

# Rapport de projet OS13

## Analyse de politique de maintenance

TRAN QUOC NHAT HAN & ADRIEN WARTELLE

3 janvier 2019

### Sommaire

<b>1</b>	<b>Maintenance basée sur l'âge</b>	<b>1</b>
1.1	Rappel . . . . .	1
1.2	Modéliser la durée de vie du système . . . . .	2
1.3	La politique de maintenance basée sur l'âge . . . . .	5
<b>2</b>	<b>Maintenance basée sur dégradation</b>	<b>6</b>
2.1	Rappel . . . . .	6
2.2	Modéliser la dégradation du système . . . . .	6
<b>3</b>	<b>Annexe</b>	<b>7</b>
3.1	L'importation de données de pannes . . . . .	7
3.2	Le premier histogramme de distribution de pannes . . . . .	7
3.3	Estimer le mixage de la loi Exponentielle et Gamma . . . . .	7
3.4	Optimiser le coût moyenne sur une durée de temps . . . . .	8
3.5	Importer les valeurs de dégradation . . . . .	9
3.6	Premiers traces de dégradation . . . . .	9

### 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 basée 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 construirons 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.

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

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  $c_c(= 1200)$  si une maintenance corrective et  $c_p(= 800)$  si préventive).

## 1.2 Modéliser la durée de vie du système

L'importation de données de **FailureTimes\_5.csv** (l'annexe 3.1) expose les dates de pannes de l'ordre grandement variée (300 à 27000) (l'annexe 3.2).

Exponentiel des valeurs extrêmes résulteront *Inf*, ce qui est indésirable. Alors nous devons forcément les réduire en les divisant par un scalaire **scale**, prenons par exemple 1000. (Figure 1)

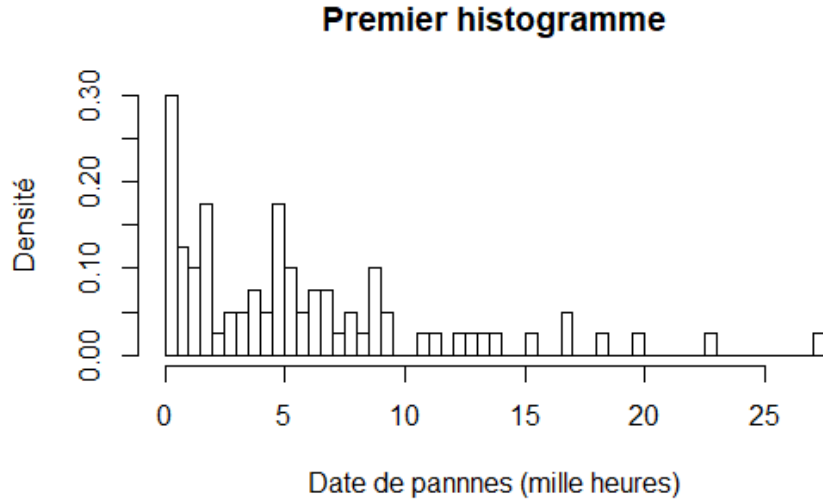


FIGURE 1 – Le premier histogramme de distribution de pannes

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 respectivement  $\exp(\lambda)$  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  $X$  la variable aléatoire de durée de vie du système. Soient  $(x_1, \dots, x_N)$  les observations.

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

$$\zeta_{ki} = \frac{p_k f_k(x_i)}{p_1 f_1(x_i) + p_2 f_2(x_i)} \forall k = \overline{1, 2} \forall i = \overline{1, N} \quad (3)$$

La fonction de vraisemblance :

$$\ln \Lambda = \sum_{i=1}^N \ln f_{\theta}(x_i) = \sum_{i=1}^N \ln (p_1 f_1(x_i) + p_2 f_2(x_i)) \quad (4)$$

Nous cherchons à maximiser  $\ln \Lambda$  en la dérivant selon  $\lambda, \alpha, \beta$ .

