ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À
L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

COMME EXIGENCE PARTIELLE
À L'OBTENTION DE LA
MAÎTRISE EN GÉNIE MÉCANIQUE
M.Ing.

PAR
Mickaël SZCZOTA

# MODÉLISATION DE L'HISTORIQUE D'OPÉRATION DE GROUPES TURBINE-ALTERNATEUR

MONTRÉAL, LE 28 JUIN 2012

**PRÉSENTATION DU JURY**

CE MÉMOIRE A ÉTÉ ÉVALUÉ

PAR UN JURY COMPOSÉ DE :

M. Souheil-Antoine Tahan, directeur de mémoire
Département de génie mécanique, École de technologie supérieure

M. Luc Marcouiller, codirecteur
Expertise Mécanique, Métallurgie et Hydroéolien, Institut de Recherche d'Hydro-Québec

M. Michel Rioux, président du jury
Département de génie de la production automatisée, École de technologie supérieure

M. André Coutu, examinateur externe
Analyses mécaniques Hydro grande puissance, Andritz Hydro Ltée

M. Jean-Pierre Kenné, membre du jury
Département de génie mécanique, École de technologie supérieure

IL A FAIT L'OBJET D'UNE SOUTENANCE DEVANT JURY ET PUBLIC

LE 22 JUIN 2012

À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

## REMERCIEMENTS

Je tiens à remercier toutes les personnes impliquées dans ce projet :

- M. Souheil-Antoine Tahan du département de génie mécanique de l'École de technologie supérieure pour avoir eu confiance en moi, pour son appui et sa disponibilité.
- M. Luc Marcouiller, chercheur à l'institut de recherche d'Hydro-Québec (IREQ), dont la grande connaissance du fonctionnement des groupes turbine-alternateur a été d'une aide précieuse.
- M. Martin Gagnon, doctorant à l'école de technologie supérieure qui a été un collègue et un appui important tout au long du projet. Son perfectionnisme et son expérience m'ont permis de progresser.
- M. Denis Thibault, chercheur à l'institut de recherche d'Hydro-Québec, pour m'avoir accueilli sur son projet et pour son soutien constant.

Je tiens également à remercier tout le personnel de l'unité mécanique, métallurgie et Hydro-éolien qui ont su rendre mes séjours à l'IREQ très agréables. Parmi eux, l'expertise en statistique de M.Luc Perreault a été d'une grande aide. Merci au programme Mitacs Accélération, à Hydro-Québec qui ont conjointement financé mes travaux de recherche. Un merci particulier à Josée qui m'a supporté (dans les deux sens du terme) tout au long de cette maîtrise et à mes parents dont l'éducation m'a permis d'arriver là où je suis. Finalement, merci aux membres du jury, d'avoir pris le temps de lire et d'évaluer ce mémoire.

# MODÉLISATION DE L'HISTORIQUE D'OPÉRATION DE GROUPES TURBINE-ALTERNATEUR

Mickaël SZCZOTA

## RÉSUMÉ

À cause du vieillissement de leurs actifs, les gestionnaires du parc de production d'Hydro-Québec ont un besoin accru d'outils d'aide à la planification des opérations de maintenance. Un projet de recherche et développement a alors été initié en 2010, avec pour objectif de classifier les groupes turbine-alternateur d'une centrale en fonction de l'état de dégradation de leur roue de turbine. Les fissures liées au phénomène de fatigue présentent un mode de dégradation prédominant. La séquence de chargement appliquée aux roues de turbines est un paramètre qui influe sur la propagation de ces fissures. Le but de ce mémoire est de proposer un générateur de séquences synthétiques de chargement. Ces séquences simulées seront utilisées comme variables d'entrée dans un modèle d'estimation de durée vie résiduelle. Les séquences simulées doivent être statistiquement équivalentes à l'historique de chargement et prendre en compte la non-stationnarité des données. Dans un premier temps, nous décrivons le fonctionnement des groupes turbines-alternateurs d'Hydro-Québec et analysons les données disponibles. Ensuite, nous passons en revue les diverses méthodes de modélisation et les techniques de validation de modèles. Nous attachons une attention toute particulière à la description précise de la procédure de comparaison et de validation. Ce mémoire présente la comparaison minutieuse des performances des chaînes de Markov, de Semi-Markov et du Bootstrap à bloc mobile. Pour les deux premiers modèles, nous décrivons comment nous parvenons à prendre en compte la non-stationnarité. Finalement, nous montrons que les chaînes de Markov ne sont pas adaptées à notre problème, que les chaînes de Semi-Markov sont plus performantes lorsque la non-stationnarité est intégrée. Le choix final entre les chaînes de Semi-Markov et le Bootstrap à bloc mobile reste arbitraire. Toutefois, avec une vision à long terme, nous conseillons l'utilisation des chaînes de Semi-Markov pour leur flexibilité et leur adaptabilité.

**Mot-clés:** Modèle stochastique, Validation de modèle, Fiabilité, Chaîne de Semi-Markov, Chaîne de Markov, Bootstrap

# MODELING OF THE LOADING HISTORY OF HYDROELECTRIC GENERATING UNITS

Mickaël SZCZOTA

## ABSTRACT

Because of their ageing fleet, the utility managers are increasingly in needs of tools that can help them to plan efficiently maintenance operations. Hydro-Québec started a project that aim to foresee the degradation of their hydroelectric runner, and use that information to classify the generating unit. That classification will help to know which generating unit is more at risk to undergo a major failure. Cracks linked to the fatigue phenomenon are a predominant degradation mode and the loading sequences applied to the runner is a parameter impacting the crack growth. So, the aim of this memoir is to create a generator able to generate synthetic loading sequences that are statistically equivalent to the observed history. Those simulated sequences will be used as input in a life assessment model. At first, we describe how the generating units are operated by Hydro-Québec and analyse the available data, the analysis shows that the data are non-stationnary. Then, we review modelisation and validation methods. In the following chapter a particular attention is given to a precise description of the validation and comparison procedure. Then, we present the comparison of three kind of model : Discrete Time Markov Chains, Discrete Time Semi-Markov Chains and the Moving Block Bootstrap. For the first two models, we describe how to take account for the non-stationnarity. Finally, we show that the Markov Chain is not adapted for our case, and that the Semi-Markov chains are better when they include the non-stationnarity. The final choice between Semi-Markov Chains and the Moving Block Bootstrap depends of the user. But, with a long term vision we recommend the use of Semi-Markov chains for their flexibility.

**Keywords:** Stochastic models, Models validation, Reliability, Semi-Markov Chains, Markov Chains, Bootstrap

**TABLE DES MATIÈRES**

Page

## LISTE DES TABLEAUX

## LISTE DES FIGURES

## LISTE DES ABRÉVIATIONS, SIGLES ET ACRONYMES

| | |
|---|---|
| CMI | Commission Mixte Internationale |
| GTA | Groupe Turbine-Alternateur |
| HQ | Hydro-Québec |
| IREQ | Institut de recherche en électricité du Québec |
| ETS | École de technologie supérieure |
| FdR | Fonction de Répartition |
| SCADA | Supervisory Control And Data Acquisition |

***États et changements d'état des groupes turbines alternateur***

| | |
|---|---|
| A | Arrêt |
| BP | Baisse de puissance |
| F | Freinage |
| MN | Marche normale |
| MP | Montée en puissance |
| MR | Mise en rotation |
| MV | Marche à vide |
| PhA | Phase d'arrêt |
| PhD | Phase de démarrage |

| | |
|---|---|
| | ***Modèles*** |
| BBM | Bootstrap à bloc mobile |
| CMTD | Chaîne de Markov à temps discret |
| CSMTD | Chaîne de Semi-Markov à temps discret |
| DARMA | Discrete auto-regressive moving average |
| INAR | Integer auto-regressive |
| KNN | K-nearest neighbour |
| MTD | Mixture transition distribution |
| VLMC | Variable length Markov chain |
| | ***Validation*** |
| AIC | Akaike information criterion |
| BIC | Bayes information criterion |
| LKO | Leave-K-out |
| KS | Kolmogorov-Smirnov |
| MAE | Mean absolute error |
| MAPE | Mean absolute percentage error |
| VCB | Validation croisée par Bootstrap |
| VCVB | Validation croisée à V-blocs |

## LISTE DES SYMBOLES ET UNITÉS DE MESURE

| | |
|---|---|
| $\chi_i^2$ | Valeur du test statistique d'Anderson |
| $\varepsilon_t$ | Bruit associé aux modèles autorégressifs |
| $\varphi_i$, $\theta_i$ | Poids associés aux modèles autorégressifs |
| $D$ | Distance de Kolmogorov-Smirnov |
| $f_{ij}(k)$ | Fonction de répartition de Temps de maintien conditionnel |
| $h_i(k)$ | Fonction de répartition de temps de maintien |
| $k$ | Temps passé dans un régime de fonctionnement |
| $l$ | Taille de bloc pour le Bootstrap à bloc mobile |
| $m$ | Nombre d'états |
| $M$ | Nombre de simulations |
| $n$ | Nombre de blocs pour le Bootstrap à bloc mobile |
| $N_C$ | Nombre de changements d'état de type C observés |
| $N_{C_i}$ | Nombre de changements d'état de type C obtenus dans la $i^{me}$ simulation. |
| $N_{ij}$ | Nombre de changements de l'état $i$ vers l'état $j$ |
| $p_{ij}$ | Probabilité de transition de l'état $i$ vers l'état $j$ |
| $X_t$ | Processus |
| $S_t$ | Valeur prise par le processus $X_t$ au temps $t$ |
| $V$ | Nombre d'ensembles pour la validation croisée |

# INTRODUCTION

La dérèglementation des marchés et la production croissante d'électricité à partir de sources diversifiées poussent les producteurs comme Hydro-Québec (HQ) à optimiser l'exploitation de leurs équipements. Ceux-ci doivent de plus en plus être opérés à la limite ou même en dehors de leurs spécifications d'origine, augmentant ainsi les risques de défaillance. Par conséquent, les producteurs se trouvent devant l'obligation de développer et implanter des méthodes pour évaluer l'impact de ces nouveaux scénarios d'utilisation sur les fiabilités opérationnelles et résiduelles de leurs équipements.

La fatigue est une des principales causes de défaillance des turbines hydroélectriques. Il est donc essentiel de connaître son évolution temporelle en fonction des profils d'emploi (chargement, conditions d'opération, etc.) et de ses propriétés mécaniques. Malgré leur disponibilité, les séquences historiques d'utilisation des Groupes Turbines-Alternateur (GTA) sont peu documentées par Hydro-Québec. La caractérisation et la modélisation des scénarios réels d'opération sont deux étapes incontournables pour permettre une meilleure estimation de la fiabilité des groupes turbine-alternateur.

Évidemment, la présence d'incertitude est inhérente à l'interpolation et l'extrapolation des données. Ceci amène à la nécessité de développer des modèles probabilistes afin d'évaluer, dans un sens probabiliste, l'impact des différents profils d'utilisation. Le sujet de ce mémoire est la modélisation de l'historique de fonctionnement des GTA. Le but est de générer des séquences de chargement, qui serviront d'intrant à un modèle de fatigue afin de prédire l'état d'endommagement d'une roue de turbine. Et ce, afin de planifier de manière plus précise les arrêts pour inspections ou pour réfections. Ce mémoire s'intègre dans le projet PréDDIT (Prédiction de la dégradation et diagnostic intégré des turbines), réunissant plusieurs chercheurs de l'IREQ et des étudiants de divers domaines (fiabilité, métallurgie, fabrication, etc.) avec la volonté commune de caractériser la durée de vie résiduelle des GTA. Le projet PréDDIT dans sa globalité peut avoir, à terme, un impact majeur sur les performances d'Hydro-Québec. Les travaux de la présente maîtrise sont donc ancrés dans le milieu industriel et ont été possibles grâce à une collaboration étroite et soutenue avec l'IREQ [1].

Au sein du parc d'exploitation d'HQ, la centrale X est particulière [2]. Premièrement, ses GTA peuvent être synchronisés avec différents réseaux électriques. Deuxièmement, le débit turbiné par la centrale est imposé par la Commission mixte internationale (CMI) qui regroupe les États-Unis et le Canada. La CMI gère tout le système hydrique des Grands Lacs et du Saint-Laurent [3]. Ces caractéristiques font de cette centrale un élément majeur pour l'exporta-

1. Deux périodes de quatre mois ont été passées à temps complet dans les locaux de l'IREQ
2. Par soucis de confidentialité, les vrais noms de centrales ne seront pas indiqués dans ce mémoire.
3. Pour plus d'information sur la CMI, se référer à son site internet : www.ijc.org

tion d'électricité, et amènent à un facteur d'utilisation élevé de ces GTA. Le contexte particulier de la centrale X permet de comprendre pourquoi une bonne planification des opérations de maintenance est primordiale. En effet, avec un fort facteur d'utilisation, il existe peu de marge pour les arrêts pour inspection ou réparation. La centrale X est donc une candidate idéale pour développer des outils de prédiction de l'endommagement des GTA. De plus, même si l'étude présentée dans ce mémoire porte principalement sur celle-ci, nous testerons et validerons le modèle avec des données provenant d'autres centrales. En effet, il convient d'évaluer la capacité du modèle à être utilisé dans différents cas de figure.

La Figure 0.1 expose la structure du mémoire. Dans le Chapitre 1 nous abordons les travaux préliminaires nécessaires à ce type d'étude. La revue de littérature du Chapitre 2 nous permet de décrire les différentes manières de générer des séries temporelles (ou chronologiques), ainsi que les méthodes de comparaison et de validation de modèles. Le Chapitre 3 décrit la procédure de validation mise en place alors que nous présentons les modèles dans le Chapitre 4. Le Chapitre 5 est consacré aux résultats et à leur analyse. Quant au Chapitre 6, il contient les recommandations pour l'utilisation du modèle ainsi que les perspectives d'amélioration.

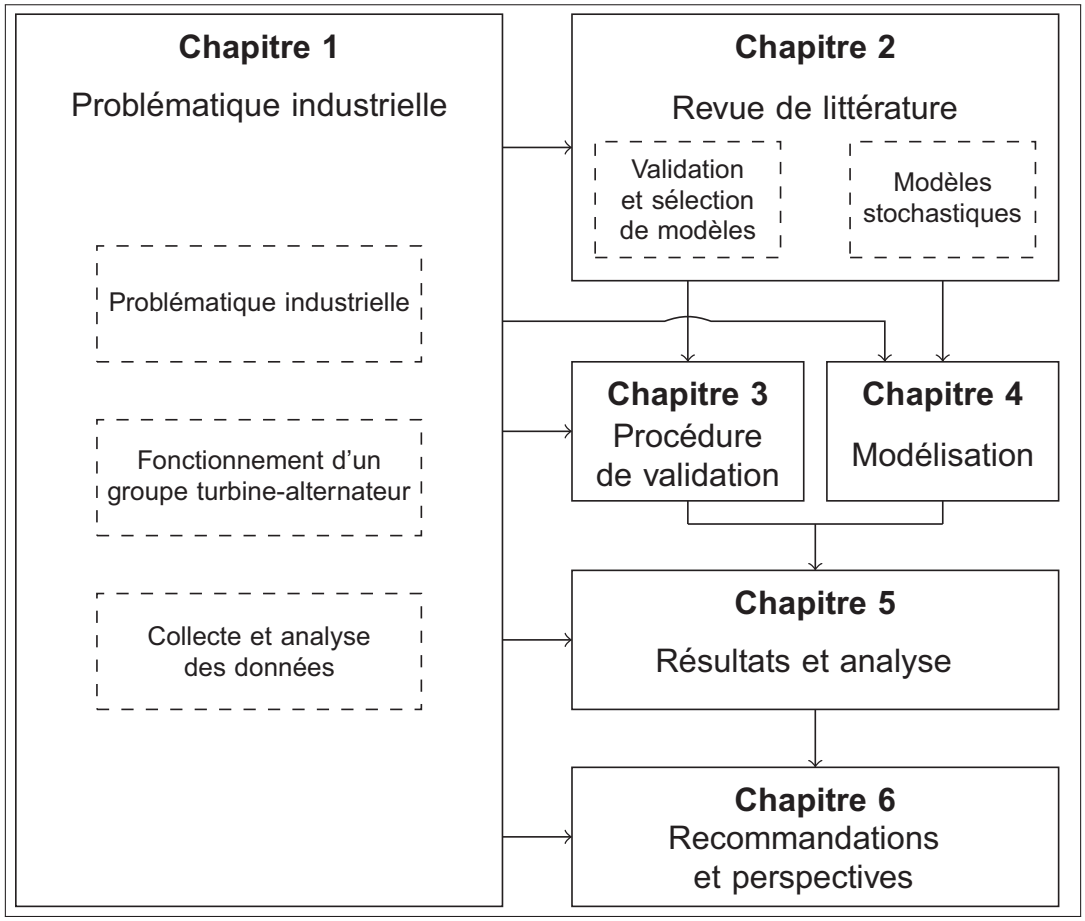


Figure 0.1 Structure du mémoire

# CHAPITRE 1

# PROBLÉMATIQUE INDUSTRIELLE

Dans ce chapitre, nous décrivons la problématique industrielle, ainsi que le fonctionnement des GTA. Nous effectuons également une analyse des données, préliminaire à tout travail de modélisation.

## 1.1 Problématique industrielle

Il est essentiel de bien comprendre l'intérêt et les objectifs de cette étude. Avec un parc de turbines hydroélectriques vieillissant, il devient de plus en plus important de développer des outils aidant à planifier avec plus de précisions les différents travaux de maintenance (inspection, réparation, réfection ou encore remplacement) de ces équipements. Actuellement, les arrêts des GTA pour inspection ou maintenance ne sont pas réalisés de façon prédictive, mais plutôt sur une base systématique : les GTA sont donc parfois arrêtés inutilement. La planification des arrêts basée sur la prédiction de l'état de dommage d'une roue de turbine permettrait d'engendrer des gains considérables. En programmant les arrêts pour maintenance en fonction du risque de subir une défaillance, nous évitons à la fois les arrêts inutiles des GTA à faible risque et nous limitons les chances de défaillance majeure en inspectant plus régulièrement les GTA à risque. Nous savons que les remplacements ou les réfections non planifiés causent des pertes supérieures à ceux qui sont inscrits dans le programme de maintenance.

Le développement de ces outils passe, entre autres, par une étude de la propagation des fissures. La fatigue est un élément qui influence significativement la fiabilité des turbines (Thibault *et al.*, 2011; Sabourin *et al.*, 2010). L'endommagement par la fatigue (la propagation des fissures) provient essentiellement de la fluctuation des contraintes et des déformations dans la turbine (ASTM E18). Ces fluctuations peuvent être très nombreuses, mais avec une faible amplitude (les vibrations que l'on retrouve dans toute machine tournante). Elles peuvent être moins fréquentes, mais avec de grandes amplitudes (les changements de régime de fonctionnement). On distingue clairement ces deux types de fluctuation dans la Figure 1.1, celle-ci présente les micro-déformations mesurées sur une turbine en fonctionnement. Selon la théorie de la mécanique de la rupture, ce sont les fluctuations de grande amplitude qui, dans un premier temps, ont le plus d'influence sur la propagation des fissures. Après avoir dépassé une taille critique de fissure (seuil), l'effet des vibrations (les fluctuations à faible amplitude) devient prépondérant (Gagnon *et al.*, 2010). Le but du modèle est de prédire l'évolution du nombre de fluctuations à grande amplitude que sont les changements de régime, et qui constituent le chargement de la roue de turbine. Nous modéliserons donc l'historique de chargement, qui est en fait la série temporelle du fonctionnement d'un GTA. La modélisation des vibrations n'est

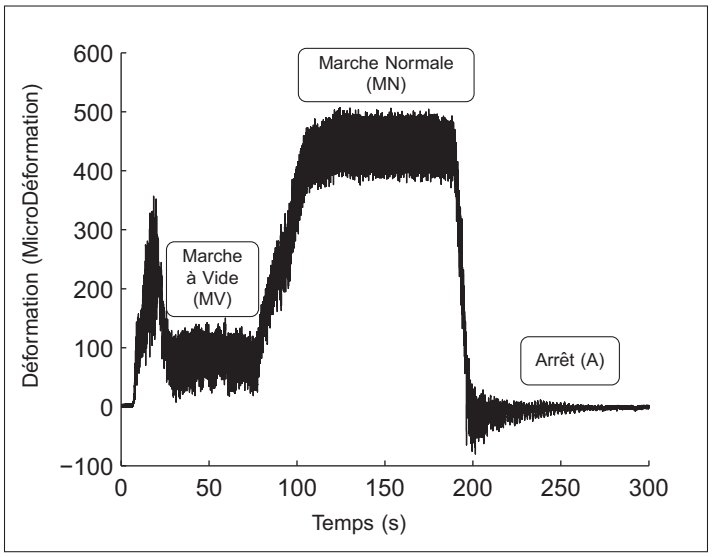


Figure 1.1 Micro-déformations mesurées sur une turbine en fonctionnement

pas envisagée dans ces travaux. Les simulations produites seront ensuite utilisées comme intrant dans un modèle de fatigue, dont l'objectif sera de classer les GTA en fonction du risque qu'ils subissent une défaillance majeure dans le futur.

## 1.2 Fonctionnement d'un groupe turbine-alternateur

Nous considérons donc les trois régimes de fonctionnement : Marche Normale (MN), Marche à Vide (MV) et Arrêt (A) (Figure 1.1). Les spécificités de ces états de fonctionnement sont résumées dans le Tableau 1.1, en marche normale, nous avons donc la turbine en rotation avec production d'électricité, en marche à vide, la turbine est toujours en rotation mais aucune électricité n'est produite, finalement en arrêt il n'y a aucune rotation de la turbine. Idéalement, un GTA devrait toujours produire de l'énergie à son point de rendement maximum. Les courbes Débit/Puissance (Figure 1.2a)) et Rendement/Puissance (Figure 1.2b)) indiquent ce point, qui se caractérise par une quasi-absence de tourbillon à la sortie de la roue. À titre d'exemple, le point de rendement maximum du groupe 19 de la centrale X se situe à une production de $37.9\,MW$, ce qui correspond à un débit turbiné d'environ $180\,m^3s^{-1}$. L'allure de ces courbes dépend de la conception de la roue de turbine et de la hauteur de chute. Dans un contexte de production, il est impossible de maintenir un GTA à ce point de fonctionnement. Le débit turbiné (donc la puissance produite) fluctue à l'intérieur d'une plage. La borne supérieure (120%) représente la capacité maximale de turbinage du GTA, la borne inférieure (80%) provient de la mécanique des fluides. En effet, si le débit turbiné se trouve entre 40% et 60% du débit nominal, le tourbillon en sortie de roue devient trop important à cause de la torche de charge partielle et provoque de fortes vibrations et de fortes variations de débit et de puissance (Cha-

pallaz, 1995). Hydro-Québec veut éviter l'apparition de ce phénomène, les GTA sont donc très rarement opérés en dessous de 80% du débit optimal.

Le débit turbiné associé à une marche à vide est le débit nécessaire pour faire tourner la turbine à la vitesse du réseau (vitesse synchrone) sans produire d'énergie. C'est donc un point de fonctionnement précis. Si l'on utilise une plage de valeur pour définir ce régime, c'est par souci de généralisation, car ce point n'est pas le même d'un GTA à l'autre, d'une conception à l'autre.

Tableau 1.1 Caractérisation des états de fonctionnement

| **Régime** | **Débit turbiné (% Débit Optimal)** | **Ouverture des directrices (%)** | **Vitesse de rotation** | **Production (% Production optimale)** |
|---|---|---|---|---|
| Marche Normale | [80 ; 120] | [80 : 100] | Vitesse synchrone | [80 ; 120] |
| Marche à Vide | ]0 ; 30] | ]0 ; 30] | Vitesse synchrone | 0 |
| Arrêt | 0 | 0 | 0 | 0 |

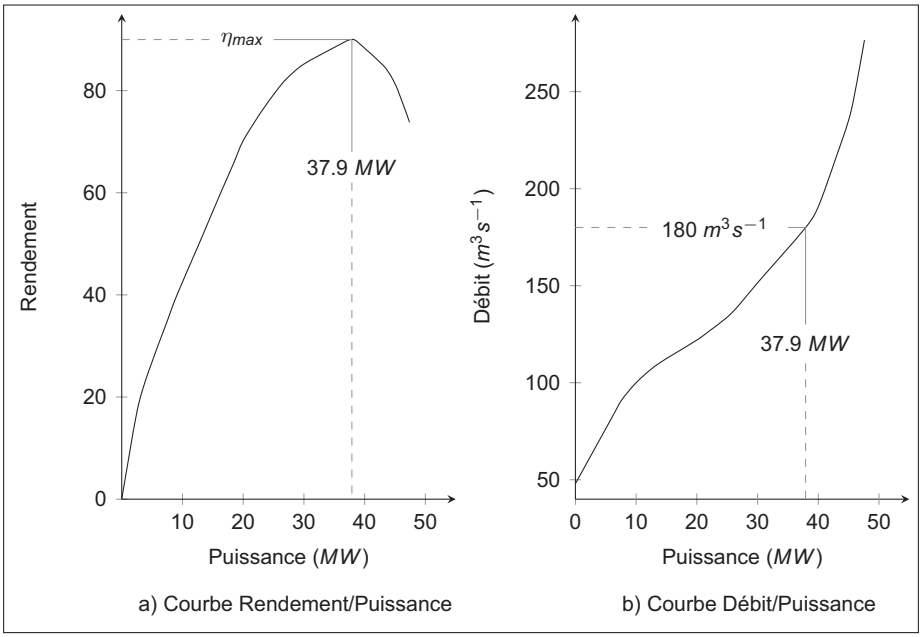


Figure 1.2 Courbes de fonctionnement du GTA 19 de la centrale X

Comme l'illustre la Figure 1.3, nous considérons que tous les régimes de fonctionnement sont communicants. Il existe donc six types de changements de régime que l'on appellera aussi *changements d'état* :

- Baisse de puissance (BP) : Passage de la marche normale vers la marche à vide.
- Montée en puissance (MP) : Passage de la marche à vide vers la marche normale.
- Phase d'arrêt (PhA) : Passage de la marche normale vers l'arrêt.
- Phase de démarrage (PhD) : Passage de l'arrêt vers la marche normale.
- Freinage (F) : Passage de la marche à vide vers l'arrêt.
- Mise en rotation (MR) : Passage de l'arrêt vers la marche à vide.

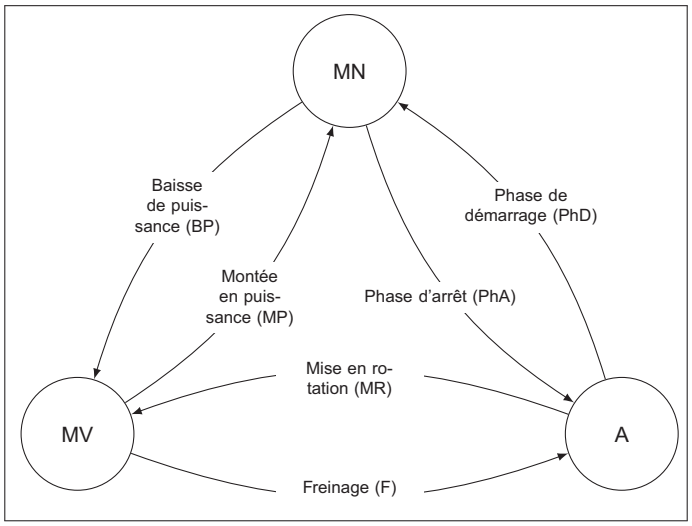


Figure 1.3 Diagramme d'état

Les changements d'état BP, MP, F et MR sont naturels, à mettre en opposition avec PhD et PhA qui sont des constructions. En effet, la procédure de démarrage (d'arrêt) des turbines veut que le GTA ait un passage en marche à vide d'une durée maximum de 5 minutes. Donc, une phase de démarrage est une mise en rotation directement suivie par une montée en puissance, alors qu'une phase d'arrêt est une baisse de puissance suivie directement d'un freinage. Même si le nombre de mises en rotation et de freinages est marginal, le but est de pouvoir différencier les montées en puissance dues aux changements de réseau de celles dues aux démarrages des GTA.

## 1.3 Collecte et analyse des données

Dans un premier temps, il nous a fallu déterminer sur quelles données se baserait notre étude. Celles-ci doivent permettre de suivre la succession des changements de régime d'un GTA pendant une période de temps. Comme le montre le Tableau 1.1, parmi les données typiques de fonctionnement d'un GTA, la vitesse de rotation est la même en MN et en MV. Et la production sera nulle en MV comme en A. On remarque par contre que l'ouverture des directrices et le débit turbiné permettent de discriminer les régimes. Or, lors de travaux de maintenance ou d'inspection, il arrive que les directrices soient maintenues ouvertes à 100%. Seul le débit turbiné donne avec certitude le régime dans lequel se trouve le GTA. Dans une optique de traitement automatique, il est plus judicieux d'utiliser cette donnée. Non seulement elle permet de déterminer avec certitude l'état des GTA, mais elle est obtenue par calcul et soumise à vérification. Il a fallu déterminer de quelle source proviendrait ces données. De nombreuses bases de données cohabitent au sein d'Hydro-Québec, nous avons donc consacré un temps important à leur comparaison. Ceci était primordial afin de travailler avec la source qui contient les données les plus fiables possibles, et ayant subi un minimum de transformations.

### 1.3.1 Données utilisées

Les données sont issues du système SCADA [1] d'Hydro-Québec. Cette base de données nous donne accès à 7 ans d'historique de débit turbiné, avec un échantillonnage aux 5 minutes. La Figure 1.4 est un exemple des données telles que nous les obtenons. Il s'agit de trois mois d'historique de fonctionnement où nous pouvons clairement distinguer les trois régimes susmentionnés. La Figure 1.5 permet de comprendre l'information contenue dans ces données. En effet, en (a) on trouve le signal réel qui est continu, les mesures prises aux 5 minutes conduisent à une discrétisation temporelle (Figure 1.5(b)). Nous faisons subir une discrétisation des valeurs de la série temporelle afin d'associer une seule valeur à chaque régime de fonctionnement(Figure 1.5(c)). Ceci nous amène à travailler une série temporelle catégorique à temps discret. L'échantillonnage de l'historique peut être responsable de perte d'information, simplement car les principales manœuvres (arrêt/départ et changement de réseau) nécessitent environ 5 minutes. Il existe donc une faible probabilité que ces manœuvres ne soient pas saisies. Dans le but de la diminuer, nous considérons le GTA en MV pour toutes valeurs de débit comprises dans l'intervalle $]0;140]$. La valeur limite $140\,m^3s^{-1}$ est choisie, car elle correspond à environ 80% du débit nominal (valeur en dessous de laquelle le phénomène de torche apparaît). Selon la politique d'HQ, un groupe n'est jamais en marche normale avec un débit inférieur à cette valeur.

1. Supervisory Control And Data Acquisition, c'est un système utilisé par la plupart des producteurs d'électricité, qui contient toutes les informations nécessaires à la gestion d'un parc de production

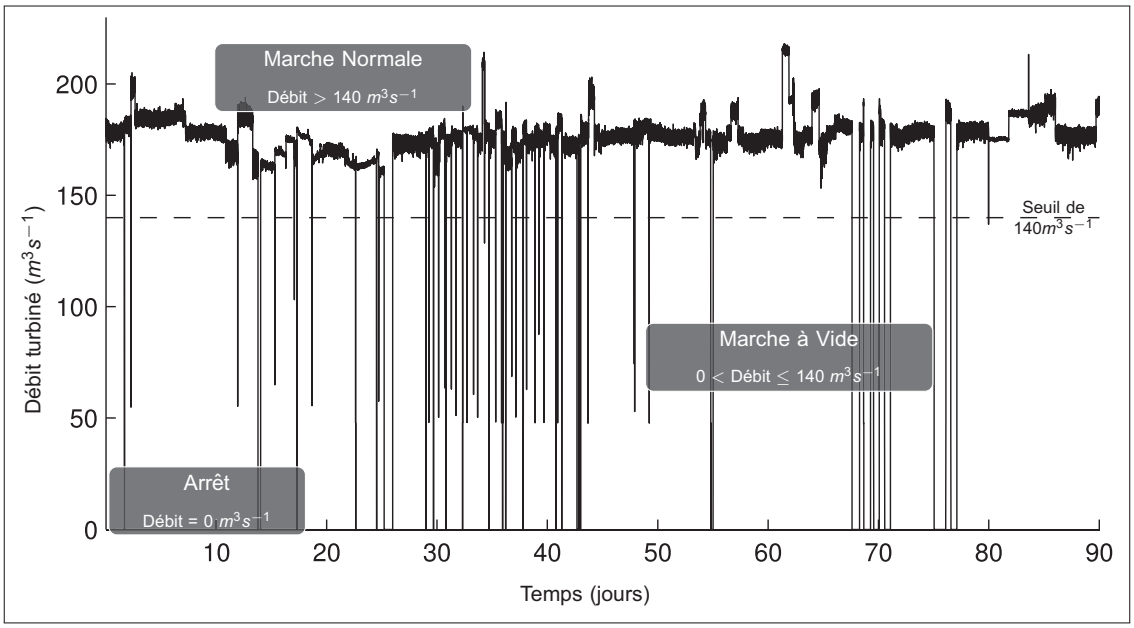


Figure 1.4 Signal brut

### 1.3.2 Analyse des données

Avant de procéder à des travaux de modélisation sur les séries temporelles, nous les analyserons. Celle-ci a pour objectif de déterminer si la série chronologique est stationnaire et si elle présente une tendance ou une périodicité [2]. L'approche de modélisation peut changer radicalement en fonction des caractéristiques de la série. En effet, dans le cas non stationnaire, les paramètres d'un modèle doivent varier avec le temps. À noter que les données d'opération du groupe 19 de la centrale X sont utilisées dans cette section afin d'illustrer l'analyse des données. Il sera fait référence à ce GTA avec le nom X19.

#### 1.3.2.1 Stationnarité

En premier lieu, nous évaluons si la série temporelle est stationnaire, c'est-à-dire si ses caractéristiques statistiques restent identiques dans le temps. Nous présentons ici plusieurs façons de déterminer si la série est stationnaire. La Figure 1.6, qui représente la durée de chaque passage du GTA dans le régime MN, montre clairement que le comportement est différent d'une année à l'autre. Si l'on regarde maintenant le taux d'évolution du nombre de BP et de PhA par jour pour chacune des six années (Figure 1.7 et Figure 1.8 respectivement) on remarque une

2. Le terme périodicité est ici préféré même si le terme saisonnalité est plus utilisé (dans le cas de séries hydrologiques par exemple). La saisonnalité est simplement une périodicité particulière.

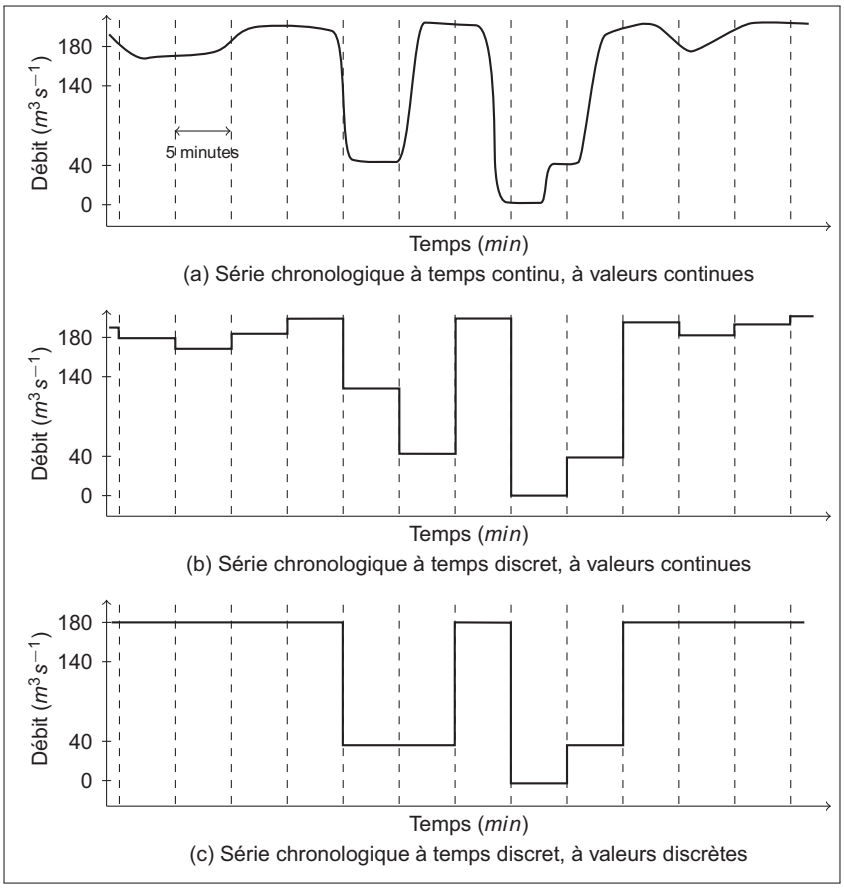


Figure 1.5 Origine et transformation des données utilisées

grande différence [3]. Ces éléments sont des signes de la non-stationnarité de la série temporelle. Afin de valider le caractère non stationnaire du processus, nous utilisons le test de stationnarité d'Anderson et Goodman (1957). Ce test se base sur les matrices de transition associées aux chaînes de Markov (nous y reviendrons plus en détail dans la Section 4.3). Il permet de tester l'hypothèse nulle $H_0$ que la série temporelle est stationnaire par rapport à l'hypothèse alternative $H_A$ qu'elle ne l'est pas. Il se déroule de la façon suivante :

1) Séparer les données en $T$ échantillons de taille égale. Dans notre cas $T = 6$.

2) Calculer les matrices de transition pour chacun des échantillons, constituées des $\hat{p}_{ij}(t)$ avec $t = 1 \cdots 6$ et $i, j = 1 \cdots k$ . Dans notre cas $k = 3$.

3) Calculer la matrice de transition globale, constituée des $\hat{p}_{ij}$ avec $i, j = 1 \cdots 3$

3. Par souci de confidentialité, les valeurs réelles des abscisses ne sont pas indiquées sur ces figures. On compare les taux d'évolution des PhA avec ceux des BP, en les présentant avec la même échelle. Cette échelle est représentée par les valeurs $BP_{A_{inf}}$ et $BP_{A_{sup}}$, qui correspondent au nombre total (maximum et minimum respectivement) de BP par an.

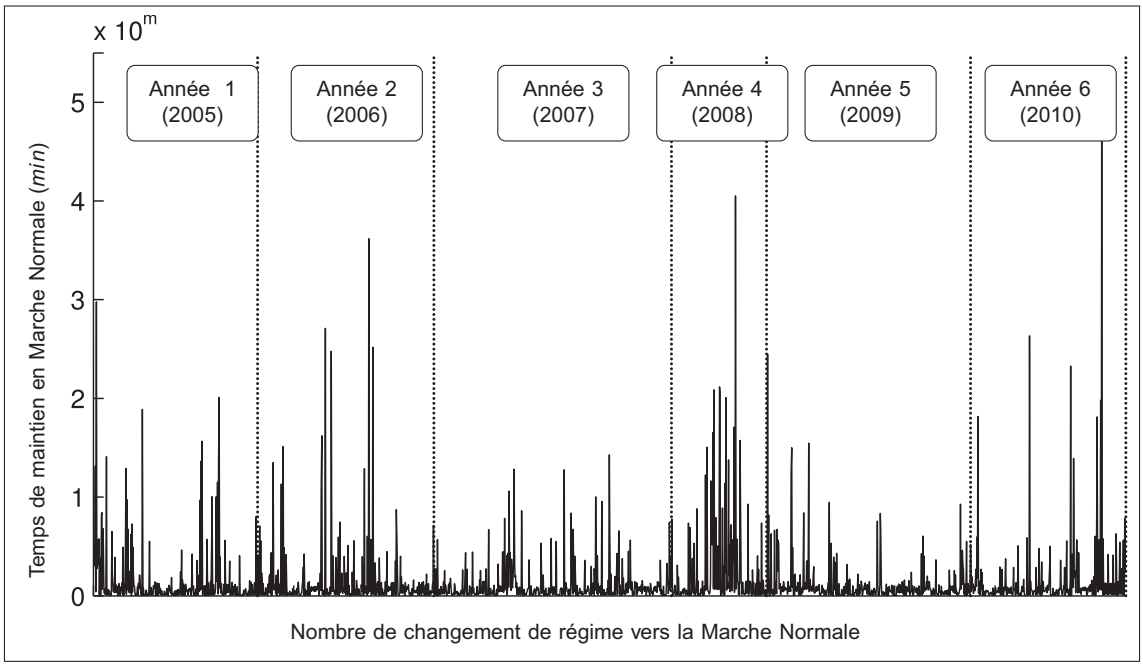


Figure 1.6 Différences entre années des temps de maintien en marche normale

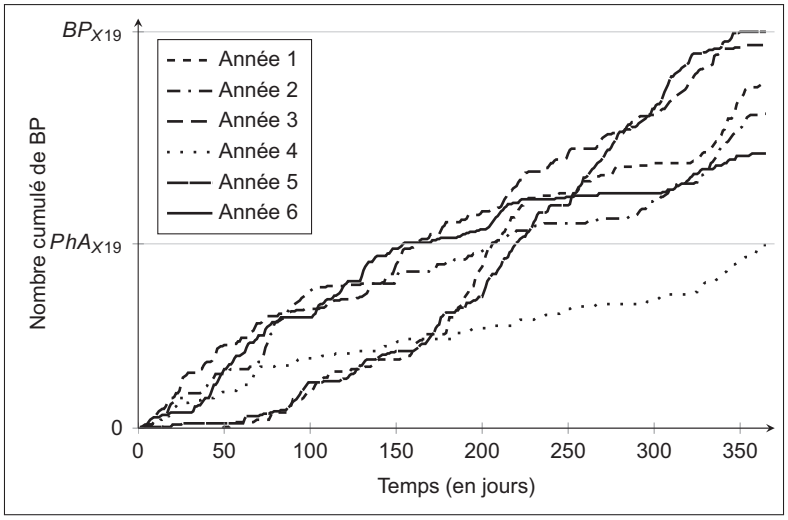


Figure 1.7 Taux d'évolution du nombre de BP par jour (X19)

4) Calculer les statistiques $\chi_i^2$ pour chaque état avec l'équation 1.1

$$\chi_i^2 = \sum_{j=1}^{3} \sum_{t=1}^{T} \frac{n_i(t)[\hat{p}_{ij}(t) - \hat{p}_{ij}]}{\hat{p}_{ij}} \tag{1.1}$$

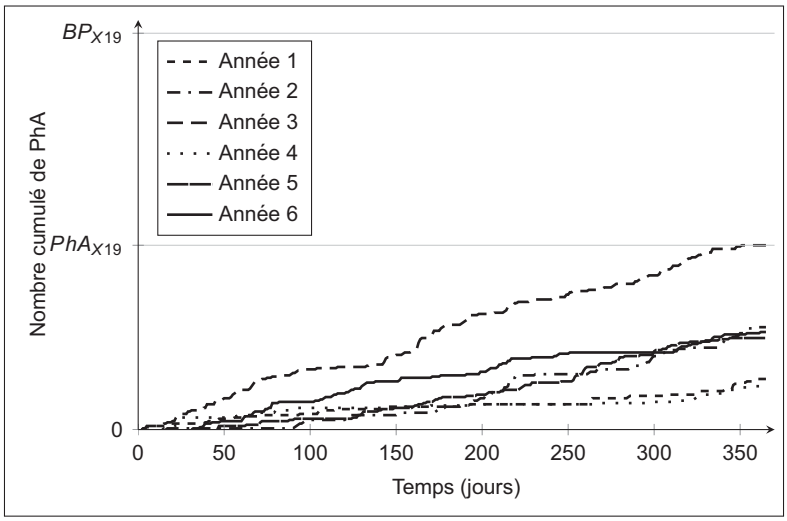


Figure 1.8 Taux d'évolution du nombre de PhA par jour (X19)

Les valeurs obtenues dans notre test sont les suivantes :

- $\chi_1^2 = 208.6$
- $\chi_2^2 = 109.3$
- $\chi_3^2 = 237.1$

5) Comparer avec la valeur limite de la distribution du $\chi^2$ avec $(k-1)(T-1) = 10$ degrés de liberté. Soit $\chi^2 = 18.31$ . Si $\chi_i^2 > \chi^2$, alors $H_0$ est rejeté.

