.

Université Paris-saclay

ENSTA ParisTech

Rapport du stage de fin d'études

Optimisation des stratégies de maintenance sous incertitude
Application à des cas-tests avec la plate-forme VME

Auteur :
Arfaoui Amani

Encadrants :
Jérôme Lonchampt
Jean-Philippe Chancelier
Enseignant Référent :
Pierre Carpentier

11 septembre 2016

Remerciement

C'est avec un immense plaisir que je rédige ces remerciements. Mes premiers remerciements vont à mes professeurs à l'ENSTA paristech et au master 2 optimisation qui m'ont aidé à comprendre différentes notions et à accroître ma culture scientifique toute au long de cette année. J'adresse également mes remerciements à Monsieur Pierre Carpentier mon encadrant académique depuis le début de l'année vous m'as toujours motivé et donné de précieux conseils.

J'adresse également mes remerciements à mon tuteur à EDF Chatou, Jérôme Lonchampt, tu étais toujours là pour m'aider, me motiver et me soutenir. et je te remercie pour toute l'attention que tu as porté à lire et corriger mes présentations et mon rapport.

Je tiens également à remercier Monsieur Jean Philippe Chancelier pour votre encadrement dans l'approche du PDMP, vos idées m'ont beaucoup aidée à avancer, Merci pour le temps que vous avez consacré à discuter et corriger mon travail.

Ma gratitude va également à tous l'équipe MRI. Un grand merci à Mathieu Couplet qui a passé deux jour pour m'aider à débloquer le logiciel NOMAD. Je remercie également tous les stagiaires de MRI, j'ai passé des bon moments avec vous.

Je tiens évidemment à remercier ma famille et mes amis pour me soutenir et surtout d'avoir relu mon rapport et mes présentations.

Résumé

Pour une compagnie dans le secteur de production d'électricité comme EDF (Électricité de France) la gestion de ses actifs de production garantie la disponibilité et la fiabilité de ses moyens de production. Donc la maintenance des actifs industriels est une tâche primordiale pour créer de la valeur ajoutée pour l'entreprise. Pour cela EDF a mis en place des stratégies de maintenance complexes qui doivent prendre en compte le vieillissement des équipements pour anticiper les défaillances. Pour aider à la prise de décision sur la stratégie de maintenance à adopter, différents indicateurs, tels que la fiabilité du matériel, la capacité de production ou le coût d'exploitation et de maintenance, sont évalués.

Dans le stage, on s'intéresse à l'effet d'une action de maintenance exceptionnelle (effectuée sur un composant) sur ce coût. Pour déterminer si une stratégie de maintenances est intéressante il est nécessaire de quantifier des indicateurs économiques traduisant le gain espéré par rapport à une stratégie sans investissement prévue. A EDF, les indicateurs économiques utilises sont issus de la Valeur Actuelle Nette (VAN). La VAN est un indicateur important pour EDF pour un choix pertinent de stratégies d'investissements et sa maximisation est l'objectif que l'on cherche à atteindre.

Le département de Management des Risques Industriels (MRI) de la R&D d'EDF a déjà développé des outils pour valoriser et optimiser les stratégies d'investissements. Ces outils utilisent la méthode de Monte Carlo pour estimer la loi de la VAN. Certains outils utilisent aussi les algorithmes génétiques pour déterminer la stratégie d'investissements ayant la VAN moyenne la plus élevée parmi celles qui ont été évaluées, lorsque la VAN moyenne peut être calculée sans avoir recours à la simulation de Monte Carlo.

L'objectif de ce stage est de donner une méthode alternative à l'algorithme génétique permettant d'optimiser les différents indicateurs issus de la VAN d'une stratégie d'investissements. Pour atteindre cet objectif deux voies ont été explorée : La première porte sur les algorithmes d'optimisation de boites noires (Black-box optimization) et la deuxième est de modéliser l'évolution stochastique des composants soumis à maintenance et celle du stock ç.à.d modéliser l'état des composants, l'état du stock, les durées de vie des composants, coûts, ... pour pouvoir écrire au moins un problème de maximisation de la moyenne de la VAN dans l'espace des dates de remplacement des composants. A cet effet dans la première voie nous avons choisi les méthodes de recherches directs et pour la deuxième ce sont les Processus Markoviens Deterministes par Morceaux (Piecewise Deterministic Markov Processes - PDMPs) qui ont été étudiés.

Abstract

Asset management processes, guarantee the availability and the reliability of EDF's nuclear plants. That's why the maintenance of industrial assets at EDF is a primary task for realising value. Evaluationg the value of an asset managment strategy may be done by calculating the Net Present Value (NPV) which is the difference in the asset life cycle cost induced by this strategy compared to a reference, or current, one. As the life cycle cost depends on the possible failures of the asset and their dates, which are uncertain according to a propabilistic reliability law.

The Department of Management and Industrial Risks (MRI) at EDF R&D. EDF has already developed tools to enhance and optimize investment strategies. These tools use the Monte Carlo methode for estimating the law of the NPV. Some tools also use genetic algorithms to determine the investment strategy having the highest average NPV among those that were evaluated, when the average NPV can be calculated without using the Monte Carlo simulation.

The objective of this internship is to give an alternative to the genetic algorithm to optimize the various indicators from the NPV of an investment strategy. To achieve this, two ways have been explored :
The first one, focuses on black-box optimization algorithms, the second one, aims to modelise the life cycle of a composent to be able to write at least a maximization problem of NPV's average in the space of replacement dates of components. For this purpose, in the first path we have chosen the methods of direct research and for the second, it is deterministic Markov Processes by Pieces.

Table des figures

Liste des tableaux

Table des matières

ACRONYMES

EDF	Électricité De France
R&D	Recherche et développement
MRI	Management des Risques Industriels
Cs	Algorithme de recherche par coordonnées (Coordinate Search)
Gps	Algorithme de recherche par motifs (Generalized Pattern Search)
MADS	Algorithme de recherche par treillis adaptatifs
PDMP	Piecewise Deterministic Markov Process
VAN	Valeur actuelle nette

Chapitre 1

Introduction

Dans le cadre de ma troisième année de formation d'ingénieur à l'école nationale supérieure de techniques avancées et du master 2 optimisation à l'université Paris-Saclay, le choix de mon stage s'est basé sur la possibilité d'approfondir mes connaissances dans le domaine de l'optimisation et de ses applications. Le fait d'être confronté à une problématique industrielle réelle , de faire de la recherche et de pouvoir élargir mes compétences dans les mathématiques appliqués ont été les raisons pour lesquelles j'ai choisi de faire mon stage dans le département management des risques industriels(MRI) à EDF R&D.

Les mots "optimiser, optimisation, etc." sont supposés refléter cette idée du "mieux possible", c'est en fait le besoin des industries : avoir une meilleure disponibilité des moyens de production et c'est dépenser moins, "produire plus pour moins cher". C'est dans ce contexte qu' EDF cherche à optimiser le coût des stratégies d'investissements. EDF R&D a ainsi développé depuis plusieurs années des outils et méthodes de R&D visant à aider les unités opérationnelles à gérer ses actifs. A ce titre, l'outil VME (Valorisation de maintenance exceptionnelle) intercompare les stratégies de maintenance exceptionnelle.
Les stratégies de maintenance sont hiérarchisées via l'indicateur VAN(t), Valeur Actuelle Nette, calculé chaque année t : la VAN(t) quantifie la différence entre le coût de cycle de vie de l'installation industrielle induit par une stratégie et celui induit par une autre stratégie , à l'année t. Le coût d'un cycle de vie est aléatoire car il dépend de la survenue d'aléas sur les matériels l'installation industrielle : dégradation, défaillance. Il dépend aussi d'autre facteurs tels que la qualité de la maintenance (impact sur la fiabilité du matériel), de la politique maintenance (corrective, préventive,...), de la politique de pièces de rechange .. Ainsi VAN(t) est une variable aléatoire dont il s'agit de connaître la loi de probabilité à chaque instant t.

Un des objectifs des études de gestion d'actifs est d'optimiser l'espérance de la variable aléatoire VAN(t) à un instant t particulier. En revanche, modéliser les stratégie de maintenance pour écrire l'expression analytique de la VAN est une tache difficile à cause de la complexités des stratégies de maintenance. Ce qui explique l'utilisation au département MRI d' outils d'optimisation basés sur les simulations de lois comme Monte Carlo et des algorithmes métaheuristiques comme les algorithmes génétiques. Dans stage j'ai abordé deux techniques d'optimisation :

D'abord, l'optimisation de boites noires : Vu que le logiciel VME est un programme informatique complexe et qu'il est en inaccessible pour des raisons de confidentialité, on peut considérer que cet outil est une boite noire et exploiter seulement sa réponse (La VAN) à des entrés données (par exemple des dates de remplacement préventive des stratégies..) et chercher l'entrée optimal qui optimise la réponse de cette boite noire. Plusieurs algorithmes sans dérivées conçus pour résoudre ce type de problèmes ont été développés et ne requièrent aucune hypothèse sur la fonction. Mais, dans notre cas d'application la réponse qui est la VAN est bruité et la solution obtenue par les algorithmes d'optimisation de boites noires n'est pas acceptable. Ces raisons nous amène à envisager des solutions d'amélioration d'un al-

gorithme d'optimisation de boite noire, appelé MADS, pour l'appliquer à des cas tests de calculs de VAN bruitée par la simulation.

La seconde technique consiste à ouvrir la boite noire VME et tenter de modéliser les stratégie de maintenance : Le vieillissement des composants et la complexité des stratégies de maintenance étudiées nous obligent a avoir recours à de nouveaux modèles probabilistes afin de répondre à la problématique. Les modèles probabilistes que l'on peut trouver dans la littérature sont les processus markoviens. Dans ce stage j'ai utilisé les PDMP (Piecewise Deterministic Markov Processes - PDMPs) pour modéliser le cycle de vie d'un composant nucléaire et écrire l'expression analytique de la VAN en appliquant la méthode de quantification des PDMP pour résoudre un problème d'arrêt optimal.

Dans ce rapport, je commence tout d'abords par une présentation d'EDF et du contexte industrielle. Ensuite, je présente les différents algorithmes d'optimisation des boites noire et l'amélioration de l'algorithme MADS pour optimiser les fonctions bruitées et je finis cette partie par l'explication du code c++ de l'algorithme que j'ai implémenté. Je consacre ensuite une partie à la présentation des processus markoviens en particulier les PDMPs . Avant de détailler, la modélisation de mon problème par les PDMP et enfin expliquer les outils mathématiques utilisés dans la méthode de quantification et l'application de cette méthode pour résoudre mon problème.

Chapitre 2

Présentation industrielle

Le groupe Électricité de France (EDF) est aujourd'hui le premier électricien mondial associant tous les métiers de la production, du négoce et des réseaux d'électricité. Établissement public devenu une société anonyme en 2004, EDF a employé 159 112 collaborateurs dans le monde en 2015. Avec 37,6 millions de clients dans plus de 30 pays, il a réalisé un chiffre d'affaires de 75 milliards d'euros (2015).

