Contribution à la conception robuste de réseaux électriques de grande dimension au moyen des métaheuristiques d'optimisation

Boussaad Ismail

UNIVERSITÉ PARIS-EST

ÉCOLE DOCTORALE MATHEMATIQUES ET STIC

(MSTIC, E.E. 532)

THÈSE DE DOCTORAT EN INFORMATIQUE

par

Boussaad ISMAIL

Sujet de la thèse

Contribution à la conception robuste de réseaux électriques de grande dimension au moyen des métaheuristiques d'optimisation

Thèse dirigée par le Professeur Patrick SIARRY

soutenue le 06 mai 2014

Jury :

Laurent DUMAS	Professeur à l'Université de Versailles St Quentin en Yvelines	Rapporteur
Farouk YALAOUI	Professeur à l'Université de Technologie de Troyes	Rapporteur
Gérard BERTHIAU	Professeur à l'Université de Nantes	Examinateur
Mohamed SLIMANE	Professeur à l'Université de Tours	Examinateur
Frédéric HELIODORE	Ingénieur, Alstom Grid	Examinateur
Serge POULLAIN	Ingénieur, Alstom Grid	Examinateur
Amir NAKIB	Maitre de Conférences à l'Université Paris-Est Créteil	Examinateur
Patrick SIARRY	Professeur des Universités à l'Université Paris-Est Créteil	Directeur de thèse

REMERCIEMENTS

Je tiens tout d'abord à remercier mon directeur de thèse Patrick Siarry, mes co-encadrants Amir Nakib, Frédéric Héliodore, Serge Poullain et Eric Courbon pour m'avoir encadré tout au long de cette thèse, pour leurs conseils si pertinents et si précieux et pour nos discussions très bénéfiques. Je tiens à remercier encore une fois mon équipe @RiskTeam, Frédéric Héliodore, Serge Poullain et Eric Courbon pour la bonne ambiance tout au long de ma thèse et pour le partage de leurs compétences si vastes et si précieuses avec moi.

Je tiens à remercier les professeurs Laurent Dumas et Farouk Yalaoui d'avoir bien voulu relire mon manuscrit de thèse et pour leurs commentaires et critiques si constructifs. Je remercie aussi les professeurs Gérard Berthiau et Mohamed Slimane d'avoir accepté de participer au jury de ma thèse.

Enfin, j'adresse mes remerciements chaleureux à toute ma famille, ma tante et sa famille, et à ma femme pour leur soutien indéfectible.

cette thèse est dédiée à : - Ma femme et ma famille

- L'équipe @RiskTeam d'Alstom Grid

RÉSUMÉ

Comme beaucoup de systèmes, un réseau électrique doit faire face à des pannes qui, compte tenu de sa grande connectivité, peuvent s'étendre à des régions entières : on parle alors de *blackout* (phénomène d'avalanche), c'est-à-dire ayant des conséquences à grande échelle. La taille des réseaux électriques et leur complexité rendent diffcile la compréhension de ces phénomènes qui émergent localement. Un certain nombre de travaux existe et se fond sur un usage intensif des outils de physique statistique. L'adaptation de méthodes de percolation et les *systèmes critiques auto-organisés* sont autant d'outils de choix pour décrire les propriétés statistiques et topologiques d'un réseau. Les outils d'optimisation par métaheuristiques, plus particulièrement l'optimisation par essaim de particules (OEP, ou PSO en anglais) et les algorithmes génétiques (AGs), se sont révélés être la pierre angulaire de ce travail et ont permis de définir des structures opérationnelles.

Les travaux développés dans ce domaine sont encore émergents et cette thèse y amène une contribution à plusieurs titres. Nous avons mis tout d'abord à profit des techniques d'optimisation afin de mieux "rigidifier" un réseau électrique en couplant la topologie de ce dernier au maintien des tensions aux noeuds du réseau par implémentation de FACTS (*Flexible Alternative Current Transmission System*). Pour le placement optimal de FACTS, l'objectif est de déterminer la répartition optimale de la puissance réactive, en relation avec la localisation et le dimensionnement optimal de FACTS, afin d'améliorer les performances d'un réseau électrique. Quatre principales questions sont alors abordées : Où placer des FACTS dans le réseau ? Combien de FACTS ? Quelle puissance attribuer à ces FACTS ? Quel(s) type(s) de FACTS ? A quel prix ?.

