VAM, OUTIL LOGICIEL ADAPTATIF ET LIBRE POUR PRENDRE EN COMPTE L'EFFET DES MAINTENANCES ET DU VIEILLISSEMENT

VAM, AN ADAPTATIVE OPEN SOURCE SOFTWARE TO TAKE INTO ACCOUNT THE EFFECT OF MAINTENANCES AND AGEING

Laurent Doyen, Rémy Drouilhet
Univ. Grenoble Alpes
Laboratoire LJK, 51 rue des mathématiques, 38 041 Grenoble Cedex 9
laurent.doyen@imag.fr, remy.drouilhet@upmf-grenoble.fr

Résumé

Les hypothèses basiques sur l'effet de la maintenance sont "maintenance parfaite" ou As Good As New (le système est remis à neuf) et "maintenance minimale" ou As Bad As Old (la maintenance n'a aucun effet sur l'occurrence des défaillances futurs). En pratique, il est clair que pour des systèmes industriels importants on se situe entre ces deux cas extrêmes. Les modèles de maintenance imparfaite proposent une description d'un effet intermédiaire des maintenances. Dans cette soumission nous proposons une démonstration de notre solution logicielle VAM. VAM, pour Virtual Age Model, est un package R open source mettant en œuvre les principaux modèles de maintenance imparfaite. L'utilisation de VAM est basée sur l'écriture d'une formule qui décrit le jeu de données et le modèle utilisé pour l'analyser. Cette formule rend le package très adaptatif car c'est l'utilisateur qui va y définir le comportement du système neuf non maintenu, les types et effets des différentes maintenances préventives et correctives, et la façon dont la maintenance préventive est planifiée. On peut ensuite utiliser les fonctionnalités du package pour simuler de nouveaux jeux de données, estimer par maximum de vraisemblance les paramètres du modèle, calculer et représenter graphiquement différents indicateurs.

Summary

The basic assumptions on maintenance efficiency are known as perfect maintenance or As Good As New (the system is renewed) and minimal maintenance or As Bad As Old (maintenance has no effect on future failures occurrences). Obviously reality falls between this two extreme cases. An intermediate maintenance effect can be described thanks to imperfect maintenance models. This paper presents an introductory tutorial for our VAM software. VAM, for Virtual Age Model, is an R open source package that implements the principal imperfect maintenance models. VAM usage is based on a formula which specify the characteristic of the data set to analyse and the model used for that. Thanks to this formula description the package becomes adaptative. In fact, the formula is defined by the user and characterises the behaviour of the new unmaintained system, the types and effects of the preventive and corrective maintenances, and how preventive maintenance times are planned. Then, the package functionalities enable to simulate new data sets, to estimate with maximum likelihood method the parameters of the model, to calculate and plot different indicators.

Introduction

L'exploitation d'un matériel industriel réparable doit être conduite avec le double souci de la sécurité et de l'efficacité technico-économique. Dans ce contexte, la fonction maintenance revêt une importance stratégique déterminante. En effet, elle contribue de façon essentielle à la fiabilité opérationnelle. Elle joue également un rôle important dans la gestion des risques et constitue un élément crucial dans le maintien des performances des équipements. Pour l'industriel, il y a donc un enjeu majeur à l'évaluation quantitative de l'effet des maintenances et à l'optimisation de ces actions dans le respect des contraintes de sécurité, de disponibilité et de coût (Doyen et al., 2015).

Les hypothèses de base sur l'efficacité de la maintenance sont connues sous les noms de maintenance minimale ou As Bad As Old (ABAO) et maintenance parfaite ou As Good As New (AGAN). Dans le cas ABAO, chaque maintenance remet le système en fonctionnement dans l'état exact où il était juste avant la défaillance. Les processus aléatoires correspondants sont les processus de Poisson non homogènes (NHPP). Le plus utilisé des NHPP est le processus de puissance (Power-Law Process) ou modèle de Duane. Ce modèle est aux systèmes réparables ce que la loi de Weibull est aux systèmes non réparables. Dans le cas AGAN, chaque maintenance répare parfaitement le système, c'est-à-dire le remet à neuf. Les processus aléatoires correspondant sont les processus de renouvellement. En pratique, il est clair que l'on se situe entre ces deux extrêmes. En effet, il est raisonnable de penser que la maintenance a un effet plus que minimal, c'est-à-dire que le système après maintenance est meilleur que vieux. Pour autant, pour des systèmes industriels importants, il est peu vraisemblable que la maintenance remette le système à neuf. Le système après maintenance est donc moins bon que neuf. Cette situation intermédiaire se retrouve dans la littérature sous le nom de maintenance imparfaite. De nombreux modèles de maintenance imparfaite ont été proposés (Pham et al., 1996). Mais les plus utilisés sont certainement les modèles d'âge virtuel (Kijima, 1989) qui supposent que l'effet de la maintenance est de rajeunir le système.

En collaboration avec EDF R&D, le Laboratoire Jean Kuntzmann (LJK) a développé (E. Rémy et al., 2013) le logiciel MARS (Maintenance Assessment of Repairable Systems). MARS permet la modélisation et l'évaluation du vieillissement et de l'efficacité des maintenances. Il est disponible gratuitement sur autorisation de EDF R&D. Il a rencontré un vif succès et a été téléchargé à la fois par des universitaires et des industriels. Il nous a également conduit, au sein du LJK, à collaborer avec de nouveaux partenaires industriels sur la problématique de la maintenance imparfaite : Hydro Québec, ENGIE, SNCF Infra et Schneider Electric. Lors de ces études nous avons pu nous rendre compte de l'intérêt et de l'originalité du logiciel MARS, mais aussi de certaines limitations. Tout d'abord, MARS a été développé dans le but de pouvoir être utilisé par des ingénieurs de maintenance, les différentes fonctionnalités sont donc accessibles via une interface graphique conviviale. Cette interface était incontournable pour permettre l'utilisation de MARS par des ingénieurs, mais elle s'avère complexe à maintenir et à faire évoluer pour s'adapter à la diversité des problématiques rencontrées. D'autre part, MARS a été développé pour analyser des jeux de données avec un nombre limité de matériels en parallèles, chaque matériel étant observé pendant une durée relativement importante. Or, nos nouvelles collaborations nous ont amené à considérer un autre type de structure de données dû à un nombre important (voir très important) de matériels

en parallèle, chaque matériel n'étant observé que pendant un temps relativement restreint et ne rencontrant donc que très peu de défaillances. Enfin, MARS ne permet de prendre en compte qu'un seul type de maintenance préventive.