2.1 EDF Recherche & Développement

A l'heure où la numérisation bouleverse les façons de produire et de consommer, les recherches sur la production, les réseaux de distribution et la consommation de l'électricité sont d'une importance décisive. Pour réussir la transition énergétique, les collaborateurs de la R&D d'EDF, de 29 nationalités différentes, travaillent sur de nombreux projets qui visent à la fois une production bas-carbone, des réseaux de distribution plus « intelligents » et une consommation plus responsable. Les compétences des chercheurs d'EDF R&D sont regroupées en une trentaine de macro-compétences au service de la performance des unités opérationnelles sous forme d'études, d'essais et d'expertises. Dans le cadre de la préparation de l'avenir, EDF R&D contribue également à faire émerger des relais de croissance pour le groupe à moyen et long termes. Avec plus de 2000 collaborateurs répartis dans 15 départements, EDF R&D est organisée autour de sept sites, dont trois situés en France, en région parisienne. Le centre de recherche de Chatou (Yvelines) qui accueille ce stage, regroupe près de 500 chercheurs et techniciens. Les activités du site sont globalement centrées sur l'énergétique appliquée aux moyens de production d'électricité, qu'ils soient conventionnels (thermique, hydraulique, nucléaire) ou qu'ils relèvent des énergies renouvelables. En étroite collaboration avec la Direction Production Ingénierie (DPI), le service Recherche & Développement travaille à optimiser le fonctionnement du parc EDF et à prolonger sa durée de vie.

2.1.1 Département MRI

Le département Management des Risques Industriels (MRI) a pour objet l'étude des systèmes socio-techniques à risques, exploité au sein du groupe EDF, comme les centrales nucléaires et thermiques, les ouvrages hydrauliques ou encore le réseau de transport. Hébergé au sein des sites de Chatou et Saclay, MRI appuie d'autres départements de la R&D grâce à ses compétences en sûreté de fonctionnement, statistiques et propagation d'incertitudes. Le département s'articule autour de six compétences-clefs lui permettant de couvrir l'ensemble des problématiques liées au risque industriel :

- Études Probabilistes de Sûreté
- Analyses de Risques Systèmes,
- Approches Probabilistes et Statistiques des Phénomènes Physiques,
- Facteurs Humains et Organisationnels,
- Aide à la Décision & Performance des Actifs,
- Modélisation & Simulation Numérique des Procédés.

2.1.2 Groupe MODIF

Au sein du département MRI, le groupe Maintenance, Optimisation des Décisions d'Investissement et Fiabilité (MODIF) dans lequel s'inscrit ce stage, travaille au développement d'outils et méthodes liés à 3 sous-compétences :

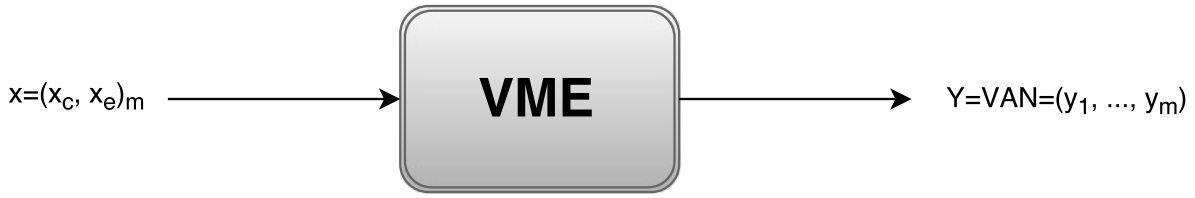
- Analyse statistique des phénomènes physiques
- Aide à décision et performance des actifs industriels
- Solutions robotisées pour la maintenance

85% des activités du groupe MODIF sont orientées vers la filière nucléaire. Actuellement, le groupe présente un effectif de 16 Ingénieurs-Chercheurs, dont 1 Senior, 4 Experts, 2 Chefs de projet, 1 Doctorant et 1 Chef de Groupe.

FIGURE 2.1 – Quelques activités au sein du groupe MODIF du département MRI/EDF R&D

2.2 Le simulateur VME

Afin de favoriser la prise de décision en ce qui concerne l'entretien de ses différentes unités, EDF R&D a développé un outil scientifique qui s'appelle VME. L'entrée de ce simulateur est un couple $x = (x_c, x_e)$ où x_e représente les variables dites de contrôle sur lesquelles l'utilisateur peut agir, comme par exemple les dates de maintenance des différents composants du système et x_c les variables dites environnementales sur lesquelles l'utilisateur ne peut pas agir, comme le taux d'actualisation ou le prix des composants par exemples ou encore les lois de fiabilité.
La sortie est un batch d'un nombre de réalisations de la Valeurs Actualisées Nettes, ou VAN, qui représente la valeur actualisée (c'est à dire en euro constant) de ce qu' a rapporté (ou fait perdre) une stratégie d'investissement. Cette VAN, qui peut être négative(mauvais investissement) ou positive (bon investissement), est aléatoire et calculée par une simulation de type Monte Carlo.



Chapitre 3

Optimisation de boites noires

Introduction

L'optimisation de type boite noire est dédiée aux problèmes d'optimisation où les fonctions définissant le problème sont complexes, difficiles à évaluer et leur expression analytique est inconnue. Ce concept (boîte-noire) permet de faire abstraction de la fonction dont l'exploitation se fait uniquement par l'analyse de la réponse à une entrée donnée. Dans la littérature,il existe de nombreuses plusieurs utilisations de ce concept,on trouve une application des boîtes-noires dans le domaine de l'hydroélectricité est décrite par [1] pour le problème de localisation de stations de mesure du couvert nival.

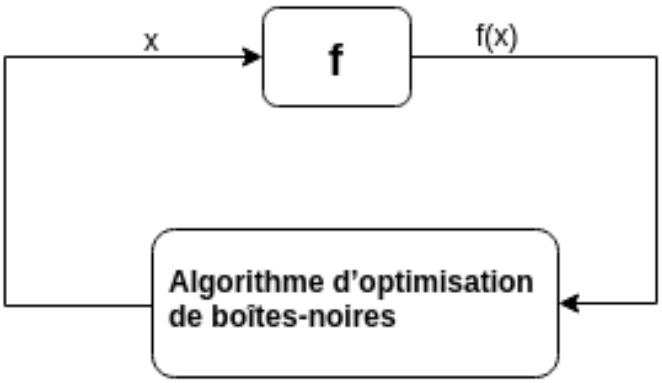


FIGURE 3.1 – Schéma d'optimisation de boîte-noire.

L'optimisation de boite noire consiste à évaluer la boîte-noire f en des points candidats générés par un algorithme d'optimisation de boîtes-noires puis, la valeur de f est renvoyée à l'algorithme afin de l'analyser et de générer des nouveaux points.

3.1 Revue de la littérature

Dans la littérature plusieurs algorithme d'optimisation sans dérivée ont été développées.
On peut citer par exemples les méthodes de recherche direct comme celles de Nelder et Mead(Nelder, J.A. and Mead, R. (1965)) appelée parfois méthode du simplexe puisqu' elle effectue des opérations sur un simplexe au cours des itérations de façon que ce simplexe se déforme, se déplace et se réduise progressivement jusqu'à ce que ses sommets se rapprochent d'un point où la fonction est localement minimale. On peut également citer les méthodes de région de confiance qui utilisent un modèle pour approcher la fonction objectif, en supposant l'existence de la dérivée de la fonction mais sans l'utiliser formellement. On peut citer aussi l'algorithme dividing rectangle introduit par Jones (1993) il s'agit

d'un algorithme de partitionnement qui échantillonne des points dans un domaine donné, puis affine le domaine de recherche à chaque itération sur la base des valeurs de la fonction aux points échantillonnés.

Les algorithmes de recherche direct que j'ai appliqué dans mon stage pour optimiser la réponse du simulateur VME (la VAN) sont les algorithmes de recherche directe directionnelle.

3.1.1 Méthodes de recherche directe directionnelle

Ces méthodes utilisent uniquement les valeurs de la fonction f pour conduire l'optimisation. En effet, le principe de ces méthodes repose sur la génération de directions qui permettent d'explorer l'espace de recherche de la solution optimal [3].

Le premier algorithme de recherche directe est l'algorithme de recherche par coordonnées (CS : Coordinate Search) de Fermi et Metropolis (1952). Une évolution de (CS) mène à l'algorithme de recherche par motifs (GPS : Generalized Pattern Search) présenté par Torczon (1997) et généralisé par Audet et Dennis (2006) pour donner naissance à l'algorithme de recherche sur treillis adaptifs (MADS : Mesh Adaptive Direct Search).

3.1.1.1 Algorithme de recherche par coordonnées(Coordinate Search) :

Cet algorithme est dédié aux problèmes sans contrainte. Il peut être décrit comme suit :

- x_0 le point initial et Δ_0 le pas initial sont fournis par l'utilisateur.
- À l'itération k $\in \{1, 2, 3, ...\}$ x_k est la meilleur solution courante appelée "the current incumbent solution".

A chaque itération, on évalue f en des points qui sont générés en se déplaçant à un pas de longueur Δ_k de l'itéré courant x_k suivant les directions $\{e_i \in \mathbf{R}^n 1 \leq i \leq n\}$ et leurs opposées.
Cette exploration des points d'essai dans l'espace discrétisé, appelé treillis, a pour but de rechercher un point ayant une plus petite valeur de l'objectif que celle de l'itéré courant. l'ensemble des itérés courants de chaque itération est :

$$P_k = \{x_k \pm \Delta_k^m e_i, i = 1..n\} \tag{3.1}$$

Après l'évaluation de la fonction f en chaque point de cet espace il y aura deux possibilités :

- Si $\exists x \in P_k$ telque $f(x) \leq f(x_k)$ alors l'itération est un succès et le prochain itéré devient x.
- Sinon l'itération est un échec et on réduit le pas Δ_k^m à la moitié dans la prochaine itération.

Cette méthode est très facile à implémenter néanmoins elle présente plusieurs lacunes. La recherche se fait suivant un nombre restreint de directions, qui sont les mêmes à chaque itération, des régions de l'espace potentiellement intéressantes risquent de ne jamais être visitées. D'autre part, la recherche se fait uniquement de manière locale.De plus, même si un meilleur point a été trouvé parmi l'ensemble P_k, la fonction est évaluée à tous les autres points de P_k au lieu d'arrêter les évaluations au premier succès.

3.1.1.2 Algorithme de recherche par motifs (Generalized Pattern Search) :

Cet algorithme est une généralisation du GPS et ses principales améliorations consistent en :

- La recherche locale est effectuée sur un ensemble fini de directions diversifiées plutôt que les 2n directions de la base usuelle
- À chaque itération, l'algorithme permet l'exploration en un nombre fini de points autres que les points de l'espace P_k des points courants.

