



Atelier 7: Modèles linéaires généralisés (mixtes)

Série d'ateliers R

Centre de la Science de la Biodiversité du Québec

À propos de cet atelier



Packages requis

- `ggplot2`
- `lme4`
- `MASS`
- `vcdExtra`
- `bbmle`
- `DescTools`

```
install.packages(c('ggplot2', 'lme4', 'MASS', 'vcdExtra', 'bbmle', 'DescT
```

Objectifs d'apprentissage

1. Pourquoi être normal?
2. GLM avec variable binaire
3. GLM avec des données d'abondance
4. GLM avec effet aléatoires

Pourquoi être normal?

Y aurait-il un meilleur modèle?

Limitations des modèles linéaires (mixtes)

Charger les données et appliquer un modèle linéaire (`lm()`) :

```
# vérifiez que vous êtes dans le bon répertoire de travail  
mites ← read.csv('data/mites.csv')  
head(mites)  
str(mites)
```

Le jeu de données chargé contient une partie du jeu de données classique des **mites Oribatid (Acaria, Oribatei)** des sphaignes (Sphagnum sp.) du Lac Geai, **Station de Biologie de l'Université de Montréal** :

70 échantillons de mousses et mites

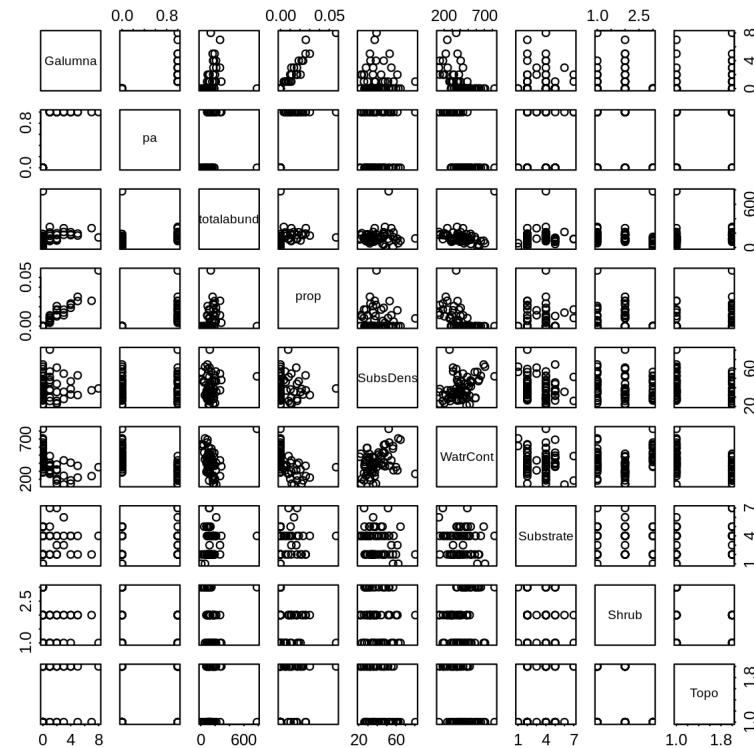
5 variables environnementales, abondance de la mite *Galumna sp.*, et abondance totale des mites

Objectif: Modéliser l'abondance (`abund`), l'occurrence (`pa`), et la proportion (`prop`) de Galumna en fonction des 5 paramètres environnementaux.

Explorer les relations

Pouvons-nous voir une/des relation(s) entre *Galumna* et les 5 variables environnementales?

```
plot(mites)
```



Galumna VS
WatrCont??!

Explorer les relations

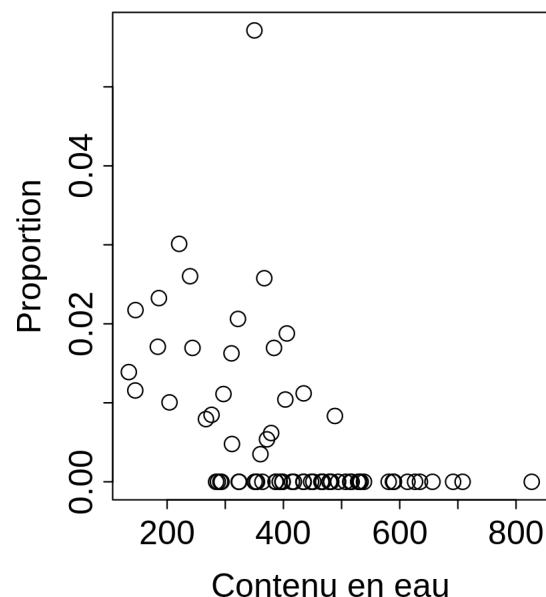
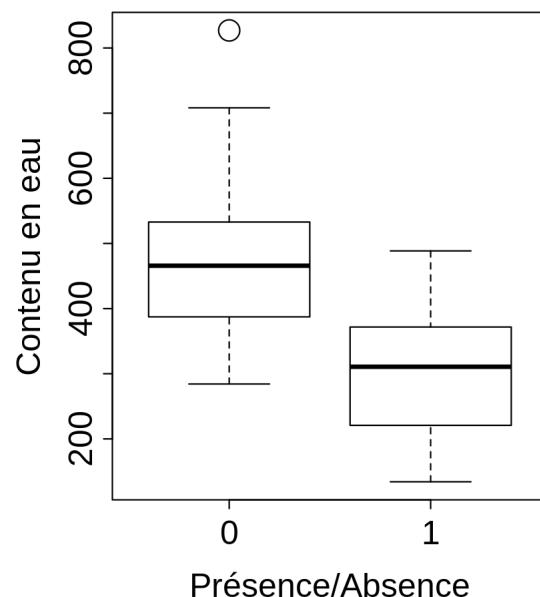
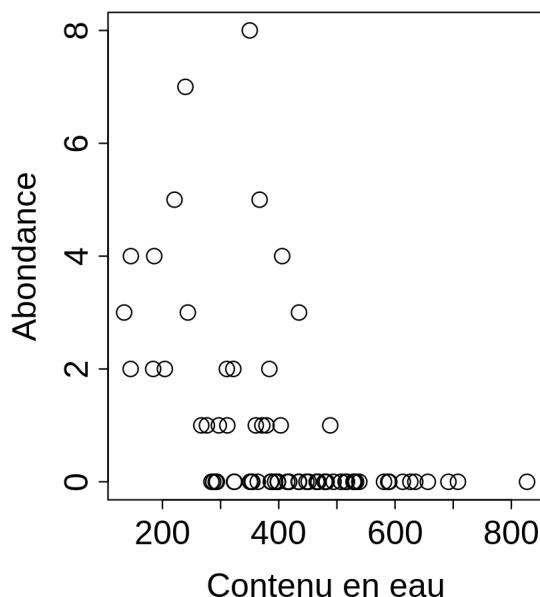
Une relation négative entre Galumna et le contenu en eau du sol?

```
par(mfrow = c(1, 3), cex = 1.4)
```

```
plot(Galumna ~ WatrCont, data = mites, xlab = 'Contenu en eau', ylab= 'Abondance')
```

```
boxplot(WatrCont ~ pa, data = mites, xlab= 'Présence/Absence', ylab = 'Contenu en eau')
```

```
plot(prop ~ WatrCont, data = mites, xlab = 'Contenu en eau', ylab= 'Proportion')
```



Tester la linéarité

Utiliser des modèles linéaires pour voir si l'abondance, `abund`, la présence/absence, `pa`, et/ou la proportion, `prop`, varient en fonction du contenu d'eau.

```
lm.abund ← lm(Galumna ~ WatrCont, data = mites)
## summary(lm.abund)
lm.pa ← lm(pa ~ WatrCont, data = mites)
## summary(lm.pa)
lm.prop ← lm(prop ~ WatrCont, data = mites)
## summary(lm.prop)
```

Tester la linéarité

Utiliser des modèles linéaires pour voir si l'abondance, `abund`, la présence/absence, `pa`, et/ou la proportion, `prop`, varient en fonction du contenu d'eau.

```
summary(lm.abund)$coefficients[,  
# (Intercept) WatrCont  
# 3.981563e-08 1.206117e-05  
summary(lm.abund)$coefficients[,  
# (Intercept) WatrCont  
# 3.981563e-08 1.206117e-05  
summary(lm.abund)$coefficients[,  
# (Intercept) WatrCont  
# 3.981563e-08 1.206117e-05
```

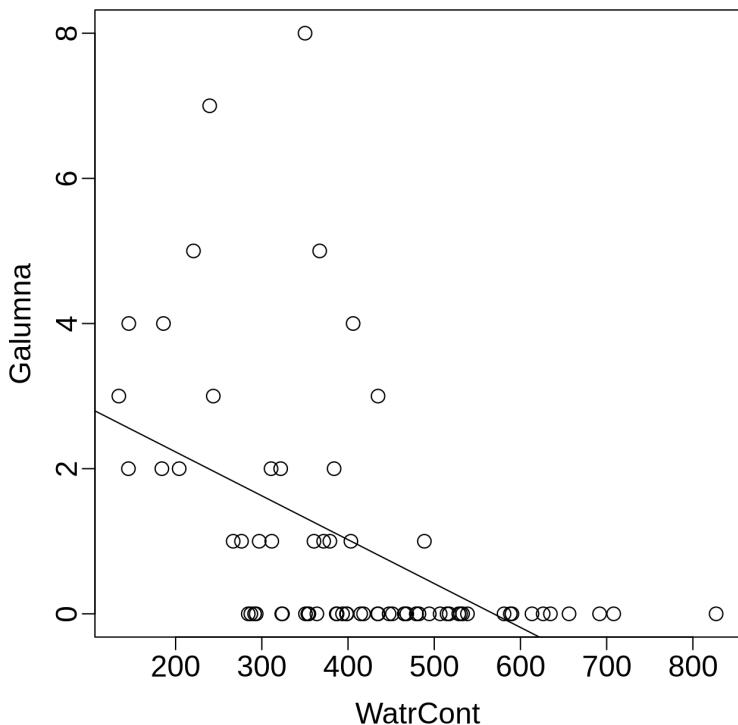
Un effet significatif dans tous les modèles!

Mais...

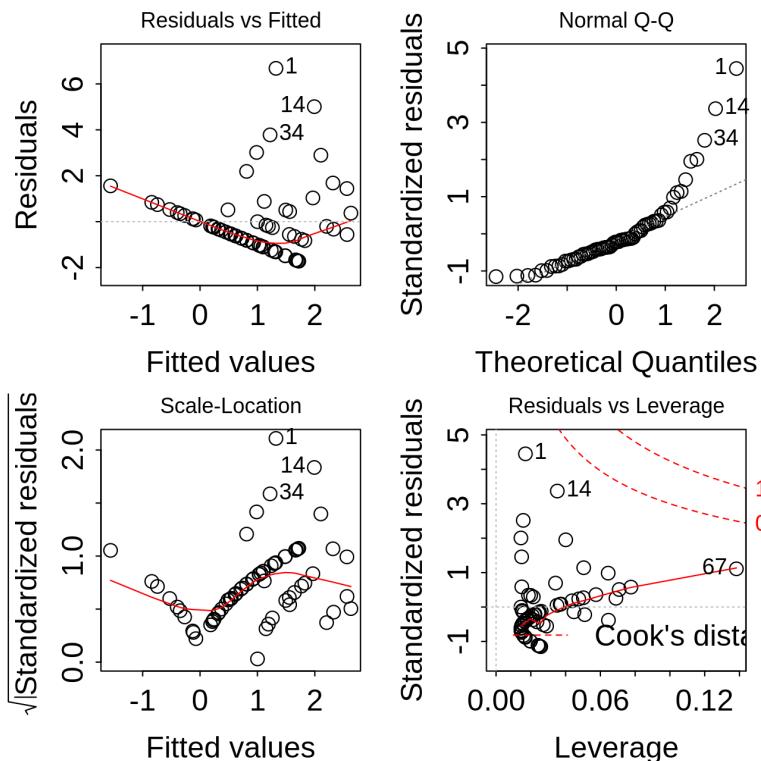
Tester la linéarité

Un effet significatif dans tous les modèles! **Attends une minute...**

```
plot(Galumna ~ WatrCont, data = m  
abline(lm.abund)
```



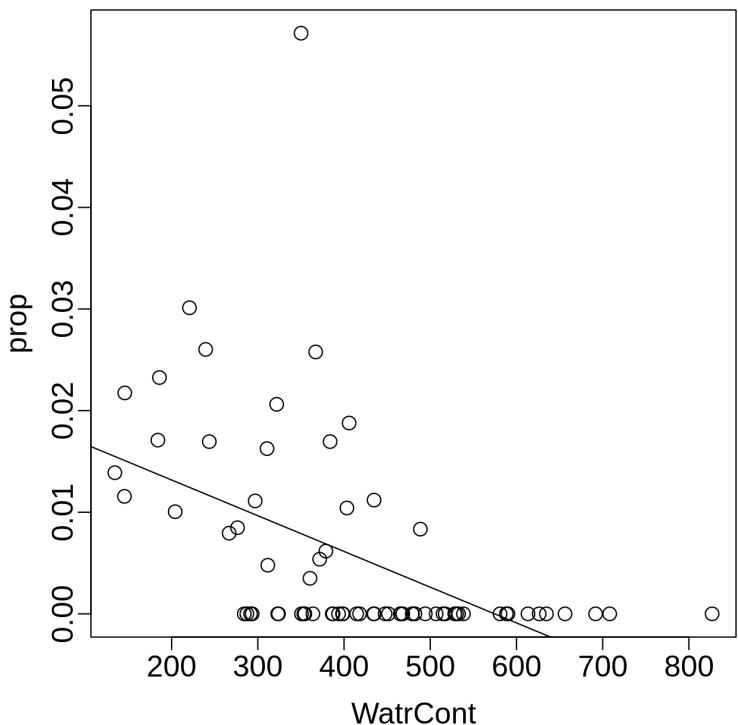
```
par(mfrow = c(2, 2), cex = 1.4)  
plot(lm.abund)
```



