Reanalyse des données Challenger

Sébastien Guyader

November 12, 2018

Informations sur l'environnement de calcul

```
# Données sur le système
Sys.info()
```

```
##                                     sysname
##                                     "Linux"
##                                     release
##                           "4.18.17-1-MANJARO"
##                                     version
## "#1 SMP PREEMPT Sun Nov 4 18:19:14 UTC 2018"
##                                    nodename
##                                  "sg-lenovo"
##                                     machine
##                                    "x86_64"
##                                       login
##                                   "sguyader"
##                                        user
##                                   "sguyader"
##                                effective_user
##                                   "sguyader"
```

```
# Données sur R et les librairies chargées
sessionInfo()
```

```
## R version 3.5.1 (2018-07-02)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Manjaro Linux
##
## Matrix products: default
## BLAS: /usr/lib/libopenblasp-r0.3.3.so
## LAPACK: /usr/lib/liblapack.so.3.8.0
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=en_US.UTF-8        LC_COLLATE=en_US.UTF-8
##  [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] ggplot2_3.1.0
##
```

```
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.19     bindr_0.1.1      knitr_1.20       magrittr_1.5
##  [5] tidyselect_0.2.5 munsell_0.5.0    colorspace_1.3-2 R6_2.3.0
##  [9] rlang_0.3.0.1    stringr_1.3.1    plyr_1.8.4       dplyr_0.7.7
## [13] tools_3.5.1      grid_3.5.1       gtable_0.2.0     withr_2.1.2
## [17] htmltools_0.3.6  assertthat_0.2.0 yaml_2.2.0       lazyeval_0.2.1
## [21] rprojroot_1.3-2  digest_0.6.18    tibble_1.4.2     crayon_1.3.4
## [25] bindrcpp_0.2.2   purrr_0.2.5      glue_1.3.0       evaluate_0.12
## [29] rmarkdown_1.10   stringi_1.2.4    compiler_3.5.1   pillar_1.3.0
## [33] scales_1.0.0     backports_1.1.2  pkgconfig_2.0.2
```

Import des données au format CSV

Le fichier de données est téléchargé depuis cet endroit.