Soit donc à chaque itération k, l'ensemble des directions $D_k = d_k^1, d_k^2, ..., d_k^{p_k}$ et D l'ensemble des directions possibles, elles doivent satisfaire les conditions suivantes :

- D_k représente un ensemble générateur positif [1] et $D_k \subset$ D

1. Un ensemble fini de vecteurs $D = \{d_1, d_2, ..., d_k\}$ est dit un ensemble générateur positif pour $\mathbf{R}^n$, si chaque vecteur v $\in \mathbf{R}^n$ peut s'exprimer par une combinaison linéaire positive des vecteurs de D

- Chaque direction $d_j \in D$, j = 1, ..., n_D, doit être issue d'un produit d'une matrice nonsingulière G $\in R^{n\times n}$ par un vecteur d'entiers $z_j \in \mathbf{Z}^n$
- l'ensemble des itérés courants de chaque itération (the poll set) est :

$$P_k = \{x_k \pm \Delta_k^m e_i, i = 1..n\} \tag{3.2}$$

L'utilisation des ensembles générateurs positifs à une grande importance. En effet , si $D = \{d_1, d_2, ..., d_k\}$, $d_i \neq 0$ ensemble de vecteurs engendre R^n linéairement, alors pour chaque vecteur v non nul dans $\mathbf{R}^n$, il existe un indice i pour lequel $v^t d_i > 0$, c'est-à-dire qu'il existe au moins une direction dans chaque demi-espace. De plus, si la dérivée existe en un point x et $\nabla f \neq 0$ on prenant $v = -\nabla f(x)$ il existe un indice i tel que $\nabla f d_i < 0$ et on peut affirmer qu'il existe au moins une direction de descente dans D.

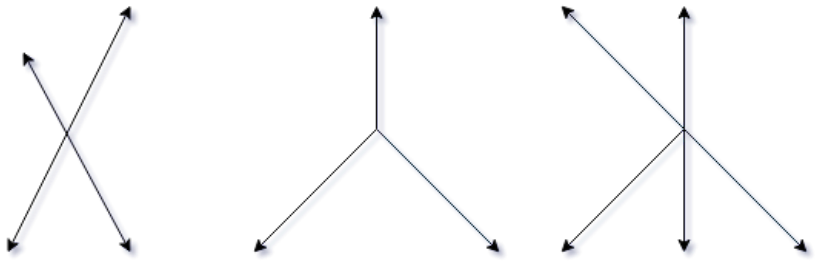


FIGURE 3.2 – Ensembles générateurs positifs dans $\mathbf{R}^2$

Soit l'ensemble V_k contenant l'historique des points où la fonction f a été évaluée depuis le début de l'itération k. Tous les itérés générés doivent appartenir au treillis, qui est une discrétisation de l'espace des variables, défini par :

$$M_k = \{x + \Delta_k D z \;\; z \in \mathbf{N}^n, x \in V_k\}, P_k \subset M_k \tag{3.3}$$

Dans l'algorithme de general pattern search, avant l'étape de recherche au voisinage de l'itéré courant ç.à.d dans l'ensemble P_k, l'algorithme propose une étape optionnelle appelée la recherche globale (search step). Cette étape de recherche globale consiste en l'évaluation de la fonction objectif en un ensemble de points générés sur le treillis M_k avec une stratégie de recherche. Cet étape est flexible car elle donne la possibilité à l'utilisateur de choisir des solutions qu'il juge intéressantes pour le type de problème traité. Il peut aussi implémenter ses propres stratégies de recherche ou des stratégies plus génériques comme la recherche aléatoire.

3.1.2 Algorithme de recherche par treillis adaptatifs

L'algorithme MADS : Mesh Adaptive Direct Search est une généralisation de l'algorithme GPS qui permet de résoudre des problèmes d'optimisation avec et sans contraintes.
En effet, l'algorithme MADS a amélioré un point faible de l'algorithme GPS le fait que les directions utilisées pour construire la discrétisation et l'ensemble P_k des itérés courants sont choisies dans un ensemble fini fixé D $\in \mathbf{R}^n$. L'analyse de convergence de l'algorithme dépend de cette condition et à cause de cela l'algorithme peut ne pas converger.

Pour remédier à ce problème l'algorithme MADS a considéré deux "step size" Δ_k^m et Δ_k^p pour construire respectivement the poll set P_k et l'ensemble M_k.

Les points de P_k sont construits autour du point courant x_k et la distance séparant x_k de chaque point de P_k est limitée par Δ_k^p.

Le paramètre de taille de maille Δ_k^m est mis à jour de façon que sa convergence vers zéro est plus rapide que celle de Δ_k^p . En conséquence les points de P_k etM_k seront choisit dans un maillage plus fin.

$$M_k = \{x + \Delta_k^m Dz : \ \ x \in V_k, z \in N^n\} \subset \mathbf{R}^n \tag{3.4}$$

$$P_k \subseteq \{x \in M_k : \|x - x_k\| \leq c\Delta_k^p\} \tag{3.5}$$

Par ailleurs, les paramètres Δ_k^m et Δ_k^p doivent respecter les conditions suivantes :

- $\Delta_k^m \leq \Delta_k^p \ \forall k$
- $\lim_{k\in K} \Delta_k^m = 0 <=> \lim_{k\in K} \Delta_k^p = 0 \ \forall$ K un sous-ensemble fini d'indices d'itérations.

La prise en compte des deux paramètres Δ_k^m et Δ_k^p séparées dans MADS, permet de construire un ensemble dense de directions donnant la possibilité d'évaluer des points dans n'importe quelle direction à l'intérieur du cadre de l'ensemble P_k.

On peut ainsi résumer l'algorithme MADS : À chaque itération la fonction f est évaluée pour un ensemble fini de points générés sur le treillis M_k : c'est l'étape de recherche globale. Ces valeurs sont comparées à la valeur de la fonction au point courant x_k. Dés qu'un point de l'ensemble donne une valeur de l'objectif inférieure à celle de la solution courante, ce point est retenu et l'itération prend donc fin suivant une stratégie opportuniste et est considérée comme un succès. Ainsi, l'itération suivante débute par une mise à jour de la nouvelle solution courante, x_{k+1} telle que $f(x_{k+1}) < f(x_k)$, et un accroissement de la taille du treillis Δ_m^k afin d'accélérer la convergence. Par contre, un échec de l'étape search conduit à la deuxième étape the poll step juste avant la fin de l'itération. Cette étape consiste en l'exploration locale de l'espace des variables autour de la solution courante dans des directions choisies avec plus de flexibilité.

3.1.3 Résultats de convergence

Les résultats de convergence sont basés sur deux hypothèses fondamentales [2] :

- **H1 :** La valeur de f à un point de départ réalisable est fini.
- **H2 :** Tous les itérés générés par MADS appartiennent à un ensemble compact.

Définition : La sous-suite des centres de sonde $\{x_k\}_{k\in K}$ dont les itérations ont été qualifiées d'échec, est dite une sous-suite raffinante si $\{\Delta_k^p\}_{k\in K}$ converge vers zéro, où K est un sous-ensemble d'indices. Soit x̂ le point vers lequel la suite raffinante converge. Si $\lim_{k\in L} \frac{d_k}{\|d_k\|}$ existe pour $L \subset K$,$d_k \in D_k$, et si $x_k + \Delta_k^m d_k \in \Omega$ pour un nombre infiniment grand k ∈ L, alors cette limite est dite une direction raffinante pour x̂.
Sous les hypothèses H1 et H2, il existe au moins une suite raffinante convergente.

Théorème : Sous les deux hypothèses H1 et H2, il existe au moins un point x̂ vers lequel la suite x_k converge. Si f est semi-continue inférieurement près de x̂, alors $\lim_{k\in K} f(x_k)$ existe et $\lim_{k\in K} f(x_k)$ ≥ f(x̂).

Définition : La dérivée généralisée de Clarke de f en x̂ dans la direction v ∈ $\mathbf{R}^n$ est

$$f^0(\hat{x}; v) = \limsup_{x->\hat{x}t\downarrow 0} \frac{f(x + tv) - f(x)}{t} \tag{3.6}$$

Théorème d'analyse de convergence de l'algorithme MADS : Si x̂ est la limite d'une suite raffinante, et si d est une direction dans D pour laquelle f a été évaluée un nombre infini de fois pour tous les itérés dans l'étape de recherche locale.
Si f est lipschitz près de x̂ alors la dérivée généralisée de Clarke de f au point x̂ dans la direction d est non négative : $f^0(\hat{x}; v) \geq 0$

3.1.4 Les versions de MADS :

MADS est conçu comme une généralisation de l'algorithme de recherche par coordonnées et l'algorithme de recherche par motif. En effet, il régle les inconvénients de deux algorithmes en gardant leurs principales propriétés. Il y a différentes versions de MADS, le seul point qui les distingue est le type des direction choisis dans l'étape de recherche local (poll step).

- GPS =MADS quand $\Delta_k^m = \Delta_k^p$
- LTMADS [2] :génère des matrices triangulaires non singulières de façon aléatoire pour générer les directions qui construisent the poll set P_k.
  En effet, Le principal résultat de la convergence de GPS est d'identifier les points limites x, où les dérivée généralisée de Clarke sont non négatives dans un ensemble fini de directions, appelées directions de raffinage. LTMADS propose une instance de MADS pour lesquels les directions de raffinage sont denses dans le cône hypertangent à x avec une probabilité de 1 à chaque fois que les itérés associés aux directions de raffinage convergent à un seul x.
- ORTHOMADS [6] : cette variance introduit une nouvelle stratégie déterministe pour générer les directions (le poll step). Cette nouvelle méthode génère une base orthogonale. L'ensemble des directions positives à chaque itération est $D_k = [H_k, -H_k]$ où les colonnes H_k forme une base orthogonale de $\mathbf{R}^n$. De plus, l'union de toutes les directions OrthoMads sur l'ensemble des itérations est dense dans la sphère unité.

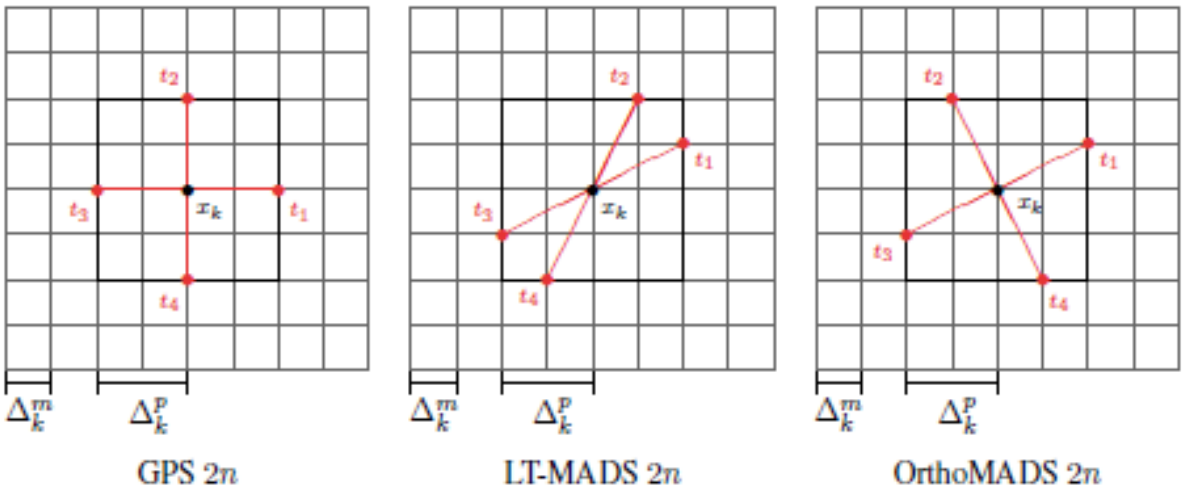


FIGURE 3.3 – Les différentes types de directions

3.1.5 Conclusion

Considérer les outils d'EDF qui calculent la VAN d'une stratégie de gestion d'actifs industriels comme une boite noire est l'idée qui m'a amené à étudier les méthodes de recherche direct qui sont très utilisées pour résoudre les problèmes d'optimisation de type boite noire. Ces méthodes ont été étudiées en détail par l'équipe GERAD au Canada qui les a implémentées dans un logiciel baptisé NOMAD.