Dans cette thèse, toutes ces questions seront modélisées et abordées d'un point de vue électrique et optimal en appliquant, dans un premier temps, l'optimisation par essaim de particules OEP basique puis, dans un deuxième temps, en proposant un nouvel algorithme OEP (α-SLPSO) et une recherche locale (α-LLS) s'inspirant ainsi du concept de l'OEP basique et des lois de probabilité stables dites «α-stables de Lévy». Par ailleurs, l'ampleur du projet défini par l'équipe **@RiskTeam** *d'Alstom Grid* oblige l'utilisation de plusieurs techniques (tirées de la physique, des statistiques, etc.) destinées à des fins particulières dont l'estimation des paramètres des lois α-stable de Lévy. Face à l'échec des techniques déjà existantes pour l'estimation des lois $\alpha-$stable de paramètre $\alpha < 0.6$, nous proposons un nouvel estimateur semi-paramétrique de cette famille de probabilité utilisant les métaheuristiques pour résoudre le problème d'optimisation sous-jacent.

Enfin, en annexe de cette thèse, un outil d'aide à la décision destiné à une équipe interne d'*Alstom Grid* qui consiste en l'optimisation de la topologie interne d'un parc éolien est détaillé dans le dernier chapitre.

Mots clés : *Optimisation, métaheuristiques, optimisation par essaim de particules, algorithmes génétiques, FACTS, réseaux électriques, lois α-stables, topologie, parc éolien.*

ABSTRACT

Like many systems, an electrical power grid must contend with faillures which, given its higth connectivity, could spread to entire regions : this is referred to *blackout* (avalanche phenomena), ie. with large-scale consequences. The size of power grids and their complexity make difficult to grasp these locally emergent phenomena. There is a number of existing works that were based on extensive use of statistical physics tools. The adaptation of percolation's methods and the Self-Organized-Criticality systems provide practical tools to describe the statistical and topological properties of a network. Optimization tools by metaheuristics particularly, particle swarm optimization (PSO) and genetic algorithms (GA) have proved to be the cornerstone of this work and helped to define operational structures.
Works developed in this area are still emerging. This thesis brings a contribution in several ways. First of all, we have taken advantage in oprtimization technics to better "stiffen" a power grid by coupling its topology with maintaining voltages at the nodes of the network using FACTS (Flexible Alternative Current Transmission System). In the optimal location FACTS problem, the objective is to determine the optimal allocation of reactive power, in relation to the location and optimal sizing of FACTS, in order to improve the performance of the power grid. Four main issues are then discussed : 1) Where to place FACTS in the network ? How many FACTS ? What power attributed to these FACTS ? What type(s) attributed to these FACTS ? At what prices ?
In this thesis, all these questions will be modeled and discussed from the point of view of optimal power by applying, firstly, the strandard particle swarm optimization and by proposing a novel particle swarm optimization (α-SLPOS) and a local search (α-LLS). These two algorithms are inspired by the basic concept of PSO and the stable distributions (α-stable laws). Moreover, the scope of the project defined by the team **@RiskTeam** Alstom Grid requires the use of several techniques (from physics, statistics, etc) for particular purposes including the α-stable parametere estimation problem. Facing the failure of the existing methods for estimating the parameters of α-stable laws for $\alpha < 0.6$, we propose a novel semi-parametric estimator for such of probability distribution familly using metaheuristic to solve the underlying problem of optimization.

Finally, in the end of the thesis, a decision support tool is designed for an internal team of Alstom Grid to optimize the internal topology of a wind farm.

Keywords : *Optimization, metaheuristics, particle swarm optimization, genetic algorithms, FACTS, electrical power grid, α-satble law, topology, wind farm.*

TABLE DES MATIÈRES

INTRODUCTION GÉNÉRALE

Cette thèse CIFRE a été préparée au sein de l'Université Paris-Est Créteil (UPEC), *dans le Laboratoire Images, Signaux et Systèmes Intelligents* (**LiSSi, E.A. 3956**), et du groupe **@RiskTeam** au sein de l'entreprise *Alstom Grid* (anciennement *Areva TD*). Elle a été dirigée par le Professeur P. Siarry, Directeur de l'équipe Traitement de l'Image et du Signal, et co-encadrée par Monsieur A. Nakib, maître de conférences à l'université Paris-Est Créteil et tutorée, au sein d'*Alstom Grid*, par Monsieur F. Héliodore, responsable du groupe *@RiskTeam* et Messieurs S. Poullain et E. Courbon.

Les réseaux électriques constituent des éléments clés pour assurer une alimentation fiable et de qualité en énergie électrique. Ces dernières années, ceux-ci ont connu un accroissement considérable des interconnexions et ont été exploités au plus près de leurs limites de stabilité et de sécurité en raison des contraintes économiques et d'une opposition croissante à la construction de nouveaux ouvrages. La gestion de ces grands réseaux et, en particulier la coordination des systèmes de contrôle / commande sont rendues difficiles par effet de taille. Les perturbations inévitables, quelle que soit leur nature, peuvent amener le réseau à tout instant en dehors de sa zone de stabilité. Les exemples récents des différents *blackouts* en sont une illustration patente.