La nécessité de faire évoluer MARS nous a conduit à développer au sein du LJK une nouvelle solution logicielle plus adaptative. Pour ce faire, nous avons choisi de créer un package R. R (Lafaye de Micheaux et al., 2011) est un logiciel libre de traitement des données et d'analyses statistiques. Si R dispose dans sa version de base de la plupart des fonctionnalités utiles pour la statistique courante, ses possibilités s'élargissent dès que l'on utilise les packages (ou "extensions") mis librement à disposition. Ces packages couvrent un très large champ. Cependant, R reste très peu utilisé dans le domaine de la fiabilité au détriment d'autres logiciels commerciaux, très bien implantés sur le marché. Notre idée est donc de commencer à étoffer l'éventail des packages R avec des librairies spécifiques à la fiabilité en commençant par les modèles de maintenance imparfaite. L'un des intérêts majeurs de développer notre package sous R est de pouvoir ensuite interfacer simplement les nouvelles fonctionnalités que nous proposons avec le considérable ensemble de ressources déjà préexistantes sous R.

L'objectif de cette communication est de montrer comment on peut utiliser notre nouveau package R, nommé VAM pour Virtual Age Model, pour modéliser et évaluer l'effet conjoint des maintenances et du vieillissement.

Hypothèses et modélisation mathématiques

1 Les différents types de maintenances

On distingue deux grands types de maintenance.

- La maintenance corrective (MC) "exécutée après détection d'une panne et destinée à remettre un bien dans un état dans lequel il peut accomplir une fonction requise" (NF EN 13306).
- La maintenance préventive (MP) "exécutée à des intervalles prédéterminés ou selon des critères prescrits et destinée à réduire la probabilité de défaillance ou la dégradation du fonctionnement d'un bien" (NF EN 13306).

Parmi les MP, la norme NF EN 13306:20102 distingue :

- la maintenance systématique exécutée à intervalles de temps préétablis ou selon un nombre défini d'unités d'usage mais sans contrôle préalable de l'état du bien ;
- la maintenance conditionnelle qui comprend une combinaison de surveillance en fonctionnement et/ou d'inspection et/ou d'essai, d'analyse et les actions de maintenance qui en découlent.

Du point de vue de la modélisation probabiliste des dates de MP, ce qui distingue la maintenance conditionnelle, c'est le fait que, après avoir fait une maintenance, on ne connaît pas la date de la prochaine MP : elle sera déterminée en fonction des futures observations de la surveillance (non encore connues au moment de la dernière maintenance) faites sur le matériel. Ainsi, conditionnellement à l'ensemble des observations faites à la date de la dernière maintenance, la date de la prochaine MP conditionnelle est aléatoire. Ce n'est pas le cas pour la maintenance systématique. Parmi les maintenances systématiques, le cas le plus simple est celui où l'ensemble des dates de MP suit un planning de maintenance préétabli avant la mise en service du matériel. Dans ce cas, les dates de MP sont déterministes. On peut aussi avoir des dates de MP systématiques aléatoires. C'est par exemple le cas quand, suite à une analyse de la maintenance basée sur la fiabilité, on est amené à modifier la périodicité des MP suite à l'apparition d'un trop grand nombre de défaillances. Même si les dates de MP sont alors périodiques, elles doivent être considérées comme aléatoires car ce changement dans la périodicité n'avait pas été prévu initialement. Un autre exemple est le cas où le planning de MP est modifié suite à une optimisation de critères utilisant les dates de défaillance déjà observées.

Le package VAM permet de prendre en compte des MC et des MP systématiques. Dans sa version actuelle, il ne permet pas de considérer de MP conditionnelles. D'autre part VAM ne permet pas, pour l'instant, de considérer des types différents de MC, alors qu'il peut considérer plusieurs types différents de MP. Enfin les durées de réalisation des différentes actions de maintenance ne sont pas prises en compte.

2 Les modèles d'âge virtuel

Pour un système, on note $\{T_i\}_{i\geq 1}$ (avec $T_0 = 0$) les instants successifs de maintenance et $\{U_i\}_{i\geq 1}$ leur type.

- $U_i = -1$ si la $i^{\text{ème}}$maintenance est une MC,
- $U_i = 1$, ou 2, ou ..., si c'est une MP (1,2,..., représentent les différents types de MP possibles).

Pour tout instant $t \geq 0$, on note également N_t le nombre de maintenances observées entre 0 et t. D'un point de vue mathématique, la suite des instants de maintenance d'un matériel réparable forment un processus aléatoire ponctuel. Il peut être caractérisé par son intensité de défaillance.

$$\forall t \geq 0, \quad \lambda_t = \lim_{\Delta t \to 0} \frac{1}{\Delta t} P(N_{t+\Delta t} - N_{t^-} = 1 \,|\, N_{t^-}, U_{N_{t^-}+1} = -1, T_{N_{t^-}}, U_{N_{t^-}}, ..., T_1, U_1), \qquad \{1\}$$

où N_{t^-} représente la limite à gauche de N_t, c'est-à-dire le nombre de défaillances observées avant t (sans prendre en compte ce qui s'est passé en t). Pour Δt suffisamment petit, $\Delta t \times \lambda_t$ représente la probabilité d'avoir une défaillance entre t et $t + \Delta t$ sachant l'ensemble des MC et MP déjà réalisées jusqu'à l'instant t.