3.2 Le logiciel NOMAD

NOMAD [8] (Nonlinear Optimization with the Mads Algorithm) est un logiciel qui implémente l'algorithme MADS codé en c++. Il est utilisé par des industriels, comme Boeing , Exxon Mobil et Hydro-Québec. L'idée est ici de voir si on peut se servir de ce logiciel pour les problématiques de gestion d'actifs industriels à EDF.
Le logiciel est conçu pour l'optimisation sous contraintes de fonctions boîtes-noires de la forme :

$$\min_{x \in \Omega} f(x) \tag{3.7}$$

où Ω est l'ensemble réalisable

$$\Omega = \{x \in X \qquad c_j(x) \leq 0, j \in J\} \tag{3.8}$$

X : un ensemble qui regroupe toutes les contraintes non quantifiables ainsi que les contraintes de bornes

3.2.1 Application de NOMAD à un cas non bruité

Après avoir étudié l'algorithme MADS , l'utilisation de NOMAD et ses paramètres. J'ai voulu le tester sur un cas de l'outil de EDF.

Comme déjà indiqué la réponse de VME est bruitée, donc pour pouvoir appliquer MADS à cette boite noire, on a besoin d'éliminer ce bruit. Pour analyser dans un premier temps les performances de NOMAD, un cas non bruité de calcul de VAN obtenu, non pas par simulation mais par une fonction numérique a été utilisé comme une boîte noire. Le cas traité est celui de quatre transformateurs de technologie identique installés sur deux tranches indépendantes. Les transformateurs d'une même tranche sont aussi indépendants. Chaque tranche fonctionne en série, c'est-'a-dire que la panne d'un transformateurs entraîne la mise en arrêt de sa tranche.

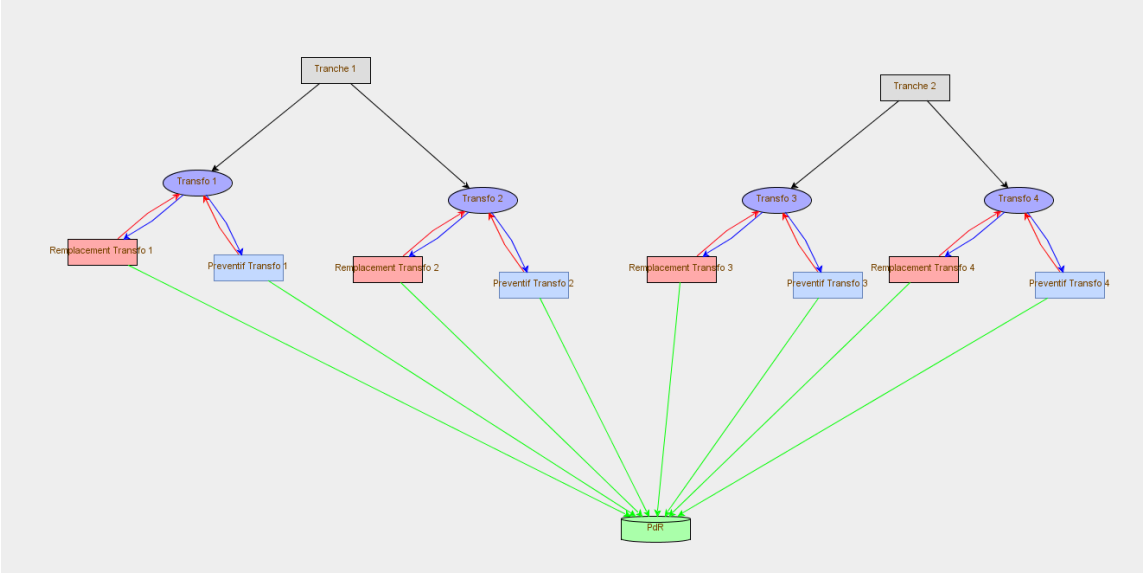


FIGURE 3.4 – Schémas de deux tranches dans le VME

Certains paramètres de NOMAD

- **La stratégie de recherche :** Dans l'étape de recherche globale dans NOMAD, il y a plusieurs stratégies qui peuvent être adoptées.
  La stratégie utilisée par défaut dans le logiciel NOMAD est la recherche spéculative : Si l'itération courante est qualifié de succès, cette stratégie consiste à évaluer la fonction à un point généré plus loin suivant la direction du dernier succès. Par ailleurs, les points générés par l'algorithme au cours d'une itération ne sont pas tous évalués car l'itération s'arrête au premier succès (opportunisme).
- **Les types des directions :** ce sont les types des direction utilisés dans l'étape de recherche locale (poll step). NOMAD utilise les 3 types de directions
  — Les direction de GPS ç.à.d la base de R^n
  — Les directions de LT-MADS :
  — Les directions de ORTHOMADS
  On précise aussi le nombre de directions utilisés.

Résultats : Il s'agit ici de minimiser la fonction f qui est l'opposé de la VAN(i.e : maximiser la VAN)

X_0	f	meilleur solution	Nb iter
(11 18 23 22 10)	-5554.35	(11.13 18.92 22.92 21.863 10.08)	812
(1 1 1 1 1)	-5554.58	(11.19 18.9 22.91 21.92 10.18)	23122
(20 20 20 20 20)	-5554.58	(11.19 18.91 22.91 21.92 10.18)	1935
(11 19 22 23 10)	-5554.52	(11.12 18.89 22.77 21.82 10.05)	1237

TABLE 3.1 – Les résultats pour un problème de minimisation de (-VAN) sans contraintes en nombres réels

X_0	f	meilleur solution	Nb iter
(11 18 23 22 10)	-5550.85	(11 18 23 22 10)	55
(1 1 1 1 1)	-5553.64	(11 19 23 22 10)	615
(20 20 20 20 20)	-5284.34	(13 18 22 40 12)	589
(11 19 22 23 10)	-5553.64	(11 19 22 23 10)	63

TABLE 3.2 – Les résultats pour un problème de minimisation de (-VAN) sans contraintes en nombres entiers

On converge dans le cas sans contraint vers le même optimum quelque soit le point de départ. En comparant les résultats obtenus en nombre entiers à ceux en nombres réels. On constate qu'on ne converge pas vers le même optimum pour différents points initiaux dans le cas des nombres entiers. Cela est due au pas de discrétisation Δ_k^m qui a le même type que les variables. Donc en se rapprochant de 0, Δ_k^m est tronqué à 0 ou à 1 dans le cas du nombres entiers et par la suite il y a des points proches de l'optimum qui ne sont pas évalués.

X_0	f	Feasible sol	Nb iter
(1 1 1 1 1)	-4835.22	(5.91 18.83 22.97 21.97 4.85)	3191
(11 19 22 23 10)	-4835.74	(5.93 18.84 21.73 22.69 4.87)	2161
(20 20 20 20 20)	-4835.59	(5.89 18.80 21.45 22.46 4.85)	2783
(1 36 2 2 30)	-4835.74	(5.93 18.82 21.73 22.70 4.86)	2920

TABLE 3.3 – Les résultats pour un problème de minimisation de (-VAN) avec contraintes

Les résultats de l'algorithme génétique utilisé à EDF sur le même problème non bruité sans contrainte donnent une VAN égale à 5553.64 qui est une valeur inférieur à la valeur donnée par NOMAD 5554 . Ce qui s'explique par le fait que cet algorithme génétique traite le problème en nombre entiers. Ce résultat nous a encouragé à essyer d'utiliser NOMAD dans le cas du VME bruité. Pour cela, il nous faut un algorithme d'optimisation robuste.

3.3 RobustMads

L'idée proposée consiste à appliquer une technique de lissage sur les valeurs de la fonction f permettant de diminuer le bruit [9]. En effet, pour déterminer le prochain itéré dans l'algorithme MADS, il faut comparer les points générés au cours de l'itération courante. Se fier aux évaluations de f n'est pas adéquat à cause du bruit présent dans cette fonction. Par ailleurs, étant donnée que l'on considère le VME comme une boîte-noire, on a seulement accès aux valeurs de la fonction. On applique donc une technique de lissage [9] sur la fonction f créant ainsi un algorithme capable de considérer une approximation de la fonction débruitée. La technique de lissage utilisée consiste à approcher la fonction f en un point x par une somme pondérée des observations (les valeurs de f en des points u) en choisissant des poids qui dépendent de la distance entre le point x et les points u. De façon que ces poids tend vers 1 en se rapprochant de x et vers 0 en s'éloignant.

$$\widetilde{f_k} = \frac{1}{P_k} \sum_{u \in V_k \cup W_k} g \circ d(x,u) f(u) \tag{3.9}$$

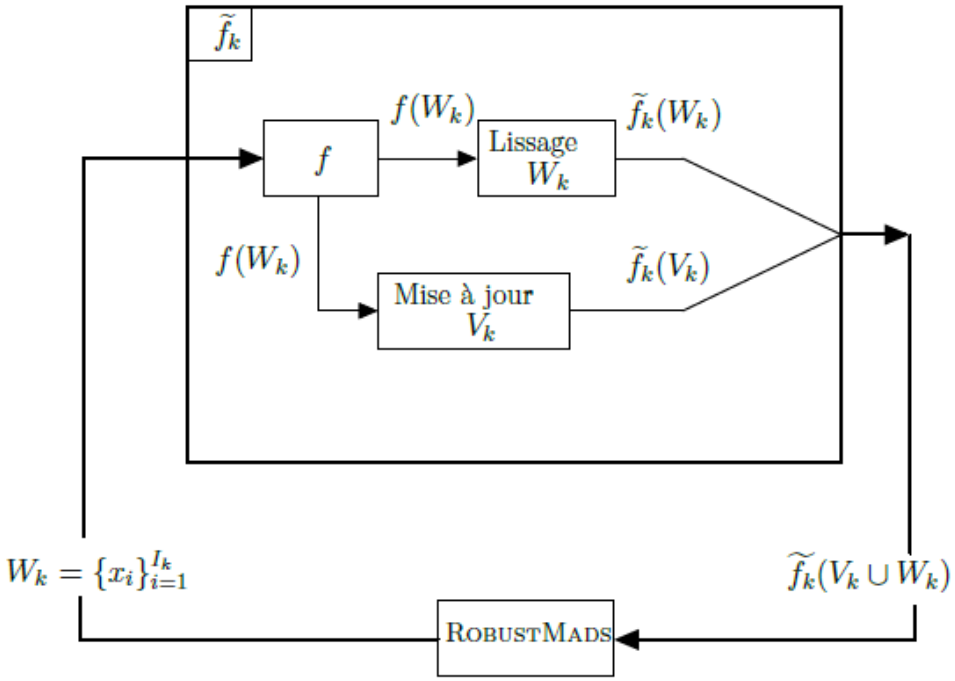
$$P_k = \sum_{u \in V_k \cup W_k} g \circ d(x,u) \tag{3.10}$$

- V_k : l'ensemble des points ou la fonction a été évaluée depuis la première itération jusqu'à l'itération k-1.
- W_k : le bloc de points d'entré à la boite noire.

avec une fonction de poids qui suit la loi normale, donnant aux points les plus proches un poids supérieur.

$$g(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x}{\sigma})^2} \tag{3.11}$$

et d(x,u) est la distance euclidienne.



