. Il s'agit d'un format fréquemment utilisé en science politique, mais ce n'est pas le seul – si vous continuez à utiliser R, il se peut que vous ayez besoin d'importer d'autres formats de données. Mais pour l'instant, on va s'en tenir au format `.csv`. Pour importer des données de ce format, on utilise la fonction `read.csv()`.

```
> cses = read.csv("CSES.csv", sep = ",")
```

N'oubliez pas de les importer dans un objet ! J'ai nommé mon objet `cses`, parce qu'il s'agit de la banque de données du *Comparative Study of Electoral Systems*. C'est donc plus intuitif. Un fois votre banque importée, elle se trouve dans votre Environnement. En cliquant dessus, vous pouvez l'explorer.

Importons aussi la banque de données de *Quality of Governance* :

```
> quality = read.csv("quality_governance.csv", sep = ",")
```

Explorer vos données

On peut explorer nos banques de données à l'aide de certaines fonctions.

```
> # Voir les premières rangées de la banque de données.
> head(cses)
>
> # Voir les dernières rangées.
> tail(quality)
```

date_naissance	pays	age	genre	education	statut_marital	syndicat	revenu	religion	immigration	region
1960	Austria	57	0	4	couple	0	3	1	0	village
1994	Austria	23	1	4	celibataire	0	2	0	0	village
1947	Austria	70	0	4	couple	0	2	1	0	banlieue
1955	Austria	62	1	4	couple	0	2	0	0	village
1943	Austria	74	0	4	couple	0	2	1	0	banlieue
1961	Austria	56	0	4	couple	0	4	1	1	banlieue
1962	Austria	55	1	4	couple	0	3	1	0	village
1967	Austria	50	0	4	couple	1	2	0	1	village
1939	Austria	78	1	3	couple	0	3	1	0	village
1967	Austria	50	1	4	couple	1	3	0	0	village
1978	Austria	39	1	4	couple	1	NA	1	1	banlieue

FIGURE 3 – Exemple de banque de données

La fonction `summary()` nous résume toutes les variables qui se trouvent dans notre banque de données.

```
> summary(cses)
> summary(quality)
```

Certaines variables sont de format *caractère* (des mots). Pour ces variables, la fonction `summary()` de R nous donne quelques exemples des différents choix possibles. Par exemple, dans la banque `cses`, la variable `pays` peut correspondre à : United States of America, Germany, Italy, Chile, etc. D'autres variables sont de format *numérique* (des chiffres). Pour ces variables, la fonction `summary()` de R nous donne quelques statistiques descriptives, comme la moyenne, la médiane, les valeurs minimales et maximales, etc. Par exemple, dans la banque `quality`, la variable `deplaces` va de 8 (min) à 3 655 000 (max). Sa moyenne est 120 998 (mean) et sa médiane est 2 900 (median).

Pour savoir comment sont codées chacune des variables des 2 banques de données, veuillez vous référer au document explicatif du travail final.

Explorer une variable

Dans R, le signe de dollar sert à aller chercher un élément qui se trouve dans un autre élément. Dans notre cas, nous avons des variables qui se trouvent *dans* une banque de données. On pourrait donc explorer les différentes valeurs d'une variable en se servant du signe de \$.

Par exemple :

```
> cses$region
```

Vous avez peut-être été surpris.e! C'est parce que R nous a montré TOUT ce qui se trouve dans la variable `region`. Pour avoir un résumé d'une variable de façon plus concise, il est possible de faire un tableau, à l'aide de la fonction `table()` :

```
> table(cses$region)
##
##    banlieue grande ville petite ville    village
##      1933      4802      3136      3563
```

Le tableau nous indique que nous avons 1933 répondants qui vivent en banlieue, 4802 répondants qui vivent dans une grande ville, 3146 répondants qui vivent dans une petite ville et 3563 répondants qui vivent dans un village.

