Rapport de projet OS13
Analyse de politique de maintenance

TRAN Quoc Nhat Han & Adrien WARTELLE

1er janvier 2019

Sommaire

Résumé

Soient des données liées à la fonctionnement de système, nous déterminons un modèle approprié et puis choisir une politique de maintenance optimal.

1 Maintenance basant sur l'âge

1.1 Rappel

Considérons un système non maintenu. En l'observant, nous obtenons un liste des dates de panne, grâce auquel nous construirerons une politique de remplacement systématique basée sur l'âge : *Nous remplaçons lorsque le système tombe en panne ou qu'il survit une durée* t_0.

Le but est de minimiser le coût moyen cumulé.

$$\mathbb{E}(C) = \frac{\mathbb{E}(C(S))}{\mathbb{E}(S)} \tag{1}$$

Où S est la variable aléatoire représentant la date de remplacement et $C(S)$ est le coût de maintenance cumulé à l'instant S (sachant que $C(S)$ est c_c si une maintenance corrective et c_p si préventive).

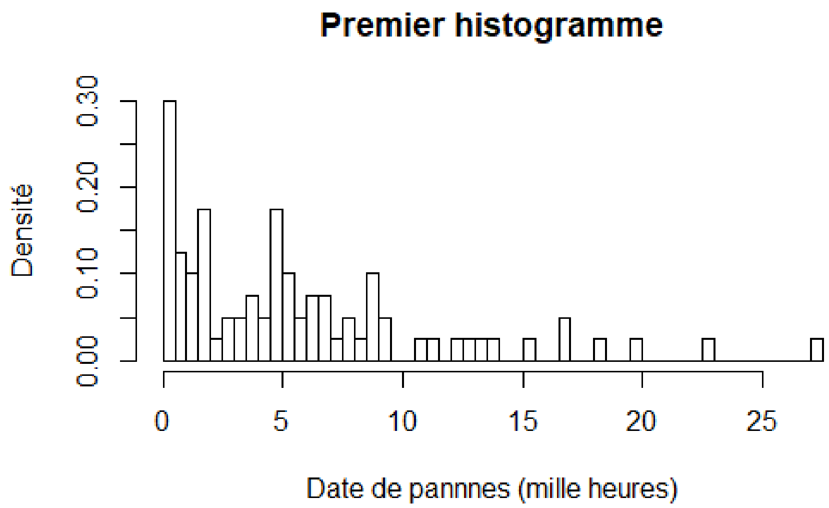

FIGURE 1 – Le premier histogramme de distribution de pannes

1.2 Modéliser la durée de vie du système

L'importation de données nous montre que les dates de pannes sont de l'ordre grandement variée (300 à 27000). Exponentiel des valeurs extrèmes résulteront Inf, ce qui est indésirable. Alors nous devons forcément les réduire en les divisant par un scalaire `scale`, prenons par example 1000. (Figure 1.2)

Les pannes se concentrent autour de 2 sommets, l'un à $[0; 0,5]$ et l'autre à $[4,5; 5]$. Ceci nous fait penser naturellement à un mixage de deux lois.

Comme les valeurs sont positives, et que l'un sommet se situe auprès de zéro et l'autre à une valeur non nulle, nous essayons d'estimer un mixage de loi *Exponentielle* et *Gamma*.

La fonction de densité avec le paramètre $\theta = (p_1, p_2, \lambda, \alpha, \beta)$:

$$\begin{aligned} f_\theta(x) &= p_1 f_1(x) + p_2 f_2(x) \\ &= p_1 \lambda e^{-\lambda x} + p_2 \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x} \end{aligned} \tag{2}$$

Où f_1, f_2 désignent réspectivement $exp(\lambda)$ et $\Gamma(\alpha, \beta)$; $p_1, p_2 > 0 : p_1 + p_2 = 1$.

Nous allons utiliser l'algorithme EM, la méthode la plus efficace pour estimer le MLE de mixage fini.

Soit $\mathbf{X} = (x_1, ..., x_N)$ le vecteur des données existants.