Il est clair que d'une année à l'autre le fonctionnement d'un GTA ne peut être considéré comme stationnaire.

#### 1.3.2.2 Tendance et périodicité

Tendance et périodicité sont des cas déterministes de non-stationnarité, en effet ils peuvent être pris en compte dans les modèles de façon paramétrique. Il est donc important de savoir si les variations des caractéristiques statistiques d'une série temporelle sont périodiques ou suivent une pente particulière. En produisant les courbes de l'évolution du nombre de changements d'état par mois (Figure 1.9 [4]), on peut détecter visuellement la présence de tendances. L'allure des courbes ne semble suivre aucune augmentation (ou diminution) constante du nombre de changements d'état. Par contre, une périodicité est envisageable. De la même façon que pour les Figures 1.7 et 1.8, l'échelle des Figures 1.9a) à b) est représentée par $Ref_m$ qui correspond ici au nombre maximum de BP par mois. Nous déterminons la présence d'une périodicité dans le

4. À noter que les courbes concernant les freinages et mises en rotation ne sont pas présentées, car elles n'apportent que très peu d'informations

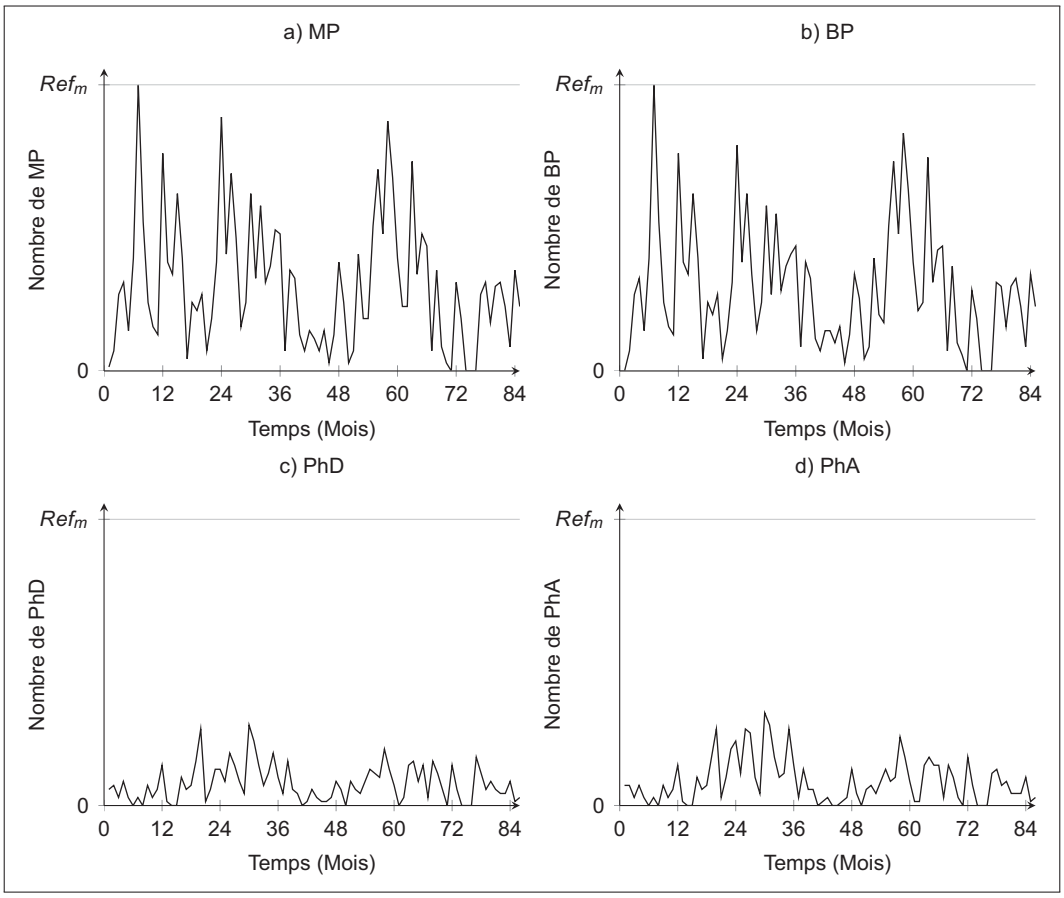


Figure 1.9 Séries temporelles du nombre de changements d'état par mois (X19)

processus à l'aide d'une analyse spectrale. Nous produisons dans la Figure 1.10 le spectre de puissance de la série temporelle du nombre de changements d'état par mois à laquelle nous avons retranché la moyenne [5]. Le spectre présente une fréquence prépondérante, ce qui pourrait signifier que la série temporelle contient une périodicité d'environ 28 mois. La quantité de données n'est pas suffisante pour s'assurer que la présence d'une telle périodicité n'est pas uniquement propre aux données disponibles, mais bien caractéristique du fonctionnement du GTA. L'utilisation de cette périodicité risque de nous induire en erreur, elle devra être confirmée (ou infirmée) dans le futur. Pour la suite du travail, nous choisissons d'ignorer la présence de périodicité et de considérer les changements de comportement comme entièrement aléatoire. L'analyse nous a indiqué que la série est non stationnaire et qu'aucune caractéristique ne permet de prédire avec précision l'évolution du nombre de changements d'état. Cette analyse n'est valide que pour le GTA 19 de la centrale X. Nous devrons réaliser une nouvelle analyse chaque fois que nous voudrons modéliser un autre GTA.

5. Nous ne montrons que l'analyse spectrale des BP et PhA car le nombre de MP (respectivement de PhD) a la même évolution que le nombre de BP (respectivement de PhA).

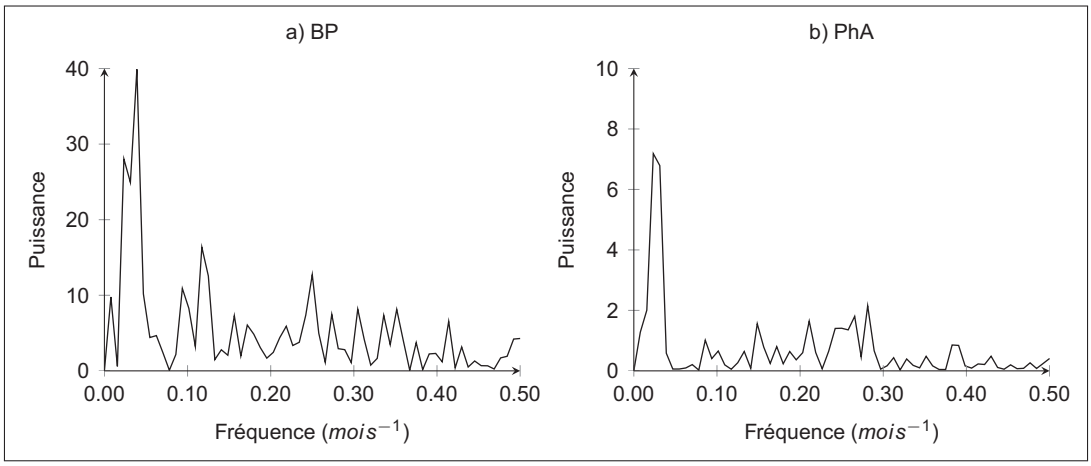


Figure 1.10 Spectre de fréquence du nombre de changements d'états par mois (X19)

## 1.4 Conclusion du chapitre 1

Le choix des données, ainsi que de la base de données, ont constitué la première étape de cette maîtrise et ont conduit à la rédaction d'un rapport interne à Hydro-Québec. Cette étape a permis de nous familiariser à la fois au fonctionnement d'Hydro-Québec et au domaine de l'hydroélectricité. Deux points sont à retenir de ce chapitre :

- Pour les besoins de l'étude, nous avons ramené le fonctionnement des GTA à un processus alternant entre trois états.
- Le processus n'est pas stationnaire, mais ne présente aucune tendance ou périodicité. Ceci nous amène à traiter cette non-stationnarité de manière entièrement stochastique.

L'absence de saisonnalité (périodicité de 12 mois) dans le fonctionnement du GTA 19 est surprenante. Même s'il est issu d'une centrale au fil de l'eau, il exploite une ressource dont les variations sont saisonnières. Nous pouvons faire deux hypothèses. La première est que la périodicité ne peut se voir qu'à l'échelle de la centrale. En effet, d'une année à l'autre le nombre de changements d'état global à la centrale peut se répéter, mais les conditions particulière à une année peuvent conduire les gestionnaires à choisir d'arrêter ou de faire subir des changements de réseaux à différents GTA. Donc, individuellement, les GTA ne présentent aucune saisonnalité. La seconde hypothèse est que la saisonnalité se retrouve dans les variations du débit turbiné lorsqu'un GTA se trouve en marche normale.

# CHAPITRE 2

# REVUE DE LITTÉRATURE

*Tous les modèles sont faux, mais certains sont utiles.*
George E.P. Box

La revue de littérature que nous présentons dans ce chapitre est constituée de deux parties. La première concerne la modélisation de l'historique de fonctionnement, la seconde cible les différentes techniques et procédures à suivre afin de comparer et valider des modèles. Ces deux parties sont évidemment complémentaires, car ce genre de travail est complet uniquement s'il est accompagné d'une procédure de validation. Dans la suite, nous nous concentrerons sur l'approche stochastique, car celle déterministe semble, a priori, peu adaptée à notre problème.

## 2.1 Modèles stochastiques

La littérature qui traite des modèles stochastiques est très riche et concerne toutes sortes de domaines (économique, industriel, climatologique, sportif...). Le but n'est pas d'en faire une revue exhaustive. Nous focaliserons sur la littérature décrivant les approches capables de modéliser des séries chronologiques catégoriques à temps discret.

Nous regroupons les modèles en quatre catégories : ceux de la famille des chaînes Markov, les autres modèles autorégressifs discrets, les modèles de régression et les modèles de rééchantillonnage. Celles-ci ont été définies à partir de la littérature consacrée aux séries temporelles catégoriques et à celle consacrée aux modèles générateurs de séries temporelles. Srikanthan et McMahon (2001) décrivent quatre familles de modèles capables de générer des séries temporelles de chutes de pluie : le modèle à deux parties, les modèles à probabilité de transitions, les modèles de rééchantillonnage et les modèles de type ARMA. Le modèle à deux parties est constitué comme suit : une partie qui détermine si les jours futurs seront pluvieux ou secs, et une autre qui détermine la quantité de pluie dans le cas d'un jour pluvieux. Les modèles à probabilité de transitions sont les modèles de la famille de Markov. Varin et Vidoni (2006) considèrent les chaînes de Markov (et ses variations), les modèles autorégressifs discrets et les modèles de régression. Ces catégories sont également présentes dans l'ouvrage de MacDonald et Zucchini (1997). La Figure 2.1 récapitule les modèles et leurs catégories.

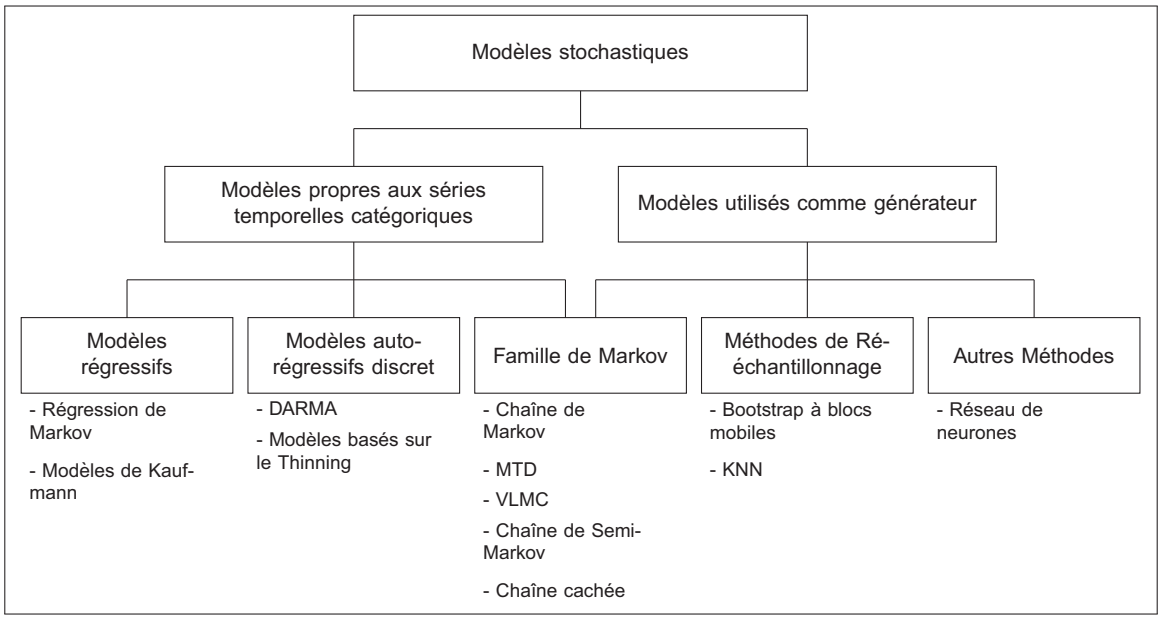


Figure 2.1 Les modèles et leur classification

### 2.1.1 Famille de Markov

Historiquement, les chaînes de Markov ont été largement utilisées pour modéliser les séries chronologiques catégoriques. En effet, leur fonctionnement basé sur les probabilités de transitions d'un état à un autre, fait un excellent parallèle avec ce type de série temporelle. Les exemples d'application des chaînes de Markov en tant que générateur de séries temporelles sont nombreux. Que ce soit des séries de successions des jours pluvieux/jours secs dans Dubrovský (1997), d'évolution de la vitesse du vent (Shamshad *et al.*, 2005), et même de chargement dans le but d'étude de fatigue (Kam, 1992). Les hypothèses liées aux chaînes de Markov limitent tout de même leurs champs d'applications.

Par abus de langage, ce que l'on appelle une chaîne de Markov est en fait une chaîne de Markov d'ordre 1. C'est-à-dire que la probabilité qu'un processus $X_t$, pouvant prendre $m$ valeurs ($S = 1...m$) change vers un état futur $S_{t+1}$, dépend uniquement de l'état présent $S_t$ et non des $i$ états passés (Équation 2.1).

$$P(X_{t+1} = S_{t+1}|X_1 = S_1, ..., X_{t-1} = S_{t-1}, X_t = S_t) = P(X_{t+1} = S_{t+1}|X_t = S_t) \tag{2.1}$$

Ceci est la propriété de Markov. Celle-ci, très contraignante, peut être une bonne approximation dans certains cas, mais dans d'autres elle ne permettra pas au modèle de reproduire toutes les caractéristiques des données. Afin de mieux respecter la structure d'une série, il est possible de travailler avec des chaînes de Markov d'ordre $q$, où les probabilités de transition vers

$S_{t+1}$ dépendent des $q$ états passés (Équation 2.2).

$$P(X_{t+1} = S_{t+1} | X_1 = S_1, ..., X_{t-1} = S_{t-1}, X_t = S_t) = P(X_{t+1} = S_{t+1} | X_{t-q+1} = S_{t-q+1}, ..., X_t = S_t) \quad (2.2)$$

De cette façon l'éventail d'application s'élargit, mais génère un problème de taille : l'augmentation rapide du nombre de paramètres (Kaufmann, 1987). En effet, le nombre de paramètres d'une chaîne de Markov d'ordre $q$ à $m$ états est de $(m-1)m^q$. Ceci conduit souvent à préférer la chaîne d'ordre 1. C'est dans l'optique de limiter le nombre des paramètres pour les chaînes de Markov à ordre élevé que Raftery (1985) propose le *Mixture Transition Distribution* modèle (MTD). Ce modèle se base sur l'équation 2.3. Il considère que la probabilité de changement vers $S_{t+1}$ s'obtient par une combinaison linéaire des probabilités de changements entre $S_{t+1}$ et les $q$ états passés.

$$P(X_{t+1} = S_{t+1} | X_1 = S_1, ..., X_{t-1} = S_{t-1}, X_t = S_t) = \sum_{i=1}^{q} \lambda_i P(X_{t+1} = S_{t+1} | X_{t-i+1} = S_{t-i+1}) \quad (2.3)$$

L'estimation des paramètres des MTD est plus complexe et moins directe que pour les chaînes de Markov, et se rapporte à un problème d'optimisation. Berchtold et Raftery (2002) proposent une revue des techniques d'estimation, ainsi que des exemples d'application des MTD en tant que générateur de séries temporelles.

Les chaînes de Markov à longueur variable, ou VLMC de l'anglais *Variable Length Markov Chain*, permettent également de réduire le nombre de paramètres des chaînes de Markov à ordre élevé. Les VLMC telles que décrites par Bühlmann et Kunsch (1999) sont proches des modèles liés aux théories de l'information. L'augmentation du nombre de paramètres d'une chaîne de Markov d'ordre $q$ est due à la création d'états supplémentaires. Ces états sont en fait tous les chemins de longueur $q$ que peut emprunter le processus pour arriver à l'état présent. Parmi eux, certains sont inutiles et peuvent donc être combinés. L'idée des VLMC est de formaliser la procédure qui permet cette combinaison. Ferrari et Wyner (2003) fournissent une explication claire de la construction et de l'utilisation des VLMC. Un algorithme (l'algorithme de contexte) permet de diminuer le nombre d'états. La première étape est de supprimer les états/chemins qui apparaissent moins de deux fois dans l'historique. La seconde est de combiner les chemins qui ont peu d'influence sur les probabilités de transitions. Nous illustrons le principe des VLMC avec la Figure 2.2. De la même façon que pour les MTD, les VLMC visent à réduire la complexité du modèle. Cependant, ceci se fait au prix de l'augmentation de la complexité de la procédure d'estimation des paramètres.

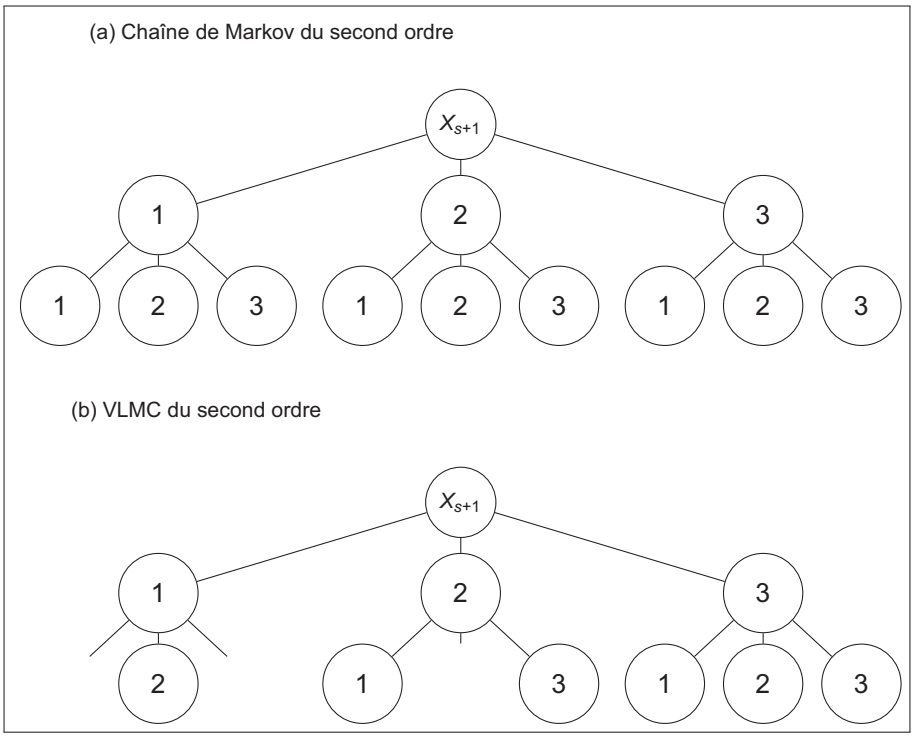


Figure 2.2 Principe des VLMC

Les chaînes de Semi-Markov permettent aussi de s'affranchir des hypothèses contraignantes des chaînes de Markov. Ce type de modèle traite séparément les temps de maintien dans chaque état et les changements d'état et permet ainsi d'utiliser des distributions de temps de maintien plus proches de la réalité des données. À l'opposé, la propriété de Markov (processus sans mémoire) oblige les temps de maintien à avoir une distribution géométrique (exponentielle en temps continu). Tel qu'expliqué par Barbu et Limnios (2008), l'état présent $X_n$ uniquement permet de déterminer à la fois le temps $k$ durant lequel le processus va rester dans cet état, et quel sera l'état suivant $X_{n+1}$ (Équation 2.4).

$$P(X_{n+1} = S_{n+1} | X_1 = S_1, ..., X_{n-1} = S_{n-1}, X_n = S_n) \\ = P(X_{n+1} = S_{n+1}, T_n = k | X_n = S_n) \quad (2.4)$$

Nous remarquons dans cette équation que nous avons recours à des indices du nombre de changements d'état en place des indices de temps. Malgré cette flexibilité, la littérature consacrée aux séries temporelles catégoriques accorde peu de considération aux chaînes de Semi-Markov. Toutefois, elles sont utilisées comme générateur de séries temporelles (Fowler *et al.*, 2000; Montopoli *et al.*, 2008; Kharoufeh *et al.*, 2010).

Nous signalons également l'existence des chaînes de Markov et Semi-Markov cachées. Ces modèles sont très employés dans le domaine de la reconnaissance vocale ou de caractères (Rabiner, 1989; Yu, 2010). L'intérêt de leur application apparaît surtout lorsque le processus de génération des données n'est pas connu, et entraîne une incertitude quant aux nombres d'états, d'où l'utilisation du terme *caché*. Visser (2011) propose une excellente vulgarisation des chaînes de Markov cachées. Malgré toutes les qualités de ces modèles, nous comprenons qu'ils ne sont pas vraiment adaptés à notre problème, car ils apporterant une complexité qui n'est pas nécessaire. En effet, le processus sous-jacent de nos données est très bien connu et est décrit dans le Chapitre 1.

### 2.1.2 Modèles autorégressifs discrets

Les modèles autorégressifs existent depuis longtemps, mais c'est l'approche proposée par Box *et al.* (1970), avec les modèles *ARMA*, qui les a popularisés. L'équation 2.5 caractérise un modèle *ARMA(p,q)*, composé d'une part autorégressive (AR) et d'une part moyenne mobile (MA).

$$X_t = \varepsilon_t + \sum_{i=1}^{p} \varphi_i X_{t-i} + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} \tag{2.5}$$

La partie autorégressive définit la dépendance temporelle, c'est-à-dire l'influence des $p$ valeurs passées sur les valeurs futures. La partie moyenne mobile est une combinaison linéaire des $q$ bruits blancs associés à chaque série temporelle. Les modèles *ARMA* ne peuvent toutefois pas être utilisés avec des séries temporelles catégoriques (Mckenzie, 2003). Deux approches ont été proposées dans le passé : les modèles *DARMA* (pour Discrete Auto Regressive Moving Average) par Jacobs et Lewis (1983) d'un côté et de l'autre les modèles basés sur le *thinning* tel que le modèle *INAR(1)* de McKenzie (1985).

Au dire de Weiß (2011), les *DARMA* sont complexes et difficilement interprétables, il recommande les *NDARMA* (une version simplifiée des *DARMA*) qui sont décrits dans le même article (Jacobs et Lewis, 1983). Même s'ils ont le mérite d'avoir lancé de nombreux travaux sur le sujet des séries temporelles catégoriques, les modèles *DARMA* et *NDARMA* sont très peu utilisés aujourd'hui. L'article de Chung et Salas (2000) est l'une des seules applications récentes. Le modèle *INAR(1)* (pour Integer Auto Regressive) est le même que *AR(1)* décrit par l'équation 2.6, excepté que dans le cas d'une série temporelle catégorique on ne peut assurer l'indépendance entre $\varepsilon_t$ et $X_{t-1}$. D'où l'utilisation de l'opération de *thinning* à la place de la multiplication scalaire.

$$X_t = \varepsilon_t + \varphi X_{t-1} \tag{2.6}$$

L'équation 2.6 devient alors 2.7.

$$X_t = \varepsilon_t + \varphi * X_{t-1} \tag{2.7}$$

Avec $\varphi * X_{t-1} = \sum_{i=1}^{X_{t-1}} B_i(\alpha)$, où $B_i(\alpha)$ est une séquence de valeurs aléatoires binaires indépendantes et identiquement distribuées, avec $P(B_i(\alpha) = 1) = \alpha$. Ce type de modèle est très présent dans la littérature consacrée aux séries temporelles catégoriques (MacDonald et Zucchini, 1997; Mckenzie, 2003). Cependant, ils souffrent du même problème que les *DARMA*, c'est à dire du manque d'exemple d'application pratique (MacDonald et Zucchini, 1997; Jung et a.R. Tremayne, 2006).

### 2.1.3 Modèles de régression

Les modèles présentés ici sont inspirés des modèles linéaires généralisés. Leur avantage est qu'ils sont, par construction, capables de prendre en compte la non-stationnarité des séries temporelles. Parmi eux, on retrouve le modèle de régression de Markov décrit par Zeger et Qaqish (1988), qui est plutôt adapté aux séries temporelles de comptages. La valeur future provient d'une distribution dont les paramètres varient en fonction des valeurs passées, d'une tendance et d'une saisonnalité bâtie avec des fonctions périodiques du type sinus et cosinus. Sur le même principe, le modèle de Kaufmann (1987), où la probabilité $\pi_t$ d'être dans un des $p$ états au temps $t$ dépend d'une fonction $h$, est plus adapté à notre situation (voir équation 2.8).

$$\pi_t = h(\beta_0 + \beta_1 y_{t-1} + ...\beta_q y_{t-q} + \alpha_1 x_{t1} + ... + \alpha_k x_{tk}) \tag{2.8}$$

$\pi_t$ ne dépend plus uniquement des $q$ valeurs passées comme pour une chaîne de Markov d'ordre $q$, mais également de $k$ variables externes. Celles-ci peuvent être liées à $t$ si l'on souhaite introduire tendance et/ou saisonnalité. Fokianos et Kedem (2003) proposent des exemples d'applications d'un modèle très proche de celui de Kaufmann, mais avec une méthode d'estimation différentes pour les $\beta_i$.

### 2.1.4 Modèles de rééchantillonnage (Bootstrap)

La littérature des séries temporelles catégoriques n'inclut pas les modèles de rééchantillonnage. Pourtant, ils peuvent constituer une alternative intéressante, car ils sont très simples à mettre en place. Parmi eux, le Bootstrap à bloc mobile (BBM) et le Bootstrap des K plus proches voisins (KNN pour *K nearest neighbor*) ont été utilisés avec succès.

Le BBM de Kunsch (1989) est inspiré du Bootstrap d'Efron et Tibshirani (1993) qui consiste au rééchantillonnage, élément par élément, d'un ensemble de données. Mais si les éléments sont dépendants (une dépendance temporelle, par exemple) le Bootstrap ne peut plus être appliqué. Il faut alors rééchantillonner des blocs, c'est ce que fait le BBM. Vogel et Shallcross (1996) ont montré l'efficacité du BBM par rapport à un modèle ARMA. Lahiri (1999) compare le BBM avec trois autres variations : le Bootstrap à bloc non chevauchant (BBNC) de Carlstein (1986), le Bootstrap à bloc circulaire (BBC) de Politis et Romano (1992) et le Bootstrap stationnaire (BS)

proposé par Politis et Romano (1994). Le BBNC est identique au BBM à ceci prêt que les blocs ne peuvent se chevaucher, cette différence fait que le BBNC est moins efficace que le BBM. Le BBC est peu utilisé, mais présente les mêmes performances que le BBM. Quant au BS, il permet de s'assurer que les séries rééchantillonnées restent stationnaires, en rendant aléatoire la taille de blocs. Lall et Sharma (1996) ont montré l'intérêt de se servir du KNN pour générer des séries chronologiques. Le but est donc de prédire la valeur de $x_{T+1}$ d'une série temporelle $x_t$ avec $t = 1...T$, le principe est le suivant :

- Un décalage $d$ est défini, celui ci défini un vecteur de référence $V_r = [x_T; x_{T-1}, ...x_{T-d}]$.
- Les $K$ voisins les plus proches de $V_r$ contenu dans $x_t$ sont sélectionnés à l'aide d'une mesure de distance.
- La valeur $x_{T+1}$ est déterminée

Pour la dernière étape, la façon dont la valeur $x_{T+1}$ est obtenue peut varier. En effet, dans le cas d'une série temporelle catégorique, celle-ci est pigée avec replacement parmi une des $K$ valeurs qui suivent les $K$ vecteurs voisins. Alors que dans le cas d'une série continue, on l'a définie comme étant la moyenne des $K$ valeurs (Toth *et al.*, 2000).

### 2.1.5 Autres modèles

Les réseaux de neurones sont traités à part, car, par leur versatilité, ils peuvent être utilisés comme modèles autorégressifs ou modèles de régression. Leur force est de pouvoir trouver des relations complexes entre différentes variables. Ils sont très appliqués comme outils de classification (Khan *et al.*, 2001) et reconnaissance de forme (Bishop, 1995), mais aussi comme outils de prédiction. La plupart des réseaux de neurones sont proches des modèles de régression. C'est-à-dire qu'ils utilisent comme entrées, l'historique de la variable à prédire ainsi que des variables externes (Park *et al.*, 1991; Siqueira *et al.*, 2010). Nous trouvons tout de même quelques exemples comme modèle autorégressif (Toth *et al.*, 2000; Ture et Kurt, 2006). Nous ne rentrerons pas plus dans le détail de ces modèles, car de la même façon que les chaînes de Markov (ou Semi-Markov) cachées, ils deviennent intéressants si l'on ne connaît pas le processus qui génère les données. De plus, le côté « boîte noire » peut être problématique dans la mesure où l'on peut difficilement analyser la manière dont les prévisions sont faites.

### 2.1.6 Prise en compte de la non-stationnarité

Parmi les modèles présentés précédemment, seuls les modèles de régression prennent en compte la non-stationnarité des données. Typiquement, un modèle dont les paramètres varient au cours du temps est non-stationnaire. Il est important de faire la distinction entre la stationnarité d'un processus, l'homogénéité d'une chaîne de Markov, et celle d'une chaîne de

Semi-Markov. Un processus stationnaire (dont les propriétés restent les mêmes dans le temps) peut être décrit efficacement par des chaînes de Markov ou Semi-Markov homogènes, c'est à dire dont les paramètres restent inchangés dans le temps. Mais le terme homogène conduit à deux situations bien différentes pour une chaîne de Markov ou Semi-Markov. Dans le premier cas, l'homogénéité signifie que les taux de transitions contenus dans la matrice sont constants, et amène donc à la distribution exponentielle des temps de maintien : par définition une chaîne de Markov homogène aura une telle distribution pour les temps de maintien. Une chaîne de Semi-Markov homogène ne l'est pas au sens markoviens du terme, mais l'est dans le sens statistique du terme. En effet, les caractéristiques statistiques de la chaîne (taux de changements d'état et temps de maintien) sont constantes dans le temps.

Les techniques présentées ici sont proches des techniques de rééchantillonnage, car l'analyse des données a montré que le processus n'est pas stationnaire. Cette non-stationnarité n'est pas déterministe, c'est à dire avec une tendance bien définie ou une saisonnalité ou périodicité précise. Dans le cas contraire, des méthodes paramétriques peuvent être appliquées, comme pour les modèles de régression. Carpinone *et al.* (2010) utilisent le principe de fenêtre mobile pour générer des séries temporelles de puissance produite par des éoliennes. La puissance produite est séparée en plusieurs catégories, chacune correspondant à un niveau de production. Elle utilise des chaînes de Markov de premier et second ordre. Les changements entre les niveaux de production sont aléatoires, elle conclut donc que la série temporelle résultante est non-stationnaire. Afin de pouvoir prendre en compte cette non-stationnarité la matrice des probabilités de transitions est modifiée à chaque pas de temps. Ainsi pour prédire l'état à $x_{s+1}$, les paramètres sont obtenus avec l'historique contenu dans la fenêtre $[t_{s-U}, t_s]$. Pour estimer l'état à $x_{s+2}$, la fenêtre $[t_{s-U+1}, t_{s+1}]$ est utilisée, et ainsi de suite. Toth *et al.* (2000) s'appuie sur la même méthode pour calculer les paramètres d'un modèle ARMA. Xue *et al.* (2000) utilisent également cette idée pour créer une chaîne de Markov saisonnière, en utilisant douze matrices de probabilité de transitions, une pour chacun des mois de l'année. Les exemples de cette technique de fenêtre mobile sont présentés dans le contexte de chaîne de Markov, mais on imagine qu'elle peut être utilisée pour tous les modèles présentés, exception faite des modèles de rééchantillonnage.

## 2.2 Validation et sélection de modèles

Lors de la validation de modèle, nous devons garder en tête la citation de début de chapitre. Elle signifie qu'il est impossible d'obtenir un modèle universellement bon, mais il est possible d'en construire un capable de réaliser la tâche pour laquelle il a été conçu. La procédure de validation a pour objectif de déterminer l'utilité d'un modèle et ses limitations. Une autre façon d'aborder le problème est de répondre à cette question : « Est-ce que le modèle représente adéquatement la réalité ? » Comme le précise Bayarri *et al.* (2007), le but de cette question dans un contexte de prédiction n'est pas de savoir si le modèle est proche de la réalité ou s'il

peut aider à la compréhension du processus, mais bien de savoir s'il fournit des prédictions suffisamment précises pour être utilisé. C'est dans cette optique qu'ils proposent la procédure suivante :

**Étape 1.** Spécifier les entrées et les paramètres du modèle ainsi que leurs incertitudes.

**Étape 2.** Déterminer le (ou les) critère(s) d'évaluation.

**Étape 3.** Collecter les données.

**Étape 4.** Faire des approximations des sorties du modèle.

**Étape 5.** Analyser les résultats du modèle en les comparant avec des données observées.

**Étape 6.** Utiliser les résultats pour améliorer le modèle.

L'ordre des trois premières étapes importe peu. Dans certains cas la collecte et l'analyse des données orientent le choix du modèle. Dans d'autres, c'est le modèle qui a une incidence sur la collecte. Quantifier l'incertitude des paramètres est important : le but est de connaître l'erreur associée à leur estimation afin de pouvoir la prendre en compte. Une fois celle-ci connue, nous devons la répercuter sur les résultats par une analyse de sensibilité ou des simulations de Monte-Carlo. Nous abordons cet aspect dans le chapitre 4.

### 2.2.1 Critères d'évaluation

Le choix des critères d'évaluation est central, car c'est sur celui-ci que repose la justesse de la validation. Pour chaque étude, ils seront différents et dépendants de l'application du modèle. Par exemple, pour Fowler *et al.* (2005), l'important est de reproduire le bon nombre de jours de pluie par an, alors que Carpinone *et al.* (2010) regardent la précision de la prédiction à chaque pas de temps. Shamshad *et al.* (2005) utilisent plusieurs critères : la distribution des probabilités d'apparition de chacun des états, la distance entre les fonctions d'autocorrélation et la distance entre les densités spectrales des séries temporelles. Dubrovský (1997) en vérifie également plusieurs, dont les distributions des temps de maintien dans chaque état. On peut aussi calculer directement la distance entre les séries temporelles générées et observées (Abdel-Aal et a.Z. Al-Garni, 1997), ou encore se servir de l'analyse des résidus (Ogata, 1989).

Une fois le critère choisi, nous devons déterminer la manière d'obtention de l'erreur. Selon Hyndman et Koehler (2006), il en existe trois types : celles dépendantes de l'échelle, celles basées sur les pourcentages d'erreur par rapport aux valeurs observées et enfin celles relatives à un modèle de référence. La sélection du type d'erreur dépend également de l'application du modèle. Pour chacune des catégories, nous devons faire un choix entre différentes erreurs.

Celui-ci est arbitraire, puisqu'il n'existe aucun consensus sur la supériorité d'une méthode relativement à une autre.

### 2.2.2 Validation croisée

Après avoir déterminé les critères d'évaluation et leurs méthodes de calcul, nous devons définir à partir de quelles données nous les calculerons. Suivant l'application du modèle les approches peuvent être différentes. Shmueli (2010) distingue deux principaux types de modèles : l'explicatif et le prédictif. Le premier est un modèle statistique créé pour tester des hypothèses théoriques où des données $X$ et $Y$ sont utilisés comme des outils pour obtenir le modèle $f$ le plus juste tel que $Y = f(X)$. Quant au second, il est utilisé pour prédire la valeur $Y$ en fonction d'entrées $X$, ici les données $X$ et le modèle $f(X)$ sont des outils pour estimer les valeurs $Y$ le plus précisément possible. Nous considérons la prévision temporelle comme un cas particulier de la prédiction. La grande différence entre les deux types de modèles concerne les données qui sont utilisées pour les évaluer. En effet, pour les modèles explicatifs, une procédure *in-sample* est préférée. Les données qui servent à leur validation sont les mêmes que celles qui ont permis de les construire. Par contre, pour les modèles prédictifs, s'ils collent trop parfaitement aux données d'entraînement, ils risquent d'avoir de pauvres performances en prédiction, on parle alors de surparamétrisation. Il est nécessaire d'employer une procédure *out-of-sample*, où les ensembles d'entraînement et de validation sont différents. Si le modèle est évalué avec un seul ensemble de validation, la connaissance de sa capacité de généralisation s'en trouve limitée. Les techniques de validation croisée sont préférées car elles permettent l'évaluation du modèle sur plusieurs ensembles de validation. Dans leur revue, Arlot et Celisse (2010) listent les diverses méthodes de validation croisée. Malgré le grand nombre présent dans cette revue, la plupart sont des variations d'une même procédure. Nous en retiendrons ici seulement trois : le *Leave-K-Out* (LKO), la validation croisée à V blocs (VCVB) et le validation croisée par Bootstrap (VCB). Le LKO est une procédure exhaustive qui est illustrée avec la Figure 2.3. Pour chaque valeur $X_i$ de l'ensemble de données original de taille $n$, un échantillon de taille $K$ constitué des valeurs $X_i, ...X_{i+K-1}$ est retiré pour servir d'ensemble de validation, les données restantes forment celui d'entraînement. Le modèle est ainsi évalué pour chacune des $n - p$ valeurs. Ceci permet de noter le modèle sur toutes les données disponibles, mais peut demander beaucoup de ressources système et de temps. Le cas particulier, $K = 1$, est la procédure appelée *Leave-One-Out*.

La VCVB (Figure 2.4) n'est pas une procédure exhaustive, mais elle permet d'évaluer les modèles dans des temps raisonnables. L'ensemble de données original est séparé en $V$ blocs de taille $K$. Chaque bloc est retiré pour servir d'ensemble de validation, le reste constitue celui d'entraînement. Le modèle est ainsi testé $V$ fois. À noter qu'avec $V = n$ on retrouve le *Leave-One-Out*, si l'on autorise les blocs de tailles $K$ à être pigé aléatoirement, la procédure est appelée Validation croisée de Monte-Carlo.

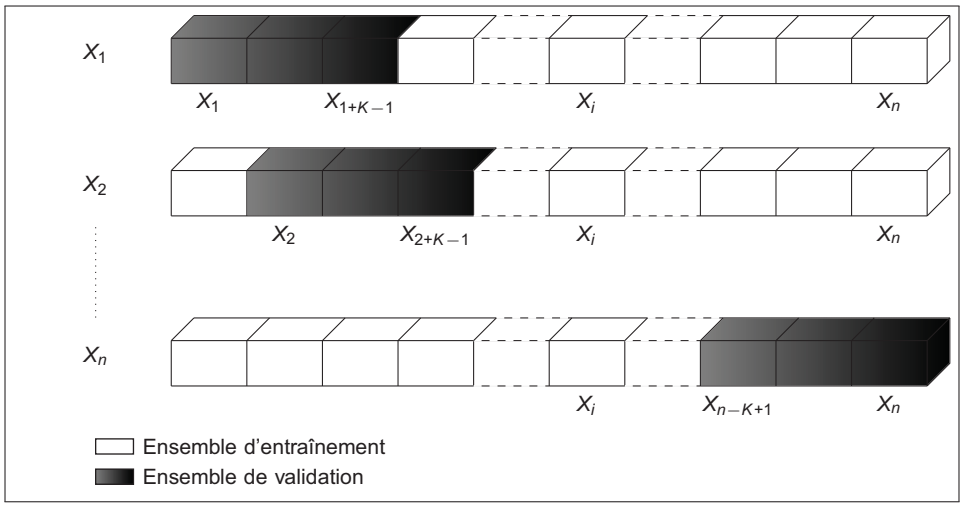


Figure 2.3 Leave-K-out

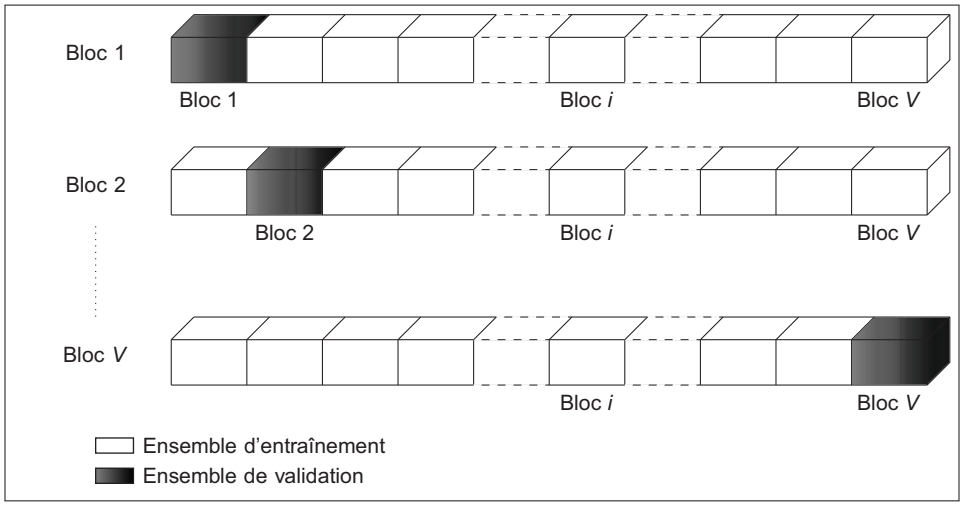


Figure 2.4 Validation croisée à V blocs

La dernière procédure utilise le Bootstrap (voir section 2.1.4), nous illustrons le principe de cette procédure à la Figure 2.5. Elle est plus complexe que les précédentes : premièrement $b$ échantillons $B_i^*$ sont générés par Bootstrap de l'ensemble original. À partir de ceux-ci, $b$ modèles sont estimés. Pour chacun d'entre eux, une simulation $S_{B_i^*}$ est réalisée. En comparant chaque $S_{B_i^*}$ avec son échantillon $B_i^*$ ($i = 1...b$) correspondant ont obtient l'erreur apparente $e_A$. Si l'on compare les $S_{B_i^*}$ avec l'ensemble original, on calcule l'erreur de prédiction $e_P$. Enfin, une dernière simulation est réalisée avec le modèle dont les paramètres ont été estimés avec l'ensemble original. Cette simulation est comparée avec l'ensemble orignal pour déterminer

une troisième erreur $e_O$. L'erreur $E$ du modèle est alors calculée selon l'Équation 2.9.

$$E = e_O + (e_P - e_A) \tag{2.9}$$

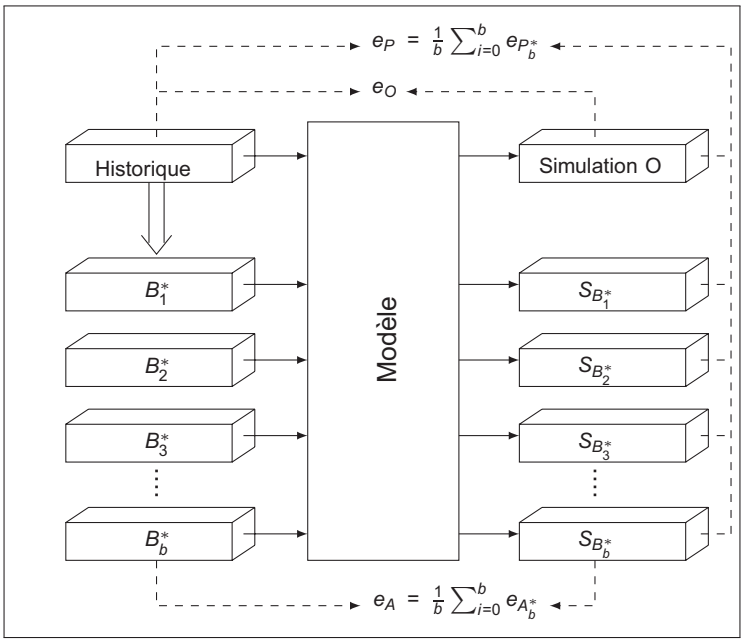


Figure 2.5 Validation croisée par Bootstrap

Un bémol est cependant mis sur l'utilisation de la validation croisée avec des données dépendantes (les séries temporelles par exemple). En effet, ces méthodes ont été créées pour des données indépendantes pour lesquelles l'indépendance entre les ensembles d'entraînement et de validation est assurée. Ce qui n'est pas le cas avec des séries temporelles. Pour régler ce problème, la principale approche, selon Arlot et Celisse (2010), est d'utiliser des ensembles de validation et de tests suffisamment éloignés pour qu'ils ne soient pas dépendent entre eux. Comme l'illustre Racine (2000), qui propose la validation croisée HK-bloc (basée sur le principe du LKO) dans le but d'assurer cette indépendance. Pour chaque élément $i$ de l'ensemble original, on défini un bloc de données en prenant $v$ données de chaque côté de l'élément $i$ que l'on retire pour servir d'ensemble de validation. On défini ensuite un « tampon » de taille $h$ aux extrémités de cet ensemble que l'on retire de l'ensemble original. Un ensemble d'entraînement de taille $n-(2v+1)-2h$ et un ensemble de validation de taille $2v+1$ sont ainsi créés. On peut facilement penser à l'adaptation de ce principe à la VCVB : on enlève à l'ensemble d'entraînement un tampon de chaque côté du bloc de validation. L'application aux données dépendantes de la VCB pourrait se faire s'il l'on s'appuie sur le Bootstrap à bloc mobile. La

méthode de Racine ne prend pas en considération l'ordre des données qui peut avoir son importance dans le contexte des séries temporelles (particulièrement avec la présence d'une tendance). Cette méthode conduit à prédire le passé avec le futur. Pour éviter cela, Bengio et Chapados (2003) proposent une adapation de la VCVB : la validation séquentielle. Elle permet de garder la structure temporelle grâce à l'utilisation d'un ensemble de validation glissant et d'un ensemble d'entraînement à taille variable. Cependant, avec cette méthode, la précision d'un modèle ne dépend plus uniquement de la différence entre les ensembles de validation, mais également de la taille de l'ensemble d'entraînement.

