

# La régression logistique en épidémiologie

## Codes R

### Sommaire

Préambule.....	4
Commandes préalables .....	4
Note sur les packages .....	5
Fonctions .....	7
Fonction logit_crb .....	7
Fonction ORcl_pf .....	8
Fonction ORcl_sp.....	9
Chapitre 1 .....	10
Chapitre 2 .....	11
II. Estimation des paramètres de la régression logistique et de leur IC (tableaux 1 à 3) .....	11
Tableau 1 .....	11
Tableaux 2 et 3 .....	11
IV. Test des paramètres du modèle logistique .....	14
Test du rapport de vraisemblances (Tableau 4) .....	14
Test du khi2 (Tableau 5) .....	14
Test de wald .....	15
Test simultané de plusieurs coefficients .....	16
Chapitre 3 .....	18
I. Règle générale d'interprétation du coefficient d'une variable (tableau 1) .....	18
II. Variable dichotomique .....	18
Changement de codage de salp (Tableau 2).....	18
III. Variable qualitative nominale à plus de 2 classes .....	19
III.2. Interprétation des coefficients d'une variable indicatrice.....	19
III.3 OR entre les catégories d'une variable indicatrice .....	20
III.3.b Commande lincom (Tableau 8).....	21
III.4. Test de l'association entre Y et X décomposée en variables indicatrices.....	21
IV. Variable qualitative ordinale .....	22
Tableau 12 .....	22
IV.1 Modélisation de l'association GEU - tabac .....	22
IV.2. Choix des valeurs de X.....	25
IV.3. Choix entre variables indicatrices et modélisation linéaire (test de linéarité) .....	27
Test de linéarité (tableau 15) .....	27

VI. Prise en compte d'une interaction	28
VI.1. Interaction entre variables qualitatives	28
VI.2. Interaction avec F et analyses séparées selon les niveaux de F	31
<b>Chapitre 4</b>	<b>33</b>
IV. Courbes lowess (figure 8)	33
V - Figure 9 : Représentation des Logit P observés	35
VI. Modélisation avec une fonction en escalier	36
Modèle linéaire (figure 11)	36
Coefficients des modèles A et B	37
Comparaison des modèles 1 et A, et 1 et B pour voir qu'il n'y a pas de message d'erreur	39
Comparaison des modèles A et B	39
Tableau 2	39
Fonctions en escalier (Figure 12)	39
Tableau 3 et Figure 13	40
VII. Modélisation avec des polynômes (figure 14)	42
VIII. Modélisation avec des polynômes fractionnaires	45
Polynôme fractionnaire de degré 1 (figure 17)	45
Polynôme fractionnaire de degré 2 (Figure 18)	46
Polynôme fractionnaire de degré 4	48
Modélisation simultanée de plusieurs variables quantitatives, procédure mfp (Tableau 6)	48
IX. Modélisation avec des fonctions splines	50
IX.4 Splines linéaires	50
IX.5 Splines cubiques (figure 27)	56
IX.6 Splines cubiques restreints à 3 noeuds (figure 28)	57
Stratégie de choix du modèle avec des fonctions splines	59
X.3. Présentation quantitative des résultats dans un tableau (tableau 11)	59
OR non modélisés (colonne 1)	59
Avec un polynôme fractionnaire de degré 2 (colonne 2) : fonction ORcl.fp	59
Avec une fonction splines cubiques restreints à 3 noeuds (colonne 3) : fonction ORcl_sp	60
<b>Chapitre 5</b>	<b>61</b>
IV.3. Colinéarité	61
V.3 Sélection fondée sur le changement de l'estimation de l'odds-ratio (module chest)	62
Modèle complet (tableau 1)	62
Chest (tableau 2)	62
V.4. Méthodes de sélection pas à pas (stepwise)	64
Tableau 4	64
Tableau 5	65
<b>Chapitre 6</b>	<b>67</b>
Régression logistique multinomiale	67
II.2.Exemple et interprétation des résultats avec une seule covariable	67
§ II.3. Régression logistique multinomiale versus plusieurs régressions binomiales (plusieurs covariables)	69
II.4. Comparaison des OR associés à X selon les catégories de Y	72
II.5. Changement de classe de référence pour Y (Tableau 7)	73
Régression logistique ordinale	74

IV. Modèle cumulative-odds	74
Tableaux 8 et 9	75
IV.4. Regroupement de classes (Tableau 10)	76
IV.4. Test de l'hypothèse des odds proportionnels (Tableau 11)	77
IV.5. Présentation des résultats (Tableau 12)	78
Présentation des résultats (Figure 1)	79
Plusieurs variables indépendantes $X_i$ (Tableau 13)	80
Modèle cumulative odds avec l'hypothèse des odds proportionnels sauf pour la variable mrc (Tableau 14)	82
§ V. Modèle continuation-ratio	83
Modèle continuation-ratio avec odds proportionnels (Tableau 16)	84
§ V.3. Modélisation de la durée d'infécondité	86
VI. Modèle adjacent-category	89
Tableau 22	89
Tableau 23	90
Tableaux 24 et 25	91
VII.2. Un peu d'humilité sur l'importance du choix ...	93
Tableau 26	93
Chapitre 7	96
III. Tests d'adéquation	96
Test du $\chi^2$ de Pearson et de la déviance	96
Tableau 2	96
Tableau 3 : Profils	97
Tableau 4 : test de Hosmer-Lemeshow	97
IV. Courbe ROC	99
IV.1 Tableaux 5 et 6	99
Figure 1 : Sensibilité et spécificité du classement par le modèle logistique selon le seuil choisi	100
IV.2. Aire sous la courbe ROC	101
V. Diagnostics de régression	102
Construction de la table des résidus	102
Figures 4 à 6	103
Tableau 7 : Caractéristiques des profils tels que $\Delta \chi^2$ ou $\Delta D^2$ est supérieur à 4	105
Tableau 8 : Variations des coefficients pour les profils les plus influents	106

# Préambule

Ce document donne les codes R ainsi que les résultats correspondant à ce qui est donné en Stata au sein de chaque chapitre du livre "La régression logistique en épidémiologie" édité par EDP Sciences.. Il doit beaucoup à Guillemette Antoni, ingénieure de recherche à l'Inserm qui y a beaucoup contribué. Je suis cependant responsable du résultat final, notamment des erreurs qu'il pourrait contenir, et en tout cas des imperfections de la programmation R.

Il est écrit en langage R Markdown qui permet d'avoir des sorties plus lisibles. Les scripts R sont dans les zones grisées. Les résultats sont dans les lignes précédées de `##`. Le reste est des commentaires.

Vous trouverez une brève introduction à Markdown à l'adresse <https://lms.fun-mooc.fr/c4x/UPSUD/42001S02/asset/RMarkdown.html> qui vous donne des liens vers d'autres pages web. Il y a aussi beaucoup d'autres informations sur Markdown ailleurs sur internet.

Les bases de données nécessaires sont disponibles à l'adresse <https://laboutique.edpsciences.fr/produit/1504/9782759838189/la-regression-logistique-en-epidemiologie> indiquée dans le chapitre 1 du livre. Elles existent selon différents formats. Le plus pratique pour une exploitation à l'aide de R est le format `.RData`. Téléchargez `geu.rdata`, et enregistrez-le sur votre poste de travail, par exemple dans un directory `.../EQ/Regression Logistique/Codes R/data/`. Puis ouvrir une session de RStudio.

## Commandes préalables

Pour chaque chapitre, les codes de commandes eux-mêmes doivent être précédés d'une série de commande pour fixer l'environnement de travail. Ce sont les suivantes.

Nettoyage de l'environnement R :

```
rm(list=ls())
```

Choix du directory de travail (pour éviter d'avoir à le répéter)

```
setwd("/Mon directory")
```

Importation du fichier de données

S'il est au format `Rdata` :

```
load("/fic.rdata")
```

Alternativement, s'il est au format `.csv`,

```
geu <- read.csv("/geu.csv")
```

Fonctions

Plusieurs fonctions "fabriquées" sont utilisées dans les codes R des chapitres 4 et suivants (`logit_crb`, `ORcl_pf` et `ORcl_sp`). Elles sont utiles pour tracer des courbes et/ou faire des calculs que je n'ai pas trouvés dans R (ni d'ailleurs dans Stata). Elles sont décrites plus loin. Il faut les placer dans un directory spécifique et signaler à R à quel endroit (directory) les trouver par 3 commandes du type :

```
source("/directory/fonction logit_crb.R")
```

Packages (voir aussi "note sur les packages" plus bas)

Il s'agit de l'ensemble des packages utiles pour les différents chapitres qu'il faut charger par les commandes :

```
library(epiDisplay)
```

```
library(aod)
library(biostat3)
library(car)
library(lsplines)
library(Hmisc)
library(splines)
library(mfp)
library(tidyverse)
library(magrittr)
library(questionr)
library(chest)
library(nnet)
library(VGAM)
library(vcdExtra)
library(plotROC)
library(epiR)
library(reshape)
library(gmodels)
library(gtsummary)
```

## Note sur les packages

Lorsque vous procédez à l'installation de R, plusieurs packages sont installés en même temps sur votre poste de travail. Un package est un ensemble de fonctions rassemblées dans un même "paquet" (par exemple, le package {survival} rassemble un nombre important de fonctions permettant d'analyser des données de survie).

Il existe en outre une multitude de packages plus spécifiques et souvent très utiles, qui ne sont pas installés automatiquement, mais qui sont disponibles sur le site du *Comprehensive R Archive Network* (CRAN).

Il faut procéder à leur installation, sur le disque dur de l'ordinateur. Cette installation se fait très facilement dans RStudio avec un clic ou depuis la console avec la commande suivante :

```
install.packages("epiDisplay")
```

Cette opération d'installation n'a besoin d'être réalisée qu'une seule fois pour un ordinateur donné. Cependant, vous pourrez être amenés à procéder à une réinstallation dans les cas suivants : vous avez connaissance d'une mise à jour intéressante du package; vous avez mis à jour le logiciel R (dans ce cas, la réinstallation de tous les packages additionnels dont vous avez besoin est obligatoire). En effet, lorsque R installe les packages via la commande `install.packages()`, l'installation se fait dans un directory spécifique à la version en cours de R. Au moment de la mise à jour de R, il va créer un nouveau dossier, dans lequel on ne trouve que les packages basiques.

Pour connaître le directory dans lequel R installe tous les packages la commande est : `.libPaths()`

Ensuite, à *chaque nouvelle session de R*, il faut charger le package à l'aide de la fonction `library(nom du package)`.

### Ordre des packages

L'ordre d'installation des packages joue un rôle car certaines commandes ont le même nom dans plusieurs packages, mais pas exactement la même fonction ou ne s'appliquent pas aux mêmes objets. Par exemple la commande `lrtest` du package `epiDisplay` (utilisée pour comparer les vraisemblances de 2 modèles logistiques) existe aussi dans le package `VGAM` où elle ne s'applique pas aux modèles `glm`. Comme `VGAM` est chargé après, le `lrtest` de `VGAM` "masque" celui de `epiDisplay` et c'est lui qui est appelé quand on tape `lrtest`.

Ce masquage est mentionné quand on fait le chargement, mais on n'y fait souvent pas attention. On se rend compte du problème avec les messages d'erreur lorsque on veut utiliser le `lrtest` de `epiDisplay`. Pour le résoudre, il faut dire explicitement que c'est la commande `lrtest` de `epiDisplay` qu'on veut et écrire `epiDisplay::lrtest(mod0, mod1)`

Voici la liste des packages utilisés, dans l'ordre, assez arbitraire, où ils sont chargés et la raison pour laquelle je les ai utilisés.

- `epiDisplay`  
pour sa fonction `lrtest()` (rapport de vraisemblance) et également sa fonction `logistic.display()` pour afficher les OR et leurs intervalles de confiance, à partir des coefficients bêta d'une régression logistique estimés par la fonction `glm()`.
- `aod`  
pour la fonction `wald.test()` permettant de réaliser un test de Wald global
- `biostat3`  
pour la fonction `lincom()` permettant de tester des combinaisons linéaires de paramètres d'une régression.
- `car`  
pour sa fonction `linearHypothesis()` permettant de tester des hypothèses un peu "élaborées" sur les coefficients estimés par la régression; et pour sa fonction `recode()` (anecdotique, mais pratique pour la création d'une nouvelle variable).
- `lspline`  
fonctions splines linéaires
- `Hmisc`  
pour les fonctions splines cubiques restreints ayant X pour première fonction spline (un autre package "splines" est déjà installé en même temps que R, mais il fait d'autres fonctions splines cubiques restreints)
- `splines`
- `mfp`  
polynômes fractionnaires
- `tidyverse`  
package très riche (qui rassemble en fait plusieurs packages), avec notamment beaucoup de fonctions qui facilitent le data management, les graphiques, les opérations en boucle...
- `magrittr`  
pour l'écriture des commandes avec le pipe `%>%`. Pour le principe de pipe, voir par exemple ["http://perso.ens-lyon.fr/lise.vaudor/utiliser-des-pipes-pour-enchaîner-des-instructions/"](http://perso.ens-lyon.fr/lise.vaudor/utiliser-des-pipes-pour-enchaîner-des-instructions/)
- `questionr`  
Pour la fonction `odds.ratio`
- `chest`  
Ce package contient différentes fonctions utiles pour la sélection des variables fondée sur le changement de l'estimation de l'odds-ratio (voir chapitre 5)
- `nnet`  
Package permettant de réaliser des régressions multinomiales
- `VGAM`  
Package pour les modèles linéaires ou additifs généralisés. Dont le modèle multinomial et les modèles ordinaux
- `vcdExtra`  
Pour le test d'adéquation d'Hosmer-Lemeshow

- plotROC

Pour les courbes ROC

- epiR

Package donnant de nombreux résultats utiles en épidémiologie. Je l'utilise ici pour les statistiques d'adéquation

- reshape

Package très général pour manipuler les fichiers de données. Utilisée ici pour un graphique du chapitre 6

- gmodels

Pour la fonction CrossTable qui permet une présentation agréable de la distribution croisée de 2 variables qualitatives (attention cependant qu'elle ne donne pas le khi-2 global mais ces composantes par case du tableau).

- gtsummary

Package utile pour la présentation des données et des résultats

## Fonctions

De façon générale, on utilise des fonctions pour exécuter des tâches répétitives. Une fonction est un programme R avec, en général, des arguments (paramètres). Il faut la charger avec la commande "source" : `source("/fonction.R")`

Dans ce livre, il s'agit de réaliser les courbes, ou de calculer des OR après modélisation d'une variable quantitative comme cela est présenté notamment dans le chapitre 4.

3 fonctions ont été programmées :

### Fonction `logit_crb`

Cette fonction permet de construire des courbes de modélisation des variables quantitatives montrées dans le chapitre 4 du livre. Son programme figure dans le fichier `fonction_logit_crb.R`.

Les arguments de la fonction sont :

`logit_crb(data, mod, reponse, x, x_class=NULL, lab.x, mod, observ=T, courbe=T, logit_min=-6, logit_max=2, title=NULL, c1="black", c2="brown3")`

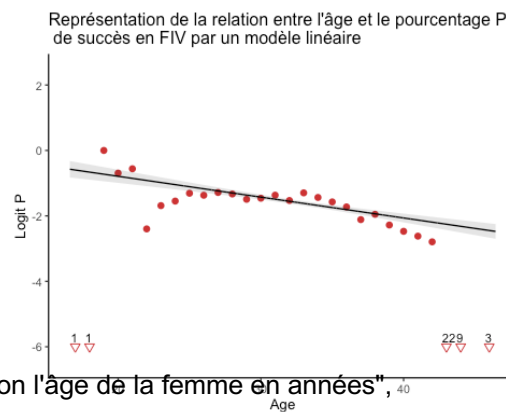
Pour les arguments suivis de =, la valeur qui suit est la valeur par défaut

- data : fichier de données (celui sur lequel le modèle désigné par "mod" a été estimé)
- mod : modèle logistique pour lequel on veut la courbe de relation entre X et Y
- reponse : variable Y
- x : variable X
- x\_class : variable X en classes pour lesquelles on veut marquer les points observés. Il n'est pas nécessaire de spécifier x\_class si on ne veut pas représenter les points observés (c'est pourquoi sa valeur par défaut est NULL)
- lab.x : titre de l'axe des X
- obs : T ou F selon qu'on veut représenter les points observés ou pas (défaut=T)
- courbe : T ou F selon qu'on veut représenter la courbe modélisée ou pas (défaut=T)
- logit\_inf et logit\_max : ordonnées où doivent être positionnés les points où Logit P n'est pas défini  $\pm$  infini). Ces valeurs déterminent aussi le cadre de la courbe (si les valeurs prédites par le modèle les dépassent, la courbe est tronquée par le haut ou le bas)
- title : titre du graphique
- c1 et c2 : respectivement couleurs de la courbe et des points observés

## Exemple

```
logit_crb (
  data = cycles3,
  reponse = "acc",
  x = "age",
  x_class = "agea",
  mod = fig9,
  lab.x = "âge en années",
  obs = T,
  courbe = F,
  logit_min = -6,
  logit_max = +2,
  title = "Représentation des Logit P observés selon l'âge de la femme en années",
  c1="red",
  c2="green"
)
```

## Résultat



## Fonction ORcl\_pf

Calcul des OR par classes après modélisation par polynômes fractionnaires (fonction principalement utilisé dans le chapitre 4 du livre)

Les arguments de la fonction sont : `ORcl.pf(x, res.mfp, ref, cl)`

- `x` = nom de la variable transformée en polynômes fractionnaires (à mettre entre " ")
- `res.mfp` : résultat du modèle obtenu par `mfp`
- `ref` : référence (centre de la classe de référence)
- `cl` : valeurs pour lesquelles on veut les OR par rapport à `ref` (centres des classes `x1`, `x2`, ... pour lesquelles on veut les OR). Doit être mis sous la forme `cl=c(x1,x2,...)`

Exemple avec le fichier `cycles3` utilisé dans le chapitre 4

```
fp2 <- mfp(acc~fp(age+ovo,df=4,scale=T), family=binomial(), data=cycles3, select=0.05, verbose = TRUE)
ORcl_pf(x="age",res.mfp=fp2,ref=27,cl=c(17,22,32,37,42))
```

## Résultat

Modélisation : `mfp(formula = acc ~ fp(age, df = 4, scale = FALSE), data = cycles3, family = binomial(), select = 0.05, verbose = TRUE)`

Puissance(s) de la variable `age` transformée en polynômes fractionnaires : 3 3

OR et IC pour la variable `age`

```
ref = 27 classe = 17 OR = 0.57 95% IC : [ 0.38 - 0.86 ]
ref = 27 classe = 22 OR = 0.80 95% IC : [ 0.66 - 0.98 ]
ref = 27 classe = 32 OR = 0.96 95% IC : [ 0.85 - 1.10 ]
ref = 27 classe = 37 OR = 0.61 95% IC : [ 0.51 - 0.72 ]
ref = 27 classe = 42 OR = 0.21 95% IC : [ 0.15 - 0.29 ]
```



## Fonction ORcl\_sp

Calcul des OR par classes après modélisation par fonctions splines (fonction principalement utilisée dans le chapitre 4 du livre)

La fonction est appelée après une transformation de la variable X par des fonctions splines cubiques restreints réalisée avec `rcspline.eval()` du package `Hmisc`, suivie d'un modèle de régression logistique avec la variable X transformée et éventuellement d'autres variables.

Les arguments de la fonction sont :

`ORcl_sp(x, spl, reglog, ref, cl)`

- x : nom de la variable transformée en fonctions splines cubiques restreints (à mettre entre "")
- spl : résultat de la transformation en fonctions splines par `rcspline.eval()`
- reglog : résultat du modèle de régression logistique
- ref : référence (en général, centre de la classe de référence)
- cl : valeurs pour lesquelles on veut les OR par rapport à ref (centres des classes pour lesquelles on veut les OR). Doit être mis sous la forme `cl=c(x1,x2,...)`

Exemple avec le fichier `cycles3` utilisé dans le chapitre 4

```
rsc <- rcspline.eval(cycles3$age, nk = 3, inclx = TRUE) # construction des splines
mrsc <- glm(acc ~ rsc+ovo, family = binomial(), data = cycles3) # Modèle logistique
ORcl_sp(x="age",spl=rsc,reglog=mrsc,ref=27,cl=c(17,22,32,37,42))
```

Résultat

Modèle logistique où age est remplacé par 2 fonctions splines ( 3 noeuds)

OR et IC pour la variable age ajustée sur les autres variables du modèle

```
ref = 27  classe = 17  OR = 0.73  95% IC : [ 0.52 - 1.03 ]
ref = 27  classe = 22  OR = 0.85  95% IC : [ 0.72 - 1.02 ]
ref = 27  classe = 32  OR = 1.07  95% IC : [ 0.93 - 1.24 ]
ref = 27  classe = 37  OR = 0.66  95% IC : [ 0.55 - 0.78 ]
ref = 27  classe = 42  OR = 0.29  95% IC : [ 0.22 - 0.39 ]
```

# Chapitre 1

Pas de code pour ce chapitre

# Chapitre 2

## II. Estimation des paramètres de la régression logistique et de leur IC (tableaux 1 à 3)

### Tableau 1

Pour simplement connaître les effectifs :

```
table(geu$ct, geu$salp)
```

```
##
##      0      1
## 0 1122    26
## 1  466    91
```

Pour avoir un résultat plus agréable à lire et disposer d'option

```
CrossTable(geu$ct, geu$salp,
            prop.chisq = FALSE, # Ne pas afficher le test du chi-carré
            prop.r = F,        # Pourcentages par ligne
            prop.c = F,        # Pourcentages par colonne
            prop.t = F,        # Pourcentages par cellule
            format = "SPSS",   # Format pour avoir effectifs et pourcentages
            dnn = c("Grossesse Extra-Utérine", "ATCD Salpingite")) # Noms des
variables
```

```
##
##      Cell Contents
## |-----|
## |                      Count |
## |-----|
##
## Total Observations in Table:  1705
##
##      | ATCD Salpingite
## Grossesse Extra-Utérine |      0      1 | Row Total |
## -----|-----|-----|-----|
##                      0 |    1122    26 |    1148 |
## -----|-----|-----|-----|
##                      1 |     466    91 |     557 |
## -----|-----|-----|-----|
##      Column Total |    1588    117 |    1705 |
## -----|-----|-----|-----|
##
##
```

### Tableaux 2 et 3

Estimation des paramètres de la régression logistique et de leur IC avec la fonction `glm()` (voir la remarque sur `glm` plus bas) : la variable Y (ici `ct`) est indiquée d'abord, elle est séparée des X par `~`, les X sont séparés par des `+` s'il y en a plusieurs (voir plus loin)

```

mod1 <- glm(ct~salp, family=binomial(logit), data=geu)
mod1

##
## Call:  glm(formula = ct ~ salp, family = binomial(logit), data = geu)
##
## Coefficients:
## (Intercept)          salp
##      -0.8787         2.1314
##
## Degrees of Freedom: 1704 Total (i.e. Null);  1703 Residual
## (20 observations deleted due to missingness)
## Null Deviance:      2154
## Residual Deviance: 2046  AIC: 2050

summary(mod1)

##
## Call:
## glm(formula = ct ~ salp, family = binomial(logit), data = geu)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.87868    0.05511 -15.944  <2e-16 ***
## salp         2.13145    0.22910   9.303  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 2154.5  on 1704  degrees of freedom
## Residual deviance: 2046.1  on 1703  degrees of freedom
## (20 observations deleted due to missingness)
## AIC: 2050.1
##
## Number of Fisher Scoring iterations: 4

logistic.display(mod1)

##
## Logistic regression predicting 0:acc 1:GEU
##
##                                OR(95%CI)          P(Wald's test)
## salp clinique prouv ou susp: 1 vs 0  8.43 (5.38,13.2)  < 0.001
##
##                                P(LR-test)
## salp clinique prouv ou susp: 1 vs 0  < 0.001
##
## Log-likelihood = -1023.0534
## No. of observations = 1705
## AIC value = 2050.1068

```

### Remarques

- La commande glm (generalized linear model) indique explicitement que le modèle logistique est un modèle linéaire généralisé avec le lien logit et des résidus suivant la loi binomiale. Il faut préciser dans les paramètres de la fonction que les résidus suivent une loi binomiale : family=binomial; sans cela, la commande glm() considèrera par défaut que les résidus suivent

une loi normale (régression linéaire). Ici la précision (logit) peut être enlevée car elle est implicite lorsque l'on a précisé que les résidus suivent une loi binomiale.

- La fonction `summary(mod1)` affiche plus d'information que `mod1` issu de la fonction `glm()`.
- La sortie de R ne donne malheureusement pas les OR (mais les coefficients) ni les intervalles de confiance (mais les écarts-types des estimations des coefficients). On peut les obtenir avec d'autres commandes comme la fonction `logistic.display(nom.du.modèle)` du package `{epiDisplay}`.

Sinon, il faut un petit travail supplémentaire pour les obtenir. Pour afficher un tableau plus synthétique avec les coefficients, les p et les IC, il faut le programmer explicitement :

```
coef1 <- coef(mod1)
p1 <- summary(mod1)$coefficients[, 'Pr(>|z|)']
IC1 <- confint.default(mod1) # c'est l'IC du livre
                                # avec confint(mod1), ce serait un autre calcul,
                                # très peu différent
cbind(coef1, p1, IC1)
```

```
##                coef1                p1          2.5 %          97.5 %
## (Intercept) -0.8786825 3.137960e-57 -0.9866974 -0.7706675
## salp         2.1314454 1.359189e-20  1.6824140  2.5804769
```

Pour obtenir les OR (et leurs IC) à la place des coefficients (et leurs IC), on en calcule l'exponentielle :

```
cbind(exp(coef1), p1, exp(IC1))

##                p1          2.5 %          97.5 %
## (Intercept) 0.4153298 3.137960e-57 0.3728059 0.4627041
## salp        8.4270386 1.359189e-20 5.3785240 13.2034328
```

### Remarques

- Les opérations ci-dessus ont été réalisées avec les opérations basiques de R. Avec l'expérience, soit vous préférerez continuer à coder vous-mêmes ce genre d'opération (on s'y habitue très vite, et rapidement cela ne vous coûtera plus ni en temps ni en effort...), soit vous préférerez découvrir et utiliser des packages qui ont des fonctions déjà programmées pour effectuer ces opérations et obtenir directement les résultats que vous souhaitez. Tout est bien, c'est une histoire de goût !
- Un avantage quand même à programmer vous-même les opérations : généralement leur exécution prendra moins de temps de calcul qu'une fonction "clé en main" car vous ne demanderez à R que l'essentiel contrairement à la fonction qui fera des calculs supplémentaires dont vous n'aurez pas nécessairement besoin. Pour une simple analyse, la différence ne sera pas perceptible. Mais si vous faites une étude de simulation ou utilisez un bootstrap ou n'importe quelle méthode ré-échantillonnage, la différence peut être notable sur des dizaines de milliers de fois...

Deux autres possibilités pour obtenir l'OR :

- la fonction `cc()`, du package `{epiDisplay}` qui calcule directement (sans modélisation logistique) l'OR d'une association entre un outcome et une variable d'exposition :

```
cc(geu$ct, geu$salp, graph=F)

##
##      geu$salp
## geu$ct      0      1 Total
##   0      1122    26  1148
##   1       466    91   557
## Total 1588   117  1705
##
## OR = 8.43
## 95% CI = 5.38, 13.2
```

```
## Chi-squared = 116.21, 1 d.f., P value = 0
## Fisher's exact test (2-sided) P value = 0
```

- la fonction odds.ratio du package {questionr}

```
odds.ratio(mod1)

## Waiting for profiling to be done...

##              OR    2.5 %  97.5 %           p
## (Intercept) 0.41533 0.37250 0.4623 < 2.2e-16 ***
## salp        8.42704 5.45930 13.4504 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

odds.ratio(table(geu$ct,geu$salp))

##              OR    2.5 %  97.5 %           p
## Fisher's test 8.4149 5.3109 13.75 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

## IV. Test des paramètres du modèle logistique

### Test du rapport de vraisemblances (Tableau 4)

```
mod0 <- glm(ct~NULL, data=geu[!is.na(geu$salp),] , family="binomial")
# !is.na(geu$salp) est là pour enlever les données manquantes
# et estimer mod0 sur les mêmes sujets que mod1
epiDisplay::lrtest(mod0, mod1)

## Likelihood ratio test for MLE method
## Chi-squared 1 d.f. = 108.3551 , P value = 2.246953e-25
```

#### Remarques

- Si on écrit seulement "lrtest(mod0, mod1)", on obtient un message d'erreur "Error: unable to find an inherited method for function 'lrtest' for signature 'object = \"glm\"'", pas très clair comme la plupart des messages d'erreur des logiciels ...

Ce qui se passe ici, c'est qu'un autre package chargé après epiDisplay a aussi une commande lrtest qui ne s'applique pas aux modèles glm (ici il s'agit de VGAM. Il faut dire explicitement que c'est la commande lrtest de epiDisplay qu'on veut et écrire epiDisplay::lrtest(mod0, mod1)

- Si on oublie d'enlever les données manquantes, on obtient un message d'erreur. Exemple :

```
mod0.oubli.mq <- glm(ct~NULL, data=geu , family="binomial")
epiDisplay::lrtest(mod0.oubli.mq, mod1)

## Error in epiDisplay::lrtest(mod0.oubli.mq, mod1): Number of observation not equal!!
```

- Par contre, si deux modèles ne sont pas emboîtés, mais réalisés sur le même nombre de sujets, il n'y a pas de message d'erreur. Il faut donc vérifier soi-même que les modèles sont bien emboîtés.

### Test du khi2 (Tableau 5)

Pour réaliser le test du khi2 de Pearson avec la fonction de base chisq.test() :

```
a <- table(geu$ct, geu$salp, dnn= c("Grossesse Extra-Utérine" , "ATCD Salpingite"))
chisq.test(a, corr=F)

##
## Pearson's Chi-squared test
##
## data:  a
## X-squared = 116.21, df = 1, p-value < 2.2e-16
```

Avec CrossTable()

```
CrossTable(geu$ct, geu$salp, prop.chisq = T, prop.c=F, prop.r=F, prop.t=F, format = "SPSS", dnn = c("Grossesse Extra-Utérine" , "ATCD Salpingite"))

##
## Cell Contents
## |-----|
## | Count |
## | Chi-square contribution |
## |-----|
##
## Total Observations in Table: 1705
##
## Grossesse Extra-Utérine | ATCD Salpingite |
## | 0 | 1 | Row Total |
## |-----|-----|-----|
## | 0 | 1122 | 26 | 1148 |
## | 2.605 | 35.359 | |
## |-----|-----|-----|
## | 1 | 466 | 91 | 557 |
## | 5.369 | 72.876 | |
## |-----|-----|-----|
## | Column Total | 1588 | 117 | 1705 |
## |-----|-----|-----|
##
```

On note que le résultat est la contribution au khi2 de chaque case. Il faut donc additionner ces contributions pour avoir la valeur du khi2 (et cela ne donne pas son degré de signification ...)

```
##
```

## Test de wald

```
summary(mod1)

##
## Call:
## glm(formula = ct ~ salp, family = binomial(logit), data = geu)
##
## Coefficients:
## Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.87868 0.05511 -15.944 <2e-16 ***
## salp 2.13145 0.22910 9.303 <2e-16 ***
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
```

```
##
## Null deviance: 2154.5 on 1704 degrees of freedom
## Residual deviance: 2046.1 on 1703 degrees of freedom
## (20 observations deleted due to missingness)
## AIC: 2050.1
##
## Number of Fisher Scoring iterations: 4
```

Les colonnes “z value” et “p value” correspondent au test de Wald.

### Test simultané de plusieurs coefficients

```
mod2 <- glm(ct~salp+univf+fprof, data=geu, family="binomial")
summary(mod2)

##
## Call:
## glm(formula = ct ~ salp + univf + fprof, family = "binomial",
## data = geu)
##
## Coefficients:
## Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.1044 0.1223 -9.034 < 2e-16 ***
## salp 2.0004 0.2631 7.604 2.87e-14 ***
## univf -0.1653 0.1562 -1.058 0.2898
## fprof 0.3141 0.1448 2.169 0.0301 *
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 1533.5 on 1229 degrees of freedom
## Residual deviance: 1461.9 on 1226 degrees of freedom
## (495 observations deleted due to missingness)
## AIC: 1469.9
##
## Number of Fisher Scoring iterations: 4
```

### Test de Wald de plusieurs coefficients (Tableau 7)

```
wald.test(b = coef(mod2), Sigma = vcov(mod2), Terms = 3:4)

## Wald test:
## -----
##
## Chi-squared test:
## X2 = 5.1, df = 2, P(> X2) = 0.08
```

La commande `wald.test()` du package `{aod}` nécessite de préciser quelle matrice de variances-covariances des coefficients on veut utiliser. Pour tester simultanément `univf` et `fprof`, il faut préciser que l’on s’intéresse aux 3ème et 4ème coefficients. Il ne faut pas oublier la constante qui est le 1er coefficient ...

Pour avoir plus de décimales, mais ce n’est pas essentiel ...