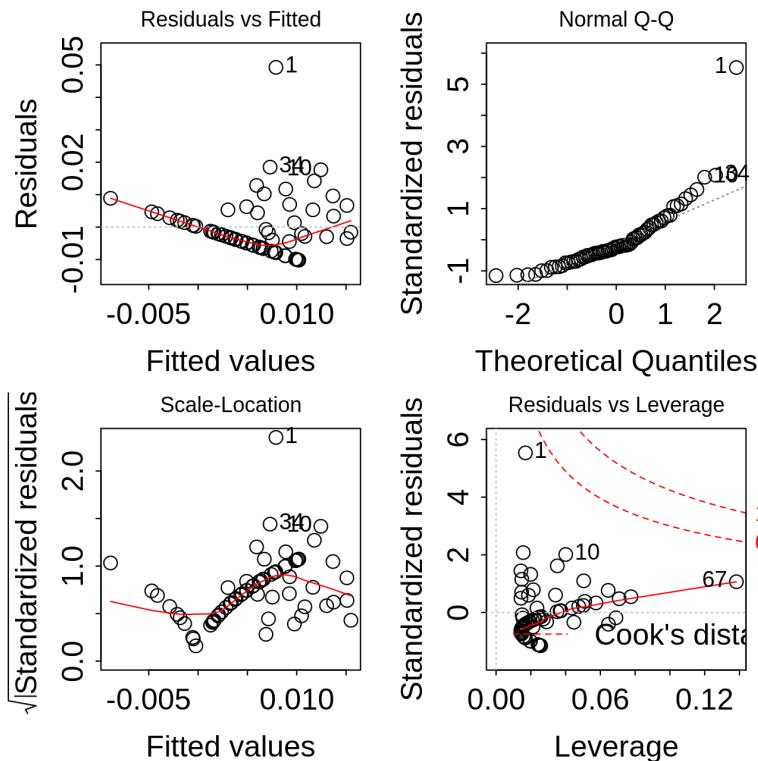
Tester la linéarité

Encore pire pour les autres modèles (Proportion `prop`) :

```
plot(prop ~ WatrCont, data = mite  
abline(lm.prop))
```



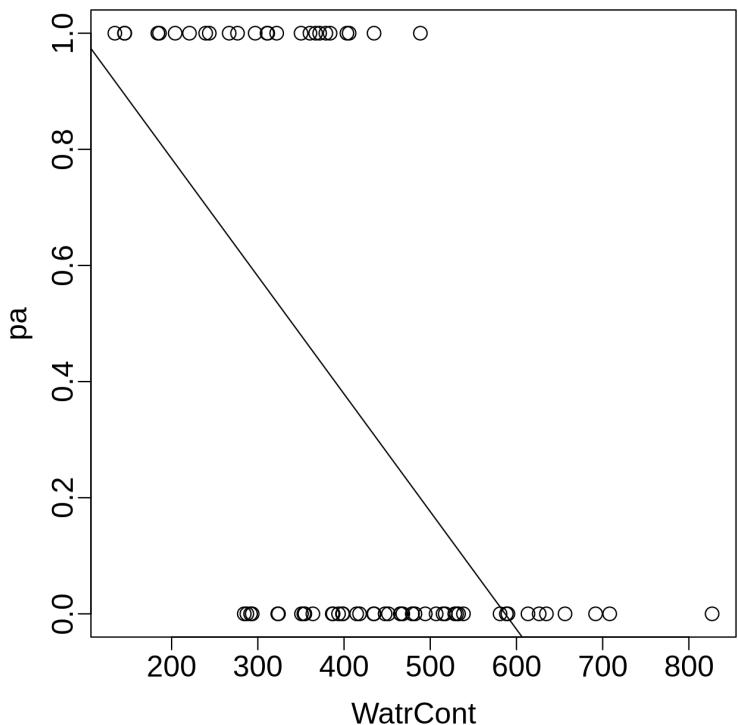
```
par(mfrow = c(2, 2), cex = 1.4)  
plot(lm.prop)
```



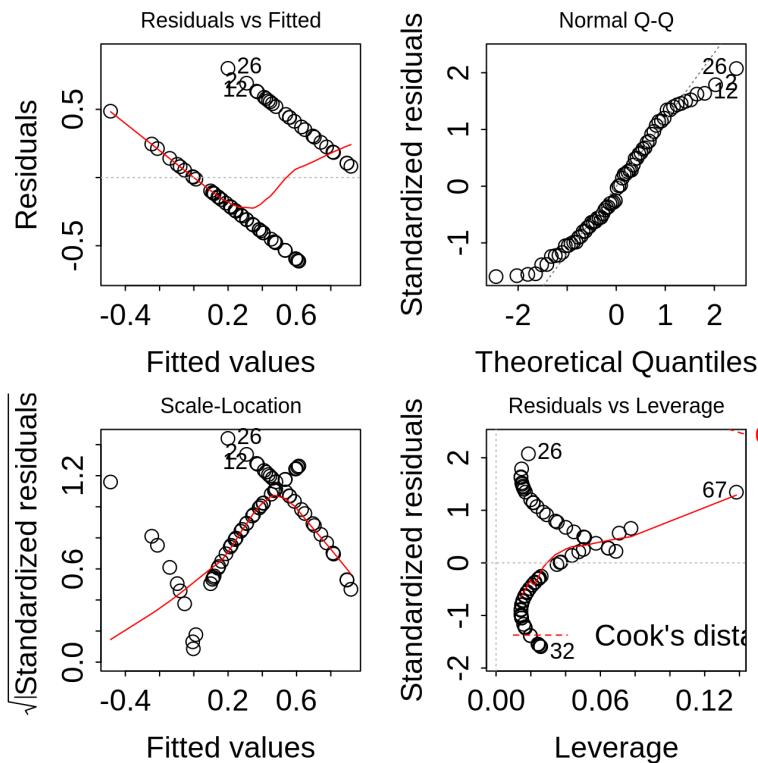
Tester la linéarité

Encore pire pour les autres modèles (Présence/Absence `pa`) :

```
plot(pa ~ WatrCont, data = mites)
abline(lm.pa)
```



```
par(mfrow = c(2, 2), cex = 1.4)
plot(lm.pa)
```



Conditions d'application du modèle linéaire

Il est **très commun** en écologie que les conditions d'application du modèle linéaire ne soient pas respectées. C'est pourquoi nous avons souvent besoin des **modèles linéaires généralisées** (GLMs).

Rafraîchissons-nous la mémoire à propos des conditions d'application du modèle linéaire.

Conditions d'application du modèle linéaire

Équation du modèle:

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$$

où:

- y_i = valeur estimée pour la $i^{\text{ème}}$ variable réponse,
- β_0 = ordonnée à l'origine de la droite,
- β_1 = pente,
- x_i = $i^{\text{ème}}$ valeur de la variable observée.
- ε_i = résidus du modèle obtenus d'une distribution normale de moyenne 0 et de variance constante (qui est à estimer).

Distribution normale

Une autre manière d'écrire le modèle linéaire est :

$$Y_i \sim \mathcal{N}(\mu = \beta_0 + \beta_1 x_i, \sigma^2)$$

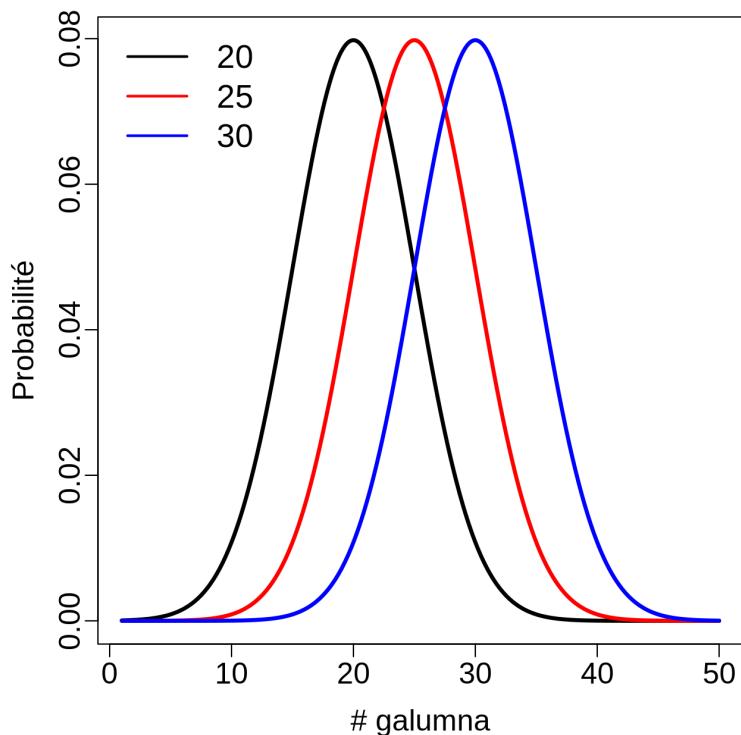
Ce qui signifie que y_i (réalisation de la variable aléatoire Y_i) est échantillonnée dans une distribution normale ayant les paramètres μ (dont la valeur dépend de x_i) et σ (indépendantes de x_i)

Essayons de prédire l'abondance de `Galumna` en fonction du contenu d'eau en utilisant `lm()` que nous avons vu plus tôt.

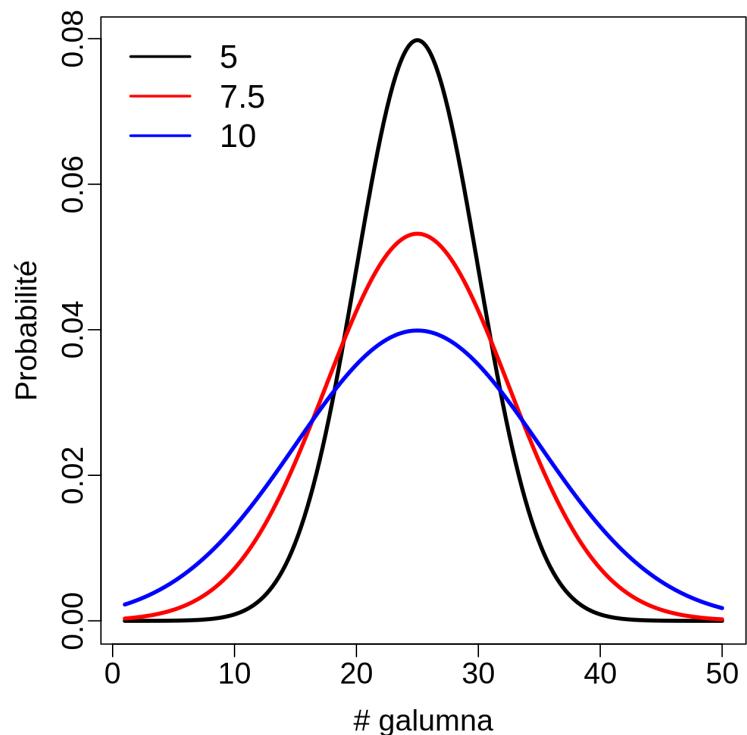
Distribution normale

Rappel : La distribution normale a deux paramètres, μ (moyenne) et σ (variance)

3 valeurs pour $\mu, \sigma = 5$



$\mu = 25$, 3 valeurs pour σ



Distribution normales des résidus

Pour utiliser correctement le modèle linéaire, il faut s'assurer que les résidus sont distribués selon une loi normale centrée.

Il faut donc **vérifier la normalité des résidus** et non celle la distribution de la variable de réponse, comme le Andrew MacDonald dans un tweet à propos de cette dernière pratique:

This is the wrong thing to do because what matters is how your "response variable" is distributed *AFTER* you do your model, not before! [@rlmcelreath](#) called this practice "histomancy", which name I love

— Andrew MacDonald  (@polesasunder) [February 5, 2020](#)

Prédiction du modèle

Nous avons besoin des coefficients de régression (β_0 et β_1) et de σ :

```
coef(lm.abund)
# (Intercept)      WatrCont
# 3.439348672 -0.006044788
summary(lm.abund)$sigma
# [1] 1.513531
```

Quels sont les paramètres de la distribution normale utilisés pour modéliser y lorsque le contenu d'eau = 300?

$$y_i \sim N(\mu = \beta_0 + \beta_1 x_i, \sigma^2)$$

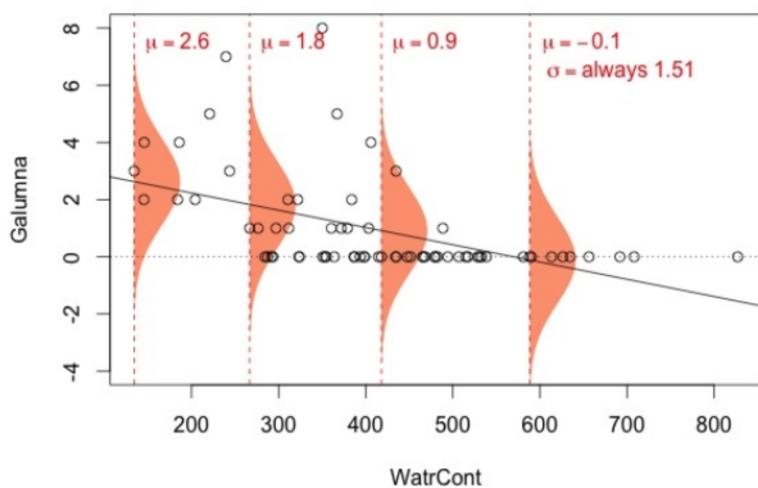
$$\mu = 3.44 + (-0.006 \times 300) = 1.63$$

$$\sigma = 1.51$$

Prédiction du modèle

- Lorsque $x_i = 300$, Y_i suit une distribution normale avec $\mu = 1.63$ et $\sigma^2 = 1.51$.
- Lorsque $x_i = 400$, Y_i suit une distribution normale avec $\mu = 1.02$ et $\sigma^2 = 1.51$, etc.

Graphiquement, notre modèle ressemble à :



Problèmes:

- σ^2 n'est pas homogène, mais `lm()` nous constraint d'utiliser toujours la même valeur de σ^2 ,
- Les valeurs estimées devraient être des nombres entiers.

