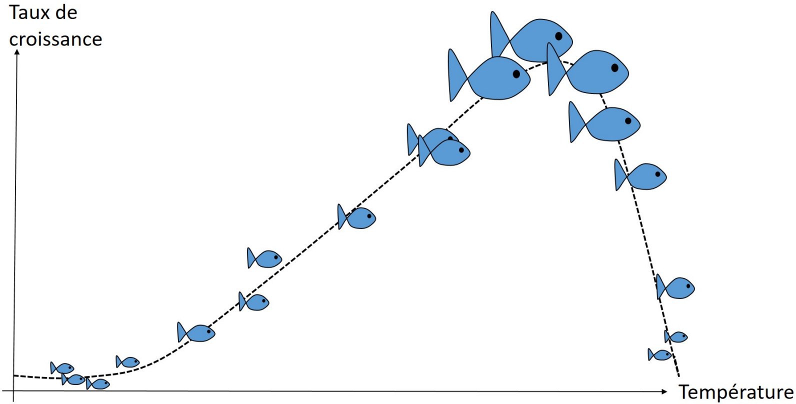
Introduction à la régression non linéaire avec la fonction nls() et le package nlstools

Marie Laure Delignette-Muller

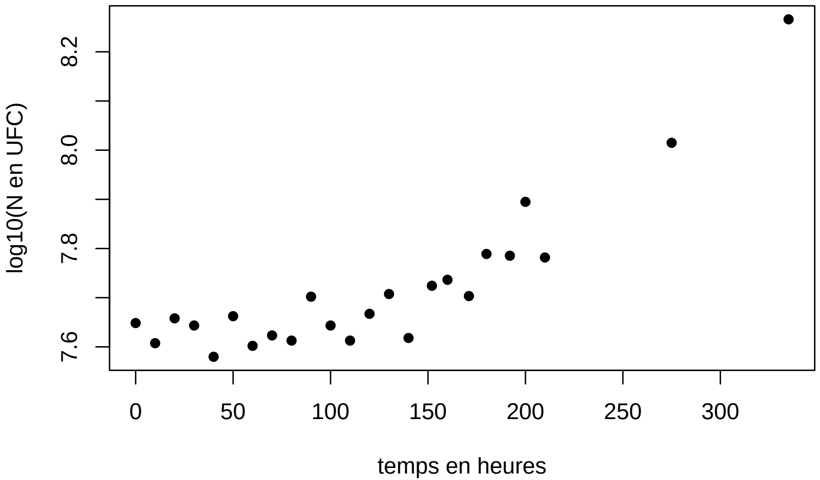
05 janvier, 2023

Illustration



Exemple de données de croissance bactérienne

On étudie la cinétique de croissance en bouillon de culture d'une souche de *Listeria monocytogenes* (latence et croissance exponentielle).



Exemple de modèle non linéaire utilisable sur ce type de données

Modèle de Baranyi (1994) à 3 paramètres défini par :

$\frac{dN}{Ndt} = \mu_{max} \times \frac{q(t)}{1+q(t)}$ avec $N(0) = N_0$,

$\frac{dq}{qdt} = \mu_{max}$ avec $q(0) = q_0 = \frac{1}{e^{\mu_{max} \times lag} - 1}$

de **solution analytique** :

$$y(t) = log_{10}(N(t)) = y_0 + \frac{\mu_{max}}{ln(10)} t + log_{10}(e^{-\mu_{max} \times t}(1 - e^{-\mu_{max} \times lag}) + e^{-\mu_{max} \times lag})$$

Le modèle non linéaire gaussien

Le modèle non linéaire gaussien

On suppose dans cette présentation que vous maîtrisez bien le cours sur le modèle linéaire.

Modèle non linéaire gaussien et moindres carrés

$Y_i = f(X_i, \theta) + \epsilon_i$

avec

$\epsilon_i \sim N(0, \sigma)$

Par rapport au modèle linéaire gaussien seule la partie déterministe est modifiée.