De façon générale, l'intensité d'un modèle d'âge virtuel vérifie :

$$\forall t \geq 0, \quad \lambda_t = \varepsilon'_{N_{t^-}}(t)\, h(\varepsilon_{N_{t^-}}(t)), \qquad \{2\}$$

avec

- $h(x)$ qui représente le taux de défaillance du système neuf non maintenue, appelé taux de défaillance initial.
- $\varepsilon'_k(t)$ qui est la dérivée par rapport t de $\varepsilon_k(t)$.
- $\varepsilon_k(t)$, pour $k \in \{0, 1, ...\}$ et $t \in]T_k, T_{k+1}]$, qui représente l'âge virtuel du système à l'instant t. C'est une variable aléatoire dépendant des dates et types des k premiers instants de maintenance.

Cela signifie qu'après la k ème maintenance, le système est équivalent à un système neuf non maintenu qui aurait déjà survécu un temps $\varepsilon_k(T_k)$ sans tomber en panne. En effet, la probabilité conditionnelle de défaillance à l'instant t vérifie pour tout $k \geq 0$:

$$\forall t \geq 0, \quad P(T_{k+1} > t \,|\, U_{k+1} = -1, T_k, U_k, ..., T_1, U_1) = \mathbb{1}_{\{t > T_k\}} e^{-(H(\varepsilon_k(t)) - (H(\varepsilon_k(T_k))))} = P(T_1 > \varepsilon_k(t) \,|\, T_1 > \varepsilon_k(T_k), U_1 = -1) \qquad \{3\}$$

avec $H(t) = \int_0^t h(u)\,du$ (voir Doyen et Gaudoin 2011 pour plus de précisions). Un modèle d'âge virtuel est caractérisé par une expression particulière des âges virtuels. On retrouve ainsi les modèles basiques. Un effet ABAO revient à considérer que l'âge virtuel est égal au temps calendaire :

$$\forall k \in \mathbb{N}, \forall t \geq 0, \quad \varepsilon_k(t) = t. \tag{4}$$

Et un effet AGAN suppose que l'âge virtuel est réinitialisé après chaque maintenance.

$$\forall k \in \mathbb{N}, \forall t \geq 0, \quad \varepsilon_k(t) = t - T_k. \tag{5}$$

Les intensités de défaillance correspondantes sont respectivement représentées pour des jeux de données simulées (avec un taux de défaillance initiale de type Weibull : $h(x) = 0.025x^{1.5}$) sur les figures 1. Les étoiles sur l'axe des abscisses représentent les instants de défaillance.

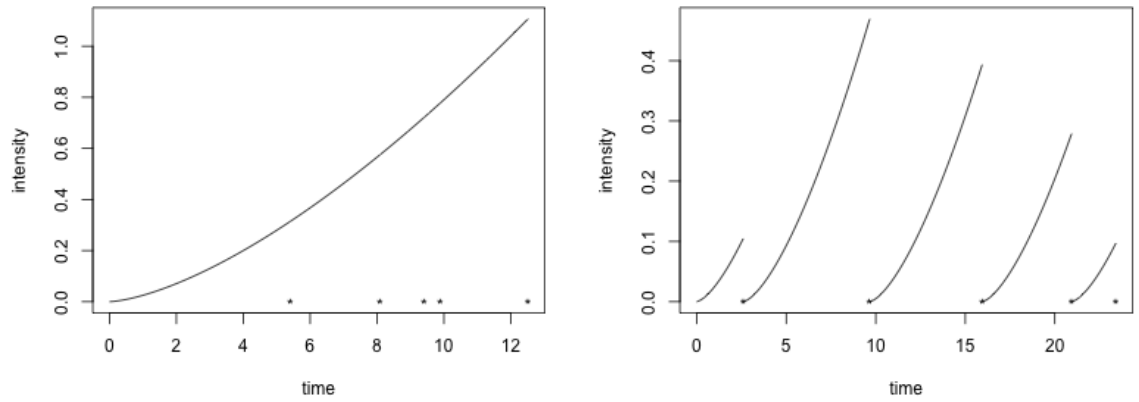

Figure 1. Intensité dans les cas ABAO (à gauche) et AGAN (à droite).

Enfin, dans le cas où le système est soumis à des MP, pour définir complètement le processus des MC et MP, il faut également caractériser la politique de MP, c'est à dire la façon dont les dates et types de MP sont choisis. Par exemple, la plus simple des politiques de maintenance consiste à réaliser des MP périodiques. Mais on peut aussi concevoir des politiques plus évoluées.

Le package VAM

3 Un modèle décrit par une formule

La très grande souplesse du package VAM vient de l'utilisation de *"formule"* comme cela se fait en général en R. La forme générique d'une formule VAM est :

$$Reponse \sim (Effet\_MC\,|\,h)\,\&\,(Effet\_MP1 + Effet\_MP2 + \ldots\,|\,Politique1\_MP * Politique2\_MP * \ldots) \tag{6}$$

La partie à droite du $\sim$ décrit le modèle que l'on souhaite utiliser, et la partie gauche décrit les éventuelles données à expliquer avec le modèle.

h décrit la loi de durée de vie du système neuf non maintenu, c'est à dire le taux de défaillance initial. Le package VAM permet pour l'instant de considérer les cas suivants.