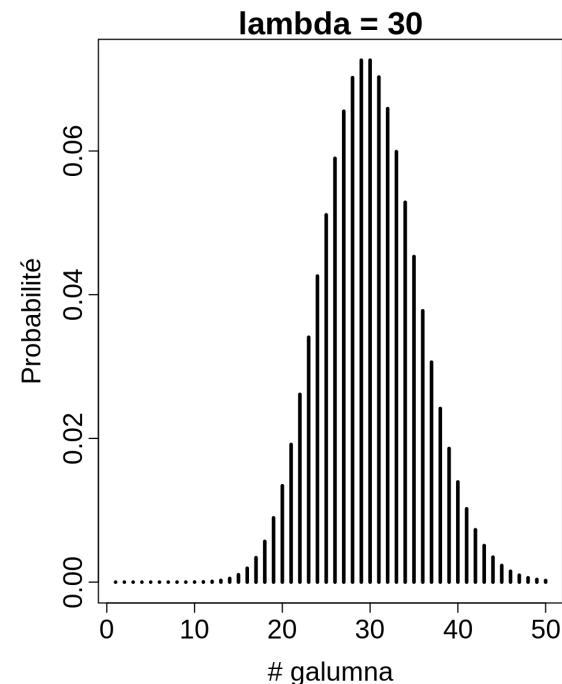
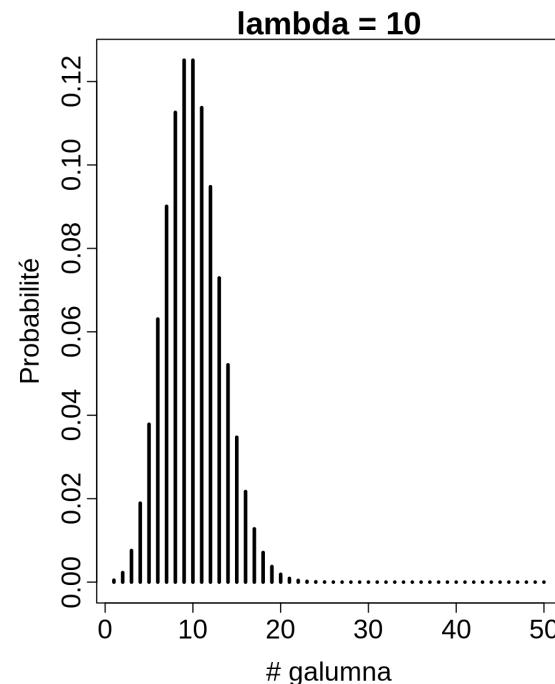
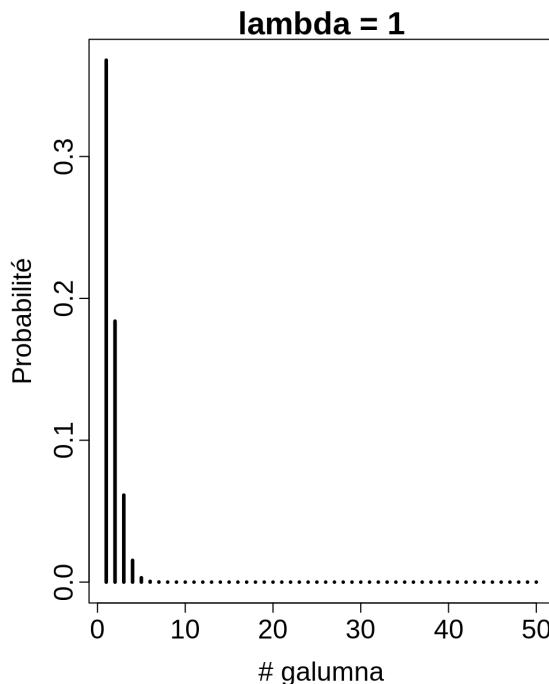
Données biologiques & distributions

- Les statisticiens ont développé **de nombreuses lois de probabilité (distributions)** correspondant à divers types de données
- Une distribution donne la probabilité d'observer chaque issue possible d'une expérience ou échantillonage (e.g. *abund* = 8 Galumna)
- Les distributions peuvent être **discrètes** (que des nombres entiers) ou **continues** (incluent aussi des fractions)
- Toutes les distributions ont des **paramètres** qui déterminent leur forme (e.g. μ et σ^2 pour la distribution normale)

Données biologiques & distributions

L'abondance de *Galumna* suit une distribution discrète (que des nombres entiers). Pour modéliser les données d'abondance, la **loi de Poisson** est souvent utilisée:

- une distribution discrète avec un seul paramètre, λ (lambda), qui détermine la moyenne et la variance de la distribution:



Données biologiques & distributions

Galumna semble suivre une loi de Poisson avec une faible valeur de λ :

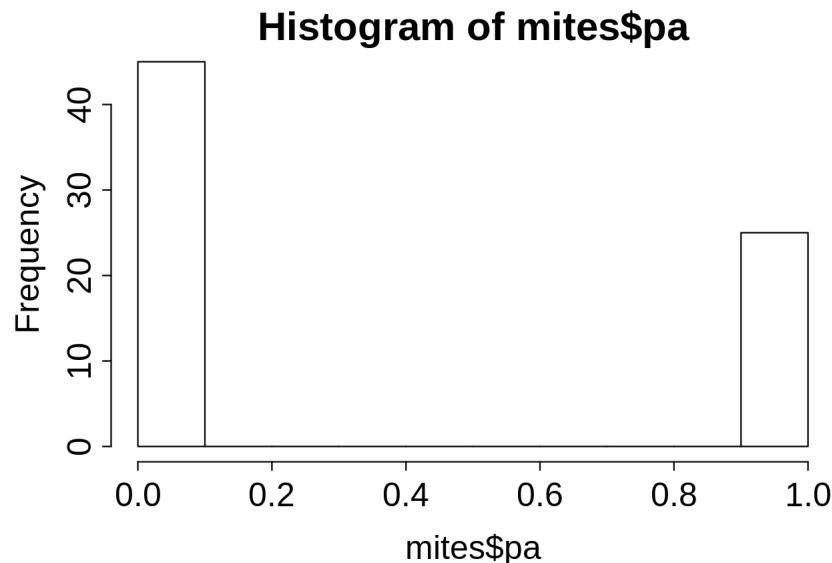
```
mean(mites$Galumna)
# [1] 0.9571429
hist(mites$Galumna)
```

Données biologiques & distributions

Présence-absence par contre prend une autre forme :

- Inclut seulement des 0s et des 1s
- La loi de Poisson n'est pas appropriée pour cette variable

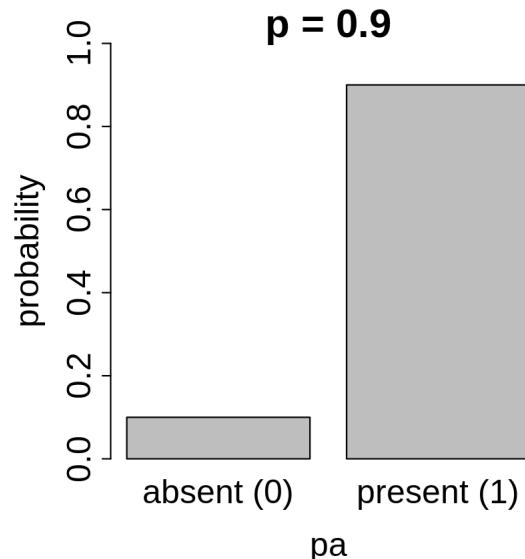
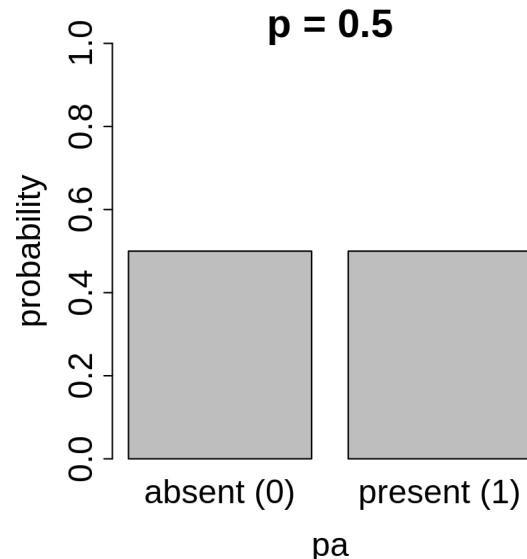
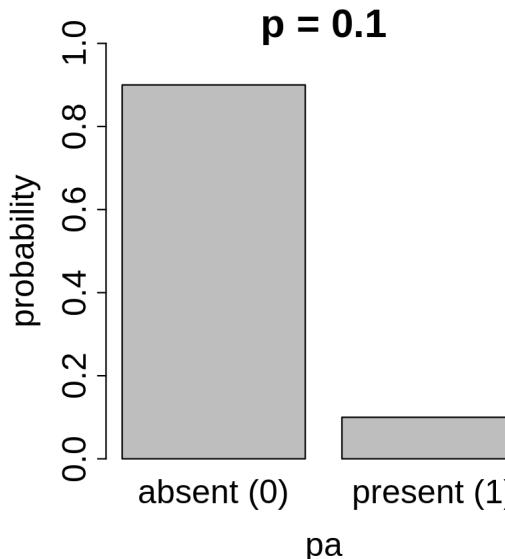
```
hist(mites$pa)
```



Données biologiques & distributions

Distribution de Bernoulli :

- N'inclut que deux issues possibles dans son ensemble: succès (1) ou échec (0)
- N'a qu'un paramètre, p , la probabilité de succès

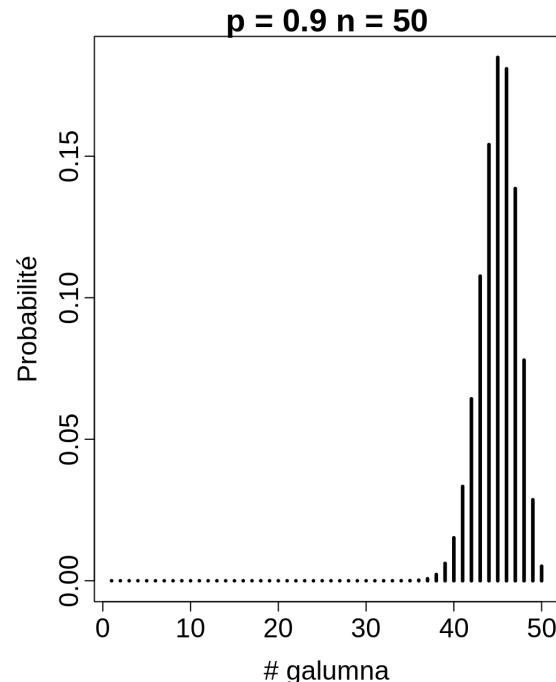
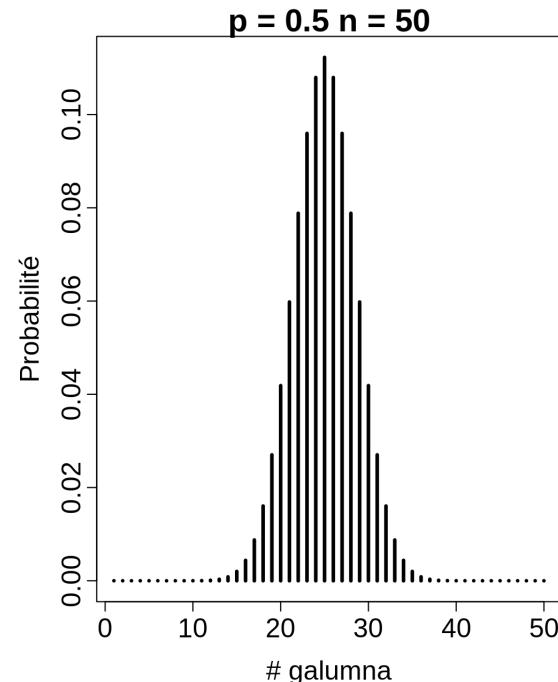
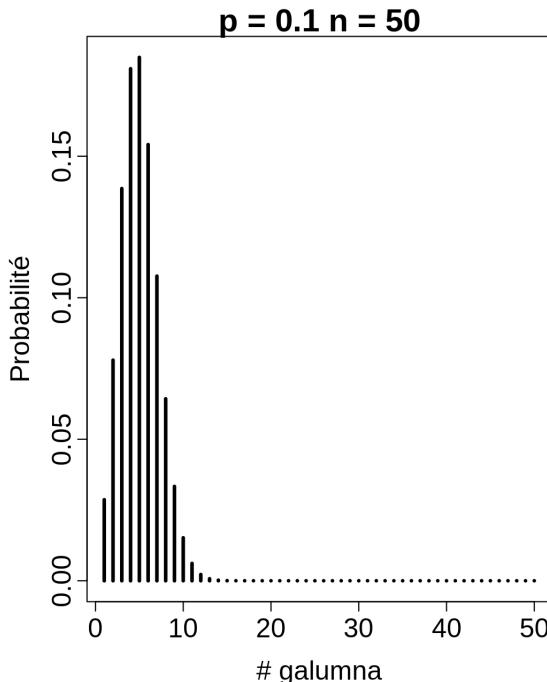


Nous pouvons utiliser la loi de Bernoulli pour calculer la probabilité d'obtenir l'issue "Galumna present" (1) vs. "Galumna absent" (0)

Données biologiques & distributions

Distribution binomiale : Lorsqu'il y a plusieurs épreuves (chacune avec un succès/échec), la loi de Bernoulli se transforme en loi binomiale

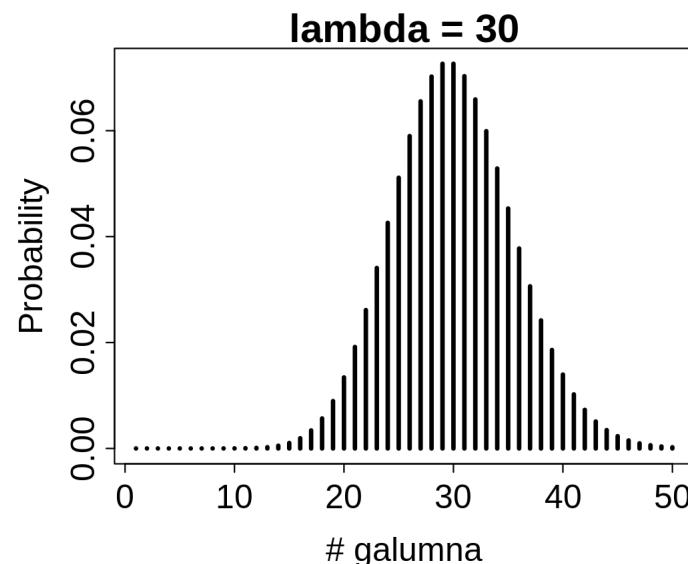
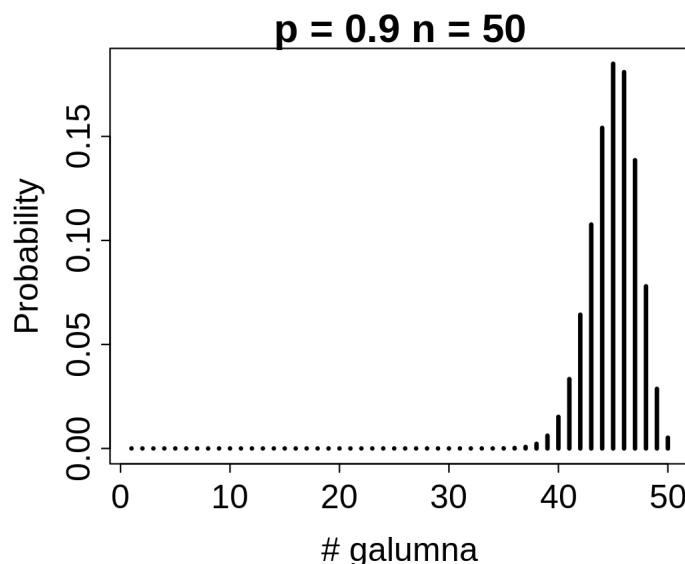
- Inclut le paramètre additionnel n , le nombre d'épreuves
- Prédit la probabilité d'observer une certaine proportion de succès, p , sur le nombre total d'épreuves, n



Données biologiques & distributions

Distribution binomiale : utilisée pour modéliser des données lorsque le nombre de succès est donné par un nombre entier, et lorsque le nombre d'épreuves, n , est connu.