La maximisation de la vraisemblance et donc toujours équivalente à la minimisation de la Somme des Carrés des Ecarts (SCE)} :

$SCE = \sum_{i=1}^{n} e_i^2$ avec $e_i = Y_i - \hat{Y}_i$

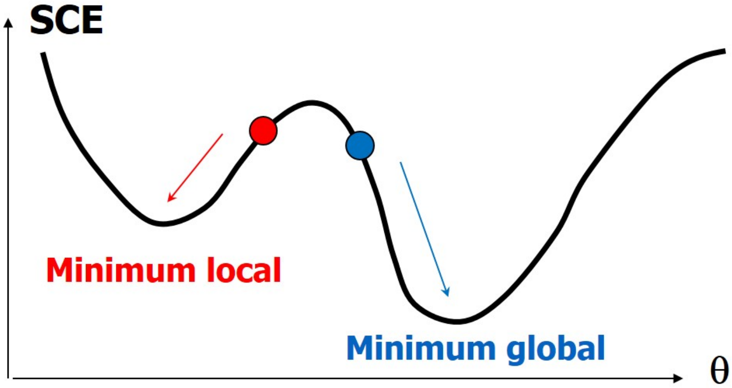
MAIS ce problème d'optimisation n'admet plus de solution analytique dans le cas général !

Minimisation de la SCE en régression non linéaire

Minimisation de la SCE en régression non linéaire

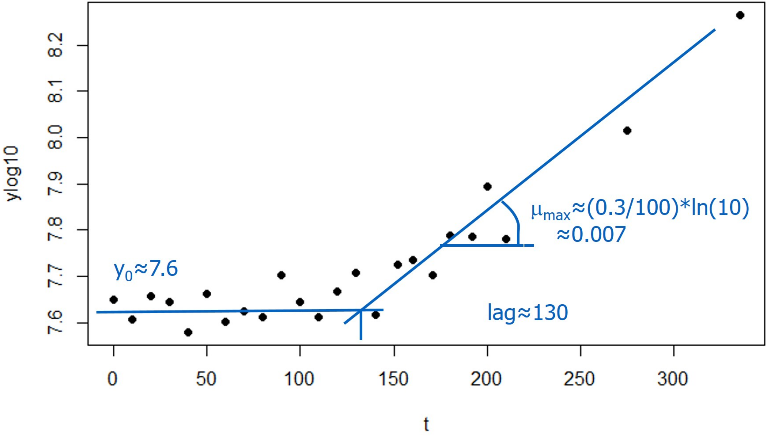
En pratique il falloir définir des **valeurs initiales proches de l'optimum pour les paramètres** à estimer puis utiliser un algorithme itératif pour atteindre le maximum de vraisemblance (i.e. minimum de la SCE).

Illustration pour un modèle à un seul paramètre



Estimation des valeurs initiales sur notre exemple

Tâche généralement pas trop difficile si les paramètres ont un sens biologique simple.



Procédure d'ajustement du modèle avec nls() et les fonctions du package nlstools

Ecriture du modèle sous forme de formule avec des noms de variables cohérentes avec leur codage dans le jeu de données.

Regardons le codage du jeu de données.

```
d <- read.table("DATA/lagCFU.txt", header = TRUE)
str(d)
```

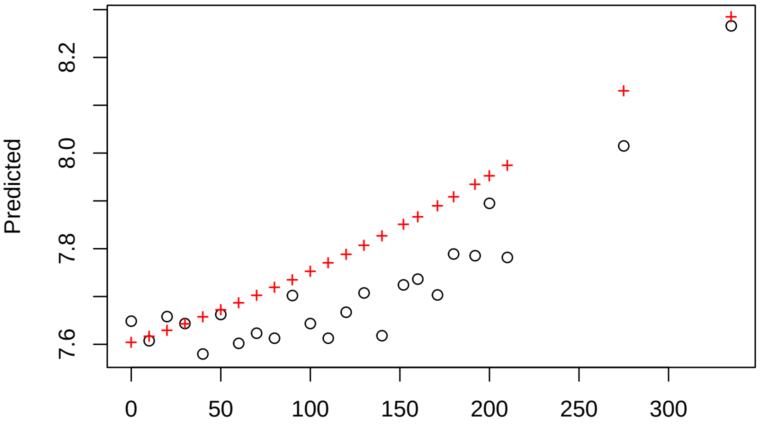
```
## 'data.frame':    24 obs. of  2 variables:
##  $ t     : int  0 10 20 30 40 50 60 70 80 90 ...
##  $ ylog10: num  7.65 7.61 7.66 7.64 7.58 ...
```

Codons le modèle.

```
baranyi = as.formula(ylog10 ~ y0 + mu * t/log(10) +
          log10(exp(- mu * t) * (1 - exp(- mu * lag)) +
          exp(- mu * lag)))
```