- $Weibull(a,b)$ pour une loi de Weibull avec la paramétrisation $h(x) = a\,b\,x^{b-1}$ avec $a > 0$ et $b > 0$.
- $LogLinear(a,b)$ pour une loi log-linéaire avec la paramétrisation $h(x) = a\,e^{bx}$ avec $a > 0$ et $b > 0$.
- $Weibull3(a,b,c)$ pour une loi de Weibull décalée avec la paramétrisation $h(x) = a\,b\,(x+c)^{b-1}$ avec $a > 0$, $b > 0$ et $c > 0$.

D'autres lois de survie sont à l'étude.

$Effet\_MC$ et $Effet\_MP1$, $Effet\_MP2$, ..., décrivent les modèles d'âge virtuel utilisés pour caractériser respectivement l'effet des MC et des éventuelles différents types de MP considérés. Toutes les MP d'un même type sont supposées avoir le même effet au sens où elles suivent le même modèle d'effet avec les mêmes paramètres. Les différents modèles d'âge virtuel disponibles dans le package VAM sont les suivants.

- $AGAN()$ pour AGAN.
- $ABAO()$ pour ABAO.
- $AGAP()$ pour As Good As Previous. La maintenance remet le matériel dans le même état où il était juste après la précédente maintenance.
- $QAGAN()$ pour Quasi As Good As New. La maintenance remet l'age virtuel à zéro sans nécessairement remettre à neuf le matériel. C'est à dire que l'effet de la maintenance réduit l'âge virtuel du système de sa valeur avant la maintenance.
- $ARAInf(rho)$ pour Arithmetic Reduction of Age (ARA) avec une mémoire infinie. La maintenance réduit l'âge virtuel de *rho* fois sa valeur avant la maintenance.
- $ARA1(rho)$ pour ARA avec une mémoire 1. La maintenance réduit l'âge virtuel de *rho* fois le supplément d'âge virtuel accumulé entre les deux dernières maintenances.
- $ARAm(rho\,|\,m)$ pour ARA avec une mémoire m. Dans le modèle $ARAInf$ la réduction est proportionnelle à la valeur globale de l'âge, alors que dans le modèle $ARA1$ on ne réduit que le supplément d'âge entre les deux dernières maintenances. L'idée du modèle $ARAm$ est d'avoir un effet qui porte sur les m derniers temps entre maintenances.

- $QR(rho)$ pour Quasi Renewal (aussi connu sous le nom de geometric process). Soit $Y_1, Y_2, ...$, des variables aléatoires indépendantes et de même loi que la durée de vie du système neuf non maintenu. Considérons un système soumis uniquement à des MC, le modèle QR classique suppose que les durées successives entre défaillances ont la même loi que $Y_1, aY_2, a^2Y_3, ...$, avec $a = 1/rho$.
- $GQR(rho\,|\,f)$ pour Generalized Quasi Renewal (aussi connu sous le nom de extended geometric process). L'évolution géométrique du modèle QR semble souvent trop explosive pour être réaliste d'un point de vue pratique. L'idée du GQR est donc de remplacer le a^k par un $a^{f(k)}$ ou f est une fonction croissante. Le package VAM propose pour f les fonctions : $identity$ pour $f(k) = k$ (le GQR correspond alors à un QR), $sqrt$ pour $f(k) = \sqrt{k}$, et log pour $f(k) = \ln(k+1)$.
- $GQR\_ARAInf(rho\,|\,f)$, $GQR\_ARA1(rho\,|\,f)$, $GQR\_ARAm(rho\,|\,f,m)$ qui appliquent simultanément un effet ARA et GQR. Pour ces modèles le paramètre d'effet de maintenance rho est un vecteur de dimension deux (représentant dans l'ordre les efficacités des modèles GQR et ARA correspondants).

Le package VAM permet de combiner l'ensemble de ces modèles d'âge virtuel dans toutes les configurations possibles et avec n'importe quel nombre de types différents de MP.

Enfin, $Politique1\_MP$, $Politique2\_MP$, ..., décrit les éventuelles politiques de MP. Les politiques de MP pour l'instant disponibles sont :

- $periodic$ pour des MP périodiques,
- $AtTimes$ pour des MP à des dates spécifiées non nécessairement périodiques,
- $AtIntensity$ pour des MP dès que l'intensité de défaillance atteint un certain seuil,
- $AtVirtualAge$ pour des MP dès que l'âge virtuel atteint un certain seuil,
- $AtFailureProbability$ pour des MP dès que la probabilité conditionnelle de défaillance atteint un certain seuil.

Nous expliciterons par la suite le fonctionnement et l'intérêt de ces différentes politiques de MP. D'autres politiques de MP sont à l'étude.

4 Un système soumis uniquement à des MC

L'instruction *sim.vam* permet de construire un simulateur. Par exemple, l'instruction ci-dessous crée un simulateur pour un modèle avec un taux de défaillance initial Weibull et des MC ARAInf d'efficacité 0.3.

```
> simMC <- sim.vam( ~ (ARAInf(0.3) | Weibull(0.01,3)) )
```

Les parties en police verbatim, comme ci-dessus, correspondent à des instructions successives tapées sous R et aux sorties correspondantes. Dans la formule ci-dessus, on n'est pas obligé de compléter la partie gauche de la formule (avant le $\sim$). En effet, dans le cas d'un simulateur, on n'a pas de jeu de données à analyser : on est en train d'en créer un. La fonction *simulate* permet ensuite de simuler selon ce modèle un jeu de données constitué par exemple de 20 instants successifs de maintenance.

```
> DataMC <- simulate(simMC, 20)
```

Par défaut, la fonction *simulate* génère une *Data Frame*, c'est à dire une structure de données R. Chaque ligne correspond à un événement. La première colonne, nommée *Time* par défaut, contient les instants successifs et la deuxième colonne, nommée *Type* par défaut, donne les types correspondants. Ils valent ici toujours -1 puisqu'il y a uniquement de MC.

```
> DataMC
        Time Type
1   5.099485   -1
2   6.775430   -1
3   7.824516   -1
...
18 27.381875   -1
19 30.094268   -1
20 30.733159   -1
```

Chaque nouvel appelle à la fonction *simulate* va générer un nouveau jeu de données. Le dernier jeu de données simulé reste stocké dans l'objet *sim.vam* (c'est à dire dans l'exemple ci-dessus *simMC*).