Différence principale avec la loi de Poisson : L'étendue de la loi binomiale a une limite supérieure, n . Par conséquent, elle est asymétrique et décalée à gauche lorsque p est faible, mais décalée à droite lorsque p est élevé.



Données biologiques & distributions

Retournons à notre problème pour changer la distribution de Y_i de normale à Poisson :

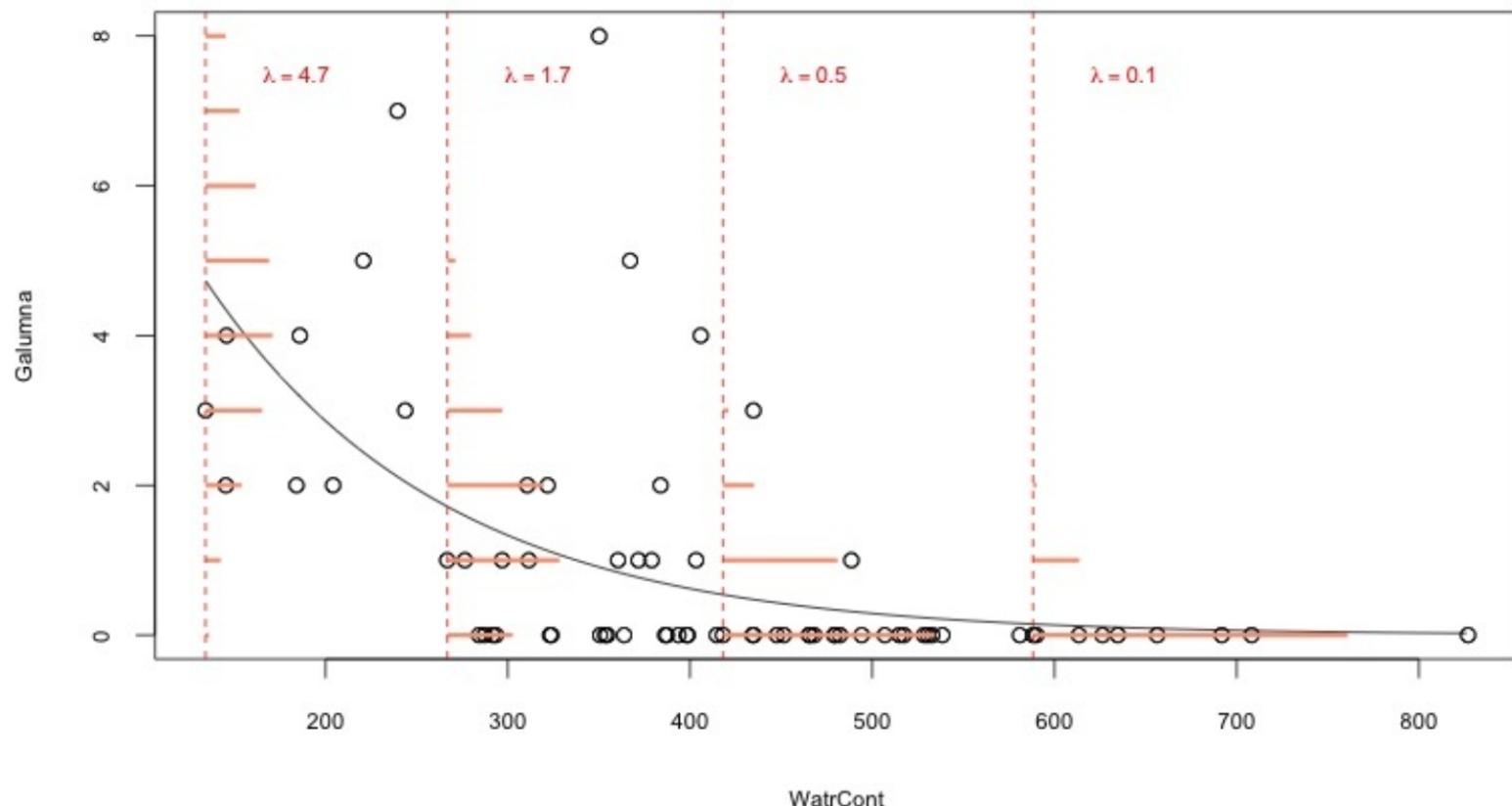
$$Y_i \sim \text{Poisson}(\lambda = \beta_0 + \beta_1 x_i)$$

Le problème est résolu!

1. λ varie avec x (contenu d'eau) et la variance des résidus changera aussi avec x , et nous venons de nous défaire de la supposition d'homogénéité de la variance!
2. Les valeurs estimées seront des nombres entiers plutôt que des nombres décimaux;
3. Ce modèle ne prédira jamais de valeurs négatives (Poisson est toujours strictement positif).

Données biologiques & distributions

Ce modèle est **presque** un GLM de Poisson, qui ressemble à ça:



Les probabilités (en orange) sont maintenant des nombres entiers, et la variance et la moyenne de la distribution déclinent lorsque λ diminue avec le contenu d'eau.

GLM avec variable binaire

Variables binaires

Une variable réponse commune dans les jeux de données en écologie : variable binaire, on observe un phénomène X ou son "absence"

- Présence/Absence d'une espèce
- Présence/Absence d'une maladie
- Succès/Échec d'observer un comportement
- Survie/Mort d'un organisme

On veut déterminer si $P/A \sim Environment$

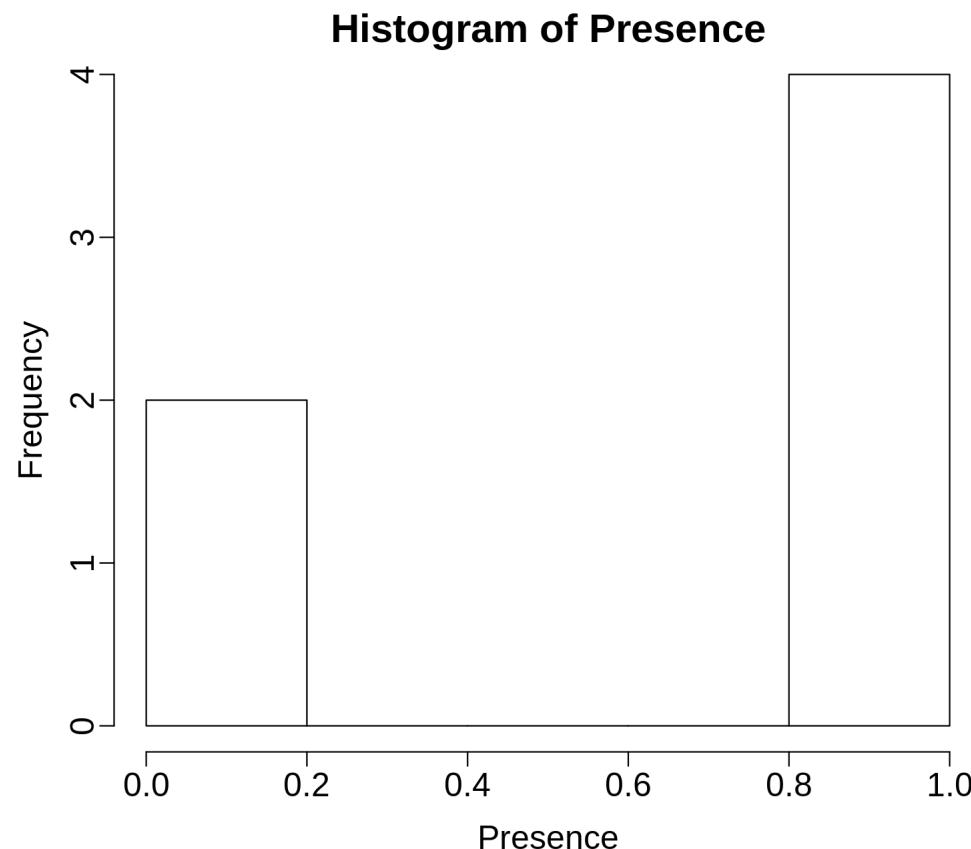
Régression logistique ou modèle logit

Variables binaires

Dans `R`, on code une variable binaire avec `1` et `0`:

#	Site	Presence	
# 1	A	1	1 = Présence
# 2	B	0	
# 3	C	1	
# 4	D	1	0 = Absence
# 5	E	0	
# 6	F	1	

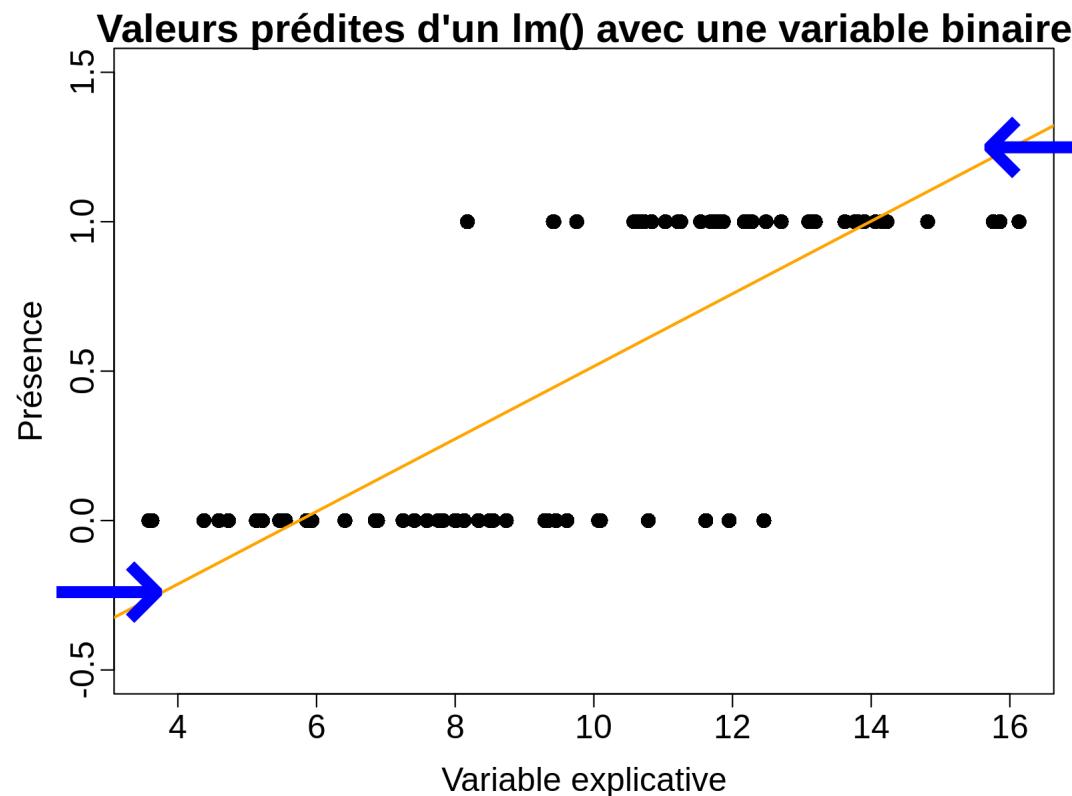
Variables binaires



Avec un modèle linéaires, les résidus ne seront pas distribués normalement!

Variables binaires

Les valeurs prédictes peuvent se trouver hors de l'intervalle $[0, 1]$ avec `lm()`:



Distribution de probabilité

La distribution de Bernoulli est adaptée pour des variables réponses binaires

$$E(Y) = p$$

→ **Moyenne de la distribution**

Probabilité (p) d'observer un résultat

$$Var(Y) = p \times (1 - p)$$

→ **Variance de la distribution** La variance décroît quand (p) est proche de 0 ou 1

Régression logistique

La fonction `glm()`!

```
logit.reg ← glm(formula, data, family)
```

Pour utiliser la bonne distribution, il faut spécifier:

1. distribution de probabilité **ET**
2. une **fonction de lien** (il y a une fonction de lien par défaut pour une distribution donnée!) qui définit aussi une relation entre la moyenne et la variance.

Dans R, cela se fait en utilisant l'argument `family` (voir `?family`):

La fonction de lien

Pour un modèle linéaire de base d'une variable réponse normalement distribuée, l'équation pour la valeur attendue est :

$$\mu = x\beta$$

où

- μ représente les valeurs prédites par le modèle,
- x est la matrice du modèle (*i.e.* les variables explicatives),
- β est le vecteur des paramètres estimés (*i.e.* intercept & pente).

$x\beta$ est appelé le **prédicteur linéaire**

La fonction de lien

$\mu = x\beta$ est vrai seulement pour la distribution normale

Si ce n'est pas le cas, on utilise une transformation g sur μ

$$g(\mu) = x\beta$$

où g est appelée **fonction de lien**. Cela permet de travailler avec des distributions autres que la loi normale.

La fonction de lien

Pour les données binaires, on utilise la transformation **logit** :

$$g(\mu) = \log\left(\frac{\mu}{1-\mu}\right)$$

μ = variables prédites (probabilité que $Y = 1$)

1. $\frac{\mu}{1-\mu}$ met les valeurs prédites sur une échelle de 0 à $+\text{Inf}$
2. La transformer en log ce qui projette les valeurs prédites sur entre $-\text{Inf}$ à $+\text{Inf}$

→ Les valeurs prédites **transformées** sont reliées **linéairement** au prédicteur linéaire

Exercice 1

Spécifiez un modèle de régression logistique avec le jeu de données mites.

```
#setwd('...')  
mites <- read.csv("mites.csv", header = TRUE)  
str(mites)  
  
# 'data.frame':    70 obs. of  9 variables:  
# $ Galumna     : int  8 3 1 1 2 1 1 1 2 5 ...  
# $ pa          : int  1 1 1 1 1 1 1 1 1 1 ...  
# $ totalabund: int  140 268 186 286 199 209 162 126 123 166 ...  
# $ prop         : num  0.05714 0.01119 0.00538 0.0035 0.01005 ...  
# $ SubsDens    : num  39.2 55 46.1 48.2 23.6 ...  
# $ WatrCont    : num  350 435 372 360 204 ...  
# $ Substrate   : Factor w/ 7 levels "Barepeat","Interface",..: 4 3 2 4 4 4  
# $ Shrub        : Factor w/ 3 levels "Few","Many","None": 1 1 1 1 1 1 1 2 2  
# $ Topo         : Factor w/ 2 levels "Blanket","Hummock": 2 2 2 2 2 2 2 1 1
```

Exercice 1

Spécifiez un modèle de la présence et de l'absence de *Galumna sp.* en fonction du contenu en eau du sol et de la topographie.

```
logit.reg ← glm(pa ~ WatrCont + Topo, data=mites,  
family = binomial(link = "logit"))  
  
summary(logit.reg)
```

Exercice 1

```
summary(logit.reg)
#
# Call:
# glm(formula = pa ~ WatrCont + Topo, family = binomial(link = "logit"),
#      data = mites)
#
# Deviance Residuals:
#       Min        1Q     Median        3Q        Max
# -2.0387 -0.5589 -0.1594  0.4112  2.0252
#
# Coefficients:
#             Estimate Std. Error z value Pr(>|z|)
# (Intercept) 4.464402  1.670622  2.672 0.007533 **
# WatrCont    -0.015813  0.004535 -3.487 0.000489 ***
# TopoHummock 2.090757  0.735348  2.843 0.004466 **
# ---
# Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# (Dispersion parameter for binomial family taken to be 1)
#
```

Défi 1



En utilisant le jeu de données 'bacteria', spécifiez un modèle de la présence de *H. influenzae* en fonction du traitement et de la semaine de test.

