Rapport de projet OS13
Analyse de politique de maintenance

TRAN Quoc Nhat Han & Adrien WARTELLE

2 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).

1.2 Modéliser la durée de vie du système

L'importation de données (l'annexe 2.1) nous montre que les dates de pannes sont de l'ordre grandement variée (300 à 27000) (l'annexe 2.2). 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)

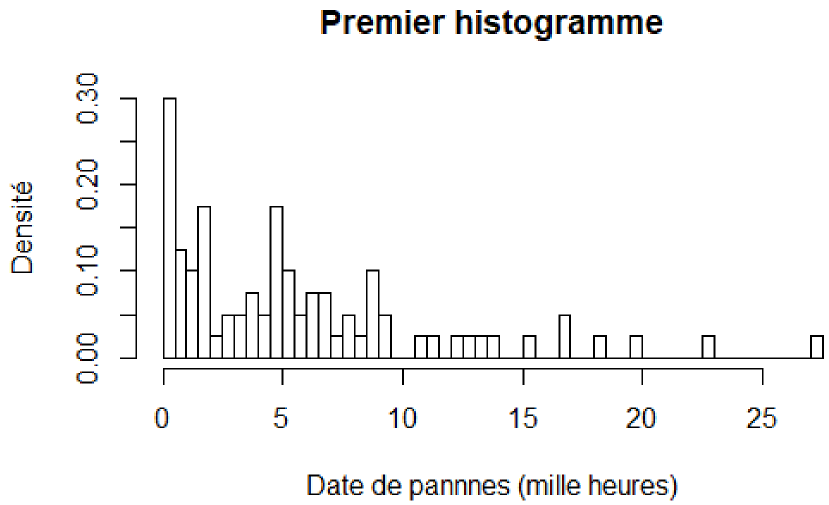

FIGURE 1 – Le premier histogramme de distribution de pannes

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 formule itératif de Newton-Rashphon [1] :

$$\frac{1}{\alpha_{r+1}} = \frac{1}{\alpha_r} + \frac{\ln (\alpha_r) - \Psi (\alpha_r) - c}{a_r^2 \left(\frac{1}{\alpha_r} - \Psi' (\alpha_r) \right)} \tag{6}$$

Où $c = \ln \left(\frac{\sum\limits_{i=1}^{N} \zeta_{2i} x_i}{\sum\limits_{i=1}^{N} \zeta_{2i}} \right) - \frac{\sum\limits_{i=1}^{N} \zeta_{2i} \ln(x_i)}{\sum\limits_{i=1}^{N} \zeta_{2i}}$; Ψ et Ψ' sont les fonctions digamma et trigamma dans l'ordre. L'itération part avec $\alpha_0 = \frac{0.5}{c}$.
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 α, l'itération se termine quand $|\alpha_{r+1} - \alpha_r| < \varepsilon_\alpha$ où 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 s'arrête et $\theta = \theta_{vieux}$.
   Sinon, reviens à l'étape E avec $\theta_{vieux} \leftarrow \theta_{nouveau}$.

Basé sur le résultat obtenu, nous traçons la fonction de densité f_θ trouvé (figure 2) et réalisons un test de Kolmogorov-Smirnov. $p - value = 0,9663111$ signifie $96,63\%$ de nous tromper si nous rejetons ce modèle. Nous acceptons donc le modèle trouvé. Le code est trouvable à l'annexe 2.3.