À l'itération k : MADS analyse l'historique des points précédemment évalués et génère un bloc W_k de candidats par la recherche locale et globale et il envoie ce bloc à la boite noire qui va fournir les valeurs de f en ces points puis avec la technique de lissage on construit la fonction débruitée en tous points de W_k. On envoie enfin à MADS les valeurs de la fonction débruitée en W_k et V_k autrement dit V_{k+1} ($V_{k+1} = V_k \cup W_k$).
Notant que le calcul de la fonction débruité est mise à jour pour les points de V_k en considérant les nouveaux points de W_k.

3.4 Implémentation de l'algorithme

Pour implémenter cet algorithme dans NOMAD, j'ai été amenée à regarder le code source de NOMAD , de le comprendre et d'ajouter les modifications nécessaires pour implémenter RobustMads. Cette étape était difficile dans le sens ou on se trouve devant un code qui contient des milliers de lignes sans accés à la documentation de développement. Comprendre l'implémentation de différentes étapes de MADS dans le code de NOMAD était une étape primordiale :

L'exécution de l'algorithme MADS se fait en appelant dans le main la méthode **"run"** de la classe MADS de la bibliothèque NOMAD : l'algorithme commence par l'évaluation du point initial en appelant la méthode **"eval_x0"** de la classe **"Evaluator_control"**. Dans une boucle des itérations s'exécutent les étapes de recherche locale (par la méthode poll), et de recherche global (par la méthode search) et après le mise à jour des paramètres. Avec un appel de la méthode **"select_poll_center "** au début de chaque itération pour choisir le point courant qui est la meilleur solution à l'itération k-1. A chaque itération, la fonction des évaluations des points **"private_eval_list_of_points"** utilise la liste de points **_eval_lop** (attribut de la classe **"Evaluator_control"**) qui contient tous les points à évaluer (construite par the search and poll step) à l'itération k. Pour des raisons de parallélisation du code, Nomad évalue ces points par blocs avec la taille du bloc qui peut être fixée par l'utilisateur dans le fichier de paramètres puis le bloc des évaluations est envoyé pour faire le mise à jour des paramètres.

La structure de données centrale de NOMAD est **le cache** . Il correspond à l'ensemble V_k. Tous les

points qui ont été évaluées sont stockées dans la mémoire cache qui est consultée par NOMAD avant chaque nouvelle évaluation pour éviter les évaluations doubles.

Pour Modifier le code, j'ai commencée d'abords par enlever l'évaluation par blocs puisque j'ai besoin pour la construction de la fonction de lissage de tous le blocs d'évaluation envoyés par les recherches locales et globales en gardant bien évidement des implémentations identiques dans NOMAD. Ensuite, j'ai implémenté ma fonction de lissage dans la classe **"Evaluator_control"** qui code les formules (3.9) et (3.10) en utilisant la liste des points évalués à l'itération k et le cache de NOMAD.

J'ai aussi été amenée à changer la fonction d'évaluation, en effet, dans NOMAD, l'évaluation des points est effectuée seulement pour les points qui ne sont pas dans le cache. Dans le cas de ROBUST MADS, on n'a pas besoin de vérifier si le point existe dans le cache ou pas puisque la fonction objectif est bruitée et que l'on a besoin d'évaluer plusieurs fois un même point puisque l'évaluation sera différentes

Comme la stratégie de recherche utilisée dans NOMAD par défaut est la recherche spéculative soit dans la recherche globale comme je l'ai déjà expliqué , soit dans la recherche locale. Ainsi, dés que l'on trouve, à l'itération k ,un point meilleur que l'itéré courant , la nouvelle itération commence par ce point. Cette stratégie n'est plus possible à appliquer dans ROBUST MADS puisqu'à chaque itération la fonction de lissage de chaque points du cache est reconstruite ce qui nous oblige à choisir la meilleur solution de l'itération courante en comparant tous les points depuis le début des itérations.

Après avoir implémenté cette nouvelle méthode et compilé mon programme, j'ai appliqué le nouvel outil à une fonction test appelé **STYBLINSKI-TANG**

$$f(x) = 1/2 \sum_{i=1}^{d} x_i^4 - 16x_i^2 + 5x_i \tag{3.12}$$

En bruitant la fonction avec une variable qui suit la loi uniforme.

Malheureusement, l'algorithme n'a pas convergé et les fonctions de lissages n'étaient pas bien construites.
Après un débogage j'ai remarqué qu' après un certain nombre des itérations le cache est écrasé et tous les points évalués sont effacés. Cela s'explique par le fait qu'on changeant l'instanciation de la classe **"Evaluator_controle"** son cache est détruit par son destructeur. J'ai donc eu l'idée de faire un cache static, ce qui m'a permis de rendre cet attribut global et indélébile en modifiant l'instanciation de la classe.

Enfin, l'algorithme a convergé pour cette fonction vers une valeur proche de l'optimum qui est, dans la dimension deux $-78{,}33198$ et il est atteint en x^*= (-2.904, -2.904).

Le résultat est obtenu en utilisant valgrind qui est un outil performant de débogage mais il existe encore des problèmes informatiques qui empêchent l'outil de converger dans le mode exécutable .

3.5 Conclusion

L'objectif de mon travail dans cette partie était de chercher une méthode d'optimisation de boîte noire bruitée. Je pense que l'idée de lissage qui a été ajoutée peut nous donner des résultats performants dans le cas de notre étude, si on arrive bien sur à coder correctement l'algorithme dans le logiciel NOMAD. Bien que cette idée est adaptée à pour l'optimisation sans contraintes, elle pourra être améliorer pour optimiser des fonctions bruités sous des contraintes probabilistes. La fin de mon stage sera consacré au débugage complet du code afin de pouvoir le tester sur des cas de gestion d'actifs.

Chapitre 4

Modélisation d'actifs industriels pour l'optimisation robuste des stratégies de maintenance

Introduction

Les transformateurs sont des composants électriques qui ont un rôle très important dans l'exploitation d'une centrale de production d'électricité. En effet, pour rendre l'électricité transportable, les transformateurs augmentent le niveau de tension pour réduire au minimum les pertes d'énergie. En général, plusieurs transformateurs peuvent être placés sur une même tranche et pour des raisons de sécurité, lors de la panne d'un transformateur, sa tranche est arrêtée. La panne d'un transformateur peut être très coûteuse car elle peut induire une longue période d'indisponibilité de la tranche due a un délai d'approvisionnement très long si des pièces de rechange ne sont pas disponibles. Il est donc nécessaire de surveiller l'état des transformateurs et du stock de pièces de rechange et de planifier la maintenance des transformateurs ou leur remplacement et d'éviter des longues indisponibilités . Les transformateurs étant des composants électriques très coûteux, il est important de construire des stratégies de maintenance précises qui prennent en compte le vieillissement des transformateurs et d'anticiper les commandes de pièces de rechange.

4.1 Présentation du système :

On considère un seul transformateur. Ses durées de remplacement sont négligeables, on ne fait pas d'approvisionnement et on a un stock de pièces de rechange infini. Ce transformateur est soumis à une stratégie de maintenance préventive et une stratégie de maintenance corrective.

4.1.1 Stratégie de maintenance corrective

La maintenance corrective consiste a remplacer un composant défaillant par autre neuf.

4.1.2 Stratégie de maintenance préventive

En plus des remplacements correctifs effectues, la stratégie de maintenance préventive consiste à remplacer le composant à une date prévue, par un composant neuf également.

4.2 Présentation des processus markoviens déterministes par morceaux :

En faisant une analyse de la littérature sur le sujet, on constate que les processus stochastiques tel que les processus markoviens sont les plus utilisés pour modéliser l'évolution d'un système au cours du

temps [5]

4.2.1 Les processus semi-markoviens

la structure d'un processus semi-markovien (PMS) est celle d'un processus markoviens de sauts ç.à.d une chaîne de markov qui saute d'état en état au cours du temps, Afin d'introduire les processus semi-markoviens, je commence par définir un processus de renouvellement markovien :

Processus de renouvellement markovien Soit E un ensemble fini, soit $(Y_n)_{n\geq 0}$ un processus à valeurs dans E et $(T_n)_{n\geq 0}$ une suite croissante de variables positives :

$$0 = T_0 \leq T_1 \leq T_2 \leq ... \leq T_n \leq ... \tag{4.1}$$

Le processus $(Y, T) = (Y_n, T_n)_{n\geq 0}$ est un processus de renouvellement markovien a valeurs dans E si, pour tous $n \geq 0, i, j, i0, ..., i_{n-1}$ dans E et $t, t1, ..., t_n$ dans $\mathbf{R}_+$:

$$\mathbf{P}(Y_{n+1} = j, T_{n+1} - Tn \leq t | Y_0 = i_0, Y_1 = i_1, .., Y_n = i, T_n = t_n)) \tag{4.2}$$

$$= \mathbf{P}(Y_{n+1} = j, T_{n+1} - T_n \leq t | Y_n = i) \tag{4.3}$$

La suite $(Y_n)_{n\geq 0}$ est en fait une chaîne de Markov à valeurs dans E dont les temps entre les sauts suivent une loi à valeurs dans $\mathbf{R}_+$de type général.
Y_k modélise l'état dans lequel se trouve la chaîne de Markov après k sauts . Les instants de saut de la chaîne sont représentés par la suite de variables aléatoires $(T_n)_{n\geq 0}$.