L'instruction *mle.vam* permet de créer un objet d'estimation paramétrique. La méthode d'estimation utilisée est le maximum de vraisemblance. Par exemple, l'instruction ci-dessous crée, pour le jeu de données précédemment simulé, une méthode d'estimation pour un modèle avec un taux de défaillance initial Weibull et des MC ARAInf. Le jeu de données simulées peut sans aucun problème être remplacé par des données réelles, il suffit de créer sa propre *Data Frame* avec les mêmes caractéristiques que celles produites par la fonction *simulate*.

```
> mleMC_ARAInf <- mle.vam( Time & Type ~ (ARAInf(0.5) | Weibull(1,2.5)), data=DataMC)
```

La partie gauche de la formule ci-dessus précise le nom des colonnes dans lesquelles on trouve dans l'ordre les instants successifs de maintenance et leurs types dans la Data Frame *DataMC*. Les valeurs des paramètres dans la partie droite de la formule servent d'initialisation pour la méthode d'optimisation de la log-vraisemblance. Différents algorithmes d'optimisation sont disponibles, on peut également contraindre ou fixer la valeur de certains paramètres. La fonction *coef* renvoie les valeurs estimées des paramètres (d'abord ceux du taux initial puis ceux des différents effets de maintenance dans l'ordre où ils apparaissent dans la formule), et la fonction *formula* renvoie la formule correspondant au modèle estimé (en prenant pour valeur des paramètres les valeurs estimées).

```
> coef(mleMC_ARAInf)
[1] 0.002881542 3.576837667 0.243315132
> formula(mleMC_ARAInf)
Time & Type ~ (ARAInf(0.243315131540241) | Weibull(0.00288154234094617,
    3.57683766702878))
```

Les estimations sont dans cette exemple de très bonne qualité puisqu'elles sont très proches des vraies valeurs qui ont servi à générer les données (cf. la déclaration de *simMC*).

La fonction *plot* permet de tracer différentes caractéristiques : intensité de défaillance (*plot* de type *"i"*), intensité de défaillance cumulées ("I"), âge virtuel ("v"), fonction de survie conditionnelle du prochain instant de défaillance ("S"), ...

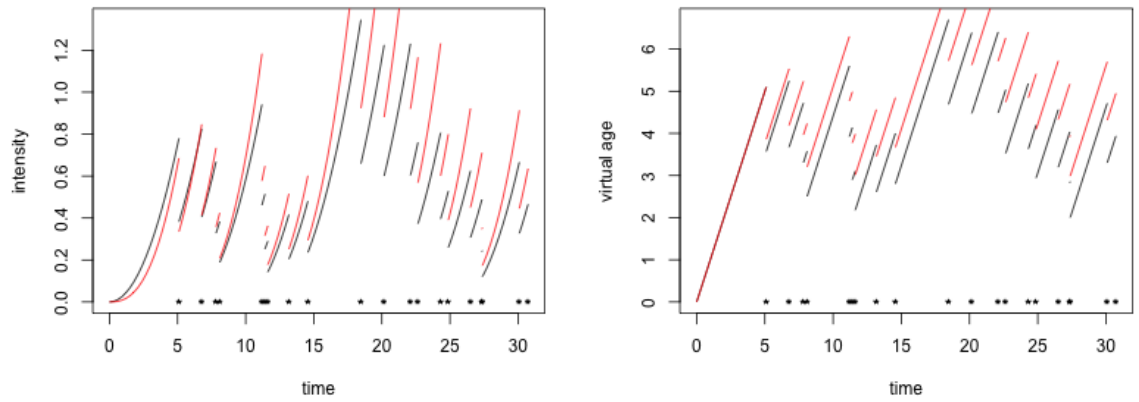

Figure 2. Intensité (à gauche) et âge virtuel (à droite) pour des MC ARAInf.

```
> plot(simMC, 'i'); plot(mleMC_ARAInf, 'i', col='red', add=TRUE)
> plot(simMC, 'v'); plot(mleMC_ARAInf, 'v', col='red', add=TRUE)
```

La première ligne d'instructions ci-dessus permet de générer la partie gauche de la figure 2 et la deuxième ligne génère la partie droite. Les graphes superposent l'intensité (respectivement l'âge virtuel) pour le vrai modèle (ayant servi à générer les données simulées, *simMC*), en noire, et pour le modèle estimé (à partir des données, *mleMC_ARAInf*), en rouge. Aussi bien pour le vrai modèle que celui estimé, on voit que les MC sont efficaces. Par exemple, pour le vrai modèle, l'âge du système après la MC est égal à $1-0.3=70\%$ de ce qu'il était avant la MC. L'intensité de défaillance correspond alors simplement à appliquer la forme du taux de défaillance initiale en l'âge virtuel.
L'avantage d'implanter nos fonctionnalités dans un langage en ligne de commande, comme le R, est qu'on peut ensuite facilement créer de nouveaux programmes utilisant les fonctionnalités du package et également les interfacer avec le nombre considérable de ressources déjà préexistantes en R. Par exemple, les quelques lignes de code ci-dessous mettent en œuvre une procédure de Monte Carlo afin d'appréhender la qualité d'estimation à laquelle on peut s'attendre avec un jeu de données similaire à celui que l'on vient de simuler. L'histogramme ainsi obtenu est représenté sur la figure 3.

```
> MonteCarrlo<- function() { update(mleMC_ARAInf, simulate(simMC,20)); coef(mleMC_ARAInf)[3] }
> Est_rho<- replicate(500,MonteCarrlo())
> summary(Est_rho)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.09356 0.24510 0.28900 0.32060 0.36220 1.18700
> hist(Est_rho)
> abline(v=0.4,lwd=4,col="red")
```