Commencez avec un modèle saturé et trouvez le modèle le plus parcimonieux.

```
#install.packages("MASS")
library(MASS)
data(bacteria)
str(bacteria)
# 'data.frame': 220 obs. of 6 variables:
# $ y    : Factor w/ 2 levels "n", "y": 2 2 2 2 2 2 1 2 2 2 ...
# $ ap   : Factor w/ 2 levels "a", "p": 2 2 2 2 1 1 1 1 1 1 ...
# $ hilo : Factor w/ 2 levels "hi", "lo": 1 1 1 1 1 1 1 1 2 2 ...
# $ week: int 0 2 4 11 0 2 6 11 0 2 ...
# $ ID   : Factor w/ 50 levels "X01", "X02", "X03", ..: 1 1 1 1 2 2 2 2 3 3 ...
# $ trt  : Factor w/ 3 levels "placebo", "drug", ..: 1 1 1 1 3 3 3 3 2 2 ...
```

Solution



```
model.bact1 <- glm(y ~ trt * week, data = bacteria, family = binomial)

model.bact2 <- glm(y ~ trt + week, data = bacteria, family = binomial)

model.bact3 <- glm(y ~ week, data = bacteria, family = binomial)

anova(model.bact1, model.bact2, model.bact3, test = "LRT")
# Analysis of Deviance Table
#
# Model 1: y ~ trt * week
# Model 2: y ~ trt + week
# Model 3: y ~ week
#   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
# 1       214    203.12
# 2       216    203.81 -2   -0.6854  0.70984
# 3       218    210.91 -2   -7.1026  0.02869 *
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Interpréter la sortie

Regardez à nouveau les coefficients du modèle `logit.reg` :

```
summary(logit.reg)$coefficients
#               Estimate Std. Error   z value Pr(>|z|)
# (Intercept) 4.46440199 1.670622482 2.672299 0.0075333598
# WatrCont    -0.01581255 0.004535069 -3.486728 0.0004889684
# TopoHummock 2.09075654 0.735348234  2.843220 0.0044660283
```

La sortie indique que le contenu d'eau et la topographie sont significatifs
Mais comment interprète-on les coefficients de la pente?

Interpréter la sortie

Rappelez-vous que nous avons utilisé une transformation logit!

Pour bien interpréter les coefficients du modèle, il faut les utiliser avec la fonction de lien inverse (g^{-1}):

$$g^{-1}(\beta_0 + \beta_1 x)$$

Dans notre exemple, la fonction *logit* inverse est utiliser pour obtenir les probabilités :

$$\text{logit}^{-1} = \frac{1}{1 + e^{-(\beta_0 + \beta_1 \text{WatrCont} + \beta_2 \text{TopoHummock})}}$$

Pouvoir prédictif et ajustement du modèle

Le **pseudo- R^2** , un concept analogue au R^2 pour les modèles estimés par maximisation de la vraisemblance:

$$\text{pseudo-}R^2 = \frac{\text{déviance nulle} - \text{déviance résiduelle}}{\text{déviance nulle}}$$

pseudo- R^2 = variance expliquée par le modèle

Pouvoir prédictif et ajustement du modèle

Comparer la déviance du modèle (déviance résiduelle) à la déviance d'un modèle nul (déviance nulle)

Le **modèle nul** est un modèle sans variables explicatives, simplement l'ordonnée à l'origine

```
null.model ← glm(Response.variable ~ 1, family = binomial)
```

Pouvoir prédictif et ajustement du modèle

Dans R, nous pouvons extraire les déviances résiduelles et nulles directement à partir de l'objet `glm` :

```
objects(logit.reg)
# [1] "aic"                      "boundary"           "call"
# [4] "coefficients"             "contrasts"          "control"
# [7] "converged"                 "data"                "deviance"
# [10] "df.null"                  "df.residual"        "effects"
# [13] "family"                   "fitted.values"     "formula"
# [16] "iter"                     "linear.predictors" "method"
# [19] "model"                    "null.deviance"    "offset"
# [22] "prior.weights"           "qr"                 "R"
# [25] "rank"                     "residuals"          "terms"
# [28] "weights"                  "xlevels"            "y"
```

Pouvoir prédictif et ajustement du modèle

Dans R, nous pouvons extraire les déviances résiduelles et nulles directement à partir de l'objet `glm` :

```
pseudoR2 ← (logit.reg$null.deviance - logit.reg$deviance) / logit.reg$nu  
pseudoR2  
# [1] 0.4655937
```

Ainsi, le modèle explique 46.6% de la variabilité des données

Pouvoir prédictif et ajustement du modèle

Nouvelle statistique - **coefficient de discrimination (D)** évalue le pouvoir prédictif d'une régression logistique

- Mesure à quel point la régression logistique est capable de bien classifier un résultat en succès ou échec

Pour évaluer l'ajustement du modèle, les graphiques de diagnostique ne sont pas utiles, il vaut mieux utiliser le **test de Hosmer-Lemeshow**:

- Compare le nombre de résultats obtenus et attendus
- Similaire à un test de *Chi*²

Exercice 2

La fonction R `PseudoR2` dans le package `DescTools` permet de calculer plusieurs Pseudo R². En spécifiant `which = all`, calculez toutes les statistiques en même temps.

```
library(DescTools)
fit ← PseudoR2(logit.reg, which = "all")
fit
```

	McFadden	McFaddenAdj	CoxSnell	Nagelkerke	Aldrich and Nelson
#	0.4655937	0.3998373	0.4549662	0.6245898	0.6245898
# VeallZimmermann		Efron McKelveyZavoina		Tjur	Tjur
#	0.6674318	0.5024101	0.7064093	0.5114661	0.5114661
#	BIC	logLik	logLik0	G2	G2
#	61.5078819	-24.3811981	-45.6229593	42.4835224	42.4835224

Exercice 2: Faites le test de Hosmer-Lemeshow

```
library(vcdExtra)
HLtest(logit.reg)
# Hosmer and Lemeshow Goodness-of-Fit Test
#
# Call:
# glm(formula = mites$pa ~ mites$WatrCont + mites$Topo, family = binomial)
# ChiSquare df   P_value
# 3.421693  8 0.9051814
```

Une valeur non significative indique un ajustement adéquat!

Défi 2



1. En utilisant le modèle créé avec le jeu de données 'bacteria', évaluez le pouvoir prédictif et l'ajustement de ce modèle.
2. Comment faire pour améliorer le pouvoir explicatif du modèle?



Solution

1 :

```
null.d ← model.bact2>null.deviance  
resid.d ← model.bact2$deviance  
bact.pseudoR2 ← (null.d - resid.d) / null.d  
HLtest(model.bact2)
```

2 : Ajouter des variables explicatives pertinentes pourrait certainement augmenter le pouvoir explicatif du modèle.

GLM et données de proportions

Parfois, les données de proportions sont plus similaires à un régression logistique que ce que vous pensez...

Si on mesure un nombre d'occurrences et qu'on connaît la taille d'échantillon, on obtient des données de proportions!

Supposons qu'on mesure la prévalence d'une maladie sur dix cerfs dans 10 populations différentes :

$$\frac{x \text{ infected deer}}{10 \text{ deer}}$$

→ toujours entre 0 et 1!

Exercice 3

Dans R, on doit spécifier le nombre de fois qu'un événement s'est produit et le nombre de fois qu'un événement ne s'est pas produit:

```
prop.reg ← glm(cbind(Galumna, totalabund - Galumna) ~ Topo + WatrCont, d  
summary(prop.reg)
```

Exercice 3

```
summary(prop.reg)
#
# Call:
# glm(formula = cbind(Galumna, totalabund - Galumna) ~ Topo + WatrCont,
#      family = binomial, data = mites)
#
# Deviance Residuals:
#       Min        1Q     Median        3Q       Max
# -1.4808  -0.9699  -0.6327  -0.1798   4.1688
#
# Coefficients:
#                 Estimate Std. Error z value Pr(>|z|)
# (Intercept) -3.288925  0.422109 -7.792 6.61e-15 ***
# TopoHummock  0.578332  0.274928  2.104  0.0354 *
# WatrCont    -0.005886  0.001086 -5.420 5.97e-08 ***
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# (Dispersion parameter for binomial family taken to be 1)
#
```

Exercice 3

On peut coder le modèle directement avec les proportions:

```
prop.reg2 ← glm(prop ~ Topo + WatrCont, data = mites,  
                  family = binomial, weights = totalabund)
```

GLM avec des données d'abondance

Modéliser des données d'abondance

Que sont des données d'abondance?

Importez le jeu de données `faramea.csv` dans R

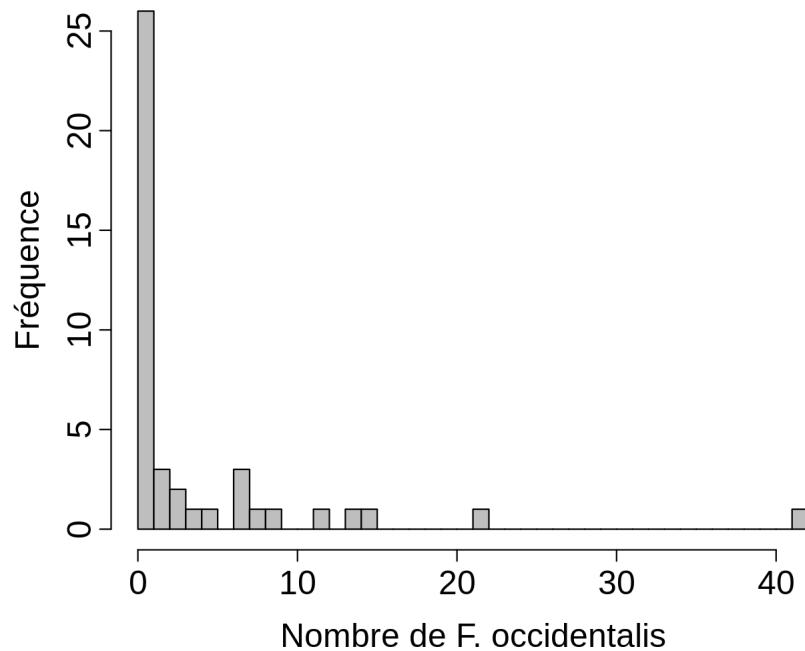
```
faramea ← read.csv('faramea.csv', header = TRUE)
```

Le nombre d'arbres de l'espèce *Faramea occidentalis* a été compté dans 43 quadrats sur l'île de Barro Colorado (Panama). Des données environnementales, comme l'élévation et la précipitation ont aussi été mesurées.

Examinons maintenant à quoi ressemble la distribution du nombre d'arbres par transect.

Modéliser des données d'abondance

Que sont des données d'abondance?



Modéliser des données d'abondance

Que sont des données d'abondance?

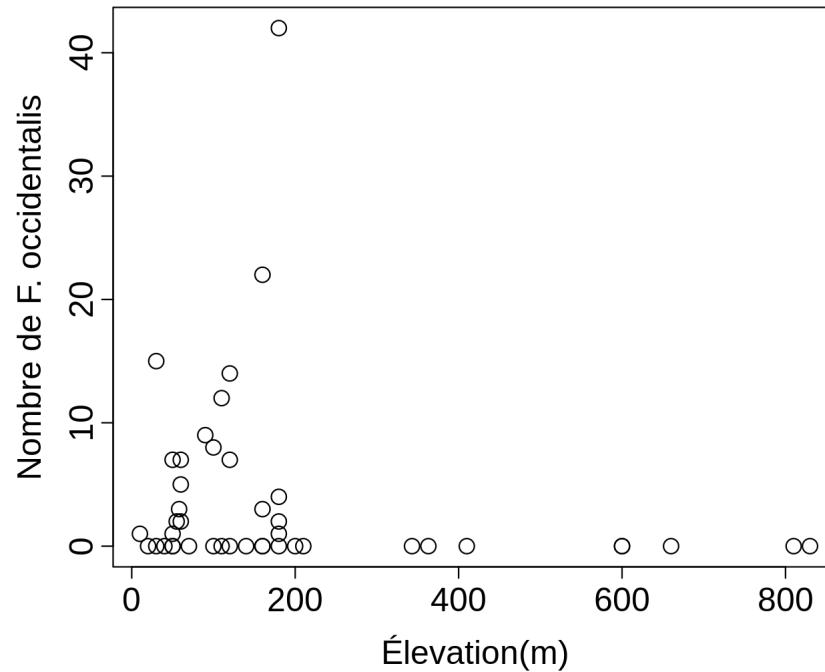
Les données d'abondance sont caractérisées par:

- des valeurs positives : on ne peut pas compter -7 individus
- des valeurs entières : on ne peut pas compter 7.56 individus
- une plus grande variance pour les fortes valeurs

Modéliser des données d'abondance

Comment modéliser des données d'abondance?

L'élévation influence-t-elle l'abondance de *F. occidentalis*?



Modéliser des données d'abondance

Comment modéliser des données d'abondance?

L'élévation influence-t-elle l'abondance de *F. occidentalis*?

La **distribution de Poisson** semble être le choix parfait pour modéliser ces données, ainsi des **GLMs Poisson** sont généralement une bonne façon pour commencer à modéliser des données d'abondance.

La distribution de Poisson

La distribution de poisson, qui spécifie la probabilité d'une variable aléatoire discrète Y , est données par :

$$f(y, \mu) = Pr(Y = y) = \frac{\mu^y \times e^{-\mu}}{y!}$$

$$E(Y) = Var(Y) = \mu$$

Propriétés :

- μ est le paramètre de la distribution de Poisson
- spécifie la probabilité pour des valeurs entières uniquement
- la probabilité pour des valeurs négatives est nulle ($P(Y < 0) = 0$)
- moyenne = variance (permet l'hétérogénéité)

Que se cache-t-il derrière un GLM Poisson?

Un GLM Poisson va modéliser la valeur de μ en fonction de différentes variables explicatives.

Trois étapes

Étape 1. On suppose que Y_i suit une distribution de Poisson de moyenne et variance μ_i

$$Y_i = \text{Poisson}(\mu_i)$$

$$E(Y_i) = \text{Var}(Y_i) = \mu_i$$

$$f(y_i, \mu_i) = \frac{\mu_i^{y_i} \times e^{-\mu_i}}{y!}$$

μ_i correspond au nombre attendu d'individus

Que se cache-t-il derrière un GLM Poisson?

Étape 2. On spécifie le prédicteur linéaire comme dans un modèle linéaire

$$\underbrace{\alpha}_{\text{One intercept}} + \underbrace{\beta}_{\text{slope of 'Elevation'}} \times \text{Elevation}_i$$

Étape 3. La fonction de lien entre la moyenne de Y_i et la partie systématique est un log

$$\log(\mu_i) = \alpha + \beta \times \text{Elevation}_i$$

OU

$$\mu_i = e^{\alpha + \beta \times \text{Elevation}_i}$$

Ajuster un GLM Poisson sous R

La fonction `glm()` permet de spécifier un GLM Poisson