À partir des résultats de la validation croisée, il est possible de se servir de tests d'hypothèse pour déterminer le meilleur modèle (Dietterich, 1997). Ceux-ci permettent de comparer seulement deux modèles. Dans le cas de d'un grand nombre de modèles, nous devons comparer tous les modèles deux à deux, ce qui pourrait se révéler très fastidieux. Certains, tel Armstrong (2007), critiquent l'omniprésence et la confiance aveugle dans les tests d'hypothèse. Même s'ils semblent être des incontournables dans la validation de modèles, ils ne sont pas exempts de défauts. Bernardo et Rueda (2002) les répertorient, de l'aspect arbitraire dans la sélection du test et du niveau de signification, en passant par l'absence de procédure générale pour leur utilisation. Nous pensons alors qu'il est plus judicieux de ne pas baser notre choix de modèle sur des tests d'hypothèse, mais principalement sur des analyses approfondies des résultats.

Dans le cas où une procédure *out-of-sample* est impossible, par exemple à cause d'une quantité de données insuffisante, il est envisageable d'utiliser les deux indicateurs suivant : le AIC (*Akaike Information Criterion*) et le BIC (*Bayes Information Criterion*). Le but de ces deux indicateurs est de limiter le risque de surparamétrisation, en pénalisant un modèle en fonction du nombre de paramètres. Ainsi, un modèle moyennement précis mais avec peu de paramètres sera préféré à un modèle très précis avec beaucoup de paramètres. Comme le signale Shmueli (2010), AIC et BIC sont souvent mis en concurrence, mais selon la littérature, les deux auraient des applications bien différentes. En effet, l'AIC mesure la précision de prédiction tandis que le BIC rend compte du maximum de vraisemblance.

## 2.3 Conclusion du Chapitre 2

Nous avons montré dans ce chapitre que de nombreux modèles cohabitent. Certains font l'objet de beaucoup de développements théoriques, mais manque d'utilisations pratiques (les modèles autorégressifs discrets). D'autres, tel que les modèles non paramétriques, voient leur nombre d'applications pratiques augmenter. Certains sont moins adaptés à nos besoins car ils présentent une complexité dont nous n'avons pas besoin (réseaux neurones et chaînes de Markov cachées). De leur côté, les modèles de la famille de Markov ont connu et connaissent encore de nombreux développements théoriques et d'applications pratiques. La large littérature

disponible montre l'efficacité que peuvent avoir ces modèles dans des applications proches de la nôtre.

Associé à la création des modèles, il y a leur validation. Les points à retenir sont les suivants :

- Le choix des critères d'évaluation est à la discrétion du modélisateur, mais ils doivent être représentatifs de ce qui est attendu des modèles.
- La sélection des outils utilisés pour calculer les performances des modèles selon les critères d'évaluation dépend de ces derniers.
- Il est préférable d'utiliser des techniques de validation croisée afin d'évaluer les modèles avec plusieurs ensembles de données. L'emploi de ces techniques doit être faite avec précaution dans le cas de séries temporelles.

# CHAPITRE 3

# PROCÉDURE DE VALIDATION

Notre procédure de validation est présentée dans ce chapitre. Si l'on se réfère à la procédure proposée par Bayarri *et al.* (2007), déjà vue dans la Section 2.2, nous ne discuterons ici que des points concernant le choix des critères d'évaluation ainsi que de l'analyse des résultats.

## 3.1 Choix des critères d'évaluation

Évaluer les performances en se basant uniquement sur un critère n'est pas toujours suffisant, il arrive souvent que l'on souhaite que le modèle reproduise adéquatement plusieurs caractéristiques des données observées (Dubrovský, 1997; Shamshad *et al.*, 2005). Nous avons retenu plusieurs critères d'évaluation qui nous permettent de nous assurer que **les séquences simulées par les modèles sont statistiquement équivalentes aux séquences observées**. En effet, dans notre cas, le modèle doit être capable de reproduire d'une part le nombre total de chaque changement d'état et d'autre part le taux d'évolution journalier du nombre de changements d'état.

### 3.1.1 Critère 1 : Nombre de changements d'état

Les fluctuations de contraintes (responsable de la propagation des fissures) sont intrinsèquement liées au nombre de changements d'état. Ces derniers ont une influence directe sur la durée de vie des roues de turbines. Il est donc important de connaître l'erreur du modèle, celui-ci doit être le plus précis possible tout en intégrant tous les comportements présents dans les données observées. L'analyse des données nous a en effet montré qu'il y a de grandes variations du nombre de changements d'état d'une année à l'autre. Ces variations étant a priori aléatoires, nous pouvons difficilement prévoir si l'année suivante sera une année avec peu ou beaucoup de changement.

Nous évaluons les performances des modèles pour ce critère en calculant deux versions de l'erreur de prédiction du nombre total de changements d'état : la MAPE (*Mean Absolute Percentage Error*), définie par l'équation (3.1), et la MAE (*Mean Absolute Error*) définie par l'équation (3.2). MAPE et MAE sont calculées pour chacun des six changements d'état.

$$MAPE_C = \frac{1}{M}\sum_{i=1}^{M}\frac{|N_{Ci} - N_C|}{N_C} \tag{3.1}$$

$$MAE_C = \frac{1}{M}\sum_{i=1}^{M}|N_{Ci} - N_C| \tag{3.2}$$

Nous définissons $N_{Ci}$ le nombre de changements d'état $C$ produit par la simulation $i$, $N_C$ le nombre de changements d'état observé et $M$ le nombre de simulations. $C$ est l'indice représentant le type de changement d'état, tel que $C = [BP, MP, PhA, PhD, F, MR]$. La première erreur est recommandée par Hyndman et Koehler (2006) pour comparer la performance de modèle sur différents ensembles de données. La seconde permet d'avoir une vision plus complète et plus proche de la réalité physique du modèle.

### 3.1.2 Critère 2 : Taux d'évolution du nombre de changements d'état

Trois types de cycles cohabitent dans le chargement des GTA chacun avec sa propre amplitude. Il y a ceux entre MN et A, ceux entre MN et MV et dans une moindre mesure ceux entre MV et A. L'interaction entre ces différents cycles doit être prise en compte (Skorupa, 1998). En effet,

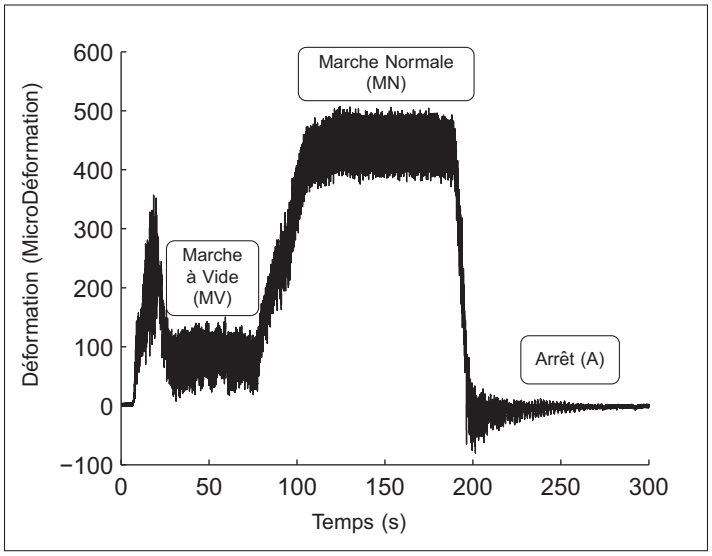


Figure 3.1 Micro-déformations mesurées sur une turbine en fonctionnement

une fissure ne se propagera pas de la même façon si les cycles sont répartis également dans le temps ou regroupés par blocs. Évaluer la capacité des modèles à produire des taux d'évolution réalistes permet de s'assurer que la répartition des cycles simulés soit équivalente à celle des cycles observés. Comme la répartition des cycles est directement dépendante de la répartition des changements d'état, c'est celle-ci que nous évaluons.

La Figure 3.2a présente le taux d'évolution du nombre de BP par jour de l'année 1, ainsi que trois cas de taux d'évolution. Les graphes de taux d'évolution sont recommandés par Weiß (2008) pour visualiser les séries temporelles catégoriques (Un taux d'évolution non linéaire est typique des processus non stationnaires). À la Figure 3.2b, nous trouvons les fonctions

de répartition (FdR) du nombre de BP par jour correspondants aux taux d'évolution. Une

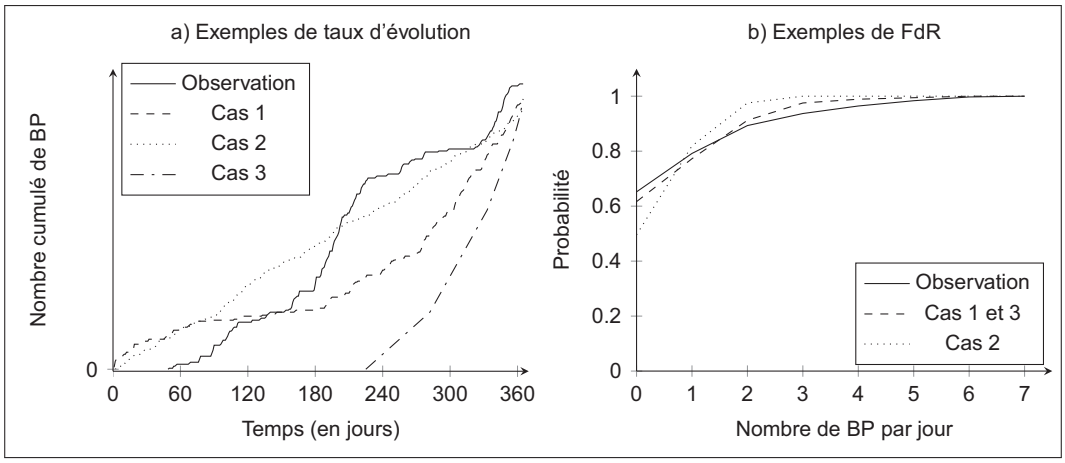


Figure 3.2 Lien entre taux d'évolution et FdR

manière directe d'évaluer ce second critère aurait été de calculer une distance entre les taux d'évolution. Cependant, cela reviendrait à préférer un modèle dont les simulations ont un taux d'évolution linéaire (cas 2 sur la Figure 3.2) au détriment du cas 1. Le cas 1, même s'il n'est pas identique aux données, possède un comportement non stationnaire similaire. Pour cette raison nous utilisons la distance entre les fonctions de répartition (FdR) observées et simulées du nombre de chacun des changements d'état par jour et la distance ente les FdR, observées et simulées, des temps de maintien. Nous voyons avec la Figure 3.2b, que le cas 1 sera alors préféré car la distance entre les deux FdR est moins grande. Le critère 2 est donc évalué à partir de deux indicateurs, que l'on appellera $D_C$ et $D_E$, avec $C$ l'indice du type de changement d'état, et $E$ celui de l'état, tel quel $E = [MN, MV, A]$. Nous avons choisi de travailler avec deux indicateurs car le premier, $D_C$, ne permet pas de faire la distinction entre le cas 1 et le cas 2 (Figure 3.2b). La réalisation d'une simulation qui conduirait au taux d'évolution du Cas 3 nécessite que le GTA ne subisse aucun changement de régime pendant plus de la moitié de l'année, chose qui modifie grandement les distributions de temps de maintien. D'où l'utilisation de ce second indicateurs, et ce, même si la construction et le fonctionnement des modèles rendent très improbable la réalisation du cas 3.

L'éloignement entre deux FdR sera mesuré par la distance de Kolmogorov-Smirnov. Pour toutes les valeurs $x_m$ prises par $S_i$ et $R$, nous calculons l'écart entre les probabilités cumulées associées à $x_m$. La distance de Kolmogorov-Smirnov est définie comme l'écart maximum. Cette distance est calculée pour les FdR du nombre de changements d'état par jour et pour les FdR des temps de maintien. L'erreur $D_i$ (présenté à la Figure 3.3) est associée à une seule simulation.

$D_C$ et $D_E$ sont calculés avec l'équation 3.3, $M$ étant le nombre de simulations.

$$D = \frac{1}{M}\sum_{i=1}^{M} D_i \tag{3.3}$$

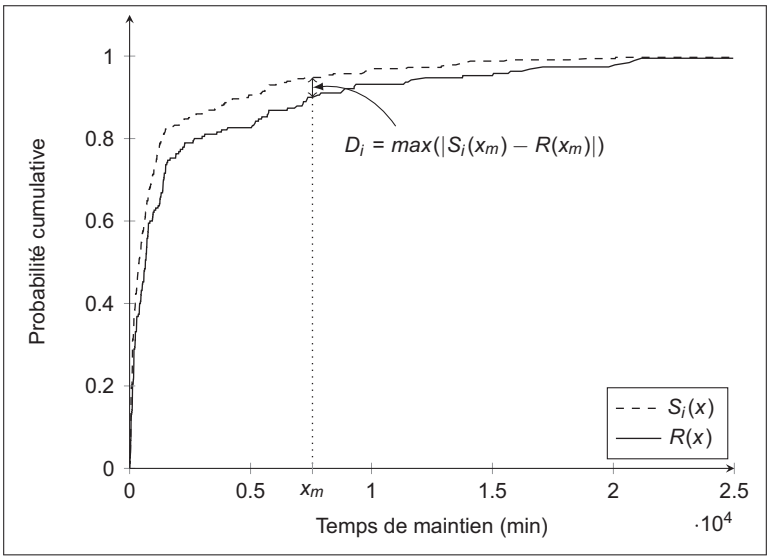


Figure 3.3 Distance de Kolmogorov-Smirnov

## 3.2 Procédure d'analyse

### 3.2.1 Choix de la méthode de validation croisée

Trois méthodes de validation croisée ont été décrites dans la Section 2.2.2 : le Leave-K-out, la validation croisée par Bootstrap, et la validation croisée à V blocs. L'exhaustivité du Leave-K-out est dans notre cas problématique, car l'importante taille de la série de données rend cette méthode trop chronophage. La validation croisée par Bootstrap est plus récente et rassemble de plus en plus d'adeptes, tels Lendasse *et al.* (2003) qui suggèrent qu'elle est plus précise que les autres. De leur côté Arlot et Celisse (2010) signalent que très peu de validations théoriques de ce principe existent. Finalement, la validation croisée à V blocs, plus ancienne, est recommandée dans de nombreux cas (Nadeau et Bengio, 2003; Salzberg, 1997). Ayant fait ses preuves, c'est cette validation croisée non exhaustive que nous utiliserons.

Les données, qui couvrent sept années, sont donc séparées en sept blocs d'une année. Toutefois, l'application de la validation croisée aux séries temporelles peut poser deux problèmes, comme déjà mentionné dans la section 2.2.2. Le premier est le risque d'avoir des ensembles

d'entraînement et de validation qui ne sont pas indépendants. Le second est la non-prise en compte de l'ordre temporel des blocs pouvant amener à prédire le passé avec le futur. Afin de régler le premier problème, Racine (2000) a proposé de retirer des données tampons entre l'ensemble d'entraînement et celui de validation. Dans notre cas, avec les informations disponibles, rien ne permet de penser que le nombre de changements d'état d'une année aura une influence sur celui de l'année suivante. La seule influence va provenir de la coupure entre deux années en supprimant un changement d'état ou en tronquant un temps de maintien. Si nous coupons nos données en 7 années, un maximum de six changements d'état disparaîtrons, ou un maximum de six temps de maintien seront tronqués, ce qui n'aura qu'une très faible influence sur les statistiques globales de la série temporelle. Concernant l'ordre des blocs, le but est de savoir si la validation séquentielle (décrite dans Bengio et Chapados (2003)) est préférable à la validation classique. L'analyse des données a montré que les variations des données ne suivent ni tendance, ni périodicité. Nous avons retenu l'hypothèse que le comportement de l'année à venir peut être celui d'une des six années précédentes. Ceci a comme corollaire d'annuler toute importance de l'ordre des années. Nous présentons, avec la Figure 3.4, la validation croisée à V blocs telle qu'appliquée à nos données. Les modèles sont donc testés sur sept blocs, chaque bloc d'une année est retiré à son tour pour servir d'ensemble de validation et est considéré comme le futur à reproduire.

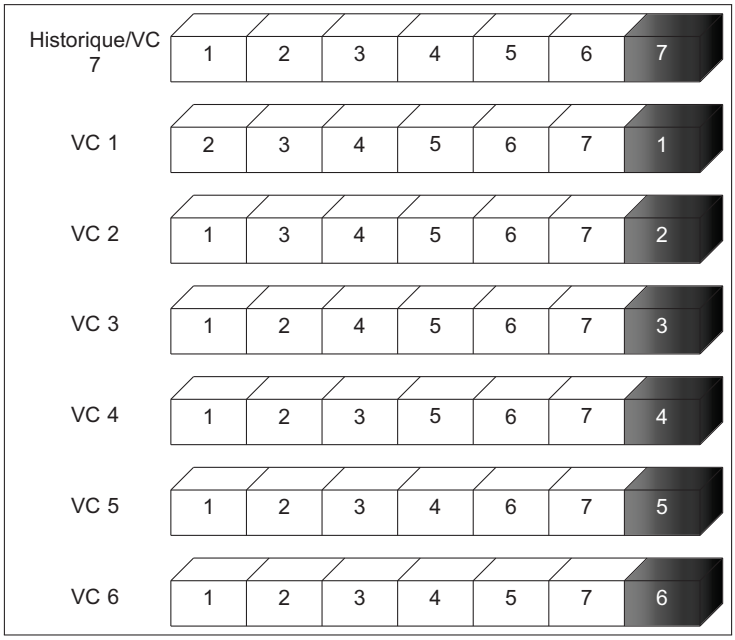


Figure 3.4 Validation croisée à V blocs telle qu'appliquée à nos données

### 3.2.2 Calcul des erreurs de généralisation

L'erreur de généralisation est définie comme la moyenne des erreurs sur chacun des sept blocs. Les erreurs de généralisation basées sur les critères définis précédemment sont regroupées dans le Tableau 3.1.

Tableau 3.1 Erreurs de généralisation calculées

| | **Critère 1** | **Critère 2** |
|---|---|---|
| Changements d'état<br>$C = [BP, MP, PhA, PhD, F, MR]$ | $GE_{MAPE} = \frac{1}{V}\sum_{b=1}^{V} MAPE_{C_b}$<br>$GE_{MAE} = \frac{1}{V}\sum_{b=1}^{V} MAE_{C_b}$ | $GE_{D_C} = \frac{1}{V}\sum_{b=1}^{V} D_{C_b}$ |
| Temps de maintien<br>$E = [MN, MV, A]$ | | $GE_{D_E} = \frac{1}{M}\sum_{b=1}^{M} D_{E_b}$ |

Comme nous préférons prendre une décision sur le choix du modèle basée sur des analyses complètes et non sur des résultats de tests d'hypothèse, nous n'utiliserons pas les tests associés au calcul d'erreur de généralisation décrits par Dietterich (1997).

## 3.3 Conclusion du chapitre 3

Nous avons vu dans ce chapitre comment nous évaluerons la qualité des modèles. Cette évaluation se base sur des principes largement utilisés dans la littérature. Toutefois, nous avons décidé de limiter notre utilisation des tests d'hypothèse qui peuvent parfois être trompeurs. Nous utiliserons la validation croisée à V blocs pour obtenir les erreurs de généralisation des deux critères sélectionnés. Nous calculons pour le critère 1 deux versions de l'erreur sur le nombre de changements d'état, alors que le critère 2 est évalué avec deux indicateurs complémentaires. Nous décrivons dans le chapitre suivant la construction et le fonctionnement des trois modèles que nous avons choisis de comparer.

# CHAPITRE 4

# PRÉSENTATION DES MODÈLES ET PREMIÈRE COMPARAISON

Dans ce chapitre nous présentons les trois modèles qui seront comparés, à savoir le Bootstrap à bloc mobile, les chaînes de Markov à temps discret et les chaînes de Semi-Markov à temps discret. En fin de chapitre, nous présentons une première comparaison du fonctionnement de ces modèles.

## 4.1 Choix des modèles

Parmi les modèles décrits dans la Section 2.1, la plupart sont paramétriques et quelques-uns non-paramétriques. Ces derniers sont plus fidèles aux données, car ils ne nécessitent pas les approximations inhérentes à l'estimation de paramètres. En revanche, ils se limitent à la reproduction d'événements passés, sans la possibilité de générer de nouveaux scénarios.

Certains choisissent d'augmenter le nombre de paramètres pour améliorer la fidélité d'un modèle paramétrique envers la réalité. Ceci comporte toutefois un risque : la surparamétrisation du modèle, pouvant conduire conduire à une surspécialisation du modèle. Le modèle se retrouve alors avec les mêmes limitations qu'un modèle non paramétrique : il ne pourra produire des résultats autres que les données historiques. Ceci peut toutefois être bénéfique dans le cas d'un modèle explicatif, mais ne l'est sûrement pas pour un modèle prédictif. Pour le cas prédictif, il est conseillé d'utiliser des modèles simples, avec peu de paramètres (principe de parcimonie), et une bonne capacité de généralisation. Comme nous l'avons vu dans le chapitre précédent, l'important n'est pas de reproduire le mieux la réalité, mais de prédire au mieux le futur. D'autres préfèrent l'approche semi-paramétrique, qui consiste à construire un modèle avec une partie paramétrique et une autre non paramétrique. Cette approche permet à un modèle paramétrique d'améliorer sa fidélité aux données sans augmenter dramatiquement le nombre de paramètres, et à un modèle non paramétrique d'être moins limité à ses données d'entrée.

À la Figure 2.1, nous remarquons que les modèles de la famille de Markov sont les seuls qui sont à la fois très présents dans la littérature des séries temporelles catégoriques et dans les travaux qui décrivent des générateurs de séries temporelles. Pour cette étude, nous avons donc privilégié cette famille de modèles. Nous avons vu que le cadre très strict des chaînes de Markov peut parfois limiter leur utilisation. Plusieurs modèles visent à relaxer ce cadre. Parmi eux, les chaînes de Semi-Markov apportent beaucoup de flexibilité tout en gardant l'estimation des paramètres relativement simple. Elles possèdent trois avantages remarquables :

1. Contrairement aux chaînes de Markov, les temps de maintien peuvent suivre n'importe quelles distributions statistiques (exponentielle, log-normale ...).
2. D'une manière similaire aux chaînes de Markov, les paramètres sont très concrets.
3. Elles ont la capacité d'être employées comme modèle paramétrique ou semi-paramétrique (Greenwood *et al.*, 2004).

Afin de voir les différences entre modèles non paramétriques, paramétriques et semi-paramétriques, nous comparerons le Bootstrap à bloc mobile, les chaînes de Markov et celles de Semi-Markov.

## 4.2 Bootstrap à bloc mobile (BBM)

Nous décrivons dans cette section le Bootstrap à bloc mobile. Dans un premier temps, nous faisons une présentation générale du modèle, nous abordons ensuite le choix de la taille de bloc, élément qui aura une influence sur les performances d'un tel modèle. Enfin, nous présentons l'algorithme qui nous permettra de générer des séries temporelles synthétiques avec ce modèle.

### 4.2.1 Description générale

Le Bootstrap à Bloc Mobile, proposé par Kunsch (1989), est une adaptation du Bootstrap (décrit par Bradley Efron en 1979). Le Bootstrap a été créé à l'origine pour être utilisé avec des données indépendantes. Le BBM permet d'appliquer ce principe aux données dépendantes comme les séries temporelles. Le BBM est, avant tout, destiné à être employé avec des données dépendantes stationnaires, mais de nombreuses études ont montré sa robustesse lorsqu'utilisé sur des données non stationnaires (Gonçalves et White, 2002; Synowiecki, 2007).

Lorsqu'utilisé à des fins de prédiction, on comprend que le BBM est inadapté si la série temporelle à prédire présente une tendance. Par contre, il est applicable aux séries temporelles périodiques (Synowiecki, 2007). Dans nos données, ni tendance ni périodicité n'ont été détectée, donc rien ne semble contre-indiquer l'application du BBM. Le principe est présenté dans la Figure 4.1 [1]. Le but est de recréer une série chronologique synthétique en mettant bout à bout des blocs pigés, avec replacement, aléatoirement dans l'ensemble des données disponibles.

### 4.2.2 Choix de la taille de blocs

Le choix de la taille de blocs est l'élément le plus critique du BBM. Celle-ci a un effet majeur sur les performances de ce modèle (Bühlmann, 2002). Une taille de bloc trop petite risque, en effet, de « casser » la dépendance temporelle des données, alors que si elle est trop grande, elle peut

1. La taille du bloc est ici arbitraire pour les besoins de l'illustration.

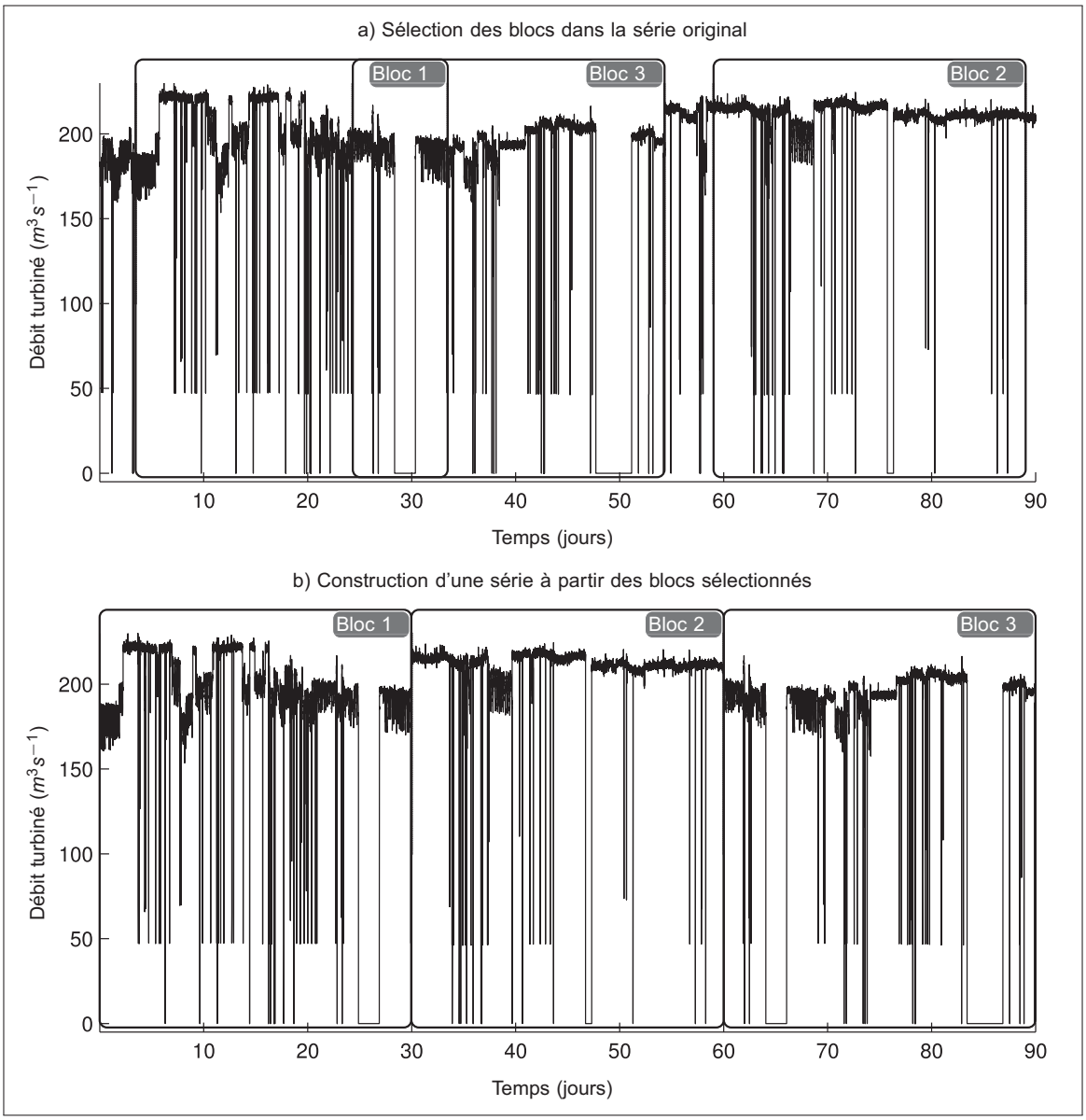


Figure 4.1 Exemple du Bootstrap à bloc mobile

ne pas prendre en compte toute la variabilité et la dispersion des données. La taille de bloc optimale est propre à chaque série temporelle. Plusieurs méthodes ont été proposées afin de l'obtenir. Selon Bühlmann (2002), Hall *et al.* (1995) proposent la plus générale. L'obtention de la taille de bloc optimale se fait en deux étapes. Premièrement une taille de bloc $\hat{l_m}$ est obtenue en évaluant la performance du BBM avec différentes tailles de bloc sur des sous-échantillons de taille $m$. La taille de bloc optimale $\hat{l_n}$ pour l'échantillon complet de taille $n$ est alors définie comme étant $\hat{l_n} = (n/m)^{1/k}\hat{l_m}$, avec $k = 3, 4$ ou 5. Afin de déterminer la taille de bloc idéale, nous évaluerons directement les performances du BBM avec différentes tailles.

### 4.2.3 Algorithme

L'algorithme 4.1 décrit l'utilisation du Bootstrap dans l'optique de générer de nouvelles séries temporelles. L'entrée est la série temporelle originale que l'on considère composée de $b = n - l + 1$ blocs qui se chevauchent, $n$ étant la taille de la série original et $l$ la taille de bloc.

**Entrées** : Série temporelle de taille $n$ coupée en $b$ blocs de taille $l$
**Sorties** : Série temporelle synthétique de taille $T$
**début**
    $T \longleftarrow Temps\,de\,simulation$ `// Longueur de la série à générer`
    $N \longleftarrow T/l$
    **pour** $i = 1$ *à* $N$ **faire**
        $f \longleftarrow random[0,b]$ `// Tirage au sort d'un nombre entre 0 et b`
        $B_i^* \longleftarrow$ Bloc de donnée $\#f$
    **fin**
    Série $\longleftarrow (B_1^*, B_2^* ... B_N^*)$
**fin**

Algorithme 4.1 Génération d'une série temporelle à partir du BBM

Si $T < n$, nous sortons du cadre du BBM tel que décrit par Künsch, où la taille de la série de sortie doit être la même que celle d'entrée. Nous sommes alors dans le cadre du $m$ parmi $n$ Bootstrap.

## 4.3 Chaine de Markov à temps discret

Cette section est consacrée à la description des chaînes de Markov à temps discret, nous revenons en détail sur le fonctionnement de ce type de modèle. Du principe de fonctionnement à l'algorithme permettant de générer des séries temporelles en passant par la façon dont les paramètres sont estimés.

### 4.3.1 Description générale

Nous décrivons ici les chaînes de Markov d'ordre 1, telles qu'elles sont présentées dans la littérature. Comme le montre la Figure 4.2, les chaînes de Markov considèrent que chaque pas de temps est une transition. La transition vers l'état $j$ au temps $t+1$ ne dépend que de l'état $i$ dans lequel se trouve la chaîne au temps présent $t$. Ceci se traduit par l'égalité 4.1.

$$p_{ij} = P(J_{s+1} = j | J_s = i) \tag{4.1}$$

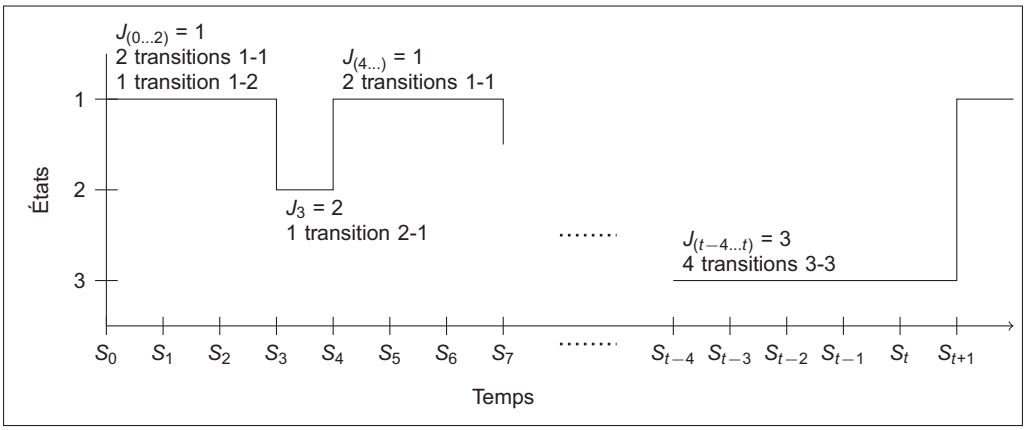


Figure 4.2 Exemple d'une chaîne de Markov

Seules les $p_{ij}$, que l'on appelle probabilités de transitions, sont nécessaires aux chaînes de Markov. Elles peuvent être regroupées dans la matrice de probabilités de transition ($P_T$, équation 4.2).

$$P_T = \begin{pmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{pmatrix} \tag{4.2}$$

Nous verrons plus loin que les chaînes de Semi-Markov sont décrites à la fois par les $p_{ij}$ et par les distributions de temps de maintien. Pour les chaînes de Markov, les temps de maintien découlent des $p_{ii}$. Leurs distributions ne sont définies que par un seul paramètre, celui de la distribution exponentielle. Les chaînes de Markov ont donc obligatoirement leurs temps de maintien exponentiellement distribués.

### 4.3.2 Estimation des paramètres

L'estimation des $p_{ij}$ est très simple, car ce sont les proportions de chaque type de transition par rapport au nombre total. Anderson et Goodman (1957) montrent que l'équation 4.3 est l'estimateur du maximum de vraisemblance des paramètres calculés à partir d'une série temporelle de taille $U$. Où $N_{ij}(U)$ représente le nombre de transitions de $i$ vers $j$ et $N_i(U)$ le nombre de transitions qui sortent de $i$.

$$\hat{p}_{ij} = \frac{N_{ij}(U)}{N_i(U)} \tag{4.3}$$

### 4.3.3 Algorithme

Pour la génération de séries temporelles avec les chaînes de Markov à temps discret, nous utilisons les probabilités de transitions cumulées, qui sont obtenues avec l'équation 4.4.

$$P_{ij} = \sum_{m=1}^{j} p_{im} \tag{4.4}$$

L'algorithme 4.2 présente la méthode de génération des séries temporelles synthétiques. À chaque itération, nous avons une nouvelle transition et donc une séquence synthétique 5 minutes plus longue. Donc pour obtenir une séquence synthétique de 4 heures, il faudra $4 * 60/5 = 48$ itérations.

**Entrées** : $p_{ij}$
**Sorties** : Série temporelle synthétique
**début**
  $T \longleftarrow Temps\,de\,simulation$ `// Longueur de la série à générer`
  $i \longleftarrow 1$ `// État initial mis sur Marche normale`
  **tant que** $t < T$ **faire**
    `// Détermination de l'état futur` $j$
    $p \longleftarrow random[0,1]$ `// Tirage au sort d'un nombre entre 0 et 1`
    **cas où** $i = 1$
      **cas où** $p < P_{11}$ **alors** $j \longleftarrow 1$
      **cas où** $P_{11} < p < P_{12}$ **alors** $j \longleftarrow 2$
      **cas où** $p > P_{12}$ **alors** $j \longleftarrow 3$
    **fin**
    **cas où** $i = 2$
      **cas où** $p < P_{21}$ **alors** $j \longleftarrow 1$
      **cas où** $P_{21} < p < P_{22}$ **alors** $j \longleftarrow 2$
      **cas où** $p > P_{22}$ **alors** $j \longleftarrow 3$
    **fin**
    **cas où** $i = 3$
      **cas où** $p < P_{31}$ **alors** $j \longleftarrow 1$
      **cas où** $P_{31} < p < P_{32}$ **alors** $j \longleftarrow 2$
      **cas où** $p > P_{32}$ **alors** $j \longleftarrow 3$
    **fin**
    $t \longleftarrow t + 5min$
    $i \longleftarrow j$
  **fin**
**fin**

Algorithme 4.2: Génération d'une série temporelle à partir de CMTD

## 4.4 Chaîne de semi-Markov à temps discret

Cette section décrivant les chaînes de semi-Markov à temps discret est structurée de la même manière que la section précédente.

### 4.4.1 Description générale

Nous présentons ici les chaînes de Semi-Markov telles qu'elles sont décrites dans le livre de Barbu et Limnios (2008). Une chaîne de Semi-Markov peut être décrite comme une combinaison de trois chaînes :

- $J = (J_n)_{n\in[1,2,3]}$ décrit les états successifs du processus ;
- $S = (S_n)_{n\in\mathbb{N}}$ est celle des temps auxquels ont eu lieu des changements d'état ;
- $X = (X_n)_{n\in\mathbb{N}}$ est celle des temps de maintien.

Ces trois chaînes sont illustrées dans la Figure 4.3. On notera que les chaînes de Markov considèrent une transition à chaque pas de temps, alors que celles de Semi-Markov ne prennent en compte que les changements d'état.

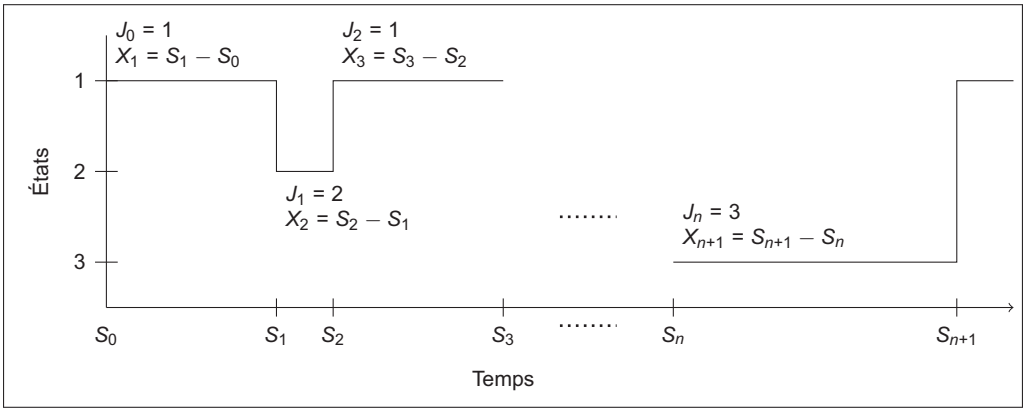


Figure 4.3 Exemple d'une chaîne de Semi-Markov

$(J,S) = (J_n,S_n)_{n\in\mathbb{N}}$ est une chaîne de Markov de renouvellement si l'égalité 4.5 est satisfaite pour tout $n,k \in \mathbb{N}$ et $i,j \in [1,2,3]$.

$$P(J_{n+1} = j, S_{n+1} - Sn = k|J_0,...,J_n;S_0,...,S_n) = P(J_{n+1} = j, S_{n+1} - Sn = k|J_n) \tag{4.5}$$

Le noyau de la chaîne de Semi-Markov (équation 4.6) est la probabilité que le processus reste dans l'état $i$ pour une durée $k$ et que le prochain état soit $j$. Ce noyau est aussi le produit de la

probabilité de changement d'état $p_{ij}$ et de la probabilité $f_{ij}(k)$ que le temps de maintien dans l'état présent soit $k$.

$$q_{ij}(k) = P(J_{n+1} = j, S_{n+1} - Sn = k | J_n = i) \tag{4.6}$$

$p_{ij}$ est défini par l'équation 4.7. On remarque que $J$ est une chaîne de Markov, avec $p_{ii} = 0$ et $\sum_{j=1}^{3} p_{ij} = 1$.

$$p_{ij} = P(J_{n+1} = j | J_n = i) \tag{4.7}$$

Les $p_{ij}$ peuvent être regroupées dans la matrice de probabilité de changement d'état ($P_{CE}$, équation 4.8), qui est à différencier de la matrice de probabilité de transition utilisée pour les chaînes de Markov.

$$P_{CE} = \begin{pmatrix} 0 & p_{12} & p_{13} \\ p_{21} & 0 & p_{23} \\ p_{31} & p_{32} & 0 \end{pmatrix} \tag{4.8}$$

$f_{ij}(k)$ peut être défini de deux façons. Soit par l'équation 4.9, on parle alors de distribution de temps de maintien conditionnel, car le temps de maintien dépend à la fois de l'état présent $i$, mais également de l'état futur $j$, soit par l'équation 4.10 et l'on parle alors de distribution de temps de séjour dans un état donné, car le temps de maintien dépend uniquement de l'état présent $i$.

$$f_{ij}(k) = P(X_{n+1} = k | J_n = i, J_{n+1} = j) \tag{4.9}$$

$$h_i(k) = P(X_{n+1} = k | J_n = i) \tag{4.10}$$

Aucun élément, dans le fonctionnement des GTA, ne nous indique que le temps de maintien dans un état présent peut dépendre de l'état futur. Par conséquent, nous utiliserons les distributions de temps de maintien dans un état donné (équation 4.10).