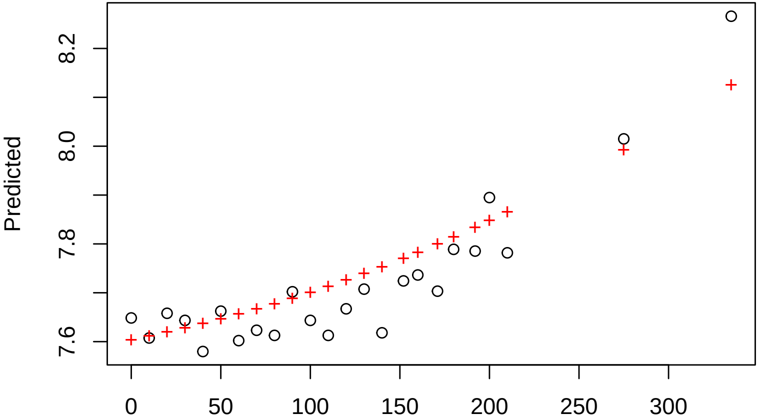
Visualisation du modèle pour les valeurs initiales

```
preview(baranyi, data = d,
        start = list(y0 = 7.6, mu = 0.007, lag = 130))
```



Ajustement manuel éventuel des valeurs initiales

```
preview(baranyi, data = d,
        start = list(y0 = 7.6, mu = 0.007, lag = 200))
```



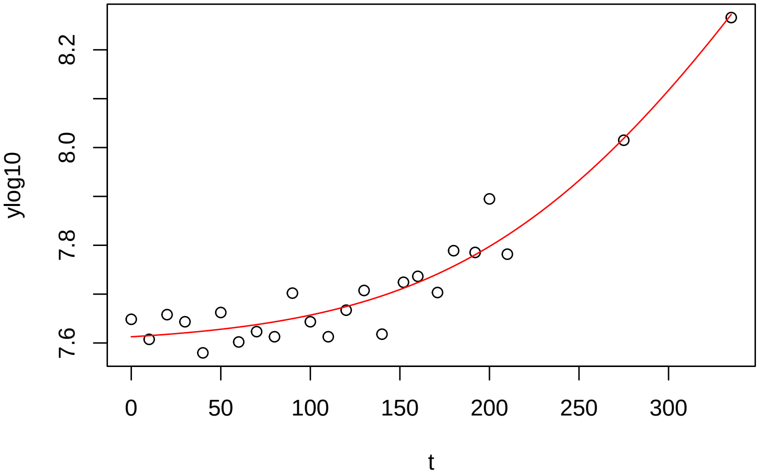
Ajustement avec nls() du modèle avec ces valeurs initiales

```
(m <- nls(formula = baranyi,
    start = list(y0 = 7.6, mu = 0.007, lag = 200),
    data = d))
```

```
## Nonlinear regression model
##   model: ylog10 ~ y0 + mu * t/log(10) + log10(exp(-mu *
##    data: d
##       y0       mu      lag
##   7.6127   0.0137 241.4031
##  residual sum-of-squares: 0.0349
##
## Number of iterations to convergence: 6
## Achieved convergence tolerance: 2.5e-06
```

Visualisation de l'ajustement

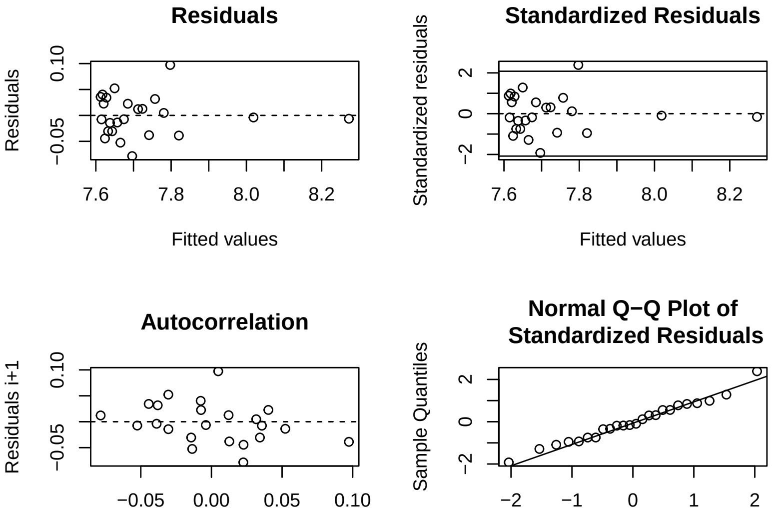
```
plotfit(m, smooth = TRUE)
```



Vérification des conditions d'utilisation du modèle

Graphe des résidus (comme en modèle linéaire !)

```
resi <- nlsResiduals(m)
plot(resi)
```



Linéarité au voisinage de l'estimation

Pour la suite il peut être utile de vérifier la linéarité du modèle au voisinage du minimum de la SCE, ce qui permettra de valider l'utilisation de certains outils associé modèle linéaire.