```
glm.poisson = glm(Faramea.occidentalis~Elevation, data=faramea, family=pois
```

L'argument `family` permet de spécifier le type de distribution et la fonction de lien (log)

Tout comme avec `lm()`, vous pouvez accéder au résumé du modèle à l'aide de la fonction `summary()`

```
summary(glm.poisson)
```

Résumé du modèle

```
summary(glm.poisson)
#
# Call:
# glm(formula = Faramea.occidentalis ~ Elevation
#      data = faramea)
#
# Deviance Residuals:
#       Min        1Q    Median        3Q        Max
# -3.3319 -2.7509 -1.5451  0.1139 11.3995
#
# Coefficients:
#                      Estimate Std. Error z value Pr(>
# (Intercept) 1.7687001  0.1099136 16.092 < 2
# Elevation   -0.0027375  0.0006436 -4.253 2.11
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.
#
# (Dispersion parameter for poisson family taken
#
# Null deviance: 414.81 on 42 degrees of f
```

Estimés :

Intercept = α

Élevation = β

Qu'en est-il de **Null deviance** et **Residual deviance**?!

Estimation des paramètres

Dans notre modèle, les paramètres à estimer sont l'ordonnée à l'origine (α) et le coefficient de régression de l'élevation (β)

$$\log(\mu_i) = 1.769 - 0.0027 \times \text{Élevation}_i$$

ou

$$\mu_i = e^{1.769 - 0.0027 \times \text{Élevation}_i}$$

La déviance

Rappelez vous que pour estimer les paramètres inconnus, l'estimation par maximum de vraisemblance est utilisée

La déviance résiduelle est approximativement la différence entre la vraisemblance d'un modèle saturé (n paramètres pour chaque observation) et le modèle complet (p paramètres):

$$\text{Res dev} = 2 \log(L(y; y)) - 2 \log(L(y; \mu))$$

Dans un GLM Poisson, la déviance résiduelle doit être égale au nombre de degrés de liberté résiduels

388.12 >> 41

La surdispersion

Quand la déviance résiduelle est supérieure au nombre de degrés de liberté résiduels, le modèle est **surdispersé**

$$\phi = \frac{\text{Déviance résiduelle}}{\text{Degrés de liberté résiduels}}$$

Se produit lorsque la variance dans les données est plus grande que la moyenne. Dans ce cas la distribution de Poisson n'est plus appropriée (beaucoup de zéros, covariables manquantes, etc.)

Solutions

1: Corriger la surdispersion en utilisant en **GLM quasi-Poisson**

2: Choisir une autre distribution : **la negative binomial**

GLM Quasi-Poisson

La variance du modèle tient compte de la **surdispersion** en ajoutant le paramètre de surdispersion:

$$E(Y_i) = \mu_i$$

$$Var(Y_I) = \phi \times \mu_i$$

le **prédicteur linéaire** et la **fonction de lien** restent les mêmes
 ϕ est le paramètre de dispersion. Il sera estimé avant les paramètres. Corriger pour la surdispersion ne va pas affecter l'estimation des paramètres, mais leur **significativité**. En effet, les écarts-types des paramètres seront multipliés par $\sqrt{\phi}$.

Certaines p-values marginalement significatives peuvent devenir non significatives!

Ajuster un GLM quasi-Poisson sous R

Créez un nouveau GLM à l'aide de la famille 'quasipoisson' ou actualisez le modèle précédent:

```
glm.quasipoisson = glm(Faramea.occidentalis ~ Elevation, data = faramea,  
                        family=quasipoisson)  
glm.quasipoisson = update(glm.poisson, family = quasipoisson)
```

Ajuster un GLM quasi-Poisson sous R

```
summary(glm.quasipoisson)
#
# Call:
# glm(formula = Faramea.occidentalis ~ Elevation
#      data = faramea)
#
# Deviance Residuals:
#       Min        1Q    Median        3Q       Max
# -3.3319  -2.7509  -1.5451   0.1139  11.3995
#
# Coefficients:
#                   Estimate Std. Error t value Pr(>|t|)
# (Intercept)  1.768700  0.439233  4.027 0.000
# Elevation   -0.002738  0.002572 -1.064 0.293
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.
#
# (Dispersion parameter for quasipoisson family
#
# Null deviance: 414.81 on 42 degrees of f
```

Mêmes estimés mais

Les écarts-types des paramètres sont multipliés par

$$\sqrt{\phi} = 4$$

0.0006436 * 4 =
0.00257

$\leftarrow \phi$

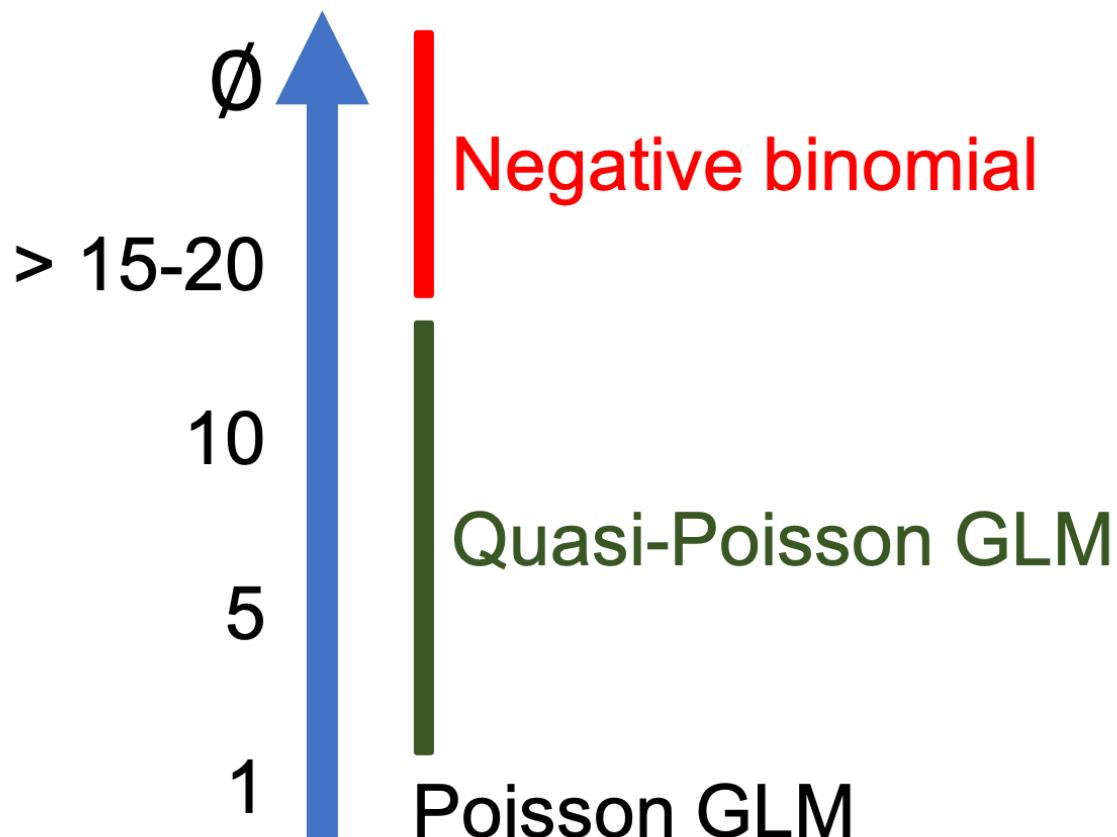
\leftarrow Pas d'AIC!

Ajuster un GLM quasi-Poisson sous R

Testons l'effet de l'élévation par une analyse de déviance :

```
null.model <- glm(Faramea.occidentalis ~ 1, data = faramea,
                     family = quasipoisson)
anova(null.model, glm.quasipoisson, test = "Chisq")
# Analysis of Deviance Table
#
# Model 1: Faramea.occidentalis ~ 1
# Model 2: Faramea.occidentalis ~ Elevation
#   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
# 1       42     414.81
# 2       41     388.12  1    26.686   0.1961
```

Paramètre de dispersion



GLM binomiale négative

Une distribution binomiale négative est favorisée quand la surdispersion est forte

- La distribution a **deux paramètres** μ and k . k contrôle pour la dispersion (plus la dispersion est forte, plus k est petit)
- C'est une combinaison de deux distributions (**Poisson** et **gamma**)
- Les Y_i suivent une distribution de Poisson dont la moyenne μ suit une distribution Gamma!

$$E(Y_i) = \mu_i$$

$$Var(Y_i) = \mu_i + \frac{\mu_i^2}{k}$$

Ajuster une binomiale négative sous R

NB La distribution binomiale n'est pas dans la fonction `glm()` donc il faut installer et charger la paquet `MASS`

```
install.packages('MASS')
```

```
glm.negbin = glm.nb(Faramea.occidentalis ~ Elevation, data = faramea)
```

```
summary(glm.negbin)
```

Ajuster une binomiale négative sous R

```
#  
# Call:  
# glm.nb(formula = Faramea.occidentalis ~ Elevation, data = faramea,  
#         init.theta = 0.2593107955, link = log)  
  
#  
# Deviance Residuals:  
#       Min        1Q    Median        3Q       Max  
# -1.36748  -1.17564  -0.51338  -0.05226   2.25716  
  
#  
# Coefficients:  
#                 Estimate Std. Error z value Pr(>|z|)  
# (Intercept)  2.369226  0.473841   5.00 5.73e-07 ***  
# Elevation   -0.007038  0.002496  -2.82  0.00481 **  
# ---  
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1  
  
#  
# (Dispersion parameter for Negative Binomial(0.2593) family taken to be 1)  
  
#  
#     Null deviance: 41.974 on 42 degrees of freedom  
# Residual deviance: 36.343 on 41 degrees of freedom
```

Représenter le modèle final

Étape 1 Représenter les données et utiliser les estimations des paramètres pour représenter le modèle

$$\mu_i = e^{2.369 - 0.007 \times Elevation_i}$$

Utilisez `summary()` pour obtenir les paramètres

```
summary(glm.negbin)$coefficients[1, 1]
summary(glm.negbin)$coefficients[2, 1]
```

Représenter le modèle final

Étape 2 Utilisez les écarts-types pour construire l'intervalle de confiance

```
summary(glm.negbin)$coefficients[1, 2]  
summary(glm.negbin)$coefficients[2, 2]
```

$$\text{Limite sup} = e^{[\alpha - 1.96 \times SE_{\alpha}] + [\beta - 1.96 \times SE_{\beta}] \times \text{Elevation}_i}$$

$$\text{Limit inf} = e^{[\alpha + 1.96 \times SE_{\alpha}] + [\beta + 1.96 \times SE_{\beta}] \times \text{Elevation}_i}$$

Représenter le modèle final

```
pp <- predict(glm.negbin, newdata = data.frame(Elevation = 1:800), se.fit = TRUE)
linkinv <- family(glm.negbin)$linkinv ## inverse-link function
pframe$pred0 <- pp$fit
pframe$pred <- linkinv(pp$fit)
sc <- abs(qnorm((1-0.95)/2)) ## Normal approx. to likelihood
pframe <- transform(pframe, lwr = linkinv(pred0-sc*pp$se.fit), upr = linkinv(pred0+sc*pp$se.fit))
# sinon, utiliser predict() avec type="response"

plot(faramea$Elevation, faramea$Faramea.occidentalis, ylab = 'Number of Faramea occidentalis')
lines(pframe$pred, lwd = 2)
lines(pframe$upr, col = 2, lty = 3, lwd = 2)
lines(pframe$lwr, col = 2, lty = 3, lwd = 2)
```



Défi 3

Utilisez le jeu de données `mites`! Modélisez l'abondance de l'espèce *Galumna* en fonction des caractéristiques du substrat (son contenu en eau `WatrCont` et sa densité `SubsDens`)

- Faut-il contrôler pour la surdispersion?
- Quelles variables explicatives ont un effet significatif?
- Selectionnez le meilleur modèle!