Nous pourrions aussi vouloir faire un tableau croisé, avec 2 variables. Voici comment faire :

```
> table(cses$pays, cses$region)
##
##               banlieue grande ville petite ville village
##   Austria             164          279           123    637
##   Chile                54         1316           368    243
##   Germany             141          422           741    728
##   Greece              161          620           220     71
##   Hong Kong            0            0            0      0
##   Hungary              0          211           604    393
##   Ireland             214          337           105    331
##   Italy                0            0            0      0
##   Lithuania            0          614           384    502
##   Montenegro          185          271           318    438
##   Republic of Korea   510          554            0    135
##   Taiwan              0            0            0      0
##   United States of America 504          178           273     85
```

Le tableau croisé nous indique combien de répondants vivent dans les différentes catégories (banlieue, grande ville, petite ville, village) dans chaque pays de la banque de données.

Exercices (I)

1. Faites un tableau croisé avec les variables `att_elites` et `opinion_nation` de la banque de données du *Comparative Study of Electoral Systems*.
2. Comment est-elle codée la variable `pop_urbain` de la banque de données *Quality of Governance* ?
3. Combien y a-t-il d'observations dans chaque catégorie de la variable `region` de la banque de données *Quality of Governance* ?
4. Combien y a-t-il d'observations dans la banque de données du *Comparative Study of Electoral Systems* ? Que signifie le nombre d'observations dans ce cas-ci ?

Statistiques descriptives (univariées et bivariées)

Une fois votre banque de données explorée, vous pouvez vous lancer dans l'analyse statistique. Une première étape pertinente avant de se lancer dans l'analyse de régression est de faire des statistiques *descriptives*.

Pour décrire les variables dichotomiques ou catégoriques (ordinales et nominales), il est parfois plus logique d'utiliser les tableaux. Vous avez déjà appris à faire des tableaux et des tableaux croisés dans la section précédente. Or, pour les variables continues, il peut être utile de *calculer* des statistiques descriptives comme les mesures de tendance centrale (moyenne, médiane, etc.), de dispersion ou de position. Voici donc la marche à suivre.

Plus bas, je vous montrerai aussi à faire des visualisations graphiques. Elles peuvent être utiles pour tous les types de variables.

Mesures de tendance centrale

Vous trouverez ici comment calculer deux mesures de tendance centrale : 1. la moyenne ; 2. la médiane.

```
> # 1. Calcul de la moyenne: la moyenne se calcule avec la fonction mean()
> mean(cses$age)
## [1] NA
>
> # R renvoie la valeur 'NA' parce que la variable contient des valeurs
> # manquantes. Pour remédier à la situation, précisons que nous voulons la
> # moyenne, sans les valeurs manquantes.
> mean(cses$age, na.rm = TRUE)
## [1] 48.84813
>
> # La moyenne de l'âge est de 48.85 ans.
>
> # 2. Calcul de la médiane (en ignorant les valeurs manquantes): la médiane
> # se calcule avec la fonction median().
> median(cses$age, na.rm = TRUE)
## [1] 49
>
> # La médiane de l'âge est de 49 ans.
>
> # Pour les curieux... On peut calculer des statistiques sur des
> # sous-groupes de répondants. Par exemple: l'âge moyen des personnes qui
> # ont voté pour le parti sortant.
> mean(cses$age[cses$vote_sortant == 1], na.rm = TRUE)
## [1] 51.91759
>
> # Les personnes qui ont voté pour le parti sortant ont en moyenne 51.92 ans.
```

Mesures de position

Vous trouverez ici comment obtenir : 1. les quartiles ; 2. les déciles ; 3. les centiles d'une variable.

```
> # La fonction quantile() dans R permet de calculer les mesures de position.
> quantile(cses$age, na.rm = TRUE)
##    0%   25%   50%   75%  100%
##   16   35   49   62   100
```



```

>
> # Par défaut, la fonction renvoie les quartiles (chaque valeur qui divise à
> # 25%).
>
> # Pour calculer les déciles ou les centiles, il faut le spécifier dans un 2e
> # argument, qui suit la virgule.
> quantile(cses$age, c(0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1), na.rm = TRUE)
## 0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
## 16 25 32 38 44 49 54 59 65 72 100
>
> # Et ainsi de suite.