Soit la matrice de probabilité d'appartenance (ζ_{ki}). ζ_{ki} vaut la probabilité que x_i suive la loi f_k.

$$\zeta_{ki} = \frac{p_k f_k(x_i)}{p_1 f_1(x_i) + p_2 f_2(x_i)} \forall k = \overline{1,2} \forall i = \overline{1,N}$$

La fonction de vraisemblance :

$$\ln \Lambda = \sum_{i=1}^{N} \ln f_\theta (x_i) = \sum_{i=1}^{N} \ln (p_1 f_1 (x_i) + p_2 f_2 (x_i)) \tag{3}$$

Nous cherchons à maximiser $\ln \Lambda$ en la dérivant selon λ, α, β.
Pour λ :

$$\begin{aligned}
\frac{\partial}{\partial \lambda} \ln \Lambda &= \sum_{i=1}^{N} \frac{p_1 e^{-\lambda x_i} - p_1 \lambda x_i e^{-\lambda x_i}}{p_1 f_1 (x_i) + p_2 f_2 (x_i)} \\
&= \sum_{i=1}^{N} \frac{p_1 f_1 (x_i)}{p_1 f_1 (x_i) + p_2 f_2 (x_i)} \left(\frac{1}{\lambda} - x_i \right) \\
&= \frac{1}{\lambda} \sum_{i=1}^{N} \zeta_{1i} - \sum_{i=1}^{N} \zeta_{1i} x_i = 0 \\
\Leftrightarrow \lambda &= \frac{\sum\limits_{i=1}^{N} \zeta_{1i}}{\sum\limits_{i=1}^{N} \zeta_{1i} x_i} \qquad (4)
\end{aligned}$$

Pour β :

$$\begin{aligned}
\frac{\partial}{\partial \beta} \ln \Lambda &= \sum_{i=1}^{N} \frac{p_2 x_i^{\alpha-1}}{\Gamma(\alpha)} \frac{\alpha \beta^{\alpha-1} e^{-\beta x_i} - \beta^{\alpha} x_i e^{-\beta x_i}}{p_1 f_1 (x_i) + p_2 f_2 (x_i)} \\
&= \sum_{i=1}^{N} \frac{p_2 f_2 (x_i)}{p_1 f_1 (x_i) + p_2 f_2 (x_i)} \left(\frac{\alpha}{\beta} - x_i \right) \\
&= \frac{\alpha}{\beta} \sum_{i=1}^{N} \zeta_{2i} - \sum_{i=1}^{N} \zeta_{2i} x_i = 0 \\
\Leftrightarrow \beta &= \alpha \frac{\sum\limits_{i=1}^{N} \zeta_{2i}}{\sum\limits_{i=1}^{N} \zeta_{2i} x_i} \qquad (5)
\end{aligned}$$

Pour α, le calcul devient tellement difficile que la démonstration ici ne sera pas suffisante. Nous n'écrivons que le résultat approximatif de Newton-Rashphon : Fixer un entier non nul n_r indiquant le nombre d'itération :

$$\alpha_{r+1} = \alpha_r - \frac{\ln(\alpha_r) - \Psi(\alpha_r) - \ln\left(\frac{\sum_{i=1}^{N} \zeta_{2i} x_i}{\sum_{i=1}^{N} \zeta_{2i}} \right) + \frac{\sum_{i=1}^{N} \zeta_{2i} \ln(x_i)}{\sum_{i=1}^{N} \zeta_{2i}}}{\frac{1}{\alpha_r} - \Psi'(\alpha_r)} \tag{6}$$

Où Ψ et Ψ' sont les fonctions digamma et trigamma dans l'ordre.
Et puis, pour p_k :

$$p_k = \frac{\sum\limits_{i=1}^{N} \zeta_{ki}}{N} \forall k = \overline{1,2} \tag{7}$$

Etant donné (4), (5), (6) et (7), nous définissons l'algorithme EM :

1. **Initialisation :** Choisir un θ_{vieux}.
2. **Etape E :** Evaluer (ζ_{ki}) sachant θ_{vieux}.
3. **Etape M :** Calculer $\theta_{nouveau}$ à l'aide des équations (4), (5), (6) et (7).
   Note : Pour *alpha*, l'itération se termine quand $|\alpha_{r+1} - \alpha_r| < \varepsilon_\alpha$ avec ε_α est un réel positif fixé à l'initialisation.
4. **Evaluation :** Si $|\theta_{nouveau} - \theta_{vieux}| < \varepsilon_\theta$ (ε_θ est un réel positif fixé à l'initialisation), l'algorithme finit et $\theta = \theta_{vieux}$.
   Sinon, reviens à l'étape E avec $\theta_{vieux} \leftarrow \theta_{nouveau}$.