Définition : Soit (Y, T) un processus de renouvellement markovien a valeurs dans E

$$X_t = Y_n \quad pour \quad T_n \leq t < T_{n+1} \tag{4.4}$$

Le processus$(X_t)_{t\leq 0}$ est le processus semi-markovien associe a (Y,T)

Exemple Prenons l'exemple d'un système de deux composants en parallèle. Le système subit une maintenance corrective lorsque les deux composants sont défaillants. Contrairement au cas précédent, nous considérons ici que les lois de durée de vie des composants ne sont plus de type exponentiel mais de type général ce qui peut modéliser un phénomène de vieillissement des composants. Soit T_1 la durée de fonctionnement du composant 1 et T2 la durée de fonctionnement du composant 2. La défaillance du système survient à l'instant $T_d = \max(T_1, T_2)$. Ce système peut être modélisé par un processus semi-markovien. L'espace E est composé de deux états : l'état de marche parfaite et l'état de défaillance, E =$\{(1, 1), (0, 0)\}$. Le noyau de transition de l'état (1, 1) vers l'état (0, 0) correspond à la loi de la variable aléatoire T_d. Cependant, une légère modification de la stratégie de maintenance du système peut rendre la modélisation par un processus semi-markovien impossible. Par exemple, si la défaillance d'un composant entraîne son remplacement sans tenir compte de l'état de l'autre composant, le système n'est plus modélisable par un processus semi-markovien. La difficulté vient du fait, qu'après la première défaillance de l'un des deux composants à l'instant $\min(T_1, T_2)$, le processus perd la mémoire de son passé et oublie la durée de fonctionnement de l'autre composant. Ce phénomène s'explique par le fait que le couple de deux processus semi-markoviens ne forme pas un processus semi-markovien.
Ainsi, je cherche un processus markovien plus souple capable de modéliser le vieillissement des composants, leurs dépendances et de nombreux types de stratégie de maintenance.
Il s'agit du processus markovien déterministe par morceaux (PDMP pour Piecewise Deterministic Markov Process)

4.2.2 Définition d'un PDMP

Les Processus Markoviens Déterministes par Morceaux ont été introduits par [Davis, 1984]. Ce sont des processus utilises en fiabilité dynamique pour modéliser des systèmes en interaction avec leur environnement. Un PDMP est un processus hybride $(Z_t) = (I_t; X_t)_{t\geq 0}$ les deux variables n'étant pas du même type. La première variable I_t est discrète et évolue dans un espace fini ou dénombrable E. Dans le cas d'un composant, cette variable peut représenter son état de marche ou de panne. La deuxième variable continue, représente les conditions environnementales (la température, la pression, $\cdots$) qui influent sur le système. Elle est aussi appelée variable physique. Dans nos cas d'applications, la variable X_t ne modélise pas les conditions environnementales mais la date de future panne des composants du système. La variable X_t prend ses valeurs dans l'espace continu $\mathbf{R}^d$. Le processus $(I_t, X_t)_{t\geq 0}$saute à des instants aléatoires isolés et évolue de la manière suivante :

- Le taux de transition de la variable I_t d'un état i vers un état j dépend de la variable physique X_t et est une fonction $a(i, j, X_t)$,
- Lorsque la variable I_t saute de l'état i vers l'état j à l'instant t, la variable physique peut sauter simultanément. La loi de X_t après ce saut, notée $\mu(i, j, x)(dy)$, dépend de l'état processus juste avant le saut et de l'état j de I_t après le saut,
- entre deux sauts du processus, la variable discrète I_t est constante et la variable continue X_t évolue de manière déterministe selon une équation différentielle qui dépend de l'état discret i :

$$\frac{dy}{dt} = v(i, y) \tag{4.5}$$

où v est est une fonction donnée. On note g (i, x, t) l'unique solution de l'équation (4.5) telle que

$$g(i, x, 0) = x \ \ \forall (i, x) \in E * \mathbf{R}^d. \tag{4.6}$$

- La transition du PDMP de l'état (i, x) vers l'état (j, y) est régie par le noyau de transition :

$$\tau(i, x; j, dy) = a(i, j, x)\mu(i, j, x)(dy) \tag{4.7}$$

L'évolution d'un PDMP est représentée dans la figure (4.1)

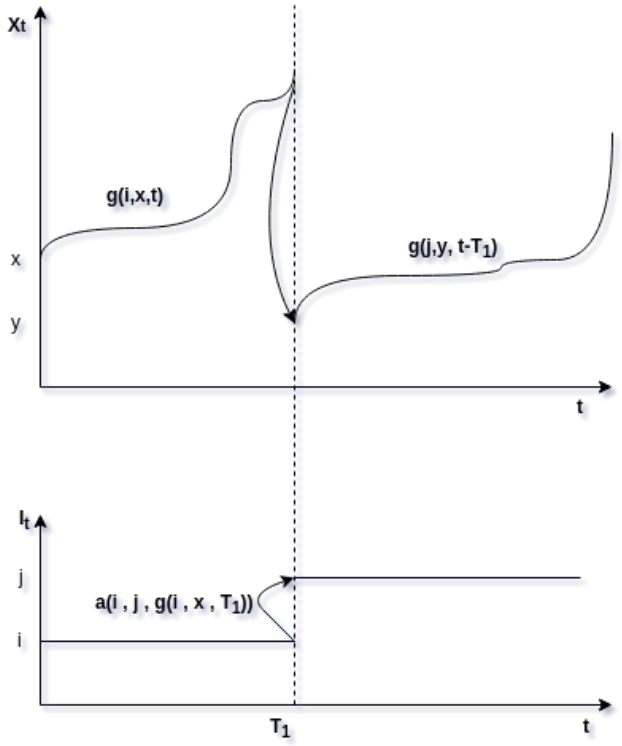


FIGURE 4.1 – Schéma d'évolution d'un PDMP

Un PDMP $(I_t, X_t)_{t\geq 0}$ peut également sauter lorsque la variable X_t atteint la frontière de son domaine. En fiabilité, la frontière peut représenter une date prévue pour une maintenance préventive des composants d'un système par exemple. Je n'utilise pas ce type de modélisation, puisque je considère que le temps de sortie de la frontière est l'horizon temporel H sur lequel je travaille.

4.2.3 Un exemple

Dans une usine une machine peut tomber en panne. Elle est alors envoyée à l'atelier pour réparation. Par ailleurs, la direction de l'usine a décidé que, lorsque la machine a fonctionné toute une année sans panne, elle est également envoyé à l'atelier pour la maintenance. On suppose que la réparation et la maintenance ont toujours la même durée de 7 jours et on suppose également qu'à l'issue de cette période, la machine est totalement réparée.

Modélisation par un PDMP :

L'état de cette machine est modélisée par un PDMP noté $Z_t = (X_t, I_t, t)$ avec :

- I_t prend 3 valeurs dans le temps 1, 2 ou 3.
  m=1 si la machine fonctionne bien
  m=2 si la machine est en réparation
  m=3 si la machine est en maintenance

- X_t est le temps depuis le dernier changement de mode.

4.2.4 Structure sous-jacente des PDMP

Un PDMP, processus à temps continu, est régi par une structure sous-jacente définie par son état aux instants de sauts. La trajectoire du PDMP étant déterministe entre deux sauts, tout l'aléa du processus se trouve aux instants de sauts. Il est alors possible de reconstruire entièrement le processus continu 'à partir de l'information aux différents instants de sauts.

la chaîne de Markov sous-jacente d'un PDMP permet de modéliser l'aléa du PDMP et elle est formée

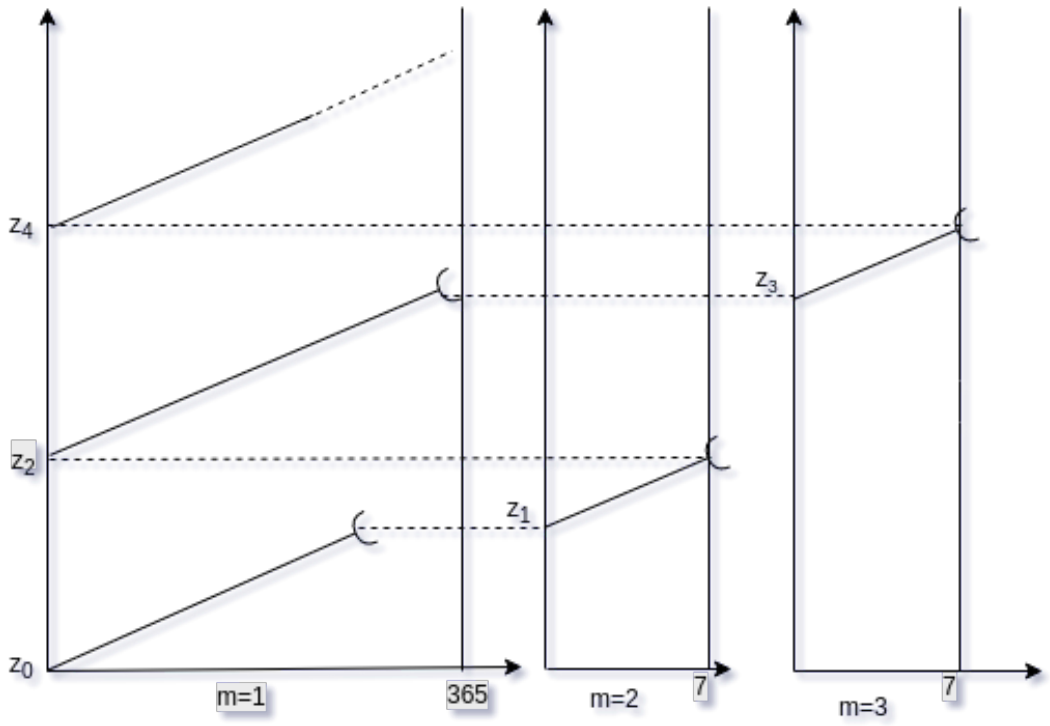


FIGURE 4.2 – La trajectoire du modèle de l'atelier de maintenance

des instants de sauts et de l'état du PDMP à ces instants. si $(T_k)_{k\geq 0}$ (avec $T_0 = 0$) représente les instants de sauts du PDMP $(I_t, X_t)_{t\geq 0}$ alors la chaîne de Markov sous-jacente est définie par :$(I_{T_k}, X_{T_k}, T_k)_{k\geq 0}$. Le processus continu est alors reconstruit à partir de la chaîne de Markov $(I_{T_k}, X_{T_k}, T_k)_{k\geq 0}$ par la relation suivante : pour $T_k \leq t < T_k + 1$

$$\begin{cases} I_t = I_{T_k} \\ X_t = g(I_{T_k}, X_{T_k}, t - T_k) \end{cases}$$

4.3 Modélisation de l'évolution d'un transformateur par PDMP

Notations générales :

- F : fonction de répartition de la loi de durée de vie du transformateur.
- H : horizon de temps
- ξ : instant prévue de maintenance préventive,
- α : tau d'actualisation ,
- c_r : coût d'un remplacement correctif,
- c_p : coût d'un remplacement préventif,

Je commence tout d'abord par modéliser l'évolution du système sous la stratégie corrective.

4.3.1 Modélisation de la stratégie de maintenance corrective :

Dans cette stratégie, le composant est remis en marche à chaque panne. Le PDMP considéré est constitue des variables suivantes :

- X_t^c la date de futur panne.
- Y_t^c est une variable discrète à valeur dans **N**. Le rôle de cette variable sera expliqué dans la méthode de quantification du PDMP.

- C_t^c : représente le coût cumulé actualisé à l'instant t.