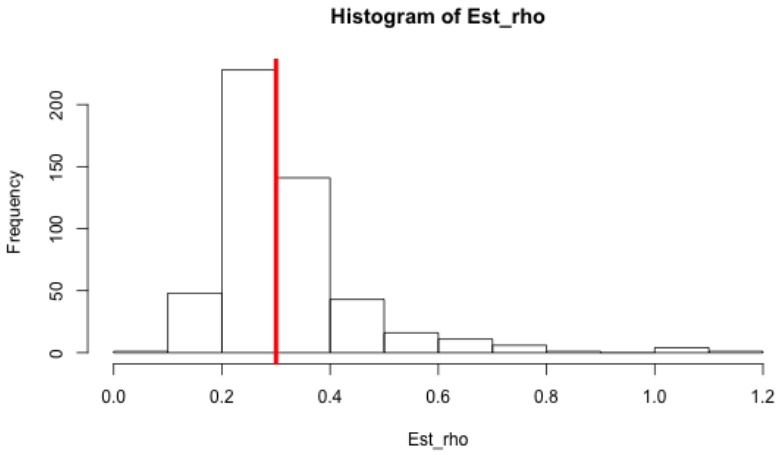

Figure 3. Histogramme de la distribution des estimations de l'effet de MC ARAInf.

Dans ce cas, on a donc, avec 20 observations successives, un estimateur de l'efficacité de MC de bonne qualité : il est centré sur la vraie valeur et il a une dispersion qui semble raisonnable. De façon générale ce genre de calcul peut requérir un temps d'exécution assez long en R, car R est un langage interprété et non compilé. Cependant, ce n'est pas le cas pour les fonctionnalités de notre package, car toute les étapes de calculs intensifs sont programmées et compilées en C++ via le package Rcpp.

On peut refaire le travail de simulation, de tracé de graphique (figure 4) et d'estimation sur un modèle avec un même taux de défaillance initial de type Weibull et des MC QR d'efficacité 0.9. Dans ce cas, les durées entre défaillances successives suivent alors des lois de Weibull indépendantes de même paramètre de forme. L'effet de chaque maintenance se traduit par le fait que l'espérance de la durée entre défaillances suivante est multipliée par 1/0.9 par rapport à la précédente.

```
> simMC_QR <- sim.vam( ~ (QR(0.9) | Weibull(0.01,3)) )
> DataMC_QR <- simulate(simMC_QR, 20)
> mleMC_QR <- mle.vam( Time & Type ~ (QR(0.5) | Weibull(1,2.5)), data=DataMC_QR)
> coef(mleMC_QR)
[1] 0.002149178 3.648930667 0.912420123
> plot(simMC_QR, 'i'); plot(mleMC_QR, 'i', col='red', add=TRUE)
> plot(simMC_QR, 'v'); plot(mleMC_QR, 'v', col='red', add=TRUE)
```

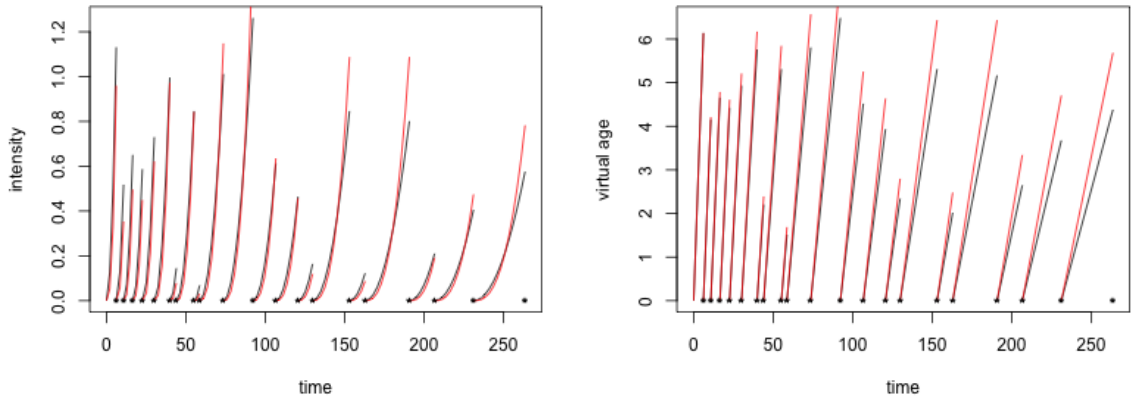

Figure 4. Intensité (à gauche) et âge virtuel (à droite) pour des MC QR.

5 Plusieurs systèmes soumis uniquement à des MC

Il peut s'avérer intéressant d'un point de vue statistique de considérer simultanément plusieurs systèmes identiques et indépendants. L'instruction suivante utilise le simulateur *simMC* construit précédemment pour simuler un groupe de 6 systèmes identiques et indépendants dont l'observation est censurée pour chaque système à la date 8.

```
> (DatasMC<-simulate(simMC,T>(RightCensorship=8),nb.system=6))
   System     Time Type
1       1 5.526632   -1
2       1 6.192353   -1
3       1 7.459635   -1
4       1 8.000000    0
5       2 1.794627   -1
6       2 2.521671   -1
7       2 7.052157   -1
...
23      6 6.276466   -1
24      6 8.000000    0
```

La *Data Frame* contient maintenant une colonne supplémentaire, nommée *System* par défaut, qui précise pour chaque événement à quel système il correspond. Pour chaque système le dernier événement a lieu dans ce cas à la date 8 et il est du type 0 : c'est la censure.