```
x<-wald.test(b = coef(mod2), Sigma = vcov(mod2), Terms = 3:4)
print(x,digits=3)

## Wald test:
## -----
```



```
##
## Chi-squared test:
## X2 = 5.06, df = 2, P(> X2) = 0.0797
```

### Test du rapport de vraisemblances avec plusieurs variables (Tableau 7)

Le test doit être fait sur les mêmes sujets.

```
mod2b <- glm(ct~salp+univf+fprof, data=geu, family="binomial")
mod3b <- glm(ct~salp, data=na.omit(geu[, c("ct", "salp", "univf", "fprof")]), family="binomial")
epiDisplay::lrtest(mod2b, mod3b)

## Likelihood ratio test for MLE method
## Chi-squared 2 d.f. = 5.134777, P value = 0.07673569
```

Il est aussi possible de créer, dans l'environnement R, une nouvelle table dont les colonnes correspondent aux seules variables utiles et en ne gardant que les lignes des sujets dont les observations sont complètes pour ces variables. Puis, de faire le test sur les deux modèles emboîtés sans plus se préoccuper du problème des données manquantes

```
data.lrt <- na.omit(geu[, c("ct", "salp", "univf", "fprof")])

mod2c <- glm(ct~salp+univf+fprof, data=data.lrt, family="binomial")
mod3c <- glm(ct~salp, data=data.lrt, family="binomial")
epiDisplay::lrtest(mod2c, mod3c)

## Likelihood ratio test for MLE method
## Chi-squared 2 d.f. = 5.134777, P value = 0.07673569
```

#### Remarque

Les modèles mod2, mod2b et mod2c sont en fait identiques car la régression ne se fait de toute façon que sur les sujets sans données manquantes. Ce n'est que pour comparer à mod3b qu'il faut faire attention aux données manquantes.

# Chapitre 3

## I. Règle générale d'interprétation du coefficient d'une variable (tableau 1)

```
mod.age <- glm(ct~salp, family=binomial(logit), data=geu)
summary(mod.age)

##
## Call:
## glm(formula = ct ~ salp, family = binomial(logit), data = geu)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.87868    0.05511 -15.944  <2e-16 ***
## salp         2.13145    0.22910   9.303  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##    Null deviance: 2154.5  on 1704  degrees of freedom
## Residual deviance: 2046.1  on 1703  degrees of freedom
## (20 observations deleted due to missingness)
## AIC: 2050.1
##
## Number of Fisher Scoring iterations: 4
```

## II. Variable dichotomique

### Changement de codage de salp (Tableau 2)

Le codage de la variable salp0\_2 est 0 et 2 au lieu de 0 et 1

```
geu$salp0_2 <- 2*geu$salp
mod.salp0_2 <- glm(ct ~ salp0_2 , data=geu , family="binomial")
summary(mod.salp0_2)$coefficients

##             Estimate Std. Error    z value      Pr(>|z|)
## (Intercept) -0.8786825 0.05511066 -15.943965 3.137960e-57
## salp0_2      1.0657227 0.11455094   9.303483 1.359189e-20

summary(mod.salp0_2)

##
## Call:
## glm(formula = ct ~ salp0_2, family = "binomial", data = geu)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.87868    0.05511 -15.944  <2e-16 ***
## salp0_2      1.06572    0.11455   9.303  <2e-16 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##    Null deviance: 2154.5  on 1704  degrees of freedom
## Residual deviance: 2046.1  on 1703  degrees of freedom
## (20 observations deleted due to missingness)
## AIC: 2050.1
##
## Number of Fisher Scoring iterations: 4

logistic.display(mod.salp0_2)

##
## Logistic regression predicting 0:acc 1:GEU
##
##              OR(95%CI)      P(Wald's test) P(LR-test)
## NA (cont. var.) 2.9 (2.32,3.63) < 0.001      < 0.001
##
## Log-likelihood = -1023.0534
## No. of observations = 1705
## AIC value = 2050.1068
```

### III. Variable qualitative nominale à plus de 2 classes

#### III.2. Interprétation des coefficients d'une variable indicatrice

###Tableau 4)

```
geu$gind.fact <- factor(geu$gind , labels=c("Non Ind" , "Hcg" , "Clomid" , "Autre"))

mod.gindf <- glm(ct ~ gind.fact, data=geu , family="binomial")
summary(mod.gindf)

##
## Call:
## glm(formula = ct ~ gind.fact, family = "binomial", data = geu)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -0.73306    0.05295  -13.844   <2e-16 ***
## gind.factHcg    0.47070    0.42394   1.110   0.2669
## gind.factClomid 0.66637    0.26371   2.527   0.0115 *
## gind.factAutre  0.95620    0.67291   1.421   0.1553
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##    Null deviance: 2186.9  on 1718  degrees of freedom
## Residual deviance: 2177.7  on 1715  degrees of freedom
## (6 observations deleted due to missingness)
## AIC: 2185.7
##
## Number of Fisher Scoring iterations: 4
```

Il est aussi possible de faire la transformation de gind directement dans la commande glm. On obtient le même modèle, seuls les intitulés des lignes changent

```
mod.gindf2 <- glm(ct ~ as.factor(gind) , data=geu , family="binomial")
summary(mod.gindf2)

##
## Call:
## glm(formula = ct ~ as.factor(gind), family = "binomial", data = geu)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -0.73306    0.05295  -13.844   <2e-16 ***
## as.factor(gind)1  0.47070    0.42394   1.110   0.2669
## as.factor(gind)2  0.66637    0.26371   2.527   0.0115 *
## as.factor(gind)3  0.95620    0.67291   1.421   0.1553
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 2186.9  on 1718  degrees of freedom
## Residual deviance: 2177.7  on 1715  degrees of freedom
## (6 observations deleted due to missingness)
## AIC: 2185.7
##
## Number of Fisher Scoring iterations: 4
```

### Remarque

La 1ère méthode avec factor() est conseillée pour prendre le temps de bien définir les variables qualitatives avant de commencer les analyses (cela se fait habituellement dans les étapes "data management"). Elle permet de plus de changer facilement de catégorie de référence avec la commande relevel() (voir un peu plus loin).

### Tableau 5

```
table(geu$ct, geu$gind, dnn= c("Grossesse Extra-Utérine" , "Grossesse induite")
)

##              Grossesse induite
## Grossesse Extra-Utérine      0      1      2      3
##              0 1099      13      31      4
##              1  528      10      29      5
```

## III.3 OR entre les catégories d'une variable indicatrice

### Changement de catégorie de référence (Tableau 6)

```
geu$gind.f.clomid <- relevel(geu$gind.fact , "Clomid")
mod.gind.clomid <- glm(ct ~ gind.f.clomid , data=geu , family="binomial")
logistic.display(mod.gind.clomid)

##
## Logistic regression predicting 0:acc 1:GEU
##
##              OR(95%CI)              P(Wald's test) P(LR-test)
## NA: ref.=Clomid              0.027
##      Non Ind      0.51 (0.31,0.86)  0.012
```

```
##      Hcg          0.82 (0.31,2.16)  0.692
##      Autre        1.34 (0.33,5.47)  0.687
##
## Log-likelihood = -1088.8724
## No. of observations = 1719
## AIC value = 2185.7447
```

### Tableau 7

```
geu$gind.f.3 <- relevel(geu$gind.fact, "Autre")
mod.gind.3 <- glm(ct ~ gind.f.3, data=geu, family="binomial")
logistic.display(mod.gind.3)
```

```
##
## Logistic regression predicting 0:acc 1:GEU
##
##              OR(95%CI)          P(Wald's test) P(LR-test)
## NA: ref.=Autre                                0.027
##      Non Ind      0.38 (0.1,1.44)    0.155
##      Hcg          0.62 (0.13,2.9)    0.54
##      Clomid       0.75 (0.18,3.06)    0.687
##
## Log-likelihood = -1088.8724
## No. of observations = 1719
## AIC value = 2185.7447
```

### III.3.b Commande lincom (Tableau 8)

```
combi.lin1 <- lincom(model = mod.gindf, specification = "gind.factAutre-gind.factClomid")
combi.lin1
```

```
##              Estimate      2.5 %   97.5 %   Chisq
## gind.factAutre-gind.factClomid 0.2898349 -1.119079 1.698749 0.1625656
##              Pr(>Chisq)
## gind.factAutre-gind.factClomid 0.6868053

exp(combi.lin1[,1:3])

##              Estimate      2.5 %   97.5 %
## gind.factAutre-gind.factClomid 1.336207 0.3265804 5.467103
```

### III.4. Test de l'association entre Y et X décomposée en variables indicatrices

#### Test de Wald (Tableau 9)

```
print(wald.test(b=coef(mod.gindf), Sigma = vcov(mod.gindf), Terms = 2:4), digits = 3)

## Wald test:
## -----
##
## Chi-squared test:
## X2 = 9.36, df = 3, P(> X2) = 0.0248
```

#### Test du rapport de vraisemblances (Tableau 10)

```
mod.null <- glm(ct ~ NULL, data=na.omit(geu[, c("ct", "gind.fact")]), family="binomial")
epiDisplay::lrtest(mod.gindf, mod.null)
```

```
## Likelihood ratio test for MLE method
## Chi-squared 3 d.f. = 9.20205 , P value = 0.02672171
```

## IV. Variable qualitative ordinale

Tableau 12

```
table(geu$ct, geu$tabfc, dnn = c("Grossesse Extra-Utérine" , "Consommation de t
abac"))
```

```
##                               Consommation de tabac
## Grossesse Extra-Utérine      0   1   2   3
##                               0 809 158 106  71
##                               1 264  78 113  94
```

### IV.1 Modélisation de l'association GEU - tabac

Figure 1

```
data.fig1 <- na.omit( geu[ , c("ct","tabfc")])
mod.lin<-glm(ct ~ tabfc , data=geu , family="binomial")
pred<-predict(mod.lin) ## valeurs prédites de Logit P par une droite

cl.tabf <- geu %>% ## création d'une table pour les valeurs observées par clas
se de tabfc
group_by(tabfc) %>%
  summarize(N=n(),
    prop = mean(ct),
    logitP = log(prop/(1-prop))) %>%
  ungroup()
names(cl.tabf) <- c("tabfc", "N","prop","logitP")

p<-ggplot (data=data.fig1) +
  geom_line (aes(x=tabfc,y=pred,color="Logit P"),linewidth=0.5)+
  geom_point (data=cl.tabf,
    aes(x=tabfc, y=logitP,color="Logits observés"),size=2)+
  scale_y_continuous(breaks=seq(-1.5,1,0.5)) +
  labs(x="Tabac",y="Logit P")+
  scale_color_manual(values = c("Logit P" = "black", "Logit observés" = "grey"))
) +
  theme_classic() + theme(legend.position="none")
print(p)

## Warning: Removed 1 row containing missing values or values outside the scale
range
## (`geom_point()`).
```

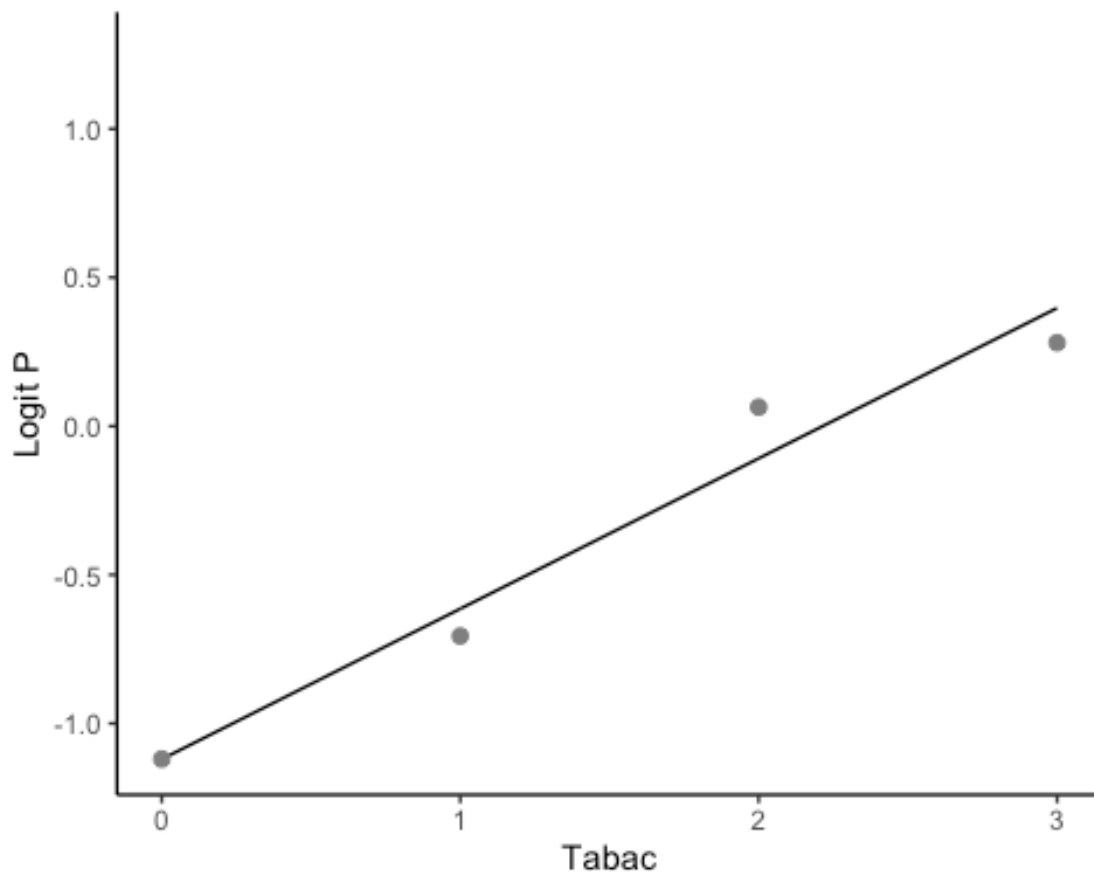


Tableau 13, partie supérieure

```
geu$tab.f <- factor(geu$tabfc , labels=c("nf", "1-9", "10-19", ">=20"))
mod.tabc <- glm(ct ~ tab.f , data=geu , family="binomial")
logistic.display(mod.tabc)
```

```
##
## Logistic regression predicting 0:acc 1:GEU
##
##              OR(95%CI)          P(Wald's test) P(LR-test)
## NA: ref.=nf                      < 0.001
##   1-9      1.51 (1.12,2.05)  0.008
##   10-19    3.27 (2.42,4.41) < 0.001
##   >=20     4.06 (2.89,5.69) < 0.001
##
## Log-likelihood = -1012.8708
## No. of observations = 1693
## AIC value = 2033.7416
```

On remarque qu'à la place du nom de la variable, R affiche NA. C'est parce qu'il cherche son label, qu'on n'a pas déclaré.

Pour une sortie plus jolie :

```
ncol(geu)

## [1] 40

# C'est le nombre de colonnes (variables) du fichier geu. C'est donc aussi le
# rang de la dernière, celle qu'on vient de créer : tab.f
attr(geu,"var.labels")[ncol(geu)] <- "Tabagisme"
# ici cela revient au même que attr(geu,"var.labels")[40] <- "Tabagisme"
```

```
mod.tabc <- glm(ct ~ tab.f , data=geu , family="binomial")
logistic.display(mod.tabc)

##
## Logistic regression predicting 0:acc 1:GEU
##
##               OR(95%CI)           P(Wald's test) P(LR-test)
## Tabagisme: ref.=nf                < 0.001
##   1-9                1.51 (1.12,2.05)  0.008
##   10-19              3.27 (2.42,4.41) < 0.001
##   >=20              4.06 (2.89,5.69) < 0.001
##
## Log-likelihood = -1012.8708
## No. of observations = 1693
## AIC value = 2033.7416
```

### Tableau 13, partie inférieure

*# valeurs des 3 OR données par Le modèle Linéaire*

```
summary(mod.lin)

##
## Call:
## glm(formula = ct ~ tabfc, family = "binomial", data = geu)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.12028    0.06732  -16.64  <2e-16 ***
## tabfc        0.50586    0.05000   10.12  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##    Null deviance: 2133.4  on 1692  degrees of freedom
## Residual deviance: 2028.4  on 1691  degrees of freedom
## (32 observations deleted due to missingness)
## AIC: 2032.4
##
## Number of Fisher Scoring iterations: 4

logistic.display(mod.lin)

##
## Logistic regression predicting 0:acc 1:GEU
##
##               OR(95%CI)           P(Wald's test)
## 0:nf;1:1-9;2:10-19;3:>=20 (cont. var.) 1.66 (1.5,1.83) < 0.001
##
##               P(LR-test)
## 0:nf;1:1-9;2:10-19;3:>=20 (cont. var.) < 0.001
##
## Log-likelihood = -1014.1795
## No. of observations = 1693
## AIC value = 2032.3591

combi.lin1 <- lincom(model = mod.lin, specification = "2*tabfc=0")
combi.lin1
```



```
##           Estimate    2.5 %   97.5 %   Chisq   Pr(>Chisq)
## 2*tabfc=0 1.011716 0.8157296 1.207702 102.3674 4.612267e-24

exp(combi.lin1[,1:3])

##           Estimate    2.5 %   97.5 %
## 2*tabfc=0 2.750317 2.260825 3.345788

# syntaxe "ramassée"
exp(lincom(model = mod.lin, specification = "3*tabfc=0")[,1:3])

##           Estimate    2.5 %   97.5 %
## 3*tabfc=0 4.561146 3.399385 6.119948

# syntaxe avec %>%
lincom(model = mod.lin, specification = "2*tabfc=0") %>%
  .[,1:3] %>%
  exp()

##           Estimate    2.5 %   97.5 %
## 2*tabfc=0 2.750317 2.260825 3.345788
```

## IV.2. Choix des valeurs de X

Tableau 14

```
geu$tabf2<-
  if_else(geu$tabfc==1,5,
    if_else(geu$tabfc==2,15,
      if_else(geu$tabfc==3,25,geu$tabfc)))

mod.tabf2 <- glm(ct ~ tabf2 , data=geu , family="binomial")
summary(mod.tabf2)

##
## Call:
## glm(formula = ct ~ tabf2, family = "binomial", data = geu)
##
## Coefficients:
##           Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.083099   0.065001  -16.66  <2e-16 ***
## tabf2        0.062261   0.006221   10.01  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 2133.4 on 1692 degrees of freedom
## Residual deviance: 2030.2 on 1691 degrees of freedom
## (32 observations deleted due to missingness)
## AIC: 2034.2
##
## Number of Fisher Scoring iterations: 4

logistic.display(mod.tabf2)

##
## Logistic regression predicting 0:acc 1:GEU
##
```

```
##              OR(95%CI)      P(Wald's test) P(LR-test)
## NA (cont. var.) 1.06 (1.05,1.08) < 0.001      < 0.001
##
## Log-likelihood = -1015.1054
## No. of observations = 1693
## AIC value = 2034.2108

lincom(model = mod.tabf2, specification = "5*tabf2=0") %>%
  .[,1:3] %>%
  exp()

##              Estimate      2.5 %    97.5 %
## 5*tabf2=0 1.365206 1.284469 1.451018

lincom(model = mod.tabf2, specification = "15*tabf2=0") %>%
  .[,1:3] %>%
  exp()

##              Estimate      2.5 %    97.5 %
## 15*tabf2=0 2.544454 2.119195 3.055049

lincom(model = mod.tabf2, specification = "25*tabf2=0") %>%
  .[,1:3] %>%
  exp()

##              Estimate      2.5 %    97.5 %
## 25*tabf2=0 4.742321 3.496377 6.43226
```

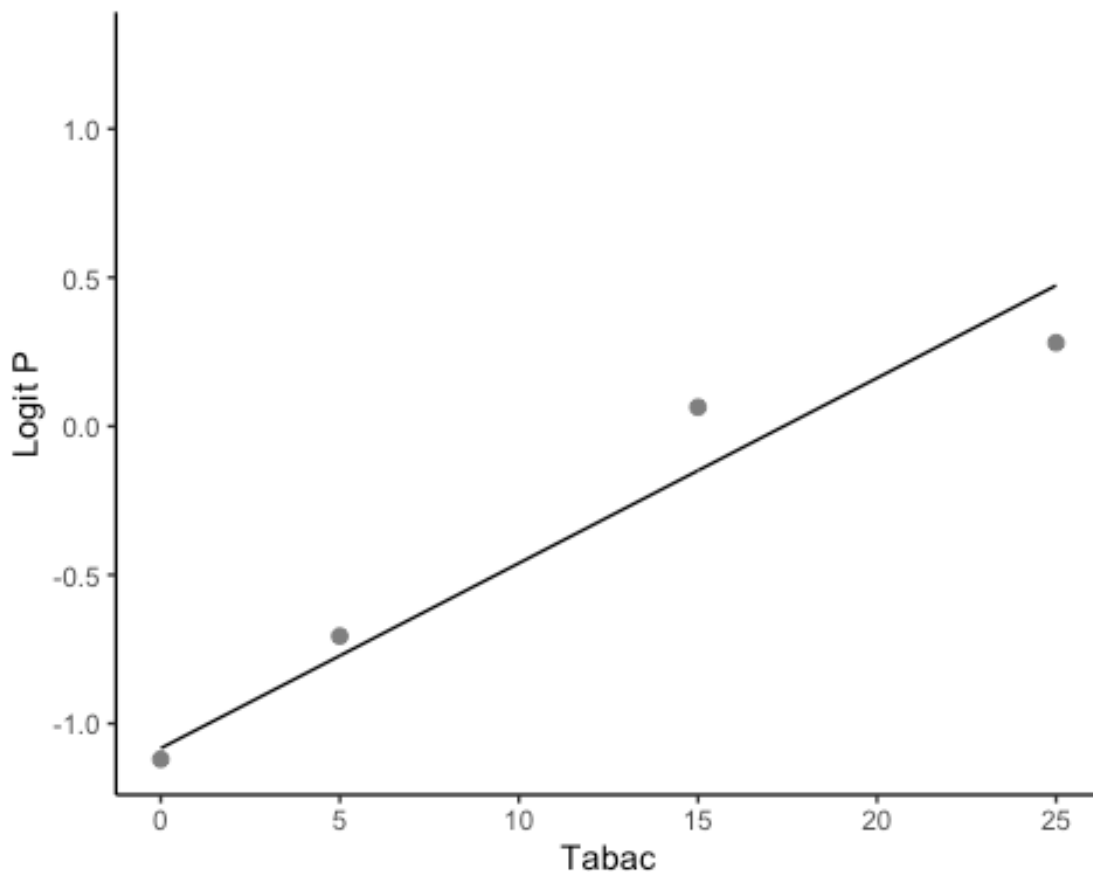
**Figure 2**

```
data.fig2 <- na.omit( geu[ , c("ct","tabf2")])
mod.lin2<-glm(ct ~ tabf2 , data=geu , family="binomial")
pred<-predict(mod.lin2) ## valeurs prédites de Logit P par une droite

cl.tabf2 <- geu %>% ## création d'une table pour les valeurs observées par classe de tabf2
group_by(tabf2) %>%
  summarize(N=n(),
    prop = mean(ct),
    logitP = log(prop/(1-prop))) %>%
  ungroup()
names(cl.tabf2) <- c("tabf2", "N","prop","logitP")

p<-ggplot (data=data.fig2) +
  geom_line (aes(x=tabf2,y=pred,color="Logit P"),linewidth=0.5)+
  geom_point (data=cl.tabf2,
    aes(x=tabf2, y=logitP,color="Logits observés"),size=2)+
  scale_y_continuous(breaks=seq(-1.5,1,0.5)) +
  labs(x="Tabac",y="Logit P")+
  scale_color_manual(values = c("Logit P" = "black", "Logit observés" = "grey"))
) +
  theme_classic() + theme(legend.position="none")
print(p)

## Warning: Removed 1 row containing missing values or values outside the scale
## range
## (`geom_point()`).
```



Test de linéarité

```
mod.tabf2c <- glm(ct ~ as.factor(tabf2) , data=geu , family="binomial")
epiDisplay::lrtest(mod.tabf2, mod.tabf2c)

## Likelihood ratio test for MLE method
## Chi-squared 2 d.f. = 4.469182 , P value = 0.1070359
```

### IV.3. Choix entre variables indicatrices et modélisation linéaire (test de linéarité)

Figure 3 : je n'ai pas programmé la figure 3 en R. Ce n'est de toute façon pas essentiel pour la pratique de l'analyse de données. Au-delà de cette excuse facile, j'ai l'intention de le faire un jour ....

#### Test de linéarité (tableau 15)

```
epiDisplay::lrtest (mod.tabc, mod.lin)

## Likelihood ratio test for MLE method
## Chi-squared 2 d.f. = 2.617404 , P value = 0.2701705

linearHypothesis(mod.tabc, c("tab.f10-19 = 2*tab.f1-9",
                             "tab.f>=20 = 3*tab.f1-9"), test="Chisq")

## Linear hypothesis test
##
## Hypothesis:
## tab.f10-19 = 2*tab.f1-9
## tab.f> = 20 = 3*tab.f1-9
```

```
##
## Model 1: restricted model
## Model 2: ct ~ tab.f
##
##   Res.Df Df    Chisq Pr(>Chisq)
## 1    1691
## 2    1689  2  2.6153    0.2705
```

## VI. Prise en compte d'une interaction

### VI.1. Interaction entre variables qualitatives

Tableau 16

```
mod.inter <- glm(ct ~ clomid*age30 , data=geu , family="binomial")
summary(mod.inter)

##
## Call:
## glm(formula = ct ~ clomid * age30, family = "binomial", data = geu)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.02670    0.07446 -13.789  < 2e-16 ***
## clomid        1.23434    0.38074   3.242  0.00119 **
## age30         0.65112    0.10726   6.071  1.27e-09 ***
## clomid:age30 -1.12703    0.53542  -2.105  0.03530 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##    Null deviance: 2136.9  on 1682  degrees of freedom
## Residual deviance: 2092.2  on 1679  degrees of freedom
## (42 observations deleted due to missingness)
## AIC: 2100.2
##
## Number of Fisher Scoring iterations: 4

logistic.display(mod.inter)

##
## Logistic regression predicting 0:acc 1:GEU
##
##              crude OR(95%CI)  adj. OR(95%CI)
## induction par clomid: 1 vs 0  2.01 (1.19,3.38)  3.44 (1.63,7.25)
##
## age >30: 1 vs 0              1.84 (1.5,2.26)  1.92 (1.55,2.37)
##
## clomid:age30                 -                0.32 (0.11,0.93)
##
##              P(Wald's test) P(LR-test)
## induction par clomid: 1 vs 0  0.001      0.001
##
## age >30: 1 vs 0              < 0.001      < 0.001
##
```

```
## clomid:age30          0.035          0.034
##
## Log-likelihood = -1046.124
## No. of observations = 1683
## AIC value = 2100.2479
```

La deuxième colonne donne les exponentielles des coefficients du modèle, c'est-à-dire les odds ratios (en faisant bien sûr attention à leur interprétation en raison du terme d'interaction, voir § VI.1.b).

La première colonne (crude OR) donne l'OR brut de chaque variable, calculé dans des modèles univariés séparés (un avec age30 et un autre avec clomid) et à partir d'une base de données dans laquelle sont exclus les sujets pour lesquels *l'une ou l'autre* des variables (ici clomid et age30) est manquante.

Ici, pour l'analyse de l'interaction, ce n'est pas intéressant, et il est préférable de ne pas afficher cette colonne en utilisant l'option "simplified=TRUE".

```
logistic.display(mod.inter, simplified=T)
```

```
##
##              OR lower95ci upper95ci      Pr(>|Z|)
## clomid        3.4361068 1.6292060 7.2469840 1.187333e-03
## age30         1.9176937 1.5541063 2.3663433 1.274303e-09
## clomid:age30  0.3239952 0.1134459 0.9253126 3.529642e-02
```

### Tableau 17

Pour obtenir l'OR de clomid, et son IC, lorsque age30==1, il y a 2 possibilités :

- 1ère méthode : Changer la classe de référence de age30

```
mod.inter.ref <- glm(ct ~ clomid*relevel(factor(age30),"1") , data=geu , family="binomial")
summary(mod.inter.ref)

##
## Call:
## glm(formula = ct ~ clomid * relevel(factor(age30), "1"), family = "binomial"
## ,
##     data = geu)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -0.37558    0.07721  -4.865 1.15e-06 ***
## clomid           0.10731    0.37644   0.285  0.7756
## relevel(factor(age30), "1")0 -0.65112    0.10726 -6.071 1.27e-09 ***
## clomid:relevel(factor(age30), "1")0 1.12703    0.53542  2.105  0.0353 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 2136.9 on 1682 degrees of freedom
## Residual deviance: 2092.2 on 1679 degrees of freedom
## (42 observations deleted due to missingness)
## AIC: 2100.2
##
## Number of Fisher Scoring iterations: 4
```

```
logistic.display(mod.inter.ref,simplified=T)

##
##                                OR lower95ci upper95ci      Pr(>|Z
|)
## clomid                        1.1132821 0.5323271 2.3282620 7.755893e-
01
## relevel(factor(age30), "1")0  0.5214597 0.4225929 0.6434566 1.274303e-
09
## clomid:relevel(factor(age30), "1")0 3.0864656 1.0807159 8.8147773 3.529642e-
02
```

- 2ème methode : utiliser une combinaison linéaire.

Cette méthode est plus directe

```
combi.lin2<- lincom(mod.inter, specification = "clomid+clomid:age30=0")
combi.lin2

##              Estimate      2.5 %    97.5 %      Chisq Pr(>Chisq)
## clomid+clomid:age30=0 0.1073125 -0.6304971 0.8451221 0.08126582 0.7755893

exp(combi.lin2[1:3])

##              Estimate      2.5 %    97.5 %
## clomid+clomid:age30=0 1.113282 0.5323271 2.328262
```

### Interaction et facteur à plus de 2 classes (tableau 18)

```
geu$agec3.f <- factor(geu$agec3, labels=c("<30", "30-34", ">=35"))

mod.inter2 <- glm(ct ~ clomid*agec3.f , data=geu, family=binomial)
summary(mod.inter2)

##
## Call:
## glm(formula = ct ~ clomid * agec3.f, family = binomial, data = geu)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -1.02670     0.07446 -13.789  < 2e-16 ***
## clomid          1.23434     0.38074   3.242  0.00119 **
## agec3.f30-34    0.52515     0.11738   4.474 7.68e-06 ***
## agec3.f>=35     1.00423     0.16739   5.999 1.98e-09 ***
## clomid:agec3.f30-34 -1.29240     0.59129  -2.186  0.02884 *
## clomid:agec3.f>=35  -0.70104     0.83712  -0.837  0.40235
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 2136.9  on 1682  degrees of freedom
## Residual deviance: 2083.2  on 1677  degrees of freedom
## (42 observations deleted due to missingness)
## AIC: 2095.2
##
## Number of Fisher Scoring iterations: 4

logistic.display(mod.inter2,simplified=T)
```

```
##
##              OR   lower95ci upper95ci   Pr(>|Z|)
## clomid        3.4361068 1.62920596 7.2469840 1.187333e-03
## agec3.f30-34   1.6907086 1.34325079 2.1280430 7.676320e-06
## agec3.f>=35    2.7297959 1.96630204 3.7897462 1.980031e-09
## clomid:agec3.f30-34 0.2746101 0.08617988 0.8750387 2.883623e-02
## clomid:agec3.f>=35 0.4960688 0.09615716 2.5591883 4.023450e-01
```

### Test de l'interaction

- Comparaison de deux modèles emboîtés

```
mod.ss.inter <- glm(ct ~ clomid+agec3.f , data=geu, family=binomial)
epiDisplay::lrtest(mod.inter2, mod.ss.inter)
```

```
## Likelihood ratio test for MLE method
## Chi-squared 2 d.f. = 4.94586 , P value = 0.0843374
```

- Test de Wald global

```
print(wald.test(b = coef(mod.inter2),
                 Sigma = vcov(mod.inter2),
                 Terms = 5:6),
      digits = 3)
```

```
## Wald test:
## -----
##
## Chi-squared test:
## X2 = 4.82, df = 2, P(> X2) = 0.0898
```

Remarque : Dans la commande précédente, il faut savoir quel est le rang des termes qu'on veut tester dans le mod.inter2 (ici 5 et 6). On le trouve en affichant les résultats du modèle (par exemple avec `summary(mod.inter2)$coefficients`).

## VI.2. Interaction avec F et analyses séparées selon les niveaux de F

Tableau 19

```
mod.inter2 <- glm(ct ~ clomid*age30 , data=geu, family=binomial)

mod.inf30 <- glm(ct ~ clomid , data=geu[geu$age30==0 , ] , family="binomial")
mod.sup30 <- glm(ct ~ clomid , data=geu[geu$age30==1 , ] , family="binomial")

summary(mod.inter2)