```

Mesures de dispersion

Vous trouverez ici comment calculer deux mesures de dispersion : 1. la variance; 2. l'écart-type.

```

> # 1. Calcul de la variance d'une variable: la variance se calcule avec la
> # fonction var().
> var(cses$age, na.rm = TRUE)
## [1] 294.4615
>
> # 2. Calcul de l'écart type (en ignorant les valeurs manquantes):
> # l'écart-type se calcule avec la fonction sd().
> sd(cses$age, na.rm = TRUE)
## [1] 17.15988
>
> # L'écart-type associé à l'âge est de 17,16 années.
>
> # On peut même vérifier que l'écart-type est bel et bien la racine carré de
> # la variance, à l'aide de la fonction sqrt().
> sqrt(var(cses$age, na.rm = TRUE))
## [1] 17.15988
>
> # La racine carrée de la variance est égale à l'écart-type, soit 17,16
> # années.

```

Visualisation graphique

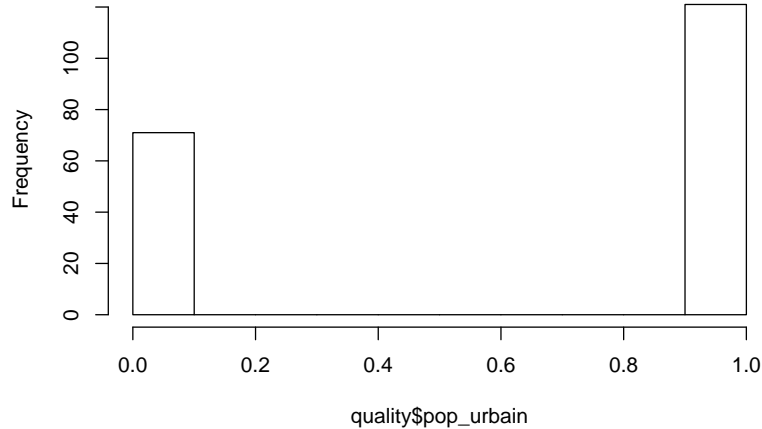
Dans cette section, j'explique comment visualiser graphiquement des variables dichotomiques, catégoriques nominales ou ordinales, et continues. Commençons par les distributions discrètes (pour les variables 1. dichotomiques, 2. ordinales et 3. nominales).

```

> # La fonction hist() dans R permet de tracer un histogramme.
>
> # Pour une variable dichotomique:
> hist(quality$pop_urbain)

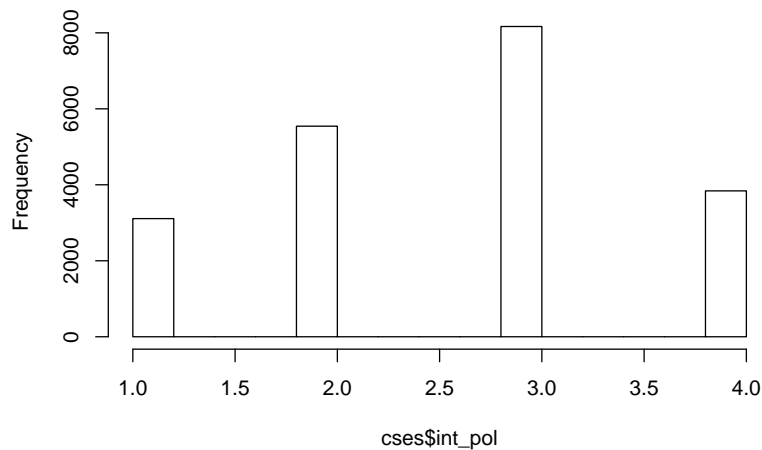
```