Pour  $\lambda$  :

$$\begin{aligned} \frac{\partial}{\partial \lambda} \ln \Lambda &= \sum_{i=1}^N \frac{p_1 e^{-\lambda x_i} - p_1 \lambda x_i e^{-\lambda x_i}}{p_1 f_1(x_i) + p_2 f_2(x_i)} \\ &= \sum_{i=1}^N \frac{p_1 f_1(x_i)}{p_1 f_1(x_i) + p_2 f_2(x_i)} \left( \frac{1}{\lambda} - x_i \right) \\ &= \frac{1}{\lambda} \sum_{i=1}^N \zeta_{1i} - \sum_{i=1}^N \zeta_{1i} x_i = 0 \\ \Leftrightarrow \lambda &= \frac{\sum_{i=1}^N \zeta_{1i}}{\sum_{i=1}^N \zeta_{1i} x_i} \end{aligned} \quad (5)$$

Pour  $\beta$  :

$$\begin{aligned} \frac{\partial}{\partial \beta} \ln \Lambda &= \sum_{i=1}^N \frac{p_2 x_i^{\alpha-1} \alpha \beta^{\alpha-1} e^{-\beta x_i} - \beta^{\alpha} x_i e^{-\beta x_i}}{\Gamma(\alpha) (p_1 f_1(x_i) + p_2 f_2(x_i))} \\ &= \sum_{i=1}^N \frac{p_2 f_2(x_i)}{p_1 f_1(x_i) + p_2 f_2(x_i)} \left( \frac{\alpha}{\beta} - x_i \right) \\ &= \frac{\alpha}{\beta} \sum_{i=1}^N \zeta_{2i} - \sum_{i=1}^N \zeta_{2i} x_i = 0 \\ \Leftrightarrow \beta &= \alpha \frac{\sum_{i=1}^N \zeta_{2i}}{\sum_{i=1}^N \zeta_{2i} x_i} \end{aligned} \quad (6)$$

Pour  $\alpha$  :

$$\begin{aligned}
\frac{\partial}{\partial \alpha} \ln \Lambda &= \sum_{i=1}^N \frac{p_2 e^{-\beta x_i}}{p_1 f_1(x) + p_2 f_2(x)} \left( \frac{\beta (\ln \beta + \ln x_i) (\beta x_i)^{\alpha-1}}{\Gamma(\alpha)} - \beta^\alpha x^{\alpha-1} \frac{\Psi(\alpha)}{\Gamma(\alpha)} \right) \\
&= \sum_{i=1}^N \frac{p_2 f_2(x_i)}{p_1 f_1(x) + p_2 f_2(x)} (\ln \beta + \ln x_i - \Psi(\alpha)) \\
&= \left( \sum_{i=1}^N \zeta_{2i} \right) \ln \beta + \sum_{i=1}^N \zeta_{2i} \ln x_i - \Psi(\alpha) \left( \sum_{i=1}^N \zeta_{2i} \right) = 0 \\
&\Leftrightarrow 0 = \ln \alpha + \ln \frac{\sum_{i=1}^N \zeta_{2i}}{\sum_{i=1}^N \zeta_{2i} x_i} + \frac{\sum_{i=1}^N \zeta_{2i} \ln x_i}{\sum_{i=1}^N \zeta_{2i}} - \Psi(\alpha) \quad (\text{substitué par (6)}) \\
&\Leftrightarrow 0 = \ln \alpha - \Psi(\alpha) - c
\end{aligned}$$

Où  $c = \ln \left( \frac{\sum_{i=1}^N \zeta_{2i} x_i}{\sum_{i=1}^N \zeta_{2i}} \right) - \frac{\sum_{i=1}^N \zeta_{2i} \ln(x_i)}{\sum_{i=1}^N \zeta_{2i}}$  ;  $\Psi$  est la fonction digamma.

Selon la méthode de Newton-Rashphon, nous pouvons résoudre  $\alpha$  numériquement avec ce formul itératif :

$$\alpha_{r+1} = \alpha_r - \frac{\ln \alpha_r + \Psi(\alpha_r) - c}{\frac{1}{\alpha_r} - \Psi'(\alpha_r)}$$

[1] propose un autre formule convergeant plus vite :

$$\frac{1}{\alpha_{r+1}} = \frac{1}{\alpha_r} + \frac{\ln(\alpha_r) - \Psi(\alpha_r) - c}{a_r^2 \left( \frac{1}{\alpha_r} - \Psi'(\alpha_r) \right)} \quad (7)$$

Avec  $\Psi'$  la fonction trigamma. L'itération part avec  $\alpha_0 = \frac{0.5}{c}$ .