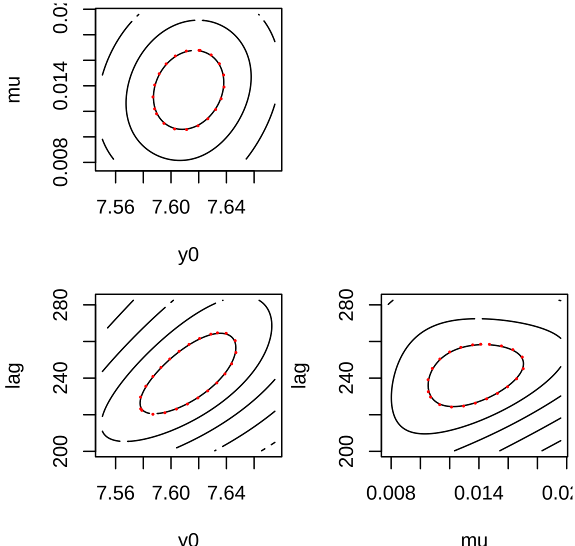
En effet certains résultats asymptotiques sur des statistiques permettant de faire des tests ou de calculer des intervalles de confiance à partir d'un modèle linéaire, ne seront applicables que si le modèle est proche d'un modèle linéaire au voisinage du min de la SCE.

On vérifie cette hypothèse par exemple en regardant les **contours de la SCE autour de son minimum, qui devraient être quasi elliptiques** si c'est le cas.

D'ailleurs plus ces contours sont tordus, plus on est en droit de se questionner sur l'estimation des paramètres (difficulté d'atteindre le minimum gobal et éventuelles corrélations fortes entre paramètres).

Contours de la SCE au voisinage du minimum elliptiques ?

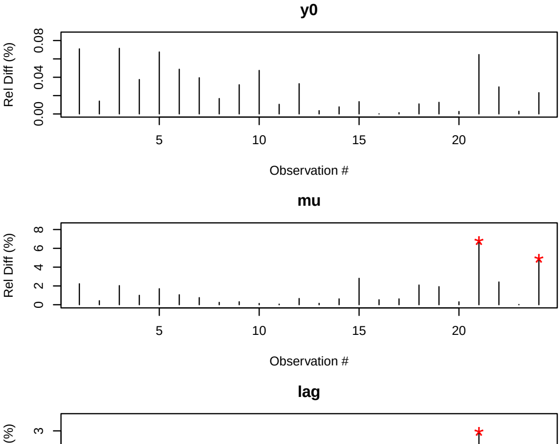
```r
contSCE <- nlsContourRSS(m)
plot(contSCE, col = FALSE, nlev = 6)
```



Détection des valeurs influentes par jacknife

Même principe qu'en modèle linéaire mais on regarde l'influence de chaque observation sur chaque paramètre.

```
jack <- nlsJack(m)
plot(jack)
```



Inférence statistique

Intervalles de confiance

Comme avec un modèle linéaire mais ne donne pas l'intervalle sur valeur prédite (plus compliqué avec un modèle non linéaire !)

Intervalles de confiance sur les paramètres

```
confint(m)
```

```
##          2.5%    97.5%
## y0     7.581   7.6422
## mu     0.011   0.0167
## lag 219.553 260.3216
```

Valeur prédite

```
predict(m, data.frame(t = 250))
```

```
## [1] 7.93
```

Résumé et statistiques d'ajustement

r^2 non donné car il n'a plus le sens de part de variation expliquée qu'il a avec le modèle linéaire !

```
summary(m)
```

```
##
## Formula: ylog10 ~ y0 + mu * t/log(10) + log10(exp(-mu * t) *
##     lag)) + exp(-mu * lag))
##
## Parameters:
##     Estimate Std. Error t value Pr(>|t|)
## y0  7.61e+00   1.50e-02  508.68  < 2e-16 ***
## mu  1.37e-02   1.42e-03    9.67  3.5e-09 ***
## lag 2.41e+02   9.88e+00   24.45  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '
##
## Residual standard error: 0.0408 on 21 degrees of freedom
##
## Number of iterations to convergence: 6
## Achieved convergence tolerance: 2.5e-06
```