Histogram of quality\$pop_urbain

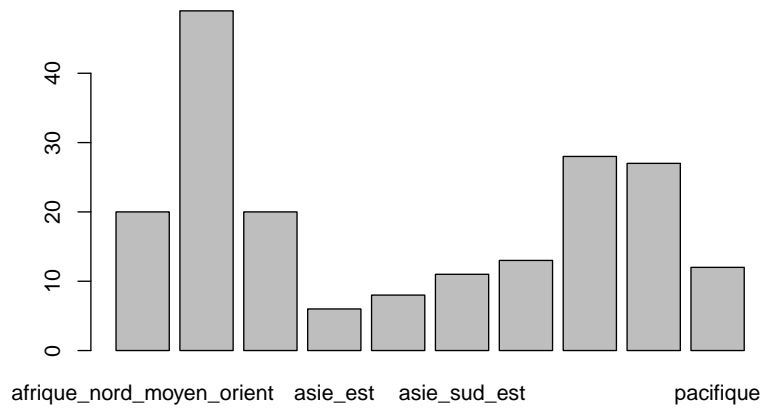


```
>
> # L'histogramme nous montre que la variable 'pop_urbain' peut seulement
> # prendre deux valeurs (0 ou 1). Un peu plus de 70 observations sont codées
> # 0, tandis qu'environ 120 observations sont codées '1'.
>
> # 2. Pour une variable ordinaire, c'est le même principe. Référez-vous
> # toujours au document explicatif des banques de données pour savoir comment
> # chaque variable est codée.
> hist(cses$int_pol)
```

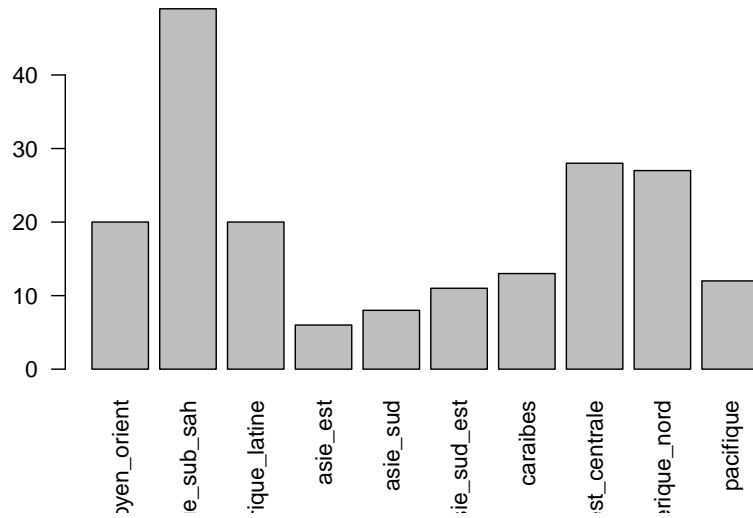
Histogram of cses\$int_pol



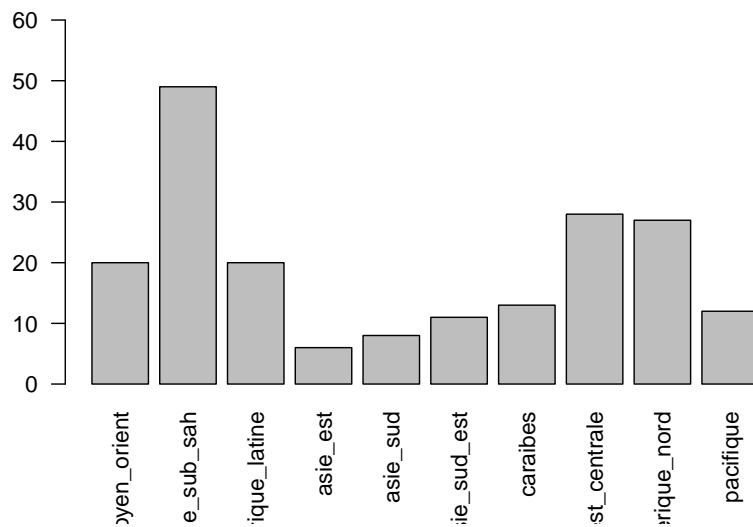
```
>
> # 3. Pour une variable nominale, c'est un peu plus compliqué. La fonction
> # hist() ne permet pas de visualiser des variables nominales, puisqu'elles
> # sont codées avec du texte. Une façon alternative est d'utiliser la
> # fonction barplot(). Dans cette fonction, on précise qu'on veut visualiser
> # le 'tableau' de la variable qui nous intéresse.
> barplot(table(quality$region))
```



```
>
> # On ne voit pas toutes les catégories! Mettons-les à la verticale:
> barplot(table(quality$region), las = 2)
```



```
>
> # L'axe Y est trop petit à mon goût. On peut l'agrandir comme suit:
> barplot(table(quality$region), las = 2, ylim = c(0, 60))
```