##
## Call:
## glm(formula = ct ~ clomid * age30, family = binomial, data = geu)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.02670    0.07446 -13.789  < 2e-16 ***
## clomid        1.23434    0.38074   3.242  0.00119 **
## age30         0.65112    0.10726   6.071 1.27e-09 ***
## clomid:age30 -1.12703    0.53542  -2.105  0.03530 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 2136.9  on 1682  degrees of freedom
## Residual deviance: 2092.2  on 1679  degrees of freedom
## (42 observations deleted due to missingness)
## AIC: 2100.2
##
## Number of Fisher Scoring iterations: 4

summary(mod.inf30)

##
## Call:
## glm(formula = ct ~ clomid, family = "binomial", data = geu[geu$age30 ==
##      0, ])
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.02670     0.07446 -13.789  < 2e-16 ***
## clomid       1.23434     0.38074   3.242  0.00119 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 1122.1  on 957  degrees of freedom
## Residual deviance: 1111.8  on 956  degrees of freedom
## (19 observations deleted due to missingness)
## AIC: 1115.8
##
## Number of Fisher Scoring iterations: 4

summary(mod.sup30)

##
## Call:
## glm(formula = ct ~ clomid, family = "binomial", data = geu[geu$age30 ==
##      1, ])
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.37558     0.07721  -4.865 1.15e-06 ***
## clomid       0.10731     0.37644   0.285   0.776
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 980.53  on 724  degrees of freedom
## Residual deviance: 980.45  on 723  degrees of freedom
## (27 observations deleted due to missingness)
## AIC: 984.45
##
## Number of Fisher Scoring iterations: 4
```



# Chapitre 4

## IV. Courbes lowess (figure 8)

```
for (span in c(.8 , .1) ) {      # c(.8 , .1)
  lfit<-loess(cycles3$acc~cycles3$age,span=span)
  lpred<-log(predict(lfit)/(1-predict(lfit)))

  # Valeurs observées par classe de agea (c'est une partie du code de la fonction plot_predict)

  tranche.age <- cycles3 %>%
    group_by(agea) %>%
    summarize(N=n(),
              prop = mean(acc),
              logitP = log(prop/(1-prop))) %>%
    ungroup()
  names(tranche.age) <- c("age", "N","prop","logitP")
  tranche.age$point <- 16
  tranche.age$point[tranche.age$logitP== -Inf] <- 25
  tranche.age$point[tranche.age$logitP== Inf] <- 24

  tranche.age$logitP[tranche.age$logitP== -Inf] <- -6
  tranche.age$logitP[tranche.age$logitP== Inf] <- 2

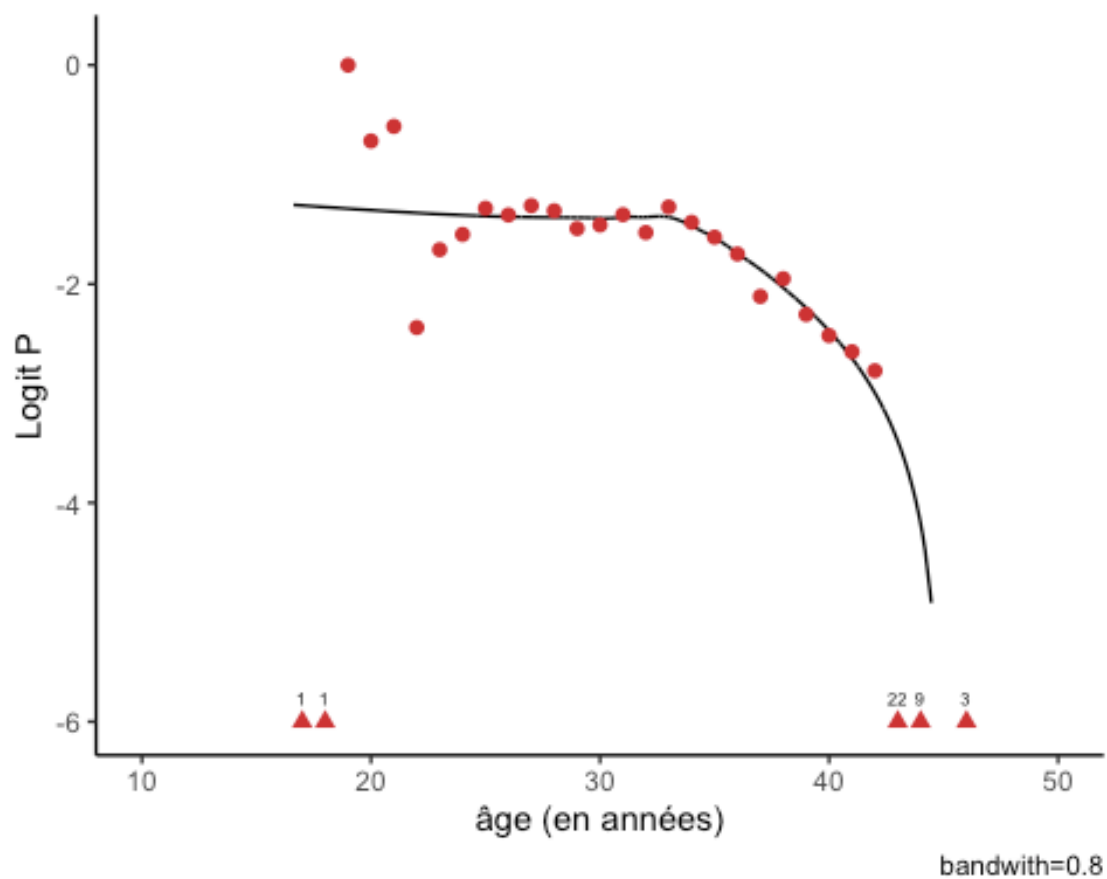
  p<-ggplot (data=cycles3) +
    geom_line (aes(x=age,y=lpred,color="Prévision (loess)",linewidth=0.5) +
    geom_point (data=tranche.age,
                aes(x=age, y=logitP,color="Logits observés",shape = factor(point)),size=2)+
    geom_text (data=tranche.age,
              aes(x=age, y=logitP,
                  label=ifelse(point %in% c(24,25), N,'')),size=2,
              hjust=0.4,vjust=0, nudge_x = -0.1, nudge_y = 0.15) +
    scale_y_continuous(breaks=seq(-6,2,2)) +
    labs(x="âge (en années)",y="Logit P",caption=paste0("bandwith=",span)) +
    scale_x_continuous(limits = c(10, 50)) +
    theme_classic() + theme(legend.position="none")+
    scale_color_manual(values = c("Prévision (loess)" = "black", "Logits observés" = "brown3"))

  print(p)
} # fin de la boucle

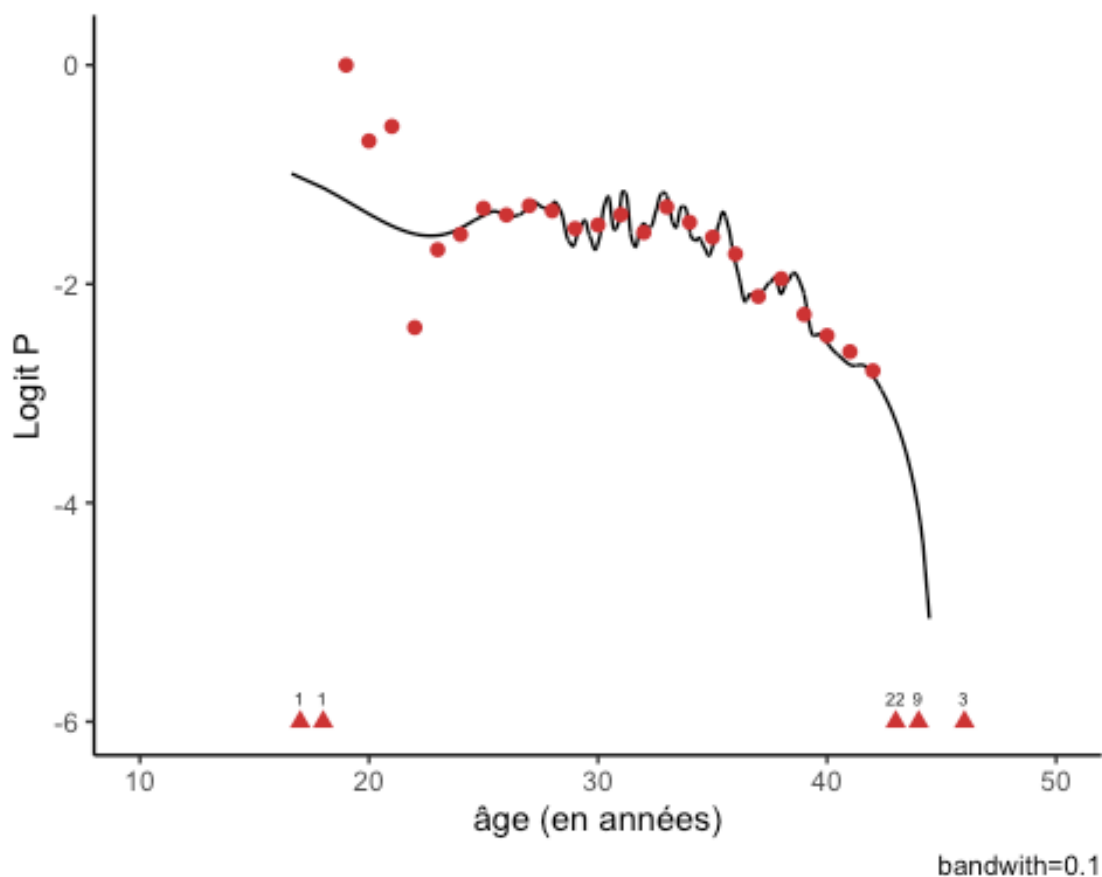
## Warning in log(predict(lfit)/(1 - predict(lfit))): NaNs produced

## Warning: Removed 3 rows containing missing values or values outside the scale range
## (`geom_line()`).

## Warning in log(predict(lfit)/(1 - predict(lfit))): NaNs produced
```



```
## Warning: Removed 3 rows containing missing values or values outside the scale range
## (`geom_line()`).
```



Remarques :

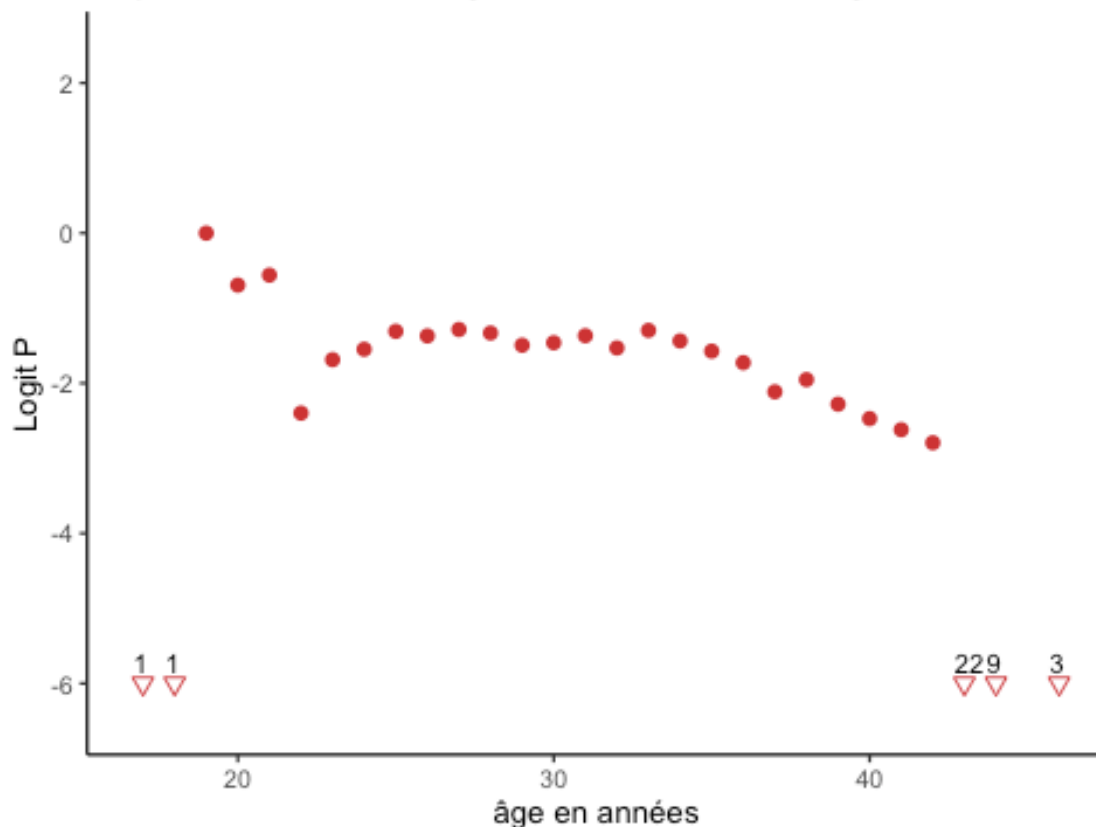
- NaN = not a number (càd valeur impossible).
- Le warning “NaNs produced” indique que la fonction log peut conduire à des valeurs “impossibles” (+ ou - infini).

## V - Figure 9 : Représentation des Logit P observés

```
fig9<-glm (acc~age,family="binomial",data=cycles3)
```

```
logit_crb(data=cycles3,
  reponse = "acc",
  x="age",
  x_class = "agea",
  mod = fig9,
  lab.x = "âge en années",
  obs = T,
  courbe=F,
  logit_min = -6,logit_max = +2,
  title="Représentation des Logit P observés selon l'âge de la femme
en années")
```

## Représentation des Logit P observés selon l'âge de la femme



## VI. Modélisation avec une fonction en escalier

### Modèle linéaire (figure 11)

```
mod.1<-glm (acc~age,family="binomial",data=cycles3)
summary (mod.1)

##
## Call:
## glm(formula = acc ~ age, family = "binomial", data = cycles3)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.484869   0.255138   1.900   0.0574 .
## age         -0.063720   0.007802  -8.167 3.16e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##    Null deviance: 5777.9  on 6399  degrees of freedom
## Residual deviance: 5710.1  on 6398  degrees of freedom
## AIC: 5714.1
##
## Number of Fisher Scoring iterations: 4

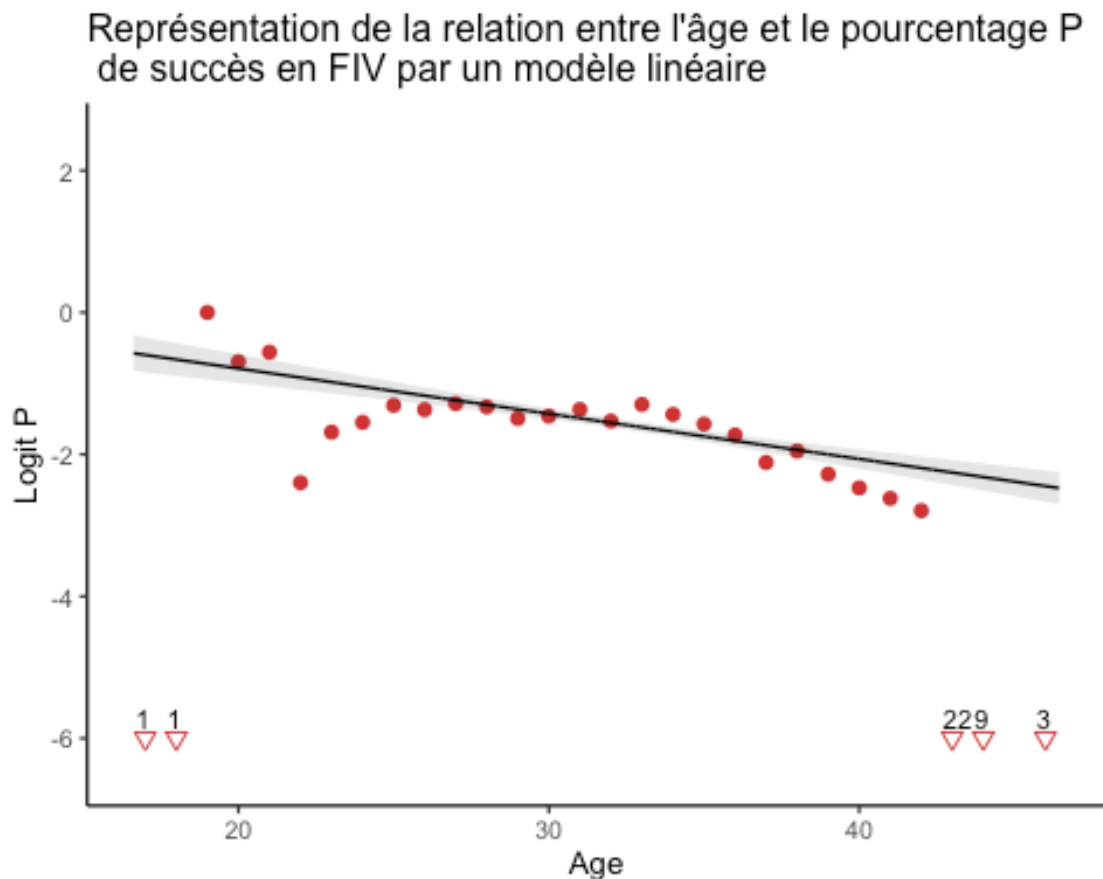
logit_crb (data=cycles3,
           reponse = "acc",
           x="age",
```

```

x_class = "agea",
mod = mod.1,
lab.x = "Age",
obs = T,
courbe = T,
logit_min = -6, logit_max = 2,
title="Représentation de la relation entre l'âge et le pourcentage
P \n de succès en FIV par un modèle linéaire")

## Scale for colour is already present.
## Adding another scale for colour, which will replace the existing scale.

```



### Coefficients des modèles A et B

```

mod.A <- glm (acc~agea,family="binomial",data=cycles3)
summary (mod.A)

##
## Call:
## glm(formula = acc ~ agea, family = "binomial", data = cycles3)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.502205   0.256110   1.961   0.0499 *
## agea        -0.064250   0.007833  -8.203 2.35e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 5777.9  on 6399  degrees of freedom

```

```
## Residual deviance: 5709.4 on 6398 degrees of freedom
## AIC: 5713.4
##
## Number of Fisher Scoring iterations: 4

cycles3$age2<-
  if_else(cycles3$agea<=19,18,
    if_else(cycles3$agea>=42,44,cycles3$agea))
cycles3$age2f<-factor(cycles3$age2)
mod.B <- glm (acc~relevel(age2f,ref="30"),family="binomial",data=cycles3)
summary (mod.B)

##
## Call:
## glm(formula = acc ~ relevel(age2f, ref = "30"), family = "binomial",
## data = cycles3)
##
## Coefficients:
##
## Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.46059 0.11449 -12.758 < 2e-16 ***
## relevel(age2f, ref = "30")18 0.36198 1.16036 0.312 0.755075
## relevel(age2f, ref = "30")20 0.76745 1.23008 0.624 0.532696
## relevel(age2f, ref = "30")21 0.90098 0.63715 1.414 0.157343
## relevel(age2f, ref = "30")22 -0.93730 1.05072 -0.892 0.372363
## relevel(age2f, ref = "30")23 -0.22581 0.50014 -0.451 0.651642
## relevel(age2f, ref = "30")24 -0.08697 0.36658 -0.237 0.812467
## relevel(age2f, ref = "30")25 0.15226 0.27680 0.550 0.582278
## relevel(age2f, ref = "30")26 0.09111 0.23434 0.389 0.697444
## relevel(age2f, ref = "30")27 0.17626 0.19790 0.891 0.373114
## relevel(age2f, ref = "30")28 0.12936 0.17654 0.733 0.463720
## relevel(age2f, ref = "30")29 -0.03277 0.17287 -0.190 0.849632
## relevel(age2f, ref = "30")31 0.09391 0.15903 0.590 0.554866
## relevel(age2f, ref = "30")32 -0.06949 0.16015 -0.434 0.664362
## relevel(age2f, ref = "30")33 0.16429 0.15345 1.071 0.284312
## relevel(age2f, ref = "30")34 0.02404 0.16083 0.149 0.881202
## relevel(age2f, ref = "30")35 -0.11203 0.16630 -0.674 0.500519
## relevel(age2f, ref = "30")36 -0.26557 0.17834 -1.489 0.136456
## relevel(age2f, ref = "30")37 -0.65350 0.20456 -3.195 0.001400 **
## relevel(age2f, ref = "30")38 -0.49151 0.19476 -2.524 0.011615 *
## relevel(age2f, ref = "30")39 -0.81865 0.23560 -3.475 0.000511 ***
## relevel(age2f, ref = "30")40 -1.01107 0.26492 -3.816 0.000135 ***
## relevel(age2f, ref = "30")41 -1.15879 0.33262 -3.484 0.000494 ***
## relevel(age2f, ref = "30")44 -1.63045 0.43278 -3.767 0.000165 ***
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 5777.9 on 6399 degrees of freedom
## Residual deviance: 5665.1 on 6376 degrees of freedom
## AIC: 5713.1
##
## Number of Fisher Scoring iterations: 5
```

## Comparaison des modèles 1 et A, et 1 et B pour voir qu'il n'y a pas de message d'erreur

- le résultat est anormal pour la comparaison entre 1 et A (d.f. = 0)
- le résultat n'a rien d'apparemment anormal pour la comparaison entre 1 et B

```
epiDisplay::lrtest(mod.1,mod.A)

## Likelihood ratio test for MLE method
## Chi-squared 0 d.f. = 0.6220586 , P value = 0

epiDisplay::lrtest(mod.1,mod.B)

## Likelihood ratio test for MLE method
## Chi-squared 22 d.f. = 44.98321 , P value = 0.002666888
```

## Comparaison des modèles A et B

```
epiDisplay::lrtest(mod.A,mod.B)

## Likelihood ratio test for MLE method
## Chi-squared 22 d.f. = 44.36115 , P value = 0.003195538
```

## Tableau 2

```
logistic.display(mod.B, simplified = T)

##
##               OR lower95ci upper95ci      Pr(>|Z|)
## relevel(age2f, ref = "30")18 1.4361702 0.14774191 13.9607301 0.7550752719
## relevel(age2f, ref = "30")20 2.1542553 0.19330697 24.0074949 0.5326956645
## relevel(age2f, ref = "30")21 2.4620061 0.70622499 8.5829219 0.1573428002
## relevel(age2f, ref = "30")22 0.3916828 0.04995265 3.0712165 0.3723627294
## relevel(age2f, ref = "30")23 0.7978723 0.29937097 2.1264596 0.6516415438
## relevel(age2f, ref = "30")24 0.9167044 0.44688133 1.8804700 0.8124668590
## relevel(age2f, ref = "30")25 1.1644623 0.67687535 2.0032825 0.5822778064
## relevel(age2f, ref = "30")26 1.0953841 0.69198336 1.7339524 0.6974444295
## relevel(age2f, ref = "30")27 1.1927515 0.80927338 1.7579427 0.3731137663
## relevel(age2f, ref = "30")28 1.1380971 0.80520663 1.6086121 0.4637203687
## relevel(age2f, ref = "30")29 0.9677578 0.68963936 1.3580360 0.8496321972
## relevel(age2f, ref = "30")31 1.0984569 0.80429265 1.5002097 0.5548658122
## relevel(age2f, ref = "30")32 0.9328695 0.68155161 1.2768593 0.6643621640
## relevel(age2f, ref = "30")33 1.1785612 0.87244270 1.5920891 0.2843115578
## relevel(age2f, ref = "30")34 1.0243273 0.74737107 1.4039162 0.8812015366
## relevel(age2f, ref = "30")35 0.8940160 0.64534302 1.2385112 0.5005193714
## relevel(age2f, ref = "30")36 0.7667688 0.54058104 1.0875973 0.1364563573
## relevel(age2f, ref = "30")37 0.5202226 0.34839203 0.7768019 0.0013999199
## relevel(age2f, ref = "30")38 0.6117021 0.41759630 0.8960316 0.0116153893
## relevel(age2f, ref = "30")39 0.4410286 0.27792410 0.6998539 0.0005112929
## relevel(age2f, ref = "30")40 0.3638298 0.21646947 0.6115048 0.0001353659
## relevel(age2f, ref = "30")41 0.3138650 0.16353590 0.6023830 0.0004943608
## relevel(age2f, ref = "30")44 0.1958414 0.08385370 0.4573901 0.0001649596
```

## Fonctions en escalier (Figure 12)

```
pred.1 <- predict(mod.1, type="link")
pred.A <- predict(mod.A, type="link")
pred.B <- predict(mod.B, type="link")

data.plot <- data.frame(age = cycles3$age,
                        agea = cycles3$agea,
```

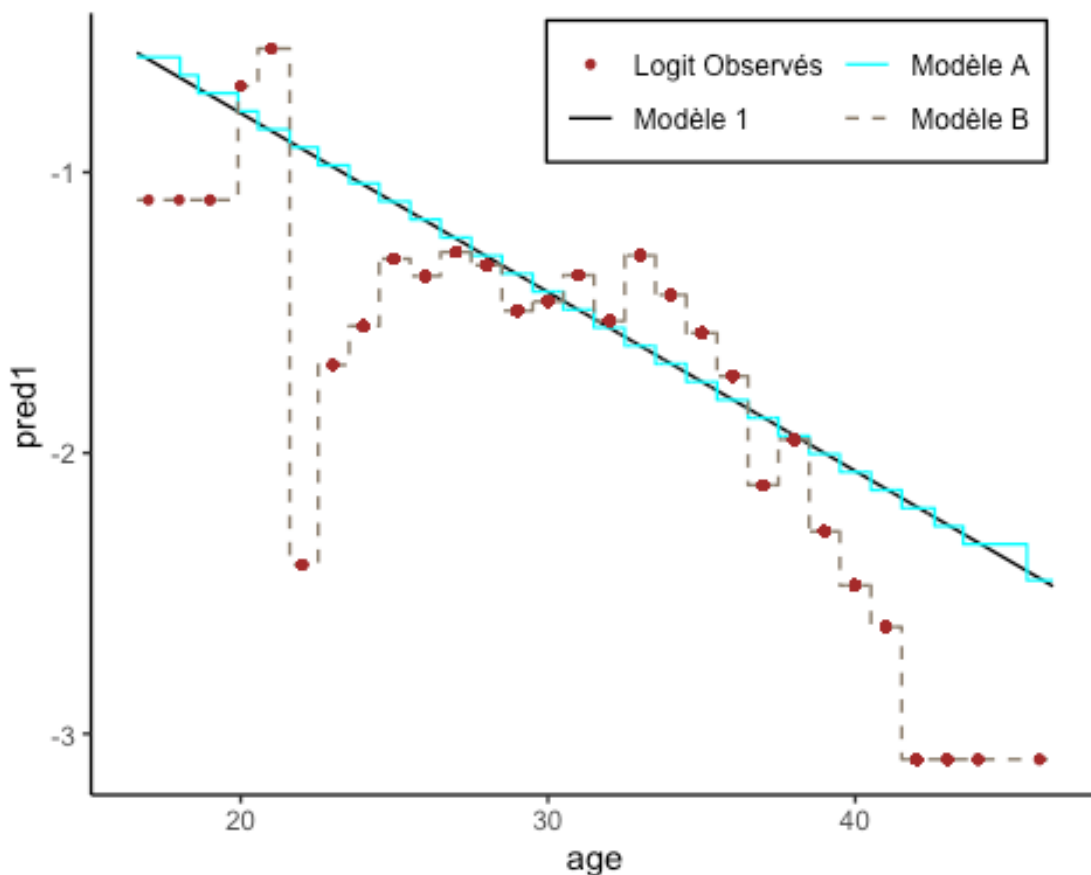
```

    pred1 = pred.1,
    predA = pred.A,
    predB = pred.B)

ggplot(data = data.plot) +
  geom_line(aes(x = age, y = pred1, color = "Modèle 1"), linewidth = 0.5) +
  geom_step(aes(x = age, y = predA, color = "Modèle A"), linewidth = 0.5) +
  geom_step(aes(x = age, y = predB, color = "Modèle B"), linewidth = 0.5, linetype="dashed") +
  geom_point(aes(x = agea, y = predB, color = "Logit Observés"), size = 1) +
  scale_color_manual(values = c("Modèle 1" = "black",
                                "Modèle A" = "cyan1",
                                "Modèle B" = "bisque4",
                                "Logit Observés" = "brown")) +

  labs(color = NULL) +
  theme_classic() +
  theme(legend.position = "inside", legend.position.inside = c(0.7, 0.9), # Position de la légende
        legend.box = "horizontal", # Boîte horizontale pour les éléments
        legend.background = element_rect(fill = "white", color = "black"), # Fond blanc avec bord noir
        legend.title = element_text(face = "bold"), # Titre en gras
        legend.text = element_text(size = 9)) + # Taille du texte
  guides(color = guide_legend(ncol = 2)) # 2 colonnes pour la légende

```



**Tableau 3 et Figure 13**

```

cycles3$agec <- cut(cycles3$agea, c(0, 19, 24, 29, 34, 39, 97), right = T, labels = c("1", "2", "3", "4", "5", "6"))

```



Remarque : Il est possible de mettre les complets : `labels = c("<=19 ans", "20-24 ans", "25-29 ans", "30-34 ans", "35-39 ans", ">=40 ans")` C'est plus clair mais la sortie devient peu lisible car les lignes sont coupées en 2.