Ainsi les pannes et incidents associés à un réseau électrique de grande taille (ou à des réseaux électriques) font l'objet d'études spécifiques d'ingénierie, en lien direct avec la conception et l'évolution de ceux-ci. En ce qui concerne les problèmes tels que l'effondrement de la tension, la congestion de réseaux ou le phénomène de *blackout* (phénomène d'avalanche), c'est-à-dire ayant des conséquences à grande échelle, la complexité topologique du réseau électrique ne peut en être la seule cause et des paramètres tels que dérégulation et commerce énergétique tendent à amener le réseau dans ses conditions limites de fonctionnement.

Le réseau électrique peut alors être considéré comme un système complexe en conditions limites de fonctionnement où l'augmentation régulière de la demande électrique associée à l'amélioration de la qualité du réseau (dimensionnement, entretien, stabilité, ...) peuvent devenir des forces antagonistes, conduisant alors à des perturbations sensibles.

L'analyse des travaux issus de la littérature scientifique laisse apparaître différentes voies de modélisation, les unes privilégiant la nature topologique du réseau, les autres s'orientant sur la dynamique des phénomènes mis en jeu. Ces travaux bâtissent le pont rapprochant le domaine de la physique statistique

du monde électrique. Le dénominateur commun devient l'analyse de la complexité pour mieux la servir. Le premier ordre dans cette complexité est l'existence de lois d'échelles explicitant ainsi une dynamique entre ces différents niveaux d'échelles à travers, par exemple, l'apparition de lois de puissance. Le modèle de *criticalité auto-organisée* [Dobs01a] est maintenant largement accepté par la communauté scientifique, il est à la base des travaux développés dans cette thèse.

Comme il a été mentionné, cette thèse a été développée en partenariat et s'est intégrée dans un programme de travail initié à Alstom Grid visant à développer des outils d'analyse pluridisciplinaires, en particulier dans le domaine de l'analyse de risques. Si les outils d'optimisation sont déjà largement implantés dans l'ingénierie électrique, les outils d'analyse statistique semblent maintenant aussi vouloir s'imposer dès lors que l'estimation de risques technologiques, économiques ou financiers devient une nécessité, en particulier dans l'aide à la décision.

Plusieurs objectifs ont été fixés dans cette thèse, certains scientifiques, d'autres applicatifs, avec comme fil d'Ariane, la volonté d'intégrer du pluridisciplinaire sans pour autant produire un catalogue de techniques. Le métier a évolué et l'on doit pouvoir être à même d'analyser, formuler et développer dans des domaines connexes et dans des délais raisonnables.

Le synoptique des travaux réalisés dans cette thèse est alors le suivant :

– Chapitre 1

Un état de l'art est nécessaire dès lors que l'on veut se positionner sur une thématique scientifique. Le cœur de nos préoccupations étant relatif à l'optimisation, nous nous sommes focalisés sur les techniques d'optimisation par métaheuristiques et surtout l'optimisation par essaim de particules (OEP). L'interrogation principale portera ici sur les recherches locale et globale des informatrices et leur combinaison. Les éléments essentiels des outils utilisés pour analyser une certaine complexité seront présentés à la suite. Il s'agit des propriétés décrites par les graphes tant topologique que dynamique. Nous aborderons quelques propriétés issues de la théorie de la percolation. Ce chapitre se terminera par une présentation de dispositifs électriques permettant de gérer la puissance réactive. Ces dispositifs appelés FACTS sont des réponses concrètes à la stabilisation du réseau électrique.

– Chapitre 2

Nous avons prétendu nous intéresser au champ pluridisciplinaire. Le chapitre 2 peut en être une réponse en présentant un estimateur des paramètres des lois α-stables, basé sur l'utilisation des métaheuristiques notamment, l'optimisation par essaim de particules. L'intérêt des lois $\alpha-$stables nous est apparu quand nous nous sommes intéressés aux recherches locale et globale (intensification & diversification). Inspirée de la nature et démontrée comme optimale, i.e. la recherche de nourriture par des animaux, cette stratégie de recherche ouvre des perspectives intéressantes si l'on se place dans le cadre de l'invariance d'échelle. Cet aspect a été établi par B. Mandelbrot [[Mand63]] par l'introduction des effets Noah et Joseph, respectivement généralisation du bruit Gaussien et effet de persistance. Nous présentons quelques propriétés fondamentales des lois α-stables, le test statistique de Kolmogorov-Smirnov et le générateur de lois α-stables de McCulloch [Cham76] qui serviront de plate-forme à la construction de notre estimateur. Le problème d'optimisation sous-jacent et l'algorithme d'estimation des paramètres α-stables par essaim de particules sont ensuite détaillés. Le chapitre se termine par une analyse expérimentale accompagnée d'une proposition de parallélisation de l'algorithme mis en œuvre.