Le processus $(Z_t^c)_{t\geq 0} = (X_t^c, Y_t^c, C_t^c, t)$ est un PDMP et la chaîne de Markov associé à ce processus est $Z_k = (X_{T_k}, Y_{T_k}, C_{T_k}, T_k)$ où T_k la date de saut de chaîne de Markov.

$$\begin{cases} Z_0^{(c)} = (X_0^{(c)}, 0, 0) \\ Z_{k+1}^{(c)} = \Phi(Z_k^{(c)}, W_{k+1}) \end{cases}$$

avec X_0 qui suit la loi de Weibull [1] et de fonction de répartition notée F et W_k variable aléatoire de loi uniforme dans [0,1)et la fonction de transition Φ est définit par :

$$\Phi((x, y, c, t), w) = (x + F^{-1}(w), y + 1, c + c_r \exp^{-\alpha x}, t) \tag{4.8}$$

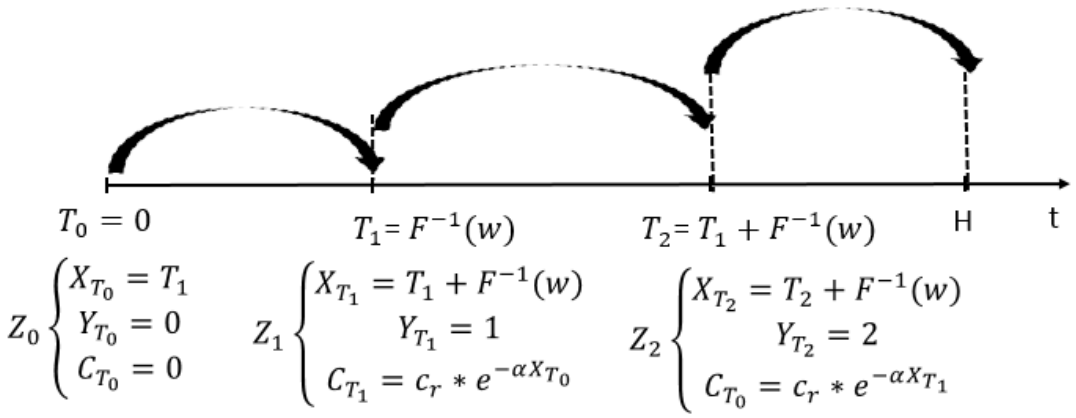


FIGURE 4.3 – La PDMP qui modélise la stratégie corrective

4.3.2 Modélisation de la stratégie de maintenance préventive :

La stratégie de maintenance préventive ne diffère de la stratégie de maintenance corrective qu'à travers le remplacement préventif de tous les composants à l'instant ξ.Ainsi, la chaîne de Markov qui modélise son évolution dans le temps va différer de celle de la stratégie corrective à partir l'instant ξ et est définie par :

$$\begin{cases} Z_k^{(c)} & k \leq N_\xi^{(c)} \\ Z_{k-(N_\xi^{(c)}+1)}^{(p)} & k > N_\xi^{(c)} \end{cases}$$

où $N_\xi^{(c)}$ est le nombre de sauts de la chaîne de Markov $Z^{(c)}$sur l'intervalle [0, ξ]

$$N_\xi^{(c)} = \sum_{k=0}^{N} \mathbb{1}_{\{T_k^{(c)} < \xi\}} \tag{4.9}$$

1. la loi de weibull est la loi utilisé à EDF pour pour générer les dates de pannes des composants . sa fonction de densité est $f(\lambda, x, k) = \frac{k}{\lambda}(\frac{x}{\lambda})^{k-1} e^{-(x/\lambda)^k}$

La chaîne $Z^{(p)}= (Z_k^{(p)})_{k\geq 0}$ décrit le système dans la stratégie de maintenance préventive à partir de l'instant ξ et évolue de la même manière que $(Z_k^{(c)})_{k\geq 0}$ à partir de ξ.
on note $Z^{(p)}$ le PMDM associé $Z^{(p)} = (Z_k^{(p)})_{k\geq 0} = (X_{T_k}^{(p)}, Y_{T_k}^{(p)}, C_{T_k}^{(p)}, T_k^{(p)})$ définit par :

$$\begin{cases} Z_0^{(p)} = (X_0^{(p)}, Y_0^{(p)}, C_0^{(p)}, \xi) \\ Z_{k+1}^{(p)} = \Phi(Z_k^{(p)}, W_{k+1}) \end{cases}$$

avec

$$X_0^{(p)} = \xi + X \tag{4.10}$$

$$C_0^{(p)} = C_{T_{N_\xi}}^{(c)} + c_p \exp^{-\alpha\xi} \tag{4.11}$$

$$Y_0^{(p)} = Y_{T_{N_\xi}}^{(c)} \tag{4.12}$$

Avec X qui suit la loi de Weibull.
En effet la chaîne $Z^{(p)}$ n'est qu'une translation dans le temps de la chaîne $Z^{(c)}$ de 0 à ξ.

Donc la VAN à maximiser s'écrit :

$$VAN(H) = C^{(c)}([0, H]) - C([0, H]) \tag{4.13}$$

avec

$$C^{(c)}([0, H]) = C_{T_{N_H^{(c)}}}^{(c)} \tag{4.14}$$

C'est le coût cumulé jusqu'à l'horizon H de la stratégie correctif.
Et

$$C([0, H]) = C_{T_{N_{H-\xi}^{(p)}}}^{(p)} \tag{4.15}$$

C'est le coût cumulé jusqu'à l'Horizon H de la stratégie préventive.

Comme notre objectif est de maximiser l'espérance de la VAN sur les dates de remplacement préventive. Donc pour des simplifier l'expression de la fonction objectif , on s'intéresse seulement à maximiser (- C([0,H)]).

4.4 Problème d'arrêt optimal

Mon objectif est de trouver la stratégie de maintenance préventive optimale. Il s'agit donc de déterminer la date de maintenance optimale. Cette date est choisie de manière à minimiser la fonction coût, autrement dit, à maximiser La VAN. De plus, cette optimisation doit prendre en compte le caractère aleatoire des pannes et de l'évolution du système.
En se basant sur les travaux réalises par de [4] concernant l'arrêt optimal sur les PDMP, j'ai voulu utilise une méthode qui me permet de chercher le meilleur instant pour effectuer la maintenance préventive.

4.4.1 Principe de l'arrêt optimal d'un PDMP :

Pour Simplifier le problème je considère que mon PDMP fait au plus deux sauts (N=2). Soit $z_0 = (x, 0, 0)$le point de départ de mon processus. Mon but est de trouver un instant $\tau < H$ qui maximise la fonction $\mathbf{E}_{z_0}[g(Z_\tau)]$. avec g est la fonction de performance du système. τ un temps d'arrêt adapte à la filtration du PDMP Z_t. Ce problème est typiquement un problème d'arrêt optimal qui consiste à résoudre le problème d'optimisation suivant :

$$v_0(z_0) = \sup_{\tau<H} \mathbf{E}_{z_0}[g(Z_\tau)] \tag{4.16}$$

La fonction v_0 s'appelle la fonction valeur du problème. Elle représente la performance maximale que l'on peut atteindre en partant de z_0. Elle peut être calculée de manière récursive en utilisant l'équation de programmation dynamique :

$$\begin{cases} v_N = g \\ v_n = L(v_{n+1}, g) \ \ pour\ 0 \leq n \leq N-1 \end{cases}$$

où l'opérateur L, permettant (partant de v_N) de calculer les v_k jusqu'à la fonction valeur v_0 est défini de la facon suivante :

$$L(w,h)(z) = \sup_{0 \leq u \leq H} J(w,h)(z,u) \vee Kw(z), \tag{4.17}$$

$$J(w,h)(z,u) = E[h(\Phi(Z_n,u)\mathbb{1}_{S_{n+1} \geq u} + w(Z_{n+1})\mathbb{1}_{S_{n+1}<u}|Z_n = z] \tag{4.18}$$

$$Kw(z) = E[w(Z_{n+1})|Z_n = z] \tag{4.19}$$

où $S_n = T_n - T_{n-1}$ et Φ est le flot de la PDMP Z_t.
Sur l'intervalle de temps avant le premier saut, la performance optimale est obtenue en prenant la meilleure des deux options suivantes : ne rien faire jusqu'au prochain saut (opérateur K) ou bien choisir l'instant donnant la performance maximale sur la portion déterministe de la trajectoire entre 0 et H(opérateur sup J).

Dans notre cas d'étude, où N=2, ont peut réécrire ces équations de la façon suivante :

$$\begin{cases} v_2 = g \\ v_1 = L(v_2, g) \\ v_0 = L(v_1, g) \end{cases}$$

4.5 Méthode numérique d'arrêt optimal :

Les équations de programmation dynamique qui portent sur les fonctions v_n peuvent se récrire comme une récursion sur les variables aléatoires $V_n = v_n(Z_n)$. En effet, on a $V_N = g(Z_N)$ et pour tout $0 \leq n \leq N-1$
Dans notre cas d'étude, le flot du PDMP est constant entre les saut puisque $X_t = T_{k+1}\ \forall\ T_k \leq t \leq T_{k+1}$

$$V_n = \sup_{0 \leq s \leq H} (E[g(Z_n \mathbb{1}_{S_{n+1} \geq s} + V_{n+1}\mathbb{1}_{S_{n+1}<s}|Z_n]) \vee E[V_{n+1}|Z_n], \tag{4.20}$$

Pour approcher les valeurs de cette suite j'utilise la méthode expliquée dans [**?**] qui consiste à :

- Discrétiser le temps sur une grille de temps notée G associé à l'intervalle [0,H].
- L'étape de quantification qui permet de transformer cette chaîne de Markov à espace d'état continu en des variables aleatoires discrètes. La quantification permet de construire des grilles adaptées à la loi du processus et non de discrétiser l'espace d'état du processus de façon régulière et arbitraire.

4.5.1 Discrétisation du temps

Soit :

- $\Delta \in [0, H]$
- $n = \mathbf{E}[\frac{H}{\Delta}] - 1$ avec $\mathbf{E}[x]$ la partie entière de x.

L'ensemble des points $(t_i)_{i\in\{0..n\}}$ avec est noté $t_i = i\Delta$ forme la grille du temps G.
Donc L'opérateur L après la discrétisation du temps s'écrit :

$$V_n = \sup_{u\in G}(E[g(Z_n \mathbb{1}_{S_{n+1}\geq u} + V_{n+1}\mathbb{1}_{S_{n+1}<u}|Z_n]) \vee E[V_{n+1}|Z_n], \tag{4.21}$$

4.5.2 Quantification de la chaîne de Markov

4.5.2.1 Principe de la quantification

La quantification d'une variable aléatoire X consiste en la recherche d'une grille finie de l'espace d'état telle que la projection $\hat{X}$ de X sur cette grille minimise une norme L_p de l'erreur $X - \hat{X}$.
La quantification d'une chaîne de Markov consiste en la quantification à chaque temps k de la variable aléatoire X_k.
Pour chaque $n \in \{0, 1, 2\}$, (que nous noterons par la suite Γ_n^Z) de N_n points $w_n = (w_n^1, ..., w_n^{N_n})$ à laquelle est associée la partition de Voronoi [7] $C_i(w_n)$ $i = 1, ..., N^n$ définie de la façon suivante : $C_i(w_n)$ est une partition borélienne telle que :

$$C_i(w_n) \subset \{y \in \mathbf{R}^3 : |y - w_n^i| = \min_{1\leq j\leq N_n} |y - w_n^j|\} \tag{4.22}$$

On considère alors pour tout n, le quantifieur de Voronoi de Z_n sur la grille w_n, la variable aléatoire $\hat{Z}_n^{w_n}$

$$\hat{Z}_n^{w_n} = Proj_{w_n}(Z_n) = \sum_{i=1}^{N_n} w_n^i \mathbb{1}_{C_i(w_n)}(Z_n) \tag{4.23}$$

Pour une grille donnée w_n, nous noterons alors $\hat{Z}_n^{w_n}$ la projection de Z_n sur la grille w_n. J'ai utilisé l'algorithme CLVQ (Competitive Learning Vector Quantization) pour construire des grilles optimales. Le processus $\hat{Z}_n^{w_n}$ ne conserve pas la propriété de Markov.Mais, on peut calculer la matrice de probabilité de transition à partir de $\hat{Z}_n$:

$$P(\hat{Z}_n = w_n^i|\hat{Z}_{n-1} = w_{n-1}^j) = \frac{P((\hat{Z}_n, \hat{Z}_{n-1}) = (w_n^i, w_{n-1}^j)}{P(\hat{Z}_{n-1} = w_{n-1}^j)} = \frac{P((Z_n, Z_{n-1}) \in (C_i(w_n), C_j(w_{n-1}))}{P(Z_{n-1} \in C_j(w_{n-1}))} \tag{4.24}$$