```
cycles3$age3f<-factor(cycles3$agec)
mod.C <- glm (acc~relevel(age3f,ref="3"),family="binomial",data=cycles3)

logistic.display(mod.C, simplified = T)

##
##               OR lower95ci  upper95ci      Pr(>|Z|)
## relevel(age3f, ref = "3")1 1.3209366 0.1368023 12.7547051 8.098776e-01
## relevel(age3f, ref = "3")2 0.8853086 0.5403971  1.4503619 6.286118e-01
## relevel(age3f, ref = "3")4 0.9630135 0.8117757  1.1424276 6.654755e-01
## relevel(age3f, ref = "3")5 0.6153729 0.5074968  0.7461796 7.919051e-07
## relevel(age3f, ref = "3")6 0.2808290 0.1947047  0.4050490 1.074494e-11

pred.C <- predict(mod.C, type="link")

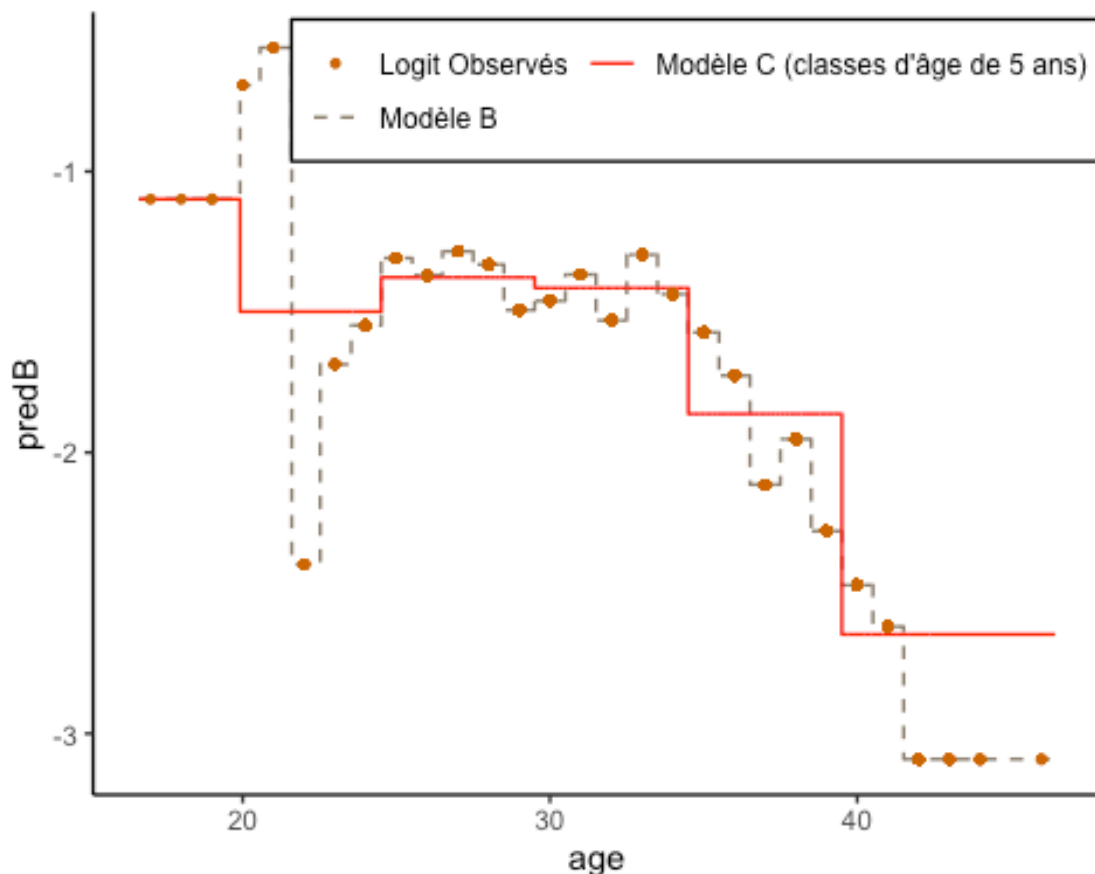
data.plot2 <- cbind(data.plot,predC=pred.C)

ggplot(data = data.plot2) +
  geom_step(aes(x = age, y = predB, color = "Modèle B"), linewidth = 0.5,linet
ype = "dashed") +
  geom_step(aes(x = age, y = predC, color = "Modèle C (classes d'âge de 5 ans)"
), size = 0.5) +
  geom_point(aes(x = agea, y = predB, color = "Logit Observés"), size = 1) +
  scale_color_manual(values = c("Modèle C (classes d'âge de 5 ans)" = "red",
                                "Modèle B" = "bisque4",
                                "Logit Observés" = "darkorange3")) +

  labs(color = NULL) +
  theme_classic() +
  theme(legend.position = "inside", legend.position.inside = c(0.6, 0.9), # Po
sition de la légende
        legend.box = "horizontal",      # Boîte horizontale pour les éléments
        legend.background = element_rect(fill = "white", color = "black"), # Fo
nd blanc avec bord noir
        legend.title = element_text(face = "bold"), # Titre en gras
        legend.text = element_text(size = 9)) + # Taille du texte
  guides(color = guide_legend(ncol = 2)) # 2 colonnes pour la légende

## Warning in geom_step(aes(x = age, y = predB, color = "Modèle B"), linewidth =
## 0.5, : Ignoring unknown parameters: `linewidth`

## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## ⓘ Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



## VII. Modélisation avec des polynômes (figure 14)

```
for (i in 2:7) {
cycles3[,paste0("a",i)]<-cycles3$age^i
}

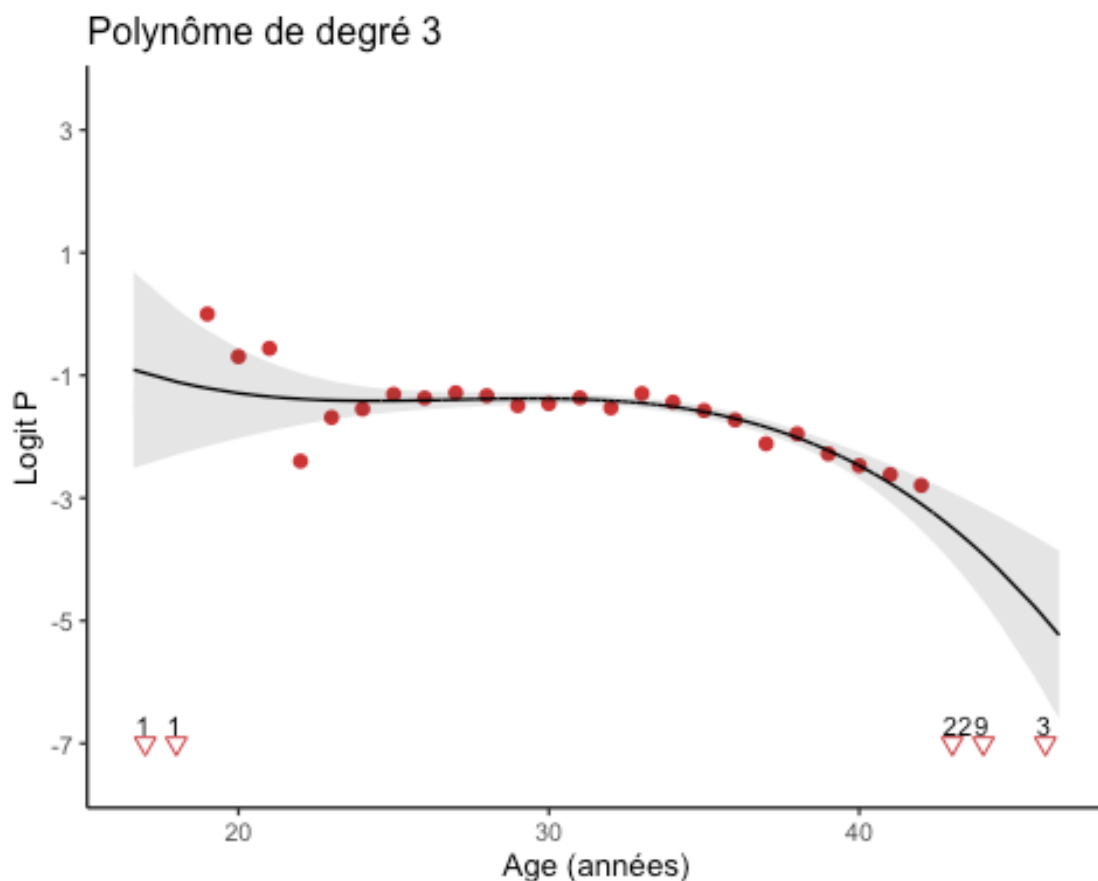
mod.p3<-glm(acc~age+a2+a3,family="binomial",data=cycles3)
mod.p5<-glm(acc~age+a2+a3+a4+a5,family="binomial",data=cycles3)
mod.p7<-glm(acc~age+a2+a3+a4+a5+a6+a7,family="binomial",data=cycles3)
summary(mod.p3)

##
## Call:
## glm(formula = acc ~ age + a2 + a3, family = "binomial", data = cycles3)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  8.9773460  7.9379602   1.131   0.2581
## age         -1.1771532  0.7609997  -1.547   0.1219
## a2           0.0441072  0.0240917   1.831   0.0671 .
## a3          -0.0005458  0.0002518  -2.168   0.0301 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 5777.9  on 6399  degrees of freedom
## Residual deviance: 5676.4  on 6396  degrees of freedom
## AIC: 5684.4
```

```
##
## Number of Fisher Scoring iterations: 5

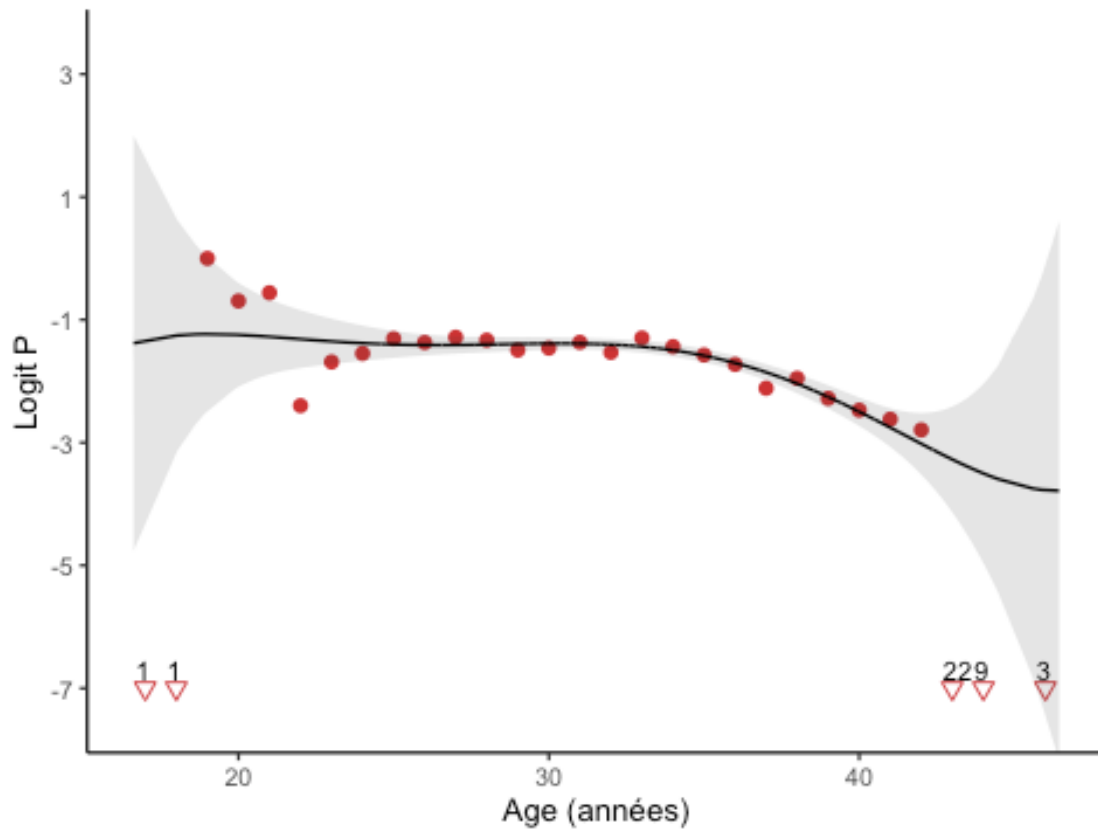
puiss<-c("3","5","7")
for (i in puiss) {
  model_name<-paste0("mod.p",i)
  model<-get(model_name)
  logit_crb(data=cycles3,
            reponse = "acc",
            x="age",
            x_class = "agea",
            mod = model,
            lab.x = "Age (années)",
            obs = T,
            courbe = T,
            logit_min = -7,logit_max = 3,
            title=paste("Polynôme de degré", i))
}

## Scale for colour is already present.
## Adding another scale for colour, which will replace the existing scale.
## Scale for colour is already present.
## Adding another scale for colour, which will replace the existing scale.
```

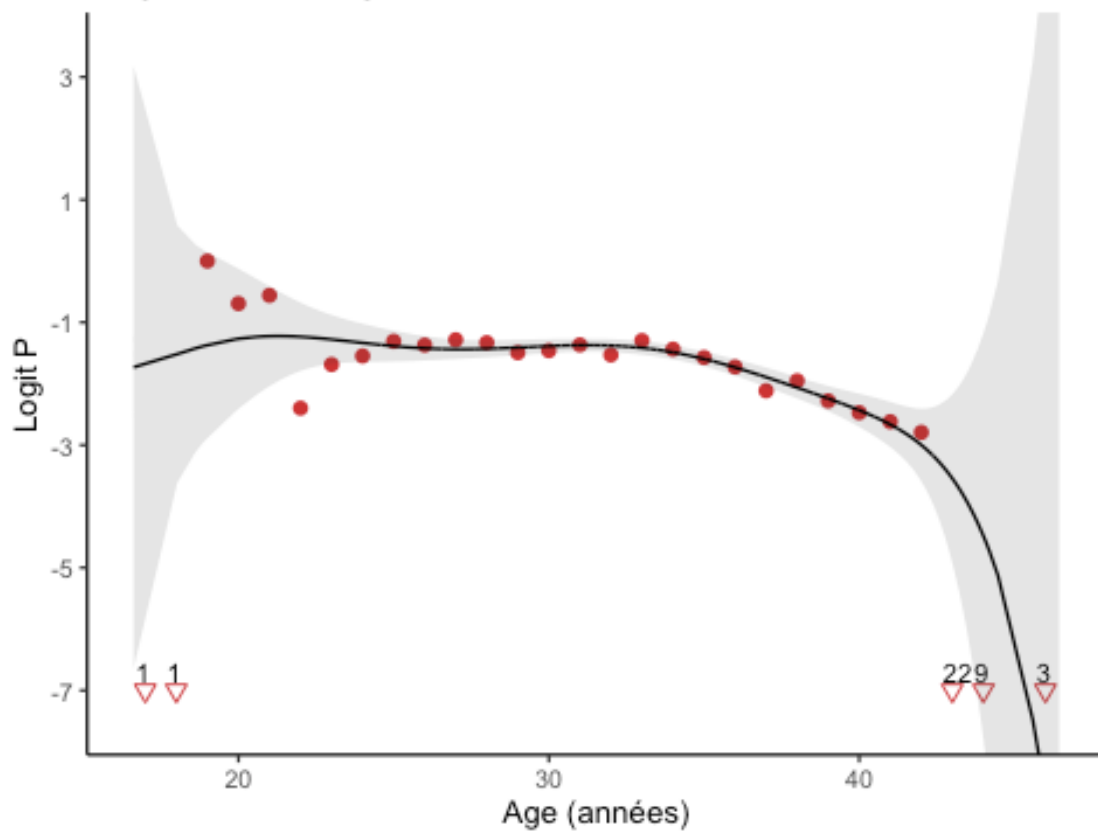


```
## Scale for colour is already present.
## Adding another scale for colour, which will replace the existing scale.
```

## Polynôme de degré 5



## Polynôme de degré 7



## VIII. Modélisation avec des polynômes fractionnaires

### Polynôme fractionnaire de degré 1 (figure 17)

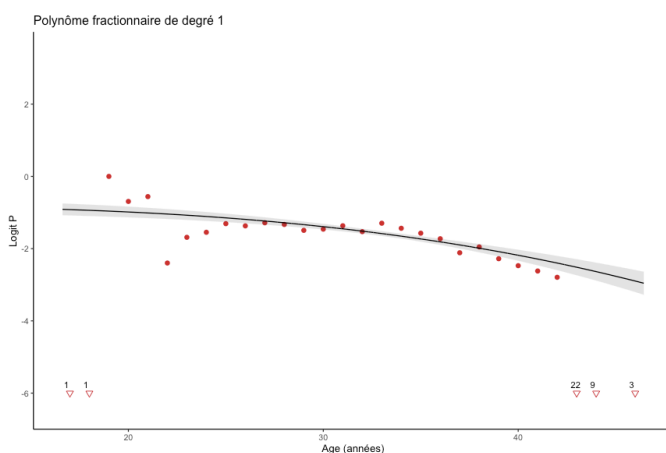
```
fp1 <- mfp(acc~fp(age, df=2, scale=FALSE), family=binomial(), data=cycles3, select=0.05, verbose=TRUE)
```

```
##
## Variable      Deviance      Power(s)
## -----
## Cycle 1
## age
##           5777.901
##           5710.067    1
##           5697.09    3
##
##
## Transformation
##      shift scale
## age      0      1
##
## Fractional polynomials
##      df.initial select alpha df.final power1 power2
## age           2   0.05  0.05           2       3     .
##
##
## Transformations of covariates:
##      formula
## age I(age^3)
##
##
## Deviance table:
##           Resid. Dev
## Null model      5777.901
## Linear model    5710.067
## Final model     5697.09
##
summary(fp1)
##
## Call:
## glm(formula = acc ~ I(age^3), family = binomial(), data = cycles3)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -8.143e-01  9.345e-02  -8.713   <2e-16 ***
## I(age^3)     -2.139e-05  2.453e-06  -8.720   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 5777.9  on 6399  degrees of freedom
## Residual deviance: 5697.1  on 6398  degrees of freedom
## AIC: 5701.1
```

```
##
## Number of Fisher Scoring iterations: 4

logit_crb (data=cycles3,
           reponse = "acc",
           x="age",
           x_class = "agea",
           mod = fp1,
           lab.x = "Age (années)",
           obs = T,
           courbe = T,
           logit_min = -6, logit_max = 3,
           title="Polynôme fractionnaire de degré 1")

## Scale for colour is already present.
## Adding another scale for colour, which will replace the existing scale.
```



### Remarques

- L'option `scale=F` permet que la variable `age` ne soit pas transformée comme c'est fait dans le livre. Sinon, R divise `age` par 100 (et non par 10). Cela ne change rien à la courbe, mais modifie les coefficients du modèle
- L'option `df=2` permet de se limiter aux polynômes fractionnaires de degré 1
- L'option `verbose=TRUE` permet de suivre (un peu ...) la sélection des puissances en donnant le meilleur polynôme FP1 et le polynôme FP1 retenu

### Polynôme fractionnaire de degré 2 (Figure 18)

```
fp2 <- mfp(acc~fp(age, df=4, scale=FALSE), family=binomial(), data=cycles3, select=0.05, verbose=TRUE)
```

```
##
## Variable      Deviance      Power(s)
## -----
## Cycle 1
## age
##          5777.901
##          5710.067    1
##          5697.09    3
##          5677.977    3 3
##
##
## Transformation
##      shift scale
## age      0      1
```

```
##
## Fractional polynomials
##      df.initial select alpha df.final power1 power2
## age          4   0.05  0.05          4       3       3
##
##
## Transformations of covariates:
##              formula
## age I(age^3)+I(age^3*log(age))
##
##
## Deviance table:
##              Resid. Dev
## Null model      5777.901
## Linear model    5710.067
## Final model     5677.977

summary(fp2)

##
## Call:
## glm(formula = acc ~ I(age^3) + I(age^3 * log(age)), family = binomial(),
##      data = cycles3)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -2.564e+00  4.265e-01  -6.011 1.84e-09 ***
## I(age^3)        5.516e-04  1.362e-04   4.051 5.10e-05 ***
## I(age^3 * log(age)) -1.490e-04  3.542e-05  -4.206 2.59e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

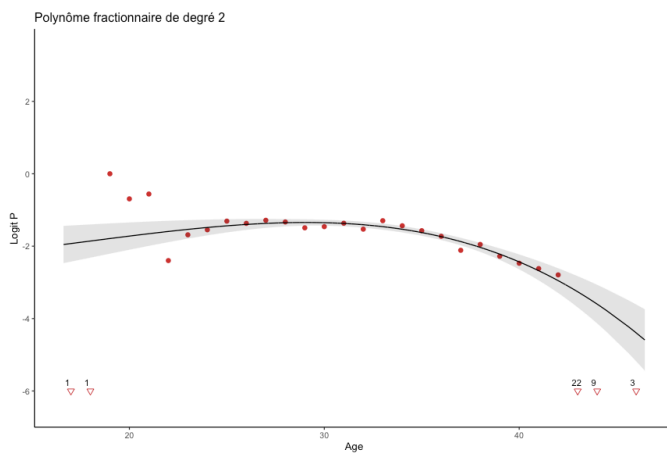
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 5777.9  on 6399  degrees of freedom
## Residual deviance: 5678.0  on 6397  degrees of freedom
## AIC: 5684
##
## Number of Fisher Scoring iterations: 5
```

### Remarque

Les coefficients du polynôme fractionnaire ne sont pas les mêmes que ceux du livre. C'est dû au fait que R ne divise pas la variable age par 10. Ici, à cause de la présence de log, cela change tous les coefficients (celui de  $x^3 \ln(X)$  n'est "que" divisé par 1000)

```
logit_crb (data=cycles3,
           reponse = "acc",
           x="age",
           x_class = "agea",
           mod = fp2,
           lab.x = "Age",
           obs = T, courbe=T,
           logit_min = -6, logit_max = 3,
           title="Polynôme fractionnaire de degré 2")

## Scale for colour is already present.
## Adding another scale for colour, which will replace the existing scale.
```



### Remarque

L'option `df=4` (option par défaut) limite le degré des PF à 2. Attention que si un PF de degré 1 n'est pas moins bon que le meilleur PF de degré 2, c'est lui qui sera retenu et pas celui de degré 2. Si on veut absolument un PF de degré 2, il faut changer les seuils de test pour sélectionner le meilleur PF, par exemple en prenant 0.9 : `fp(age10,df=4,scale=FALSE, select=0.9)`.

## Polynôme fractionnaire de degré 4

N'existe pas dans R, mais ce n'est pas très utile !

## Modélisation simultanée de plusieurs variables quantitatives, procédure mfp (Tableau 6)

```
mfp <- mfp(acc ~ fp(agea,df=4)+fp(ovo,df=4)+fp(emb, df=4), selec=0.05, alpha=0.05, verbose=T, family=binomial(),rescale=T,data=cycles3)
```

```
##
## Variable      Deviance      Power(s)
## -----
## Cycle 1
## emb
##          5665.052
##          5576.424      1
##          5386.574     -2
##          5386.182     -2 3
##
## agea
##          5432.668
##          5386.574      1
##          5376.162      3
##          5358.803      3 3
##
## ovo
##          5360.29
##          5358.803      1
##          5357.876      3
##          5356.444     -0.5 0
##
## Cycle 2
## emb
##          5660.015
##          5548.133      1
##          5360.29     -2
```



```
##           5360.141    -2 -0.5
##
## agea
##           5432.741
##           5388.179    1
##           5378.125    3
##           5360.29    3 3
##
##
## Transformation
##      shift scale
## emb      1    10
## agea      0   100
## ovo      1    10
##
## Fractional polynomials
##      df.initial select alpha df.final power1 power2
## emb          4   0.05  0.05         2     -2     .
## agea          4   0.05  0.05         4      3     3
## ovo          4   0.05  0.05         0      .     .
##
##
## Transformations of covariates:
##                                     formula
## agea I((agea/100)^3)+I((agea/100)^3*log((agea/100)))
## ovo                                     <NA>
## emb                                     I(((emb+1)/10)^-2)
##
##
## Deviance table:
##      Resid. Dev
## Null model    5759.022
## Linear model  5576.424
## Final model   5360.29

summary(mfp)

##
## Call:
## glm(formula = acc ~ I(((emb + 1)/10)^-2) + I((agea/100)^3) +
##      I((agea/100)^3 * log((agea/100))), family = binomial(), data = cycles3)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -2.129e+00  4.364e-01  -4.880 1.06e-06 ***
## I(((emb + 1)/10)^-2) -5.400e-02  5.319e-03 -10.152 < 2e-16 ***
## I((agea/100)^3)   -1.292e+02  2.753e+01  -4.691 2.71e-06 ***
## I((agea/100)^3 * log((agea/100))) -1.464e+02  3.603e+01  -4.064 4.82e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 5760.5  on 6369  degrees of freedom
## Residual deviance: 5361.1  on 6366  degrees of freedom
## (30 observations deleted due to missingness)
## AIC: 5369.1
```

```
##
## Number of Fisher Scoring iterations: 7
```

### Remarque

Attention, l'option `scale=T` est appliquée avec des changements d'échelle pour les variables différents de ceux de Stata qui sont donnés dans le Tableau 6 du livre. Cela a des conséquences sur la constance, mais aussi sur le coefficient de  $\text{age}^3$  à cause de la présence de  $\text{age}^3 \times \ln(\text{age})$ .

## IX. Modélisation avec des fonctions splines

### IX.4 Splines linéaires

Plusieurs fonctions de R permettent de construire des splines linéaires (elles sont installées en même temps que R) :

- `lspline()` où les noeuds doivent être fixés manuellement
- `qlspline()`, où les noeuds divisent l'étendue de R en intervalles de fréquences égales (par exemple quartile pour 3 noeuds)
- `elspline()` où les noeuds divisent l'étendue de R en intervalles de largeurs égales

Ces 3 fonctions ont l'option "marginal" :  
`marginal=FALSE` : les coefficients sont les pentes de chaque morceaux de droite  
`marginal=TRUE` : les coefficients sont les variations de pente entre deux morceaux de droite successifs

#### Spline linéaire à 1 noeud (Figure 25)

```
cycles3$sp.lin1<-qlspline(cycles3$age,2,marginal=T) # construction des fonctions splines
quantile(cycles3$age,probs=seq(0,1,0.5),type=7) # pour avoir l'emplacement du noeud
```

```
##      0%      50%     100%
## 16.62150 33.05688 46.44173
```

```
m1in1 <-glm(acc ~ sp.lin1, data=cycles3, family = "binomial") # estimation des paramètres du modèle logistique
summary(m1in1)
```

```
##
## Call:
## glm(formula = acc ~ sp.lin1, family = "binomial", data = cycles3)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.672471   0.463314  -3.610 0.000306 ***
## sp.lin11      0.009809   0.015202   0.645 0.518758
## sp.lin12     -0.164250   0.028655  -5.732 9.92e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 5777.9  on 6399  degrees of freedom
## Residual deviance: 5675.7  on 6397  degrees of freedom
## AIC: 5681.7
```

```
##
## Number of Fisher Scoring iterations: 4
```

Figure 25a

```
ggplot(data = cycles3) +
  geom_line(aes(age, sp.lin1[, "1"], color = "X")) +
  geom_line(aes(age, sp.lin1[, "2"], color = "(x-33.1)+")) +
  labs(x = "âge", y = "") +
  scale_color_manual(values = c("X" = "black", "(x-33.1)+" = "red")) +
  theme_classic() +
  theme(legend.position = "inside",
        legend.position.inside = c(0.1, 0.9),
        legend.box = "horizontal") +
  labs(color = NULL)
```

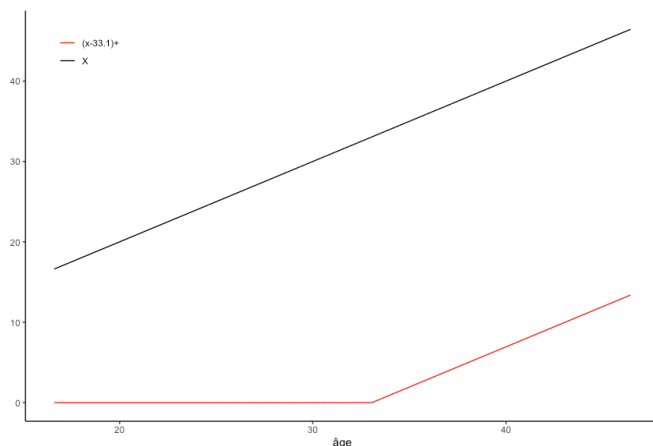
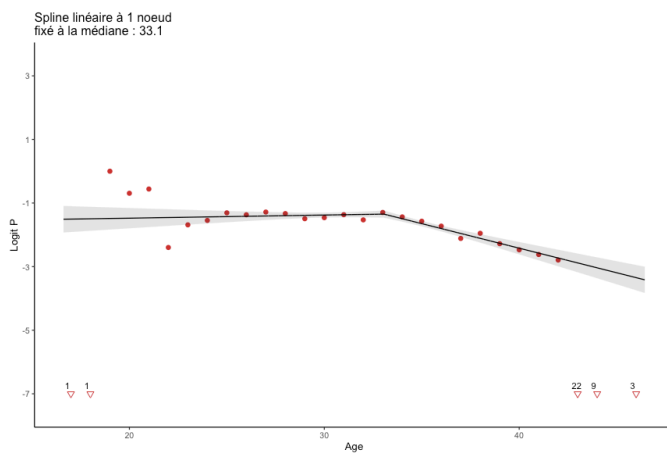


Figure 25b

```
logit_crb (data = cycles3,
  reponse = "acc",
  x = "age",
  x_class = "agea",
  mod = mlin1,
  lab.x = "Age",
  obs = T,
  courbe = T,
  logit_min = -7,
  logit_max = 3,
  title = "Spline linéaire à 1 noeud \nfixé à la médiane : 33.1"
)
```

```
## Scale for colour is already present.
## Adding another scale for colour, which will replace the existing scale.
```



### Spline linéaire à 4 noeuds (5 intervalles) à des percentiles égaux (figure 26)

Le calcul est fait ci-dessous avec la fonction `qlspline`. On peut obtenir le même résultat avec la fonction `lspline` en calculant explicitement les noeuds aux quantiles 25%, 50% et 75% (`d5`) et intégrant les valeurs dans la fonction `lspline` (avec `d5`). Les commandes sont alors :

```
d4 <- quantile(cycles3$age, probs = c(0.20, 0.40, 0.60, 0.80), names = F)
sp.lin4 <- lspline(cycles3$age, d4, marginal=TRUE)
```

```
m1in4 <- glm(cycles3$acc ~ sp.lin4, family = "binomial")
```

Avec `qlspline`, cela donne (attention, c'est le nombre d'intervalle, pas de noeuds, qui figure dans la commande) :

```
cycles3$sp.lin4 <- qlspline(cycles3$age, 5, marginal=T)
m1in4 <- glm(acc ~ sp.lin4, data=cycles3, family = "binomial")
d4 <- attr(qlspline(cycles3$age, 5, marginal=T), "knots") # position des noeuds
d4
```

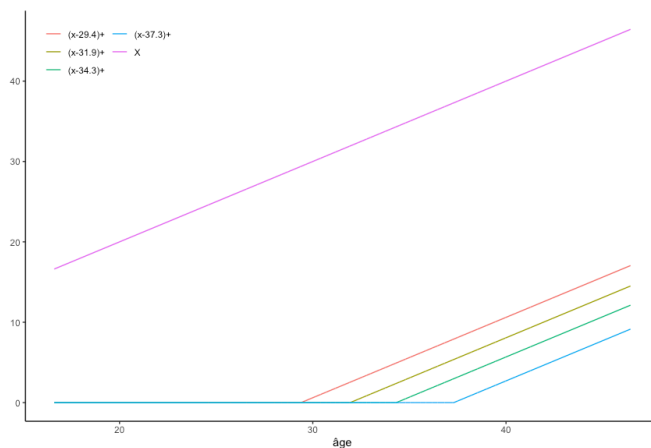
```
##      20%      40%      60%      80%
## 29.39321 31.93259 34.32147 37.29823
```

```
summary(m1in4)
```

```
##
## Call:
## glm(formula = acc ~ sp.lin4, family = "binomial", data = cycles3)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.82321    0.89585  -0.919   0.358
## sp.lin41    -0.02136    0.03238  -0.660   0.510
## sp.lin42     0.05549    0.07635   0.727   0.467
## sp.lin43    -0.07140    0.10383  -0.688   0.492
## sp.lin44    -0.13548    0.10501  -1.290   0.197
## sp.lin45    -0.01577    0.10143  -0.156   0.876
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 5777.9  on 6399  degrees of freedom
## Residual deviance: 5675.1  on 6394  degrees of freedom
## AIC: 5687.1
##
## Number of Fisher Scoring iterations: 5
```

Figure 26a

```
ggplot(data=cycles3) +
  geom_line(aes(age,sp.lin4[, "1"], color = "X"))+
  geom_line(aes(age,sp.lin4[, "2"], color = "(x-29.4)+"))+
  geom_line(aes(age,sp.lin4[, "3"], color = "(x-31.9)+"))+
  geom_line(aes(age,sp.lin4[, "4"], color = "(x-34.3)+"))+
  geom_line(aes(age,sp.lin4[, "5"], color = "(x-37.3)+")) +
  labs(x="âge",y="") +
  theme_classic() +
  theme(legend.position = "inside",
        legend.position.inside = c(0.13, 0.9),
        legend.box = "horizontal") +
  guides(color = guide_legend(ncol = 2)) +
  labs(color = NULL)
```



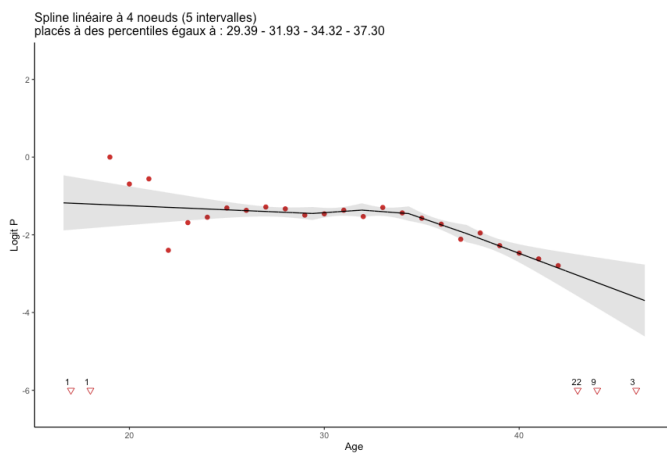
```
geom_line(aes(age, sp.lin1[, "1"], color =
  "X")) + geom_line(aes(age, sp.lin1[, "2"], color = "(x-33.1)+")) + labs(x = "âge", y = "") +
scale_color_manual(values = c("X" = "black", "(x-33.1)+ = "red")) + theme_classic() +
theme(legend.position = "inside", legend.position.inside = c(0.1, 0.9), legend.box =
  "horizontal") + labs(color = NULL)
```

Figure 26b

```
# commande pour construire un titre incluant les positions des noeuds
titre<- paste(formatC(d4,format="f", digits=2),collapse = " - ")
```

```
logit_crb (data=cycles3,
           reponse = "acc",
           x="age",
           x_class = "agea",
           mod = mlin4,
           lab.x = "Age",
           obs = T,
           courbe = T,
           logit_min = -6,logit_max = 2,
           title=paste0("Spline linéaire à 4 noeuds (5 intervalles) \nplacés
à des percentiles égaux à : ",titre))
```

```
## Scale for colour is already present.
## Adding another scale for colour, which will replace the existing scale.
```



### Réduction du nombre de noeuds avec une procédure pas à pas (tableau 7)

On part d'une fonction spline linéaire à 11 noeuds (12 intervalles) à des percentiles égaux suivi de stepwise.