On peut estimer les valeurs des paramètres en considérant simultanément les données des 6 systèmes identiques et indépendants. Les temps de censure sont bien entendu pris en compte.

```
> mleMC_ARAInf <- mle.vam( System & Time & Type ~ (ARAInf(0.5) | Weibull(1,2.5)), data=DatasMC)
> formula(mleMC_ARAInf)
System & Time & Type ~ (ARAInf(0.500182671606073) | Weibull(0.0166216220929515,
    3.00112537180088))
```

Les valeurs estimées des paramètres sont à nouveau assez proches de celles utilisées pour simuler les données (cf. la déclaration de *simMC*).

6 Systèmes soumis à des MC et des MP

Considérons maintenant un système avec :

- une durée de vie du système neuf non maintenue de type log-linéaire,
- des maintenances correctives ARA1 d'efficacité 0.8,
- des maintenances préventives ARAInf d'efficacité 0.4 effectuées périodiquement toutes les 7 unités de temps,
- des maintenances préventives AGAN effectuées dès que l'intensité de défaillance dépasse le seuil de 2.5.

```
> simMC_MP <- sim.vam( ~ (ARA1(0.8) | LogLinear(0.03,0.8))
+   & (ARAInf(0.4) + AGAN() | Periodic(7) * AtIntensity(2.5)) )
> (DataMC_MP<-simulate(simMC_MP, 15, nb.system=4))
   System      Time Type
1       1  5.173527   -1
2       1  7.000000    1
3       1  9.761614   -1
```

```
4        1 10.238326   -1
5        1 10.401342   -1
6        1 13.474834   -1
7        1 14.000000    1
8        1 16.648120   -1
9        1 18.759020   -1
10       1 19.706530   -1
11       1 21.000000    1
12       1 23.794432    2
...
59       4 25.747482   -1
60       4 26.154539   -1
> plot(simMC_MP,'i',system.index=2); abline(h=2.5, col='green')
```

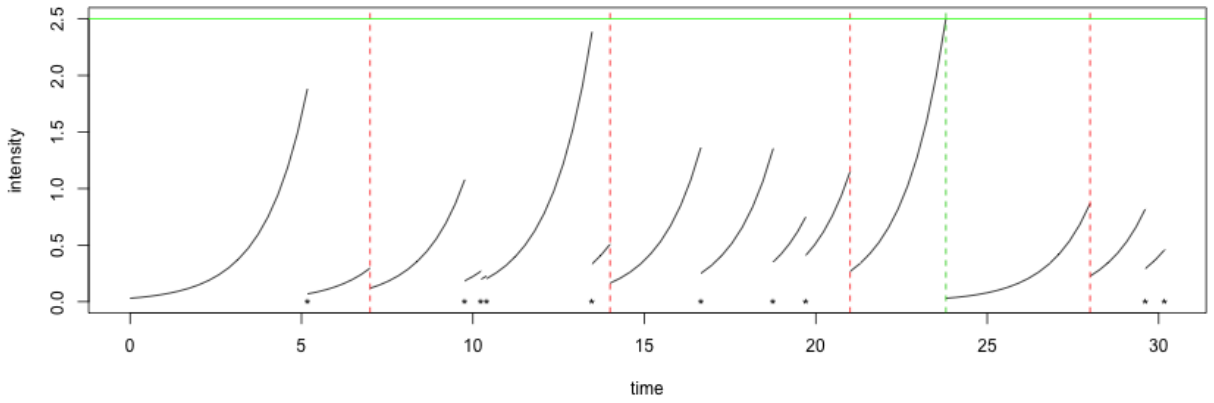

Figure 5. Intensité pour des MC ARA1 et des MP ARA3 et AGAN.

On a simulé 4 systèmes identiques et indépendants et on représente sur la figure 5 uniquement l'intensité pour le deuxième système (ce qui correspond à la dernière ligne des instructions ci-dessus). Les MC réduisent le supplément d'âge accumulé depuis la dernière maintenance de 80%. Il y a maintenant deux types de MP. Les MP de type 1 (trait verticaux rouge sur la figure 5) sont réalisées tous les 7 unités de temps et réduisent l'âge virtuel global de 40%. Les MP de type 2 (trait verticaux vert sur la figure 5) sont réalisées dès que l'intensité de défaillance atteint le seuil de 2.5 et elles renouvellent le système.

On peut à partir du jeu de données simulé (constitué des données des 4 systèmes identiques et indépendants) essayer de réestimer les valeurs des paramètres.

```
> mleMC_MP <- mle.vam( System & Time & Type ~ (ARA1(0.5) | LogLinear(1,0.5))
+  & (ARAInf(0.5) + ARAInf(0.5)), data=DataMC_MP )
> coef(mleMC_MP)
[1] 0.01262576 0.91514762 0.89931670 0.26306759 0.86403132
```

On retrouve bien, encore une fois, des valeurs proches de celles qui ont servi à simuler les données. En particulier, les MP de type 2, qui renouvellent le système, sont estimées avec un modèle ARAInf comme ayant une efficacité assez proche de 1 ce qui est équivalent à un effet AGAN.

Conclusion

Avec le package VAM nous proposons une solution logicielle très adaptative permettant la mise en œuvre de nombreux modèles de maintenance imparfaite. La grande flexibilité du package vient en grande partie de l'utilisation d'une formule définie à l'utilisation et décrivant les données à analyser et le modèle à mettre en œuvre. Cette formule permet de combiner comme on le souhaite les modèles entre eux avec en particulier un nombre quelconque de types différents de MP. L'utilisation de ce package nécessite une programmation en ligne de commande pour mettre en œuvre les différentes fonctionnalités. Le package s'adresse donc a priori plus à des statisticiens qu'à des ingénieurs en maintenance. Pour autant il existe sous R des packages permettant de développer simplement des surcouches graphiques et de créer des interfaces spécifiques à chaque utilisation.