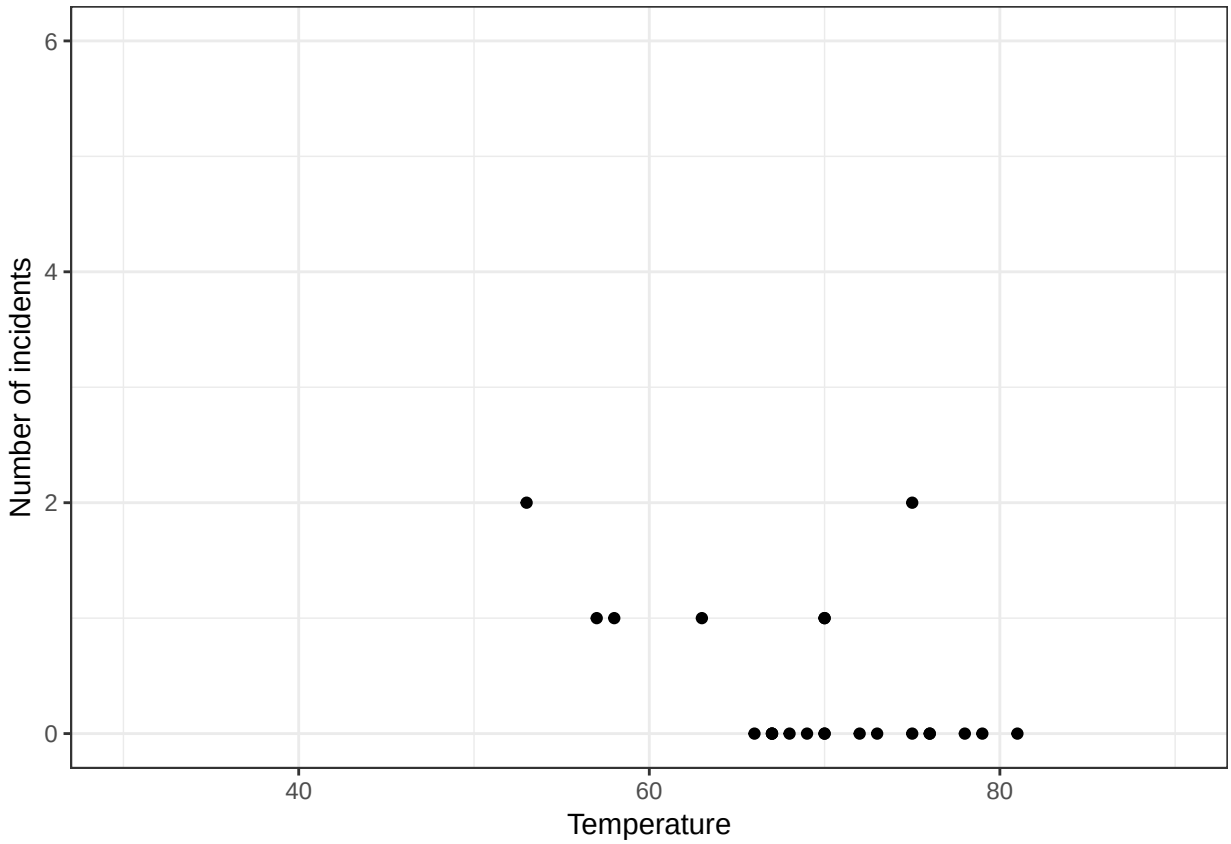
Commençons par charger ces données :

```
data <- read.csv("data.csv")
data
```

```
##         Date Count Temperature Pressure Malfunction
## 1    4/12/81     6          66       50           0
## 2   11/12/81     6          70       50           1
## 3    3/22/82     6          69       50           0
## 4   11/11/82     6          68       50           0
## 5    4/04/83     6          67       50           0
## 6    6/18/82     6          72       50           0
## 7    8/30/83     6          73      100           0
## 8   11/28/83     6          70      100           0
## 9    2/03/84     6          57      200           1
## 10   4/06/84     6          63      200           1
## 11   8/30/84     6          70      200           1
## 12  10/05/84     6          78      200           0
## 13  11/08/84     6          67      200           0
## 14   1/24/85     6          53      200           2
## 15   4/12/85     6          67      200           0
## 16   4/29/85     6          75      200           0
## 17   6/17/85     6          70      200           0
## 18 7/2903/85     6          81      200           0
## 19   8/27/85     6          76      200           0
## 20  10/03/85     6          79      200           0
## 21  10/30/85     6          75      200           2
## 22  11/26/85     6          76      200           0
## 23   1/12/86     6          58      200           1
```

Visualisation des données

```
ggplot(data=data, aes(x=Temperature, y=Malfunction)) +
  geom_point() +
  xlim(c(30,90)) + ylim(c(0,6)) +
  xlab("Temperature") + ylab("Number of incidents") +
  theme_bw()
```



Analyse par regression logistique (loi binomiale sur y=Malfunction/Count)

Comme l'influence de la pression est négligeable, analysons l'effet de la température seule :

```
logis_reg_binomial <- glm(data=data, Malfunction/Count ~ Temperature, weights=Count,
                  family=binomial(link="logit"))
summary(logis_reg_binomial)
```

```
##
## Call:
## glm(formula = Malfunction/Count ~ Temperature, family = binomial(link = "logit"),
##     data = data, weights = Count)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -0.95227  -0.78299  -0.54117  -0.04379   2.65152
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  5.08498    3.05247   1.666   0.0957 .
## Temperature -0.11560    0.04702  -2.458   0.0140 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
```

```
##     Null deviance: 24.230  on 22  degrees of freedom
## Residual deviance: 18.086  on 21  degrees of freedom
## AIC: 35.647
##
## Number of Fisher Scoring iterations: 5
```