Ne pas oublier l'option `marginal=TRUE` qui est utile pour le stepwise qui suit

```
sp.lin11 <- qlspline(cycles3$age,12, marginal=TRUE)

mlin11 <- glm(cycles3$acc ~ sp.lin11, family="binomial")
attr(qlspline(cycles3$age,11, marginal=T),"knots")

## 9.090909% 18.18182% 27.27273% 36.36364% 45.45455% 54.54545% 63.63636% 72.727
27%
## 27.48758 29.09638 30.35853 31.50209 32.54942 33.62374 34.81433 36.05
579
## 81.81818% 90.90909%
## 37.56146 39.36880

summary(mlin11)

##
## Call:
## glm(formula = cycles3$acc ~ sp.lin11, family = "binomial")
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.23771    1.57215  -1.423   0.1546
## sp.lin111    0.03546    0.06084   0.583   0.5600
## sp.lin112   -0.19154    0.17308  -1.107   0.2685
## sp.lin113    0.18094    0.28337   0.639   0.5231
## sp.lin114    0.17441    0.34980   0.499   0.6181
## sp.lin115   -0.47119    0.39563  -1.191   0.2337
## sp.lin116    0.67640    0.42390   1.596   0.1106
## sp.lin117   -0.84103    0.40750  -2.064   0.0390 *
## sp.lin118    0.60823    0.39609   1.536   0.1246
## sp.lin119   -0.68446    0.38500  -1.778   0.0754 .
## sp.lin1110   0.60465    0.37148   1.628   0.1036
## sp.lin1111  -0.32178    0.32176  -1.000   0.3173
## sp.lin1112  -0.04744    0.29021  -0.163   0.8701
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
```

```
##
##      Null deviance: 5777.9  on 6399  degrees of freedom
## Residual deviance: 5666.0  on 6387  degrees of freedom
## AIC: 5692
##
## Number of Fisher Scoring iterations: 5

sp.lin11df <- as.data.frame(sp.lin11)
cycles3.sw <- cbind(cycles3$acc, sp.lin11df)
colnames(cycles3.sw) <- c("acc", unlist(map(1:12, ~ paste0("sp", .x))))
mlq <- glm(acc ~ ., data=cycles3.sw, family = "binomial")
```

Remarques :

- la fonction `qlspline()` crée 12 variables nommées de `sp.lin111` à `sp.lin1112` (puis `sp1` à `sp12`). La première est X (ici age). Les résultats du livre viennent de Stata qui crée 11 variables qu'il faut ajouter à X dans la régression logistique. Cela revient au même, mais la numérotation des variables est décalée. La variable `sp7` de R est donc la même que la variable `vsp_1_6` de Stata.
- La construction de `mlq` est nécessaire pour pouvoir utiliser ensuite la fonction `step` (stepwise) qui n'admet pas directement comme argument `"qlspline(cycles3$age, 12, marginal=TRUE)"`.
- Dans la commande qui suit, l'option `scope` permet de "contrôler" le stepwise en déterminant le modèle qui a le moins de variables (lower) et celui qui en a le plus (upper). Ici, cela permet que `sp1` ne soit pas éliminé.
- L'option `trace` ne semble servir à rien. Je l'ai remplacée par `include=F` dans l'en-tête du Chunk de Markdown : `{r, eval=TRUE, include=F, echo=T}`. Si on veut avoir toutes les étapes du stepwise, il faut mettre `include=T`.

```
summary(mlqs)

##
## Call:
## glm(formula = acc ~ sp1 + sp7, family = "binomial", data = cycles3.sw)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.672471    0.463314  -3.610 0.000306 ***
## sp1          0.009809    0.015202   0.645 0.518758
## sp7         -0.164250    0.028655  -5.732 9.92e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 5777.9  on 6399  degrees of freedom
## Residual deviance: 5675.7  on 6397  degrees of freedom
## AIC: 5681.7
##
## Number of Fisher Scoring iterations: 4

attr(sp.lin11, "knots")

## 8.333333% 16.66667%      25% 33.33333% 41.66667%      50% 58.33333% 66.666
## 67%
## 27.21419 28.82131 30.07649 31.13005 32.11249 33.05688 34.11186 35.19
## 394
##      75% 83.33333% 91.66667%
## 36.39792 37.80898 39.55561
```

Il ne reste donc que la fonction spline `sp7` (en plus de `sp1`) ce qui correspond au total à 2 droites séparées par le noeuds n°6. La liste des noeuds indique que c'est celui du 50ème percentile (médiane), soit 33.06.

## IX.5 Splines cubiques (figure 27)

La fonction de R pour des splines cubiques non restreints est `bs()` (B-Splines) du package `splines`

```
d3 <- quantile(cycles3$age, probs=c(0.25,0.50,0.75), names=F)
# d3 30.08 - 33.06 - 36.40

mbs3 <- glm(acc~bs(age, knots=d3), family=binomial(), data=cycles3)
summary(mbs3)

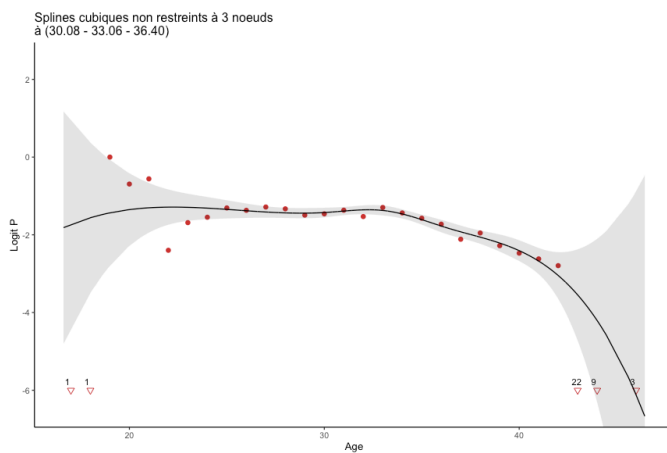
##
## Call:
## glm(formula = acc ~ bs(age, knots = d3), family = binomial(),
##      data = cycles3)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -1.81477    1.52401  -1.191   0.234
## bs(age, knots = d3)1  0.99746    2.08600   0.478   0.633
## bs(age, knots = d3)2  0.08518    1.41247   0.060   0.952
## bs(age, knots = d3)3  0.57865    1.56383   0.370   0.711
## bs(age, knots = d3)4 -0.53613    1.49558  -0.358   0.720
## bs(age, knots = d3)5 -0.51748    1.87667  -0.276   0.783
## bs(age, knots = d3)6 -4.85170    3.39310  -1.430   0.153
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 5777.9  on 6399  degrees of freedom
## Residual deviance: 5673.6  on 6393  degrees of freedom
## AIC: 5687.6
##
## Number of Fisher Scoring iterations: 6

# commande pour construire un titre incluant les positions des noeuds
titre<- paste(formatC(d3,format="f", digits=2),collapse =" - ")

logit_crb (data=cycles3,
           reponse = "acc",
           x="age",
           x_class = "agea",
           mod = mbs3,
           lab.x = "Age",
           obs = T,
           courbe = T,
           logit_min = -6, logit_max = 2,
           title=paste0("Splines cubiques non restreints à 3 noeuds \nà (", titre,")"))

## Scale for colour is already present.
## Adding another scale for colour, which will replace the existing scale.
```





## IX.6 Splines cubiques restreints à 3 noeuds (figure 28)

La fonction `ns()` (natural splines), qui est inclus dans R lui-même, construit effectivement des splines cubiques restreints, mais avec un ensemble de fonctions splines différent de celui du livre.

En particulier, la première fonction spline n'est pas une droite.

Pour retrouver les fonctions splines du cours, il faut utiliser la fonction `rcspline.eval()` du package `Hmisc`. Les noeuds sont placés par défaut aux percentiles recommandés par Harrell.

```
rcs.3 <- rcspline.eval(cycles3$age, nk=3, inclx=T)
mracs3 <- glm(acc ~ rcs.3, family=binomial(), data=cycles3)
summary(mracs3)

##
## Call:
## glm(formula = acc ~ rcs.3, family = binomial(), data = cycles3)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.99532    0.52366  -3.810 0.000139 ***
## rcs.3x       0.02198    0.01750   1.256 0.209222
## rcs.3       -0.13963    0.02529  -5.521 3.37e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 5777.9  on 6399  degrees of freedom
## Residual deviance: 5677.5  on 6397  degrees of freedom
## AIC: 5683.5
##
## Number of Fisher Scoring iterations: 5
```

### Courbe 28 a

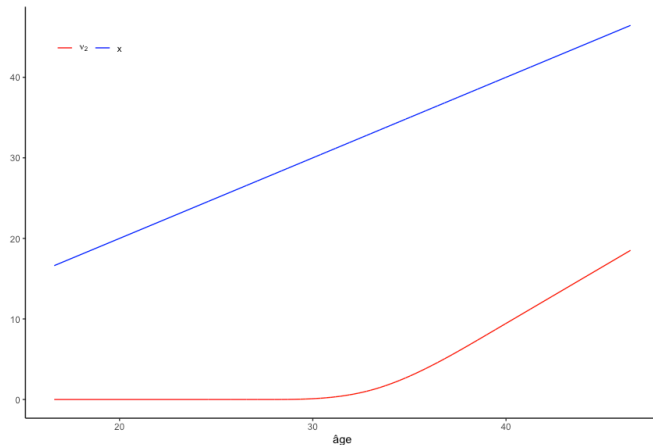
```
df.rcs3 <- as.data.frame(rcs.3)
courbe28a <- cbind(cycles3$age, df.rcs3)

ggplot(data=courbe28a) +
  geom_line(aes(cycles3$age, x, color = "x")) +
  geom_line(aes(cycles3$age, V2, color = "V2")) +
  labs(x="âge", y="") +
  theme(legend.position="top") +
```

```

theme_classic() +
  theme(legend.position = "inside",
        legend.position.inside = c(0.1, 0.9),
        legend.box = "horizontal") +
  guides(color = guide_legend(ncol = 2)) +
  labs(color = NULL) +
  scale_color_manual(
    values = c("x" = "blue", "V2" = "red"), # Vous pouvez ajuster les couleurs
    labels = c(expression(nu[2], "x")) # Remplacement de "V2" par "nu2" dans la
    Légende
  )

```



### Courbe 28b

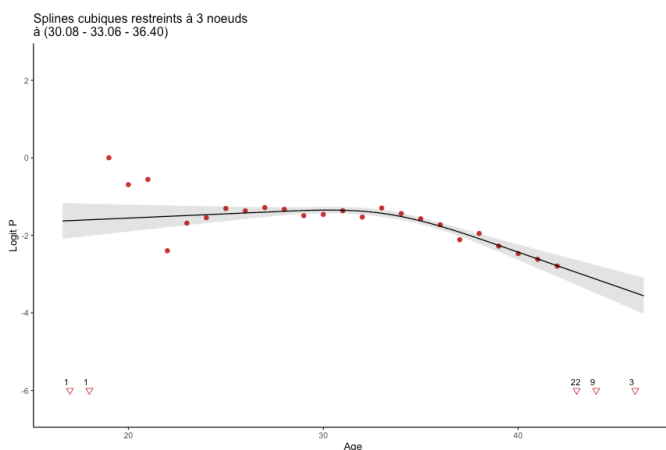
```

logit_crb (data=cycles3,
           reponse = "acc",
           x="age",
           x_class = "agea",
           mod = mrCS3,
           lab.x = "Age",
           obs = T, courbe = T,
           logit_min = -6, logit_max = 2,
           title="Splines cubiques restreints à 3 noeuds \nà (30.08 - 33.06 -
           36.40)")

```

## Scale for colour is already present.

## Adding another scale for colour, which will replace the existing scale.



## Stratégie de choix du modèle avec des fonctions splines

Les fonctions `uvs` et `mvs` de Stata n'existent pas dans R

### X.3. Présentation quantitative des résultats dans un tableau (tableau 11)

#### OR non modélisés (colonne 1)

#### Avec un polynôme fractionnaire de degré 2 (colonne 2) : fonction `ORcl.fp`

```
fp2 <- mfp(acc~fp(age,df=4,scale=FALSE),family=binomial(),data=cycles3,select=0.05,verbose=TRUE)
```

```
##
## Variable      Deviance      Power(s)
## -----
## Cycle 1
## age
##           5777.901
##           5710.067    1
##           5697.09    3
##           5677.977    3 3
##
##
## Transformation
##      shift scale
## age      0      1
##
## Fractional polynomials
##      df.initial select alpha df.final power1 power2
## age           4   0.05  0.05           4         3         3
##
##
## Transformations of covariates:
##                               formula
## age I(age^3)+I(age^3*log(age))
##
##
## Deviance table:
##              Resid. Dev
## Null model      5777.901
## Linear model    5710.067
## Final model     5677.977

ORcl_pf(x="age",res.mfp=fp2,ref=27,cl=c(17,22,32,37,42))

## Modélisation : mfp(formula = acc ~ fp(age, df = 4, scale = FALSE), data = cycles3,
##                    family = binomial(), select = 0.05, verbose = TRUE)
## Puissance(s) de la variable age transformée en polynômes fractionnaires : 3
##
## OR et IC pour la variable age
##
## ref = 27  classe = 17  OR = 0.57  95% IC : [ 0.38 - 0.86 ]
## ref = 27  classe = 22  OR = 0.80  95% IC : [ 0.66 - 0.98 ]
```

```
## ref = 27 classe = 32 OR = 0.96 95% IC : [ 0.85 - 1.10 ]
## ref = 27 classe = 37 OR = 0.61 95% IC : [ 0.51 - 0.72 ]
## ref = 27 classe = 42 OR = 0.21 95% IC : [ 0.15 - 0.29 ]
```

### Avec une fonction splines cubiques restreints à 3 noeuds (colonne 3) : fonction ORcl\_sp

```
rcl_sp <- rcspline.eval(cycles3$age, nk = 3, inclx = TRUE) # construction des splines
mrcs <- glm(acc ~ rcl_sp+ovo, family = binomial(), data = cycles3) # Modèle Logistique
ORcl_sp(x="age", spl=rcl_sp, reglog=mrcs, ref=27, cl=c(17,22,32,37,42))

## Modèle logistique où age est remplacé par 2 fonctions splines ( 3 noeuds)
## OR et IC pour la variable age ajustée sur les autres variables du modèle
##
## ref = 27 classe = 17 OR = 0.73 95% IC : [ 0.52 - 1.03 ]
## ref = 27 classe = 22 OR = 0.85 95% IC : [ 0.72 - 1.02 ]
## ref = 27 classe = 32 OR = 1.07 95% IC : [ 0.93 - 1.24 ]
## ref = 27 classe = 37 OR = 0.66 95% IC : [ 0.55 - 0.78 ]
## ref = 27 classe = 42 OR = 0.29 95% IC : [ 0.22 - 0.39 ]
```

# Chapitre 5

## IV.3. Colinéarité

```
table (geu$ctub, geu$ageu, dnn=c("ctub","ageu"))

##      ageu
## ctub    0    1
##    0 1540    1
##    1   70  114

m1<- glm(ct~ctub,data=geu,family="binomial")
logistic.display(m1)

##
## Logistic regression predicting 0:acc 1:GEU
##
##                OR(95%CI)          P(Wald's test) P(LR-test)
## chir tubaire: 1 vs 0  9.3 (6.44,13.43) < 0.001      < 0.001
##
## Log-likelihood = -1008.6796
## No. of observations = 1725
## AIC value = 2021.3592

m2<- glm(ct~ageu,data=geu,family="binomial")
logistic.display(m2)

##
## Logistic regression predicting 0:acc 1:GEU
##
##                OR(95%CI)          P(Wald's test) P(LR-test)
## atcd geu: 1 vs 0  14.78 (8.63,25.34) < 0.001      < 0.001
##
## Log-likelihood = -1023.0071
## No. of observations = 1725
## AIC value = 2050.0142

m3<- glm(ct~ctub+ageu,data=geu,family="binomial")
logistic.display(m3, simplified = T)

##
##                OR lower95ci upper95ci      Pr(>|Z|)
## ctub 4.823154  2.925536  7.951641 6.901165e-10
## ageu 3.390356  1.659791  6.925278 8.069997e-04
```

## V.3 Sélection fondée sur le changement de l'estimation de l'odds-ratio (module chest)

### Modèle complet (tableau 1)

```
complet<-glm(ct~age30+tabf+univf+afcs+aivg+ainf+clomid+ptub,data=geu,family="binomial")
summary(complet)
```

```
##
## Call:
## glm(formula = ct ~ age30 + tabf + univf + afcs + aivg + ainf +
##      clomid + ptub, family = "binomial", data = geu)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.9722      0.1177  -16.761  < 2e-16 ***
## age30          0.3315      0.1251   2.651  0.00803 **
## tabf           0.9658      0.1229   7.856  3.95e-15 ***
## univf          0.1241      0.1427   0.869  0.38462
## afcs           0.3761      0.1407   2.674  0.00750 **
## aivg           0.1161      0.1632   0.711  0.47680
## ainf           0.9001      0.1592   5.656  1.55e-08 ***
## clomid        0.2563      0.3221   0.796  0.42615
## ptub           1.4642      0.1389  10.542  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 2029.7  on 1618  degrees of freedom
## Residual deviance: 1717.3  on 1610  degrees of freedom
## (106 observations deleted due to missingness)
## AIC: 1735.3
##
## Number of Fisher Scoring iterations: 4
```

### Chest (tableau 2)

Le package chest de R ne propose qu'une méthode ascendante, contrairement au module de Stata. Les résultats diffèrent donc un peu de ceux du livre. La méthode ascendante de Stata est identique au module de R.

```
chest.age30<-chest_glm(
  crude = "ct ~ age30 + tabf", xlist = c( "univf", "afcs", "aivg", "ainf",
"clomid", "ptub"),
  na_omit = TRUE, data = geu
)
```

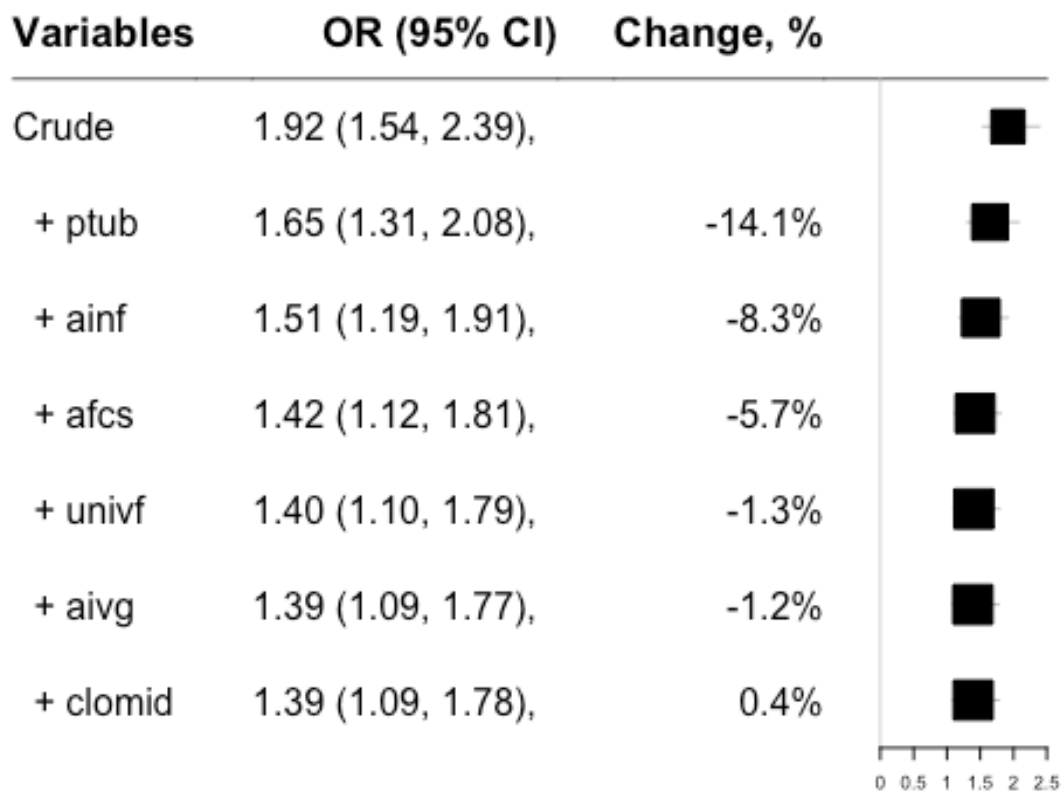
```
chest.age30
```

```
## $data
```

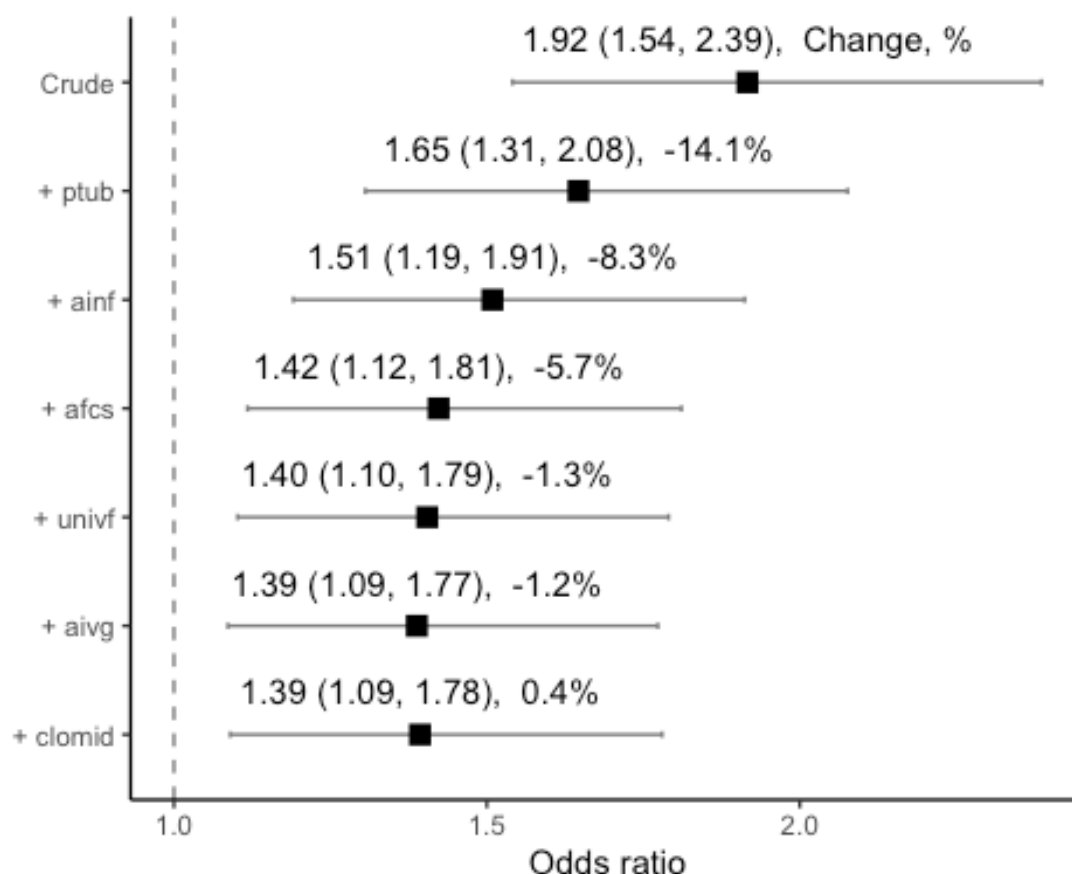
	variables	est	lb	ub	Change	p	n
## 1	Crude	1.916276	1.540962	2.386193	NA	5.466374e-09	1619
## 2	+ ptub	1.646201	1.305838	2.076388	-14.0937716	2.496523e-05	1619
## 3	+ ainf	1.508756	1.190705	1.911897	-8.3491752	6.586309e-04	1619
## 4	+ afcs	1.422795	1.117657	1.810810	-5.6975185	4.156321e-03	1619

```
## 5    + univf 1.404646 1.101702 1.790394 -1.2756024 6.061759e-03 1619
## 6      + aivg 1.388115 1.086191 1.773342 -1.1768231 8.692358e-03 1619
## 7    + clomid 1.393066 1.089908 1.779932  0.3566546 8.029587e-03 1619
##
## $fun
## [1] "chest_glm"
##
## $family
## [1] "binomial"

chest_forest(chest.age30)
```



```
chest_plot(chest.age30)
```



#### Remarques :

Les résultats semblent différents de ceux donnés dans le livre, obtenus avec Stata. Il y a 2 raisons à cela :

- le module chest de R utilise une procédure ascendante en partant du modèle sans ajustement (crude) alors que Stata a été utilisé avec une procédure descendante en partant du modèle complet. Si on retire l'option backward de la commande Stata, on retrouve les résultats de R : même ordre des variables et mêmes OR.
- La 2ème raison est plus problématique. Les pourcentages de changement donnés par R portent sur les OR et non sur les coefficients. Le seuil recommandé de 10% ne doit donc pas être utilisé. Si on calcule les pourcentages de changement des coefficients, on retrouve ceux de Stata (pas ceux du livre mais ceux avec procédure ascendante).

Au total, les deux logiciels font la même chose et conduisent à conserver les variables afcs, ainf et ptub (en plus de tabf qui est forcée dans le modèle).

## V.4. Méthodes de sélection pas à pas (stepwise)

### Tableau 4

```
geu.sw=na.omit( geu[ , c("ct","age30","tabf","univf","afcs","aivg","ainf","clomid","ptub")])
mod.compl<- glm(ct~.,data=geu.sw,family="binomial")
mod.sw<-step(mod.compl,direction = "backward", scope=list(lower=as.formula(ct ~ age30+tabf)), trace=0)
mod.sw$anova
```

##	Step	Df	Deviance	Resid. Df	Resid. Dev	AIC
## 1	NA	NA		1610	1717.300	1735.300
## 2	- aivg	1	0.5029634	1611	1717.803	1733.803



```
## 3 - clomid 1 0.6178301      1612    1718.421 1732.421
## 4 - univf 1 0.8764642      1613    1719.297 1731.297

summary(mod.sw)

##
## Call:
## glm(formula = ct ~ age30 + tabf + afcs + ainf + ptub, family = "binomial",
##      data = geu.sw)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.9234     0.1104 -17.423  < 2e-16 ***
## age30         0.3526     0.1230   2.866  0.00416 **
## tabf          0.9588     0.1210   7.923 2.31e-15 ***
## afcs          0.3724     0.1405   2.650  0.00804 **
## ainf          0.9232     0.1514   6.097 1.08e-09 ***
## ptub          1.4579     0.1385  10.530  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 2029.7  on 1618  degrees of freedom
## Residual deviance: 1719.3  on 1613  degrees of freedom
## AIC: 1731.3
##
## Number of Fisher Scoring iterations: 4
```

### Remarques :

- la création d'un data.frame geu.sw permet que le stepwise soit exécuté sur une base où il n'y a aucune donnée manquante. Un modèle logistique est toujours exécuté sur les sujets sans données manquantes, mais lorsqu'on ajoute ou retire des variables comme avec stepwise l'ensemble des sujets sans données manquantes varie. On travaille donc ici sur un ensemble de sujets qui convient à chaque étape. Stata fait la même chose, mais ne demande pas que cela soit précisé explicitement.
- Dans la commande step, l'option scope permet de forcer des variables (ici age30 et tabfc). L'option trace=0 évite que l'ensemble des étapes du stepwise soient données. Si on veut les connaître, il faut écrire trace=1.
- la commande mod.sw\$anova donne un résumé des étapes en listant les variables retirées successivement du modèle.

### Tableau 5

```
geu$agecf<-factor(geu$agec)
geu.sw2=na.omit( geu[ , c("ct","agecf","tabf","univf","afcs","aivg","ainf","clomid","ptub")])
mod.compl2<- glm(ct~.,data=geu.sw2, family = "binomial")
mod.sw2<-step(mod.compl2,direction = "backward", trace=0)
mod.sw2$anova

##          Step Df  Deviance Resid. Df Resid. Dev      AIC
## 1             NA      NA      1608    1715.056 1737.056
## 2 - aivg       1 0.3666485      1609    1715.423 1735.423
## 3 - clomid     1 0.6058202      1610    1716.029 1734.029
## 4 - univf      1 0.8021807      1611    1716.831 1732.831
```

```
summary(mod.sw2)

##
## Call:
## glm(formula = ct ~ agecf + tabf + afcs + ainf + ptub, family = "binomial",
##      data = geu.sw2)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.94258    0.16088 -12.075  < 2e-16 ***
## agecf2       0.03069    0.17378   0.177  0.85982
## agecf3       0.27392    0.18414   1.488  0.13685
## agecf4       0.55872    0.20915   2.671  0.00755 **
## tabf         0.96978    0.12142   7.987 1.38e-15 ***
## afcs         0.35602    0.14111   2.523  0.01163 *
## ainf         0.92014    0.15211   6.049 1.45e-09 ***
## ptub         1.44078    0.13881  10.380  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 2029.7  on 1618  degrees of freedom
## Residual deviance: 1716.8  on 1611  degrees of freedom
## AIC: 1732.8
##
## Number of Fisher Scoring iterations: 4
```

# Chapitre 6

## Régression logistique multinomiale

Importation du fichier de données.

```
load("data/premafiv.rdata")
```

### Commandes pour la régression logistique multinomiale

Il y a 2 commandes (au moins) pour la régression logistique multinomiale dans R :

- multinom {net}
- multinomial {VGAM}

Elles donnent des résultats très proches, les différences ne portent en général que sur la 3ème ou la 4ème décimale des estimations des coefficients. Je ne sais pas quelle est l'explication de cet écart.

La présentation des résultats est un peu différente sans que l'une soit nettement supérieure à l'autre.

Dans la suite, j'utilise la commande multinomial de {VGAM} pour les raisons suivantes :

- les résultats sont les mêmes que la commande mlogit de Stata. Cela permet d'être cohérent avec les tableaux du livre, mais je reconnais que ce n'est pas une raison majeure.
- les résultats de cette commande utilisée avec deux variables et un terme d'interaction sont identiques à ceux de plusieurs régressions logistiques classiques comme explique dans le § II.3 du chapitre 6. Ce n'est pas le cas avec multinom {net}, avec, là aussi, des différences minimes.
- le package {VGAM} comprend aussi les modèles ordinaux.

Attention cependant que les commandes de {VGAM} utilisées dans ce chapitre commencent toutes par `vglm` ...

Une explication est donnée dans la doc de VGAM : "At the heart of this package are the vector generalized linear and additive model (VGLM/VGAM) classes".

En pratique, les 2 commandes multinom et multinomial peuvent être utilisées sans réels inconvénients car leurs résultats sont très proches.

## II.2.Exemple et interprétation des résultats avec une seule covariable

**Tableau 1 : régression logistique multinomiale**

```
premafiv$cpremaf<-factor(premafiv$cpremaf,labels=c("a terme", "rupt", "spont", "
prov"))
reg.mul1<-vglm(cpremaf~age35,multinomial(refLevel="a terme"), data=(premafiv))
summary(reg.mul1)

##
## Call:
## vglm(formula = cpremaf ~ age35, family = multinomial(refLevel = "a terme"),
##      data = (premafiv))
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept):1 -4.21147    0.12216 -34.475  < 2e-16 ***
## (Intercept):2 -3.08865    0.07073 -43.668  < 2e-16 ***
## (Intercept):3 -3.40710    0.08244 -41.326  < 2e-16 ***
## age35:1         0.17615    0.19936   0.884  0.37691
## age35:2         0.08469    0.11887   0.712  0.47618
```

```
## age35:3      0.35869    0.12775    2.808    0.00499 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: log(mu[,2]/mu[,1]), log(mu[,3]/mu[,1]),
## log(mu[,4]/mu[,1])
##
## Residual deviance: 6050.365 on 22797 degrees of freedom
##
## Log-likelihood: -3025.183 on 22797 degrees of freedom
##
## Number of Fisher scoring iterations: 5
##
## Warning: Hauck-Donner effect detected in the following estimate(s):
## '(Intercept):1', '(Intercept):2', '(Intercept):3'
##
## Reference group is level 1 of the response

coef <- coef(reg.mul1)
IC <- confint.default(reg.mul1)
cbind(coef, IC)

##              coef      2.5 %      97.5 %
## (Intercept):1 -4.21147379 -4.4509036 -3.9720440
## (Intercept):2 -3.08864724 -3.2272747 -2.9500197
## (Intercept):3 -3.40710097 -3.5686876 -3.2455143
## age35:1        0.17615452 -0.2145832  0.5668922
## age35:2        0.08468803 -0.1482883  0.3176644
## age35:3        0.35869000  0.1083121  0.6090679
```

#### Remarques :

- La transformation de `cprema` en facteur (`cpremaf`) est nécessaire pour pouvoir fixer la catégorie de référence de `Y` (`refLevel="a terme"`). Par défaut, la catégorie de référence serait la dernière.
- Attention cependant que les catégories sont renumérotées de 0 à 3 (au lieu de 1 à 4) comme dans la variable `cprema`. Pour s'y retrouver, on peut noter que c'est indiqué dans la ligne "Names of linear predictors" où "on voit bien" quelles sont les numérotations des catégories de `Y`.
- Le warning "Hauck-Donner effect detected" correspond à un "défaut" du test de Wald qui peut conduire à une sur-estimation du degré de signification. Ici, il concerne le test d'une constante, peu utile en pratique, et qui, de plus, est ici déjà très significatif.