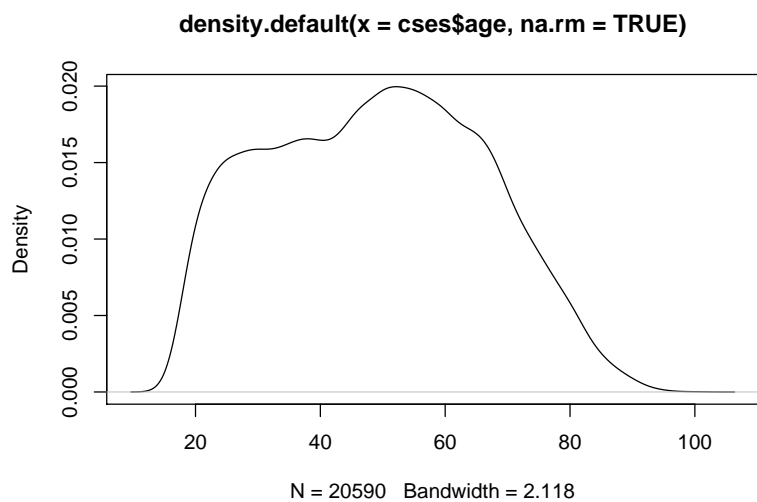


Ensuite, voyons comment simplement visualiser les variables continues.

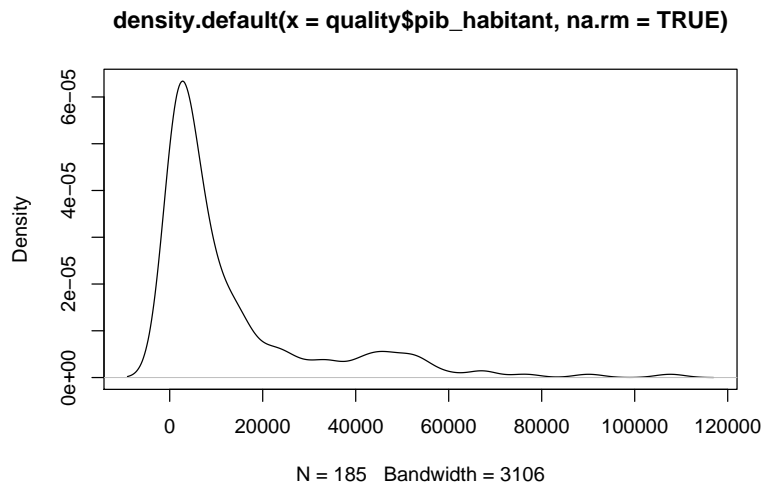
```
> # Évidemment, on pourrait encore faire un histogramme.  
> hist(cses$age)
```



```
>  
> # Or, une variable continue a une distribution continue. Comment faire?  
> plot(density(cses$age, na.rm = TRUE))
```



```
> plot(density(quality$pib_habitant, na.rm = TRUE))
```