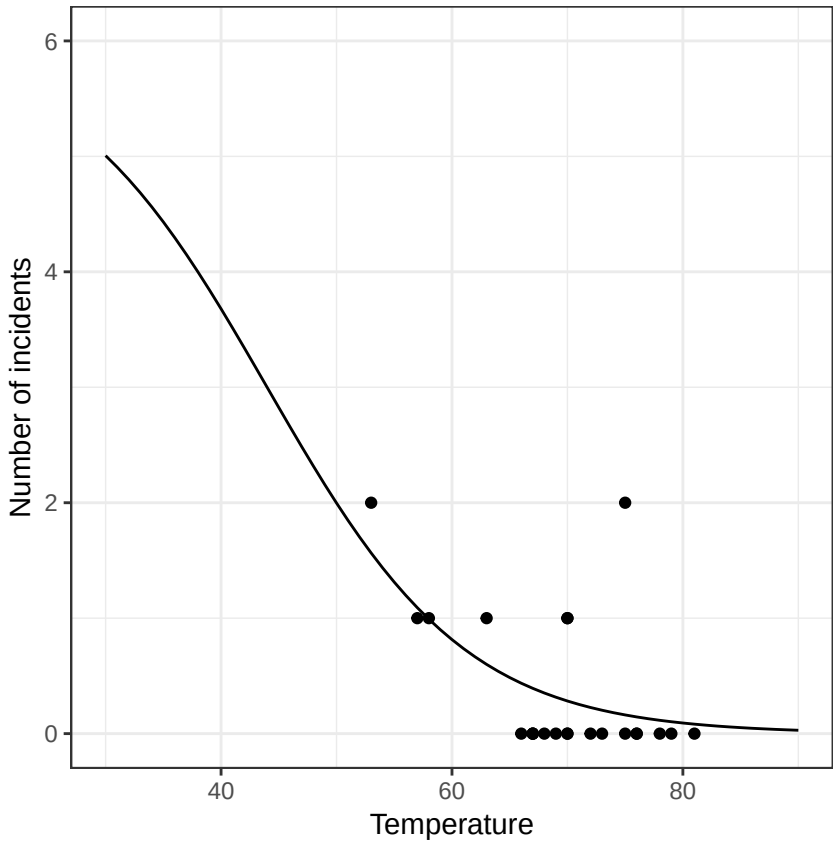
Les paramètres estimés sont $\hat{\alpha} = 5.085 \pm 3.052$ pour l'intercept et $\hat{\beta} = -0.1156 \pm 0.04702$ pour l'effet Température.

La déviance résiduelle est $G^2 = 18.086$ avec 21 degrés de liberté.

Probabilité d'incident à 31°F

Visualiser la courbe de prédiction sur le graphique :

```r
# Calculer les valeurs prédites par le modèle sur la gamme allant de 30 à 90°F
temp <- seq(30, 90)
preds_resp <- data.frame(fit=predict(logis_reg_binomial, list(Temperature=temp),
                                     type="response"))
preds_resp$temp <- temp

# La réponse étant sur l'échelle 0 à 1 (Malfunction/Count) on multiplie
# la réponse par 6 (valeur de Count) pour retrouve l'échelle initiale
ggplot(data=data, aes(x=Temperature, y=Malfunction)) +
  geom_point() +
  geom_line(data=preds_resp, aes(x=temp, y=fit*6)) +
  xlim(c(30,90)) + ylim(c(0,6)) +
  xlab("Temperature") + ylab("Number of incidents") +
  theme_bw() +
  theme(aspect.ratio=1)
```