• Les commandes équivalentes avec `multinom` {net} sont les suivantes (par défaut la catégorie de référence pour `Y` est la première avec `multinom`) :

```
reg.multinom <- multinom(cprema ~ age35, data=(premafiv))
summary(reg.multinom)
mlogit.display(reg.multinom)
```

et si on veut les IC des coefficients et les degrés de signification précis :

```
confint(reg.multinom)
coef.beta <- summary(reg.multinom)$coefficients
std.er <- summary(reg.multinom)$standard.errors
Z <- coef.beta/std.er
p <- (1 - pnorm(abs(Z), 0, 1)) * 2
p
```

### Comparaison avec des régressions logistiques binomiales (Tableau 2)

```
reg_rup <- glm( rupmemb~age35, family=binomial, data=premafiiv)
summary(reg_rup)

##
## Call:
## glm(formula = rupmemb ~ age35, family = binomial, data = premafiiv)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -4.2115      0.1222  -34.474   <2e-16 ***
## age35         0.1762      0.1994   0.884     0.377
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##    Null deviance: 1124.1  on 7014  degrees of freedom
## Residual deviance: 1123.4  on 7013  degrees of freedom
## (643 observations deleted due to missingness)
## AIC: 1127.4
##
## Number of Fisher Scoring iterations: 7

c(coef(reg_rup)[2], confint.default(reg_rup)[2,])

##      age35      2.5 %      97.5 %
## 0.1761545 -0.2145885 0.5668975
```

### Tableau 3 : RRR

```
exp(cbind(coef,IC))

##              coef      2.5 %      97.5 %
## (Intercept):1 0.01482450 0.01166802 0.01883490
## (Intercept):2 0.04556355 0.03966545 0.05233867
## (Intercept):3 0.03313713 0.02819283 0.03894853
## age35:1       1.19262233 0.80687769 1.76278020
## age35:2       1.08837748 0.86218247 1.37391511
## age35:3       1.43145299 1.11439550 1.83871673
```

## § II.3. Régression logistique multinomiale versus plusieurs régressions binomiales (plusieurs covariables)

### Sans terme d'interaction (Tableau 4)

```
reg.mul2<-vglm(cpremaf~age35+hta,multinomial(refLevel="a terme"), data=(premafiiv))

## Warning in vglm.fitter(x = x, y = y, w = w, offset = offset, Xm2 = Xm2, :
## iterations terminated because half-step sizes are very small

## Warning in vglm.fitter(x = x, y = y, w = w, offset = offset, Xm2 = Xm2, : so
## me
## quantities such as z, residuals, SEs may be inaccurate due to convergence at
## a
## half-step
```

## Remarque :

Le “Warning in vglm.fitter” porte sur les critères de convergence pour la méthode du maximum de vraisemblance utilisée pour estimer les paramètres du modèle. Sa signification n’est pas très claire. Je comprends juste que certains paramètres n’ont peut-être pas une estimation très précise, en particulier les variances des coefficients et leur test de comparaison à 0.

Je ne sais pas bien comment il faudrait en tenir compte, mais il me semble qu’une conclusion possible est que si les p sont assez loin du seuil de signification, il n’y a pas à remettre en cause le fait que les coefficients correspondants sont significatifs ou non significatifs. Si, au contraire, p est proche du seuil de signification, il faut rester prudent et modeste dans sa conclusion (mais, c’est une vérité générale ...).

```
summary(reg.mul2)

##
## Call:
## vglm(formula = cpremaf ~ age35 + hta, family = multinomial(refLevel = "a ter
me"),
##   data = (premafiv))
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept):1 -4.20764    0.12312 -34.175 < 2e-16 ***
## (Intercept):2 -3.10335    0.07171 -43.278 < 2e-16 ***
## (Intercept):3 -3.53277    0.08638 -40.896 < 2e-16 ***
## age35:1        0.17677    0.19938  0.887 0.37529
## age35:2        0.08235    0.11890  0.693 0.48855
## age35:3        0.33994    0.12875  2.640 0.00828 **
## hta:1         -0.13807    0.58960 -0.234 0.81485
## hta:2          0.40488    0.27332  1.481 0.13852
## hta:3          1.70689    0.18548  9.202 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: log(mu[,2]/mu[,1]), log(mu[,3]/mu[,1]),
## log(mu[,4]/mu[,1])
##
## Residual deviance: 5987.156 on 22794 degrees of freedom
##
## Log-likelihood: -2993.578 on 22794 degrees of freedom
##
## Number of Fisher scoring iterations: 6
##
## Warning: Hauck-Donner effect detected in the following estimate(s):
## '(Intercept):1', '(Intercept):2', '(Intercept):3'
##
## Reference group is level 1 of the response

coef(reg.mul2)

## (Intercept):1 (Intercept):2 (Intercept):3      age35:1      age35:2
## -4.20763873   -3.10335271   -3.53277278   0.17677043   0.08235155
##      age35:3      hta:1      hta:2      hta:3
## 0.33994272  -0.13807086   0.40488266   1.70688733
```

```

confint(reg.mul2)

##                2.5 %      97.5 %
## (Intercept):1 -4.44895310 -3.9663244
## (Intercept):2 -3.24389615 -2.9628093
## (Intercept):3 -3.70208410 -3.3634615
## age35:1       -0.21400402  0.5675449
## age35:2       -0.15068551  0.3153886
## age35:3        0.08759335  0.5922921
## hta:1         -1.29366485  1.0175231
## hta:2         -0.13082216  0.9405875
## hta:3          1.34334439  2.0704303

```

Comparaison avec des régressions logistiques classiques

```

reg_rup2 <- glm( rupmemb~age35+hta, family=binomial, data=premafiv)
c(coef(reg_rup2)[2], confint.default(reg_rup2)[2,])

##      age35      2.5 %      97.5 %
## 0.1768493 -0.2139333  0.5676318

c(coef(reg_rup2)[3], confint.default(reg_rup2)[3,])

##      hta      2.5 %      97.5 %
## -0.1388484 -1.2944781  1.0167813

```

#### Avec terme d'interaction (Tableau 5)

```

reg.mul2i<-vgam(cpreamaf~age35*hta,multinomial(refLevel=1), data=(premafiv))
summary(reg.mul2i)

##
## Call:
## vgam(formula = cpreamaf ~ age35 * hta, family = multinomial(refLevel = 1),
##      data = (premafiv))
##
## Names of additive predictors: log(mu[,2]/mu[,1]), log(mu[,3]/mu[,1]),
## log(mu[,4]/mu[,1])
##
## Dispersion Parameter for multinomial family: 1
##
## Residual deviance: 5986.141 on 22791 degrees of freedom
##
## Log-likelihood: -2993.071 on 22791 degrees of freedom
##
## Number of Fisher scoring iterations: 6
##
## DF for Terms and Approximate Chi-squares for Nonparametric Effects
##
##           Df Npar Df Npar Chisq P(Chi)
## (Intercept):1 1
## (Intercept):2 1
## (Intercept):3 1
## age35:1       1
## age35:2       1
## age35:3       1
## hta:1         1
## hta:2         1
## hta:3         1

```

```
## age35:hta:1      1
## age35:hta:2      1
## age35:hta:3      1

coef(reg.mul2i,matrix=TRUE)

##              log(mu[,2]/mu[,1]) log(mu[,3]/mu[,1]) log(mu[,4]/mu[,1])
## (Intercept)    -4.2114539704      -3.09780380      -3.5490784
## age35           0.1869955356       0.06659714       0.3790353
## hta             -0.0006736275       0.27197057       1.8218575
## age35:hta       -0.3693170919       0.31069709      -0.2736748
```

Comparaison avec des régressions logistiques classiques

```
reg_rup2i <- glm(rupmemb~age35*hta, family=binomial, data=premafav)
summary(reg_rup2i)

##
## Call:
## glm(formula = rupmemb ~ age35 * hta, family = binomial, data = premafav)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -4.2114540  0.1240005 -33.963  <2e-16 ***
## age35        0.1869955  0.2020469   0.926   0.355
## hta         -0.0006736  0.7230376  -0.001   0.999
## age35:hta    -0.3693171  1.2492290  -0.296   0.768
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 1124.1  on 7014  degrees of freedom
## Residual deviance: 1123.2  on 7011  degrees of freedom
## (643 observations deleted due to missingness)
## AIC: 1131.2
##
## Number of Fisher Scoring iterations: 7

c(coef(reg_rup2i)[2], confint.default(reg_rup2i)[2,])

##      age35      2.5 %      97.5 %
## 0.1869955 -0.2090091  0.5830002

c(coef(reg_rup2i)[3], confint.default(reg_rup2i)[3,])

##      hta      2.5 %      97.5 %
## -0.0006736275 -1.4178012240  1.4164539689
```

## II.4. Comparaison des OR associés à X selon les catégories de Y

```
exp(cbind(coef(reg.mul2),confint.default(reg.mul2)))

##              2.5 %      97.5 %
## (Intercept):1 0.01488147 0.01169080 0.01894293
## (Intercept):2 0.04489842 0.03901160 0.05167355
## (Intercept):3 0.02922377 0.02467205 0.03461523
## age35:1       1.19335710 0.80734514 1.76393105
## age35:2       1.08583747 0.86011816 1.37079191
```



```
## age35:3      1.40486712 1.09154415 1.80812807
## hta:1        0.87103696 0.27426380 2.76633439
## hta:2        1.49912658 0.87737380 2.56148578
## hta:3        5.51177840 3.83183726 7.92823366

linearHypothesis(reg.mul2,c("hta:1 = hta:2"),test="Chisq")

## Linear hypothesis test
##
## Hypothesis:
## hta:1 - hta:2 = 0
##
## Model 1: restricted model
## Model 2: cpremaf ~ age35 + hta
##
##   Res.Df Df  Chisq Pr(>Chisq)
## 1    22795
## 2    22794   1 0.7142      0.398

linearHypothesis(reg.mul2,c("hta:3 = hta:2"),test="Chisq")

## Linear hypothesis test
##
## Hypothesis:
## - hta:2 + hta:3 = 0
##
## Model 1: restricted model
## Model 2: cpremaf ~ age35 + hta
##
##   Res.Df Df  Chisq Pr(>Chisq)
## 1    22795
## 2    22794   1 17.037 3.666e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Test simultané de plusieurs OR (pour tester ici si les 3 OR sont homogènes)

```
linearHypothesis(reg.mul2,c("hta:1 = hta:2","hta:3 = hta:2"),test="Chisq")

## Linear hypothesis test
##
## Hypothesis:
## hta:1 - hta:2 = 0
## - hta:2 + hta:3 = 0
##
## Model 1: restricted model
## Model 2: cpremaf ~ age35 + hta
##
##   Res.Df Df  Chisq Pr(>Chisq)
## 1    22796
## 2    22794   2 22.885 1.073e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

## II.5. Changement de classe de référence pour Y (Tableau 7)

On prend la classe 3 de cprema ("spont") comme référence au lieu de la classe 1

```
reg.mul1c<-vglm(cpremaf~age35,multinomial(refLevel="spont"), data=(premafiv))
summary(reg.mul1c)

##
## Call:
## vglm(formula = cpremaf ~ age35, family = multinomial(refLevel = "spont"),
##      data = (premafiv))
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept):1  3.08865    0.07073  43.668 < 2e-16 ***
## (Intercept):2 -1.12283    0.13961  -8.043 8.78e-16 ***
## (Intercept):3 -0.31845    0.10660  -2.987 0.00281 **
## age35:1       -0.08469    0.11887  -0.712 0.47618
## age35:2        0.09147    0.22929   0.399 0.68996
## age35:3        0.27400    0.17073   1.605 0.10853
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: log(mu[,1]/mu[,3]), log(mu[,2]/mu[,3]),
## log(mu[,4]/mu[,3])
##
## Residual deviance: 6050.365 on 22797 degrees of freedom
##
## Log-likelihood: -3025.183 on 22797 degrees of freedom
##
## Number of Fisher scoring iterations: 5
##
## Warning: Hauck-Donner effect detected in the following estimate(s):
## '(Intercept):1'
##
## Reference group is level 3 of the response
```

Remarque :

Attention, de nouveau : les classes de Y (cpremaf) sont renumérotées de 0 à 3. 0 est la classe de référence choisie ("spont") et les autres classes sont numérotées 1 à 3. Par exemple, la classe 1 est donc la classe "à terme".

## Régression logistique ordinale

### IV. Modèle cumulative-odds

Attention à la lecture des résultats

- Le modèle cumulative-odds du package {vgam} modélise  $\text{LogitP}(Y < j) = \alpha_j + \sigma(\beta_{ai} X_i)$ . Avec l'option `reverse=TRUE`, il modélise  $\text{LogitP}(Y > j) = \alpha_j + \sigma(\beta_{ai} X_i)$  ce qui équivaut à  $\text{LogitP}(Y < j) = -\alpha_j - \sigma(\beta_{ai} X_i)$  de sorte qu'on retrouve bien les coefficients beta du livre.

Attention cependant que les constantes sont changées de signes par rapport à celles des tableaux du livre.

- Comme pour la régression logistique multinomiale avec vgm, les classes de Y sont renumérotées. Les numéros de lignes dans les résultats (de 1 à 4) ne correspondent pas aux valeurs de Y. Ici sf5 prend 5 valeurs de 1 à 5. Pour comprendre à quoi correspondent les lignes, il faut se reporter à la ligne "Names of linear predictors" qui indique que la ligne de résultats avec le chiffre 1 est celle de  $\text{logitlink}(P[Y \geq 2])$ , etc ...

Importation du fichier de données.

```
load("data/septaviih.rdata")
```

## Tableaux 8 et 9

```
table(septaviih$sf5, septaviih$sf3, dnn=c("sf5", "sf3"))
```

```
##      sf3
## sf5   1   2   3
##  1 121   0   0
##  2   0 210   0
##  3   0 117   0
##  4   0   0 29
##  5   0   0 14
```

# ou

```
CrossTable(septaviih$sf5, septaviih$sf3, prop.chisq = F, prop.c=T, prop.r=T, prop.
t=F, format = "SPSS", dnn = c("sf5", "sf3"))
```

```
##
##      Cell Contents
## |-----|
## |                      Count |
## |          Row Percent |
## |          Column Percent |
## |-----|
##
## Total Observations in Table:  491
##
##      sf5 | sf3
##      sf5 | 1 | 2 | 3 | Row Total |
## -----|---|---|---|-----|
##      1 | 121 | 0 | 0 | 121 |
##      | 100.000% | 0.000% | 0.000% | 24.644% |
##      | 100.000% | 0.000% | 0.000% | |
## -----|---|---|---|-----|
##      2 | 0 | 210 | 0 | 210 |
##      | 0.000% | 100.000% | 0.000% | 42.770% |
##      | 0.000% | 64.220% | 0.000% | |
## -----|---|---|---|-----|
##      3 | 0 | 117 | 0 | 117 |
##      | 0.000% | 100.000% | 0.000% | 23.829% |
##      | 0.000% | 35.780% | 0.000% | |
## -----|---|---|---|-----|
##      4 | 0 | 0 | 29 | 29 |
##      | 0.000% | 0.000% | 100.000% | 5.906% |
##      | 0.000% | 0.000% | 67.442% | |
## -----|---|---|---|-----|
```

##	5	0	0	14	14
##		0.000%	0.000%	100.000%	2.851%
##		0.000%	0.000%	32.558%	
##	-----	-----	-----	-----	-----
##	Column Total	121	327	43	491
##		24.644%	66.599%	8.758%	
##	-----	-----	-----	-----	-----

##

##

```
cum5<-vglm(sf5~precaire,cumulative(link="logitlink",parallel=T,reverse=T),data=
(septavih))
summary(cum5)
```

##

## Call:

```
## vglm(formula = sf5 ~ precaire, family = cumulative(link = "logitlink",
##      parallel = T, reverse = T), data = (septavih))
```

##

## Coefficients:

```
##      Estimate Std. Error z value Pr(>|z|)
## (Intercept):1  0.8165    0.1181   6.914 4.71e-12 ***
## (Intercept):2 -1.1113    0.1240  -8.959 < 2e-16 ***
## (Intercept):3 -2.7834    0.1841 -15.118 < 2e-16 ***
## (Intercept):4 -3.9865    0.2874 -13.872 < 2e-16 ***
## precaire      0.9491    0.1760   5.393 6.95e-08 ***
```

## ---

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

##

```
## Names of linear predictors: logitlink(P[Y>=2]), logitlink(P[Y>=3]),
## logitlink(P[Y>=4]), logitlink(P[Y>=5])
```

##

```
## Residual deviance: 1265.261 on 1959 degrees of freedom
```

##

```
## Log-likelihood: -632.6304 on 1959 degrees of freedom
```

##

```
## Number of Fisher scoring iterations: 4
```

##

```
## Warning: Hauck-Donner effect detected in the following estimate(s):
```

```
## '(Intercept):4'
```

##

##

```
## Exponentiated coefficients:
```

```
## precaire
```

```
## 2.583448
```

#### IV.4. Regroupement de classes (Tableau 10)

```
cum3<-vglm(sf3~precaire,cumulative(link="logitlink",parallel=T,reverse=T),data=
(septavih))
summary(cum3)
```

##

## Call:

```
## vglm(formula = sf3 ~ precaire, family = cumulative(link = "logitlink",
##      parallel = T, reverse = T), data = (septavih))
```

##

## Coefficients:

```
##           Estimate Std. Error z value Pr(>|z|)
## (Intercept):1    0.7961     0.1203   6.616 3.68e-11 ***
## (Intercept):2   -2.8390     0.1985  -14.299 < 2e-16 ***
## precaire         1.0443     0.2111   4.948 7.49e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: logitlink(P[Y>=2]), logitlink(P[Y>=3])
##
## Residual deviance: 788.0927 on 979 degrees of freedom
##
## Log-likelihood: -394.0463 on 979 degrees of freedom
##
## Number of Fisher scoring iterations: 4
##
## No Hauck-Donner effect found in any of the estimates
##
## Exponentiated coefficients:
## precaire
## 2.841427
```

#### IV.4. Test de l'hypothèse des odds proportionnels (Tableau 11)

```
cum5np<-vglm(sf5~precaire,cumulative(link="logitlink",parallel=F,reverse=T),dat
a=(septavih))
summary(cum5np)
```

```
##
## Call:
## vglm(formula = sf5 ~ precaire, family = cumulative(link = "logitlink",
##   parallel = F, reverse = T), data = (septavih))
##
## Coefficients:
##           Estimate Std. Error z value Pr(>|z|)
## (Intercept):1    0.8180     0.1238   6.606 3.95e-11 ***
## (Intercept):2   -1.0943     0.1317  -8.311 < 2e-16 ***
## (Intercept):3   -2.9687     0.2647 -11.213 < 2e-16 ***
## (Intercept):4   -4.1010     0.4509  -9.095 < 2e-16 ***
## precaire:1       0.9424     0.2424   3.888 0.000101 ***
## precaire:2       0.8980     0.1982   4.531 5.88e-06 ***
## precaire:3       1.2511     0.3350   3.735 0.000188 ***
## precaire:4       1.1334     0.5658   2.003 0.045150 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: logitlink(P[Y>=2]), logitlink(P[Y>=3]),
## logitlink(P[Y>=4]), logitlink(P[Y>=5])
##
## Residual deviance: 1263.866 on 1956 degrees of freedom
##
## Log-likelihood: -631.9332 on 1956 degrees of freedom
##
## Number of Fisher scoring iterations: 4
##
## Warning: Hauck-Donner effect detected in the following estimate(s):
## '(Intercept):3', '(Intercept):4'
```

```
##
##
## Exponentiated coefficients:
## precaire:1 precaire:2 precaire:3 precaire:4
## 2.566162 2.454674 3.494017 3.106286

linearHypothesis(cum5np,c("precaire:1 = precaire:2","precaire:1 = precaire:3","
precaire:1 = precaire:4"),test="Chisq")

## Linear hypothesis test
##
## Hypothesis:
## precaire:1 - precaire:2 = 0
## precaire:1 - precaire:3 = 0
## precaire:1 - precaire:4 = 0
##
## Model 1: restricted model
## Model 2: sf5 ~ precaire
##
## Res.Df Df Chisq Pr(>Chisq)
## 1 1959
## 2 1956 3 1.3548 0.7162
```

#### IV.5. Présentation des résultats (Tableau 12)

```
pred<-predictvglm(cum3,type=c("response"))
septaviah$pred<-round(predictvglm(cum3,type=c("response")),7)
table(septaviah$precaire,septaviah$pred[,1])

##
## 0.1369981 0.3108511
## 0 0 307
## 1 184 0
```

Problème : je ne comprends pas pourquoi il y a 4 valeurs pour `pred[,1]`, il ne devrait y en avoir que 2 (une par valeur de la variable `precaire`).

Il y a 3 valeurs très proches (les 7 premières décimales sont les mêmes) dont la différence est peu explicable. En arrondissant les valeurs des pourcentages pédits de chaque catégorie de `sf3`, on retrouve les chiffres du tableau 12.

```
round(aggregate(septaviah$pred[, "1"], list(septaviah$precaire), FUN=mean),2)

## Group.1 mean.septaviah$pred[, "1"]
## 1 0 0.31
## 2 1 0.14

round(aggregate(septaviah$pred[, "2"], list(septaviah$precaire), FUN=mean),2)

## Group.1 mean.septaviah$pred[, "2"]
## 1 0 0.63
## 2 1 0.72

round(aggregate(septaviah$pred[, "3"], list(septaviah$precaire), FUN=mean),2)

## Group.1 mean.septaviah$pred[, "3"]
## 1 0 0.06
## 2 1 0.14
```

## Présentation des résultats (Figure 1)

```

for (i in 1:3) {
  t<-table(septaviah$precaire,pred[,i])
}

cum.ep<-vglm(sf3~epices,cumulative(link="logitlink",parallel=F,reverse=T),data=
(septaviah))
summary(cum.ep)

##
## Call:
## vglm(formula = sf3 ~ epices, family = cumulative(link = "logitlink",
##       parallel = F, reverse = T), data = (septaviah))
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept):1  0.284594   0.186414   1.527   0.127
## (Intercept):2 -3.306486   0.325584 -10.156 < 2e-16 ***
## epices:1       0.036174   0.007382   4.900 9.57e-07 ***
## epices:2       0.031967   0.008186   3.905 9.41e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: logitlink(P[Y>=2]), logitlink(P[Y>=3])
##
## Residual deviance: 776.385 on 978 degrees of freedom
##
## Log-likelihood: -388.1925 on 978 degrees of freedom
##
## Number of Fisher scoring iterations: 4
##
## Warning: Hauck-Donner effect detected in the following estimate(s):
## '(Intercept):2'
##
## Exponentiated coefficients:
## epices:1 epices:2
## 1.036836 1.032484

pred.ep<-predictvglm(cum.ep,type=c("response"))


data.plot <- data.frame(x = septaviah$epices,
                        pred.ep)
colnames(data.plot)<-c("epices","robuste","pré-fragile","fragile")

data_long <- melt(data.plot, id = "epices")
p <- ggplot(data_long,
            aes(x = epices,
                y = value,
                color = variable,
                linetype=variable)) +
  geom_line() +
  labs(x="epices", y="Fréquence des classes de sf3") +
  scale_color_manual(values=c("black","blue","red")) +
  scale_linetype_manual(values=c("solid","dashed","dotted")) +
  theme_classic() +

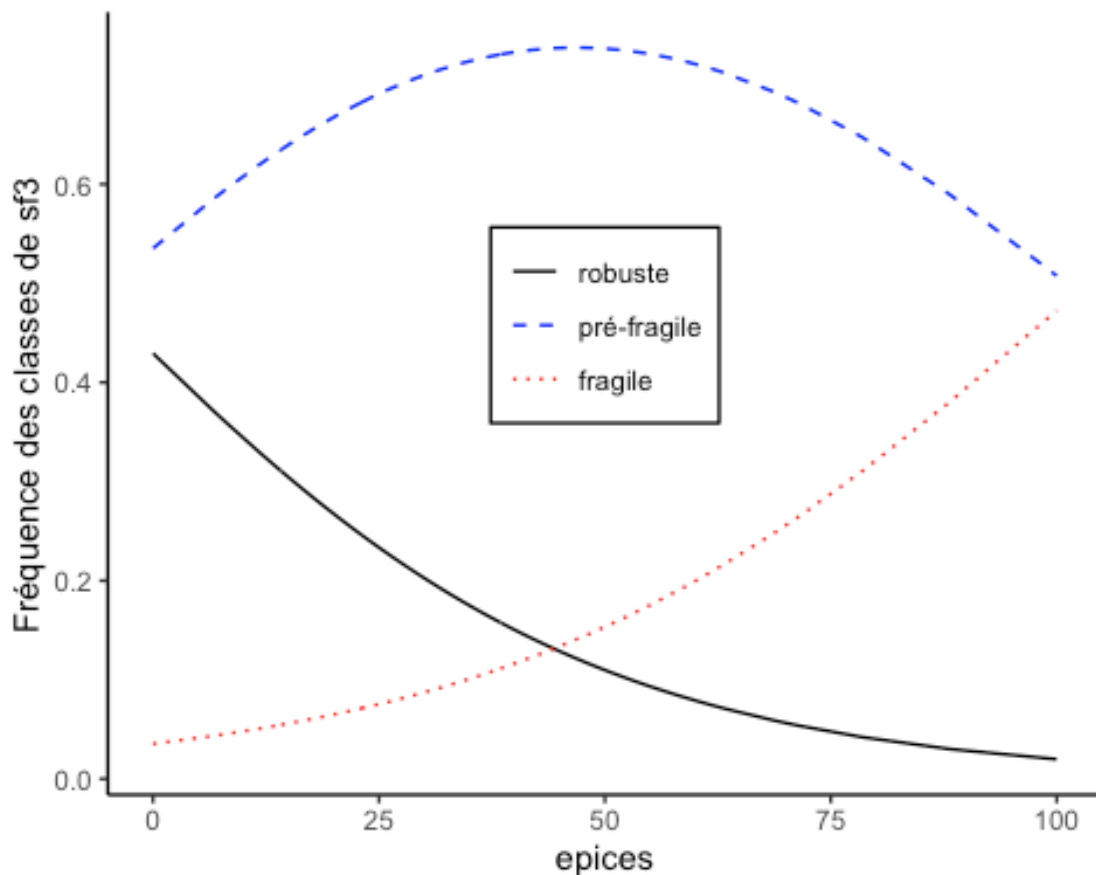
```

```

theme(legend.position = "inside", legend.position.inside = c(0.5, 0.6), legend
.title = element_blank() )+
  theme(legend.background = element_rect(color = "black", size = 0.5))

## Warning: The `size` argument of `element_rect()` is deprecated as of ggplot2
3.4.0.
##  Please use the `linewidth` argument instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
p

```



### Plusieurs variables indépendantes Xi (Tableau 13)

```

cum_prop<-vglm(sf5~precaire+cd4_350+mrc+parten, cumulative(link="logitlink", pa
rallel=T,reverse=T), data=(septavih))
summary(cum_prop)

##
## Call:
## vglm(formula = sf5 ~ precaire + cd4_350 + mrc + parten, family = cumulative(
link = "logitlink",
##   parallel = T, reverse = T), data = (septavih))
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept):1   0.8542    0.1859   4.594 4.36e-06 ***
## (Intercept):2  -1.0793    0.1890  -5.712 1.12e-08 ***
## (Intercept):3  -2.7598    0.2327 -11.861 < 2e-16 ***
## (Intercept):4  -3.9670    0.3209 -12.363 < 2e-16 ***

```



```
## precaire      0.8929      0.1820      4.907 9.27e-07 ***
## cd4_350      0.1533      0.2413      0.635  0.525
## mrc          0.1928      0.1708      1.128  0.259
## parten      -0.1877      0.1761     -1.066  0.287
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: logitlink(P[Y>=2]), logitlink(P[Y>=3]),
## logitlink(P[Y>=4]), logitlink(P[Y>=5])
##
## Residual deviance: 1262.343 on 1956 degrees of freedom
##
## Log-likelihood: -631.1715 on 1956 degrees of freedom
##
## Number of Fisher scoring iterations: 4
##
## Warning: Hauck-Donner effect detected in the following estimate(s):
## '(Intercept):4'
##
## Exponentiated coefficients:
## precaire  cd4_350      mrc    parten
## 2.4421728 1.1657079 1.2126345 0.8288881

cum_nprop<-vglm(sf5~precaire+cd4_350+mrc+parten,cumulative(link="logitlink", parallel=F,reverse=T), data=(septaviih))
summary(cum_nprop)

##
## Call:
## vglm(formula = sf5 ~ precaire + cd4_350 + mrc + parten, family = cumulative(
## link = "logitlink",
## parallel = F, reverse = T), data = (septaviih))
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept):1  0.90357    0.22253   4.060 4.90e-05 ***
## (Intercept):2 -1.05129    0.21279  -4.941 7.79e-07 ***
## (Intercept):3 -3.30534    0.40762  -8.109 5.10e-16 ***
## (Intercept):4 -4.23903    0.66946  -6.332 2.42e-10 ***
## precaire:1     0.92594    0.25005   3.703 0.000213 ***
## precaire:2     0.82776    0.20545   4.029 5.60e-05 ***
## precaire:3     1.14998    0.34649   3.319 0.000904 ***
## precaire:4     0.98174    0.58272   1.685 0.092036 .
## cd4_350:1     -0.13968    0.30626  -0.456 0.648333
## cd4_350:2      0.36019    0.27498   1.310 0.190229
## cd4_350:3      0.20412    0.43452   0.470 0.638527
## cd4_350:4      0.04078    0.76342   0.053 0.957403
## mrc:1         0.11272    0.21907   0.515 0.606868
## mrc:2         0.12937    0.20123   0.643 0.520310
## mrc:3         0.86580    0.33253   2.604 0.009223 **
## mrc:4         0.62770    0.55418   1.133 0.257350
## parten:1     -0.17234    0.22715  -0.759 0.448048
## parten:2     -0.21349    0.20655  -1.034 0.301340
## parten:3     -0.12281    0.34125  -0.360 0.718936
## parten:4     -0.15715    0.57696  -0.272 0.785330
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: logitlink(P[Y>=2]), logitlink(P[Y>=3]),
## logitlink(P[Y>=4]), logitlink(P[Y>=5])
##
## Residual deviance: 1252.289 on 1944 degrees of freedom
##
## Log-likelihood: -626.1446 on 1944 degrees of freedom
##
## Number of Fisher scoring iterations: 6
##
## Warning: Hauck-Donner effect detected in the following estimate(s):
## '(Intercept):3', '(Intercept):4'
##
## Exponentiated coefficients:
## preaire:1 preaire:2 preaire:3 preaire:4 cd4_350:1 cd4_350:2 cd4_350:3
## 2.5242521 2.2881811 3.1581204 2.6691023 0.8696361 1.4336037 1.2264453
## cd4_350:4 mrc:1 mrc:2 mrc:3 mrc:4 parten:1 parten:2
## 1.0416190 1.1193218 1.1381076 2.3769019 1.8733023 0.8416971 0.8077635
## parten:3 parten:4
## 0.8844322 0.8545728

linearHypothesis(cum_nprop,c("mrc:1 = mrc:2",
                             "mrc:1 = mrc:3",
                             "mrc:1 = mrc:4")
                  ,test="Chisq")

## Linear hypothesis test
##
## Hypothesis:
## mrc:1 - mrc:2 = 0
## mrc:1 - mrc:3 = 0
## mrc:1 - mrc:4 = 0
##
## Model 1: restricted model
## Model 2: sf5 ~ preaire + cd4_350 + mrc + parten
##
##   Res.Df Df    Chisq Pr(>Chisq)
## 1     1947
## 2     1944   3 6.0262    0.1103
```

### Modèle cumulative odds avec l'hypothèse des odds proportionnels sauf pour la variable mrc (Tableau 14)

```
cum_mrc<-vglm(sf5~preaire+ cd4_350+mrc+parten,cumulative(link="logitlink",parallel=F~ 1+mrc,reverse=T),data=(septavih))
summary(cum_mrc)

##
## Call:
## vglm(formula = sf5 ~ preaire + cd4_350 + mrc + parten, family = cumulative(
## link = "logitlink",
## parallel = F ~ 1 + mrc, reverse = T), data = (septavih))
##
## Coefficients:
## Estimate Std. Error z value Pr(>|z|)
```

```
## (Intercept):1  0.8851      0.1927   4.593 4.37e-06 ***
## (Intercept):2 -1.0561      0.1942  -5.438 5.38e-08 ***
## (Intercept):3 -3.1136      0.2928 -10.634 < 2e-16 ***
## (Intercept):4 -4.2057      0.4387  -9.586 < 2e-16 ***
## precaire      0.8859      0.1820   4.868 1.13e-06 ***
## cd4_350       0.1568      0.2415   0.649 0.51622
## mrc:1         0.1051      0.2189   0.480 0.63094
## mrc:2         0.1256      0.2007   0.626 0.53155
## mrc:3         0.8762      0.3292   2.662 0.00777 **
## mrc:4         0.6614      0.5500   1.203 0.22916
## parten       -0.1828      0.1763  -1.037 0.29970
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: logitlink(P[Y>=2]), logitlink(P[Y>=3]),
## logitlink(P[Y>=4]), logitlink(P[Y>=5])
##
## Residual deviance: 1255.817 on 1953 degrees of freedom
##
## Log-likelihood: -627.9085 on 1953 degrees of freedom
##
## Number of Fisher scoring iterations: 4
##
## Warning: Hauck-Donner effect detected in the following estimate(s):
## '(Intercept):4'
##
## Exponentiated coefficients:
## precaire  cd4_350    mrc:1    mrc:2    mrc:3    mrc:4    parten
## 2.4252213 1.1697130 1.1108684 1.1338142 2.4017755 1.9375757 0.8329385
```