Au final, pour  $p_k$  :

$$p_k = \frac{\sum_{i=1}^N \zeta_{ki}}{N} \forall k = \overline{1, 2} \quad (8)$$

Etant donné (3), (5), (6), (7) et (8), nous définissons l'algorithme EM :

1. **Initialisation** : Choisir un  $\theta_{vieux}$ .
2. **Etape E** : Evaluer  $(\zeta_{ki})$  sachant  $\theta_{vieux}$  en utilisant (3).
3. **Etape M** : Calculer  $\theta_{nouveau}$  à l'aide des équations (5), (6), (7) et (8).  
*Note* : Pour  $\alpha$ , l'itération se termine quand  $|\alpha_{r+1} - \alpha_r| < \varepsilon_\alpha$  où  $\varepsilon_\alpha$  est un réel positif fixé à l'initialisation.
4. **Evaluation** : Si  $\|\theta_{c+1} - \theta_c\| < \varepsilon_\theta$  ( $\varepsilon_\theta$  est un réel positif fixé à l'initialisation), l'algorithme s'arrête et  $\theta = \theta_{vieux}$ .  
Sinon, reviens à l'étape E avec  $\theta_{vieux} \leftarrow \theta_{nouveau}$ .

Le résultat obtenu :

$$(p_1, p_2, \lambda, \alpha, \beta) = (0.2194518; 0.7805482; 1.56738; 1.665659; 0.2332427)$$

D'où nous traçons la fonction de densité  $f_\theta$  trouvé (figure 2) et réalisons un test de Kolmogorov-Smirnov qui donne  $p\text{-value} = 0,9663111$  signifiant 96,63% de nous tromper si nous rejetons ce modèle. Nous l'acceptons alors, quoiqu'il ne génère pas 2 sommets comme la remarque initiale. Le code est trouvable à l'annexe 3.3.

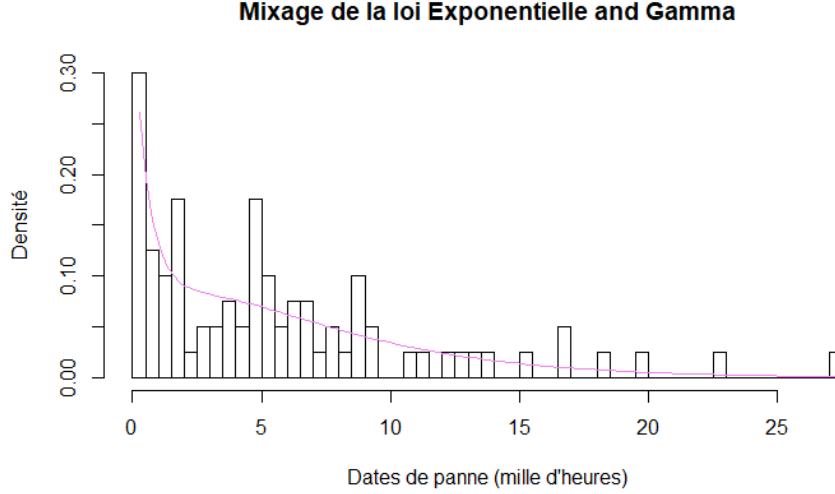


FIGURE 2 – Mixage de la loi Exponentielle et Gamma

### 1.3 La politique de maintenance basée sur l'âge

Avec la fonction  $f_\theta$  trouvée, nous construirons la politique optimale.

Nous avons par définition :  $S = \min(X, t_0)$ .

Autrement dit,  $S = X\mathbb{I}_{\{X < t_0\}} + t_0\mathbb{I}_{\{X \geq t_0\}}$ .

Traduit au coût :  $C(S) = C_c\mathbb{I}_{\{X < t_0\}} + C_p\mathbb{I}_{\{X \geq t_0\}}$ , avec  $C_c, C_p$  les coûts de maintenances correctives et préventives respectivement.