```
mites ← read.csv("mites.csv", header = TRUE)
```



Défi 3 : conseils

Sélection pas à pas en retirant à chaque fois une variable et en comparant le modèle emboité au modèle complet :

```
drop1(MyGLM, test = "Chi")
```

Spécifiez un modèle emboîté manuellement, appelez le `MyGLM2`, et utilisez la fonction `anova()`:

```
anova(MyGLM, MyGLM2, test = "Chi")
```

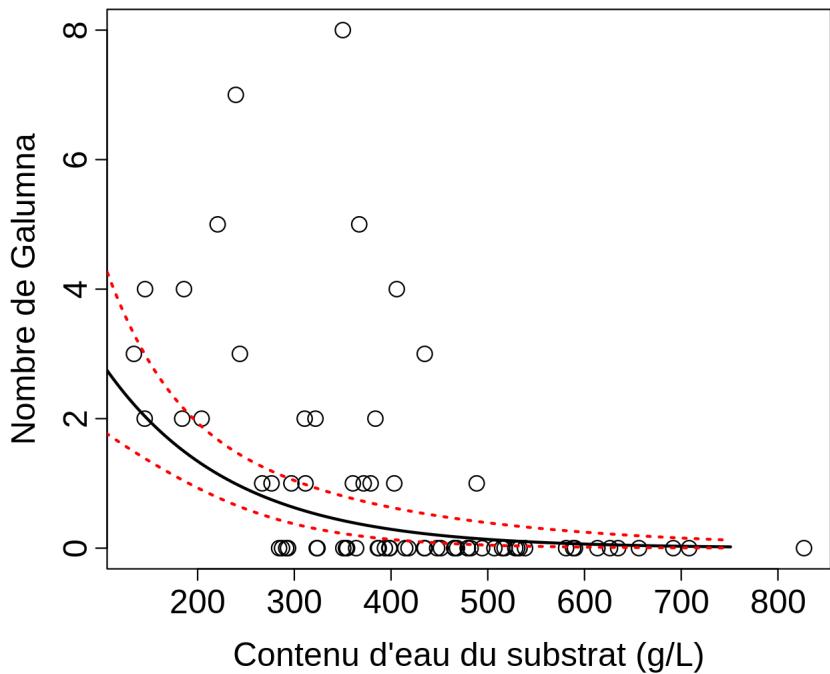
Défi 3 : solution



```
# GLM Poisson
glm.p = glm(Galumna~WatrCont+SubsDens, data=mites, family=poisson)
# GLM quasi-Poisson
glm.qp = update(glm.p,family=quasipoisson)
# sélection du modèle
drop1(glm.qp, test = "Chi")
# Single term deletions
#
# Model:
# Galumna ~ WatrCont + SubsDens
#          Df Deviance scaled dev. Pr(>Chi)
# <none>      101.49
# WatrCont   1    168.10      31.711 1.789e-08 ***
# SubsDens   1    108.05      3.125  0.07708 .
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

# ou
glm.qp2 = glm(Galumna~WatrCont, data=mites, family=quasipoisson)
anova(glm.qp2, glm.qp, test="Chisq")
```

Défi 3 : solution



Autres distributions

- **Transformation logit des données** souvent utilisée avec `lm()` pour les pourcentages et les proportions quand la distribution binomiale n'est pas appropriée. Quand non sélectionnée à partir de quantités fixées (e.g. pourcentage de couverture, grades scolaires, etc).
- **Distribution log-normal dans un `glm`**, évite d'avoir à log-transformer les données.
- **Distribution Gamma**. Similaire à une log-normal, plus flexible.
- **Distribution tweedie**. Famille de distributions flexible. Utile pour des données avec un mélange de 0 et de valeurs positives (pas forcément des comptes).
- **Poisson ou negative binomiale à inflation de zéro**. Quand les données comprennent un nombre excessif de zéros, venant d'un processus différent de celui qui génère les comptes.

GLMMs

Révision: Modèles Linéaires Mixtes

Rappel de l'atelier LMM:

- Structure dans le jeu de données ou corrélation entre les observations peut entraîner une **dépendance entre les observations** échantillonnés à partir des même sites ou points dans le temps
- On en tient compte en incluant des **termes d'effet aléatoire**

Effets aléatoires:

- Un échantillon de la population, i.e. les sujets que vous avez échantillonnés par hasard
- Explique la variation de la variable réponse

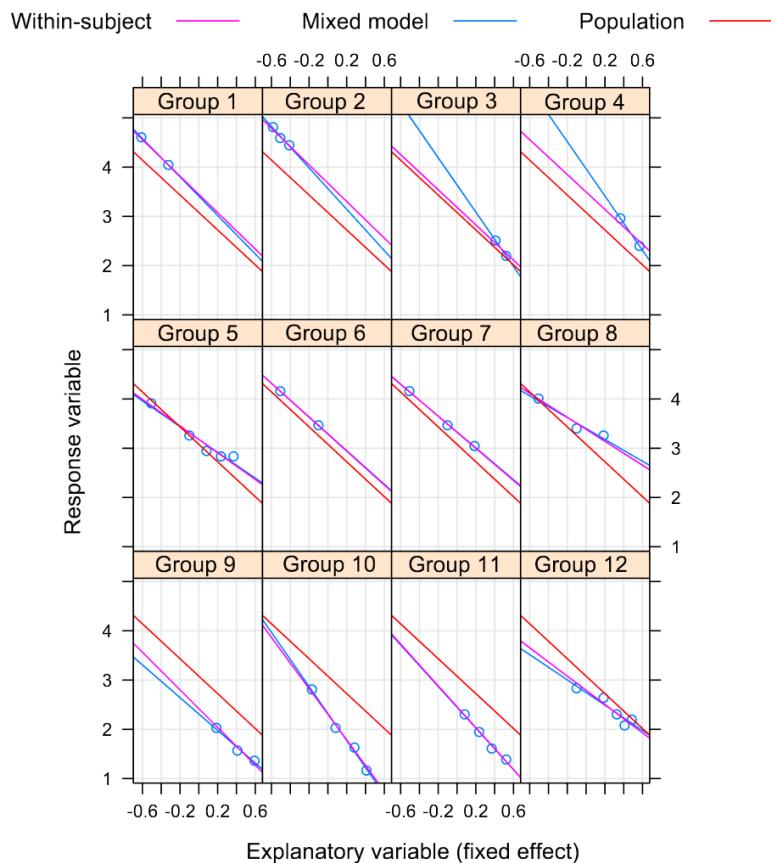
Effets fixes:

- Reproductible, i.e serait le même dans toutes les études
- Explique la moyenne de la variable réponse

Révision: Modèles Linéaires Mixtes

"Shrinkage estimates"

- Les effets aléatoires sont souvent appelés des **estimations de rétrécissement** parce qu'ils représentent une moyenne pondérée des données et de l'ajustement global (effet fixe)
- La baisse des coeff. vers l'ajustement global est plus sévère si la variabilité intra-groupe est grande par rapport à la variabilité inter-groupe



Modèles Linéaires Généralisés Mixtes (GLMMs)

Extension des GLMs tenant compte de structures supplémentaires dans les données

Suivre les étapes similaires à celles introduites lors de l'atelier sur les LMMs:

1. LMMs incorporent les effets aléatoires
2. GLMs peuvent gérer des données non-normales (en laissant les erreurs prendre différentes familles de distribution - e.g Poisson ou binomial négatif)

Comment modéliser un GLM sous R

Chargez les données `Arabidopsis banta_totalfruits.csv` dans R.

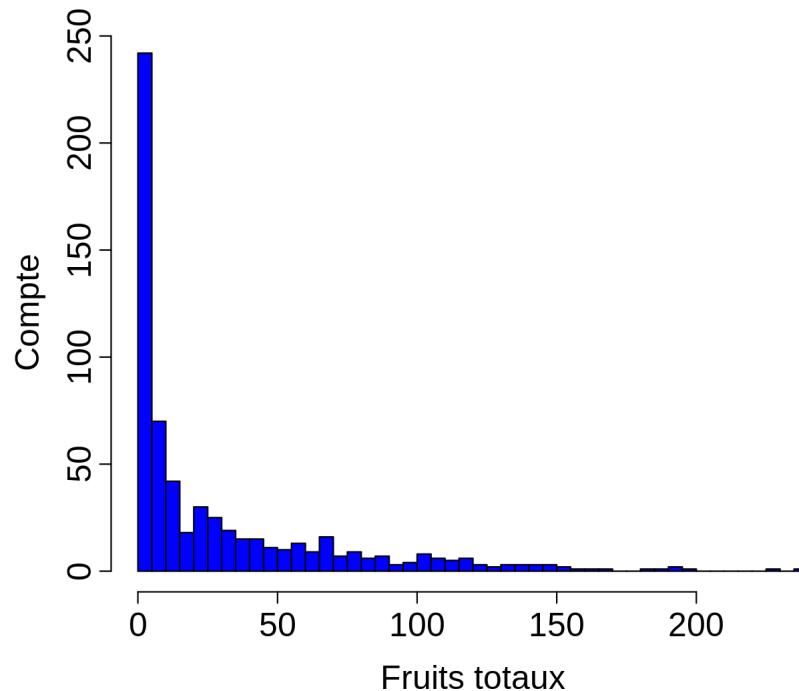
```
dat.tf ← read.csv("banta_totalfruits.csv")
```

```
# popu facteur avec un niveau pour chaque population  
# gen facteur avec un niveau pour chaque génotype  
# nutrient facteur avec niveau bas (valeur = 1) ou haut (valeur = 8)  
# amd facteur précisant l'absence ou la présence d'herbivorie  
# total.fruits nombre entier indiquant le nombre de fruits par plante
```

L'effet de la disponibilité de nutriments et d'herbivorie (**effets fixes**) sur la production de fruits d'*Arabidopsis thaliana* (arabette des dames) a été évalué en mesurant 625 plantes à travers neuf populations différentes, constituées chacune de 2 à 3 génotypes (**effets aléatoires**)

Choisir la distribution des erreurs

La variable réponse constitue des données d'abondance, donc nous devons choisir une **distribution de Poisson** (i.e variance égale à la moyenne)



Cependant, comme nous le verrons, la variance de chaque groupe augmente beaucoup plus rapidement que prévu...

Exploration de la variance

Pour illustrer l'hétérogénéité de la variance, nous allons d'abord créer des boîtes à moustaches (boxplots) de la variable réponse par rapport aux différents facteurs environnementaux

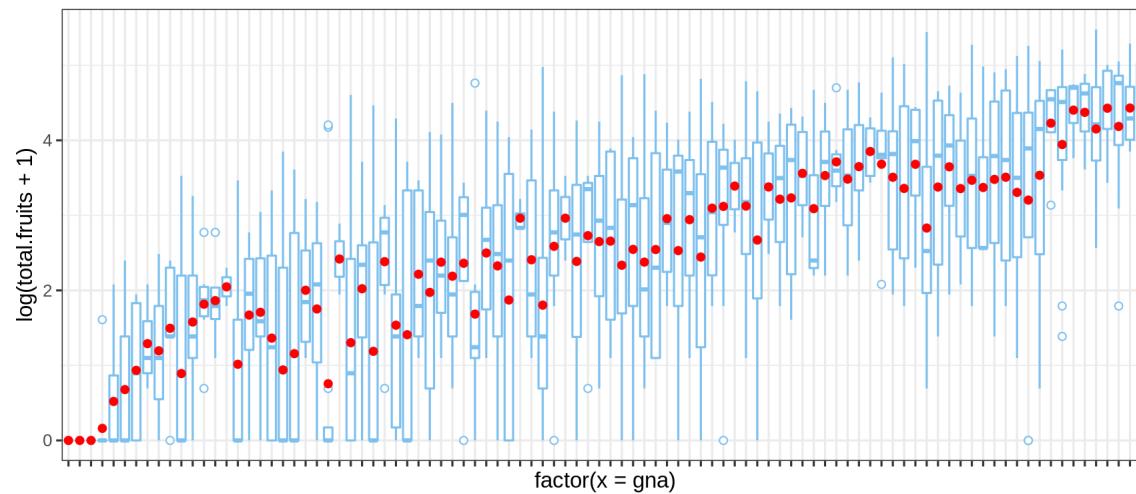
Créons de nouvelles variables qui représentent toutes les combinaisons de **nutriments x herbivorie x facteur aléatoire**

```
dat.tf <- within(dat.tf,
{
  # génotype x nutriment x herbivorie
  gna <- interaction(gen, nutrient, amd)
  gna <- reorder(gna, total.fruits, mean)
  # population x nutriment x herbivorie
  pna <- interaction(popu, nutrient, amd)
  pna <- reorder(pna, total.fruits, mean)
})
```

Exploration de la variance

```
# Boxplot du total des fruits vs interaction génotype x nutriment x herbier
```

```
library(ggplot2)
ggplot(data = dat.tf, aes(factor(x = gna),y = log(total.fruits + 1))) +
  geom_boxplot(colour = "skyblue2", outlier.shape = 21,
  outlier.colour = "skyblue2") +
  theme_bw() + theme(axis.text.x=element_blank()) +
  stat_summary(fun.y=mean, geom="point", colour = "red")
```

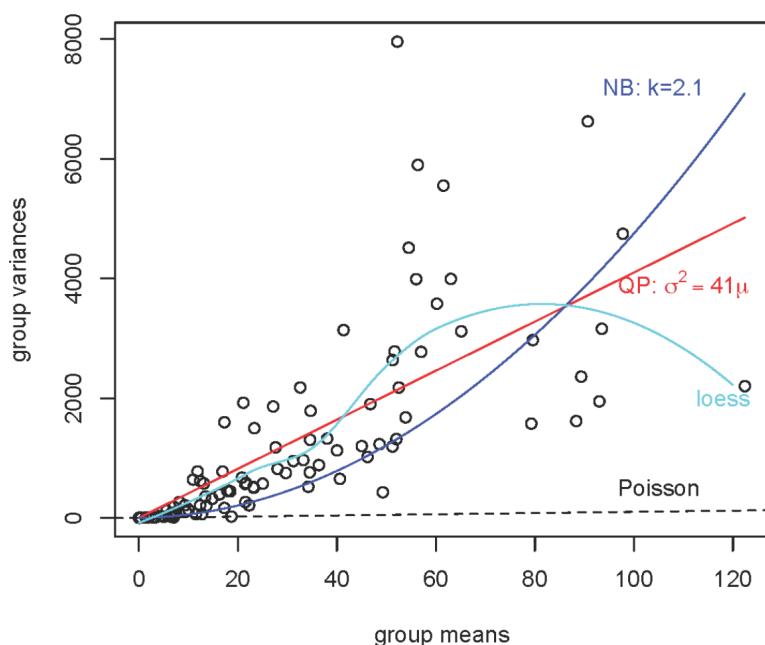


De même, la boîte à moustaches total des fruits vs population x nutriments x herbivorie montre une grande quantité d'hétérogénéité entre les populations.

Choisir la distribution des erreurs

Comme nous venons de le voir, il existe une importante hétérogénéité parmi la variance de chaque groupe, même lorsque la variable réponse est transformée

Si nous représentons graphiquement les **écart vs moyennes par groupes** (génotypes x nutriment x herbivorie), on voit que la distribution de Poisson est la moins appropriée (i.e. écart augmentent beaucoup plus vite que la moyenne)



NB = negative binomial

QP = quasi-Poisson

loess = Locally weighted regression smoothing

GLMM Poisson

Compte tenu de la relation moyenne-variance, nous avons besoin d'un modèle avec surdispersion.

- Mais commençons avec un modèle de Poisson :

Pour lancer un GLMM dans R, nous faisons appel à la fonction `glmer()`, du paquet lme4

```
library(lme4)
mp1 ← glmer(total.fruits ~ nutrient*amd + rack + status +
             (1|popu) +
             (1|gen),
             data = dat.tf, family = "poisson")
```

Effets aléatoires : `(1|popu)` contient un intercept aléatoire partagé par les mesures qui ont la même valeur pour `popu`

Vérification de la surdispersion

Nous pouvons vérifier la surdispersion en utilisant la fonction `overdisp_fun()` (Bolker *et al.* 2011) qui divise la déviance des résidus (résidus de Pearson) par les degrés de liberté des résidus et teste si le rapport est plus grand que 1

```
# Téléchargez le code glmm_funs.R de la page wiki et sourcez le pour exécuter
source(file="data/glmm_funs.R")
# Surdispersion?
overdisp_fun(mp1)
#      chisq      ratio          p      logp
# 15755.86833    25.57771  0.00000 -6578.47027
```

- Ratio est significativement $>> 1$
- Comme on s'y attendait, nous devons modéliser une distribution différente où la variance augmente plus rapidement que la moyenne

GLMM binomiale négative (Poisson-gamma)

La distribution binomiale négative satisfait la supposition que la **variance est proportionnelle au carré de la moyenne**

```
mnb1 ← glmer.nb(total.fruits ~ nutrient*amd + rack + status +  
                    (1|popu)+  
                    (1|gen),  
                    data=dat.tf, control=glmerControl(optimizer="bobyqa"))  
# Control spécifie la façon dont nous optimisons les valeurs des paramètres
```

```
# Surdispersion?  
overdisp_fun(mnb1)
```

←Le rapport est maintenant beaucoup plus près de 1 mais la valeur de p < 0.05

GLMM Poisson-lognormal

- Un autre option est la distribution **Poisson-lognormal**.
- Cela peut être réalisé simplement en plaçant un effet aléatoire de niveau d'observation dans la formule.