## § V. Modèle continuation-ratio

Importation du fichier de données.

```
load("data/suc_fiv.rdata")
```

Tableau 15

```
table(suc_fiv$rgsuc)

##
##   1    2    3    4    9
## 797 402 172  85 3544
```

```
CrossTable(suc_fiv$rgsuc)
```

```
##
##
##   Cell Contents
## |-----|
## |                      N |
## |      N / Table Total |
## |-----|
##
##
```

```
## Total Observations in Table: 5000
```

```
##
```

```
##
```

	1	2	3	4	9
	797	402	172	85	3544
	0.159	0.080	0.034	0.017	0.709

```
##
```

```
##
```

```
##
```

```
##
```

### Modèle continuation-ratio avec odds proportionnels (Tableau 16)

```
fiv1<-vglm(rgsuc~age35,cratio(link="logitlink",parallel=T,reverse=F),data=(suc_
fiv))
```

```
summary(fiv1)
```

```
##
```

```
## Call:
```

```
## vglm(formula = rgsuc ~ age35, family = cratio(link = "logitlink",
##      parallel = T, reverse = F), data = (suc_fiv))
```

```
##
```

```
## Coefficients:
```

```
##      Estimate Std. Error z value Pr(>|z|)
## (Intercept):1  1.52998    0.04233  36.147 < 2e-16 ***
## (Intercept):2  2.10981    0.05528  38.167 < 2e-16 ***
## (Intercept):3  2.91144    0.07998  36.401 < 2e-16 ***
## (Intercept):4  3.59175    0.11117  32.309 < 2e-16 ***
## age35          0.43729    0.06287   6.955 3.52e-12 ***
```

```
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
```

```
## Names of linear predictors: logitlink(P[Y>1|Y>=1]), logitlink(P[Y>2|Y>=2]),
## logitlink(P[Y>3|Y>=3]), logitlink(P[Y>4|Y>=4])
```

```
##
```

```
## Residual deviance: 9194.313 on 19995 degrees of freedom
```

```
##
```

```
## Log-likelihood: -4597.157 on 19995 degrees of freedom
```

```
##
```

```
## Number of Fisher scoring iterations: 5
```

```
##
```

```
## Warning: Hauck-Donner effect detected in the following estimate(s):
```

```
## '(Intercept):3', '(Intercept):4'
```

```
##
```

```
##
```

```
## Exponentiated coefficients:
```

```
##      age35
```

```
## 1.548502
```

```
coef1<-exp(coefficients(fiv1))
```

```
IC1<-exp(confint.default(fiv1))
```

```
cbind(coef1,IC1)
```

```
##      coef1      2.5 %      97.5 %
## (Intercept):1  4.618086  4.250438  5.017535
## (Intercept):2  8.246671  7.399895  9.190345
```

```
## (Intercept):3 18.383173 15.715914 21.503110
## (Intercept):4 36.297716 29.191285 45.134162
## age35          1.548502  1.368973  1.751575
```

### Modèle continuation-ratio sans odds proportionnels (Tableau 17) et test des odds proportionnels (Tableau 18)

```
fiv2<-vglm(rgsuc~age35,cratio(link="logitlink",parallel=F,reverse=F),data=(suc_
fiv))
summary(fiv2)

##
## Call:
## vglm(formula = rgsuc ~ age35, family = cratio(link = "logitlink",
##       parallel = F, reverse = F), data = (suc_fiv))
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept):1  1.52130    0.04525  33.618 < 2e-16 ***
## (Intercept):2  2.13515    0.06231  34.266 < 2e-16 ***
## (Intercept):3  2.91712    0.09183  31.767 < 2e-16 ***
## (Intercept):4  3.54252    0.12582  28.156 < 2e-16 ***
## age35:1        0.46956    0.08760   5.360 8.32e-08 ***
## age35:2        0.34756    0.11570   3.004 0.00267 **
## age35:3        0.41660    0.17454   2.387 0.01699 *
## age35:4        0.63033    0.25807   2.442 0.01459 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: logitlink(P[Y>1|Y>=1]), logitlink(P[Y>2|Y>=2]),
## logitlink(P[Y>3|Y>=3]), logitlink(P[Y>4|Y>=4])
##
## Residual deviance: 9192.99 on 19992 degrees of freedom
##
## Log-likelihood: -4596.495 on 19992 degrees of freedom
##
## Number of Fisher scoring iterations: 5
##
## Warning: Hauck-Donner effect detected in the following estimate(s):
## '(Intercept):3', '(Intercept):4'
##
## Exponentiated coefficients:
## age35:1 age35:2 age35:3 age35:4
## 1.599288 1.415608 1.516798 1.878228

coef2<-exp(coefficients(fiv2))
IC2<-exp(confint.default(fiv2))
cbind(coef2,IC2)

##              coef2      2.5 %    97.5 %
## (Intercept):1  4.578151  4.189587  5.002753
## (Intercept):2  8.458333  7.485919  9.557064
## (Intercept):3 18.488000 15.442775 22.133725
## (Intercept):4 34.553846 27.002309 44.217265
## age35:1        1.599288  1.346971  1.898869
## age35:2        1.415608  1.128382  1.775946
```

```
## age35:3      1.516798  1.077352  2.135491
## age35:4      1.878228  1.132604  3.114717

linearHypothesis(fiv2,c("age35:1 = age35:2","age35:1 = age35:3","age35:1 = age35:4"),test="Chisq")

## Linear hypothesis test
##
## Hypothesis:
## age35:1 - age35:2 = 0
## age35:1 - age35:3 = 0
## age35:1 - age35:4 = 0
##
## Model 1: restricted model
## Model 2: rgsuc ~ age35
##
##      Res.Df Df    Chisq Pr(>Chisq)
## 1    19995
## 2    19992   3  1.3104      0.7267

lrtest_vglm(fiv1,fiv2)

## Likelihood ratio test
##
## Model 1: rgsuc ~ age35
## Model 2: rgsuc ~ age35
##      #Df LogLik Df    Chisq Pr(>Chisq)
## 1 19995 -4597.2
## 2 19992 -4596.5 -3  1.3228      0.7237
```

### § V.3. Modélisation de la durée d'infécondité

#### Age et durée d'infécondité en variables dichotomiques (Tableau 19)

```
fiv3<-vglm(rgsuc~dinf5,cratio(link="logitlink",parallel=T,reverse=F),data=(suc_
fiv))
summary(fiv3)

##
## Call:
## vglm(formula = rgsuc ~ dinf5, family = cratio(link = "logitlink",
##      parallel = T, reverse = F), data = (suc_fiv))
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept):1  1.62588    0.04676  34.769  <2e-16 ***
## (Intercept):2  2.20978    0.05871  37.638  <2e-16 ***
## (Intercept):3  3.01223    0.08237  36.569  <2e-16 ***
## (Intercept):4  3.69334    0.11288  32.718  <2e-16 ***
## dinf5          0.07718    0.05599   1.378    0.168
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: logitlink(P[Y>1|Y>=1]), logitlink(P[Y>2|Y>=2]),
## logitlink(P[Y>3|Y>=3]), logitlink(P[Y>4|Y>=4])
##
## Residual deviance: 9243.307 on 19995 degrees of freedom
##
```

```
## Log-likelihood: -4621.654 on 19995 degrees of freedom
##
## Number of Fisher scoring iterations: 5
##
## Warning: Hauck-Donner effect detected in the following estimate(s):
## '(Intercept):3', '(Intercept):4'
##
##
## Exponentiated coefficients:
##      dinf5
## 1.080233
```

```
cbind(exp(coefficients(fiv3)),exp(confint.default(fiv3)))
```

```
##              2.5 %      97.5 %
## (Intercept):1  5.082906  4.6377528  5.570787
## (Intercept):2  9.113739  8.1230932 10.225199
## (Intercept):3 20.332689 17.3013537 23.895139
## (Intercept):4 40.178899 32.2040316 50.128629
## dinf5          1.080233  0.9679684  1.205517
```

### Durée d'infécondité en 5 classes (dinf) sous forme de 4 variables indicatrices (Tableau 20)

```
suc_fiv$dinf.f <- factor(suc_fiv$dinf , labels=c("1-2 ans", "3-4 ans", "5-8 an
s", ">=9 ans"))
```

```
fiv4<-vglm(rgsuc~dinf.f,cratio(link="logitlink",parallel=T,reverse=F),data=(suc
_fiv))
summary(fiv4)
```

```
##
## Call:
## vglm(formula = rgsuc ~ dinf.f, family = cratio(link = "logitlink",
##      parallel = T, reverse = F), data = (suc_fiv))
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept):1  1.63674    0.07178  22.802   <2e-16 ***
## (Intercept):2  2.22050    0.08008  27.729   <2e-16 ***
## (Intercept):3  3.02277    0.09874  30.612   <2e-16 ***
## (Intercept):4  3.70372    0.12533  29.552   <2e-16 ***
## dinf.f3-4 ans -0.01612    0.08155  -0.198    0.8433
## dinf.f5-8 ans  0.02611    0.08134   0.321    0.7482
## dinf.f>=9 ans  0.18834    0.10700   1.760    0.0784 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: logitlink(P[Y>1|Y>=1]), logitlink(P[Y>2|Y>=2]),
## logitlink(P[Y>3|Y>=3]), logitlink(P[Y>4|Y>=4])
##
## Residual deviance: 9240.343 on 19993 degrees of freedom
##
## Log-likelihood: -4620.171 on 19993 degrees of freedom
##
## Number of Fisher scoring iterations: 5
##
## Warning: Hauck-Donner effect detected in the following estimate(s):
## '(Intercept):4'
```

```
##
##
## Exponentiated coefficients:
## dinf.f3-4 ans dinf.f5-8 ans dinf.f>=9 ans
##      0.9840072      1.0264582      1.2072434

cbind(exp(coefficients(fiv4)),exp(confint.default(fiv4)))

##              2.5 %      97.5 %
## (Intercept):1  5.1384145  4.4640476  5.914655
## (Intercept):2  9.2119342  7.8738812 10.777370
## (Intercept):3 20.5480373 16.9324456 24.935668
## (Intercept):4 40.5981737 31.7561473 51.902131
## dinf.f3-4 ans  0.9840072  0.8386514  1.154556
## dinf.f5-8 ans  1.0264582  0.8751996  1.203858
## dinf.f>=9 ans  1.2072434  0.9788548  1.488920
```

### Durée d'infécondité en 5 classes avec la variable d'origine (dinf) (Tableau 21)

```
fiv5<-vglm(rgsuc~dinf,cratio(link="logitlink",parallel=T,reverse=F),data=(suc_f
iv))
summary(fiv5)

##
## Call:
## vglm(formula = rgsuc ~ dinf, family = cratio(link = "logitlink",
##      parallel = T, reverse = F), data = (suc_fiv))
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept):1  1.56704      0.06271  24.990  <2e-16 ***
## (Intercept):2  2.15091      0.07208  29.840  <2e-16 ***
## (Intercept):3  2.95309      0.09237  31.970  <2e-16 ***
## (Intercept):4  3.63416      0.12037  30.191  <2e-16 ***
## dinf           0.02346      0.01224   1.917   0.0553 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: logitlink(P[Y>1|Y>=1]), logitlink(P[Y>2|Y>=2]),
## logitlink(P[Y>3|Y>=3]), logitlink(P[Y>4|Y>=4])
##
## Residual deviance: 9241.486 on 19995 degrees of freedom
##
## Log-likelihood: -4620.743 on 19995 degrees of freedom
##
## Number of Fisher scoring iterations: 5
##
## Warning: Hauck-Donner effect detected in the following estimate(s):
## '(Intercept):3', '(Intercept):4'
##
##
## Exponentiated coefficients:
##      dinf
## 1.023738

cbind(exp(coefficients(fiv5)),exp(confint.default(fiv5)))
```



```
##                2.5 %    97.5 %
## (Intercept):1  4.792452  4.2381938  5.419195
## (Intercept):2  8.592682  7.4605960  9.896552
## (Intercept):3 19.165073 15.9913189 22.968714
## (Intercept):4 37.869962 29.9111391 47.946487
## dinf           1.023738  0.9994709  1.048594
```

### *comparaison des modèles par rapport des vraisemblances*

```
lrtest_vglm(fiv4,fiv5)
```

```
## Likelihood ratio test
##
## Model 1: rgsuc ~ dinf.f
## Model 2: rgsuc ~ dinf
##      #Df  LogLik Df  Chisq Pr(>Chisq)
## 1 19993 -4620.2
## 2 19995 -4620.7  2  1.1429    0.5647
```

## VI. Modèle adjacent-category

Importation du fichier de données.

```
load("data/ado.rdata")
```

### Tableau 22

```
ado.adj<-vglm(satis~genre,acat(link="loglink",parallel=T,reverse=F),data=(ado))
summary(ado.adj)
```

```
##
## Call:
## vglm(formula = satis ~ genre, family = acat(link = "loglink",
##      parallel = T, reverse = F), data = (ado))
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept):1 -0.66177    0.03923 -16.867 < 2e-16 ***
## (Intercept):2 -2.04876    0.05333 -38.418 < 2e-16 ***
## (Intercept):3 -1.23769    0.08162 -15.164 < 2e-16 ***
## (Intercept):4 -1.05853    0.12253  -8.639 < 2e-16 ***
## genre          0.10353    0.02288   4.525 6.04e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: loglink(P[Y=2]/P[Y=1]), loglink(P[Y=3]/P[Y=2]),
## loglink(P[Y=4]/P[Y=3]), loglink(P[Y=5]/P[Y=4])
##
## Residual deviance: 26840.64 on 56631 degrees of freedom
##
## Log-likelihood: -13420.32 on 56631 degrees of freedom
##
## Number of Fisher scoring iterations: 6
##
## No Hauck-Donner effect found in any of the estimates
##
## Exponentiated coefficients:
```

```
##      genre
## 1.109076

coef6<-exp(coefficients(ado.adj))
IC6<-exp(confint.default(ado.adj))
cbind(coef6,IC6)

##              coef6      2.5 %      97.5 %
## (Intercept):1 0.5159353 0.4777478 0.5571752
## (Intercept):2 0.1288947 0.1161028 0.1430960
## (Intercept):3 0.2900532 0.2471736 0.3403714
## (Intercept):4 0.3469672 0.2728895 0.4411538
## genre         1.1090756 1.0604427 1.1599389
```

### Tableau 23

```
ado$satisf<-factor(ado$satis,labels=c("Très satisfait(e)", "Satisfait(e)", "Ni
satisfait(e), ni insatisfait(e)", "Pas très satisfait(e)", "Pas satisfait(e) du
tout"))
reg.mult.genre<-vglm(satisf~genre,multinomial(refLevel="Très satisfait(e)"), da
ta=(ado))
summary(reg.mult.genre)

##
## Call:
## vglm(formula = satisf ~ genre, family = multinomial(refLevel = "Très satisfait(e)"),
##      data = (ado))
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept):1 -0.65721    0.05807 -11.318 < 2e-16 ***
## (Intercept):2 -2.88876    0.12883 -22.422 < 2e-16 ***
## (Intercept):3 -3.92405    0.21386 -18.349 < 2e-16 ***
## (Intercept):4 -4.53666    0.32001 -14.177 < 2e-16 ***
## genre:1        0.10059    0.03618   2.780  0.00543 **
## genre:2        0.32030    0.07773   4.120  3.78e-05 ***
## genre:3        0.29544    0.12916   2.287  0.02217 *
## genre:4        0.11582    0.19792   0.585  0.55842
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: log(mu[,2]/mu[,1]), log(mu[,3]/mu[,1]),
## log(mu[,4]/mu[,1]), log(mu[,5]/mu[,1])
##
## Residual deviance: 26836.04 on 56628 degrees of freedom
##
## Log-likelihood: -13418.02 on 56628 degrees of freedom
##
## Number of Fisher scoring iterations: 6
##
## Warning: Hauck-Donner effect detected in the following estimate(s):
## '(Intercept):2', '(Intercept):3', '(Intercept):4'
##
## Reference group is level 1 of the response

cbind(coef(reg.mult.genre),confint.default(reg.mult.genre))
```

```
##              2.5 %      97.5 %
## (Intercept):1 -0.6572058 -0.77101401 -0.5433976
## (Intercept):2 -2.8887624 -3.14127177 -2.6362531
## (Intercept):3 -3.9240478 -4.34320420 -3.5048914
## (Intercept):4 -4.5366634 -5.16387594 -3.9094509
## genre:1        0.1005915  0.02967930  0.1715038
## genre:2        0.3202999  0.16794280  0.4726570
## genre:3        0.2954392  0.04229007  0.5485884
## genre:4        0.1158168 -0.27209022  0.5037238
```

```
lrtest(reg.mult.genre,ado.adj)
```

```
## Likelihood ratio test
##
## Model 1: satisf ~ genre
## Model 2: satisf ~ genre
##      #Df LogLik Df  Chisq Pr(>Chisq)
## 1 56628 -13418
## 2 56631 -13420  3  4.6028    0.2033
```

## Tableaux 24 et 25

```
ado.adj.MC<-vglm(satis~MC,acat(link="loglink",parallel=T,reverse=T),data=(ado))
summary(ado.adj.MC)
```

```
##
## Call:
## vglm(formula = satisf ~ MC, family = acat(link = "loglink", parallel = T,
##      reverse = T), data = (ado))
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept):1  0.51537    0.01851  27.847 < 2e-16 ***
## (Intercept):2  1.90063    0.03962  47.971 < 2e-16 ***
## (Intercept):3  1.08794    0.07288  14.928 < 2e-16 ***
## (Intercept):4  0.90725    0.11657   7.783 7.1e-15 ***
## MC              -0.09218    0.03311  -2.784 0.00538 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: loglink(P[Y=1]/P[Y=2]), loglink(P[Y=2]/P[Y=3]),
## loglink(P[Y=3]/P[Y=4]), loglink(P[Y=4]/P[Y=5])
##
## Residual deviance: 26853.72 on 56631 degrees of freedom
##
## Log-likelihood: -13426.86 on 56631 degrees of freedom
##
## Number of Fisher scoring iterations: 6
##
## No Hauck-Donner effect found in any of the estimates
##
## Exponentiated coefficients:
##              MC
## 0.9119416
##
cbind(coef(ado.adj.MC),confint.default(ado.adj.MC))
```

```
##              2.5 %      97.5 %
## (Intercept):1  0.5153654  0.4790922  0.55163858
## (Intercept):2  1.9006309  1.8229763  1.97828552
## (Intercept):3  1.0879352  0.9450972  1.23077320
## (Intercept):4  0.9072508  0.6787713  1.13573021
## MC              -0.0921793 -0.1570828 -0.02727579

reg.mult.MC<-vglm(satisf~MC,multinomial(refLevel="Très satisfait(e)"), data=(ad
o))

## Warning in vglm.fitter(x = x, y = y, w = w, offset = offset, Xm2 = Xm2, : so
me
## quantities such as z, residuals, SEs may be inaccurate due to convergence at
a
## half-step

summary(reg.mult.MC)

##
## Call:
## vglm(formula = satisf ~ MC, family = multinomial(refLevel = "Très satisfait(
e)"),
##      data = (ado))
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept):1 -0.50651    0.01926 -26.298 < 2e-16 ***
## (Intercept):2 -2.43977    0.04171 -58.498 < 2e-16 ***
## (Intercept):3 -3.47035    0.06800 -51.036 < 2e-16 ***
## (Intercept):4 -4.45868    0.11040 -40.387 < 2e-16 ***
## MC:1           0.02010    0.05546   0.362  0.71702
## MC:2           0.34057    0.10570   3.222  0.00127 **
## MC:3           0.02601    0.19471   0.134  0.89373
## MC:4           0.62488    0.24665   2.533  0.01129 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: log(mu[,2]/mu[,1]), log(mu[,3]/mu[,1]),
## log(mu[,4]/mu[,1]), log(mu[,5]/mu[,1])
##
## Residual deviance: 26846.22 on 56628 degrees of freedom
##
## Log-likelihood: -13423.11 on 56628 degrees of freedom
##
## Number of Fisher scoring iterations: 7
##
## Warning: Hauck-Donner effect detected in the following estimate(s):
## '(Intercept):2', '(Intercept):3', '(Intercept):4'
##
## Reference group is level 1 of the response

cbind(coef(reg.mult.MC),confint.default(reg.mult.MC))

##              2.5 %      97.5 %
## (Intercept):1 -0.50651077 -0.54426087 -0.4687607
## (Intercept):2 -2.43976980 -2.52151439 -2.3580252
```

```
## (Intercept):3 -3.47034968 -3.60362427 -3.3370751
## (Intercept):4 -4.45868071 -4.67505647 -4.2423049
## MC:1          0.02010142 -0.08859937  0.1288022
## MC:2          0.34056683  0.13339653  0.5477371
## MC:3          0.02601042 -0.35561286  0.4076337
## MC:4          0.62487667  0.14145008  1.1083033

lrtest(reg.mult.MC,ado.adj.MC)

## Likelihood ratio test
##
## Model 1: satisf ~ MC
## Model 2: satis ~ MC
##      #Df LogLik Df Chisq Pr(>Chisq)
## 1 56628 -13423
## 2 56631 -13427  3 7.498    0.05761 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

## VII.2. Un peu d'humilité sur l'importance du choix ...

### Tableau 26

```
rgsuc26<-vglm(rgsuc~age35+dinf5,cratio(link="logitlink",parallel=T,reverse=F),
data=(suc_fiv))
summary(rgsuc26)

##
## Call:
## vglm(formula = rgsuc ~ age35 + dinf5, family = cratio(link = "logitlink",
##      parallel = T, reverse = F), data = (suc_fiv))
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept):1  1.50921    0.04924  30.648 < 2e-16 ***
## (Intercept):2  2.08907    0.06075  34.389 < 2e-16 ***
## (Intercept):3  2.89062    0.08385  34.472 < 2e-16 ***
## (Intercept):4  3.57095    0.11399  31.328 < 2e-16 ***
## age35          0.43341    0.06306   6.873 6.27e-12 ***
## dinf5          0.04603    0.05625   0.818  0.413
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: logitlink(P[Y>1|Y>=1]), logitlink(P[Y>2|Y>=2]),
## logitlink(P[Y>3|Y>=3]), logitlink(P[Y>4|Y>=4])
##
## Residual deviance: 9193.643 on 19994 degrees of freedom
##
## Log-likelihood: -4596.822 on 19994 degrees of freedom
##
## Number of Fisher scoring iterations: 5
##
## Warning: Hauck-Donner effect detected in the following estimate(s):
## '(Intercept):3', '(Intercept):4'
##
##
```

```
## Exponentiated coefficients:
##   age35   dinf5
## 1.542503 1.047110

cbind(coefficients(rgsuc26), confint.default(rgsuc26))

##               2.5 %    97.5 %
## (Intercept):1 1.50920576 1.41269101 1.6057205
## (Intercept):2 2.08906768 1.97000294 2.2081324
## (Intercept):3 2.89062029 2.72626820 3.0549724
## (Intercept):4 3.57094543 3.34753404 3.7943568
## age35         0.43340651 0.30982034 0.5569927
## dinf5         0.04603401 -0.06422094 0.1562890
```

### Tableau 27

```
rgsuc27<-vglm(rgsuc~age35+dinf5,cumulative(link="logitlink",parallel=T,reverse=
T),data=(suc_fiv))
summary(rgsuc27)
```

```
##
## Call:
## vglm(formula = rgsuc ~ age35 + dinf5, family = cumulative(link = "logitlink"
,
##   parallel = T, reverse = T), data = (suc_fiv))
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept):1  1.49903    0.05118  29.290 < 2e-16 ***
## (Intercept):2  0.98663    0.04742  20.806 < 2e-16 ***
## (Intercept):3  0.80476    0.04656  17.285 < 2e-16 ***
## (Intercept):4  0.72009    0.04623  15.577 < 2e-16 ***
## age35         0.47280    0.06833   6.919 4.54e-12 ***
## dinf5         0.04551    0.06177   0.737  0.461
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: logitlink(P[Y>=2]), logitlink(P[Y>=3]),
## logitlink(P[Y>=4]), logitlink(P[Y>=5])
##
## Residual deviance: 9193.981 on 19994 degrees of freedom
##
## Log-likelihood: -4596.991 on 19994 degrees of freedom
##
## Number of Fisher scoring iterations: 4
##
## No Hauck-Donner effect found in any of the estimates
##
## Exponentiated coefficients:
##   age35   dinf5
## 1.604479 1.046561

cbind(coefficients(rgsuc27), confint.default(rgsuc27))

##               2.5 %    97.5 %
## (Intercept):1 1.49903110 1.39872319 1.5993390
## (Intercept):2 0.98663083 0.89368641 1.0795753
```

```
## (Intercept):3 0.80475501 0.71350440 0.8960056  
## (Intercept):4 0.72009028 0.62948479 0.8106958  
## age35         0.47279884 0.33887282 0.6067249  
## dinf5         0.04550973 -0.07555893 0.1665784
```

# Chapitre 7

## III. Tests d'adéquation

### Test du chi2 de Pearson et de la déviance

Je n'ai pas trouvé ces test dans R. Il faudrait les programmer (peut-être un jour ...).  
Il ne sont donc pas dans les commandes qui suivent. Je ne suis pas très sûr que ce soit une grosse perte ...

#### Tableau 2

```
mod.geu <- glm(ct ~ ctub+agea+as.factor(tabfc)+afcs+ainf, family=binomial, data
=geu)
summary(mod.geu)

##
## Call:
## glm(formula = ct ~ ctub + agea + as.factor(tabfc) + afcs + ainf,
##      family = binomial, data = geu)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -3.06147    0.37240  -8.221  < 2e-16 ***
## ctub           1.83130    0.20477   8.943  < 2e-16 ***
## agea           0.04884    0.01240   3.938 8.23e-05 ***
## as.factor(tabfc)1 0.53135    0.17025   3.121  0.0018 **
## as.factor(tabfc)2 1.32249    0.16716   7.912 2.54e-15 ***
## as.factor(tabfc)3 1.51669    0.18625   8.143 3.85e-16 ***
## afcs           0.33711    0.13946   2.417  0.0156 *
## ainf           0.81662    0.14928   5.470 4.49e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 2120.3  on 1681  degrees of freedom
## Residual deviance: 1775.7  on 1674  degrees of freedom
## (43 observations deleted due to missingness)
## AIC: 1791.7
##
## Number of Fisher Scoring iterations: 4

confint.default(mod.geu)

##              2.5 %      97.5 %
## (Intercept)  -3.79135930 -2.33158083
## ctub         1.42996078  2.23263081
## agea         0.02452943  0.07314651
## as.factor(tabfc)1 0.19766094 0.86503591
## as.factor(tabfc)2 0.99486201 1.65011147
## as.factor(tabfc)3 1.15164030 1.88173356
```



```
## afcs          0.06377850  0.61044960
## ainf          0.52404225  1.10920506
```

### Tableau 3 : Profils

La table des profils est créée par `epi.cp()` du package `epiR`

```
data.mod.geu <- model.frame(mod.geu)
pat<-epi.cp(data.mod.geu[-1]) # [-1] car il faut retirer la première colonne d
e data.mod.geu qui contient ct
# et ne doit pas être prise en compte pour déter
miner les profils

# nombre de profils
nrow(pat$cov.pattern)

## [1] 358
```

### Tableau 4 : test de Hosmer-Lemeshow

```
library(vcdExtra)
HL<-HosmerLemeshow(mod.geu,g=10)
HL

## Hosmer and Lemeshow Goodness-of-Fit Test
##
## Call:
## glm(formula = ct ~ ctub + agea + as.factor(tabfc) + afcs + ainf,
##      family = binomial, data = geu)
## ChiSquare df    P_value
##    7.922036  8 0.4411228

HL$table

##           cut total obs      exp      chi
## 1  [0.101,0.137]   203 169 177.30156 -0.62345253
## 2  (0.137,0.155]   182 162 154.87439  0.57257454
## 3  (0.155,0.175]   163 139 135.50081  0.30060499
## 4  (0.175,0.198]   139 110 112.65566 -0.25020512
## 5  (0.198,0.234]   154 121 120.69751  0.02753331
## 6  (0.234,0.306]   169 123 123.81935 -0.07363354
## 7  (0.306,0.391]   171 108 110.92871 -0.27807041
## 8  (0.391,0.468]   168  98  96.31560  0.17163142
## 9  (0.468,0.683]   164  78  72.01327  0.70547787
## 10 (0.683,0.961]   169  28  31.89313 -0.68936616
```

Remarque :

Le  $\chi^2$  d'Hosmer-Lemeshow de R ne donne pas la même valeur que celui de Stata figurant dans le livre. Pas très loin, mais quand. Je n'ai pas compris pourquoi, d'autant que le tableau des observés et prédits est le même dans les 2 logiciels. En faisant le calcul "à la main" à partir de ce tableau, je trouve le résultat de Stata. R utilise peut-être une formule différente mais laquelle et pourquoi ?

R propose d'autres fonctions pour calculer le  $\chi^2$  d'Hosmer-Lemeshow. Elles sont indiquées ci-dessous et donnent le même résultat que la fonction `HosmerLemeshow()` de `{vcdExtra}`. R est cohérent ! sauf pour la dernière fonction (`hltest`) qui fait 11 groupes et donne (donc ?) un  $\chi^2$  à 9 ddl.

```
library(performance)
```

```
## Warning: package 'performance' was built under R version 4.4.1

performance_hosmer(mod.geu, n_bins = 10)

## # Hosmer-Lemeshow Goodness-of-Fit Test
##
##   Chi-squared: 7.922
##             df: 8
##       p-value: 0.441

## Summary: model seems to fit well.

library(generalhoslem)
geu.HL<-na.omit(geu[ , c("ct","ctub", "agea","tabfc","afcs","ainf")])
logitgof(geu.HL$ct, fitted(mod.geu))

##
## Hosmer and Lemeshow test (binary model)
##
## data:  geu.HL$ct, fitted(mod.geu)
## X-squared = 7.922, df = 8, p-value = 0.4411

library(glmtoolbox)

## Registered S3 methods overwritten by 'glmtoolbox':
##   method          from
##   anova.gnm        gnm
##   coef.gnm         gnm
##   confint.gnm      gnm
##   cooks.distance.gnm gnm
##   dfbeta.gnm       gnm
##   fitted.gnm       gnm
##   model.matrix.gnm gnm
##   predict.gnm      gnm
##   residuals.gnm    gnm
##   summary.gnm      gnm
##   vcov.gnm         gnm

##
## Attaching package: 'glmtoolbox'

## The following object is masked from 'package:gnm':
##
##   gnm

hltest(mod.geu,G=10)  # fait 11 groupes (?)

##
## The Hosmer-Lemeshow goodness-of-fit test
##
##   Group Size Observed Expected
##   1 150      25 18.43799
##   2 175      22 25.07295
##   3 172      25 27.86442
##   4 169      32 31.14134
##   5 167      33 35.60956
##   6 170      48 44.89032
##   7 170      59 59.07670
##   8 167      70 70.60181
```

```
##      9 169      88 93.48219
##     10 169     140 135.99344
##     11   4       4   3.82929
##
##           Statistic = 5.5297
## degrees of freedom = 9
##           p-value = 0.78591
```

## IV. Courbe ROC

### IV.1 Tableaux 5 et 6

Les deux commandes `CrossTable()` permettent de retrouver les tableaux 5 et 6, du moins si on se souvient des définitions de la sensibilité, de la spécificité et des valeurs prédictives pour les retrouver dans les pourcentages donnés.