Nous remarquons la dualité des chaînes de Semi-Markov qui traitent séparément les probabilités de changement d'état et les temps de maintien. C'est cette dualité qui confère au CSM leur flexibilité. Les $h_i(k)$ influencent le nombre total de changements d'état, alors que les $p_{ij}$ influencent le type de changements d'état. Par exemple, si le GTA est en marche normale et qu'il change de régime au temps $t$, ce sont les $p_{1j}$ qui vont décider quel sera le nouveau régime.

### 4.4.2 Estimation des paramètres

Barbu et Limnios (2008) décrivent l'estimation non-paramétriques des quantités $h_i(k)$ et $p_{ij}$. Ils montrent que les estimateurs 4.11 et 4.12, calculés sur un échantillon de taille $M$, maximisent la fonction de vraisemblance approchée. Pour les $p_{ij}$, il s'agit du même estimateur que celui utilisé pour les chaînes de Markov. $N_i(M)$ est le nombre de fois où le processus entre dans l'état $i$, $N_{ij}(M)$ est le nombre de fois où le processus passe de l'état $i$ à $j$ et $N_i(k, M)$ est le nombre de fois où le processus reste une durée $k$ dans l'état $i$. Dans un contexte de temps continu, diverses méthodes permettent l'estimation de $h_i(k)$ et $p_{ij}$, comme celle de Greenwood et Wefelmeyer

(1996) ou encore celle d'Alvarez (2005).

$$\hat{p}_{ij} = \frac{N_{ij}(M)}{N_i(M)} \tag{4.11}$$

$$\hat{h}_i(k) = \frac{N_i(k,M)}{N_i(M)} \tag{4.12}$$

L'équation 4.12 est aussi la façon d'obtenir une fonction de densité de probabilité empirique. Nous considérons alors que de cette façon le modèle est semi-paramétrique, dans le sens où il est contrôlé d'un côté par les paramètres $p_{ij}$ et de l'autre par les FdR empiriques de temps de maintien. Si nous choisissons de décrire les temps de maintien avec des FdR paramétriques, la chaîne de Semi-Markov est alors entièrement paramétrique, comme c'est le cas chez Fowler *et al.* (2000) et Kharoufeh *et al.* (2010).

### 4.4.3 Algorithme

Toujours décrit par Barbu et Limnios (2008), nous présentons l'algorithme 4.3 qui permet de générer une série temporelle à partir d'une chaîne de Semi-Markov. Il nécessite l'emploi des probabilités cumulées pour les probabilités de changement d'état et pour les temps de maintien. Elles sont calculées avec les équations 4.4 et 4.13.

$$H_i(k) = \sum_{l=1}^{k} h_i(l) \tag{4.13}$$

## 4.5 Incertitude des paramètres

Comme l'explique Bayarri *et al.* (2007), la première étape pour valider correctement un modèle est de bien définir les incertitudes liées à l'estimation des paramètres. Et ce, afin de pouvoir les propager au modèle via des simulations de Monte-Carlo ou des analyses de sensibilité. Nous avons choisi d'utiliser la méthode de Monte-Carlo, dont le principe est de réaliser un grand nombre de simulations en prenant à chaque fois des valeurs différentes de paramètres. Les incertitudes sur les paramètres peuvent être calculées de façon paramétrique ou non paramétrique. Nous avons vu que la détermination des paramètres (et des distributions empiriques) se ramène à des calculs de proportion. L'estimation paramétrique d'un intervalle de confiance pour les paramètres peut donc se faire avec l'une des méthodes décrites par Newcombe (1998). Le principe est le même pour toutes ces méthodes : la valeur du paramètre est calculée en utilisant la globalité des données, puis un intervalle de confiance est construit autour de cette valeur.

**Entrées** : $p_{ij}$ et $h_i(k)$
**Sorties** : Série temporelle synthétique
**début**
    $T \longleftarrow Temps\,de\,simulation$ `// Longueur de la série à générer`
    $i \longleftarrow 1$ `// État initial mis sur Marche normale`
    **tant que** $t < T$ **faire**
        `// Détermination de l'état futur` $j$
        $p \longleftarrow random[0,1]$ `// Tirage au sort d'un nombre entre 0 et 1`
        **cas où** $i = 1$
            **si** $p > P_{12}$ **alors** $j \longleftarrow 3$ **sinon** $j \longleftarrow 2$
        **fin**
        **cas où** $i = 2$
            **si** $p > P_{21}$ **alors** $j \longleftarrow 3$ **sinon** $j \longleftarrow 1$
        **fin**
        **cas où** $i = 3$
            **si** $p > P_{31}$ **alors** $j \longleftarrow 2$ **sinon** $j \longleftarrow 1$
        **fin**
        `// Détermination du temps de maintien dans l'état` $i$
        $f \longleftarrow random[0,1]$ `// Tirage au sort d'un nombre entre 0 et 1`
        **si** $H_i(d-5) < f < H_i(d)$ **alors** $k \longleftarrow d$
        $t \longleftarrow t + k$
        $i \longleftarrow j$
    **fin**
**fin**

Algorithme 4.3: Génération d'une série temporelle à partir de CSMTD

L'estimation non paramétrique peut être fait avec des techniques de rééchantillonnage (tel que le Bootstrap) ou de sous-échantillonnage. L'idée derrière ce dernier est d'estimer un intervalle de confiance sur les paramètres en les calculant sur $b$ échantillons de taille $k$ tirés de la série originale (Politis *et al.*, 1999). Le sous-échantillonnage est proche du Bootstrap, mais il a l'avantage d'utiliser des morceaux de séries temporelles réelles et non créées artificiellement.

Les diagrammes à moustache présentes dans la Figure 4.4, illustrent l'incertitude des $p_{ij}$ selon les deux méthodes (paramétrique et non paramétrique). La méthode de calcul paramétrique est celle recommandée par Newcombe (1998), où les équations 4.14 et 4.15 permettent l'obtention des limites supérieure $S$ et inférieure $I$ de l'intervalle de confiance, avec $n$ la taille de l'échantillon (dans notre cas $n = N_i(M)$), $p$ la proportion dont on veut l'intervalle de confiance, $q = 1 - p$ le complémentaire de $p$ et $z$ la valeur de la table de répartition de la loi normale pour $1 - \alpha/2$.

$$S = \frac{2np + z^2 + 1 + z\sqrt{z^2 + 2 - 1/n + 4p(nq - 1)}}{2(n + z^2)} \tag{4.14}$$

$$I = \frac{2np + z^2 - 1 - z\sqrt{z^2 - 2 - 1/n + 4p(nq + 1)}}{2(n + z^2)} \tag{4.15}$$

La méthode paramétrique sous-estime largement l'incertitude. Elle ne saura rendre compte de toutes les variations rencontrées au sein des données, car celles-ci sont non stationnaires. Il est plus judicieux d'obtenir une distribution des valeurs des paramètres en utilisant le sous-échantillonnage.

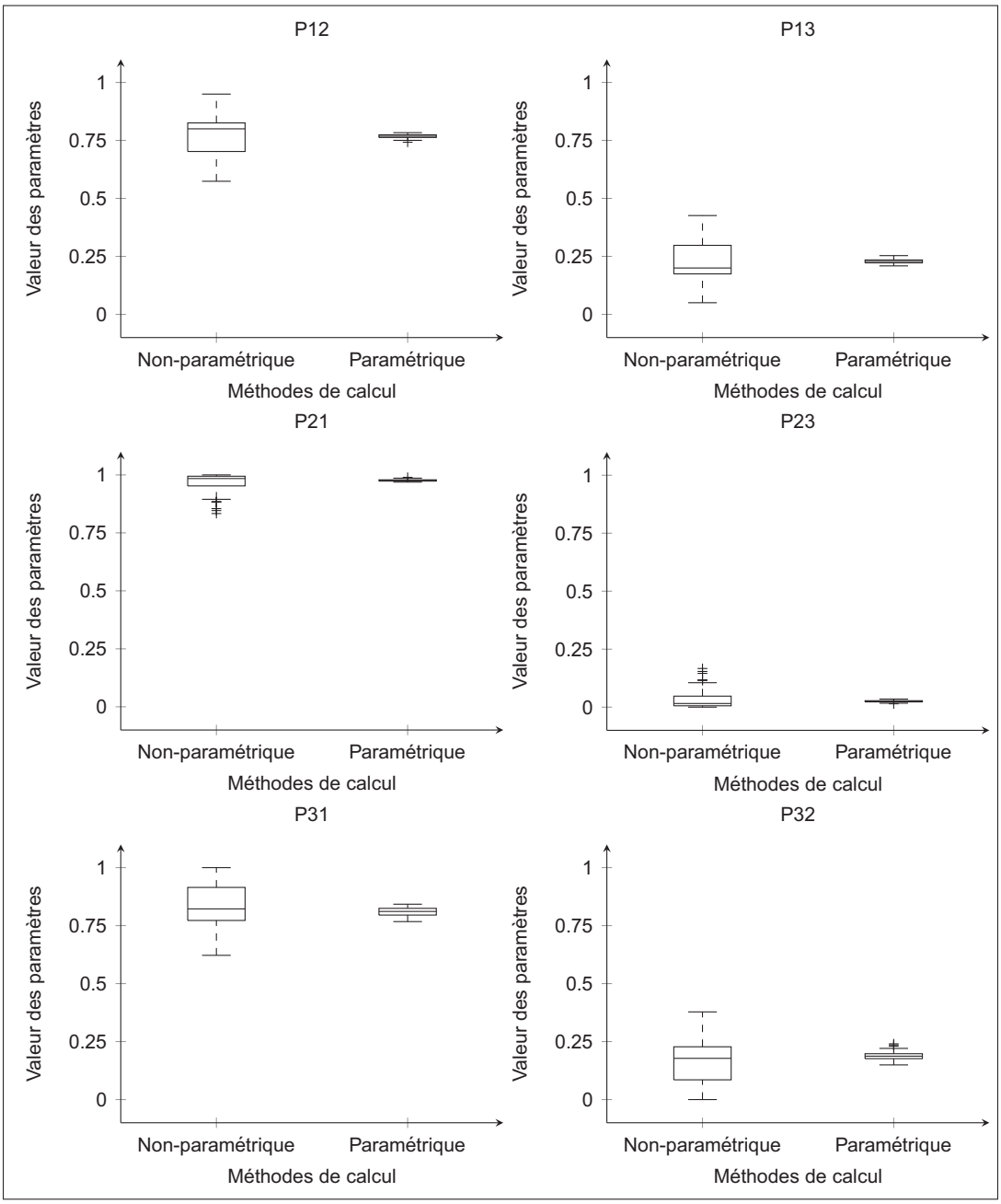


Figure 4.4 Différences d'incertitudes entre le calcul paramétrique et non-paramétrique

### 4.6 Perturbation non paramétrique

Normalement, la procédure veut que le calcul d'incertitude et les simulations soient faits en deux étapes distinctes, mais nous les imbriquerons. Nous calculerons les paramètres au fur et à mesure, à partir de fenêtres positionnées aléatoirement dans la série originale. De plus, rappelons-nous des techniques présentées dans la Section 2.1.6 qui permettent la prise en compte de la non-stationnarité : Carpinone *et al.* (2010) et Xue *et al.* (2000) mettent à jour la matrice de transition de leurs chaînes de Markov en cours de simulations. Inspiré de ceci, les valeurs des paramètres (et les distributions empiriques) ne changerons pas seulement entre les simulations, mais également en cours de simulations. Nous appelons cette mise à jour des paramètres au fil des simulations « perturbation non paramétrique » (PNP), nom emprunté à Kim et Valdes (2005). Ainsi, la génération d'une série temporelle synthétique de longueur $n$, se fera en $x$ morceaux de longueur $m$. Chaque morceau sera généré avec des valeurs de paramètres (et des distributions empiriques) différentes. Nous résumons le fonctionnement de la PNP avec la Figure 4.5. En a), nous avons les chaînes de Markov ou de Semi-Markov classiques. En b), les fenêtres ont une taille de 6 mois ; ainsi, pour produire une série de 2 ans il y aura 4 changements de paramètres. En c), la taille est inchangée, mais à la façon de Xue *et al.* (2000), le fonctionnement est périodique. En d), la taille de la fenêtre est de 3 mois au lieu de 6. Ces trois versions de la PNP sont trois exemples parmi de nombreuses possibilités.

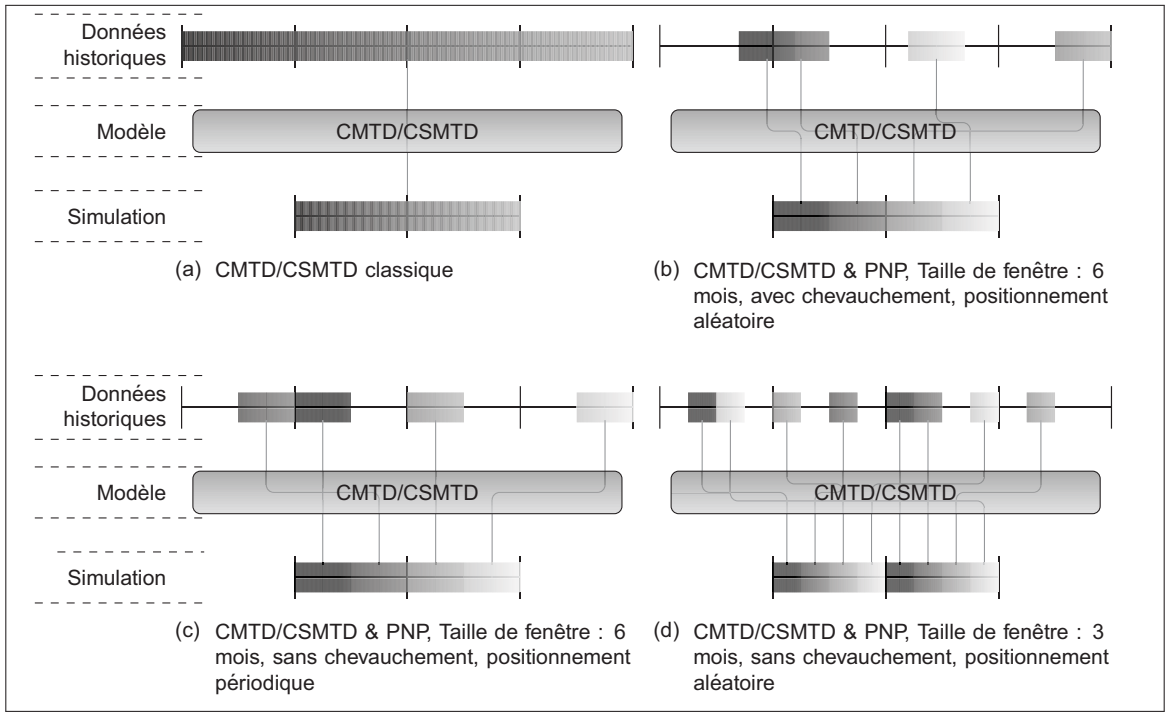


Figure 4.5 Prise en compte de l'incertitude des paramètres par une perturbation non paramétrique

## 4.7 Différences entre les modèles

Afin de bien comprendre les différences entre les trois modèles, et l'importance de la prise en compte de la non-stationnarité, nous avons mené une première étude comparative. Celle-ci a conduit à la rédaction d'un article technique et sa présentation lors de la conférence Hydrovision International en 2011. Nous comparons les modèles avec les critères d'évaluation décrits dans la Section 3.1, mais dans une procédure *in-sample*. Ceci va à l'encontre de ce que nous avons conseillé dans le chapitre précédent, car le but n'était alors pas de faire la validation de ces modèles. Nous utilisons les cinq années disponibles [2] pour générer des séries temporelles synthétiques (d'une longueur de cinq ans) avec les trois modèles présentés précédemment. Afin de montrer l'effet de la prise en compte de la non-stationnarité sur les performances des chaînes de Markov et Semi-Markov, nous les évaluerons dans trois configurations :

- La configuration *5/5* : la valeur des paramètres est calculée sur la globalité des données.
- La configuration *1/5* : utilisée pour illustrer les conséquences de considérer qu'une seule année peut être représentative du comportement des GTA.
- La configuration *1/1* : reprend le principe des PNP, en changeant la valeur des paramètres pour chaque année simulée.

Pour les configurations *1/5* et *1/1*, les paramètres sont calculés à partir des données contenues dans une fenêtre d'un an. Nous avons voulu tester si le degré de chevauchement de ces fenêtres pouvait avoir une influence sur les résultats. Pour cela nous autorisons la distance entre le début de chaque fenêtre à être égale à 30 jours, 6 jours et 1 jour. Nous évaluons les performances du BBM avec plusieurs tailles de bloc : 1 an, 30 jours, 6 jours et 1 jour. Les modèles ainsi comparés sont au nombre de 18, dont nous résumons les caractéristiques dans les Tableaux 4.1 et 4.2. Nous présentons sous forme de diagrammes à moustaches les performances [3] des 18 modèles suivant les deux critères décrits dans le chapitre précédent. Ces diagrammes représentent la dispersion des résultats entre les différentes simulations réalisées. Nous rappelons que le critère 1 concerne le nombre total de changements d'état et que le critère 2 concerne le taux d'évolution (Figure 4.6). Selon le critère 1 (Figure 4.6a), si l'on compare les chaînes de Markov aux chaînes de Semi-Markov avec des configurations identiques (le modèle #1 avec le modèle #8, #2 avec #9, #5 avec #11, etc.), nous constatons que les différences sont faibles. De plus, nous pouvons voir l'influence de la taille de bloc sur les performances du BBM (modèles #15 à #18), la diminution de celle-ci conduit à une moindre dispersion entre les simulations. Le critère 2 nous permet de constater les différences entre les CMTD et les CSMTD, si nous reprenons cette comparaison configuration à configuration, nous pouvons voir que globalement

2. À l'époque de la réalisation de l'étude, seules cinq années complètes avaient été récupérées
3. Seuls les résultats pour les baisses de puissance et la marche normale sont présentés, car suffisant pour comprendre le fonctionnement des modèles.

Tableau 4.1 Caractéristiques des chaînes de Markov et Semi-Markov testées

| # du modèle | Type de modèle | Type de Configuration | Degré de chevauchement |
|---|---|---|---|
| 1 | Chaînes de Markov | *5/5* | 1825 |
| 2 | | *1/5* | 30 |
| 3 | | *1/5* | 6 |
| 4 | | *1/5* | 1 |
| 5 | | *1/1* | 30 |
| 6 | | *1/1* | 6 |
| 7 | | *1/1* | 1 |
| 8 | Chaînes de Semi-Markov | *5/5* | 1825 |
| 9 | | *1/5* | 30 |
| 10 | | *1/5* | 6 |
| 11 | | *1/5* | 1 |
| 12 | | *1/1* | 30 |
| 13 | | *1/1* | 6 |
| 14 | | *1/1* | 1 |

Tableau 4.2 Caractéristiques des Bootstrap à blocs mobiles testés

| # du modèle | Type de modèle | Taille de bloc |
|---|---|---|
| 15 | Bootstrap à blocs mobiles | 365 jours |
| 16 | | 30 jours |
| 17 | | 6 jours |
| 18 | | 1 jour |

les CMTD présentent des erreurs supérieures aux CSMTD (Figure 4.6b et 4.6c). Ces résultats ne sont pas surprenants, les performances des CMTD sont nettement inférieures au CSMTD au niveau des FdR des temps de maintien, car les CMTD utilisent des distributions exponentielles, alors que ce n'est pas la réalité des données. Ceci est évident si l'on pense à l'opération des GTA, ceux-ci ne sont pas démarrés pour rester en marche normale durant un temps très court, or c'est aux temps très courts que la distribution exponentielle accorde le plus de probabilité. Dans une moindre mesure, les CSMTD sont également supérieurs pour les FdR du nombre de changements d'état par jour. Comme pour les deux modèles, le degré de chevauchement ne semble avoir que très peu d'influence, il sera fixé à 30 jours. Concernant les trois

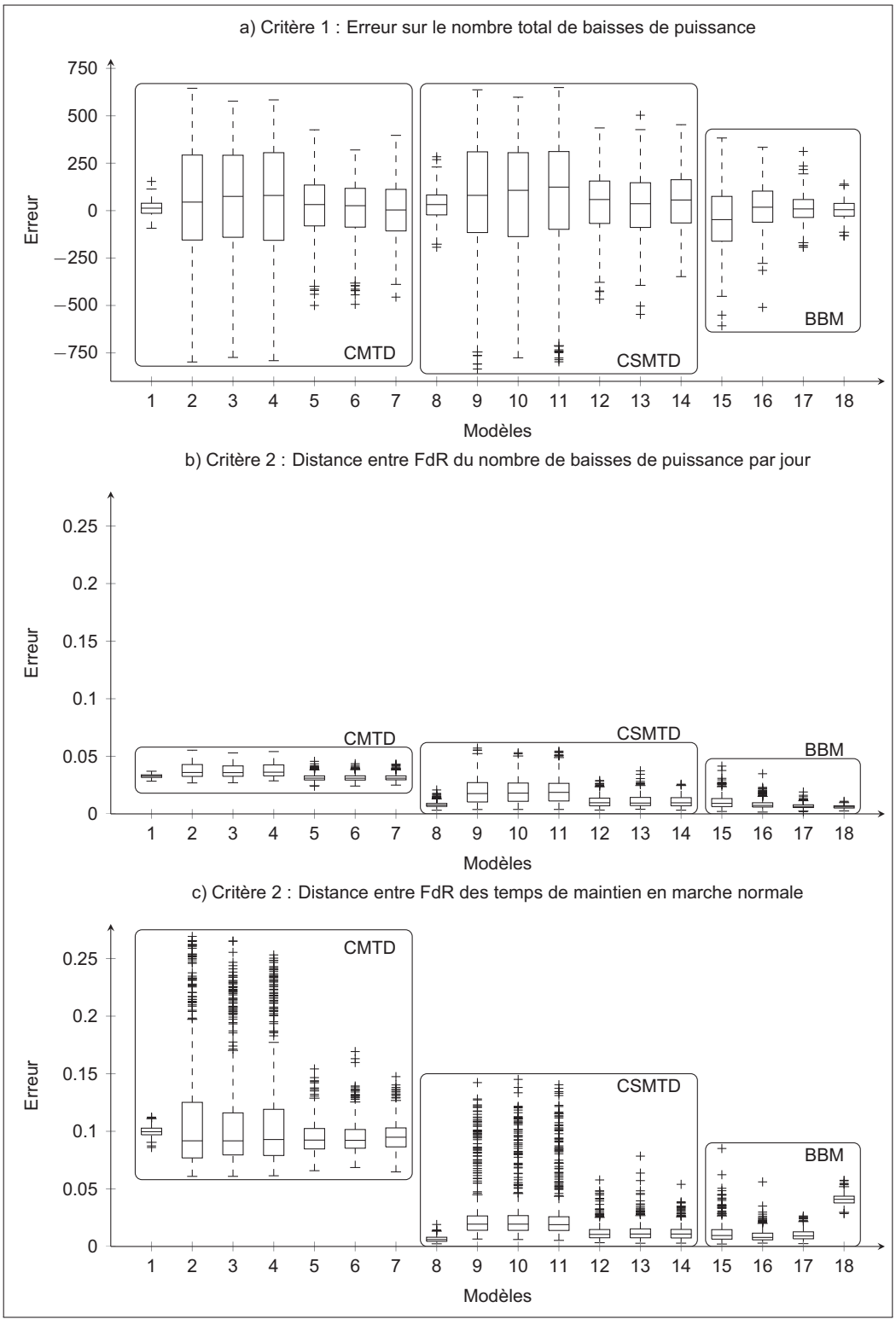


Figure 4.6 Performance des modèles

types de configurations pour CMTD et CSMTD, il est clair que la configuration *1/5* n'est pas recommandable : une seule année de données ne saurait être représentative de l'opération du GTA. La configuration *5/5* semble être la plus précise, ce qui n'est pas surprenant, car celle-ci considère que le comportement réel est un comportement moyennant les différences d'une année à l'autre. Ceci peut être problématique dans le cas de prédiction. Finalement, la configuration *1/1* amène à de plus grandes variations que la configuration *5/5* et lorsqu'elle est appliquée au CSMTD, elle présente des résultats similaires au BBM. Concernant le BBM, que ce soit suivant le critère 1 ou le critère 2, la réduction de la taille de bloc entraîne une augmentation de ses performances. Néanmoins, avec une taille de bloc de 1 jour, celles-ci se dégradent pour les temps de maintien (modèle #18 dans la Figure 4.6c), ce qui peut indiquer qu'à partir de cette taille la dépendance temporelle est brisée. De plus le BBM avec un bloc de 365 jours montre des performances équivalentes à celles des CSMTD dans la configuration*1/1*.

La Figure 4.7 permet de visualiser les différences de comportement entre les chaînes de Markov et les chaînes de Semi-Markov. Nous y comparons les graphes de taux d'évolution de séquences simulées à ceux provenant de séquences observées [4]. Observons les figures 4.7a et 4.7c qui représentent les graphes de taux d'évolution provenant de séquences générer avec des CMTD dans les configurations *5/5* et *1/5* respectivement. Nous constatons que ces graphes sont linéaires, ceci signifie que le taux d'évolution est constant, et donc, que les temps de maintien suivent une distribution exponentielle. Alors que dans ces deux configurations, les CSMTD (figures 4.7b et 4.7d) présentent des variations dans le taux d'évolution, ce qui semble être plus fidèle au comportement réel. En configuration *1/1*, la chaîne de Markov n'est plus stationnaire (le taux d'évolution n'est plus constant) et les différences entre CMTD et CSMTD s'amenuisent. Même si cela ne se répercute pas sur les critères, nous pouvons faire l'hypothèse que plus la taille de fenêtre diminue, moins l'approximation exponentielle n'a d'importance. Nous pouvons penser que le type de FdR qui décrit les temps de maintien a finalement peu d'impact sur le nombre total de changements d'état, mais en a sur la répartition dans le temps de ces changements d'état.

4. Nous n'indiquons pas les valeurs réelles, mais l'échelle est définie par le nombre total de BP observé (*Obs*), les nombres de BP maximum ($Max_{1/1}$) et minimum ($Min_{1/1}$) obtenus par simulation des chaînes de Semi-Markov dans la configuration *1/1*.

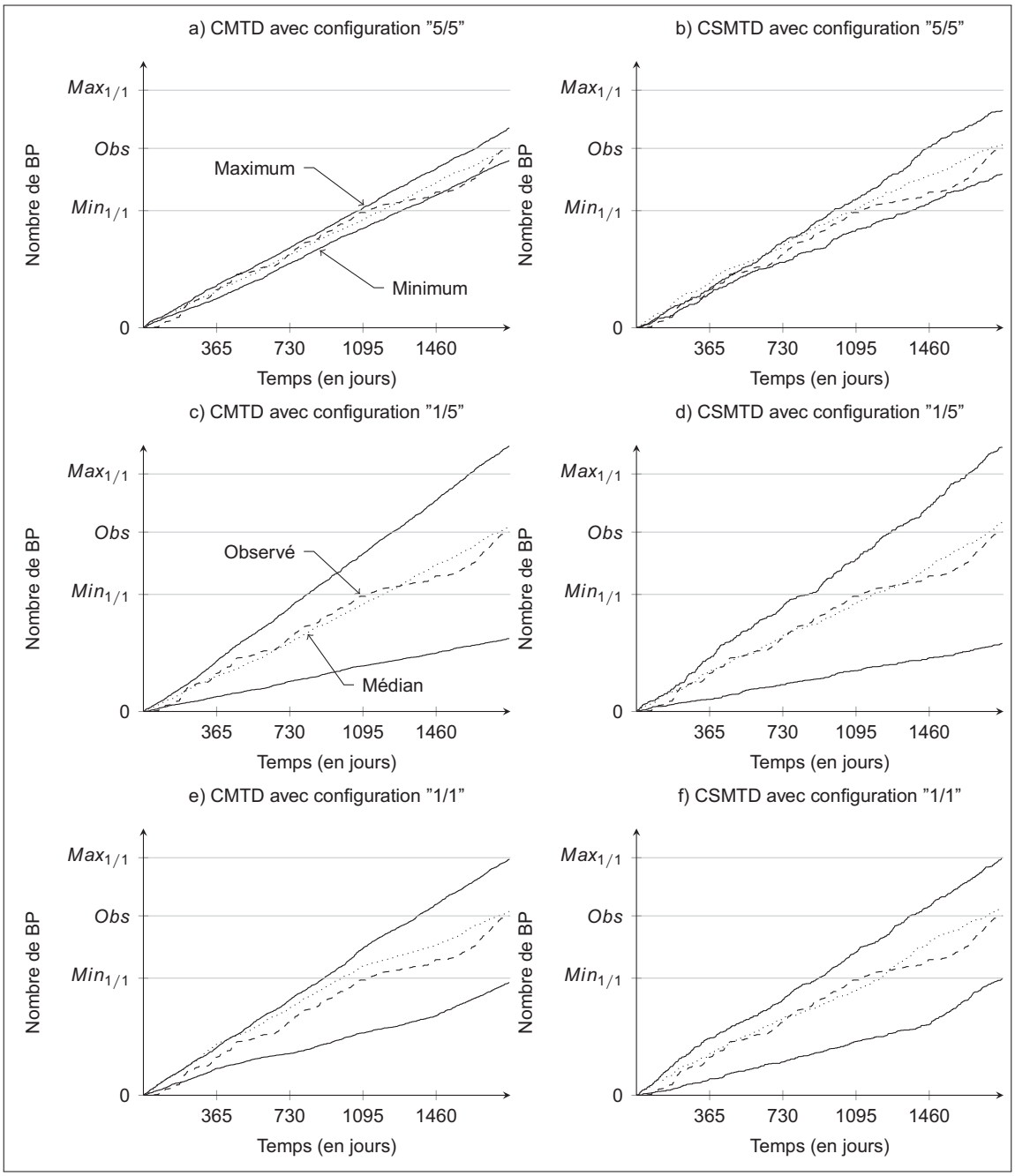


Figure 4.7 Différence de taux d’évolution entre les chaînes de Markov et de Semi-Markov

## 4.8 Conclusion du chapitre 4

Nous avons vu dans ce chapitre comment construire et utiliser le Bootstrap à bloc mobile, les chaînes de Markov et les chaînes de Semi-Markov. Ces modèles sont assez simples et l’on peut facilement en comprendre le fonctionnement, car ils restent en contact avec la réalité physique

du problème. Les chaînes de Markov semblent moins bien adaptées à notre problème, principalement car les temps de maintien doivent être exponentiellement distribués. Toutefois, il serait pertinent de mener une étude sur l'impact réel de ces différences de taux d'évolution sur la propagation de fissure. En l'absence de cette information, nous préférons les considérer comme important. Nous avons également détaillé la prise en compte de la non-stationnarité des données, qui est réalisée en changeant la valeur des paramètres (et les distributions empiriques) en cours de simulation. Nous avons montré l'impact sur les résultats si celle-ci n'est pas correctement prise en compte. Nous présentons dans le chapitre suivant la validation des trois modèles présentés ici.

# CHAPITRE 5

# RÉSULTATS ET ANALYSE

Le but du présent chapitre est de déterminer quel modèle est le plus recommandable pour générer des séquences de chargement qui pourront être utilisées comme intrant dans un modèle de fatigue. Nous avons testé les modèles avec des données de plusieurs GTA qui proviennent de différentes centrales afin de nous assurer qu'ils peuvent être appliqués à toute sorte de profils d'opération. Nous évaluerons les modèles selon la procédure décrite dans la Section 3.2. Les modèles seront comparés en analysant de façon approfondie les résultats obtenus à partir des données d'opérations de deux GTA : le GTA 19 de la centrale X [1] et le GTA 5 de la centrale Y. Par la suite, nous ferons référence à ces GTA sous les noms X19 et Y5 respectivement.

## 5.1 Simulations effectuées

Nous comparons cinq types de modèles : les versions classiques (stationnaires) des chaînes de Markov et Semi-Markov, les versions non stationnaires de ces chaînes (avec l'ajout de perturbations non paramétriques) et le Bootstrap à bloc mobile (avec plusieurs tailles de blocs). Ces modèles sont répertoriés dans le Tableau 5.1.

Comme nous l'avons expliqué dans la Section 4.5 nous évaluons les incertitudes des modèles par l'intermédiaire de simulations de Monte-Carlo. Nous devons déterminer le nombre minimum de simulations à effectuer pour assurer une stabilité des résultats. S'il est trop petit, le risque est de ne pas capter toute la variabilité des modèles et d'avoir une vision tronquée de leurs incertitudes. S'il est trop grand, le seul inconvénient serait de consommer plus de ressources que nécessaire. Une des méthodes est de suivre l'évolution de cette variabilité en fonction du nombre de simulations. Il existe un palier au-dessus duquel il n'y a plus de changements significatifs de la dispersion. Ce palier correspond au nombre optimal de simulations. La Figure 5.1 présente l'écart entre le 1er et le 99e percentile du nombre de BP par an. Comme on peut le voir, cet écart fluctue peu, et n'augmente pas, avec l'accroissement de 50 à 2800 simulations. Afin de limiter le risque de travailler avec un trop petit nombre de simulations, nous avons alors décidé arbitrairement d'en effectuer 400.

## 5.2 Étude de cas

Nous analysons en détail les performances des modèles lorsqu'ils sont utilisés avec les données d'opérations de X19 et de Y5. Les caractéristiques de fonctionnement de ces deux GTA sont présentées avec les Figures 5.2 et 5.3. La centrale X est du type « au fil de l'eau », ce qui

1. Il s'agit du GTA dont nous nous sommes servis depuis le début de ce mémoire.

Tableau 5.1 Caractéristiques des modèles comparés par validation croisée

| # du modèle | Type de modèle | Taille de fenêtre |
|---|---|---|
| #1 | CSMTD | - |
| #2 | CMTD | - |
| #3 | CSMTD & PNP | 365 |
| #4 | | 180 |
| #5 | | 90 |
| #6 | | 60 |
| #7 | | 30 |
| #8 | CMTD & PNP | 365 |
| #9 | | 180 |
| #10 | | 90 |
| #11 | | 60 |
| #12 | | 30 |
| #13 | BBM | 180 |
| #14 | | 90 |
| #15 | | 60 |
| #16 | | 30 |
| #17 | | 6 |

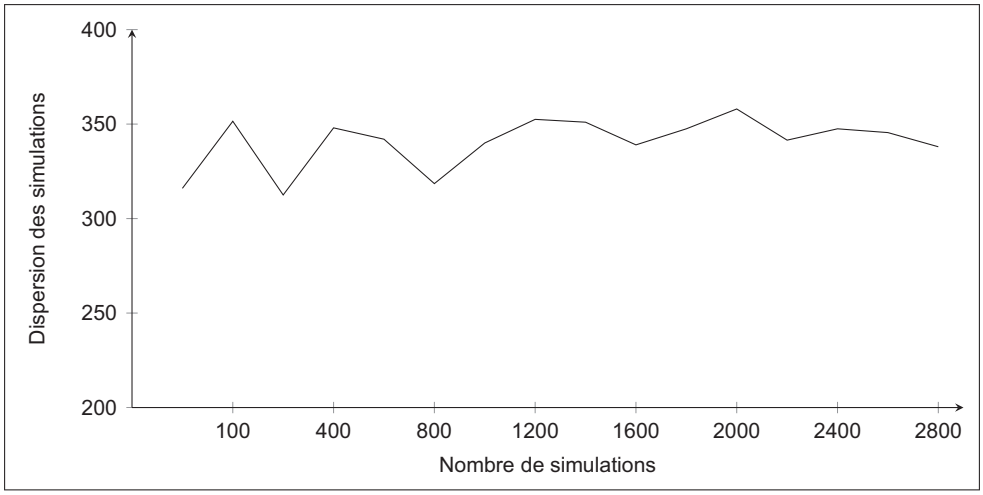


Figure 5.1 Dispersion des résultats en fonction du nombre de simulations

explique que le facteur d'utilisation ($\frac{Temps\ en\ MN}{Temps\ en\ A}$) soit très élevé. En effet, toute eau non turbinée est définitivement perdue. De plus, ses GTA peuvent se synchroniser sur plusieurs réseaux

électriques. La manœuvre de changement de réseau nécessite un passage en marche à vide, ce qui conduit à un grand nombre de baisses et de montées en puissance. Quant à la centrale Y, elle est à réservoir : ses GTA sont donc plus faciles à arrêter sans risque de perdre de l'énergie potentielle. Ce amène un facteur d'utilisation un peu moins élevé. Comme les arrêts ne sont pas plus nombreux, ils sont donc plus longs. De plus, les changements de réseaux n'étant pas possibles, le nombre de BP (et de MP) est quasi inexistant. La Figure 5.2 nous permet de déduire que la chute du nombre de BP dans l'année 4 n'est pas due à un arrêt de longue durée, mais à un comportement atypique. Comme rien ne nous permet d'affirmer que ce type de comportement ne se reproduira pas, nous devons garder l'année 4 intégrée dans nos données d'évaluation. De son côté, la chute du nombre de BP dans l'année 7 est due à un arrêt de longue durée. Afin de ne pas fausser le modèle, nous retirons la partie des données correspondant à cet arrêt : pour le bloc 7 de la validation croisée, seuls 275 jours seront simulés. Pour Y5, la Figure 5.3 montre que l'année 2 est atypique, car malgré un arrêt de longue durée, le nombre de PhA reste élevé. Rien ne nous permet d'affirmer que le comportement à l'année 2 ne peut pas se reproduire, elle sera donc prise en compte. Toutefois, nous retirerons la partie qui renferme l'arrêt supérieur à 1 mois. Par conséquent, pour le bloc 2 de la validation croisée, seuls 328 jours seront simulés.  Nous avons réalisé à la section 1.3.2 une analyse des données de

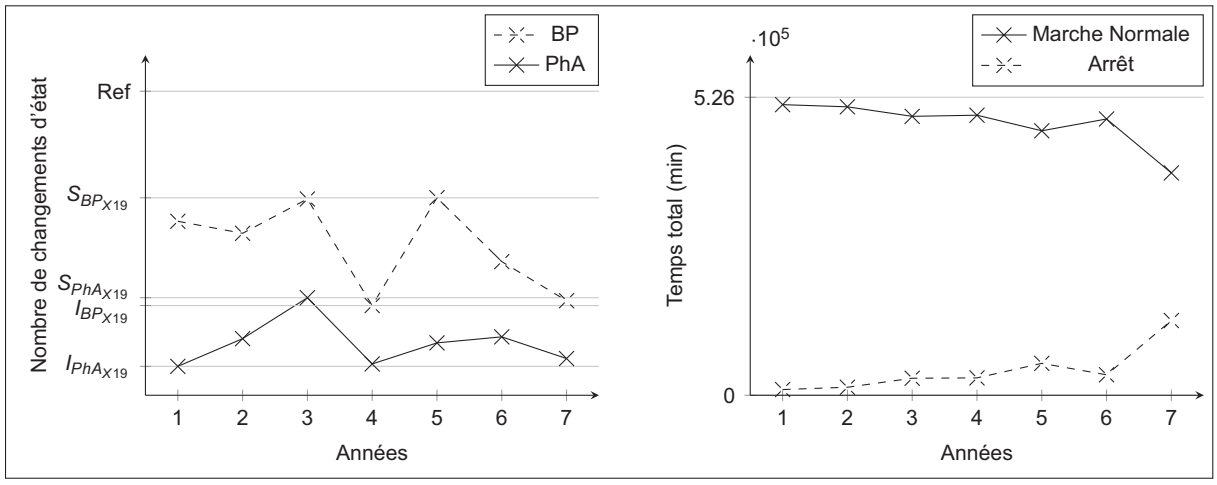


Figure 5.2  Fonctionnement historique de X19

fonctionnement de X19. Celle-ci a conclu que les données ne sont pas stationnaires, qu'il n'y a pas de tendance et que nous ne pouvons utiliser aucune périodicité pour nous aider à prévoir l'évolution du nombre de changements d'état. Nous devons faire la même analyse pour Y5. Tout d'abord, avec le taux d'évolution du nombre de PhA par jour (Figure 5.4), nous constatons que le fonctionnement du groupe est non-stationnaire (car le taux d'évolution n'est pas linéaire). La Figure 5.5a montre qu'il n'y pas de tendance et avec la Figure 5.5b nous constatons trois fréquences qui semblent prépondérantes, dont une à 6 mois et une autre à 12. Ces

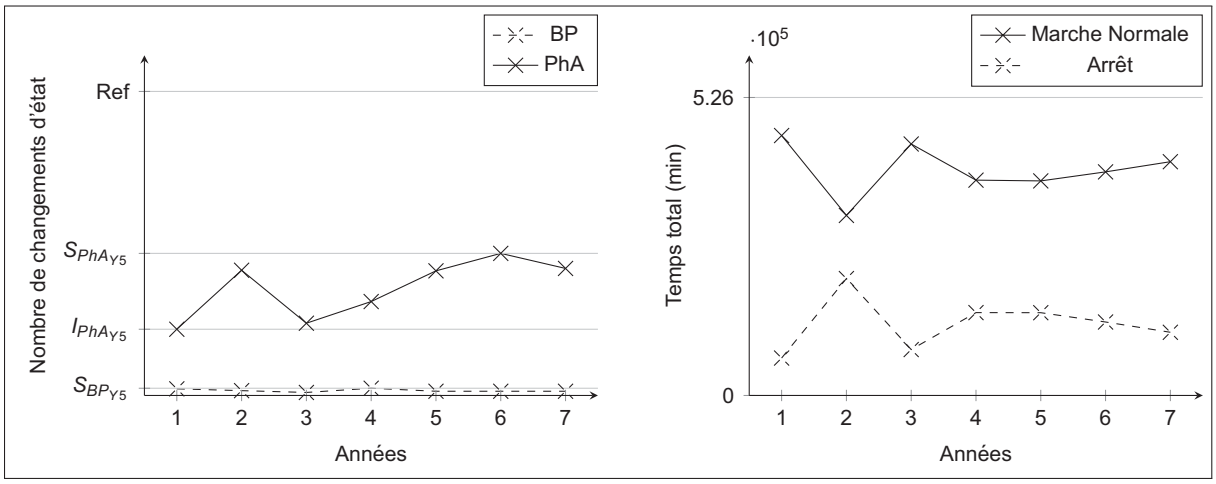


Figure 5.3 Fonctionnement historique de Y5

périodes ne nous permettent pas de prédire le nombre de changements d'état dans les années futures, en effet elles indiquent uniquement la succession de points hauts et bas au sein d'une année, et ne renseignent pas sur leurs valeurs. Pour la suite, nous préférons considérer que ces variations sont aléatoires.