Comparaison de modèles avec le test F des modèles emboîtés

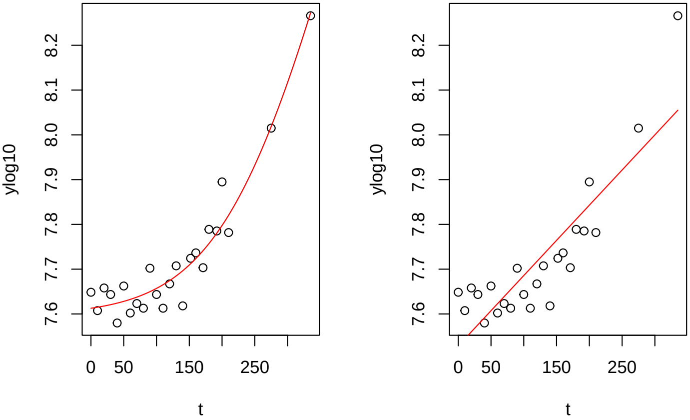
Utilisable si les modèles sont proches de la linéarité au voisinage du min de la SCE

```
# Modèle emboîté sans lag (lag = 0)
msimplifie <- nls(as.formula(ylog10 ~ y0 + mu * t/log(10)),
    start = list(y0 = 7.6, mu = 0.007), data = d)
anova(m, msimplifie)
```

```
## Analysis of Variance Table
##
## Model 1: ylog10 ~ y0 + mu * t/log(10) + log10(exp(-mu *
## Model 2: ylog10 ~ y0 + mu * t/log(10)
##   Res.Df Res.Sum Sq Df Sum Sq F value  Pr(>F)
## 1     21     0.0349
## 2     22     0.1402 -1 -0.105    63.4 8.9e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
```

Illustration des deux modèle comparés

```
par(mfrow = c(1,2))
plotfit(m, smooth = TRUE)
plotfit(msimplifie, smooth = TRUE)
```



Utilisation des critères d'information pour comparer des modèles possible aussi

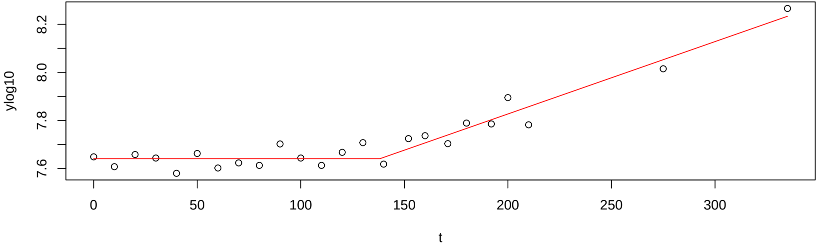
Comme pour des modèles linéaires !

```
AIC(m, msimplifie)
```

```
##            df   AIC
## m           4 -80.7
## msimplifie  3 -49.3
```

A vous de jouer !

En utilisant le même jeu de données, ajustez par régression non linéaire un **modèle linéaire par morceaux** (une phase sans croissance avec $y = y_0$ jusqu'à $t = lag$, suivie d'une phase linéaire de pente μ_{max}), puis comparez-le au modèle de Baranyi, notamment en utilisant l'AIC.



Ce modèle nécessite l'utilisation de la régression non linéaire car il est non linéaire par rapport au paramètre de rupture de pente *lag*, à partir du moment où celui-ci n'est pas supposé connu *a priori* mais estimé à partir des données.

Pour aller plus loin !

D'autres fonctions utilisables dans le package nlstool notamment pour calculer des **intervalles de confiance sur valeurs prédites** ou **sur n'importe quelle fonction des paramètres estimés par bootstrap** (fonction nlsBoot()).

Article décrivant le package nlstools accessible :

Baty, F., Ritz, C., Charles, S., Brutsche, M., Flandrois, J. P., & Delignette-Muller, M. L. (2015). A toolbox for nonlinear regression in R: the package nlstools. Journal of Statistical Software, 66(5), 1-21. accessible sur le site de la revue :
https://www.jstatsoft.org/article/view/v066i05

Quelques éléments de correction de l'exercice

```
modeleaseuil = as.formula(ylog10 ~
          y0 + mu * (t - lag) /log(10) * (t > lag) )
mseuil <- nls(formula = modeleaseuil,
    start = list(y0 = 7.6, mu = 0.007, lag = 130), data = d)
coef(mseuil)
```

```
##       y0       mu      lag
## 7.64e+00 6.94e-03 1.38e+02
```

```
AIC(m, mseuil)
```

```
##        df   AIC
## m       4 -80.7
## mseuil  4 -79.8
```

```
plotfit(mseuil, smooth = TRUE)
```