Analyses bivariées

R peut servir à analyser la relation entre deux variables. De nombreux tests, comme la corrélation de Pearson, sont disponibles sur R.

```
> # la fonction cor() permet de calculer la corrélation de Pearson
> cor(quality$invest_direct, quality$piib_habitant, use = "complete.obs")
## [1] 0.2146809
>
> # La corrélation entre le pourcentage du PIB qui provient d'investissements
> # directs et le PIB par habitant est de 0.21. On a utilisé l'argument 'use =
> # 'complete.obs'' pour que la fonction ignore les cas manquants.
```

Régression linéaire

Dans le cours, nous avons passé beaucoup de temps à comprendre la méthode de la régression linéaire par les moindres carrés. À partir de cette méthode, nous avons vu comment des variables indépendantes (X) pouvaient nous aider à prédire un phénomène (Y). On a aussi vérifié si les résultats d'une analyse de régression pouvaient être généralisés à la population (signification statistique). Dans R, les analyses de régression linéaire s'effectuent grâce à la fonction `lm()`. Consultons le fichier d'aide de cette fonction :

```
> ?lm
```

La fonction `lm()` a deux arguments essentiels. Premièrement, il faut lui fournir la formule. Deuxièmement, il faut lui indiquer quelles données utiliser. Voyons un exemple :

Est-ce que le nombre d'espèces animales en danger est associé au pourcentage de terre arable dans un pays?

On pourrait poser l'hypothèse selon laquelle il y a plus d'espèces animales en danger dans les pays où une plus grande proportion du territoire est utilisée pour les terres arables.

```
> # Estimation du modèle
> modele <- lm(especes_danger ~ terre_arable, data = quality)
```

J'enregistre mon modèle dans un objet nommé `modele`. La variable dépendante (`especes_danger`) se trouve en premier. Le symbole `~` est l'équivalent du symbole "=" dans notre équation de régression. Les variables indépendantes se trouvent à droite de `~`. Puis, on précise la banque de données d'où proviennent ces variables. Dans notre cas, il s'agit de la banque de données de *Quality of Governance*, enregistrée dans l'objet `quality`.

Pour voir les résultats de la régression, on peut simplement écrire le nom de l'objet. Dans notre cas, c'est l'objet `modele`.

```
> # Montrer les résultats dans la console
> modele
##
## Call:
## lm(formula = especes_danger ~ terre_arable, data = quality)
##
## Coefficients:
## (Intercept)  terre_arable
##      177.123      -0.577
```

Les résultats qui s'affichent dans la console sont très limités. Il s'agit seulement des valeurs pour l'intercept et pour le coefficient de la variable `terre_arable`. Or, on ne voit pas toutes les statistiques associées aux estimés. Pour obtenir les résultats complets, il faut utiliser la fonction `summary()`.

```
> summary(modele)
##
## Call:
## lm(formula = especes_danger ~ terre_arable, data = quality)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -174.19  -111.74   -55.64    41.96   824.34
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  177.1232    18.9529   9.345  <2e-16 ***
## terre_arable  -0.5770     0.9435  -0.612    0.542
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 176.2 on 186 degrees of freedom
## (6 observations deleted due to missingness)
## Multiple R-squared:  0.002007, Adjusted R-squared:  -0.003359
## F-statistic: 0.374 on 1 and 186 DF, p-value: 0.5416
```

Nous nous intéressons surtout au petit tableau qui suit `Coefficients`. Le tableau a deux rangées, soit l'intercept et la variable `terre_arable`. Il a quatre colonnes :

- `Estimate` : estimés
- `Std. Error` : erreur type
- `t value` : statistique t
- `Pr(>|t|)` : valeur p

Portez votre regard sur la colonne `Estimate`. Le coefficient associé à la variable `terre_arable` est égal à -0.5770. Cela signifie que pour une augmentation d'un point de pourcentage dans la superficie du territoire dédiée aux terres arables, le nombre d'espèces animales en danger diminue de 0,58. L'effet est contraire à mon attente. Qui plus est, le coefficient n'est pas statistiquement significatif. La statistique t est inférieure à 2 en valeur absolue (0,612) et la valeur p associée à cette statistique t est plus grande que 0,05. Elle est égale à 0,542. Les étoiles nous indiquent le niveau de signification statistique. Le niveau généralement accepté en science politique est de 0,05, ce qui correspond à une étoile.