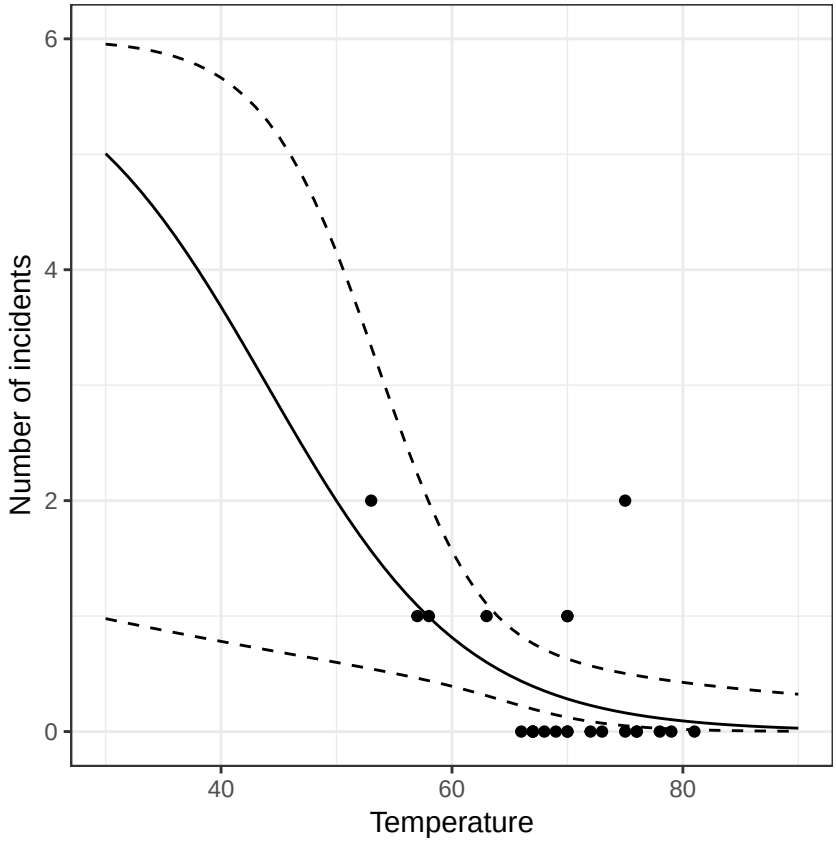


Le graphique ressemble bien à la figure 4 de l'article. La probabilité de dysfonctionnement d'un joint à 31°F est de 0.8178.

Maintenant essayons de calculer l'enveloppe de confiance autour des valeurs prédites.

```
# Pour calculer la bonne enveloppe de confiance on utilise l'option "se.fit=TRUE",
# mais il faut calculer les prédictions de l'erreur standard dans l'échelle linéaire
# du lien logit en choisissant l'option type="link"
preds_link <- predict(logis_reg_binomial, list(Temperature=temp),
                      type = "link", se.fit = TRUE)
preds_link$temp <- temp

upr <- preds_link$fit + (1.96 * preds_link$se.fit)
lwr <- preds_link$fit - (1.96 * preds_link$se.fit)
fit <- preds_link$fit

# Mainenant on utilise la fonction invers ("linkinv") pour convertir les valeurs
# dans l'échelle non linéaire de la réponse
upr2 <- logis_reg_binomial$family$linkinv(upr)
lwr2 <- logis_reg_binomial$family$linkinv(lwr)
# En utilisant la fonction inverse sur la prédiction de la réponse, on doit retrouver
# la même chose que ce que nous avons obtenu plus haut avec type="response"
fit2 <- logis_reg_binomial$family$linkinv(fit)

preds_2 <- data.frame(upr2=upr2, lwr2=lwr2, fit2=fit2)

ggplot(data=data, mapping=aes(x=Temperature, y=Malfunction)) + geom_point() +
  geom_line(data=preds_2, aes(x=temp, y=fit2*6)) +
```

```
  geom_line(data=preds_2, aes(x=temp, y=upr2*6), linetype=2) +
  geom_line(data=preds_2, aes(x=temp, y=lwr2*6), linetype=2) +
  xlim(c(30,90)) + ylim(c(0,6)) +
  xlab("Temperature") + ylab("Number of incidents") +
  theme_bw() +
  theme(aspect.ratio=1)
```



Nous obtenons ainsi un graphique proche des estimations de l'intervalle de confiance de prédiction à 90% obtenues par boostrap par les auteurs de l'article (voir figure 5). Ainsi, pour 30°F l'intervalle de confiance à 95% est de $[0.1631, 0.9924]$, en accord avec ce qui est attendu.