Nous prévoyons de soumettre le package VAM pour le rendre disponible sur le CRAN. En attendant vous pouvez y avoir accès simplement sur demande auprès des auteurs. Le package est disponible en open source. A ce titre il n'a pas pour l'instant de concurrent. Une équipe de chercheurs de différentes universités brésiliennes (en particulier Gustavo Gilardoni de l'université de Brasilia et Maria Luìza Guerra de Toledo de l'institut brésilienne de géographie et statistiques) avec qui nous sommes en contact sont en train de développer un autre package R sur les modèles de maintenance imparfaite mais plus orienté vers des problématiques d'optimisation de la maintenance. Nous essayons de monter un projet afin d'associer nos compétences et en particulier d'arriver à créer un package commun. Parmi les concurrents indirects, nous avions précédemment développer le logiciel MARS dont nous avons parlé dans l'introduction et dont l'exécutable est disponible sur autorisation de EDF R&D. Il existe également une solution commerciale à travers le logiciel Rexpert.

Une première version stable et documentée du package est finalisée. Cette première version du package nous permet d'envisager un grand nombre de travaux théoriques et expérimentaux qui pourront déboucher sur de nouvelles fonctionnalités pour le package. Nous envisageons déjà les développements suivants.

- Dans un certain nombre d'étude pratique, on ne dispose pas de l'ensemble de l'historique des actions menées sur le matériel étudié. L'historique ne remonte souvent que jusqu'à la date de mise en place du système de GMAO. Il nous a donc déjà été demandé d'intégrer la possibilité de prendre en compte de la censure à gauche du processus des défaillances.
- Comme pour les MP, on sait parfois qu'il est nécessaire de considérer différents types d'effet de CM.
- Avec la multiplicité de modèles que permet de considérer le package VAM il nous semble vraiment important de développer des procédures de choix de modèle (AIC, test d'hypothèse). Nous avons déjà commencé à travailler sur différents tests d'adéquation (Chauvel et al., 2016).
- On ne dispose parfois que de très petits échantillons de données, dont on sait que la taille sera de toute façon insuffisante pour espérer obtenir des estimateurs de bonne qualité. Par contre, on peut compenser ce manque de données par des informations d'expert sur l'effet des maintenances par exemple. Afin de pouvoir prendre en compte cette expertise nous avons déjà commencé à travailler sur des méthodes d'estimation bayésienne (Corset et al., 2012). Nous venons tout juste de finir le développement de méthodes d'estimation bayésienne dans le package VAM. Encore une fois c'est l'utilisateur qui définit par le biais de la formule la loi à priori de chacun des paramètres. Nous sommes encore en phase de test pour ces nouvelles fonctionalités bayésiennes.
- Dans le cas inverse où on dispose de beaucoup de données nous avons également commencé à travailler sur des méthodes d'estimation semi-paramétrique (Beutner et al., 2015). Nous envisageons à terme d'intégrer ce genre d'outil dans le package VAM.
- Pour l'instant le package ne propose que des estimations ponctuelles. Or il est parfois indispensable d'appréhender la qualité des estimations obtenues. Cela peut se faire à travers des intervalles de confiances. Le problème est que les instants successifs de maintenances ne constituent pas un échantillon de variables indépendantes et identiquement distribuées. On ne sait donc pas si les méthodes classiques de construction d'intervalle de confiance asymptotiques sont alors valides. Il y a donc ici un important travail méthodologique à faire. On peut également envisager pour répondre à ce genre de problématiques des méthodes de type bootstrap, bootstrap non paramétrique ou des intervalles de crédibilité basés sur des estimations bayésiennes.
- Enfin reste à envisager le "vaste monde" des maintenances prédictives...

7 Références

L. Doyen et E. Rémy 2015, Effet conjoint du vieillissement et de la maintenance : modélisation, évaluation, optimisation, Bivi AFNOR 2015, MAR-VI-10-65.
H. Pham et H. Wang 1996, Imperfect maintenance, European Journal of Operational Research, 94(3), p. 425-8.
M. Kijima 1989, Some results for repairable systems with general repair, Journal of Applied Probability, 26(9), p. 89-102.
E. Rémy et F. Corset et S. Despréaux et L. Doyen et O. Gaudoin 2013, An example of integrated approach for the technical and economic optimization of maintenance, Reliability Engineering and System Safety, 116, p. 8-19.
P. Lafaye de Micheaux et R. Drouilhet et B. Liquet 2011, Le logiciel R: Maîtriser le langage - Effectuer des analyses statistiques, Springer Science & Business Media.
L. Doyen et O. Gaudoin 2011, Modelling and assessment of aging and efficiency of corrective and planned preventive maintenance, IEEE Transactions on Reliability, 60 (4), p. 759-769.
C. Chauvel et J.Y. Dauxois et L. Doyen et O. Gaudoin 2016, Parametric bootstrap goodness-of-fit tests for imperfect maintenance models, MIMAR 2016 - 9 th IMA International Conference on Modelling in Industrial Maintenance and Reliability, Londre, Royaume-Uni.
F. Corset et L. Doyen et O. Gaudoin 2012, Bayesian analysis of ARA imperfect repair models, Communications in Statistics - Theory and Methods, 41 (21), p. 3915-3941.
E. Beutner et L. Bordes et L. Doyen 2015, Virtual age model with unknown event effects, IV WASA - Workshop on Survival Analysis and Applications, Belo Horizonte, Brazil.