Traitement des variables dichotomiques et nominales

Nous aurions aussi pu essayer de prédire le nombre d'espèces en danger à l'aide d'une variable nominale, comme la région. Une hypothèse (assez descriptive) pourrait être qu'il y a plus d'espèces en voie de disparition dans certaines régions du monde que dans d'autres. Dans le document explicatif sur le travail final, je vous ai indiqué quel est le type de chaque variable. Vous verrez que contrairement aux variables continues, dichotomiques ou ordinales, les variables nominales contiennent des mots, et non des chiffres.

Lorsqu'on utilise ce type de variable dans la fonction `lm`, R ignore automatiquement une catégorie, qui devient notre catégorie de référence. Par exemple :

```
> # Estimation du modèle
> mod.nominal <- lm(especes_danger ~ region, data = quality)
> summary(mod.nominal)
##
## Call:
## lm(formula = especes_danger ~ region, data = quality)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -308.98  -87.12  -32.84   29.55  826.83
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)      133.30      37.68   3.538  0.00051 ***
## regionafrique_sub_sah      38.63      44.71   0.864  0.38866
## regionamerique_latine     124.83      53.28   2.343  0.02021 *
## regionasie_est           33.88      78.43   0.432  0.66624
## regionasie_sud           18.08      70.49   0.257  0.79781
## regionasie_sud_est      176.93      63.25   2.797  0.00570 **
## regioncaraibes        -37.92      60.03  -0.632  0.52842
## regioneurope_est_centrale -42.41      49.33  -0.860  0.39110
## regioneurope_ouest_amerique_nord  37.87      49.71   0.762  0.44716
## regionpacifique         40.53      61.53   0.659  0.51085
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 168.5 on 184 degrees of freedom
## Multiple R-squared:  0.1103, Adjusted R-squared:  0.06676
## F-statistic: 2.534 on 9 and 184 DF, p-value: 0.009191
```

En vérifiant la description de la variable `region`, on peut voir que la catégorie de référence est l'Afrique du Nord et le Moyen Orient (codée `afrique_nord_moyen_orient`). C'est la seule catégorie qui n'apparaît pas dans les résultats. On peut donc dire que par rapport aux pays de l'Afrique du Nord et le Moyen Orient, il y a en moyenne 176,93 espèces de plus en voie de disparition dans les pays de l'Asie du Sud-Est (stat. significatif).

Pour les variables dichotomiques, on peut interpréter le coefficient montré dans le tableau comme l'effet d'une augmentation de 1 unité dans la variable dichotomique ou, en d'autres mots, comme l'effet de passer de 0 à 1 sur cette variable.

```
> # Estimation du modèle
> mod.dicho <- lm(especes_danger ~ pop_rural, data = quality)
> summary(mod.dicho)
##
## Call:
## lm(formula = especes_danger ~ pop_rural, data = quality)
```

```
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -166.10 -108.22  -58.97   48.38  832.03
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  165.965     15.929   10.419  <2e-16 ***
## pop_rural      1.201      26.194    0.046    0.963
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 175.2 on 190 degrees of freedom
## (2 observations deleted due to missingness)
## Multiple R-squared:  1.107e-05, Adjusted R-squared:  -0.005252
## F-statistic: 0.002103 on 1 and 190 DF,  p-value: 0.9635
```

Passer de 0 à 1 sur la variable `pop_rural` est associé avec une augmentation de 1,2 espèces en voie de disparition. En comparaison avec les pays où moins de 50% de la population vit en milieu rural (0), il y a en moyenne 1,2 espèces de plus en voie de disparition dans les pays où plus de 50% de la population vit en milieu rural (1).