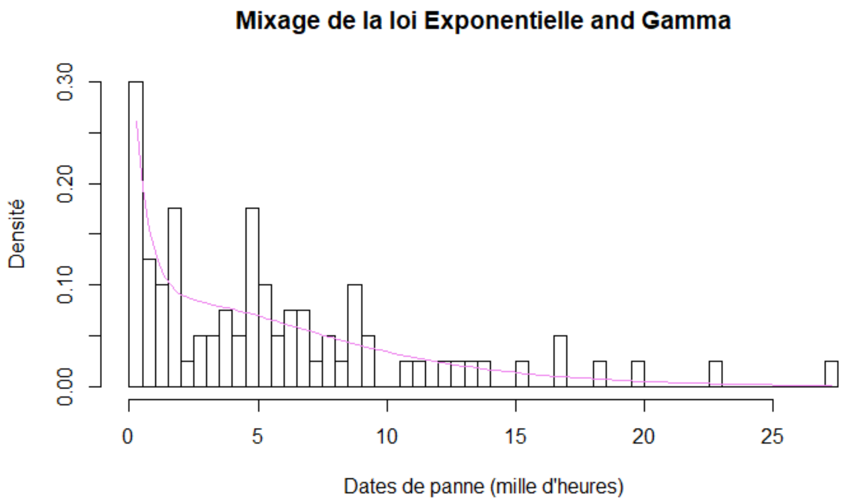

FIGURE 2 – Mixage de la loi Exponentielle et Gamma

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

```r
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
# Algorithm EM
# Initialisation
k = 2 # number of components
p = c(0.5, 0.5)
lambda = 1
alpha = 5
beta = 1
f = list(
    '1' = function(x) {
        dexp(x, rate = lambda)
    },
    '2' = function(x) {
        dgamma(x, shape = alpha, rate = beta)
    }
)
epsilon = list(
    alpha = 1e-4,
    theta = 1e-4
)
zeta = matrix(
    0,
    nrow = k,
    ncol = N
)
# Norm
normVec = function(x) sqrt(sum(x^2))
# New value
p_new = p
alpha_new = alpha
beta_new = beta
lambda_new = lambda
repeat {
    ## E Step
    # Calculate each proba
    for (l in 1:k) {
        zeta[l,] = p[[l]] * f[[l]](data)
    }
    # Normalize proba
    zeta = t(t(zeta) / rowSums(t(zeta)))
    ## M step
    # Lambda
    lambda_new = sum(zeta[1,]) / sum(zeta[1,] * data)
    # Alpha
    c = log(sum(zeta[2,] * data) / sum(zeta[2,])) - sum(zeta[2,] * log(data)) / sum(zeta[2,])
    alpha_new = 0.5 / c
    alpha_temp = 0
    repeat {
        alpha_temp = 1 / (1 / alpha_new + (log(alpha_new) - digamma(alpha_new) - c) / (alpha_new^2 * (1 / alpha_new - trigamma(alpha_new))))
        if (abs(alpha_temp - alpha_new) < epsilon$alpha) {
            break
        } else {
            alpha_new = alpha_temp
        }
    }
```

```r
    alpha_new = alpha_temp
    # Beta
    beta_new = alpha_new * sum(zeta[2,]) / sum(zeta[2,] * data)
    # P
    for (l in 1:k) {
        p_new[[l]] = mean(zeta[l,])
    }
    ## Evaluation
    if (normVec(c(alpha, beta, lambda, p[[1]], p[[2]]) - c(alpha_new, beta_new, lambda_new, p_new[[1]], p_new[[2]])) < epsilon$theta) {
        break
    } else {
        alpha = alpha_new
        beta = beta_new
        lambda = lambda_new
        p = p_new
    }
}
# Final value update
alpha = alpha_new
beta = beta_new
lambda = lambda_new
p = p_new

# Illustration
f_theta = function(x) {
    p[[1]] * f[[1]](x) + p[[2]] * f[[2]](x)
}
h_theta = hist(
    data,
    breaks = 40,
    probability = TRUE,
    main = "Mixage de la loi Exponentielle and Gamma",
    xlab = "Dates de panne (mille d'heures)",
    ylab = "Densité"
)
curve(
    f_theta(x),
    add = TRUE,
    col = "violet",
    from = min(h_theta$mids),
    to = max(h_theta$mids)
)
# Kolmogorov-Smirnov test
F_theta = function(x) {
    p[[1]] * pexp(x, rate = lambda) + p[[2]] * pgamma(x, shape = alpha, rate = beta)
}
test = ks.test(data, F_theta, exact = TRUE)
```

Références

[1] Minka, Thomas P. (2002). "Estimating a Gamma distribution"
https://tminka.github.io/papers/minka-gamma.pdf