– Chapitre 3

Le chapitre 3 est dédié à la mise en œuvre d'une solution de placement et dimensionnement de FACTS (*Flexible Alternative Current Transmission System*) dans un réseau électrique. Le choix porté sur les algorithmes d'essaim de particules se justifie par l'intégration de la stratégie de recherche basée sur l'$\alpha-$stabilité. Une description du modèle électrique "*Power Flow*" est nécessaire à la modélisation du système. A la suite, les algorithmes par essaim de particules (variables discrète / continue) sont détaillés, en insistant sur les algorithmes "*Velocity Likelihoods* " et leur version modifiée pour traiter les variables discrètes du problème de placement de FACTS. Deux nouveaux algorithmes d'optimisation par essaim de particules respectivement désignés par α- SLPSO et α -LLS forment le cœur de ce chapitre. Le premier constitue l'intégration de la stratégie de recherche mélangeant recherche locale et recherche globale. Le second est dédié à la recherche locale et fait l'objet de perspectives. La dernière partie se concentre sur la validation aux cas des réseaux électriques en considérant deux *benchmarks*, les réseaux IEEE 30 nœuds et IEEE 57 nœuds.

– Chapitre 4

Avec le quatrième chapitre, nous revenons sur l'analyse du réseau électrique en tant que système complexe et exposons quelques résultats, issus de la théorie des systèmes complexes, concernant la topologie du réseau colombien. En particulier, quelques invariants topologiques et une étude de robustesse du réseau sont présentés. Cette analyse pourrait paraître indépendante du corpus. Il n'en est rien et nous nous sommes exercés à implémenter l'ensemble de ces techniques dans un outil logiciel générique. Celui ci constitue la base des modules logiciels que nous avons souhaité développer dans ce travail.

– Chapitre 5

Le cinquième chapitre converge encore un peu plus vers la définition d'outils logiciels. Il s'agit de la mise en place d'un outil d'aide à la décision relatif à l'optimisation de la topologie interne d'un parc éolien. Cette thèse CIFRE a eu aussi pour but de valider l'adaptabilité à toute situation industrielle pouvant survenir. Nous avons défini un cadre virtuel où il nous serait demandé la délivrance d'un outil d'aide à la décision sur une problématique ciblée, à résoudre sur un temps donné tout en proposant une solution innovante. Nous détaillons alors la problématique, à savoir la détermination de l'interconnexion optimale de fermes éoliennes, la raison du choix des algorithmes génétiques et enfin, sa validation sur une ferme de 30 éoliennes.

Les annexes et les références bibliographiques complètent l'ouvrage.

CHAPITRE 1

MÉTAHEURISTIQUES D'OPTIMISATION ET SYSTÈMES COMPLEXES : ÉTAT DE L'ART

1.1. Introduction

Nous sommes très souvent confrontés à des problèmes d'optimisation dans la vie courante. En effet, en passant par un simple rangement de bureau ou encore le placement de l'immobilier de façon optimale, nous nous heurtons à divers problèmes de décision plus ou moins complexes. De même, en industrie, les problèmes d'optimisation sont aussi très fréquents : en télécommunication (conception d'antennes, routage, ...), transport (tracé autoroutier, gestion de contraintes, gestion des feux, ...), aéronautique (conception des ailes d'avion, trafic aérien, ...). En pratique, la plupart de ces problèmes sont difficiles à résoudre par des techniques classiques nécessitant un temps de calcul raisonnable. Dans la littérature, il existe plusieurs méthodes exactes (méthode du gradient, méthodes à base du lagrangien, programmation linéaire, etc) susceptibles de résoudre une certaine classe de problèmes en un temps polynomial. Cependant, de telles méthodes exigent généralement plusieurs propriétés telles que la convexité, la continuité et la dérivabilité des fonctions objectifs qui sont rarement vérifiées en pratique. Ainsi, dans les années 1980, une autre classe d'algorithmes destinée à résoudre de tels problèmes désormais classés "difficiles" a inondé le monde industriel : les métaheuristiques. Les métaheuristiques permettent une recherche globale et approchée dans un espace de recherche des solutions. Contrairement à un simple algorithme de gradient, elles sont capables d'explorer tout l'espace de recherche. Ceci permet de ne pas rester bloqué dans un optimum local.