```
mpl1 ← glmer(total.fruits ~ nutrient*amd + rack + status +  
             (1|X) +  
             (1|popu)+  
             (1|gen),  
             data=dat.tf, family="poisson",  
             control = glmerControl(optimizer = "bobyqa"))
```

(1|X) traite de la surdispersion en ajoutant des **effets aléatoires au niveau de l'observation**

```
overdisp_fun(mpl1)  
#      chisq      ratio          p      logp  
# 1.775363e+02 2.886768e-01 1.000000e+00 -3.755952e-73
```

Rapport maintenant conforme avec notre critère

GLMM Poisson-lognormal

Représentation graphique des paramètres du modèle: Une représentation graphique des paramètres du modèle peut être obtenue en utilisant la fonction `coefplot2()` du paquet `coefplot2`:

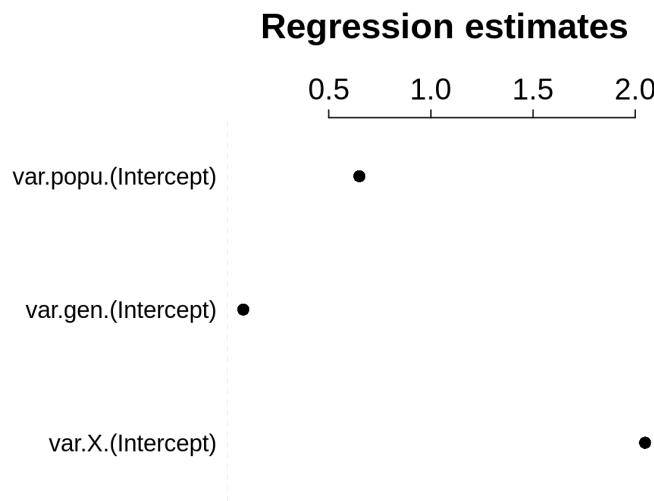
A Ce paquet n'est pas sur le CRAN! On utilise le package remotes pour l'installer depuis GitHub

```
if (!require("coefplot2"))
  remotes::install_github("palday/coefplot2", subdir = "pkg")
library(coefplot2)
```

GLMM Poisson-lognormal

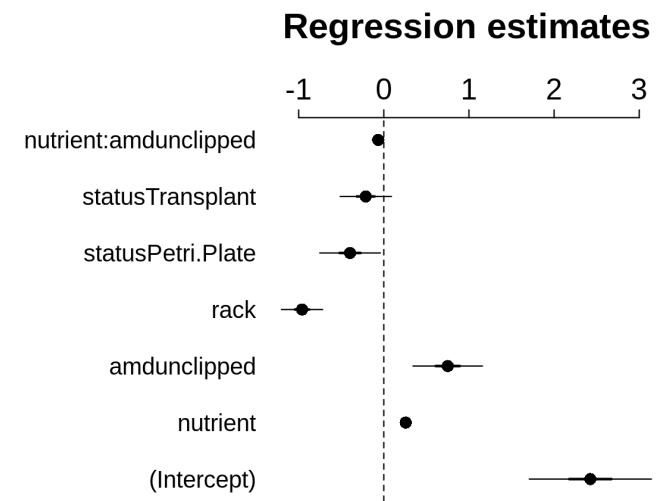
```
# Paramètres de la variance
```

```
coefplot2(mpl1, ptype = "vcov", i
```



```
# Effets fixes
```

```
coefplot2(mpl1, intercept = TRUE)
```



Note: barres d'erreur visibles seulement pour les effets fixes parce que glmer ne nous donne pas d'informations sur l'incertitude des effets aléatoires.

Visualisation des effets aléatoires

Vous pouvez aussi extraire les effets aléatoires en utilisant la fonction `ranef()` et les tracer en utilisant un `dotplot()` du paquet `lattice`

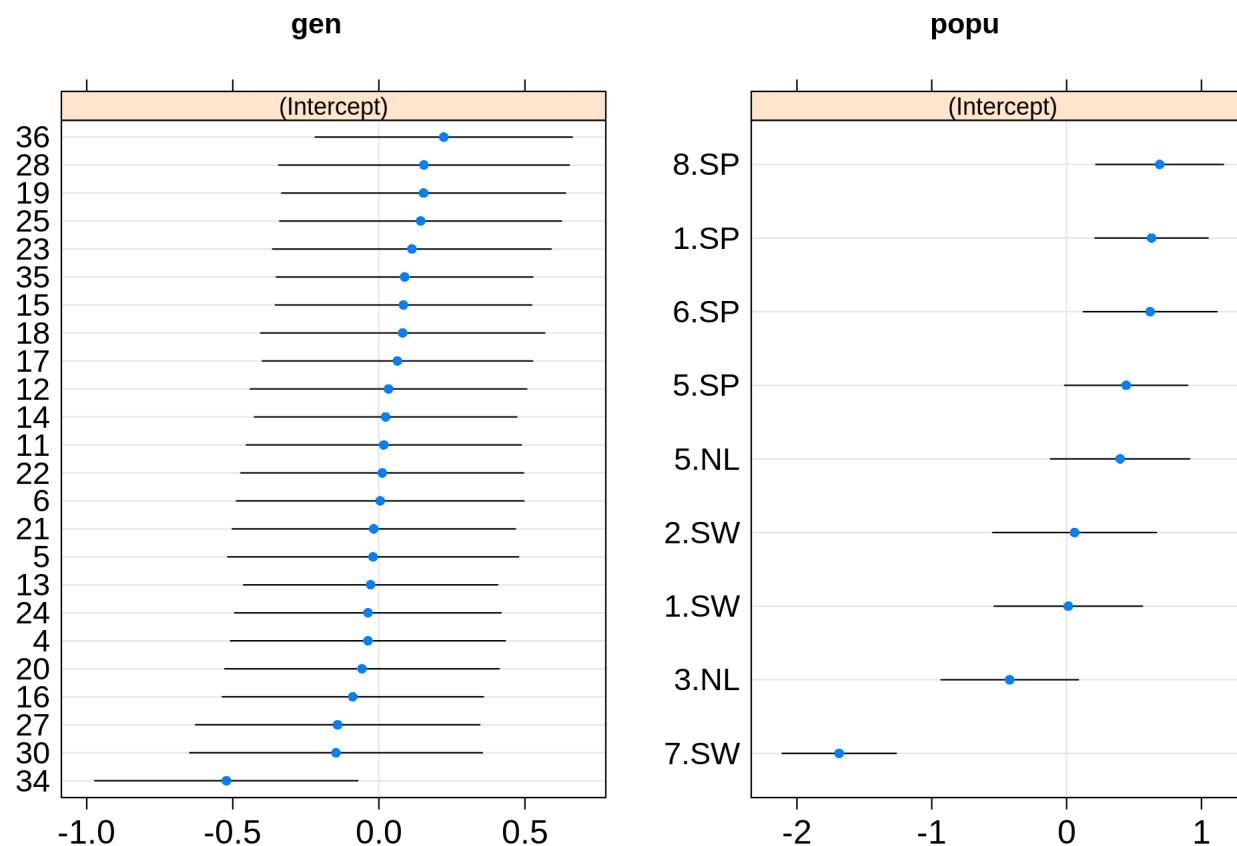
Il y a une variabilité régionale parmi les populations :

- Les populations espagnoles (SP) ont des valeurs plus élevées que les populations suédoises (SW) et néerlandaises (NL)

La différence entre les génotypes semble largement induite par génotype 34

```
library(gridExtra)
library(lattice)
# dotplot code
pp <- list(layout.widths=list(left.padding=0, right.padding=0),
           layout.heights=list(top.padding=0, bottom.padding=0))
r2 <- ranef(mpl1, condVar = TRUE)
d2 <- dotplot(r2, par.settings = pp)
grid.arrange(d2$gen, d2$popu, nrow = 1)
```

Visualisation des effets aléatoires



Sélection du modèle

Les mêmes méthodes peuvent être utilisées avec un glmm ou lmm pour choisir entre des modèles avec différents intercepts aléatoires et/ou des pentes aléatoires et pour choisir les effets fixes à conserver dans le modèle final.

- une **approche de la théorie de l'information** (e.g., AICc - Atelier 5)
- une **approche fréquentiste** (où l'importance de chaque terme est évaluée en utilisant `anova()` et le test de rapport de vraisemblance; LRT)

Sélection du modèle

Nous dérivons d'abord les modèles potentiels et les comparons en utilisant AICc*:

```
mpl2 ← update(mpl1, . ~ . - rack) # modèle sans rack
mpl3 ← update(mpl1, . ~ . - status) # modèle sans status
mpl4 ← update(mpl1, . ~ . - amd:nutrient) # modèle sans interaction amd:
bbmle::ICtab(mpl1, mpl2, mpl3, mpl4, type = c("AICc"))
#      dAICc df
# mpl1  0.0 10
# mpl4  1.4  9
# mpl3  1.5  8
# mpl2 55.0  9
```

*NB: Nous ne couvrons pas tous les modèles possibles ci-dessus, cependant, l'interaction `amd:nutrients` ne peut être évaluée que si `amd` et `nutrients` sont présents dans le modèle.

Sélection du modèle

Nous pouvons aussi utiliser les fonctions `drop1()` et `dfun()` pour évaluer nos effets fixes (`dfun()`) convertit les valeurs AIC retournées par `drop1()` en valeurs ΔAIC)

```
dd_LRT ← drop1(mpl1,test="Chisq")
(dd_AIC ← dfun(drop1(mpl1)))
# Single term deletions
#
# Model:
# total.fruits ~ nutrient * amd + rack + status + (1 / X) + (1 /
#     popu) + (1 / gen)
#           Df   dAIC
# <none>      0.000
# rack         1 55.083
# status       2  1.612
# nutrient:amd 1  1.444
```

Sélection du modèle

```
dd_LRT ← drop1(mpl1,test="Chisq")
(dd_AIC ← dfun(drop1(mpl1)))
# Single term deletions
#
# Model:
# total.fruits ~ nutrient * amd + rack + status + (1 / X) + (1 /
#     popu) + (1 / gen)
#           Df   dAIC
# <none>      0.000
# rack         1  55.083
# status        2   1.612
# nutrient:amd  1   1.444
```

- Fort effet de **rack** ($dAIC = 55.08$ si on enlève cette variable)
- Effets de **status** et de l'**interaction** sont faibles ($dAIC < 2$)
- Commençons par **enlever l'interaction non significative** afin de tester les effets principaux de nutriments et d'herbivorie

Sélection du modèle

```
mpl2 ← update(mpl1, . ~ . - and:nutrient)
# Utiliser AIC
mpl3 ← update(mpl2, . ~ . - rack) # pas de rack
mpl4 ← update(mpl2, . ~ . - status) # pas de st
mpl5 ← update(mpl2, . ~ . - nutrient) # pas de
mpl6 ← update(mpl2, . ~ . - amd) # pas d'herbiv
# bbmle::ICTab(mpl2, mpl3, mpl4, mpl5, mpl6,
#                 type = c("AICc"))

# Ou utiliser drop1
dd_LRT2 ← drop1(mpl2,test="Chisq")
dd_AIC2 ← dfun(drop1(mpl2))
```

```
library(bbmle)
ICTab(mpl2, mpl3 ,
      mpl5, mpl6,
      type = c("AI
#       dAICc df
# mpl2  0.0  10
# mpl5  0.0  10
# mpl4  1.5  8
# mpl6 10.6  9
# mpl3 55.0  9
```

Sélection du modèle

- Fort effets de **nutriments** et d'**herbivorie** (grand changement d'AIC de **135.6** (`mpl5`) et **10.2** (`mpl6`) si l'un ou l'autre sont supprimés, respectivement).
- Notre modèle final inclut l'effet fixe de nutriments, d'herbivorie, la variable nuisance de rack, l'effet aléatoire au niveau de l'observation `(1|x)` et la variation de fruits par populations et génotypes.

Prêt pour un défi?



En utilisant l'ensemble de données **inverts** (temps de développement larvaire (**PLD**) de 74 espèces d'invertébrés et vertébrés marins élevés à différentes températures et temps), répondez aux questions suivantes:

- Quel est l'effet du type d'alimentation et du climat (**effets fixes**) sur **PLD**?
- Est-ce que cette relation varie selon les taxons (**effets aléatoires**)?
- Quelle est la **meilleure famille de distributions** pour ces données?
- Finalement, une fois que vous avez déterminé la meilleure famille de distribution, re-évaluez vos effets fixes et aléatoires.

Solution

```
# inverts ← read.csv('data/inverts.csv', header = TRUE)
# head(inverts)
# table(inverts$temp, inverts$feeding.type)
#
# mod.glm ← glm(PLD ~ temp + feeding.type, family = poisson(), data = inverts)
# summary(mod.glm)
# drop1(mod.glm, test = "Chisq")
#
# boxplot(PLD ~ temp, data = inverts)
# boxplot(PLD ~ feeding.type , data = inverts)
#
# boxplot(predict(mod.glm, type = "response")~inverts$temp)
#
# plot()
#
# modglm ← glm(PLD ~ temp + feeding.type, family = poisson(), data = inverts)
#
# r2 ← ranef(modglm, condVar = TRUE)
# d2 ← dotplot(r2, par.settings = pp)
```

Ressources additionnelles sur les GLMMs

Livres :

- B. Bolker (2009) Ecological Models and Data in R. Princeton University Press.
- A. Zuur et al. (2009) Mixed Effects Models and Extensions in Ecology with R. Springer.

Articles :

- [Harrison et al. \(2018\), PeerJ, DOI 10.7717/peerj.4794](#)

Sites internet :

- GLMM for ecologists (<http://glmm.wikidot.com>) A great website on GLMM with a Q&A section!

Merci pour votre participation à cet atelier!