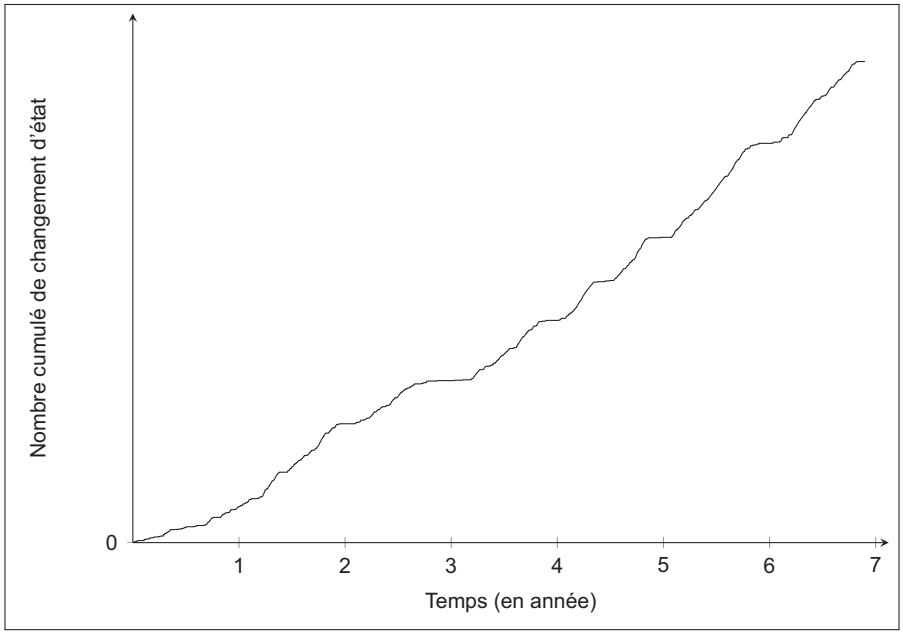


Figure 5.4 Taux d'évolution sur 7 ans du nombre de PhA (Y5)

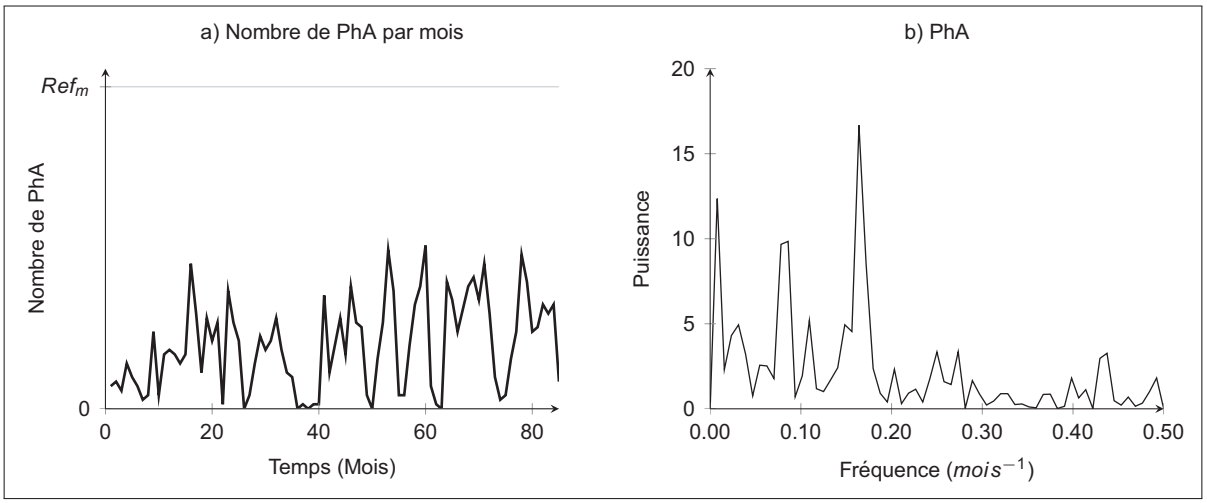


Figure 5.5 Analyse des données de fonctionnement de Y5

### 5.2.1 Résultats pour le critère 1

Les Figures 5.6 et 5.7 présentent les erreurs de généralisation pour le premier critère d'évaluation, c'est-à-dire pour le nombre total de changements d'état. Pour Y5, nous n'incluons pas les résultats de la MAPE associée au nombre de BP car à cause du faible nombre de BP la moindre erreur conduira à une MAPE disproportionnée. Par exemple, sur un nombre réel de 3 BP, si le modèle en simule 6, la MAPE sera de 200%, alors que cette différence n'aura pratiquement aucun impact sur la propagation de fissure. Les résultats vont dans le sens de l'analyse effectuée à la Section 4.7. En ce qui concerne le nombre total de changements d'état, la différence (à taille de fenêtre/bloc égale) entre CSMTD & PNP, CMTD & PNP et BBM est relativement faible. Ceci est vrai autant pour X19 que pour Y5. De plus, nous pouvons émettre deux remarques communes à X19 et Y5 : les performances s'améliorent avec la diminution de la taille de fenêtre, et les versions classiques des CSMTD et CMTD sont meilleures que leur pendant avec PNP.

Nous pouvons nous interroger sur la qualité de modèles qui ont une MAPE (voir la section 3.1.1) aux alentours de 40% pour le nombre de BP (avec les données de X19) et autour de 40% pour le nombre de PhA (avec les données de Y5). Ces résultats sont à mettre en perspective avec la variation propre aux données. Nous calculons l'écart entre le nombre de BP (et de PhA) dans chaque ensemble de validation et la moyenne annuelle du nombre de BP (et de PhA) de l'ensemble d'entraînement correspondant [2]. Nous obtenons alors une dispersion moyenne (voir les Tableaux 5.2 et 5.3). Celle-ci représente l'écart moyen entre chaque année et le reste des données. En nous rappelant que les variations interannuelles sont aléatoires et que

2. Nous ne prenons pas en considération l'année 7 pour les données de X19, car elle n'est pas complète et faussera la moyenne annuelle.

par conséquent, nous ne pouvons prédire avec exactitude leur évolution. Nous comprenons que les erreurs sont principalement dues aux écarts entre les ensembles de validation et ceux d'entraînement.

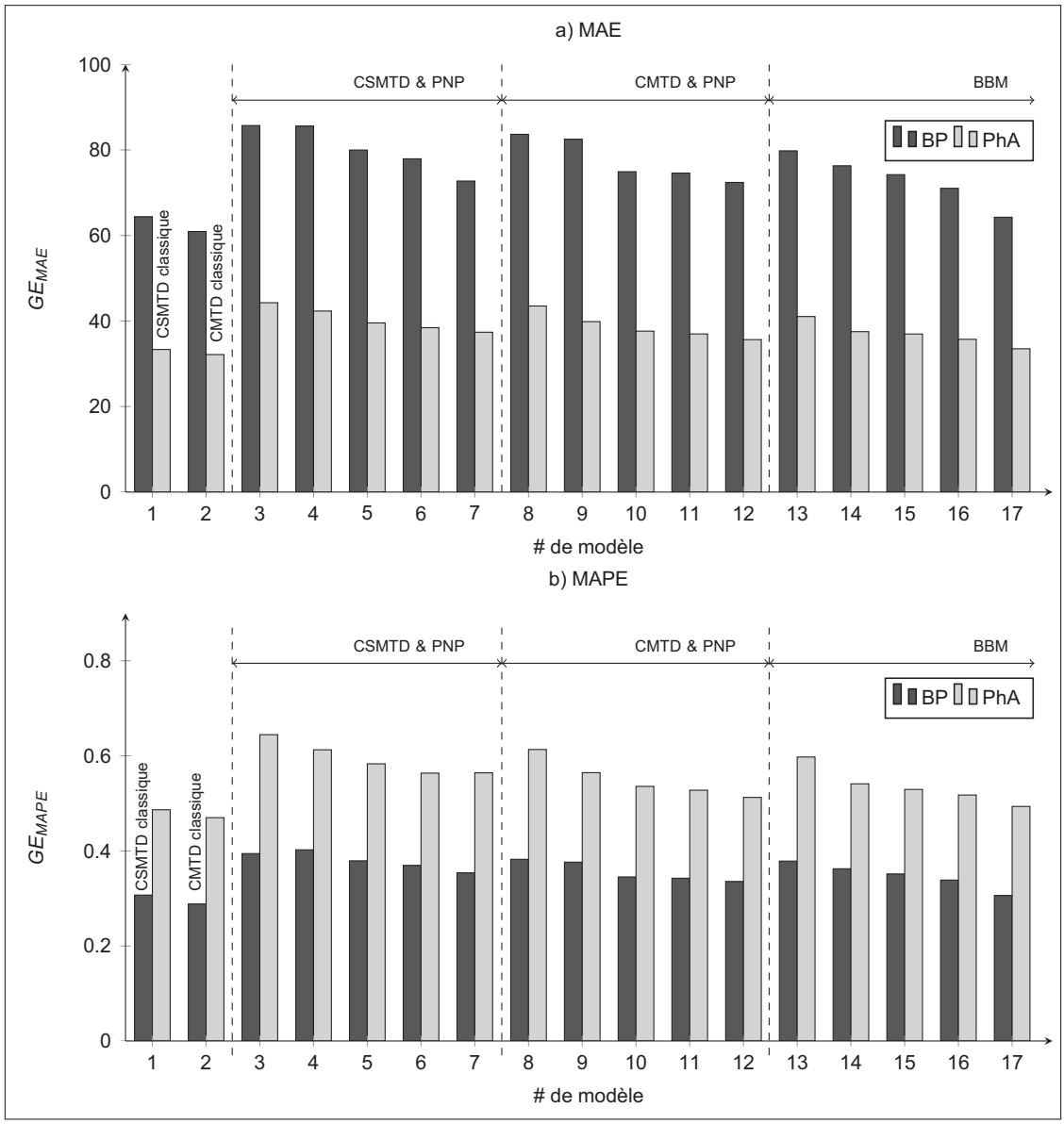


Figure 5.6 Résultat de l'erreur de généralisation pour le critère 1 (X19)

### 5.2.2 Résultats pour le critère 2

Les Figures 5.8 et 5.9 donnent les performances des modèles selon le critère 2, c'est-à-dire le taux d'évolution. Ce critère permet de voir la différence entre CSMTD et CMTD, c'est aussi

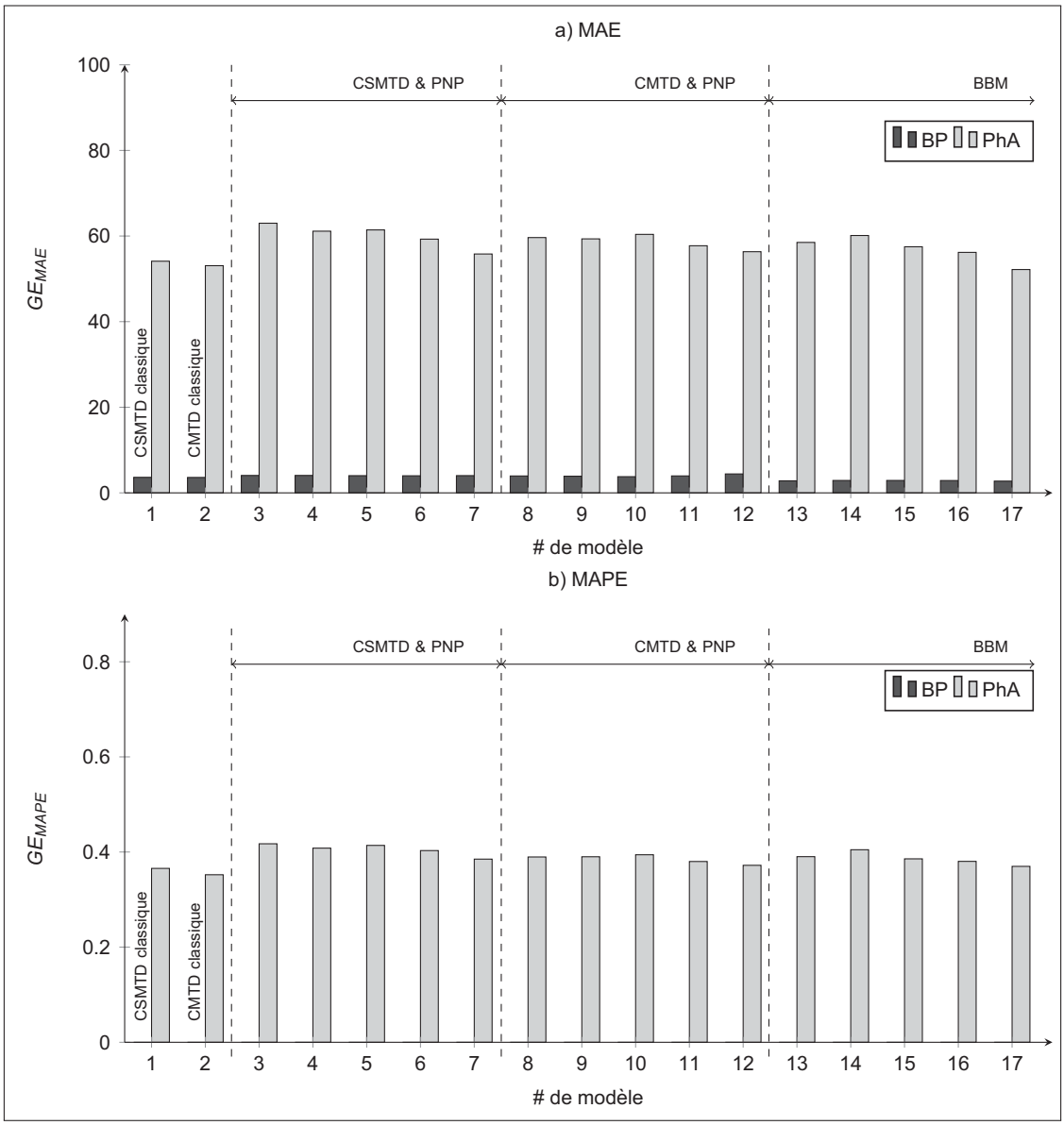


Figure 5.7 Résultat de l'erreur de généralisation pour le critère 1 (Y5)

Tableau 5.2 Dispersion du nombre de changements de régime par an (X19)

| Changement de régime | Dispersion inter-annuelle | | | | | | Dispersion moyenne |
|---|---|---|---|---|---|---|---|
| | An 1 | An 2 | An 3 | An 4 | An 5 | An 6 | |
| BP | 11.9% | 2.4% | 30.3% | 48.9% | 31.3% | 18.8% | 23.9% |
| PhA | 53.9% | 5.7% | 122.4% | 49.2% | 4.2% | 10% | 40.9% |

Tableau 5.3 Dispersion du nombre de changements de régime par an (Y5)

| | Dispersion inter-annuelle | | | | | | | Dispersion |
|---|---|---|---|---|---|---|---|---|
| | **An 1** | **An 2** | **An 3** | **An 4** | **An 5** | **An 6** | **An 7** | **moyenne** |
| PhA | 43.1% | 20.6% | 37.1% | 14.7% | 19.9% | 41.1% | 22.7% | 28.4% |

avec lui que l'influence des données sur les résultats des modèles se distingue le plus. Avec les données de X19, les CSMTD & PNP et le BBM sont nettement meilleurs que les CMTD & PNP pour les deux indicateurs du critère 2, tandis qu'avec les données que Y5, cette différence apparaît seulement au niveau des temps de maintien. Ceci n'est pas dû à de moins bonnes performances des CSMTD & PNP et du BBM, mais bien à de meilleurs résultats des CMTD & PNP. Nous n'avons pas d'explication précise sur la cause, mais cela signifie que, pour les données de Y5, les écarts sur les FdR de temps de maintien ont moins d'influence sur la répartition du nombre de changements d'état par jour.

Comme pour le critère 1, nous remarquons que la diminution de la taille de fenêtre amène l'amélioration des performances, et que les versions classiques des CMSTD et CMTD sont plus précises que leur pendant avec PNP. Similairement, nous pouvons mettre les valeurs des erreurs de généralisation du critère 2 en parallèle avec la dispersion propres aux données. Nous regroupons dans les Tableaux 5.4 et 5.5 les distances entre FdR des ensembles de validation et d'entraînement pour X19 et Y5 respectivement. Nous concluons également que les résultats sont en grande partie dus à la variation entre années de fonctionnement. Globalement, les erreurs de généralisation du critère 2 sont plus élevées pour Y5, car la dispersion des données est plus grande. Avec les données de d'opération de Y5, nous notons une importante amélioration des performances des CMTD en ce qui a trait au FdR des temps de maintien en arrêt. Il est vrai que la dispersion interannuelle est de 0.134 pour X19, contre 0.095 pour Y5, mais cela ne suffit pas pour expliquer une erreur de généralisation comprise entre 0.5 et 0.8 lorsque les CMTD sont appliquées aux données de X19. Cette amélioration avec les données de Y5 indique probablement que pour X19, les FdR sont très éloignées d'une distribution exponentielle, alors que pour Y5 l'approximation exponentielle est moins fausse.

### 5.2.3 Analyse par ensemble de validation

D'après l'étude des performances des modèles basée sur les erreurs de généralisation des deux critères d'évaluation, les CSMTD classiques sont les meilleurs modèles. Toutefois, comme nous travaillons avec des données non stationnaires, il est recommandé d'analyser les performances des modèles dans chaque ensemble de validation. Les erreurs de généralisation sont utiles pour condenser les résultats, mais elles ne permettent pas d'évaluer en détail les performances

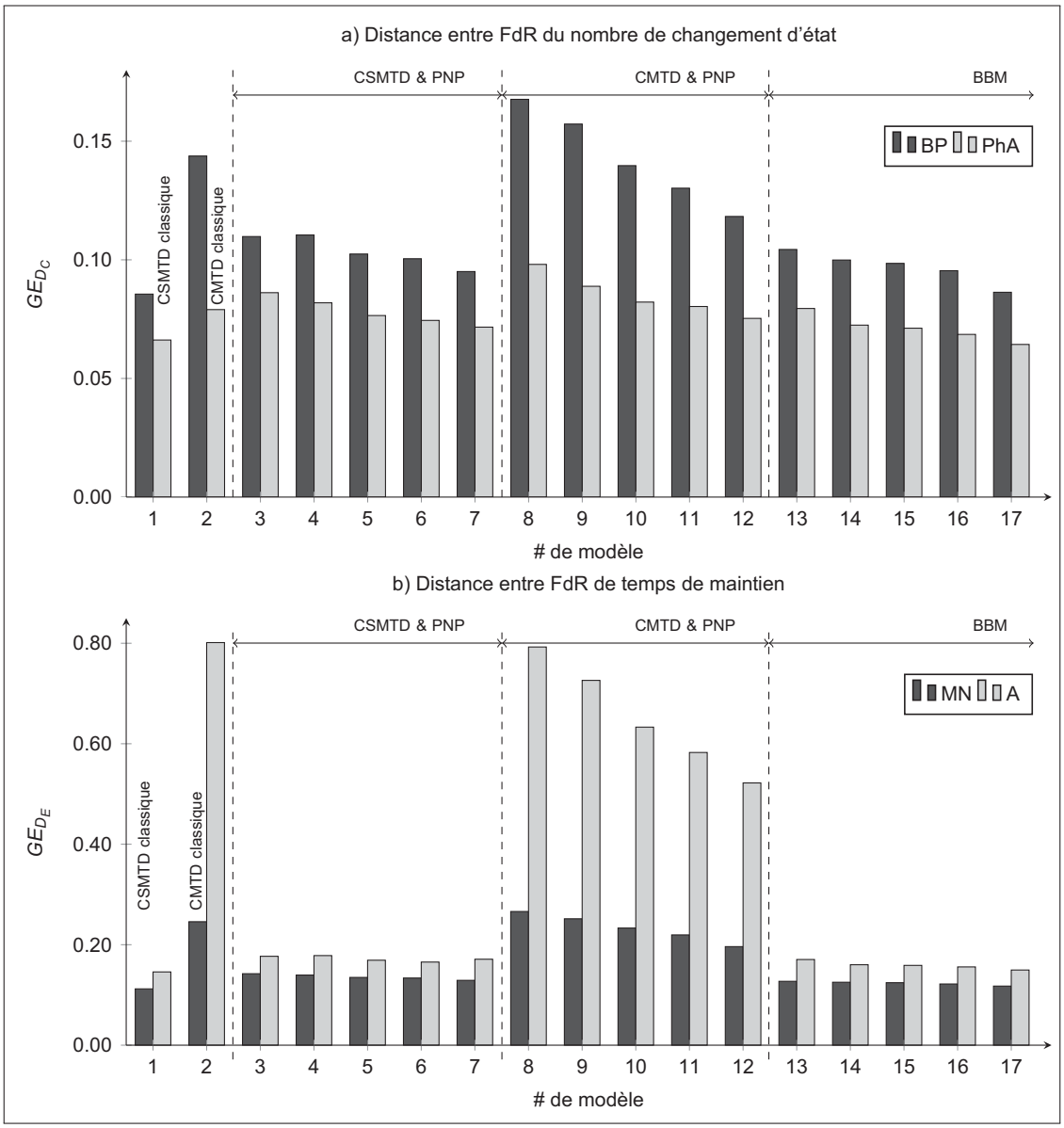


Figure 5.8 Résultat de l'erreur de généralisation pour le critère 2 (X19)

des modèles. Nous ne reprendrons pas les deux critères, mais directement le nombre total de changements d'état. Le but est de nous assurer que pour chaque ensemble de validation, les simulations intègrent la réalité observée. Pour cela nous nous servons des Figures 5.10 à 5.15 qui contiennent les boîtes à moustache du nombre simulé de changements d'état, le nombre observé de BP et PhA pour chacun des ensembles de validation en trait plein et, en pointillé, le nombre moyen par an de BP et PhA observé dans chacun des ensembles d'entraînement. Pour les résultats obtenus à partir des données de X19 (Figures 5.10 à 5.13), le septième ensemble de validation n'est pas inclus car l'année 7 n'est pas complète et fausse les moyennes annuelles. Les Figures 5.14 et 5.15 présentent donc les résultats provenant des données de Y5.

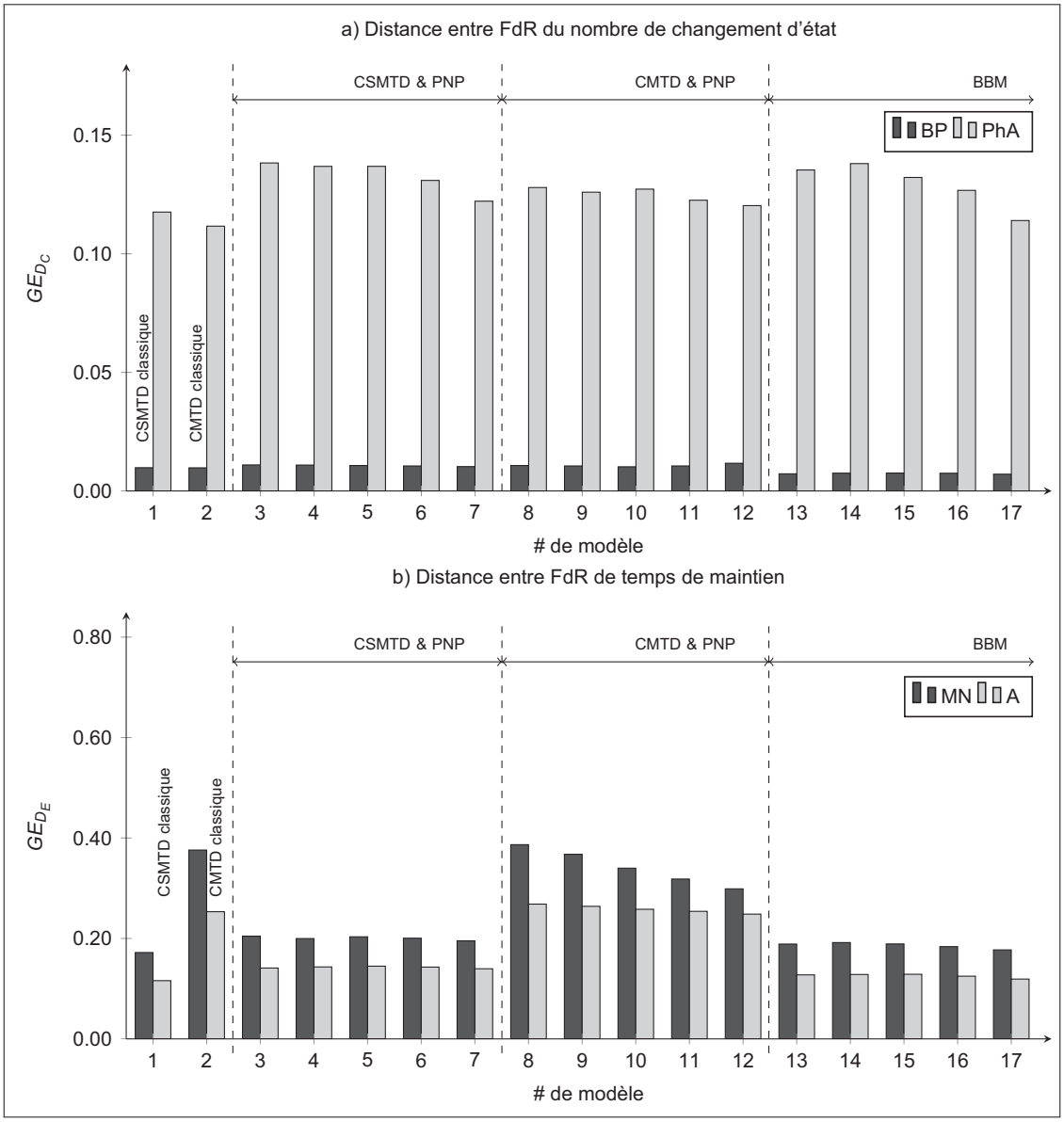


Figure 5.9 Résultat de l'erreur de généralisation pour le critère 2 (Y5)

Tableau 5.4 Dispersion entre FdR (X19)

| | **Dispersion inter-annuelle** | | | | | | | **Dispersion** |
|---|---|---|---|---|---|---|---|---|
| | **An 1** | **An 2** | **An 3** | **An 4** | **An 5** | **An 6** | **An 7** | **moyenne** |
| BP | 0.045 | 0.030 | 0.088 | 0.161 | 0.082 | 0.051 | 0.059 | 0.074 |
| PhA | 0.082 | 0.021 | 0.165 | 0.067 | 0.013 | 0.043 | 0.020 | 0.059 |
| MN | 0.097 | 0.060 | 0.065 | 0.140 | 0.069 | 0.065 | 0.216 | 0.102 |
| A | 0.224 | 0.098 | 0.132 | 0.201 | 0.041 | 0.154 | 0.091 | 0.134 |

Tableau 5.5 Dispersion entre FdR (Y5)

| | Dispersion inter-annuelle | | | | | | | Dispersion |
|---|---|---|---|---|---|---|---|---|
| | **An 1** | **An 2** | **An 3** | **An 4** | **An 5** | **An 6** | **An 7** | **moyenne** |
| BP | 0.007 | 0.016 | 0.029 | 0.006 | 0.023 | 0.023 | 0.022 | 0.018 |
| PhA | 0.171 | 0.122 | 0.152 | 0.050 | 0.057 | 0.147 | 0.096 | 0.114 |
| MN | 0.303 | 0.132 | 0.240 | 0.102 | 0.130 | 0.107 | 0.109 | 0.160 |
| A | 0.184 | 0.063 | 0.130 | 0.090 | 0.062 | 0.065 | 0.069 | 0.095 |

Nous incluons l'année 2, bien qu'incomplète, parce que nous jugeons que le biais engendré sur la moyenne annuelle est négligeable. Seuls les modèles #1 (CSMTD classique), #5 (CSMTD & PNP avec une fenêtre de 3 mois), #10 (CMTD & PNP avec une fenêtre de 3 mois) et #14 (BBM avec une taille de bloc de 3 mois) sont présentés.

La Figure 5.10 permet de comprendre pourquoi le modèle #1 a les plus faibles erreurs de généralisation, mais aussi pourquoi nous ne recommandons pas son utilisation. En effet, comme les simulations sont obtenues à partir d'un modèle dont les paramètres varient très peu, la dispersion entre elles est moins importante. Les erreurs de généralisation sont plus faibles, car il n'y a pas de simulations très éloignées de l'ensemble de validation. Ce modèle est cependant incapable de produire des séquences avec un nombre de BP inférieur ou égal à celui observé dans l'année 4 et un nombre de PhA supérieur ou égal à celui de l'année 3. Donc oui, le modèle est plus précis, mais il conduit à une importante perte d'information en ne simulant pas tous les comportements de l'historique de fonctionnement. Ceci est vrai pour les données de X19, mais cette limite du modèle #1 n'est pas visible avec celles de Y5 (Figure 5.14a). Si nous reprenons les Tableaux 5.2 et 5.3, nous constatons que bien que la dispersion moyenne de Y5 est plus élevée que celle de X19, c'est pour X19 que les dispersions interannuelles les plus grandes existent (48.9% : écart du nombre de BP dans l'année 4 par rapport au reste, et 122.4% pour le nombre de PhA). S'il existe de trop grands écarts interannuels, le modèle #1 sera incapable de les prendre en compte. Nous suggérons fortement de n'utiliser ce modèle que dans le cas où les données ont été prouvées stationnaires.

Pour les trois autres modèles, que nous les appliquions aux données de X19 ou de Y5, ils sont capables d'intégrer dans leurs simulations la totalité des comportements. Exception faite, pour les données de X19, du nombre de PhA observé dans l'année 3. Dans ce cas, aucun modèle « n'attrape » le nombre réel lorsqu'il est entraîné avec le troisième ensemble (l'ensemble dont l'année 3 est exclue). Les modèles reproduisent les variations internes aux données d'entraînement. Ainsi, si l'année à prédire a un comportement trop différent de l'historique, aucun modèle n'est capable de le prévoir, ce qui est le cas ici avec un écart de 122.4%. Par contre, ils sont capables de l'inclure lorsque l'année 3 fait partie des données d'entraînement.

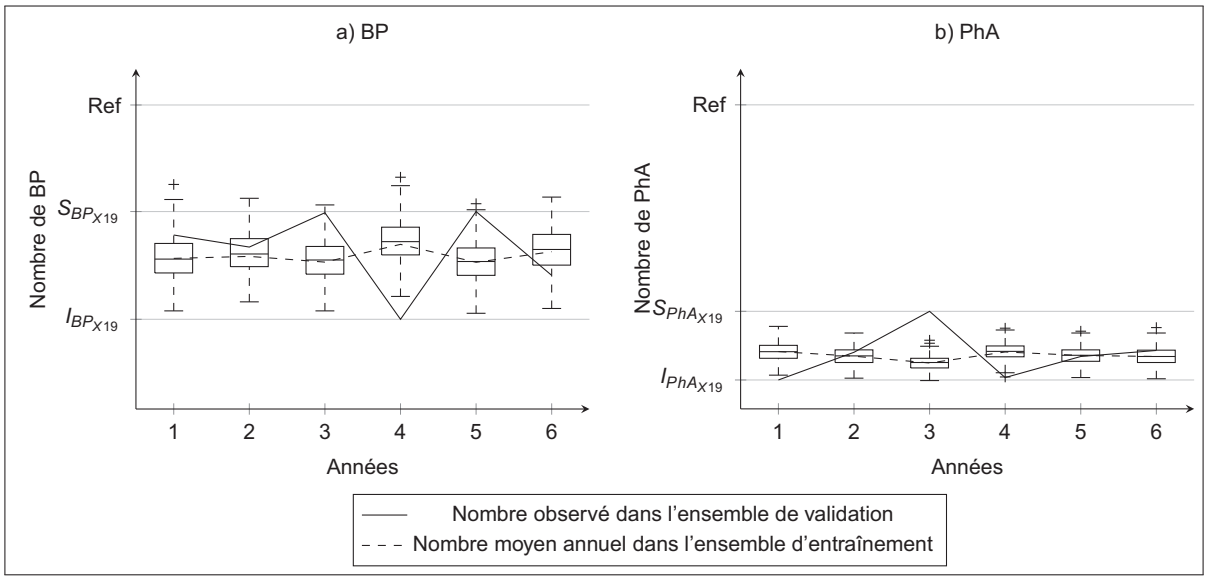


Figure 5.10 Dispersion du nombre de changements d'état simulés avec le modèle #1 (X19)

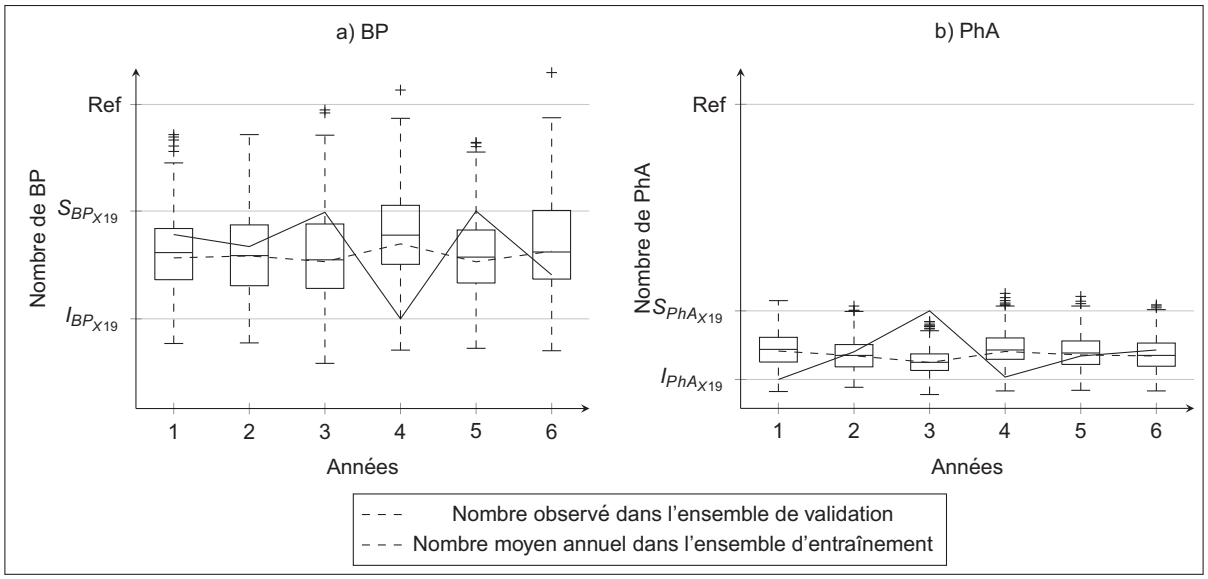


Figure 5.11 Dispersion du nombre de changements d'état simulés avec le modèle #5 (X19)

La comparaison des Figures 5.11 et 5.12 montre une autre différence entre chaînes de Markov et chaînes de Semi-Markov. Le nombre médian de BP obtenu avec les CMTD a tendance à sous-estimer la moyenne annuelle de BP dans l'ensemble d'entraînement, alors que les CSMTD ont tendance à surestimer. Nous expliquons ce phénomène par les distributions de temps de maintien en marche normale. Dans la Figure 5.16 nous présentons pour chaque modèle deux

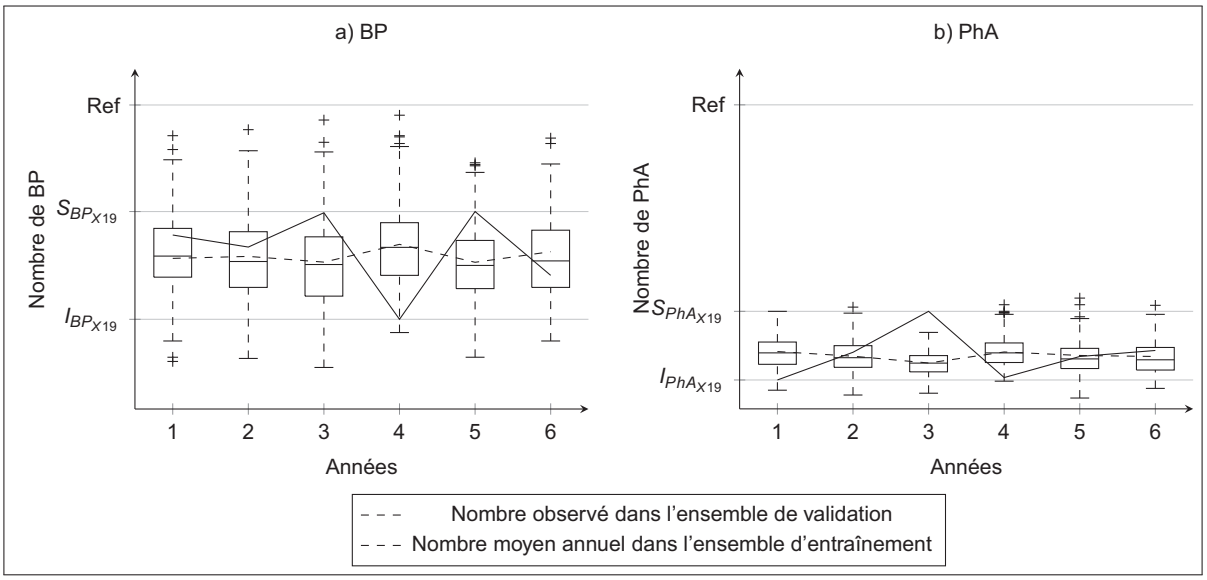


Figure 5.12 Dispersion du nombre de changements d'état simulés avec le modèle #10 (X19)

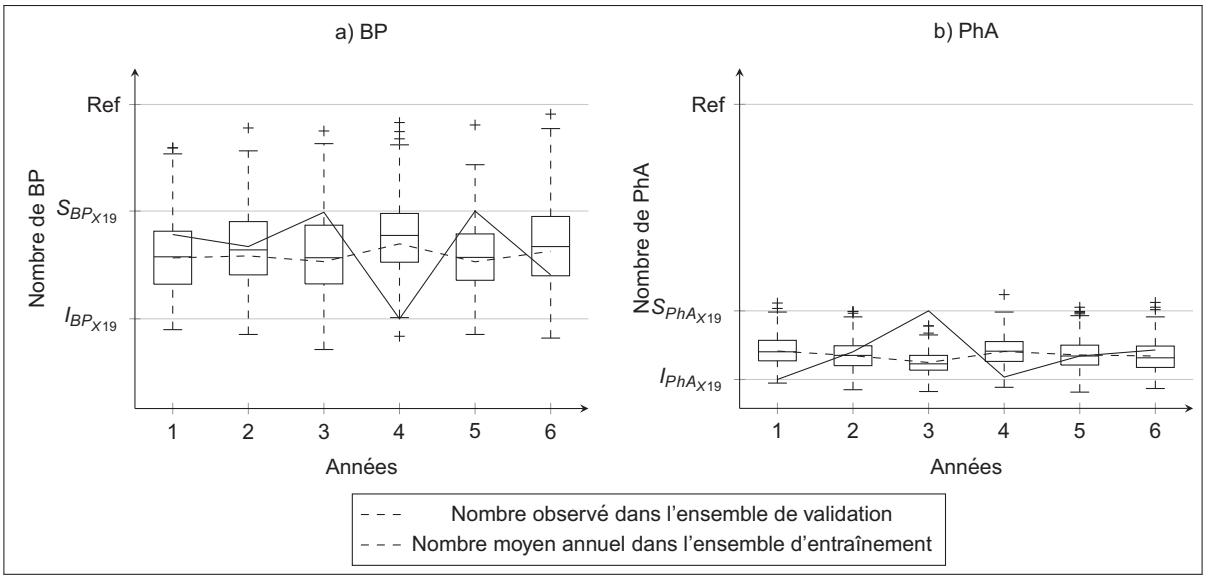


Figure 5.13 Dispersion du nombre de changements d'état simulés avec le modèle #14 (X19)

FdR de temps de maintien en marche normale. Celles-ci sont les extrêmes des FdR donnant à ±2 le nombre médian de BP. Nous constatons que dans l'intervalle de probabilité cumulée $[0, 0.85]$, les temps en MN des CMTD sont surestimés par rapport aux temps de l'ensemble d'entraînement. Si les temps en marche normale sont plus longs, il est logique que le nombre de BP soit plus bas. Les temps des CSMTD sont quant à eux très proches de la réalité observée

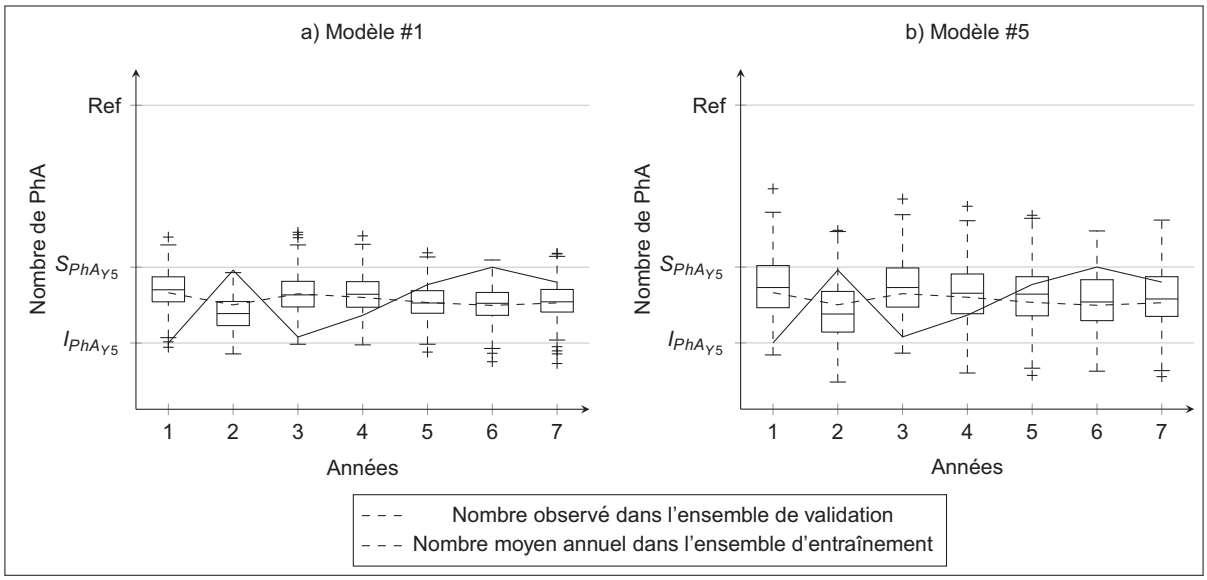


Figure 5.14 Dispersion du nombre de PhA simulés avec les modèles #1 et #5 (Y5)

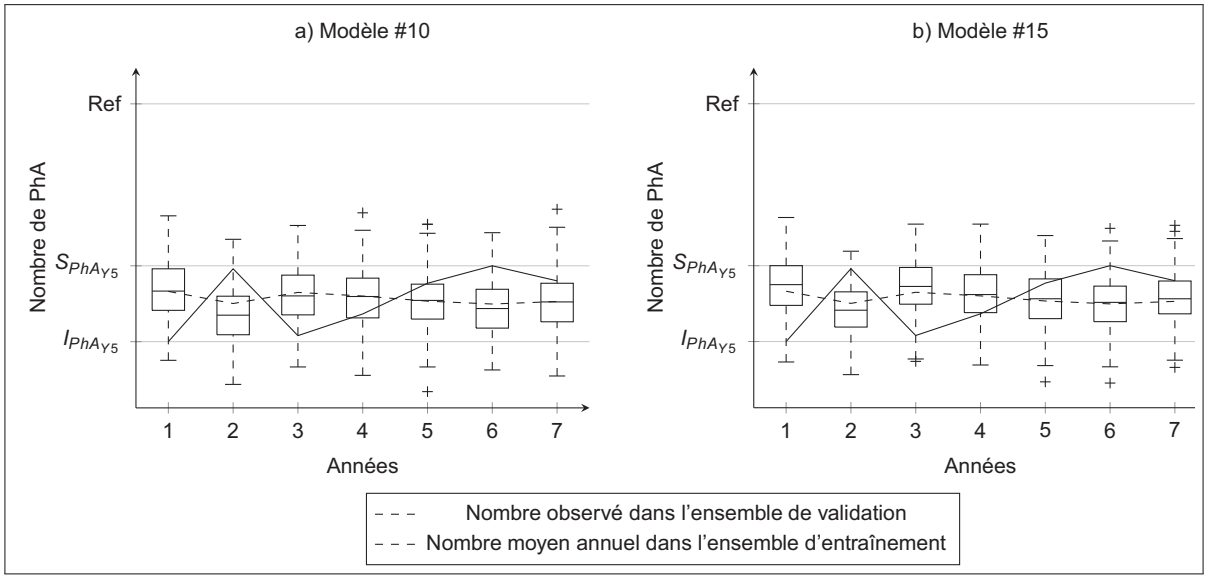


Figure 5.15 Dispersion du nombre de PhA simulés avec les modèles #10 et #14 (Y5)

dans l'intervalle $[0,0.8]$. Au-dessus de 0.8 les temps sont sous-estimés ce qui conduit à un nombre plus élevé de BP.