Régression multiple

Pour réaliser des analyses de régression multiple, on ajoute des variables X à l'aide du symbole `+`. C'est exactement comme dans notre équation de régression, où les variables indépendantes sont additionnées!

```
> modmultiple = lm(especes_danger ~ terre_arable + region, data = quality)
> summary(modmultiple)
##
## Call:
## lm(formula = especes_danger ~ terre_arable + region, data = quality)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -310.29  -88.43  -30.91   30.10  819.65
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    134.3357    38.9341   3.450  0.0007 ***
## terre_arable    -0.1257     0.9942  -0.126  0.8996
## regionafrique_sub_sah    41.9184    45.7508   0.916  0.3608
## regionamerique_latine   125.3174    53.9663   2.322  0.0214 *
## regionasie_est     12.1818    93.2471   0.131  0.8962
## regionasie_sud      20.4211    73.5704   0.278  0.7817
## regionasie_sud_est   177.5762    64.0994   2.770  0.0062 **
## regioncaraibes     -37.8763    60.6433  -0.625  0.5331
## regioneurope_est_centrale -40.5802    51.8890  -0.782  0.4352
## regioneurope_ouest_amerique_nord  44.7719    51.4616   0.870  0.3855
## regionpacifique      57.1993    65.9683   0.867  0.3871
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```



```
## Residual standard error: 170.2 on 177 degrees of freedom
## (6 observations deleted due to missingness)
## Multiple R-squared: 0.1137, Adjusted R-squared: 0.06364
## F-statistic: 2.271 on 10 and 177 DF, p-value: 0.01593
```

Après avoir contrôlé pour la région, l'effet du pourcentage de terres arables sur le nombre d'espèces en voie de disparition est de -0,13. Une augmentation d'un point de pourcentage dans la superficie du territoire dédiée aux terres arables est associée avec une diminution de 0,13 espèces animales en voie de disparition. Comme précédemment, le coefficient n'est pas statistiquement significatif.

Exercices (II)

1. Quelle est la médiane de la variable `date_naissance` dans la banque de données du CSES? Trouvez l'information de DEUX façons différentes.
2. Faites un histogramme de la variable `emploi_agri` de *Quality of Governance*. Combien de pays ont entre 10 et 20% de leur population qui travaillent dans le secteur agricole (approximativement)?
3. Quelle est la corrélation (Pearson) entre le genre et le niveau d'éducation des répondant.e.s du CSES? Qu'est-ce que signifie cette corrélation, en mots?
4. Réalisez une régression linéaire ayant comme variable dépendante le revenu et comme variable indépendante la date de naissance (données CSES).
5. À l'aide des données de *Quality of Governance*, composez un modèle de régression linéaire avec une variable dépendante continue au moins deux variables explicatives. Interprétez les coefficients ainsi que les niveaux de signification statistique.

Pour vos projets futurs

Cette introduction à R a effectué un survol des principales fonctionnalités de ce langage, très utile en analyse statistique. Il vous aura appris, je l'espère, à utiliser les fonctions de base en R, à visualiser vos données et à faire des analyses univariées, bivariées ainsi que des régressions linéaires multiples. Vous pouvez maintenant utiliser R pour mener d'autres types d'analyses statistiques, ou créer de plus beaux graphiques (La fonction `ggplot()` est très puissante pour faire des graphiques! Je ne l'ai pas montrée ici parce qu'il s'agit d'une introduction débutante). Je vous invite à explorer les sources citées plus haut si vous avez des questions en cours de cheminement. R est un outil très flexible, vous serez surpris.e des possibilités qui s'ouvrent à vous en termes de modélisation et de visualisation de données!

Note : Quand vous faites des analyses dans R, il est possible que les résultats numériques s'affichent avec des exposants. Si c'est le cas, il sera écrit `e-` (exposant négatif) ou `e+` (exposant positif) à la suite du chiffre. Par exemple, `1.184e-02` signifie 1,184 exposant -2, ou 0,0184 (quand l'exposant est négatif, on tasse la virgule à gauche). À l'inverse, `1.184e+02` signifie 1,184 exposant 2, ou 118,4 (quand l'exposant est positif, on tasse la virgule à droite).