Nous définissons ensuite la fonction de vraisemblance négative et utilisons la méthode MLE (implémentée dans le package *stats4*). Puisque les valeurs sont bornées ($p \in]0; 1[$; $\lambda > 0$; $\alpha > 0$; $\beta > 0$), il nous faillait spécifier l'approche *L-BFGS-B*. Le code complet se trouve à l'annexe.

Malheuresement, le message d'erreur

```
Error in solve.default(oout$hessian) :
  Lapack routine dgesv: system is exactly singular: U[2,2] = 0
```

indique que les valeurs initiales, qui sont critiques pour cette méthode, étaient mal choisies en l'absence de théorèmes appropriés.

Nous devons passer à un mixage plus simple de deux lois Exponentielles. Prenons le cas où les deux coefficients d'intensité sont **différents**, car sinon nous retombe à la loi Exponentielle, qui est définitivement inappropriée.

La fonction de densité avec le paramètre $\theta = (p, \lambda, \mu)$:

$$f_\theta(x) = p\lambda e^{\lambda x} + (1 - p)\,\mu e^{\mu x} \tag{8}$$

2 Annexe

2.1 L'importation de données de pannes

```
pannes = read.csv(
    file = "FailureTimes_5.csv",
    header = TRUE,
    sep = ",",
    dec = ".",
    colClasses = c("NULL", NA)
)
scale = 1000
data = pannes$Heures / scale
N = length(data)
```

2.2 Le premier histogramme de distribution de pannes

```
hist(
    data,
    breaks = 40,
    probability = TRUE,
    xlab = "Date de pannnes (mille heures)",
    ylab = "Densité",
    main = "Premier histogramme"
)
```

2.3 Estimer le mixage de la loi Exponentielle et Gamma

```r
# Fitting mixture of Exp and Gamma
s = log(mean(data[data >= 2.5])) - mean(log(data[data >= 2.5]))
theta_melExpGamma = list(
    p1 = 0.5,
    p2 = 0.5,
    lambda = 1 / sum(data[data <= 2.5]),
    alpha = (3 - s + sqrt((s - 3)^2 + 24 * s)) / (12 * s),
    beta = mean(data[data >= 2.5]) / theta_melExpGamma$alpha
)
# PDF
f_melExpGamma = function(x, theta = theta_melExpGamma) {
    theta$p1 * dexp(data, rate = theta$p) + theta$p2 * dgamma(data, shape =
        theta$alpha, rate = theta$beta)
}
# CDF
F_melExpGamma = function(x, theta = theta_melExpGamma) {
    theta$p1 * pexp(data, rate = theta$p) + theta$p2 * pgamma(data, shape =
        theta$alpha, rate = theta$beta)
}

# Fitting distribution

epsilon = list(
    alpha = 1e-4,
    theta = 1e-4
)
zeta = matrix(
    0,
    nrow = 2,
    ncol = N
)

library(stats4)
nle_melExpGamma = function(
    p,
    lambda,
    alpha,
    beta
) {
    ## for debug purpose
    theta = list(
        p = p,
        lambda = lambda,
        alpha = alpha,
        beta = beta
    )
    cat("p=", theta$p,
        " lambda=", theta$lambda,
        " alpha=", theta$alpha,
        " beta=", theta$beta,
        "\n")
    - sum(log(f_melExpGamma(data, theta)))
}
fit_melExpGamma = mle(
    nle_melExpGamma,
    start = theta_melExpGamma,
    method = "L-BFGS-B",
    lower = c(
        0,
        1e-6,
        1e-6,
        1e-6
    ),
    upper = c(
        1,
        Inf,
        Inf,
        Inf
    ),
    control = list(
        trace = 6,
```

```
        maxit = 10000,
        ndeps = c(
            1e-7,
            1e-7,
            1e-7,
            1e-7
        )
    )
)
summary(fit_melExpGamma)
theta_melExpGamma$p = fit_melExpGamma@coef["p"]
theta_melExpGamma$lambda = fit_melExpGamma@coef["lambda"]
theta_melExpGamma$alpha = fit_melExpGamma@coef["alpha"]
theta_melExpGamma$beta = fit_melExpGamma@coef["beta"]
h_melExpGamma = hist(
    data,
    breaks = 40,
    probability = TRUE,
    main = "Fitting mixture of Exp and Gamma"
)
curve(
    f_melExpGamma(x),
    add = TRUE,
    col = "violet",
    from = min(h_melExpGamma$mids),
    to = max(h_melExpGamma$mids)
)
```