Afin d'illustrer un peu plus les liens entre FdR des temps de maintien en MN et le nombre de changements d'état, nous utilisons la Figure 5.17. Nous y présentons la FdR de l'ensemble d'entraînement, et les FdR encadrant celles qui mènent à un nombre de BP proche des $25^{ieme}$

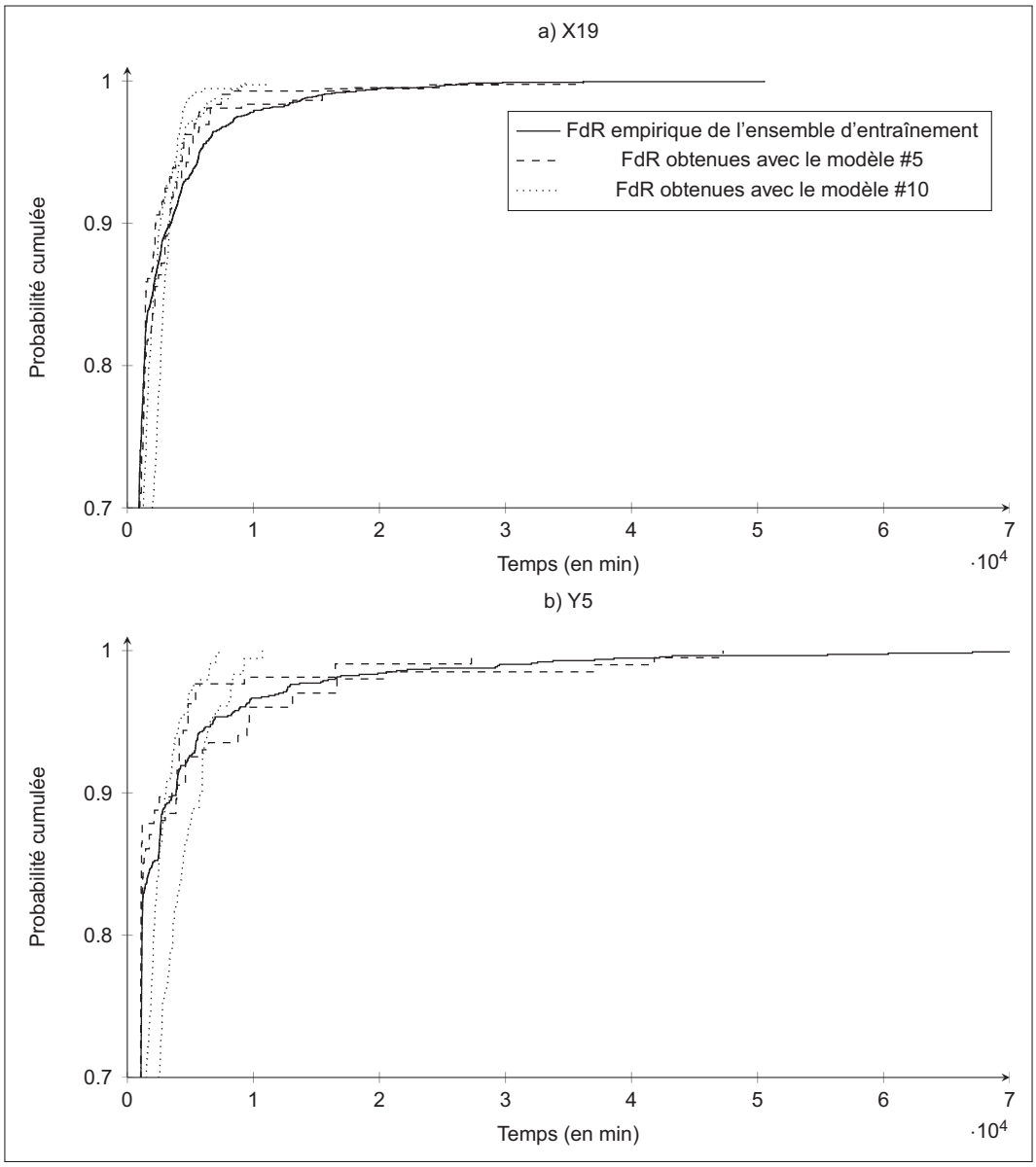


Figure 5.16 Différence entre les FdR de temps de maintien en marche normale obtenues avec les CMTD et les CSMTD

et $75^{ieme}$ quantiles. Il est évident que plus le nombre de BP est élevé plus la FdR sera décalée vers la gauche. Le lien exact entre les FdR et le nombre de BP n'a pas été déterminé. Nous remarquons que pour un même nombre de BP, les FdR peuvent être très différentes. Ceci est dû à l'influence du nombre de PhA. Dans le cas de X19, il faudra faire le lien avec le nombre total de changements d'état (BP+PhA).

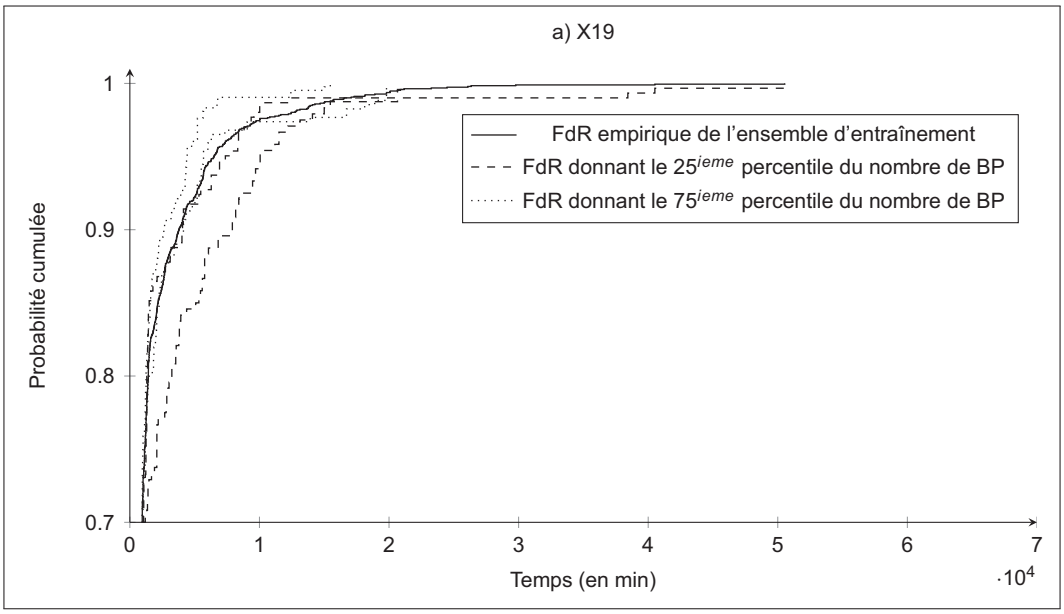


Figure 5.17 Exemple du lien entre nombre de BP et FdR des temps de maintien en marche normale

### 5.2.4 Influence de la taille de fenêtre

Nous revenons dans cette section sur l'influence de la taille de fenêtre de la PNP. Nous avons vu que la diminution de celle-ci tend à augmenter les performances des modèles. Les Figures 5.18 et 5.19 nous montrent que l'amélioration des performances est due à une diminution de la variété des simulations produites par les modèles. Ceci peut s'avérer intéressant dans le cas de Y5 car les dispersions interannuelles sont reparties de façon assez homogène. Mais dans le cas de X19, où ces dispersions peuvent être très élevées, une fenêtre d'un mois peut amener à ne pas intégrer tous les comportements dans les simulations. Cette diminution de la variété des simulations peut s'expliquer par le fait que plus nous réduisons la taille de fenêtre, moins nous autorisons aux différents comportements de durer longtemps. Ils ont donc moins d'impact sur le nombre final de changements d'état. Bien que la diminution de la taille de fenêtre améliore la précision, il augmente aussi le risque de perte d'information. Nous déconseillons donc d'utiliser les modèles avec une taille de fenêtre pour la PNP inférieure à 2 mois. Toutefois si l'on souhaite utiliser la PNP périodique, c'est-à-dire répéter des comportements qui se suivent une période précise, alors la taille de 1 mois est préférable.

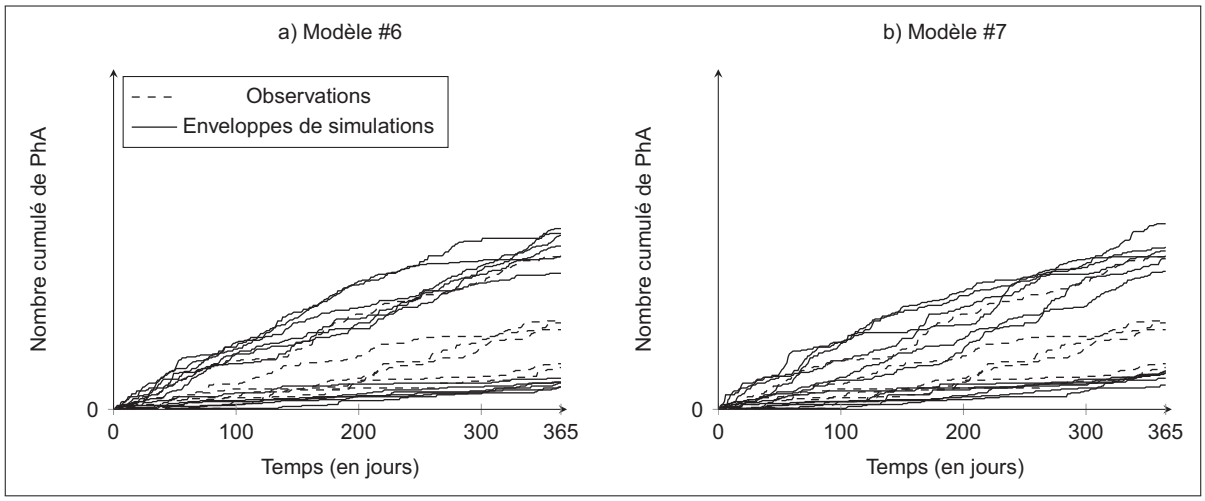


Figure 5.18 Taux d'évolution observés et leurs enveloppes de simulations pour chacun des ensembles de la validation croisée (X19)

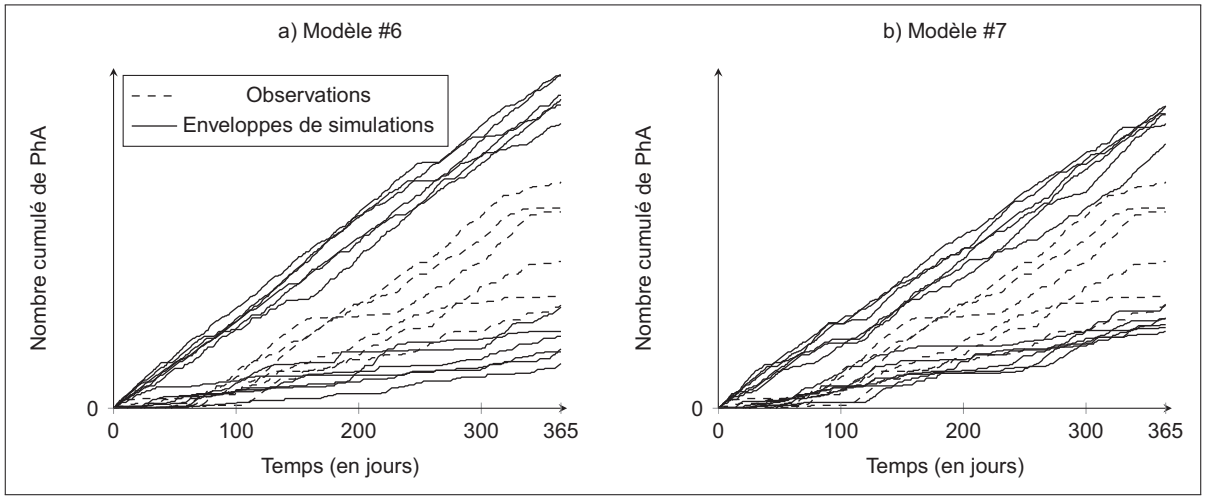


Figure 5.19 Taux d'évolution observés et leurs enveloppes de simulations pour chacun des ensembles de la validation croisée (Y5)

Un autre point se montre digne d'intérêt concernant l'influence de la taille de fenêtre sur les performances des modèles : les Figures 5.8b et 5.9b montrent que pour les distances entre les FdR des temps de maintien, seuls les CMTD s'améliorent. Nous rappelons que nous évaluons les distances entre FdR avec la distance de Kolmogorov-Smirnov (Section 3.1.2) qui est la distance maximum entre deux FdR. Comme nous l'avons vu précédemment, les CMTD ont tendance à surestimer les temps de maintien entre 0 et 0.85 de probabilité cumulée. Comme c'est dans cette zone que la pente est la plus élevée, c'est également dans cette zone que les écarts entre les FdR observées et simulées conduisent aux plus grandes distances de Kolmogorov-

Smirnov. La réduction de la taille de fenêtre augmente les probabilités de couper de longs temps de maintien. En coupant un temps de maintien, on réduit la valeur des paramètres $p_{ii}$ des CMDT. Ceci conduit à la réduction de la moyenne de la fonction de répartition exponentielle et donc à un rapprochement, dans l'intervalle de probabilité cumulée $[0, 0.85]$, entre la FdR observée et celle simulée. La Figure 5.20 illustre ce point.

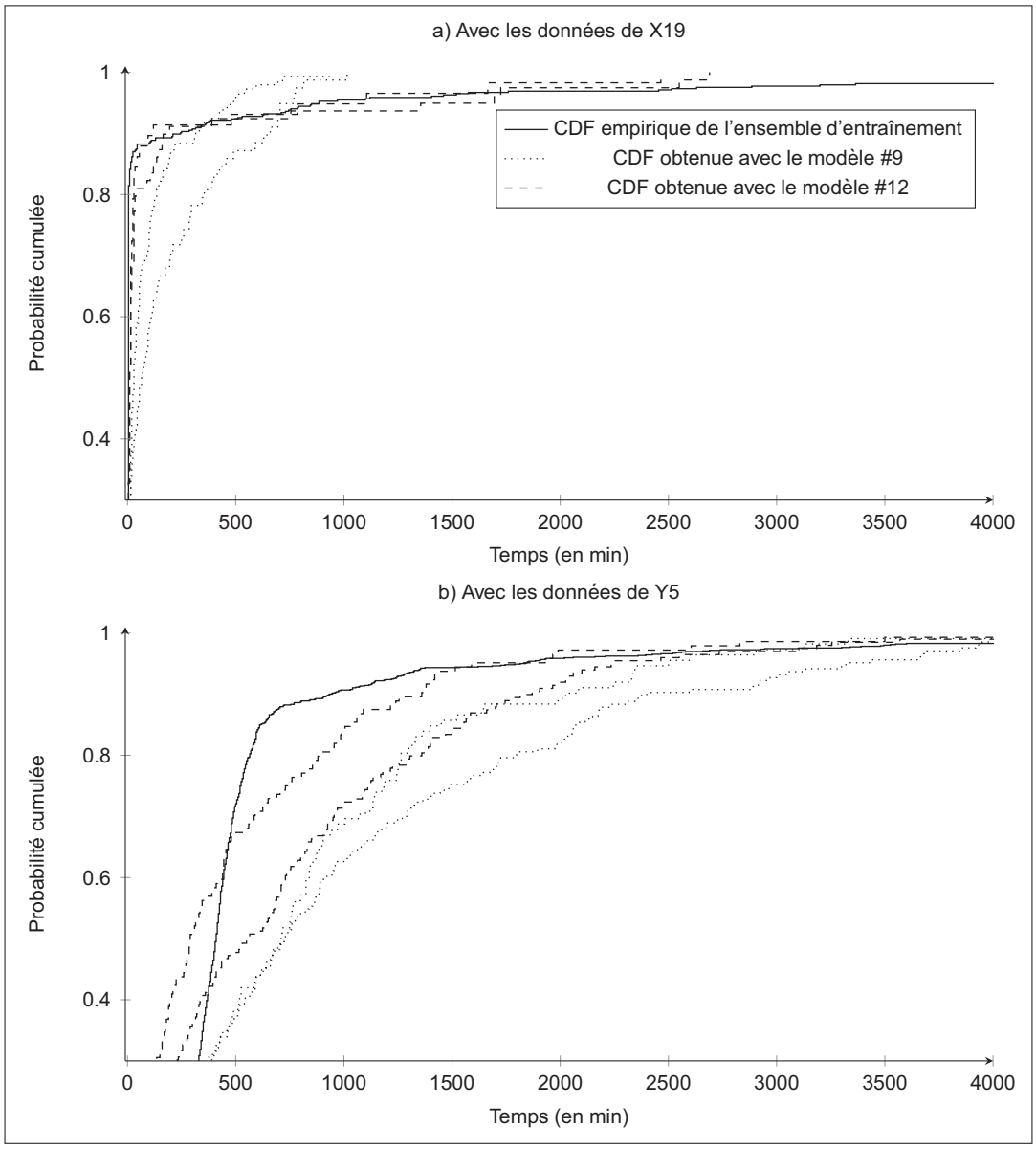


Figure 5.20 Exemple de FdR de temps de maintien en Arrêt obtenues avec les modèles #9 et #12

## 5.3 Application à d'autres GTA

Nous discutons dans cette section des résultats obtenus en appliquant les modèles à d'autres GTA. Le but est de montrer d'autres profils d'opération existants, et de voir comment les modèles arrivent à s'y adapter. Il s'agit de deux autres GTA de la centrale X, les GTA 5 et 12 (X5 et X12) et le GTA 8 (Y8) de la centrale Y. Le fonctionnement de ces GTA est résumé dans la Figure 5.21 Nous pouvons voir que X5 et X12 ont des profils d'opération bien différents de celui de X19, avec moins de changements d'état et moins de dispersion interannuelle. Les points communs à X19 sont caractéristiques de la centrale X, c'est-à-dire un facteur d'utilisation très élevé, donc des temps en arrêt relativement courts, et un nombre élevé de BP par rapport à celui de PhA. Quant au GTA Y8, il présente un profil d'opération qui n'est pas similaire à celui de Y5 : Y8 subit un plus grand nombre de PhA, et a un facteur d'utilisation bien inférieur. La dispersion interannuelle semble quant à elle être moins importante.

Nous présentons uniquement les erreurs de généralisation sans refaire l'analyse complète de la section précédente. Les erreurs de généralisation sont regroupées dans les Tableaux 5.6 à 5.8. Les résultats contenus dans ces tableaux font écho aux dispersions interannuelles moins importantes avec des erreurs de généralisation plus faible. Nous y retrouvons les points communs entre X19, X5 et X12, c'est-à-dire les erreurs concernant les FdR des temps de maintien en arrêt qui sont très élevées pour les CMTD. Ceci est dû au fort facteur d'utilisation, conduisant à des temps en arrêt très courts et donc une pente très importantes pour l'intervalle de probabilité cumulé $[0, 0.8]$. Nous avons vu que cette pente provoque des distances de KS élevées. Par contre nous ne retrouvons pas cette différence de performances entre CMTD et CSMTD pour les distances entre les FdR du nombre de changements d'état par jour. Ce qui est commun à toutes les applications est la meilleure performance des CSMTD & PNP et BBM par rapport au CMTD & PNP en ce qui concerne les FdR de temps de maintien, et le fait que le modèle #1 présente les plus faibles erreurs de généralisation.

Quel que soit le GTA sur lequel nous appliquons les modèles, les résultats demeurent consistants.

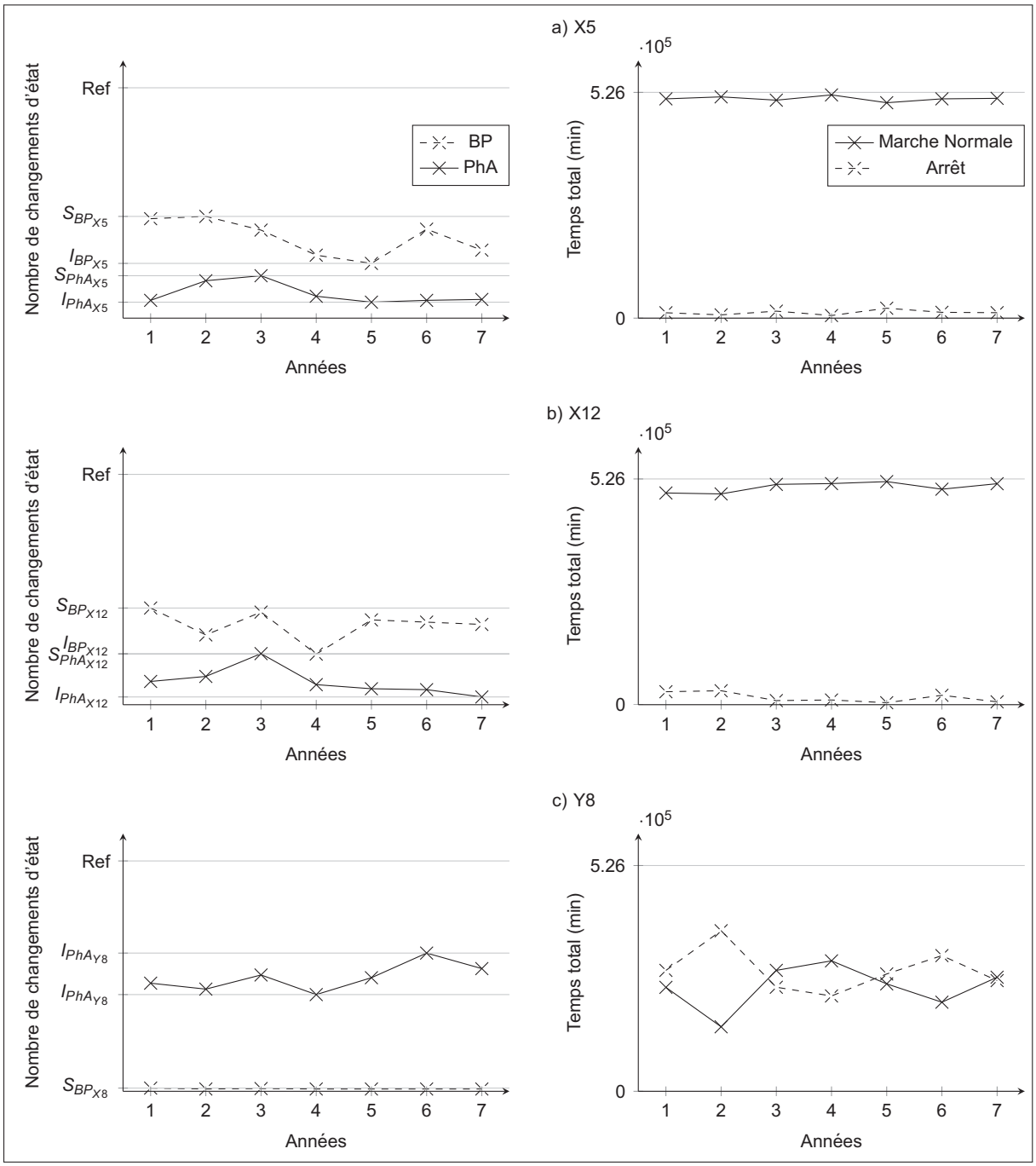


Figure 5.21 Profils d'opération des GTA X5, X12 et Y8

Tableau 5.6 Erreur de généralisation pour le critère 1

| # du modèle | Type de modèle | $GE_{MAPE}$ (BP) | | | $GE_{MAPE}$ (PhA) | | |
|---|---|---|---|---|---|---|---|
| | | **X5** | **X12** | **Y8** | **X5** | **X12** | **Y8** |
| #1 | CSMTD | 0.280 | 0.237 | - | 0.504 | 0.923 | 0.130 |
| #2 | CMTD | 0.261 | 0.216 | - | 0.480 | 0.900 | 0.109 |
| #3 | CSMTD & PNP | 0.322 | 0.280 | - | 0.589 | 1.107 | 0.162 |
| #4 | | 0.323 | 0.291 | - | 0.597 | 1.014 | 0.153 |
| #5 | | 0.339 | 0.306 | - | 0.578 | 1.001 | 0.158 |
| #6 | | 0.324 | 0.299 | - | 0.571 | 1.000 | 0.159 |
| #7 | | 0.318 | 0.292 | - | 0.572 | 1.02 | 0.155 |
| #8 | CMTD & PNP | 0.311 | 0.258 | - | 0.565 | 1.061 | 0.144 |
| #9 | | 0.314 | 0.266 | - | 0.547 | 1.026 | 0.143 |
| #10 | | 0.308 | 0.282 | - | 0.545 | 0.962 | 0.145 |
| #11 | | 0.299 | 0.271 | - | 0.533 | 0.941 | 0.145 |
| #12 | | 0.287 | 0.259 | - | 0.515 | 0.945 | 0.144 |
| #13 | BBM | 0.312 | 0.276 | - | 0.563 | 1.044 | 0.139 |
| #14 | | 0.313 | 0.271 | - | 0.537 | 0.979 | 0.142 |
| #15 | | 0.300 | 0.272 | - | 0.522 | 0.963 | 0.144 |
| #16 | | 0.296 | 0.261 | - | 0.503 | 0.936 | 0.139 |
| #17 | | 0.270 | 0.230 | - | 0.496 | 0.951 | 0.134 |

Tableau 5.7 Erreur de généralisation pour le critère 2 : Nombre de changements d'état par jour

| # du modèle | Type de modèle | $GE_{D_C}$ (BP) | | | $GE_{D_C}$ (PhA) | | |
|---|---|---|---|---|---|---|---|
| | | **X5** | **X12** | **Y8** | **X5** | **X12** | **Y8** |
| #1 | CSMTD | 0.090 | 0.068 | 0.012 | 0.058 | 0.064 | 0.078 |
| #2 | CMTD | 0.101 | 0.089 | 0.012 | 0.059 | 0.066 | 0.064 |
| #3 | CSMTD & PNP | 0.103 | 0.079 | 0.011 | 0.065 | 0.079 | 0.092 |
| #4 | | 0.101 | 0.081 | 0.010 | 0.064 | 0.073 | 0.087 |
| #5 | | 0.104 | 0.083 | 0.010 | 0.062 | 0.070 | 0.087 |
| #6 | | 0.099 | 0.081 | 0.009 | 0.061 | 0.068 | 0.087 |
| #7 | | 0.096 | 0.078 | 0.009 | 0.061 | 0.068 | 0.082 |
| #8 | CMTD & PNP | 0.112 | 0.099 | 0.011 | 0.065 | 0.082 | 0.076 |
| #9 | | 0.111 | 0.098 | 0.010 | 0.062 | 0.076 | 0.075 |
| #10 | | 0.107 | 0.097 | 0.009 | 0.062 | 0.070 | 0.076 |
| #11 | | 0.103 | 0.092 | 0.010 | 0.061 | 0.069 | 0.076 |
| #12 | | 0.097 | 0.083 | 0.012 | 0.058 | 0.067 | 0.077 |
| #13 | BBM | 0.101 | 0.080 | 0.005 | 0.063 | 0.076 | 0.084 |
| #14 | | 0.099 | 0.077 | 0.005 | 0.061 | 0.070 | 0.085 |
| #15 | | 0.096 | 0.076 | 0.005 | 0.059 | 0.069 | 0.084 |
| #16 | | 0.094 | 0.073 | 0.005 | 0.057 | 0.066 | 0.081 |
| #17 | | 0.086 | 0.064 | 0.005 | 0.056 | 0.064 | 0.074 |

Tableau 5.8 Erreur de généralisation pour le critère 2 : Temps de maintien

| # du modèle | Type de modèle | $GE_{D_E}$ (MN) | | | $GE_{D_E}$ (A) | | |
|---|---|---|---|---|---|---|---|
| | | **X5** | **X12** | **Y8** | **X5** | **X12** | **Y8** |
| #1 | CSMTD | 0.108 | 0.110 | 0.144 | 0.116 | 0.161 | 0.107 |
| #2 | CMTD | 0.259 | 0.271 | 0.221 | 0.852 | 0.798 | 0.241 |
| #3 | CSMTD & PNP | 0.126 | 0.133 | 0.174 | 0.144 | 0.194 | 0.127 |
| #4 | | 0.128 | 0.135 | 0.178 | 0.149 | 0.196 | 0.141 |
| #5 | | 0.130 | 0.136 | 0.179 | 0.154 | 0.196 | 0.142 |
| #6 | | 0.126 | 0.133 | 0.174 | 0.159 | 0.197 | 0.136 |
| #7 | | 0.124 | 0.130 | 0.165 | 0.174 | 0.196 | 0.127 |
| #8 | CMTD & PNP | 0.268 | 0.276 | 0.232 | 0.836 | 0.784 | 0.245 |
| #9 | | 0.254 | 0.270 | 0.231 | 0.739 | 0.690 | 0.241 |
| #10 | | 0.229 | 0.252 | 0.224 | 0.631 | 0.570 | 0.230 |
| #11 | | 0.212 | 0.239 | 0.217 | 0.581 | 0.534 | 0.219 |
| #12 | | 0.182 | 0.212 | 0.207 | 0.510 | 0.476 | 0.207 |
| #13 | BBM | 0.116 | 0.126 | 0.174 | 0.141 | 0.187 | 0.133 |
| #14 | | 0.117 | 0.123 | 0.173 | 0.137 | 0.180 | 0.134 |
| #15 | | 0.118 | 0.122 | 0.165 | 0.132 | 0.177 | 0.128 |
| #16 | | 0.116 | 0.119 | 0.159 | 0.128 | 0.173 | 0.119 |
| #17 | | 0.111 | 0.116 | 0.145 | 0.124 | 0.166 | 0.109 |

## 5.4 Conclusion du Chapitre 5

L'application de la procédure de validation décrite dans le Chapitre 3 nous a servi d'assise pour procéder à la comparaison et la validation des modèles. Nous avons ainsi pu avoir une connaissance objective de leurs forces et de leurs faiblesses. Dans cette procédure, nous n'avons fait l'usage d'aucun test d'hypothèse, car nous considérons que leur emploi n'est pas indispensable et qu'ils sont des outils parmi d'autres. Nous avons simplement choisi des outils qui ne se basent pas sur une limite somme toute assez arbitraire. Au final, quelles que soient les méthodes employées, le jugement de la qualité d'un modèle passe avant tout par une analyse approfondie des résultats.

Les résultats illustrent bien ce que peuvent apporter les chaînes de Semi-Markov par rapport aux chaînes de Markov. La possibilité de choisir les FdR pour les temps de maintien permet de relaxer les fortes hypothèses des chaînes de Markov afin d'avoir un modèle plus réaliste. Dans notre cas, nous contre-indiquons l'utilisation des chaînes de Markov : même si elles ont des performances équivalentes aux autres modèles pour le nombre total de chan-

gements d'état, elles modifient la structure des données et la répartition de ces changements d'état dans le temps. Quant au choix entre CSMTD et BBM, celui-ci est à la discrétion de l'utilisateur. Nous rappelons que les CSMTD ont tendance à être plus conservatrices. Malgré cela, nous conseillons leur emploi, car à long terme elles présentent un plus grand potentiel de perfectionnement.

L'analyse des résultats a mené à la confirmation que le nombre de changements d'état dépend directement de la FdR des temps de maintien en marche normale. Ceci peut être intéressant dans une perspective d'amélioration (nous y reviendrons au chapitre suivant). Enfin, l'essai des modèles sur divers GTA nous a montré que les modèles s'adaptent facilement à tout type de fonctionnement. Nous ne pouvons choisir une taille de fenêtre universelle à toutes les applications. Nous conseillons toutefois l'utilisation d'une fenêtre de 2 ou 3 mois.

Nous avons présentés les résultats de ce chapitre lors du Symposium sur la maîtrise du vieillissement organisé par l'IREQ, ils font également l'objet d'un article qui vise la publication dans *International Journal of Forecasting*

# CHAPITRE 6

# RECOMMANDATIONS ET PERSPECTIVES

*Je pense qu'il est bien plus intéressant de vivre en ne sachant pas que de vivre avec des réponses qui ne sont pas les bonnes*
Richard Feynman

## 6.1 Recommandations

Les résultats de la section précédente nous permettent de retenir le modèle qui offre le meilleur potentiel pour la suite du projet PréDDIT. Selon nous, il s'agit des chaînes de Semi-Markov à temps discret, même si le Bootstrap à bloc mobile est une excellente alternative. Les chaînes de Markov ne sont simplement pas adaptées à nos données, car elles obligent les temps de maintien à être exponentiellement distribués, ce qui ne correspond pas aux observations. Nous avons également décrit comment nous prenons en compte la non-stationnarité des données par l'intermédiaire d'une perturbation non paramétrique. Celle-ci consiste à changer les paramètres du modèle, régulièrement durant la simulation. Ce changement est opéré par le calcul de nouveaux paramètres à partir des données contenues dans une fenêtre positionnée aléatoirement dans l'historique de fonctionnement. Afin d'utiliser le modèle choisi, certaines précautions doivent être prises. La Figure 6.1 résume les étapes à suivre pour utiliser les CSMTD ou le BBM le cas échéant.

Évidemment, la première étape est la récupération des données. Il est essentiel d'importer régulièrement (sur une base trimestrielle ou annuelle) de nouvelles données, il n'est pas nécessaire de les récupérer en temps réel. Nous devons ensuite consolider les données, parce que nous avons constaté que le défaut majeur des modèles présentés est leur dépendance aux données historiques. Si un GTA subit un changement significatif de son profil d'opération, les données historiques ne seront plus utilisables. Deux approches sont alors envisageables : nous continuons de faire des prédictions avec les données passées et ces prédictions seront ajustées au fur et à mesure que de nouvelles données entrent (en supprimant progressivement les anciennes), ou nous nous servons de données d'un GTA dont l'utilisation est, ou a été équivalente. Dans tous les cas, il faut être vigilant à toutes modifications dans l'utilisation des GTA. Il faut également retirer les données qui correspondent aux arrêts de longue durée (supérieurs à un mois). Nous les considérons comme ne faisant pas partie du fonctionnement normal. Certains de ces arrêts sont planifiés et pourraient donc être intégrés dans les simulations afin d'améliorer les prédictions. Dans les cas où les GTA sont arrêtés à cause d'une défaillance, il

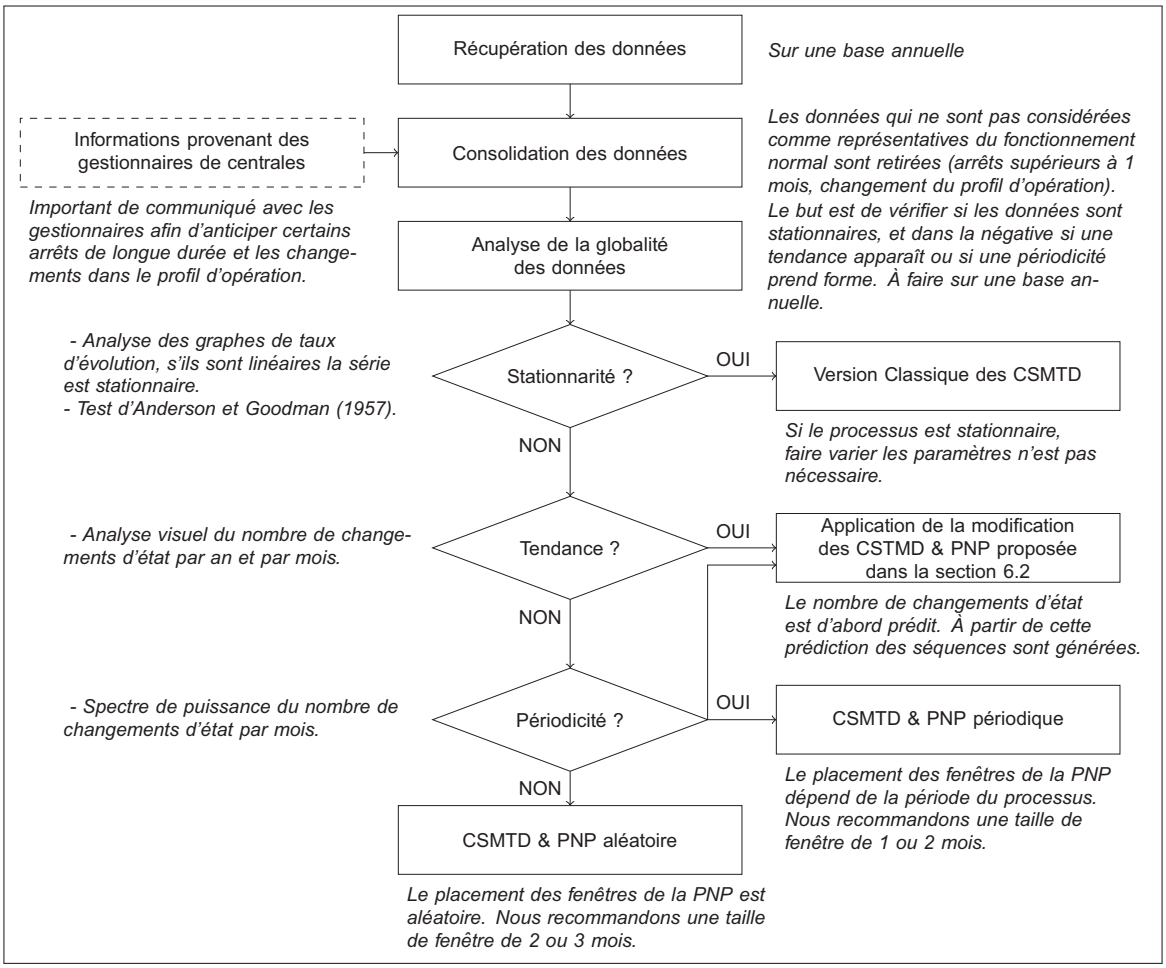


Figure 6.1 Recommandations quant à l'utilisation des modèles

faudrait corriger *a posteriori* le nombre de changements d'état prédit. Nous avons fixé arbitrairement ce seuil à 1 mois, nous pouvons l'ajuster pour chaque GTA, surtout si certains d'entre eux sont arrêtés régulièrement pour de longues périodes. Nous suggérons fortement d'établir un lien direct avec les gestionnaires de centrales afin d'anticiper à la fois les changements de politiques d'opération des GTA, ainsi que les arrêts planifiés.

L'analyse des données dans leur globalité va de pair avec leur importation et leur consolidation. Il est nécessaire de refaire une analyse descriptive après chaque importation afin de détecter un changement du profil d'utilisation ou l'apparition de tendances ou de périodicités. Il est hautement improbable que des données non stationnaires deviennent stationnaires. Toutefois, si cela arrive, il faudra supprimer les données passées non stationnaires. Nous recommandons alors l'utilisation de la version classique des CSMTD. Si l'analyse détecte la présence d'une tendance, les modèles décrits dans ce mémoire ne pourront la prendre en compte. Dans la section suivante, nous décrivons une piste d'amélioration pour contourner ce problème.

Si le fonctionnement est périodique, la façon dont nous mettons en œuvre les modèles rend impossible la soustraction de la périodicité aux données, comme cela est fait habituellement. Nous avons donc deux solutions pour l'intégrer aux simulations. La première est d'utiliser la perturbation non paramétrique périodique, où l'on va associer, pour chaque mois de la période, des valeurs de paramètres et des FdR pour les temps de maintien. Ces éléments sont obtenus avec les données contenues dans les fenêtres de 1 mois qui se situent par rapport au temps présent à une distance temporelle égale à un multiple de la période détectée. Nous simulons ensuite avec un modèle dont les paramètres se modifient tous les mois. La seconde solution est d'utiliser l'amélioration proposée dans la section suivante.

Dans le cas d'absence de tendance et de périodicité, nous utilisons la perturbation non paramétrique aléatoire. Le choix de la taille de fenêtre a son importance. Nous suggérons d'utiliser des fenêtres de 2 ou 3 mois. Toutefois, si en appliquant le modèle, il est constaté une forte tendance à surestimer le nombre de changements d'état, la cause est probablement une coupure des longs temps de maintien. Il sera alors judicieux d'augmenter la taille de fenêtre à 6 ou 12 mois. Bien que le choix final entre les CSMTD ou le BBM revienne à l'utilisateur, nous rappelons que nous recommandons les CSMTD, car elles présentent de meilleures perspectives d'amélioration. Les deux modèles produisent des simulations avec un faible niveau d'erreur systématique. De son côté, le BBM, pour une méthode de rééchantillonnage, génère une grande diversité de simulations. Ceci est probablement dû à la grande diversité de comportement dans les données. Quant aux CSMTD, elles proposent une plus grande variété de simulations. Certaines peuvent paraître non réalistes, mais sont, selon le modèle, statistiquement possibles. Les CSMTD sont donc plus conservatrices, et peuvent être intéressantes à utiliser pour les gestionnaires.

## 6.2 Perspectives

Nous discutons ici des perspectives d'amélioration dans l'usage des chaînes de Semi-Markov. Pour simuler le comportement des GTA, le principal élément qui limite la précision du modèle est l'aspect imprévisible des changements de comportement, c'est-à-dire la non-stationnarité. En réalité, les modèles font de l'extrapolation des données d'entrée. Ces extrapolations sont ensuite utilisées comme prédiction du futur (où comme approximation du passé). L'hypothèse de base derrière ceci est que, dans la prochaine année, le nombre de changements d'état se situe dans un intervalle qui intègre obligatoirement toutes les données du passé (Figure 6.2a).

Nous faisons donc l'hypothèse que nous serons capables (dans un futur plus ou moins proche) de prédire avec plus de précision le nombre de changements d'état de l'année suivante, ou même du mois suivant (ce qui peut être le cas si la présence de tendance ou de périodicité est détectée). L'intervalle des valeurs possibles s'en trouve donc réduit. Si nous augmentons l'horizon $H$ de cette prédiction, cet intervalle s'élargira jusqu'à englober toutes les valeurs

passées (Figure 6.2b) [1]. Nous avons vu qu'un lien direct entre le nombre de changements d'état et FdR des temps de maintien en marche normale existe. Actuellement, nous utilisons les FdR empiriques (non-paramétrique) pour les temps de maintien (Figure 6.2c). L'idée derrière la suggestion d'amélioration est la suivante :

1. Nous décrivons les FdR des temps de maintien en marche normale par des distributions paramétriques.
2. Nous relions le nombre total de changements d'état (nombre de BP plus nombre de PhA) aux paramètres de cette FdR.
3. Nous déterminons dans quel intervalle doivent se trouver les paramètres $\alpha$ et $\beta$ de la FdR (Figure 6.2d) pour mener à l'intervalle de prédiction pour le nombre de changements d'état.
4. Nous générons plusieurs simulations pour lesquelles les valeurs des paramètres $\alpha$ et $\beta$ sont pigées dans l'intervalle défini précédemment pour chaque simulation. Nous pouvons également envisager de les changer en cours de simulation.