```
prediction<-predict(mod.geu,type="response")
data.roc<-cbind(data.mod.geu,pred=prediction)

pred.5 <- ifelse(data.roc$pred > 0.5, 1, 0)
pred.3 <- ifelse(data.roc$pred > 0.3, 1, 0)

CrossTable(pred.5,data.roc$ct,
            prop.chisq = F,
            prop.t=F,
            format="SPSS",
            dnn=c("prediction.seuil=0.5","réalité"))

##
##      Cell Contents
## |-----|
## |                Count                |
## |                Row Percent            |
## |                Column Percent        |
## |-----|
##
## Total Observations in Table: 1682
##
## prediction.seuil=0.5 | réalité
## |-----|-----|-----|
## |                0                1                Row Total |
## |-----|-----|-----|
## | 0 | 1047 | 333 | 1380 |
## |    75.870% | 24.130% | 82.045% |
## |    92.165% | 60.989% |          |
## |-----|-----|-----|
## | 1 | 89 | 213 | 302 |
## |    29.470% | 70.530% | 17.955% |
## |    7.835% | 39.011% |          |
## |-----|-----|-----|
## | Column Total | 1136 | 546 | 1682 |
## |    67.539% | 32.461% |          |
## |-----|-----|-----|
##
##
```

```
CrossTable(pred.3,data.roc$ct,
  prop.chisq = F,
  prop.t=F,
  format="SPSS",
  dnn=c("prediction.seuil=0.3","réalité"))

##
##      Cell Contents
## |-----|
## |                Count                |
## |                Row Percent           |
## |                Column Percent        |
## |-----|
##
## Total Observations in Table:  1682
##
## prediction.seuil=0.3 | réalité
## |-----|
## |                0                1 | Row Total |
## |-----|
## |                0                1 |
## |      814                183        | 997        |
## |    81.645%            18.355%       | 59.275%     |
## |    71.655%            33.516%       |
## |-----|
## |                1                1 | 685        |
## |    47.007%            52.993%       | 40.725%     |
## |    28.345%            66.484%       |
## |-----|
## |      Column Total      1136        | 546        | 1682       |
## |    67.539%            32.461%       |
## |-----|
##
##
```

**Figure 1 : Sensibilité et spécificité du classement par le modèle logistique selon le seuil choisi**

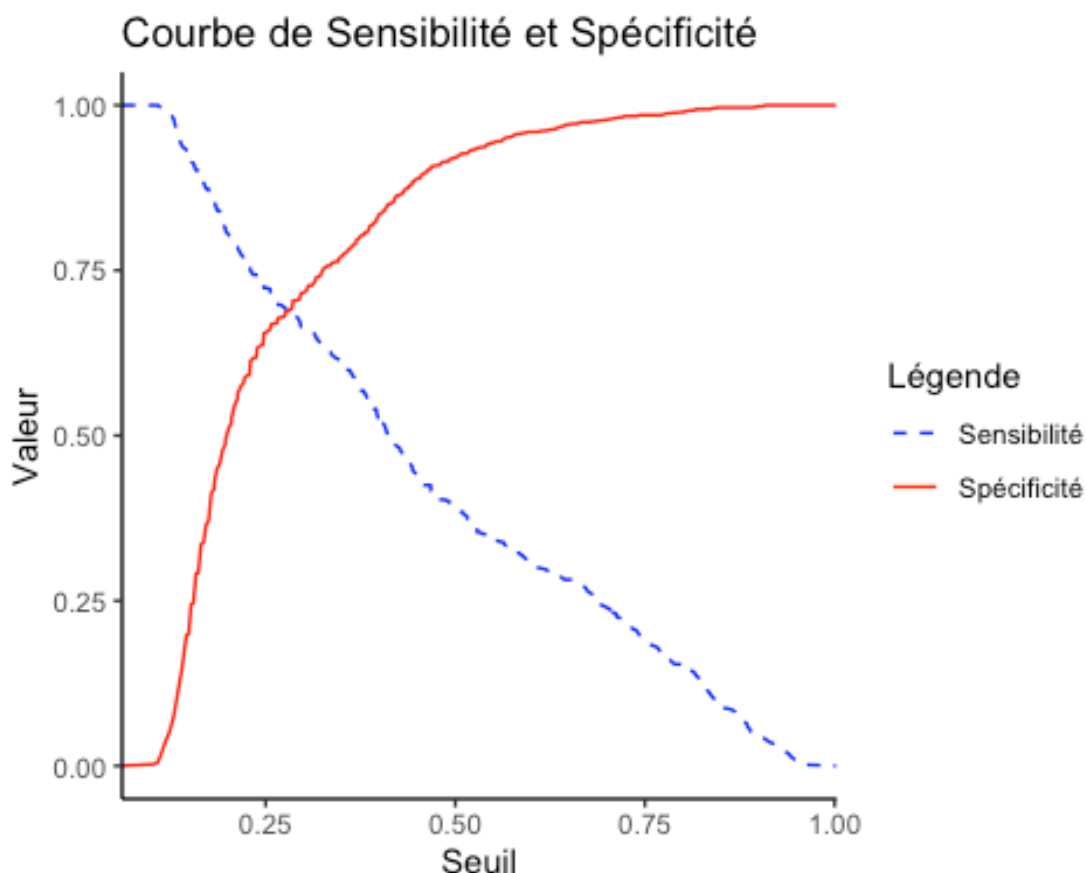
```
# Calcul de la courbe ROC
roc_curve <- roc(data.roc$ct, data.roc$pred)

## Setting levels: control = 0, case = 1
## Setting direction: controls < cases

# Extraction des valeurs de sensibilité et de spécificité
data_se_sp <- data.frame(
  seuil=roc_curve$thresholds,
  se=roc_curve$sensitivities,
  sp=roc_curve$specificities
)

# Tracé des courbes
ggplot(data_se_sp, aes(x = seuil)) +
  geom_line(aes(y = se, color = "Sensibilité"),linetype="dashed") +
  geom_line(aes(y = sp, color = "Spécificité")) +
  scale_color_manual(values = c("blue", "red")) +
  labs(title = "Courbe de Sensibilité et Spécificité",
    x = "Seuil",
    y = "Valeur",
```

```
color = "Légende") +
theme_classic()
```




#### IV.2. Aire sous la courbe ROC

```
rocplot<-ggplot(data.roc, aes(m = pred, d = ct))+
  geom_roc(n.cuts=20,labels=FALSE,pointsize=0,color="blue",linewidth=0.5)+
  labs(title="Courbe ROC",x="1-Spécificité",y="Sensibilité") +
  geom_abline(slope = 1, intercept = 0, linetype = "dashed") +
  theme_classic() +
  theme(panel.grid.major = element_line(color = "grey90",size=.3))
```

```
## Warning in geom_roc(n.cuts = 20, labels = FALSE, pointsize = 0, color = "blue",
```

```
## : Ignoring unknown parameters: `linewidth`
```

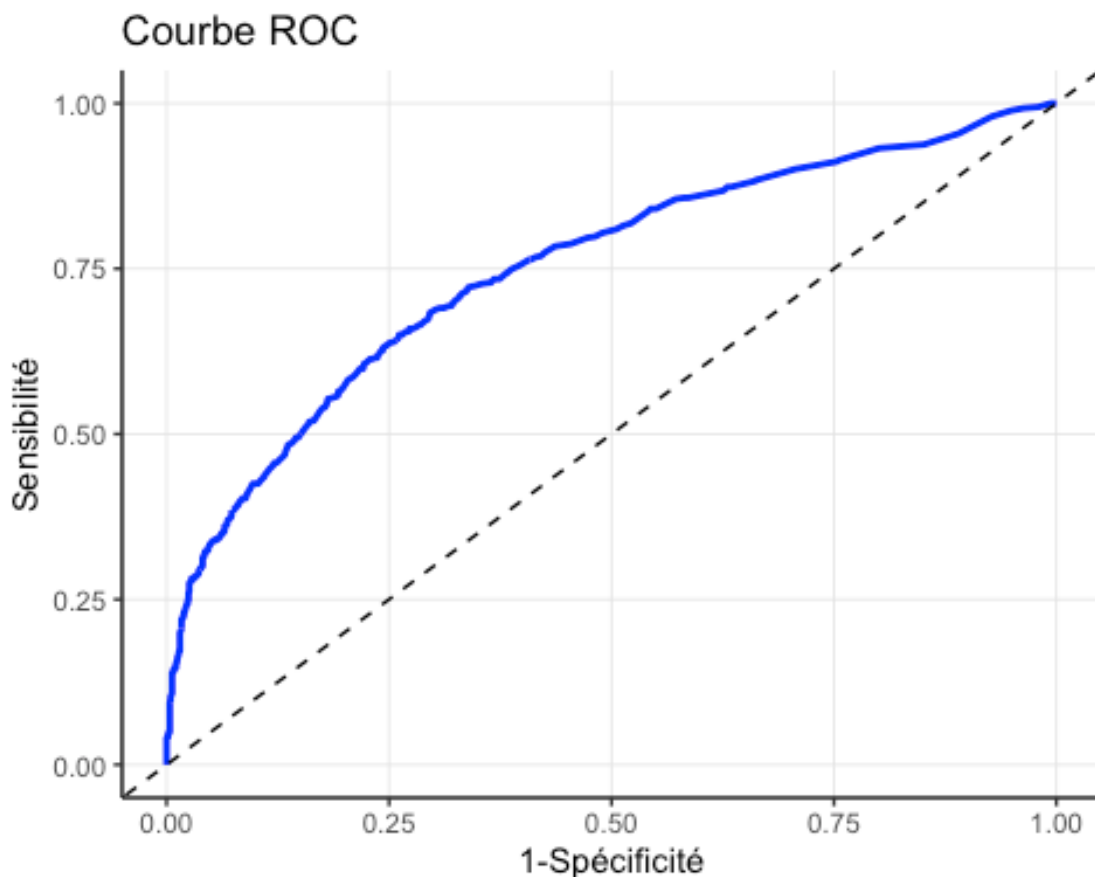
```
## Warning: The `size` argument of `element_line()` is deprecated as of ggplot2
3.4.0.
```

```
##  Please use the `linewidth` argument instead.
```

```
## This warning is displayed once every 8 hours.
```

```
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
rocplot
```



```
# ggsave(filename = "roc.png",width=9.4, height=6.8, units="cm")
AUC<-calc_auc(rocplot)
AUC

##   PANEL group    AUC
## 1      1     -1 0.75027
```

## V. Diagnostics de régression

### Construction de la table des résidus

Cette table (res.dat) est construite, à partir de la table des profils (pat construite par epi.cp pour le tableau 3) par la commande epi.cpresids qui a comme paramètres le nombre de "succès" ( $Y=1$ , c'est-à-dire GEU) observés et la probabilité prédite de succès (pred) par profil (pattern). Ces paramètres sont calculés dans un premier temps dans la table pat.

Les lignes de res.dat sont des données groupées : il y a une ligne par profil (ici 358 lignes). Chaque ligne comprend le numéro du profil, son nombre total de sujets, le nombre de sujets avec  $Y=1$  (ce qui permet de calculer le pourcentage observé de succès), la valeur prédite,  $P(Y=1)$ , avec le modèle logistique et les diverses valeurs de résidus. Il manque le delta deviance qu'il faut ajouter, ce qui est fait ci-dessous. Attention que, pour le delta beta, il faut prendre sdeltabeta (deltabeta standardisé). la table contient aussi les valeurs des variables utilisées dans le modèle logistique. Par définition, ce sont les mêmes pour tous les sujets d'un même profil.

```
succes <- as.vector(by(data.mod.geu$ct, as.factor(pat$id),FUN = sum))
pred <- as.vector(by(fitted(mod.geu), as.factor(pat$id),FUN = min)) # min peut
être remplacé par max car les valeurs prédites
# sont identiques dans un même profil.
```

```
res.dat <- epi.cpresids(obs = succes, fit = pred, covpattern = pat)

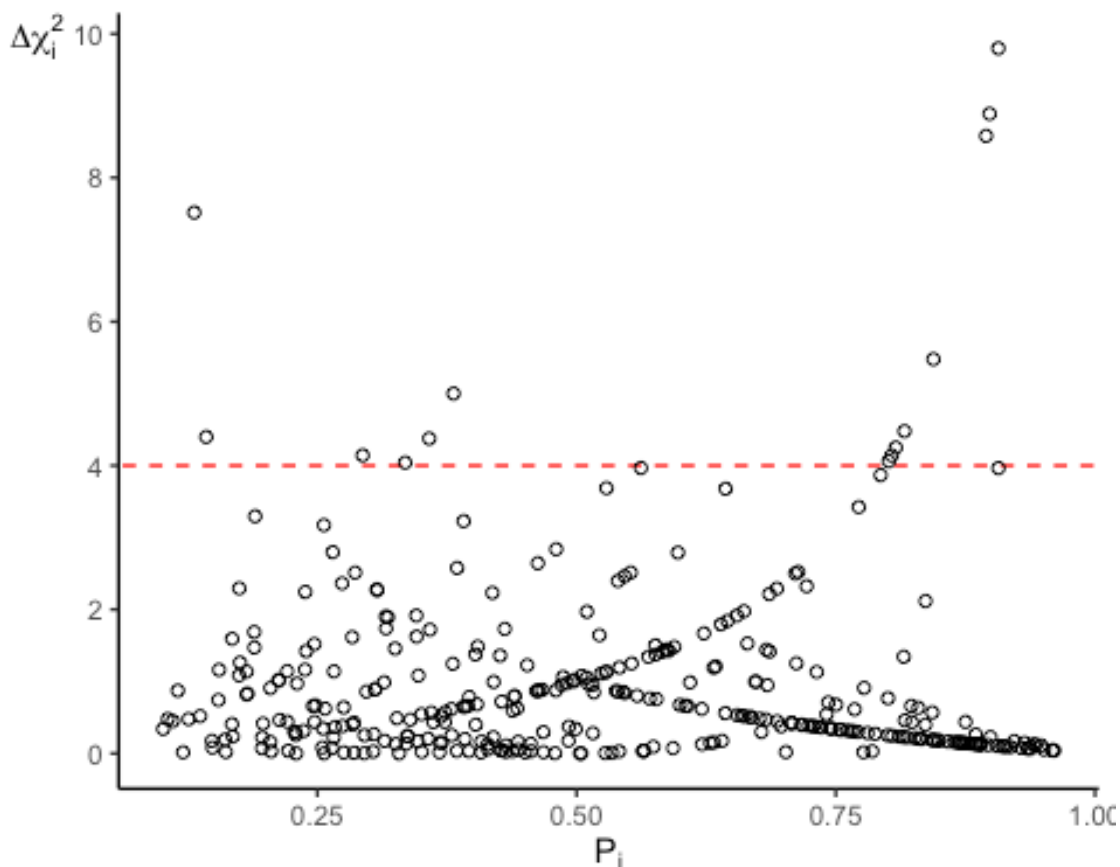
res.dat$dd <- res.dat$deviance^2 / (1 - res.dat$leverage) # voir la formule de delta
deviance dans le livre
# res.dat$db <- res.dat$pearson^2 * res.dat$leverage / (1 - res.dat$leverage)^2 vérifi
cation que c'est la même chose que sdeltabeta
```

## Figures 4 à 6

Elles sont construites à partir des éléments de la table res.dat.

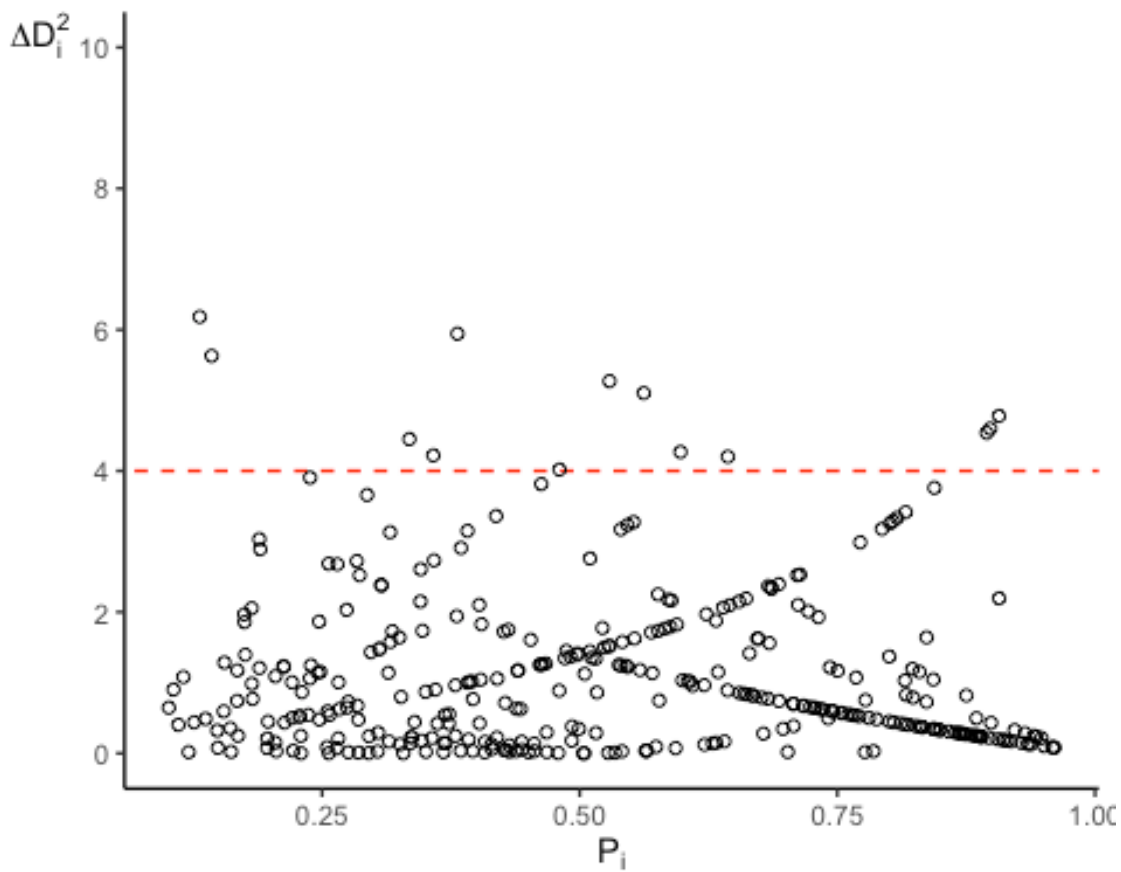
### # Figure 4

```
ggplot(res.dat, aes(pred/n, deltachi)) +
  geom_abline(slope = 0, intercept = 4, linetype = "dashed", color = "red") +
  scale_y_continuous(breaks = seq(0, 10, 2)) +
  labs(x = expression(P[i]), y = expression(paste(Delta, chi[i]^2))) +
  geom_point(shape = 1) +
  theme_classic() +
  theme(axis.title.y = element_text(angle = 0, vjust = 1, hjust = 1))
```



### # Figure 5

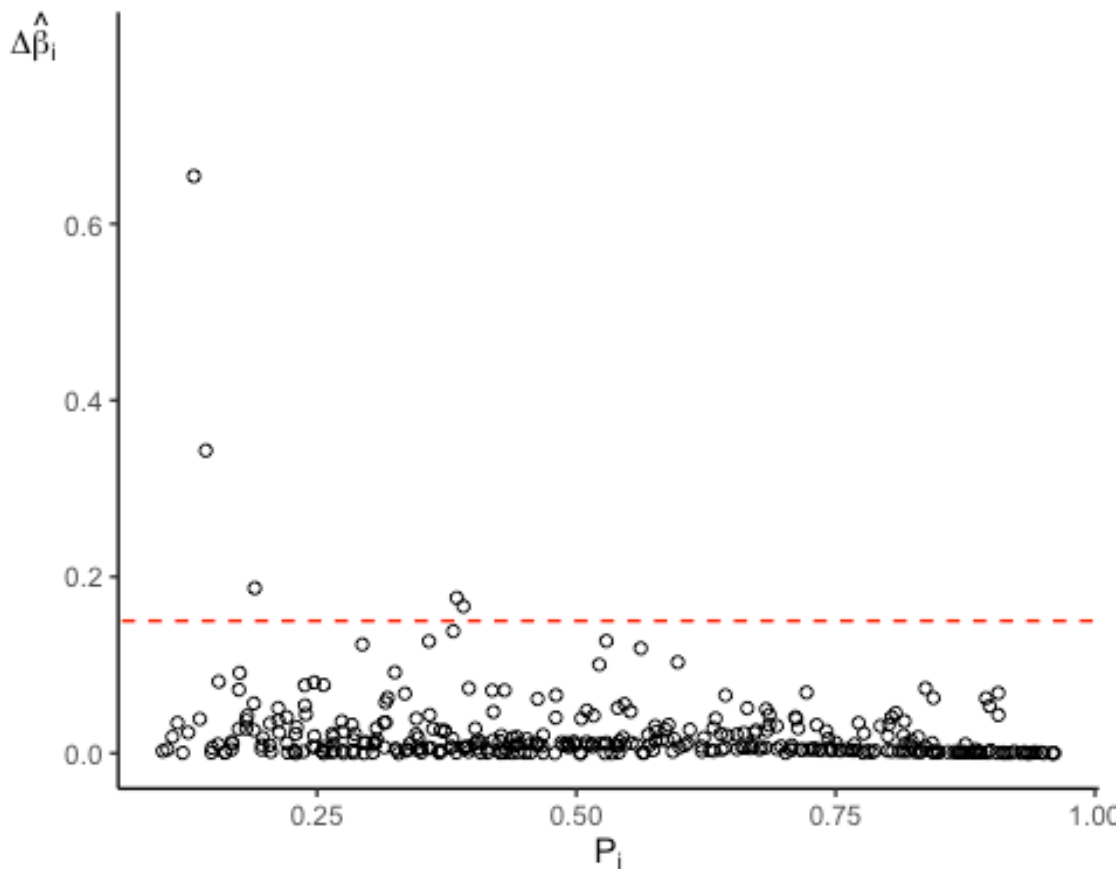
```
ggplot(res.dat, aes(pred/n, dd)) +
  geom_abline(slope = 0, intercept = 4, linetype = "dashed", color = "red") +
  scale_y_continuous(breaks = seq(0, 10, 2), limits = c(0, 10)) +
  labs(x = expression(P[i]), y = expression(paste(Delta, D[i]^2))) +
  geom_point(shape = 1) +
  theme_classic() +
  theme(axis.title.y = element_text(angle = 0, vjust = 1, hjust = 1))
```



# Figure 6

```
ggplot(res.dat, aes(pred/n, sdeltabeta))+
  geom_point(shape=1) +
  geom_abline(slope = 0, intercept = 0.15, linetype = "dashed", color="red") +
  scale_y_continuous(breaks=seq(0,0.7,0.2), limits = c(0, 0.8)) +
  labs(x=expression(P[i]), y=expression(paste(Delta, hat(beta)[i]))) +
  theme_classic() +
  theme(axis.title.y = element_text(angle = 0, vjust = 1, hjust = 1))
```





**Tableau 7 : Caractéristiques des profils tels que delta chi2 ou delta D2 est supérieur à 4**

Pour constituer ce tableau, il faut d'abord "merger" les fichiers pat et res.dat en tenant que la variable identifiant les profils n'y a pas le même nom. On ordonne ensuite le fichier résultant et on sélectionne les profils voulus.

##	cpid	n.x	ctub	agea	tabfc	afcs	ainf	sdeltabeta	deltachi	dd	ps
uc											
## 16	16	56	0	24	0	0	0	0.65401879	7.516421	6.182502	0.250000
00											
## 54	54	3	0	35	1	1	0	0.13834391	5.000711	5.943699	1.000000
00											
## 12	12	59	0	26	0	0	0	0.34311775	4.397991	5.630282	0.050847
46											
## 210	210	4	0	34	3	0	0	0.12729811	3.685990	5.271711	1.000000
00											
## 26	26	3	0	24	2	0	1	0.11898593	3.966640	5.103426	0.000000
00											
## 323	323	1	1	24	3	0	1	0.06830766	9.798163	4.779236	0.000000
00											
## 313	313	1	1	26	2	0	1	0.05364229	8.888340	4.599491	0.000000
00											
## 44	44	1	1	38	3	0	0	0.06236455	8.581396	4.539433	0.000000
00											
## 345	345	2	0	25	0	1	1	0.06760191	4.041114	4.450725	1.000000
00											
## 39	39	4	0	27	2	0	1	0.10320107	2.790833	4.268334	1.000000
00											

```
## 50      50      5      0      34      0      0      1 0.12689558 4.375082 4.220373 0.800000
## 00
## 38      38      2      0      31      2      0      1 0.06566425 3.678330 4.202032 0.000000
## 00
## 101     101      3      0      34      2      0      0 0.06578438 2.837821 4.019082 0.000000
## 00
## 288     288      1      1      42      1      1      0 0.06267448 5.478645 3.760332 0.000000
## 00
## 262     262      6      0      28      0      0      1 0.12327091 4.142553 3.659576 0.666666
## 67
## 126     126      1      1      39      0      0      1 0.03635587 4.478176 3.415843 0.000000
## 00
## 223     223      1      1      27      1      0      1 0.04567650 4.250816 3.334934 0.000000
## 00
## 284     284      1      1      27      2      0      0 0.04086205 4.140261 3.290560 0.000000
## 00
## 348     348      1      1      37      0      0      1 0.03251491 4.060999 3.256204 0.000000
## 00
##          ppred
## 16  0.1313176
## 54  0.3813838
## 12  0.1428665
## 210 0.5288947
## 26  0.5621154
## 323 0.9068062
## 313 0.8983225
## 44  0.8949523
## 345 0.3347491
## 39  0.5977901
## 50  0.3579310
## 38  0.6437362
## 101 0.4803870
## 288 0.8441561
## 262 0.2937203
## 126 0.8162479
## 223 0.8079001
## 284 0.8039138
## 348 0.8011431
```

## Tableau 8 : Variations des coefficients pour les profils les plus influents

Le fichier sur lequel il faut travailler est celui qui contient les sujets et les variables qui ont été utilisés pour le modèle `mod.geu` auquel on ajoute la variable `cpid` qui identifie les profils. On peut l'obtenir à partir du fichier groupé `res.merge` en dupliquant les données de chaque profils autant de fois qu'il y a de sujets dans le profils. C'est ce que fait `map2()`. On sélectionne ensuite les variables utiles, ce qui donne le fichier `geu2`.

```
geu2 <- mutate(.data=res.merge, ct=map2(succes, echec, ~c(rep(1, .x), rep(0, .y))))
geu2 <- unnest(data=geu2, cols=c(ct))
geu2 <- select(geu2, c("cpid", "ct", "ctub", "agea", "tabfc", "afcs", "ainf"))

# ou avec le pipe
#geu2 <- res.merge %>%
#  mutate(ct=map2(succes, echec, ~c(rep(1, .x), rep(0, .y)))) %>%
#  unnest(cols=c(ct)) %>%
#  dplyr::rename(tabfc=as.factor.tabfc.) %>%
```

```
#      select(c("cpid", "ct", "ctub", "agea", "tabfc", "afcs", "ainf"))

# Vérification que Le modèle Logistique est identique avec Les fichiers geu et
# geu2
#
# mod.geu <- glm(ct ~ ctub+agea+as.factor(tabfc)+afcs+ainf, family=binomial, data=geu)
#
# mod.geu2 <- glm(ct ~ ctub+agea+as.factor(tabfc)+afcs+ainf, family=binomial, data=geu2)
#
# summary(mod.geu)
# summary(mod.geu2)
```

Les lignes suivantes calculent les éléments nécessaires au tableau 8, les pourcentages de variations des coefficients, ainsi que les variations observées de D2 et de Chi2 doivent ensuite être calculés “à la main”.

Les profils retenus sont : 16 54 12 210 26 323 313.

```
for (prof in c(0, 16, 54, 12, 210, 26, 323, 313)) {

  mod.geutab8 <- glm(ct ~ ctub+agea+as.factor(tabfc)+afcs+ainf, family=binomial, data=subset(geu2,cpid!=prof))

  cat("\n\nprofil retiré : ", prof, "\t", "nb sujets restants : ", mod.geutab8$df.null+1, "\n\n")
#  print(summary(mod.geutab8))
  HL<-HosmerLemeshow(mod.geutab8,g=10)
#  dd<-mod.geutab8$deviance
#  print(mod.geutab8$coefficients)
  cat("coeff : cste  ctub  agea  tabfc1  tabfc2  tabfc3  afcs  ainf \n",
      "      ", sprintf("%.2f", mod.geutab8$coefficients[1]),
      "", sprintf("%.2f", mod.geutab8$coefficients[2]),
      "", sprintf("%.2f", mod.geutab8$coefficients[3]),
      "", sprintf("%.2f", mod.geutab8$coefficients[4]),
      " ", sprintf("%.2f", mod.geutab8$coefficients[5]),
      " ", sprintf("%.2f", mod.geutab8$coefficients[6]),
      "", sprintf("%.2f", mod.geutab8$coefficients[7]),
      "", sprintf("%.2f", mod.geutab8$coefficients[8]), "\n",
      "chi2 de Hosmer-Lemeshow : ", sprintf("%.2f", HL$chisq))

}

##
##
##
## profil retiré :  0      nb sujets restants :  1682
##
## coeff : cste  ctub  agea  tabfc1  tabfc2  tabfc3  afcs  ainf
##          -3.06  1.83  0.05  0.53   1.32   1.52  0.34  0.82
##  chi2 de Hosmer-Lemeshow :  7.92
##
##
## profil retiré :  16      nb sujets restants :  1626
##
```

```

## coeff : cste ctub agea tabfc1 tabfc2 tabfc3 afcs ainf
##      -3.29 1.84 0.05 0.58 1.38 1.57 0.36 0.84
## chi2 de Hosmer-Lemeshow : 2.72
##
##
## profil retiré : 54  nb sujets restants : 1679
##
## coeff : cste ctub agea tabfc1 tabfc2 tabfc3 afcs ainf
##      -3.02 1.84 0.05 0.49 1.32 1.52 0.31 0.83
## chi2 de Hosmer-Lemeshow : 7.18
##
##
## profil retiré : 12  nb sujets restants : 1623
##
## coeff : cste ctub agea tabfc1 tabfc2 tabfc3 afcs ainf
##      -2.93 1.82 0.05 0.49 1.28 1.47 0.32 0.80
## chi2 de Hosmer-Lemeshow : 5.17
##
##
## profil retiré : 210  nb sujets restants : 1678
##
## coeff : cste ctub agea tabfc1 tabfc2 tabfc3 afcs ainf
##      -3.02 1.84 0.05 0.53 1.32 1.46 0.35 0.82
## chi2 de Hosmer-Lemeshow : 10.44
##
##
## profil retiré : 26  nb sujets restants : 1679
##
## coeff : cste ctub agea tabfc1 tabfc2 tabfc3 afcs ainf
##      -3.03 1.82 0.05 0.53 1.36 1.52 0.33 0.85
## chi2 de Hosmer-Lemeshow : 6.80
##
##
## profil retiré : 323  nb sujets restants : 1681
##
## coeff : cste ctub agea tabfc1 tabfc2 tabfc3 afcs ainf
##      -3.04 1.86 0.05 0.53 1.32 1.54 0.33 0.83
## chi2 de Hosmer-Lemeshow : 7.52
##
##
## profil retiré : 313  nb sujets restants : 1681
##
## coeff : cste ctub agea tabfc1 tabfc2 tabfc3 afcs ainf
##      -3.05 1.86 0.05 0.53 1.34 1.52 0.33 0.83
## chi2 de Hosmer-Lemeshow : 7.66

# Eléments pour la dernière colonne du tableau 8
mod.geutab8 <- glm(ct ~ ctub+agea+as.factor(tabfc)+afcs+ainf, family=binomial,
data=subset(geu2,! (cpid %in% c(16, 54, 12, 210, 26, 323, 313))))
HL<-HosmerLemeshow(mod.geutab8,g=10)
#cat("\n\n\n", "profils retirés : 16, 54, 12, 210, 26, 323, 313", "nb sujets res
tants : ",mod.geutab8$df.null+1, "\n\n")

cat("profils retirés : 16, 54, 12, 210, 26, 323, 313", "nb sujets restants : ",
mod.geutab8$df.null+1, "\n\n",
"coeff : cste ctub agea tabfc1 tabfc2 tabfc3 afcs ainf \n",

```

```

"      ", sprintf("%.2f",mod.geutab8$coefficients[1]),
"", sprintf("%.2f",mod.geutab8$coefficients[2]),
"", sprintf("%.2f",mod.geutab8$coefficients[3]),
"", sprintf("%.2f",mod.geutab8$coefficients[4]),
" ", sprintf("%.2f",mod.geutab8$coefficients[5]),
" ", sprintf("%.2f",mod.geutab8$coefficients[6]),
"", sprintf("%.2f",mod.geutab8$coefficients[7]),
"", sprintf("%.2f",mod.geutab8$coefficients[8]), "\n",
"chi2 de Hosmer-Lemeshow : ",sprintf("%.2f",HL$chisq))

## profils retirés : 16, 54, 12, 210, 26, 323, 313 nb sujets restants : 1555
##
## coeff : cste ctub agea tabfc1 tabfc2 tabfc3 afcs ainf
##      -3.00  1.90  0.05  0.50  1.38  1.51  0.31  0.90
## chi2 de Hosmer-Lemeshow : 3.64

```