4.5.2.2 Opérateur associé à $\hat{Z}_n$

$$\hat{V}_n = \sup_{s\in G}(\mathbf{E}[g(\hat{Z}_n)\mathbb{1}_{\hat{S_{n+1}}\geq s} + \hat{V}_{n+1}\mathbb{1}_{\hat{S}_{n+1}<s}|\hat{Z}_n] \vee \mathbf{E}[\hat{Z}_{n+1}|\hat{Z}_n] \tag{4.25}$$

4.6 Approximation de la fonction valeur dans notre cas d'étude

Les $\hat{v}^k$ pour $k \in \{0, 1, 2\}$ sont définies de la façon suivante :

$$\begin{cases} \hat{v}_2(z) = g(z) \\ \hat{v}_1 = L(\hat{v}_2, g)(z) \\ \hat{v}_0 = L(\hat{v}_1, g)(z) \end{cases}$$

Calcul de $\hat{v}_2$:

$$\begin{cases} V_2 = g(Z_2) \\ si\ \ \hat{T}_2 \geq \xi \geq \hat{T}_1 \quad \hat{V}_2 = -(c_p e^{-\alpha\xi} + c_r(e^{-\alpha X^{(c)}_{T_0^{(c)}}} + e^{-\alpha X^{(c)}_{T_1^{(c)}}} + e^{-\alpha X^{(p)}_{T_0^{(p)}}})) \\ si\ \ \xi \geq \hat{T}_1 \geq \hat{T}_2 \quad \hat{V}_2 = -(c_p e^{-\alpha\xi} + c_r(e^{-\alpha X^{(c)}_{T_0^{(c)}}} + e^{-\alpha X^{(p)}_{T_0^{(p)}}} + e^{-\alpha X^{(p)}_{T_1^{(p)}}})) \\ si\ \ \hat{T}_2 \leq \xi \quad \hat{V}_2 = -(c_p e^{-\alpha\xi} + c_r(e^{-\alpha X^{(c)}_{T_0^{(c)}}} + e^{-\alpha X^{(c)}_{T_1^{(c)}}} + e^{-\alpha X^{(c)}_{T_2^{(c)}}})) \end{cases}$$

Calcul de $\hat{v}_1$:

$$\begin{cases} \hat{V}_1 = \sup_{s\in G}(\mathbf{E}[g(\hat{Z}_1)\mathbb{1}_{\hat{S}_2\geq s} + \hat{V}_2\mathbb{1}_{\hat{S}_2<s}|\hat{Z}_1]) \vee \mathbf{E}[\hat{V}_2|\hat{Z}_1] \\ = \sup_{s\in G}(g(\hat{Z}_1)\mathbf{P}(\hat{S}_2 \geq s|\hat{Z}_1) + \mathbf{E}[\hat{V}_2\mathbb{1}_{\hat{S}_2<s}|\hat{Z}_1]) \vee \mathbf{E}[\hat{V}_2|\hat{Z}_1] \\ si\ \ \hat{T}_1 \geq \xi \quad g(\hat{Z}_1) = -(c_p e^{-\alpha\xi} + c_r(e^{-\alpha X^{(c)}_{T_0^{(c)}}} + e^{-\alpha X^{(p)}_{T_0^{(p)}}})) \\ si\ \ \xi > \hat{T}_1 \quad g(\hat{Z}_1) = -(c_p e^{-\alpha\xi} + c_r(e^{-\alpha X^{(c)}_{T_0^{(c)}}} + e^{-\alpha X^{(c)}_{T_1^{(c)}}})) \end{cases}$$

Calcul de $\hat{v}_0$:

$$\begin{cases} \hat{V}_0 = \sup_{s\in G}(\mathbf{E}[\hat{V}_1\mathbb{1}_{\hat{S}_1<s}|\hat{Z}_0]) \vee \mathbf{E}[\hat{V}_1|\hat{Z}_0] \end{cases}$$

Remarque : les espérances sont calculées de la façon suivant à partir de grille de quantification :

$$\mathbf{E}[\hat{v}(Z_{i+1})\mathbb{1}_{\hat{S}_{i+1}<s}|\hat{Z}_i] = \sum_{(Z,S)\in\Gamma^Z_{i+1}} \hat{v}_{i+1}(Z)\mathbb{1}_{S<s}\mathbf{P}((Z,S)|\hat{Z}_i) \tag{4.26}$$

4.7 Approximation du temps d'arrêt optimal

Pour construire le temps d'arrêt optimal dans notre cas d'étude (c'est plus rigoureux de dire quasi optimal puisque on cherche le temps qui nous donne une performance proche de l'optimum) de la même manière faite dans [4] :
On définit $s^*_n(Z,S)$ qui représente l'instant qui maximise la fonction de performance lorsqu'au n − 1 ième saut, le système est défini par $Z_{n-1}, S_{n-1} = (Z,S)$

$$s^*_n(Z,S) = \min\{s \in G|\hat{J}_n(\hat{v}_n,g)(\hat{Z}_{n-1},s) = \max_{u\in G}\hat{J}_n(\hat{v}_n,g)(\hat{Z}_{n-1},u)\} \tag{4.27}$$

et $r_n(Z,S)$ qui représente soit le temps d'utilisation maximal H si sa performance est meilleure quand nous laissons le système évoluer , soit il correspond au meilleur instant s^*_n pour arrêter le système.

$$r_n(Z,S) = \begin{cases} H \ \ si\ \ \hat{K}_n(\hat{v}_n(\hat{Z}_{n-1})) > \max_{u\in G}\hat{J}_n(\hat{v}_n,g)(\hat{Z}_{n-1},u)\} \\ s^*_n(Z,S) \end{cases}$$

avec $(\hat{Z}_{n-1}, \hat{S}_{n-1})$ la projection de (Z,S) sur la grille Γ^Z_{n-1}.

On construit à partir de ces éléments les temps d'arrêt $(\tau_n)_{n\in\{1,2\}}$ telque :

$$\tau_1 = r_2(Z_0,S_0) \wedge T_1 \tag{4.28}$$

et

$$\tau_2 = \begin{cases} r_1(Z_0,S_0) \ \ si\ \ T_1 > r_1(Z_0,S_0) \\ T_1 + \tau_1 \circ \theta_{T_1} \end{cases}$$

avec $\tau_1 \circ \theta_{T_1} = r_2(Z_1, S_1) \wedge (T_2 - T1)$

Le temps d'arrêt optimal est τ_2

Le principe est le suivant :
À l'instant initial (t=0), nous calculons une première date $r_1(Z_0, S_0)$ qui dépend de (Z_0, S_0). On laisse le système évoluer jusqu'à $r_1(Z_0, S_0) \wedge T_1$:

- Si $T_1 \leq r_1(Z_0, S_0)$ alors la date de maintenance est $r_1(Z_0, S_0)$,
- Sinon, à l'instant T_1 on calcule la deuxième date $r_2(Z_1, S_1)$ Cette date correspond au temps qui maximise $\hat{v}_1(\hat{Z}_1)$ si $\hat{K}_2(\hat{v}_2(\hat{Z}_1)) < \max_{u \in G} \hat{J}_2(\hat{v}_2, g)(\hat{Z}_1, u)\}$ on laisse ensuite évoluer le système jusqu'au temps $(r_2(Z_1, S_1)+T_1) \wedge T_2$ (où T_2 est le deuxième temps de saut). Si $(r_2(Z_1, S_1)+T_1) \leq T_2$ alors notre date de maintenance est $r_2(Z_1, S_1) + T_1$ sinon cela signifie que le composant est tombé en panne avant cette date de maintenance proposée.

4.8 Conclusion

Le système sur le quel j'ai travaillé dans cette partie est simple par rapport au modèle réel à EDF, mais j'ai encore des idées pour améliorer la modélisation par les PDMP afin de prendre en compte de la gestion du stock et des stratégies d'approvisionnement mais le plus important maintenant est soit d'arriver à des résultats performants avec cette approche dans un temps de calcul raisonnable soit de conclure que ce type de méthodes n'est pas adaptés pour les cas volumineux et que le recours à de l'optimisation de boîtes noires, type MADS, est nécessaire.

Chapitre 5

Conclusion

Le sujet de mon stage m'a donné la chance d'explorer deux domaines d'optimisation différents : Le premier concerne l'optimisation de boites noires où j'ai été amenée à lire différents articles sur les méthodes de recherche directs, notamment l'algorithme MADS, qui sont capables de résoudre des problèmes d'optimisation difficile sans avoir accès aux expressions analytique du problème. J'ai eu, grâce à cette approche, l'occasion de développer dans le logiciel NOMAD et d'approfondir mes compétences en langage de programmation C++. Le second domaine que j'ai découvert pour la première fois est celui de la modélisation par les processus markoviens et la résolution des équations de programmation dynamique par la méthode de quantification.

Ces six mois de stage en tant qu'ingénieur Recherche & Développement au sein du groupe EDF, ont été, pour moi, l'occasion d'appliquer à des fins industrielles, les diverses compétences acquises tout au long de ma formation . Cette période d'immersion m'a permis de découvrir les exigences de rigueur et de travail du monde de la recherche. J'ai pu accroître mes compétences techniques mais également relationnelles notamment en collaborant étroitement avec différents corps de métier.

Enfin, l'autonomie et la marge de manœuvre laissées par mon tuteur m'ont permis de développer non seulement ma confiance en moi et en mon travail, mais aussi mes compétences organisationnelles, qui me serviront appréhender au mieux mes débuts en tant qu'ingénieur chercheur en optimisation.

Bibliographie

[1] AUDET C. GARNIER V. LE DIGABEL S. et LECLAIRE L.-A. ALARIE, S. (2013) snow water equivalent estimation using blackbox optimization pacific journal of optimization, 9, 1–21.

[2] C. Audet and Jr. J.E. Dennis. Mesh adaptive direct search algorithms for constrained optimization. siam journal on optimization, 17(1) :188–217, 2006.

[3] Charles AUDET. A survey on direct search methods for blackbox optimization and their applications.

[4] Benoîte de Saporta. Contributions à l'estimation et au contrôle de processus stochastiques.

[5] William Lair. Modélisation dynamique de systèmes complexes pour le calcul de grandeurs fiabilistes et l'optimisation de la maintenance.

[6] J.E. Dennis Jr. M.A. Abramson, C. Audet and S. Le Digabel. OrthoMADS :. A deterministic mads instance with orthogonal directions. siam journal on optimization, 20(2) :948–966, 2009.

[7] Huyên PHAM. Méthodes de quantification optimale et applications en finance.

[8] Christophe Tribes Sébastien Le Digabel and Charles Audet. Nomad user guide.

[9] Charles Audet · Amina Ihaddadene · S´ebastien Le Digabel · Christophe Tribes. Robust optimization of noisy blackbox problems using the mesh adaptive direct search algorithm.