Jusqu'à présent, nous avons parlé uniquement du nombre total de changements d'état. La distinction entre le nombre de BP et celui de PhA se fait par l'intermédiaire des probabilités de changements d'état $p_{12}$ et $p_{13}$. Nous pouvons soit les garder constantes, soit les faire varier dans le temps. Évidemment, l'hypothèse que seuls les temps de maintien en marche normale influent sur le nombre de changements est forte. Elle semble cohérente car le facteur d'utilisation des GTA étudiés est élevé. Dans le cas de GTA avec un plus faible facteur d'utilisation, les temps en arrêt ne peuvent plus être considérés comme négligeables. Nous devrons alors lier le nombre de changements d'état à la fois aux paramètres de la FdR des temps en marche normale, et à ceux de la FdR des temps en arrêt.

Actuellement, cette modification n'est pas réalisable car le manque de données ne nous permet pas de réduire les possibilités du nombre de changements d'état pour l'année ou les mois à venir. Lorsque sa mise en place deviendra intéressante, une étude complète, qui passe par un plan d'expérience, devra être effectuée. Nous stipulons que cette proposition améliorerait grandement la précision des chaînes de Semi-Markov.

1. l'horizon de cinq ans est ici arbitraire et a été choisi à titre illustratif.

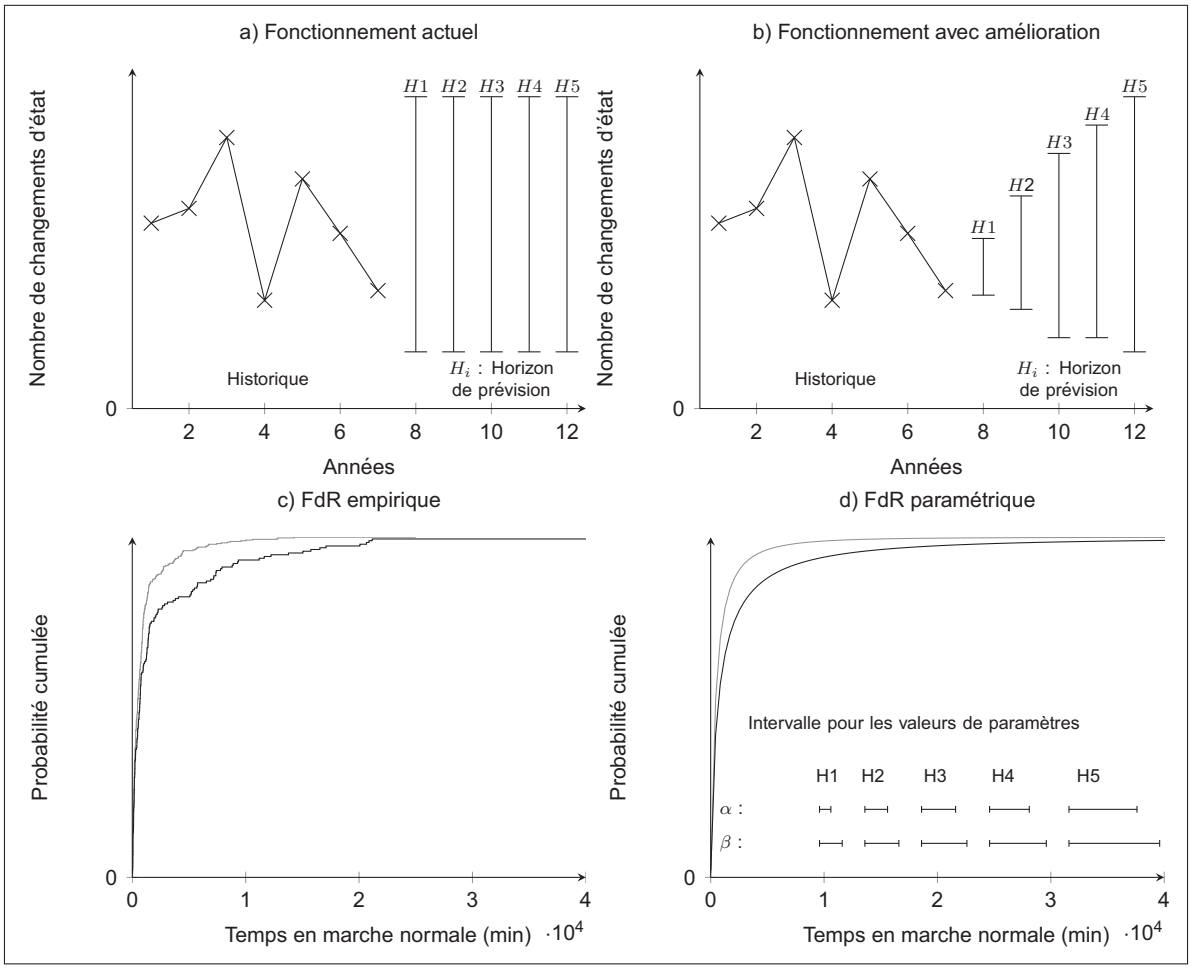


Figure 6.2 Principe de l'amélioration proposée

# CONCLUSION

Les travaux présentés dans ce mémoire ont conduit à la réalisation de générateurs de séquences synthétiques de chargement qui sont représentatives du comportement des groupes turbine-alternateur. Les modèles étudiés ont chacun leurs points positifs et négatifs. Pour notre application, nous avons mis en évidence la supériorité des chaînes de Semi-Markov et du Bootstrap à bloc mobile par rapport aux chaînes de Markov. Néanmoins, avec une vision à long terme, nous recommandons les chaînes de Semi-Markov plutôt que le Bootstrap à bloc mobile.

L'objectif que nous avions avec ce projet était de développer un modèle ayant un intérêt pratique, nous ne voulions pas nous inscrire dans le domaine de *l'art pour l'art* : nous n'aspirions pas à prouver notre habileté à construire un modèle complexe, nous voulions créer un modèle qui pourra être utilisé dans de futures études de dégradation. Tous les modèles comparés cadrent avec cette vision : ils sont simples, faciles à appliquer et leur interprétation physique est directe. Les chaînes de Semi-Markov apportent une flexibilité que n'ont pas les autres modèles. Il nous tenait à cœur de montrer les qualités, mais également les carences, de ce type de modèle. Nous souhaitions les mettre en avant, car nous pensons qu'elles méritent d'être employées plus souvent.

De plus, nous avions la volonté de décrire avec précision la procédure de validation qui nous permet de juger la qualité de ces modèles. Cet aspect est quelques fois négligé, et selon nous, il devrait toujours occuper une place aussi importante que la construction des dits modèles. Nous n'avons pas la présomption d'affirmer que la procédure mise en œuvre est universelle. Comme nous l'avons vu, les critères d'évaluation sont propres à chaque projet, mais ils doivent être choisis avec précaution. De la même manière, la sélection des outils est à la discrétion du modélisateur, toutefois ils doivent concorder avec les critères d'évaluation. Quant au fait que nous n'avons pas utilisé de tests d'hypothèse, il s'agit là d'un choix personnel mûri par notre réflexion et nos lectures. Nous n'avions pas l'objectif d'en faire une critique détaillée.

Pour nous, ces travaux sont plus un point de départ qu'une finalité. En effet, à notre connaissance, c'est la première modélisation du chargement des groupes turbines-alternateur effectuée au sein d'Hydro-Québec. Nous considérons donc ce modèle comme une base de travail qui devra être continuellement améliorée. Nous avons déjà proposé une piste d'amélioration. Il s'agit également de la première fois que la morphologie d'opération des groupes Turbines-Alternateur est étudiée. Nous tenons à souligner ce point, que nous n'avons pas abordé dans ce mémoire. En effet, les travaux de cette maîtrise ont permis de mieux comprendre les différences de fonctionnement des groupes turbine-alternateur et d'ouvrir la voie à leur analyse plus systématique. Il est dans l'intérêt d'Hydro-Québec de poursuivre l'analyse des données de chargement qui peuvent être riches en enseignement et amener, entre autres, à la mise à

jour des cahiers des charges pour le fabricant de turbines. Il reste encore beaucoup de travail à effectuer. À titre d'exemple il pourrait être intéressant d'effectuer une étude sur les liens présumés entre l'ordonnancement des groupes turbine-alternateur dans une centrale (c'est-à-dire les priorités de démarrage) et le nombre de changements d'état.

Au terme du projet PréDDIT, nous souhaitons que ces travaux soient intégrés parmi les outils permettant une planification plus efficace des arrêts pour maintenance ou inspection. Ils peuvent également amener à une opération plus éclairée des groupes turbine-alternateur afin de prolonger leur durée de vie, en incluant, par exemple, les coûts reliés à la dégradation et à la maintenance des turbines dans les modèles utilisés pour la planification de la production.

# ANNEXE I

# PROGRAMMES DE SIMULATION

Nous présentons dans cette annexe les programmes mis au point pour la création des modèles, la réalisation des simulations ainsi que de la validation croisée.

## 1 Programme central

```
% Comparaison et validation des modèles.
% numéro du GTA étudié
c=5;
%====================Importation des valeurs de débit
% Boucles d'importation des données
for annee = 2005:2011;
  % Nom du fichier à importer
  fichier_1 = ['RB_' num2str(annee) '.csv'];
  fid_1 = fopen(fichier_1);
  % Importation des données du fichier dans data_1
  data_1=textscan(fid_1,'%s %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f ','delimiter',';','headerlines',3);
  % Comme l'année 2006 contient un arrêt de longue durée, les
  % données correspondantes à cet arrêt sont retirées.
  if annee ==2006
    Debit_an(annee-2004).Debit = [data_1{c+1}(1:77140,1);...
    data_1{c+1}(87800:length(data_1{c+1}),1)];
  else
     Debit_an(annee-2004).Debit = [data_1{c+1}];
  end
  fclose(fid_1);
  clear fid_1 data_1
end
%====================
%====================Création des ensembles de validation
% et d'entraînement
Test_set = repmat(struct('Debit',[]),1,7);
Training_set = repmat(struct('Debit',[]),1,7);
for CV = 1:7;
  % Définition de la Longueur de simulation
  % (en année)
  if CV == 2
    Longueur_Sim = 0.8986;
  else
      Longueur_Sim = 1;
  end
  Test_set(CV).Debit = Debit_an(CV).Debit;
  for Tr_S = [1:CV-1,CV+1:7];
    Training_set(CV).Debit = [Training_set(CV).Debit;Debit_an(Tr_S).Debit];
  end
  %=========== Comptage données réelles
  % Comptage nombre de changement d'état par jour
  Test_set(CV).Transition = Comptage_tran(Test_set(CV).Debit,1);
  % Comptage des temps passés dans chaque état
  [Test_set(CV).Tps_MN Test_set(CV).Tps_MV Test_set(CV).Tps_A] = Comptage_tps(Test_set(CV).Debit);
  Temps(1,1) = sum(Test_set(CV).Tps_MN(:,1)-5);
  Temps(2,1) = sum(Test_set(CV).Tps_MV(:,1)-5);
  Temps(3,1) = sum(Test_set(CV).Tps_A(:,1)-5);
  %============
  %============ Calcul du nombre de changements d'état cumulé par jour
  Cum = zeros(6,ceil(Longueur_Sim*365));
  Test_set(CV).Transition_Cum = zeros(6,ceil(Longueur_Sim*365));
  for t=1:6
    Cum(t,1) = Test_set(CV).Transition(t,1);
    for i=2:ceil(Longueur_Sim*365)
      Cum(t,i) = Test_set(CV).Transition(t,i) + Cum(t,i-1);
    end
      Test_set(CV).Transition_Cum(t,:) = Cum(t,:);
  end
end
%============
clear Cum

%============ Entrée dans la boucle de validation croisée
Nombre_sim = 400;    % Nombre de simulations
Nombre_modele = 17;  % Nombre de modèle comparés
% Structure contenant le résultat des simulations
CV7 = repmat(struct('Simulation',[]),1,7);
% Structure contenant les performances des modèles
CV_Proc = repmat(struct( ...
    'BP', struct('Diff_Nbr',zeros(Nombre_sim,Nombre_modele),...
    'D_CDF',zeros(Nombre_sim,Nombre_modele)), ...
    'MP', struct('Diff_Nbr',zeros(Nombre_sim,Nombre_modele),...
```

```
    'D_CDF',zeros(Nombre_sim,Nombre_modele)), ...
    'PhA', struct('Diff_Nbr',zeros(Nombre_sim,Nombre_modele),...
    'D_CDF',zeros(Nombre_sim,Nombre_modele)), ...
    'PhD', struct('Diff_Nbr',zeros(Nombre_sim,Nombre_modele),...
    'D_CDF',zeros(Nombre_sim,Nombre_modele)), ...
    'F', struct('Diff_Nbr',zeros(Nombre_sim,Nombre_modele),...
    'D_CDF',zeros(Nombre_sim,Nombre_modele)), ...
    'MR', struct('Diff_Nbr',zeros(Nombre_sim,Nombre_modele),...
    'D_CDF',zeros(Nombre_sim,Nombre_modele)), ...
    'Tps_MN', struct('D_CDF',zeros(Nombre_sim,Nombre_modele)), ...
    'Tps_MV', struct('D_CDF',zeros(Nombre_sim,Nombre_modele)), ...
    'Tps_A', struct('D_CDF',zeros(Nombre_sim,Nombre_modele))),1,7);
for i = 1:17 % On test modèle par modèle
  % Chaque modèle est testé sur avec 7 échantillons
  % de validation et d'entraînement.
  for CV = 1:7
    %=========
    if CV == 2
      Longueur_Sim = 0.8986;
    else
      Longueur_Sim = 1;
    end
    %=========
    %========= Boucle de simulation générant 400 simulations
    % pour chaque bloc de validation croisée
    for R=1:Nombre_sim
      switch i % Permet de sélectionner le modèle à simuler
       case 1 % CSMTD Classique
       WS =  1825;% Taille de fenêtre en jours
       % Temps entre deux changements de paramètres
       Longueur_inter = ceil(Longueur_Sim*365);
       CV7(CV).Simulation(R) = CSMTD(Training_set(CV).Debit,WS,Longueur_Sim,Longueur_inter);
       case 2 % CMTD Classique
       WS = 1825;
       Longueur_inter = ceil(Longueur_Sim*365);
       CV7(CV).Simulation(R) = CMTD(Training_set(CV).Debit,WS,Longueur_Sim,Longueur_inter);
       case 3  % CSMTD & PNP : fenêtre de 1 an
       WS = 365;
       Longueur_inter = ceil(Longueur_Sim*365);
       CV7(CV).Simulation(R) = CSMTD(Training_set(CV).Debit,WS,Longueur_Sim,Longueur_inter);
       case 4 % CSMTD & PNP : fenêtre de 6 mois
       WS = 180;
       Longueur_inter = 180;
       CV7(CV).Simulation(R) = CSMTD(Training_set(CV).Debit,WS,Longueur_Sim,Longueur_inter);
       case 5 % CSMTD & PNP : fenêtre de 3 mois
       WS = 90;
       Longueur_inter = 90;
       CV7(CV).Simulation(R) = CSMTD(Training_set(CV).Debit,WS,Longueur_Sim,Longueur_inter);
       case 6 % CSMTD & PNP : fenêtre de 2 mois
       WS = 60;
       Longueur_inter = 60;
       CV7(CV).Simulation(R) = CSMTD(Training_set(CV).Debit,WS,Longueur_Sim,Longueur_inter);
       case 7 % CSMTD & PNP : fenêtre de 1 mois
       WS = 30;
       Longueur_inter = 30;
       CV7(CV).Simulation(R) = CSMTD(Training_set(CV).Debit,WS,Longueur_Sim,Longueur_inter);
       case 8 % CMTD & PNP : fenêtre de 1 an
       WS = 365;
       Longueur_inter = ceil(Longueur_Sim*365);
       CV7(CV).Simulation(R) = CMTD(Training_set(CV).Debit,WS,Longueur_Sim,Longueur_inter);
       case 9 %CMTD & PNP : fenêtre de 6 mois
       WS = 180;
       Longueur_inter = 180;
       CV7(CV).Simulation(R) = CMTD(Training_set(CV).Debit,WS,Longueur_Sim,Longueur_inter);
       case 10 %CMTD & PNP : fenêtre de 3 mois
       WS = 90;
       Longueur_inter = 90
       CV7(CV).Simulation(R) = CMTD(Training_set(CV).Debit,WS,Longueur_Sim,Longueur_inter);
       case 11 %CMTD & PNP : fenêtre de 2 mois
       WS = 60;
       Longueur_inter = 60;
       CV7(CV).Simulation(R) = CMTD(Training_set(CV).Debit,WS,Longueur_Sim,Longueur_inter);
       case 12 %CMTD & PNP : fenêtre de 1 mois
       WS = 30;
       Longueur_inter = 30;
       CV7(CV).Simulation(R) = CMTD(Training_set(CV).Debit,WS,Longueur_Sim,Longueur_inter);
       case 13 %BBM : bloc de 6 mois
       BS = 180;%  Taille de Bloc en jours
       CV7(CV).Simulation(R) = Boot_mn(Training_set(CV).Debit,Longueur_Sim,BS);
       case 14 %BBM : bloc de 3 mois
       BS = 90;
       CV7(CV).Simulation(R) = Boot_mn(Training_set(CV).Debit,Longueur_Sim,BS);
       case 15 %BBM : bloc de 2 mois
       BS = 60;
       CV7(CV).Simulation(R) = Boot_mn(Training_set(CV).Debit,Longueur_Sim,BS);
       case 16 %BBM : bloc de 1 mois
       BS = 30;
       CV7(CV).Simulation(R) = Boot_mn(Training_set(CV).Debit,Longueur_Sim,BS);
       case 17 %BBM : bloc de 6 jours
       BS = 6;
       CV7(CV).Simulation(R) = Boot_mn(Training_set(CV).Debit,Longueur_Sim,BS);
```

```
164         end
165         % Calcul des différence dans le nombre total de changement d'état
166         %=============
167         CV_Proc(CV).BP.Diff_Nbr(R,i) = ...
168          sum(CV7(CV).Simulation(R).BP(1,1:ceil(Longueur_Sim*365)))...
169          -sum(Test_set(CV).Transition(1,1:ceil(Longueur_Sim*365)));
170         CV_Proc(CV).MP.Diff_Nbr(R,i) = ...
171          sum(CV7(CV).Simulation(R).MP(1,1:ceil(Longueur_Sim*365)))...
172          -sum(Test_set(CV).Transition(2,1:ceil(Longueur_Sim*365)));
173         CV_Proc(CV).PhA.Diff_Nbr(R,i) = ...
174          sum(CV7(CV).Simulation(R).PhA(1,1:ceil(Longueur_Sim*365)))...
175          -sum(Test_set(CV).Transition(3,1:ceil(Longueur_Sim*365)) );
176         CV_Proc(CV).PhD.Diff_Nbr(R,i) = ...
177          sum(CV7(CV).Simulation(R).PhD(1,1:ceil(Longueur_Sim*365)))...
178          -sum(Test_set(CV).Transition(4,1:ceil(Longueur_Sim*365)));
179         CV_Proc(CV).F.Diff_Nbr(R,i) = ...
180          sum(CV7(CV).Simulation(R).F(1,1:ceil(Longueur_Sim*365)))...
181          -sum(Test_set(CV).Transition(5,1:ceil(Longueur_Sim*365)));
182         CV_Proc(CV).MR.Diff_Nbr(R,i) = ...
183          sum(CV7(CV).Simulation(R).MR(1,1:ceil(Longueur_Sim*365)))...
184          -sum(Test_set(CV).Transition(6,1:ceil(Longueur_Sim*365)));
185         %=============
186         % Calcul des distances de Kolmogorov-Smirnov entre CDF de temps de maintien
187         %=============
188         [h, p,CV_Proc(CV).Tps_MN.KS(R,i)] = kstest2(CV5(CV).Simulation(R).MN(:,1), Test_set(CV).Tps_MN(:,1));
189         [h, p,CV_Proc(CV).Tps_MV.KS(R,i)] = kstest2(CV5(CV).Simulation(R).MV(:,1), Test_set(CV).Tps_MV(:,1));
190         [h, p,CV_Proc(CV).Tps_A.KS(R,i)] = kstest2(CV5(CV).Simulation(R).A(:,1), Test_set(CV).Tps_A(:,1));
191         %=============
192         % Calcul des distances de Kolmogorov-Smirnov entre CDF du nombre de changements d'état par jour
193         %=============
194         [h, p,CV_Proc(CV).BP.KS(R,i)] = kstest2(CV5(CV).Simulation(R).BP(1,1:round(Longueur_Sim*365)), ...
                Test_set(CV).Transition(1,1:round(Longueur_Sim*365)));
195         [h, p,CV_Proc(CV).MP.KS(R,i)] = kstest2(CV5(CV).Simulation(R).MP(1,1:round(Longueur_Sim*365)), ...
                Test_set(CV).Transition(2,1:round(Longueur_Sim*365)));
196         [h, p,CV_Proc(CV).PhA.KS(R,i)] = kstest2(CV5(CV).Simulation(R).PhA(1,1:round(Longueur_Sim*365)), ...
                Test_set(CV).Transition(3,1:round(Longueur_Sim*365)));
197         [h, p,CV_Proc(CV).PhD.KS(R,i)] = kstest2(CV5(CV).Simulation(R).PhD(1,1:round(Longueur_Sim*365)), ...
                Test_set(CV).Transition(4,1:round(Longueur_Sim*365)));
198         [h, p,CV_Proc(CV).F.KS(R,i)] = kstest2(CV5(CV).Simulation(R).F(1,1:round(Longueur_Sim*365)), ...
                Test_set(CV).Transition(5,1:round(Longueur_Sim*365)));
199         [h, p,CV_Proc(CV).MR.KS(R,i)] = kstest2(CV5(CV).Simulation(R).MR(1,1:round(Longueur_Sim*365)), ...
                Test_set(CV).Transition(6,1:round(Longueur_Sim*365)));
200         %=============
201       end
202       % Calcul du nombre cumulé de changement d'état par jour
203       CV7(CV).Cumule = CoEvCu(CV7(CV).Simulation, Nombre_sim,Longueur_Sim);
204     end
205     % Enregistrement des simulations
206     eval([ 'CV7' num2str(i) 'M' num2str(c) '= CV7;' ]);
207     save([ 'CV7' num2str(i) 'M' num2str(c)],[ 'CV7' num2str(i) 'M' num2str(c)]);
208   end
209   % Enregistrement des performances
210   eval(['CV_Proc' num2str(c) ' = CV_Proc;' ]);
211   eval(['Training_set' num2str(c) ' = Training_set;' ]);
212   eval(['Test_set' num2str(c) ' = Test_set;' ]);
213   save(['CV_Proc' num2str(c) ],[ 'CV_Proc' num2str(c) ]);
214   save(['Training_set' num2str(c) ],[ 'Training_set' num2str(c) ]);
215   save(['Test_set' num2str(c) ],[ 'Test_set' num2str(c) ]);
```

## 2 Comptage du nombre de changements d'état

```
1   %% Comptage des changements d'état
2   % Pour chaque groupe on calcul le nombre de transitions par mois et par
3   % années
4   % Les variables d'entrées sont :
5   % - le vecteur contenant les valeurs de débit
6   % - l'intervalle pour lequel on veut le nombre de changement d'état
7   %   jour = 1, si l'on veut le nombre de changements d'état par jour,
8   %   jour = 30, si l'on veut le nombre de changements d'état par mois.
9
10  function Transition = Comptage_tran (Debit, jour)
11    % Taille de la fenêtre dans laquelle nous allons
12    %compter le nombre de changements d'état
13    taille = jour*24*60/5;
14    %Longueur du vecteur contenant les valeurs de débit
15    longueur =  length(Debit);
16    % Nombre de fenêtre nécessaire pour couvrir
17    %la longueur de vecteur
18    Nombre = round(longueur/taille);
19    % Détermination de la limite entre MN et MV
20    % On peut le faire de deux façons : soit on connaît le débit limite et
21    % on entre sa valeur dans Debit_lim, soit on ne connaît que le débit
22    % nominal, le débit limite est alors Debit_Nom*0.75;
23    Debit_Nom = 236;
24    Debit_lim = 169.4; % ou = 0.8*Debit_Nom
25    X = zeros(9,Nombre);
```

```
26
27    %======== Boucle de parcours du vecteur de débit
28    for N = 1:Nombre;
29      %========== Initialisation des variables
30      clear ind_SNL ind_marche ind_arret BP MP PhA PhD F MR;
31      NbrBP = 0;
32      NbrMP = 0;
33      NbrPhA = 0;
34      NbrPhD = 0;
35      NbrMR = 0;
36      NbrF = 0;
37      % Indices de début et de fin de la fenêtre de comptage
38      Fin = N*taille;
39      Debut = Fin - taille + 1;
40      % Dans le cas où la dernière fenêtre dépasse la fin du vecteur,
41      % celle-ci sera raccourcie.
42      if Fin > longueur
43        Fenetre = Debit(Debut:longueur);
44      else
45        Fenetre = Debit(Debut:Fin);
46      end
47      % On stocke dans ind_SNL les indices de positions auxquelles on trouve
48      % des valeurs de débit inférieure au débit limite mais non nulle.
49      ind_SNL =find(Fenetre<=Debit_lim & Fenetre>0);
50      x=length(ind_SNL);
51      % Boucle de parcour de ind_SNL
52      for SNL=1:x
53        % on place dans la variable etat_prec, la valeur de
54        % l'indice précédent directement celui de la MV
55        % S'il s'agit de la première valeur de la fenêtre
56        if ind_SNL(SNL)-1 == 0;
57        % On vérifie qu'il ne s'agit pas de la première valeur du vecteur
58         if Debut-1 <= 0
59         % si c'est le cas, on fait l'hypothèse que le GTA était précédemment en MV
60           etat_prec = 20 ;
61         else
62           etat_prec =  Debit(Debut-1);
63         end
64        else
65         etat_prec = Fenetre(ind_SNL(SNL)-1);
66        end
67        % on place dans la variable etat_suiv, la valeur de
68        % l'indice suivant directement celui de la MV
69        % S'il s'agit de la dernière valeur de la fenêtre
70        if ind_SNL(SNL)+1 > length(Fenetre);
71        % On vérifie qu'il ne s'agissent pas de la dernière valeur du vecteur
72         if Fin+1 > length(Debit)
73         % si c'est le cas, on fait l'hypothèse que le GTA sera en MV
74           etat_suiv = 20;
75         else
76           etat_suiv = Debit(Fin+1);
77         end
78        else
79         etat_suiv =  Fenetre(ind_SNL(SNL)+1);
80        end
81        if etat_prec == 0;
82        % Si au pas de temps précedent le GTA est à l'arrêt et qu'au pas de temps suivant il est en marche normale
83        % alors on considère que le GTA subit une phase de démarrage, sinon c'est une mise en rotation.
84         if etat_suiv > Debit_lim;
85           NbrPhD = NbrPhD+1;
86         else
87           NbrMR = NbrMR+1;
88         end
89        elseif etat_prec>Debit_lim;
90        % Si au pas de temps précédent le GTA est en marche normale et qu'au pas de temps suivant il est à l'arrêt
91        % alors on considère que le GTA à subit une phase d'arrêt, sinon c'est une baisse de puissance.
92         if etat_suiv==0;
93           NbrPhA = NbrPhA+1;
94         else
95           NbrBP = NbrBP+1;
96         end
97        end
98      end
99      % On stocke dans ind_marche les indices de positions auxquelles
100     % on trouve des valeurs de débit supérieure au débit limite.
101     ind_marche =find(Fenetre>Debit_lim);
102     y=length(ind_marche);
103     % Boucle de parcours de ind_marche
104     for marche=1:y;
105       if ind_marche(marche)-1 == 0;
106        if Debut-1 <= 0
107          etat_prec = Debit_Nom ;
108        else
109          etat_prec =  Debit(Debut-1);
110        end
111       else
112        etat_prec = Fenetre(ind_marche(marche)-1);
113       end
114       if ind_marche(marche)-2 <= 0;
115        if Debut-2 <= 0
116          etat_prec2 = Debit_Nom ;
```

```
117         else
118           etat_prec2 = Debit(Debut-2);
119         end
120        else
121         etat_prec2 = Fenetre(ind_marche(marche)-2);
122        end
123        if etat_prec>0 && etat_prec<Debit_lim;
124        % Si au pas de temps précedent le GTA est en marche à vide et qu'a deux pas de temps en arrière le GTA est en ...
               arrêt
125        % alors on ne compte aucun changement d'état, car celui ci a déjà été compté dans le parcours de ind_SNL.
126        % Sinon c'est une montée en puissance.
127         if etat_prec2 == 0;
128         else
129           NbrMP = NbrMP+1;
130         end
131        elseif etat_prec == 0;
132        % Si au pas de temps précedent le GTA est à l'arrêt alors on compte une phase de démarrage
133         NbrPhD = NbrPhD+1;
134        end
135      end
136      % On stocke dans ind_arret les indices de positions auxquelles
137      % on trouve des valeurs de débit nulle.
138      ind_arret =find(Fenetre==0);
139      z=length(ind_arret);
140      % Boucle de parcour de ind_arret
141      for Arret=1:z
142        if ind_arret(Arret)-1 == 0;
143         if Debut-1 <= 0
144           etat_prec = 0;
145         else
146           etat_prec =  Debit(Debut-1);
147         end
148        else
149         etat_prec = Fenetre(ind_arret(Arret)-1);
150        end
151        if ind_arret(Arret)-2 <= 0;
152         if Debut-2 <= 0
153           etat_prec2 = 0;
154         else
155           etat_prec2 =  Debit(Debut-2);
156         end
157        else
158         etat_prec2 = Fenetre(ind_arret(Arret)-2);
159        end
160        if etat_prec>0 && etat_prec<Debit_lim;
161        % Si au pas de temps précedent le GTA est en marche à vide et qu'a deux pas de temps en arrière le GTA est en ...
               marche normale
162        % alors on ne compte aucun changement d'état, car celui ci a déjà été compté dans le parcours de ind_SNL.
163        % Sinon c'est un freinage.
164         if etat_prec2>Debit_lim;
165         else
166           NbrF = NbrF+1;
167         end
168        elseif etat_prec>Debit_lim;
169        % Si au pas de temps précedent le GTA est en marche alors on compte une phase d'arrêt.
170         NbrPhA = NbrPhA+1;
171        end
172      end
173      X(1,N) = NbrBP;
174      X(2,N) = NbrMP;
175      X(3,N) = NbrPhA;
176      X(4,N) = NbrPhD;
177      X(5,N) = NbrF;
178      X(6,N) = NbrMR;
179    end
180    Transition = X;
181  end
```

## 3 Comptage des temps de maintien

```
1   %% Comptage des temps
2   % Pour chaque groupe on détermine la distribution des temps passés dans
3   % chacun des états
4   % La variable d'entrée est le vecteur contenant les valeurs de débit
5
6   function [Tps_MN Tps_MV Tps_A] = Comptage_tps(Debit)
7
8       % Détermination de la limite entre MN et MV
9       % On peut le faire de deux façons : soit on connaît le débit limite et
10      % on entre sa valeur dans Debit_lim, soit on ne connaît que le débit
11      % nominal, le débit limite est alors Debit_Nom*0.75;
12      Debit_Nom = 236;
13      Debit_lim = 169.4;
14
15      Tps_MN = zeros(500,2);
16      Tps_MV = zeros(500,2);
```

```
Tps_A = zeros(250,2);

% On stocke dans ind_SNL les indices de positions auxquelles on trouve
% des valeurs de débit inférieure au débit limite mais non nulle.
ind_SNL = find(Debit<=Debit_lim & Debit>0);
x=length(ind_SNL);
% Boucle de parcours de ind_SNL
if x>1
    comp = 1;
    tps=1;
    for MV=1:x-1
    % Si la différence entre deux indices adjacents est égale à 1,
    % c'est que le GTA n'a pas changé d'état, on ajoute 1 à la
    % variable tps qui correspond au temps de maintien.
        if (ind_SNL(MV+1)-ind_SNL(MV))==1;
            tps = tps+1;
        % Si la différence est >1 c'est qu'il y a eu changement d'état
        % on multiplie tps par 5 pour avoir le temps de maintien en
        % minute et on stocke dans Tps_MV le temps.
        else
            Tps_MV(comp,1) = tps*5;
            Tps_MV(comp,2) = ind_SNL(MV);
            comp = comp+1;
            tps=1;
        end
    end
    Tps_MV(comp,1) = tps*5;
    Tps_MV(comp,2) = ind_SNL(MV);
elseif x==1
    Tps_MV(1,1) = 5;
    Tps_MV(1,2) = ind_SNL(1);
end
% On stocke dans ind_MN les indices de positions auxquelles on trouve
% des valeurs de débit supérieur au débit limite.
ind_MN = find(Debit>Debit_lim);
x=length(ind_MN);
if x>1
    comp = 1;
    tps=1;
    for MN=1:x-1
        if (ind_MN(MN+1)-ind_MN(MN))==1;
            tps = tps+1;
        else
            Tps_MN(comp,1) = tps*5;
            Tps_MN(comp,2) = ind_MN(MN);
            comp = comp+1;
            tps=1;
        end
    end
    Tps_MN(comp,1) = tps*5;
    Tps_MN(comp,2) = ind_MN(MN);
elseif x==1
    Tps_MN(1,1) = 5;
    Tps_MN(1,2) = ind_MN(1);
end
% On stocke dans ind_A les indices de positions auxquelles
% on trouve des valeurs de débit nulle.
ind_A = find(Debit==0);
x=length(ind_A);
if x>1
    comp = 1;
    tps=1;
    for A=1:x-1
        if (ind_A(A+1)-ind_A(A))==1;
            tps = tps+1;
        else
            Tps_A(comp,1) = tps*5;
            Tps_A(comp,2) = ind_A(A);
            comp = comp+1;
            tps=1;
        end
    end
    Tps_A(comp,1) = tps*5;
    Tps_A(comp,2) = ind_A(A);
elseif x==1
    Tps_A(1,1) = 5;
    Tps_A(1,2) = ind_A(1);
end
%Permet de ré-ajuster la longueur des tableaux contenant les
%temps de maintien, dans le cas on la déclaration de variable l'a
%surestimée
if isempty(find(Tps_MN(:,2)>0,1))
else
    MN = Tps_MN(Tps_MN(:,2)>0,:);
    Tps_MN = [];
    Tps_MN = MN;
end
if isempty(find(Tps_MV(:,2)>0,1))
else
    MV = Tps_MV(Tps_MV(:,2)>0,:);
    Tps_MV = [];
```

```
108         Tps_MV = MV;
109     end
110     if isempty(find(Tps_A(:,2)>0,1))
111     else
112         A = Tps_A(Tps_A(:,2)>0,:);
113         Tps_A = [];
114         Tps_A = A;
115     end
116 end
```

## 4 Calcul des paramètres des CSMTD

```
1  %% Calcul des paramètres des CSMTD
2  % Cette fonction permet à la fois de calculer les paramètres des CSMTD et
3  % de receuillir les résultats de simulation avant de les transmettres au
4  % programme principal.
5  % Les variables d'entrées sont :
6  % - le vecteur contenant les valeurs de débit
7  % - la taille de fenêtre pour la PNP
8  % - la longueur de simulation
9  % - la longueur entre deux changements de paramètre (souvent égale à la taille de fenêtre).
10
11 function M = CSMTD(DEBIT,Size,Longueur_Sim,Long_inter)
12     % Défini le nombre de changement de paramètres nécessaires
13     %Initialisation des variables
14     Nbr_chgt = ceil(Longueur_Sim*365/Long_inter);
15     Transitions = repmat(struct('BP',zeros(1,Long_inter),...
16     'MP',zeros(1,Long_inter),'PhA',zeros(1,Long_inter),...
17     'PhD',zeros(1,Long_inter),'F',zeros(1,Long_inter),...
18     'MR',zeros(1,Long_inter)),1,Nbr_chgt);
19     Debut = 0;
20     Fin=0;
21     Etat_initial = [1 ;1];
22     % Boucle de calcul et de simulation
23     for SMC = 1:Nbr_chgt
24         %============Définition de la fenêtre utilisée pour le calcul des paramètres
25    % Un nombre aléatoire compris entre 0 et le nombre total de mois est pigé
26         mois_debut =  randi(fix(length(DEBIT)*5/(60*24*30)));
27         ind_debut = mois_debut*8640 + 1;
28         ind_fin = ind_debut + Size*24*60/5;
29         if ind_fin <= length(DEBIT)
30             Deb = DEBIT(ind_debut:ind_fin) ;
31         else
32             Manquant = ind_fin - length(DEBIT);
33             ind_fin = Manquant+1;
34             Deb = [DEBIT(ind_debut:length(DEBIT));DEBIT(1:ind_fin)];
35         end
36         %================
37         %================ Calcul des paramètres
38    % Comptage du nombre de changements d'état dans la fenêtre
39         Trans = Comptage_tran(Deb, 1);
40         Tr = ones(6,1);
41         %=========Calcul du nombre total de chacune des transitions
42         for t=1:6;
43             Tr(t,1)= sum(Trans(t,:));
44         end
45         %=========
46         % Calcul des temps de maintien
47         [Tps_MN Tps_MV Tps_A] = Comptage_tps(Deb);
48         T12 = Tr(1,1);
49         T13 = Tr(3,1);
50         T21 = Tr(2,1);
51         T23 = Tr(5,1);
52         T31 = Tr(4,1);
53         T32 = Tr(6,1);
54         Total1 = T12 + T13;
55         Total2 = T21 + T23;
56         Total3 = T31 + T32;
57         P12 = T12/Total1;
58         P13 = T13/Total1;
59         P21 = T21/Total2;
60         P23 = T23/Total2;
61         P31 = T31/Total3;
62         P32 = T32/Total3;
63         Mat = [0 P12 P13;P21 0 P23;P31 P32 0];
64         %================
65         % Si la longueur de simulation avant changement de paramètre est
66         % supérieur à la longueur totale de simualtion, celle est diminuée
67         if Long_inter*SMC > ceil(Longueur_Sim*365)
68             Long_inter = ceil(Longueur_Sim*365) - Long_inter*(SMC-1);
69         end
70         %=====================================================Simulation
71         [Transitions(SMC).BP Transitions(SMC).MP Transitions(SMC).PhA Transitions(SMC).PhD Transitions(SMC).F ...
               Transitions(SMC).MR TPS(SMC).MN TPS(SMC).MV TPS(SMC).A, Etat_final] = ...
               Simulation_CSMTD_an(Mat,Tps_MN,Tps_MV,Tps_A,Etat_initial,Long_inter);
72         %=====================================================
```

```
73          clear Tps_MN Tps_MV Tps_A
74          %======================= Stockage des résultats de simulations entre chaque changements de paramètres
75          Debut = Fin+1;
76          Fin = Debut + Long_inter-1;
77          % Stockage nombre de changements d'état.
78          M.BP(1,Debut:Fin) = Transitions(SMC).BP;
79          M.MP(1,Debut:Fin) = Transitions(SMC).MP;
80          M.PhA(1,Debut:Fin) = Transitions(SMC).PhA;
81          M.PhD(1,Debut:Fin) = Transitions(SMC).PhD;
82          M.F(1,Debut:Fin) = Transitions(SMC).F;
83          M.MR(1,Debut:Fin) = Transitions(SMC).MR;
84          % Cette boucle permet de ne pas couper un temps de maintien entre deux changements de paramètres.
85          if SMC==1
86              T_deb_mn = 1;
87              T_deb_mv = 1;
88              T_deb_a = 1;
89              Taille_mn_tps = size(TPS(SMC).MN);
90              T_fin_mn = T_deb_mn + Taille_mn_tps(1)-1;
91              M.MN(T_deb_mn:T_fin_mn,1) = TPS(SMC).MN(:,1);
92              Taille_mv_tps = size(TPS(SMC).MV);
93              T_fin_mv = T_deb_mv + Taille_mv_tps(1)-1;
94              M.MV(T_deb_mv:T_fin_mv,1) = TPS(SMC).MV(:,1);
95              Taille_a_tps = size(TPS(SMC).A);
96              T_fin_a = T_deb_a +Taille_a_tps(1)-1;
97              M.A(T_deb_a:T_fin_a,1) = TPS(SMC).A(:,1);
98          else
99              Taille_mn = size(M.MN);
100             T_deb_mn = Taille_mn(1)+1;
101             Taille_mv = size(M.MV);
102             T_deb_mv = Taille_mv(1)+1;
103             Taille_a = size(M.A);
104             T_deb_a = Taille_a(1)+1;
105             % La simulation va continuer avec le même état dans
106         % lequel elle était avant le changement de paramètres.
107         % Dépendant de l'état initial, on ajoute le premier
108         % temps de maintien obtenu avec les nouveaux paramètres
109             % au dernier temps de maintien obtenue avec les paramètres précédents.
110             if Etat_initial(1)==1;
111                 Taille_mn_tps = size(TPS(SMC).MN);
112                 T_fin_mn = T_deb_mn + Taille_mn_tps(1)-1;
113                 M.MN(T_deb_mn-1) = M.MN(T_deb_mn-1) + TPS(SMC).MN(1,1);
114                 M.MN(T_deb_mn:T_fin_mn-1,1) = TPS(SMC).MN(2:Taille_mn_tps,1);
115                 Taille_mv_tps = size(TPS(SMC).MV);
116                 T_fin_mv = T_deb_mv + Taille_mv_tps(1)-1;
117                 M.MV(T_deb_mv:T_fin_mv,1) = TPS(SMC).MV(:,1);
118                 Taille_a_tps = size(TPS(SMC).A);
119                 T_fin_a = T_deb_a +Taille_a_tps(1)-1;
120                 M.A(T_deb_a:T_fin_a,1) = TPS(SMC).A(:,1);
121             elseif Etat_initial(1)==2;
122                 Taille_mv_tps = size(TPS(SMC).MV);
123                 T_fin_mv = T_deb_mv + Taille_mv_tps(1)-1;
124                 M.MV(T_deb_mv-1) = M.MV(T_deb_mv-1) + TPS(SMC).MV(1,1);
125                 M.MV(T_deb_mv:T_fin_mv-1,1) = TPS(SMC).MV(2:Taille_mv_tps,1);
126                 Taille_mn_tps = size(TPS(SMC).MN);
127                 T_fin_mn = T_deb_mn + Taille_mn_tps(1)-1;
128                 M.MN(T_deb_mn:T_fin_mn,1) = TPS(SMC).MN(:,1);
129                 Taille_a_tps = size(TPS(SMC).A);
130                 T_fin_a = T_deb_a +Taille_a_tps(1)-1;
131                 M.A(T_deb_a:T_fin_a,1) = TPS(SMC).A(:,1);
132             elseif Etat_initial(1)==3;
133                 Taille_mn_tps = size(TPS(SMC).MN);
134                 T_fin_mn = T_deb_mn + Taille_mn_tps(1)-1;
135                 M.MN(T_deb_mn:T_fin_mn,1) = TPS(SMC).MN(:,1);
136                 Taille_mv_tps = size(TPS(SMC).MV);
137                 T_fin_mv = T_deb_mv + Taille_mv_tps(1)-1;
138                 M.MV(T_deb_mv:T_fin_mv,1) = TPS(SMC).MV(:,1);
139                 Taille_a_tps = size(TPS(SMC).A);
140                 T_fin_a = T_deb_a + Taille_a_tps(1)-1;
141                 M.A(T_deb_a-1) = M.A(T_deb_a-1) + TPS(SMC).A(1,1);
142                 M.A(T_deb_a:T_fin_a-1,1) = TPS(SMC).A(2:Taille_a_tps,1);
143             end
144             Etat_initial = Etat_final;
145         end
146         %=======================
147     end
148     %Permet de ré-ajuster la longueur des tableaux contenant les
149     %temps de maintien, dans le cas on la déclaration de variable l'ai
150     %surestimée
151     MN = M.MN;
152     M.MN = [];
153     M.MN = MN(MN(:,1)>0,:);
154     MV = M.MV;
155     M.MV = [];
156     M.MV = MV(MV(:,1)>0,:);
157     A = M.A;
158     M.A = [];
159     M.A = A(A(:,1)>0,:);
160 end
```