Le coût moyen :

$$\begin{aligned}
 \mathbb{E}(C(S)) &= c_c\mathbb{E}(\mathbb{I}_{\{X < t_0\}}) + c_p\mathbb{E}(\mathbb{I}_{\{X \geq t_0\}}) \\
 &= c_cP(X < t_0) + c_pP(X \geq t_0) \\
 &= c_cF_\theta(t_0) + c_p(1 - F_\theta(t_0)) \\
 &= (c_c - c_p)F_\theta(t_0) + c_p
 \end{aligned} \tag{9}$$

La durée moyenne :

$$\begin{aligned}
\mathbb{E}(S) &= \mathbb{E}(X\mathbb{I}_{\{X < t_0\}}) + t_0\mathbb{E}(\mathbb{I}_{\{X \geq t_0\}}) \\
&= \int_0^{t_0} x f_\theta(x) dx + t_0 P(X \geq t_0) \\
&= x F_\theta(x) \Big|_0^{t_0} - \int_0^{t_0} F_\theta(x) dx + t_0 (1 - F_\theta(t_0)) \\
&= t_0 - \int_0^{t_0} F_\theta(x) dx
\end{aligned} \tag{10}$$

De (9) et (10), nous détaillons le coût moyen sur une durée de temps (1) :

$$\mathbb{E}(C) = \frac{\mathbb{E}(C(S))}{\mathbb{E}(S)} = \frac{(c_c - c_p) F_\theta(t_0) + c_p}{t_0 - \int_0^{t_0} F_\theta(x) dx} \tag{11}$$

L'annexe (3.4) montrer comment chercher l'optimum numériquement. La valeur minimum est  $t_0 = 27,29639$  (mille heures), correspondant à un coût moyen de 210,6402. Nous constatons que  $t_0^{min}$  est très proche du maximum de durée de vie, indiquant que l'optimisation de  $t_0$  est inutile car le système ne vieillit pas.

## 2 Maintenance basée sur dégradation

### 2.1 Rappel

En observant multiples systèmes identiques, nous effectuons des mesures de dégradation sur des intervalles de temps réguliers tout au long de leurs durée de vie.

La valeur limite de dégradation est  $L = 20$ . C'est-à-dire lorsque le niveau de dégradation dépasse  $L$ , le système tombe en panne et nous ne pourrions plus le mesurer.

On souhaite de mettre en place une politique de maintenance conditionnelle, basée sur un seuil  $M$  inférieur à  $L$  et l'intervalle de temps  $\Delta T$  entre les inspections. Appellons  $X_t$  le niveau de dégradation à l'instant  $t$  (instant d'une inspection).

- Si  $X_t < M$ , nous laissons le système tel quel.
- Si  $M \leq X_t < L$ , un remplacement préventif est réalisé au coût  $c_p$ . Et puis  $X_t$  est remis à 0.
- Si  $X_t \geq L$ , un remplacement correctif est fait au coût  $c_c$ . Et puis  $X_t$  est remis à 0.

Le but est minimiser le coût moyen sur une durée de temps (1) en bien choisissant le seuil  $M$  et l'intervalle d'inspection  $\Delta T$ .

### 2.2 Modéliser la dégradation du système

Soient les données de `DegradLevel_2.csv`, nous traçons leurs processus de dégrader. (Les annexes (3.5) et (3.6))

## 3 Annexe

### 3.1 L'importation de données de pannes

```
1 pannes = read.csv(  
2   file = "FailureTimes_5.csv",  
3   header = TRUE,  
4   sep = ",",  
5   dec = ".",  
6   colClasses = c("NULL", NA)  
7 )  
8 scale = 1000  
9 data = pannes$Heures / scale  
10 N = length(data)
```

### 3.2 Le premier histogramme de distribution de pannes

```
1 hist(  
2   data,  
3   breaks = 40,  
4   probability = TRUE,  
5   xlab = "Date de pannes (mille heures)",  
6   ylab = "Densité",  
7   main = "Premier histogramme"  
8 )
```