# 5 Simulation des CSMTD

```
%% Simulation CSMTD
% Cette fonction permet de générer des séries temporelles de longueur Long_Sim
% Les variables d'entrées sont :
% - la matrice de probabilité changements d'état
% - les distributions de temps de maintien dans chaque état
% - L'état de départ de la simulation
% - la longueur de simulation

function [BP MP PhA PhD F MR MN MV A Final] = Simulation_CSMTD_an(Matrice,Distrib_MN,Distrib_MV,Distrib_A,...
Initial,Long_Sim)
    % Initialisation des variables
    s=1;
    etat = Initial(1,1);
    etat_prec = Initial(2,1);
    Nbr_MP = 0;
    Nbr_BP = 0;
    Nbr_PhA = 0;
    Nbr_MR = 0;
    Nbr_F = 0;
    Nbr_PhD = 0;
    tps_MN = 1;
    tps_MV = 1;
    tps_A = 1
    Tps_MN = zeros(500,3);
    Tps_MV = zeros(500,3);
    Tps_A = zeros(250,3);
    BP = zeros(1,Long_Sim);
    MP = zeros(1,Long_Sim);
    PhA = zeros(1,Long_Sim);
    PhD = zeros(1,Long_Sim);
    F = zeros(1,Long_Sim);
    MR = zeros(1,Long_Sim);
    format long
    Limite = Long_Sim*24*60;

    % Dans le cas où la fenêtre de débit utilisé pour obtenir les
    % paramètres ne contient aucun changement d'état, la matrice va
    % contenir des valeurs non numérique
    MZ = find(Matrice~= 0);
    T = isnan(Matrice);
    TU = find(T==1);
    % Si les valeurs non nulle de la matrice sont tous des NaN (Not a Number)
    % alors c'est qu'il n'y a pas de changements d'état
    if size(TU)==size(MZ)
        % Les distributions de temps restent donc les mêmes, sachant qu'une
        % seule sera non nulle.
        MN = Distrib_MN(:,1);
        MV = Distrib_MV(:,1);
        A = Distrib_A(:,1);
        % En détectant quelle distribution est non nulle, on sait dans
        % quelle état doit rester la série.
        if MN ~= 0
            Final(1,1) = 1;
            Final(2,1) = etat;
            if etat == 2
                MP(1,1) = 1;
            elseif etat == 3
                PhD(1,1) = 1;
            end
        elseif MV ~= 0
            Final(1,1) = 2;
            Final(2,1) = etat;
            if etat == 1
                BP(1,1) = 1;
            elseif etat == 3
                MR(1,1) = 1;
            end
        elseif A ~= 0
            Final(1,1) = 3;
            Final(2,1) = etat;
            if etat == 1
                PhA(1,1) = 1;
            elseif etat == 2
                F(1,1) = 1;
            end
        end
    else
        % Si l'on ne se situe pas dans le cas particulier décrit
        % précédement, les simulations se déroule normalement.
        % Tranformation des valeurs de temps de maintien en CDF empirique.
        [D_MN(:,1) D_MN(:,2)] = ecdf(Distrib_MN(:,1));
        [D_MV(:,1) D_MV(:,2)] = ecdf(Distrib_MV(:,1));
        [D_A(:,1) D_A(:,2)] = ecdf(Distrib_A(:,1));
        % Boucle de simulation
        while s < Limite
            %========== Détermination du temps de maintien dans l'état présent
```

```
87          % Pigeage d'un nombre aléatoire
88          % entre 0 et 1
89               f = rand(1,1);
90               if etat ==1;
91          % On détermine dans la distribution quelle est la probabilité la plus proche du nombre aléatoire
92                    [h T_mn] = min(abs(D_MN(:,1)-f));
93          % Si f est inférieur à cette probabilité, alors le temps de de maintien sera celui correspondant à cette ...
                 probabilité, sinon ce sera la temps correspondant à la probabilité supérieure.
94                    if f < D_MN(T_mn,1)
95                        temps = D_MN(T_mn,2);
96                    else
97                        temps = D_MN(T_mn+1,2);
98                    end
99                    autre_etat1 = 2;
100                   autre_etat2 = 3;
101              elseif etat == 2;
102                   [h T_mv] = min(abs(D_MV(:,1)-f));
103                   if f < D_MV(T_mv,1)
104                       temps = D_MV(T_mv,2);
105                   else
106                       temps = D_MV(T_mv+1,2);
107                   end
108                   autre_etat1 = 1;
109                   autre_etat2 = 3;
110              else
111                   [h T_a] = min(abs(D_A(:,1)-f));
112                   if f < D_A(T_a,1)
113                       temps = D_A(T_a,2);
114                   else
115                       temps = D_A(T_a+1,2);
116                   end
117                   autre_etat1 = 1;
118                   autre_etat2 = 2;
119              end
120              %==========
121              %========== Détermination de l'état futur
122              f = rand(1,1); % Pigeage d'un nombre aléatoire entre 0 et 1
123              % On se place sur la ligne de la matrice correspondant à l'état présent. Si f est inférieur à la ...
                      probabilité de changer vers i alors l'état suivant est i, sinon ce sera j.
124              if f<Matrice(etat,autre_etat1)
125                   etat_suivant = autre_etat1;
126              else
127                   etat_suivant = autre_etat2;
128              end
129              % Cette boucle compte, au fur et à mesure de la simulation, le nombre de changements d'état.
130              if etat == 1;
131                   % Si la longueur de la simulation est atteinte, le temps dans le temps présent est raccourci et les ...
                          deux derniers état sont enregistrés
132                   if s+temps > Limite
133                       temps = Limite-s;
134                       Final(1,1) = etat;
135                       Final(2,1) = etat_prec;
136                   else
137                       % Si le GTA est en marche normale et que l'état suivant est la marche à vide, on ne compte aucun ...
                              changement d'état car il sera compté plus tard. Sinon on compte une phase d'arrêt.
138                       if etat_suivant == 2;
139                       else
140                           Nbr_PhA = Nbr_PhA + 1;
141                       end
142                   end
143                   % Le temps dans l'état présent est stocké
144                   Tps_MN(tps_MN,1) = temps;
145                   Tps_MN(tps_MN,2) = (s-1)*5;
146                   tps_MN = tps_MN + 1;
147              elseif etat == 2;
148                   if s+temps > Limite
149                       temps = Limite-s;
150                       Final(1,1) = etat;
151                       Final(2,1) = etat_prec;
152                   else
153                   % Si le GTA est en marche à vide, qu'il va y rester 5
154                   % minutes, que l'état suivant est la marche normale et
155                   % que l'état précédent est l'arrêt alors on compte une
156                   % phase d'arrêt. Si le temps est supérieur à 5 minutes
157                   % alors on compte une mise rotaation et une montée en
158                   % puissance. Si l'état précédent est la marche normale,
159                   % quelque le temps en marche à vide on compte une
160                   % montée et une baisse de puissance.
161                       if etat_suivant == 1;
162                           if etat_prec == 3 && temps == 5
163                               Nbr_PhD = Nbr_PhD + 1;
164                           elseif etat_prec == 3 && temps > 5
165                               Nbr_MR = Nbr_MR + 1;
166                               Nbr_MP = Nbr_MP + 1;
167                           elseif etat_prec == 1
168                               Nbr_MP = Nbr_MP + 1;
169                               Nbr_BP = Nbr_BP + 1;
170                           end
171                   % Le même genre de démarche s'applique lorsque
172                   % l'état suivant est l'arrêt.
173                       elseif etat_suivant == 3;
```

```
                        if etat_prec == 1 && temps == 5
                            Nbr_PhA = Nbr_PhA + 1;
                        elseif etat_prec == 1 && temps > 5
                            Nbr_BP = Nbr_BP + 1;
                            Nbr_F = Nbr_F + 1;
                        elseif etat_prec == 3
                            Nbr_F = Nbr_F + 1;
                            Nbr_MR = Nbr_MR + 1;
                        end
                    end
                end
                Tps_MV(tps_MV,1) = temps;
                Tps_MV(tps_MV,2) = (s-1)*5;
                tps_MV = tps_MV + 1;
            elseif etat == 3;
                if s+temps > Limite
                    temps = Limite-s;
                    Final(1,1) = etat;
                    Final(2,1) = etat_prec;
                else
                    % Si le GTA est arrêt et que l'état suivant est la marche normale, on compte une phase de ...
                          démarrage. Sinon on ne compte aucun changement car il est compté avec la marche à vide
                    if etat_suivant == 1;
                        Nbr_PhD = Nbr_PhD + 1;
                    else
                    end
                end
                Tps_A(tps_A,1) = temps;
                Tps_A(tps_A,2) = (s-1)*5;
                tps_A = tps_A + 1;
            end
            %=========
            etat_prec = etat;
            etat = etat_suivant;
            s = s+temps;
            % Le nombre de changements d'état pas jour est stocké
            N = ceil(s/1440);
            BP(1,N) = BP(1,N) + Nbr_BP;
            MP(1,N) = MP(1,N) + Nbr_MP;
            PhA(1,N) = PhA(1,N) + Nbr_PhA;
            PhD(1,N) = PhD(1,N)+ Nbr_PhD;
            F(1,N) = F(1,N) + Nbr_F;
            MR(1,N) = MR(1,N)+ Nbr_MR;
            Nbr_MP = 0;
            Nbr_BP = 0;
            Nbr_PhA = 0;
            Nbr_MR = 0;
            Nbr_F = 0;
            Nbr_PhD = 0;
        end
        %Permet de ré-ajuster la longueur des tableaux contenant les temps de maintien, dans le cas on la ...
              déclaration de variable l'a surestimée
         MN = Tps_MN;
        if isempty(find(Tps_MN(:,1)>0, 1))
        else
            Tps_MN = MN(MN(:,1)>0,:);
            MN = [];
            MN = Tps_MN;
        end
        MV = Tps_MV;
        if isempty(find(Tps_MV(:,1)>0, 1))
        else
            Tps_MV = MV(MV(:,1)>0,:);
            MV = [];
            MV = Tps_MV;
        end
        A = Tps_A;
        if isempty(find(Tps_A(:,1)>0, 1))
        else
            Tps_A = A(A(:,1)>0,:);
            A = [];
            A = Tps_A;
        end
    end
end
```

## 6 Calcul des paramètres des CMTD

```
%% Calcul des paramètres des CMTD
% Cette fonction permet à la fois de calculer les paramètres des CMTD et
% de receuillir les résultats de simulation avant de les transmettres au
% programme principal.
% Les variables d'entrées sont :
% - le vecteur contenant les valeurs de débit
% - la taille de fenêtre pour la PNP
% - la longueur de simulation
```

```
% - la longueur entre deux changements de paramètre (souvent égale à la taille de fenêtre).

% Nous ne commenterons pas entièrement cette fonction car le fonctionnement est le même que pour les CSMTD, seul les ...
      divergence seront commentée
function M = CMTD(DEBIT,Size,Longueur_Sim,Long_inter)
    Nbr_chgt = ceil(Longueur_Sim*365/Long_inter);
    Transitions = repmat(struct('BP',zeros(1,Long_inter),...
    'MP',zeros(1,Long_inter),'PhA',zeros(1,Long_inter),...
    'PhD',zeros(1,Long_inter),'F',zeros(1,Long_inter),...
    'MR',zeros(1,Long_inter)),1,Nbr_chgt);
    Debut = 0;
    Fin=0;
    Etat_initial = [1 ;1];
    Tr = zeros(6,1);

    for SMC = 1:Nbr_chgt
        %=====================
        mois_debut =  randi([0,fix(length(DEBIT)*5/(60*24*30))]);

        ind_debut = mois_debut*8640 + 1;
        ind_fin = ind_debut + Size*24*60/5-1;

        if ind_fin <= length(DEBIT)
            Deb = DEBIT(ind_debut:ind_fin) ;
        else
            Manquant = ind_fin - length(DEBIT);
            ind_fin = Manquant-1;
            Deb = [DEBIT(ind_debut:length(DEBIT));DEBIT(1:ind_fin)];
        end
        %=====================
        %=====================
        Trans = Comptage_tran(Deb, 1);
        Tr = ones(6,1);
        for t=1:6;
            Tr(t,1)= sum(Trans(t,:));
        end
        [Tps_MN Tps_MV Tps_A] = Comptage_tps(Deb);
        % On voit ici une première divergence avec les CSMTD
        % Les temps passé dans chaque état sont considérés comme des
        % transitions de i vers i.
        T11 = (sum(Tps_MN(:,1)))/5;
        T22 = (sum(Tps_MV(:,1)))/5;
        T33 = (sum(Tps_A(:,1)))/5;
        T12 = Tr(1,1);
        T13 = Tr(3,1);
        T21 = Tr(2,1);
        T23 = Tr(5,1);
        T31 = Tr(4,1);
        T32 = Tr(6,1);
        Total1 = T11 + T12 + T13;
        Total2 = T21 + T22 + T23;
        Total3 = T31 + T32 + T33;
        % On a donc des probabilités de transition, l'état du GTA sera
        % ré-évalué à chaque pas de temps
        P11 = T11/Total1;
        P12 = T12/Total1;
        P13 = T13/Total1;
        P21 = T21/Total2;
        P22 = T22/Total2;
        P23 = T23/Total2;
        P31 = T31/Total3;
        P32 = T32/Total3;
        P33 = T33/Total3;
        Mat = [P11 P12 P13;P21 P22 P23;P31 P32 P33];
        if Long_inter*SMC > ceil(Longueur_Sim*365)
            Long_inter = ceil(Longueur_Sim*365) - Long_inter*(SMC-1);
        end
        %=====================================================
        [Transitions(SMC).BP Transitions(SMC).MP Transitions(SMC).PhA Transitions(SMC).PhD Transitions(SMC).F ...
              Transitions(SMC).MR TPS(SMC).MN TPS(SMC).MV TPS(SMC).A, Etat_final] ...
            = Simulation_CMTD_an(Mat,Etat_initial,Long_inter);
        %=====================================================
        clear Tps_MN Tps_MV Tps_A
        Debut = Fin+1;
        Fin = Debut + Long_inter-1;
        M.BP(1,Debut:Fin) = Transitions(SMC).BP;
        M.MP(1,Debut:Fin) = Transitions(SMC).MP;
        M.PhA(1,Debut:Fin) = Transitions(SMC).PhA;
        M.PhD(1,Debut:Fin) = Transitions(SMC).PhD;
        M.F(1,Debut:Fin) = Transitions(SMC).F;
        M.MR(1,Debut:Fin) = Transitions(SMC).MR;
        if SMC==1
            T_deb_mn = 1;
            T_deb_mv = 1;
            T_deb_a = 1;
            Taille_mn_tps = size(TPS(SMC).MN);
            T_fin_mn = T_deb_mn + Taille_mn_tps(1)-1;
            M.MN(T_deb_mn:T_fin_mn,1) = TPS(SMC).MN(:,1);
            Taille_mv_tps = size(TPS(SMC).MV);
            T_fin_mv = T_deb_mv + Taille_mv_tps(1)-1;
            M.MV(T_deb_mv:T_fin_mv,1) = TPS(SMC).MV(:,1);
```

```
            Taille_a_tps = size(TPS(SMC).A);
            T_fin_a = T_deb_a +Taille_a_tps(1)-1;
            M.A(T_deb_a:T_fin_a,1) = TPS(SMC).A(:,1);
        else
            Taille_mn = size(M.MN);
            T_deb_mn = Taille_mn(1)+1;
            Taille_mv = size(M.MV);
            T_deb_mv = Taille_mv(1)+1;
            Taille_a = size(M.A);
            T_deb_a = Taille_a(1)+1;
            if Etat_initial==1;
                Taille_mn_tps = size(TPS(SMC).MN);
                T_fin_mn = T_deb_mn + Taille_mn_tps(1)-1;
                M.MN(T_deb_mn-1) = M.MN(T_deb_mn-1) + TPS(SMC).MN(1,1);
                M.MN(T_deb_mn:T_fin_mn-1,1) = TPS(SMC).MN(2:Taille_mn_tps,1);
                Taille_mv_tps = size(TPS(SMC).MV);
                T_fin_mv = T_deb_mv + Taille_mv_tps(1)-1;
                M.MV(T_deb_mv:T_fin_mv,1) = TPS(SMC).MV(:,1);
                Taille_a_tps = size(TPS(SMC).A);
                T_fin_a = T_deb_a +Taille_a_tps(1)-1;
                M.A(T_deb_a:T_fin_a,1) = TPS(SMC).A(:,1);
            elseif Etat_initial==2;
                Taille_mv_tps = size(TPS(SMC).MV);
                T_fin_mv = T_deb_mv + Taille_mv_tps(1)-1;
                M.MV(T_deb_mv-1) = M.MV(T_deb_mv-1) + TPS(SMC).MV(1,1);
                M.MV(T_deb_mv:T_fin_mv-1,1) = TPS(SMC).MV(2:Taille_mv_tps,1);
                Taille_mn_tps = size(TPS(SMC).MN);
                T_fin_mn = T_deb_mn + Taille_mn_tps(1)-1;
                M.MN(T_deb_mn:T_fin_mn,1) = TPS(SMC).MN(:,1);
                Taille_a_tps = size(TPS(SMC).A);
                T_fin_a = T_deb_a +Taille_a_tps(1)-1;
                M.A(T_deb_a:T_fin_a,1) = TPS(SMC).A(:,1);
            else
                Taille_mn_tps = size(TPS(SMC).MN);
                T_fin_mn = T_deb_mn + Taille_mn_tps(1)-1;
                M.MN(T_deb_mn:T_fin_mn,1) = TPS(SMC).MN(:,1);
                Taille_mv_tps = size(TPS(SMC).MV);
                T_fin_mv = T_deb_mv + Taille_mv_tps(1)-1;
                M.MV(T_deb_mv:T_fin_mv,1) = TPS(SMC).MV(:,1);
                Taille_a_tps = size(TPS(SMC).A);
                T_fin_a = T_deb_a + Taille_a_tps(1)-1;
                M.A(T_deb_a-1) = M.A(T_deb_a-1) + TPS(SMC).A(1,1);
                M.A(T_deb_a:T_fin_a-1,1) = TPS(SMC).A(2:Taille_a_tps,1);
            end
         Etat_initial = Etat_final;
        end
        %======================
    end
    MN = M.MN;
    M.MN = [];
    M.MN = MN(MN(:,1)>0,:);
    MV = M.MV;
    M.MV = [];
    M.MV = MV(MV(:,1)>0,:);
    A = M.A;
    M.A = [];
    M.A = A(A(:,1)>0,:);
end
```

## 7 Simulation des CMTD

```
%% Simulation CMTD
% Cette fonction permet de générer des séries temporelles de longueur Long_Sim
% Les variables d'entrées sont :
% - la matrice de probabilité transition
% - L'état de départ de la simulation
% - la longueur de simulation

% Nous ne commenterons pas entièrement cette fonction car le fonctionnement est le même que pour les CSMTD, seul les ...
      divergence seront commentée
function [BP MP PhA PhD F MR MN MV A Final] = Simulation_CMTD_an(Matrice,Initial,Long_Sim)

        etat = Initial(1,1);
        etat_prec = Initial(2,1);
        Final = [1 ;1];
        Nbr_MP = 0;
        Nbr_BP = 0;
        Nbr_PhA = 0;
        Nbr_MR = 0;
        Nbr_F = 0;
        Nbr_PhD = 0;
        tps_MN = 1;
        tps_MV = 1;
        tps_A = 1;
        Tps_MN = zeros(500,3);
        Tps_MV = zeros(500,3);
```

```
Tps_A = zeros(250,3);
BP = zeros(1,Long_Sim);
MP = zeros(1,Long_Sim);
PhA = zeros(1,Long_Sim);
PhD = zeros(1,Long_Sim);
F = zeros(1,Long_Sim);
MR = zeros(1,Long_Sim);
format long
% Comme nous travaillons avec les probabilité cumulée, la matrice de probabilité de transitions doit être ...
      transformée
Matrice(1,2) = sum(Matrice(1,1:2));
Matrice(1,3) = sum(Matrice(1,2:3));
Matrice(2,2) = sum(Matrice(2,1:2));
Matrice(2,3) = sum(Matrice(2,2:3));
Matrice(3,2) = sum(Matrice(3,1:2));
Matrice(3,3) = sum(Matrice(3,2:3));
Limite = Long_Sim*24*60/5
dernier_chgt = 0;
N=1;

for s=1:Limite;
    f = rand(1,1);
    MZ = find(Matrice(etat)~= 0);
    T = isnan(Matrice(etat));
    TU = find(T==1);
    if size(TU)==size(MZ)
        if etat==1
            etat_suivant = 2;
        elseif etat==2
            etat_suivant = 1;
        else
            etat_suivant = 1;
        end
    else
        % Il faut prendre en compte maintenant 3 probabilité de transition et non juste deux probabilité de ...
             changement d'état.
        if f<Matrice(etat,1)
            etat_suivant = 1;
        elseif f > Matrice(etat,1) && f < Matrice(etat,2)
            etat_suivant = 2;
        elseif f > Matrice(etat,2)
            etat_suivant = 3;
        end
    end

    % On entre dans cette boucle si il y a effectivement un changement d'état
    if etat ~= etat_suivant;
        if etat == 1;
            % Le temps total passé dans l'état présent est enregistré
            if s == Limite
                Tps_MN(tps_MN,1) =  (s - dernier_chgt)*5;
                Tps_MN(tps_MN,2) = (s-1)*5;
                Final(1,1) = etat;
                Final(2,1) = etat_prec;
            else
                Tps_MN(tps_MN,1) =  (s - dernier_chgt)*5;
                Tps_MN(tps_MN,2) = (s-1)*5;
                tps_MN = tps_MN + 1;
                % Dans le cas où l'état présent est la marche normale, si l'état suivant est une marche à ...
                     vide on compte une baisse de puissance sinon c'est une phase d'arrêt
                if etat_suivant == 2;
                    Nbr_BP = Nbr_BP + 1;
                else
                    Nbr_PhA = Nbr_PhA + 1;
                end
            end

        elseif etat == 2;
            if s == Limite
                Tps_MV(tps_MV,1) =  (s - dernier_chgt)*5;
                Tps_MV(tps_MV,2) = (s-1)*5;
                Final(1,1) = etat;
                Final(2,1) = etat_prec;
            else
                Tps_MV(tps_MV,1) =  (s - dernier_chgt)*5;
                Tps_MV(tps_MV,2) = (s-1)*5;
                tps_MV = tps_MV + 1;
                if etat_suivant == 1;
                    % Si l'état suivant suivant est la marche normale et que l'état précédent est l'arrêt on ...
                         ajoute une phase de démarrage et on retranche une mise en rotation qui a du être ...
                         compter en trop. Si l'état précédent est autre on compte une montée en puissance
                    if etat_prec == 3
                        Nbr_PhD = Nbr_PhD + 1;
                        Nbr_MR = Nbr_MR - 1;
                    else
                        Nbr_MP = Nbr_MP + 1;
                    end
                else
                    % Dans le cas ou le l'état suivant est l'arrêt, si l'état précédent est la marche ...
                         normale on ajoute une phase d'arrêt et on retranche une baisse de puissance. Si ...
                         l'état précédent est autre on compte un freinage
```

```
109                             if etat_prec == 1
110                                 Nbr_PhA = Nbr_PhA + 1;
111                                 Nbr_BP = Nbr_BP - 1;
112                             else
113                                 Nbr_F = Nbr_F + 1;
114                             end
115                         end
116                     end
117                 else
118                     if s == Limite
119                         Tps_A(tps_A,1) =  (s - dernier_chgt)*5;
120                         Tps_A(tps_A,2) = (s-1)*5;
121                         Final(1,1) = etat;
122                         Final(2,1) = etat_prec;
123                     else
124                         Tps_A(tps_A,1) =  (s - dernier_chgt)*5;
125                         Tps_A(tps_A,2) = (s-1)*5;
126                         tps_A = tps_A + 1;
127                         % Dans le cas où l'état présent est l'arrêt, si l'état suivant est une marche normale on ...
                                compte une phase de démarrage sinon c'est une mise en rotation
128                         if etat_suivant == 1;
129                             Nbr_PhD = Nbr_PhD + 1;
130                         else
131                             Nbr_MR = Nbr_MR + 1;
132                         end
133                     end
134                 end
135                 dernier_chgt = s;
136                 etat_prec = etat;
137                 etat = etat_suivant;
138             end
139              % Le nombre de changements d'état pas jour est stocké
140              if s == N*288;
141                 % On peut avoir un nombre négatif de BP si un changement de marche normale vers arrêt en passant par ...
                        marche à vide se réalise sur deux jours (à minuit). Afin d'éviter cela, le nombre négatif est ...
                        ajouté au jour précédent.
142                 if Nbr_BP < 0 && N>1
143                     BP(1,N-1) = BP(1,N-1)+Nbr_BP;
144                     if BP(1,N-1)<0
145                         BP(1,N-1)=0;
146                     end
147                 elseif Nbr_BP < 0 && N==1
148                     BP(1,1) = 0;
149                 else
150                     BP(1,N) = Nbr_BP;
151                 end
152                 MP(1,N) = Nbr_MP;
153                 PhA(1,N) = Nbr_PhA;
154                 PhD(1,N) = Nbr_PhD;
155                 F(1,N) = Nbr_F;
156                 % Même chose que pour BP
157                 if Nbr_MR < 0 && N>1
158                     MR(1,N-1) = MR(1,N-1)+Nbr_MR;
159                     if MR(1,N-1)<0
160                         MR(1,N-1)=0;
161                     end
162                 elseif Nbr_MR < 0 && N==1
163                     MR(1,1) = 0;
164                 else
165                     MR(1,N) = Nbr_MR;
166                 end
167                 N=N+1;
168                 Nbr_MP = 0;
169                 Nbr_BP = 0;
170                 Nbr_PhA = 0;
171                 Nbr_MR = 0;
172                 Nbr_F = 0;
173                 Nbr_PhD = 0;
174              end
175         end
176          MN = Tps_MN;
177         if isempty(find(Tps_MN(:,2)>0, 1))
178         else
179             Tps_MN = MN(MN(:,2)>0,:);
180             MN = [];
181             MN = Tps_MN;
182         end
183         MV = Tps_MV;
184         if isempty(find(Tps_MV(:,2)>0, 1))
185         else
186             Tps_MV = MV(MV(:,2)>0,:);
187             MV = [];
188             MV = Tps_MV;
189         end
190         A = Tps_A;
191         if isempty(find(Tps_A(:,2)>0, 1))
192         else
193             Tps_A = A(A(:,2)>0,:);
194             A = [];
195             A = Tps_A;
196         end
```

```
197 end
```

# 8 Simulation du BBM

```
1  %% Simulation BBM
2  % Cette fonction permet de générer des séries temporelles synthétique à
3  % partir de ré-échantillonnage.
4  % Les variables d'entrées sont :
5  % - le vecteur contenant les valeurs de débit
6  % - la taille de bloc
7  % - la longueur de simulation
8
9  function M = Boot_mn(DEBIT,Long_Sim, Block_size)
10
11     B=ones(ceil(Long_Sim*365*24*60/5),1);
12     Taille_echant = length(DEBIT);
13     N = ceil((Long_Sim*365)/Block_size);
14     for i= 0:N
15    % Bornes de la fenêtre à piger
16         ind(1) = randi(Taille_echant-Block_size*24*60/5);
17         ind(2) = ind(1) + Block_size*24*60/5-1;
18         % Création de la série temporelle de taille, en ajoutant dans B, la fenêtre comprise entre les bornes ...
                définies précédemment
19         B(i*Block_size*24*60/5+1:...
20          i*Block_size*24*60/5+Block_size*24*60/5,1) = ...
21          DEBIT(ind(1):ind(2),1);
22     end
23     % Dans Bootstrap la série temporelle de longueur voulue, car la boucle précédente en produit de légèrement plus ...
           grande
24     Bootstrap = B(1:ceil(Long_Sim*365*24*60/5));
25     % Comptage du nombre de changements d'état par jour et des temps de maintien
26     Tr = Comptage_tran(Bootstrap,1);
27     [Tps_MN Tps_MV Tps_A] = Comptage_tps(Bootstrap);
28     % Stockage nombre de changements d'état.
29     M.BP = Tr(1,1:ceil(Long_Sim*365));
30     M.MP = Tr(2,1:ceil(Long_Sim*365));
31     M.PhA = Tr(3,1:ceil(Long_Sim*365));
32     M.PhD = Tr(4,1:ceil(Long_Sim*365));
33     M.F = Tr(5,1:ceil(Long_Sim*365));
34     M.MR = Tr(6,1:ceil(Long_Sim*365));
35     % Stockage des temps de maintien
36     M.MN = Tps_MN(:,1);
37     M.MV = Tps_MV(:,1);
38     M.A = Tps_A(:,1);
39 end
```

**BIBLIOGRAPHIE**


« ASTM E1823-11 Standard Terminology Relating to Fatigue and Fracture Testing ».

R.E. Abdel-Aal et a.Z. Al-Garni. 1997. « Forecasting Monthly Electric Energy Consumption in Eastern Saudi Arabia Using Univariate Time-Series Analysis ». *Energy*, vol. 22, n 11, p. 1059–1069.

E Alvarez. 2005. « Smoothed Nonparametric Estimation in Window Censored Semi-Markov Processes ». *Journal of Statistical Planning and Inference*, vol. 131, n 2, p. 209–229.

T.W. Anderson et L.A. Goodman. 1957. « Statistical Inference about Markov Chains ». *The Annals of Mathematical Statistics*, vol. 28, n 1, p. 89–110.

Sylvain Arlot et Alain Celisse. 2010. « A Survey of Cross-Validation Procedures for Model Selection ». *Statistics Surveys*, vol. 4, p. 40–79.

J Scott Armstrong. 2007. « Significance Tests Harm Progress in Forecasting ». *International Journal of Forecasting*, vol. 23, n 2, p. 321–327.

Vlad Stefan Barbu et Nikolaos Limnios, 2008. *Semi-Markov and Hidden Semi-Markov Models towards Applications*. Springer.

M.J. Bayarri, J.O. Berger, R. Paulo, J. Sacks, J.A. Cafeo, J. Cavendish, C.H. Lin, et J. Tu. 2007. « A Framework for Validation of Computer Models ». *Technometrics*, vol. 49, n 2, p. 138–154.

Yoshua Bengio et Nicolas Chapados. 2003. « Extensions to Metric Based Model Selection ». *The Journal of Machine Learning Research*, vol. 3, p. 1209–1227.

André Berchtold et Adrian Raftery. 2002. « The Mixture Transition Distribution Model for High-Order Markov Chains and Non-Gaussian Time Series ». *Statistical Science*, vol. 17, n 3, p. 328–356.

José M. Bernardo et Raul Rueda. 2002. « Bayesian Hypothesis Testing : A Reference Approach ». *International Statistical Review*, p. 1–22.

Christopoher M. Bishop, 1995. *Neural Networks for Pattern Recognition*. Clarendon press Oxford.

George E.P. Box, Gwilym M. Jenkins, et Gregory C. Reinsel, 1970. *Time Series Analysis*, volume 26. Holden-day San Francisco.

Peter Bühlmann. 2002. « Bootstraps for Time Series ». *Statistical Science*, vol. 17, n 1, p. 52–72.

Peter Bühlmann et H. Kunsch. 1999. « Block Length Selection in the Bootstrap for Time Series ». *Computational Statistics and Data Analysis*, vol. 31, p. 295-310.

E. Carlstein. 1986. « The Use of Subseries Methods for Estimating the Variance of a General Statistic from a Stationary Time Series ». *The Annals of Statistics*, vol. 14, p. 1171-1179.

A Carpinone, R Langella, A Testa, et M. Giorgio. 2010. « Very Short-term Probabilistic Wind Power Forecasting Based on Markov Chain Models ». In *IEEE 11th International Conference on Probabilistic Methods Applied to Power Systems (PMAPS)*. p. 107–112. IEEE.

J.-M. Chapallaz. 1995. *Turbines hydrauliques*. Technical report. Office fédéral de l'énergie OFEN. Programme PACER.

Chen-hua Chung et Jose D. Salas. 2000. « Drought Occurrence Probabilities and Risks of Dependent Hydrologic processes ». *Journal of Hydrologic Engineering*, , p. 259–268.

Thomas G Dietterich. 1997. « Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms ». *Science*, p. 1–24.

Martin Dubrovský. 1997. « Creating Daily Weather Series with Use of the Weather Generator ». *Environmetrics*, vol. 8, n 5, p. 409–424.

B. Efron et R.J. Tibshirani, 1993. *An Introduction to the Bootstrap*, volume 57 of *Monographs on statistics and applied probability*. Chapman & Hall, 436 p.

Fiorenzo Ferrari et Abraham Wyner. 2003. « Estimation of General Stationary Processes by Variable Length Markov Chains ». *Scandinavian Journal of Statistics*, vol. 30, n 3, p. 459–480.

Konstantinos Fokianos et Benjamin Kedem. Août 2003. « Regression Theory for Categorical Time Series ». *Statistical Science*, vol. 18, n 3, p. 357–376.

H Fowler, C Kilsby, P Oconnell, et a Burton. 2005. « A Weather-type Conditioned Multi-site Stochastic Rainfall Model for the Generation of Scenarios of Climatic Variability and Change ». *Journal of Hydrology*, vol. 308, n 1-4, p. 50–66.

H.J. Fowler, C.G. Kilsby, et P.E. OConnell. 2000. « A Stochastic Rainfall Model for the Assessment of Regional Water Resource Systems Under Changed Climatic Condition », vol. 4, n 2. p. 263–281.

Martin Gagnon, S.A. Tahan, P. Bocher, et D. Thibault. 2010. « Impact of Startup Scheme on Francis Runner Life Expectancy ». In *IAHR Symposium on Hydraulic Machinery and Systems*. p. 1–7.

Silvia Gonçalves et Halbert White. 2002. « The Bootstrap of the Mean for Dependent Heterogeneous Arrays ». *Econometric Theory*, vol. 18, p. 1367–1384.

Priscilla E. Greenwood et Wolfgang Wefelmeyer. 1996. « Empirical Estimators for Semi-Markov Processes ». *Math. Meth. Statist*, vol. 5, p. 299-315.

Priscilla E. Greenwood, Ursula U. Müller, et Wolfgang Wefelmeyer. 2004. « Efficient Estimation for Semiparametric Semi-Markov Processes ». *Communications in Statistics - Theory and Methods*, vol. 33, n 3, p. 419–435.

Peter Hall, Joel L. Horowitz, et Bing-Yi Jing. 1995. « On Blocking Rules for the Bootstrap with Dependent Data ».

R Hyndman et a Koehler. 2006. « Another Look at Measures of Forecast Accuracy ». *International Journal of Forecasting*, vol. 22, n 4, p. 679–688.

P. a. Jacobs et P. a. W. Lewis. 1983. « Stationary Discrete Autoregressive-Moving Average Time Series Generated By Mixtures ». *Journal of Time Series Analysis*, vol. 4, n 1, p. 19–36.

Robert C. Jung et a.R. Tremayne. 2006. « Coherent Forecasting in Integer Time Series Models ». *International Journal of Forecasting*, vol. 22, n 2, p. 223–238.

J Kam. 1992. « Wave Action Standard History (WASH) for Fatigue Testing Offshore structures ». *Applied Ocean Research*, vol. 14, n 1, p. 1–10.

H Kaufmann. 1987. « Regression Models for Nonstationary Categorical Time Series : Asymptotic Estimation Theory ». *The Annals of Statistics*, vol. 15, p. 79–98.

J Khan, JS Wei, LH Saal, et M Ladanyi. 2001. « Classification and Diagnostic Prediction of Cancers Using Gene Expression Profiling and Artificial Neural Networks ». *Nature medicine*, vol. 7, n 6, p. 673–679.

Jeffrey P. Kharoufeh, Christopher J. Solo, et M. Yasin Ulukus. 2010. « Semi-Markov Models for Degradation-based Reliability ». *IIE Transactions*, vol. 42, n 8, p. 599–612.

Tae-woong Kim et Juan B. Valdes. 2005. « Synthetic Generation of Hydrologic Time Series Based on Nonparametric Random Generation ». *Journal of Hydrologic Engineering*, , p. 395–404.

H.R. Kunsch. 1989. « The Jackknife and the Bootstrap for General Stationary Observations ». *The Annals of Statistics*, vol. 17, n 3, p. 1217–1241.

S.N. Lahiri. 1999. « Theoretical Comparisons of Block Bootstrap Methods ». *The Annals of Statistics*, vol. 27, n 1, p. 386–404.

Upmanu Lall et Ashish Sharma. 1996. « A Nearest Neighbor Bootstrap For Resampling Hydrologic Time Series ». *Water Resources Research*, vol. 32, n 3, p. 679.

Amaury Lendasse, Vincent Wertz, et Michel Verleysen. 2003. « Model Selection with Cross-Validations and Bootstraps : Application to Time Series Prediction with RBFN Models ». In *Proceedings of the 2003 joint international conference on Artificial neural networks and neural information processing*. p. 573–580. Springer-Verlag.

I.L. MacDonald et W. Zucchini, 1997. *Hidden Markov and Other Models for Discrete-valued Time Series*. Chapman and Hall.

E. McKenzie. 1985. « Some Simple Models for Discrete Variate Time Series ». *JAWRA Journal of the American Water*, vol. 21, n 4, p. 645–650.

E. Mckenzie. 2003. Discrete Variate Time Series. Shanblag, D. et C.R. Rao, editors, *Stochastic Processes : Modeling and Simulation*, number August, chapter 16, p. 1–34. Elsevier.

Mario Montopoli, Frank Silvio Marzano, et Gianfranco Vulpiani. 2008. « Analysis and Synthesis of Raindrop Size Distribution Time Series From Disdrometer Data ». *IEEE Transactions on Geoscience and Remote Sensing*, vol. 46, n 2, p. 466–478.

C. Nadeau et Y. Bengio. 2003. « Inference for the Generalization Error ». *Machine Learning*, p. 239–281.

R.G. Newcombe. 1998. « Two-sided Confidence Intervals for the Single Proportion : Comparison of Seven Methods ». *Statistics in Medicine*, vol. 17, n 1998, p. 857–872.

Y. Ogata. 1989. « Statistical Model for Standard Seismicity Detection of Anomalies by residual analysis ». *Earthquake*, vol. 169, p. 159–174.

D.C. Park, M.A. El-Sharkawi, R.J. Marks, L.E. Atlas, et M.J. Damborg. 1991. « Electric Load Forecasting Using an Artificial Neural Network ». *Power Systems, IEEE Transactions on*, vol. 6, n 2, p. 442–449.

Dimitris N. Politis et J. Romano, 1992. *Exploring the Limits of the bootstrap*, chapter A Circular Block Resampling Procedure for Stationary Data, p. 263-270. Wiley.

Dimitris N. Politis et J.P. Romano. 1994. « The Stationary Bootstrap ». *American Statistical Association*, vol. 89, n 428, p. 1303-1313.

Dimitris N. Politis, Joseph P. Romano, et Michael Wolf, 1999. *Subsampling*, chapter Subsampling for Nonstationary Time Series, p. 100-119. Springer Series in Statistics. Springer.

L.R. Rabiner. 1989. « A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition ».

J. Racine. 2000. « Consistent Cross-validatory Model Selection for Dependent Data : HV-block Cross-validation ». *Journal of econometrics*, vol. 99, n 1, p. 39–61.

A.E. Raftery. 1985. « A model for high-order Markov chains ». *Journal of the Royal Statistical Society. Series B (Methodological)*, p. 528–539.

Michel Sabourin, Denis Thibault, et David-alexandre Bouffard. 2010. « Hydraulic Runner Design Method for Lifetime ». *International Journal*, vol. 3, n 4, p. 301–308.

Steven L. Salzberg. 1997. « On Comparing Classifiers : Pitfalls to Avoid and a Recommended Approach ». *Data mining and knowledge discovery*, vol. 1, p. 317–328.

A Shamshad, M A Bawadi, T A Majid, W Wanhussin, et S Sanusi. 2005. « First and Second Order Markov Chain Models for Synthetic Generation of Wind Speed Time Series ». *Energy*, vol. 30, n 5, p. 693–708.

Galit Shmueli. 2010. « To Explain or to Predict ? ». *Statistical Science*, vol. 25, n 3, p. 289–310.

Adalberto N. Siqueira, Chigueru Tiba, et Naum Fraidenraich. 2010. « Generation of Daily Solar Irradiation by Means of Artificial Neural Networks ». *Renewable Energy*, vol. 35, n 11, p. 2406–2414.

M Skorupa. 1998. « Load Interaction Effects During Fatigue Crack Growth Under Variable Amplitude Loading-a Literature Review. Part I : Empirical Trends ». *Fatigue and Fracture of Engineering Materials and Structures*, vol. 21, n 8, p. 987–1006.

R. Srikanthan et T. a. McMahon. 2001. « Stochastic Generation of Annual, Monthly and Daily Climate Data : A Review ». *Hydrology and Earth System Sciences*, vol. 5, n 4, p. 653–670.

Rafal Synowiecki. 2007. « Consistency and Application of Moving Block Bootstrap for Non-stationary iTime Series with Periodic and Almost Periodic Structure ». *Bernoulli*, vol. 13, n 4, p. 1151–1178.

Denis Thibault, Philippe Bocher, Marc Thomas, Jacques Lanteigne, Pierre Hovington, et Patrice Robichaud. 2011. « Reformed Austenite Transformation During Fatigue Crack Propagation of 13%Cr-4%Ni Stainless Steel ». *Materials Science and Engineering*, vol. 528, n 21, p. 6519 - 6526.

E. Toth, a. Brath, et a. Montanari. 2000. « Comparison of Short-term Rainfall Prediction Models for Real-time Flood Forecasting ». *Journal of Hydrology*, vol. 239, n 1-4, p. 132–147.

Mevlut Ture et Imran Kurt. 2006. « Comparison of Four Different Time Series Methods to Forecast Hepatitis A Virus Infection ». *Expert Systems with Applications*, vol. 31, n June 2004, p. 41–46.

Cristiano Varin et Paolo Vidoni. 2006. « Pairwise Likelihood Inference for Ordinal Categorical Time Series ». *Computational Statistics & Data Analysis*, vol. 51, n 4, p. 2365–2373.

Ingmar Visser. 2011. « Seven Things to Remember About Hidden Markov Models : A Tutorial on Markovian Models for Time Series ». *Journal of Mathematical Psychology*, vol. 55, n 6, p. 403–415.

Richard M. Vogel et Amy L. Shallcross. 1996. « The Moving Blocks Bootstrap Versus Parametric Time Series Models ». *Water Resources Research*, vol. 32, n 6, p. 1875-1882.

Christian H. Weiß. 2008. « Visual Analysis of Categorical Time Series ». *Statistical Methodology*, vol. 5, n 1, p. 56–71.

Christian H. Weiß. 2011. « Generalized Choice Models for Categorical Time Series ». *Journal of Statistical Planning and Inference*, vol. 141, n 8, p. 2849–2862.

Y. Xue, A. Leetmaa, et M. Ji. 2000. « ENSO Prediction with Markov Models : The Impact of Sea Level ». *Journal of Climate*, vol. 13, p. 849–871.

S. Yu. 2010. « Hidden Semi-Markov Models ». *Artificial Intelligence*, vol. 174, n 2, p. 215–243.

S.L. Zeger et B. Qaqish. 1988. « Markov Regression Models for Time Series : a Quasi-likelihood Approach ». *Biometrics*, vol. 44, n 4, p. 1019–1031.