### 3.3 Estimer le mixage de la loi Exponentielle et Gamma

```
1 # Fitting mixture of Exp and Gamma  
2 # Algorithm EM  
3 # Initialisation  
4 k = 2 # number of components  
5 p = c(0.5, 0.5)  
6 lambda = 1  
7 alpha = 5  
8 beta = 1  
9 f = list(  
10   '1' = function(x) {  
11     dexp(x, rate = lambda)  
12   },  
13   '2' = function(x) {  
14     dgamma(x, shape = alpha, rate = beta)  
15   }  
16 )  
17 epsilon = list(  
18   alpha = 1e-4,  
19   theta = 1e-4  
20 )  
21 zeta = matrix(  
22   0,  
23   nrow = k,  
24   ncol = N  
25 )  
26 # Norm  
27 normVec = function(x) sqrt(sum(x^2))  
28 # New value  
29 p_new = p  
30 alpha_new = alpha  
31 beta_new = beta  
32 lambda_new = lambda  
33 repeat {  
34   ## E Step  
35   # Calculate each proba  
36   for (l in 1:k) {  
37     zeta[l,] = p[[l]] * f[[l]](data)  
38   }
```

```

39 # Normalize proba
40 zeta = t(t(zeta) / rowSums(t(zeta)))
41 ## M step
42 # Lambda
43 lambda_new = sum(zeta[1,]) / sum(zeta[1,] * data)
44 # Alpha
45 c = log(sum(zeta[2,] * data) / sum(zeta[2,])) - sum(zeta[2,] * log(data))
46   / sum(zeta[2,])
47 alpha_new = 0.5 / c
48 alpha_temp = 0
49 repeat {
50   alpha_temp = 1 / (1 / alpha_new + (log(alpha_new) - digamma(alpha_new)
51   ) - c) / (alpha_new^2 * (1 / alpha_new - trigamma(alpha_new)))
52   if (abs(alpha_temp - alpha_new) < epsilon$alpha) {
53     break
54   } else {
55     alpha_new = alpha_temp
56   }
57 }
58 alpha_new = alpha_temp
59 # Beta
60 beta_new = alpha_new * sum(zeta[2,]) / sum(zeta[2,] * data)
61 # p
62 for (l in 1:k) {
63   p_new[[l]] = mean(zeta[l,])
64 }
65 ## Evaluation
66 if (normVec(c(alpha, beta, lambda, p[[1]], p[[2]]) - c(alpha_new, beta_
67 new, lambda_new, p_new[[1]], p_new[[2]])) < epsilon$theta) {
68   break
69 } else {
70   alpha = alpha_new
71   beta = beta_new
72   lambda = lambda_new
73   p = p_new
74 }
75 }
76 # Final value update
77 alpha = alpha_new
78 beta = beta_new
79 lambda = lambda_new
80 p = p_new
81 # Illustration
82 f_theta = function(x) {
83   p[[1]] * f[[1]](x) + p[[2]] * f[[2]](x)
84 }
85 h_theta = hist(
86   data,
87   breaks = 40,
88   probability = TRUE,
89   main = "Mixage de la loi Exponentielle et Gamma",
90   xlab = "Dates de panne (mille d'heures)",
91   ylab = "Densité"
92 )
93 curve(
94   f_theta(x),
95   add = TRUE,
96   col = "violet",
97   from = min(h_theta$mids),
98   to = max(h_theta$mids)
99 )
100 # Kolmogorov-Smirnov test
101 F_theta = function(x) {
102   p[[1]] * pexp(x, rate = lambda) + p[[2]] * pgamma(x, shape = alpha, rate
103     = beta)
104 }
105 test = ks.test(data, F_theta, exact = TRUE)

```

### 3.4 Optimiser le coût moyenne sur une durée de temps



```

1 # Finding optimal t_0
2 # Given F_theta
3 c_c = 1200
4 c_p = 800
5 E_C_S = function(x) {
6   (c_c - c_p) * F_theta(x) + c_p
7 }
8 E_S = function(x) {
9   x - integrate(F_theta,0,x)$value
10 }
11 E_C = function(x) {
12   E_C_S(x) / E_S(x)
13 }
14 o = optimize(
15   E_C,
16   c(min(data),max(data)),
17   tol = 1e-5
18 )
19 d = seq(
20   10,
21   30,
22   0.01
23 )
24 plot(
25   d,
26   lapply(
27     d,
28     E_C
29   ),
30   main = "Coût moyenne sur une durée de temps",
31   xlab = "t_0",
32   ylab = "",
33   type = "l"
34 )

```

### 3.5 Importer les valeurs de dégradation

```

1 # Import degradation
2 table = read.csv(
3   file = "DegradLevel_2.csv",
4   header = TRUE,
5   sep = ",",
6   dec = "."
7 )
8 time = table$Temps
9 nbProcess = length(table) - 2
10 process = vector("list", nbProcess)
11 for (i in 1:nbProcess) {
12   process[[i]] = table[[i + 2]]
13 }

```

### 3.6 Premiers traces de dégradation

```

1 # Plot process curves

```

## Références

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