Ce chapitre passe en revue les métaheuristiques les plus utilisées. Dans un premier temps, nous définissons brièvement les problèmes difficiles comprenant des variables discrètes et/ou continues. Dans un deuxième temps, un arsenal de métaheuristiques à base d'un seul agent (recherche tabou, recuit simulé) ou de plusieurs agents (les algorithmes évolutionnaires AEs, l'optimisation par essaim de particules OEP et les algorithmes de colonies de fourmis CAO) est présenté. Nous détaillons plus particulièrement l'optimisation par essaim de particules (principale métaheuristique mise en oeuvre dans cette thèse).

A la suite, nous présentons les principales techniques utilisées pour décrire les systèmes complexes (graphes). Le cadre applicatif de ce travail réside en effet dans la caractérisation des réseaux électriques au niveau topologique (que nous traiterons dans le chapitre IV).

Enfin, nous consacrons quelques lignes aux dispositifs FACTS qui sont des appareillages spécifiques des réseaux électriques. L'application des métaheuristiques leur est consacrée.

1.2. Méthodes d'optimisation

L'intérêt de l'optimisation est de répondre aux problèmes plus ou moins complexes que l'on rencontre dans la vie afin d'obtenir une amélioration, tels que par exemple, le problème du voyageur de commerce à la recherche du chemin le plus court, le problème de rangement (de placement) afin de trouver la meilleure disposition ou encore dans un processus industriel, afin d'optimiser le travail de chaque employé. On définit une fonction objectif f (fonction de coût ou fonction profit), que l'on cherche à optimiser (minimiser ou maximiser) par rapport à tous les " paramètres " concernés et un certain ensemble de contraintes $\overrightarrow{g_{i\in N}}$ dans un espace de recherche fini E. Dans certains cas où la fonction objectif f et/ou l'ensemble des contraintes $\overrightarrow{g_{i\in N}}$ présentent certaines caractéristiques telles que la convexité, la continuité et la dérivabilité, il existe dans la littérature un large éventail de méthodes déterministes (ou exactes) permettant de résoudre ces problèmes en un temps " raisonnable " (généralement polynomial). A titre indicatif, nous citons parmi ces méthodes : les méthodes de programmation linéaire [schr98], quadratique [Noce00] et/ou dynamique [Bert95], la méthode de Newton [Noce00], la méthode du simplex [Neld65] ou encore la méthode du gradient [Avri12].

Cependant, en pratique, il est généralement très rare que le problème étudié présente de telles caractéristiques (absence de convexité, de continuité ou de dérivabilité, présence de bruit, etc). Cette classe de problèmes est appelée " problèmes NP-difficiles " (ie. il est au moins aussi difficile que tout problème de NP - possibilité de vérifier une solution en un temps polynomial -) ou le cas échéant, " problèmes difficiles ". Dans la littérature, deux sortes de problèmes d'optimisation reçoivent cette appellation :

1. certains problèmes d'optimisation combinatoire, qui prennent comme solutions des valeurs discrètes ou binaires, pour lesquels on ne connaît pas d'algorithme exact "rapide" (en temps polynomial). En effet, dans ce type de problèmes, l'espace de recherche croît exponentiellement avec le nombre de variables à déterminer. Ce type de problème appartient à la classe d'algorithmes NP-complets (ie. non seulement il est au moins aussi difficile que tout problème de NP, mais il est en plus dans NP, par exemple, le problème du voyageur de commerce).

2. certains problèmes d'optimisation à variables continues, pour lesquels on ne connaît pas d'algorithmes permettant de repérer un optimum global à coup sûr, en un nombre fini de calculs (en temps polynomial). A titre d'exemple, citons le problème d'identification dans les circuits RLC où l'on a besoin de faire coller des modèles avec des circuits réels. La fonction objectif correspond ici à l'erreur entre le modèle et le circuit réel. Les paramètres X_i sont les valeurs des composants, la fonction objectif devant tendre vers 0. Clairement, la fonction objectif n'est accessible que par simulation (présence de bruit).

En pratique, l'objectif n'est pas d'obtenir un optimum absolu, mais seulement une bonne solution et la garantie de l'inexistence d'une solution sensiblement meilleure. Pour atteindre cet objectif en un temps de calcul raisonnable, on peut faire appel à des méthodes appelées "heuristiques " et "métaheuristiques

Algorithme 1.1 Recuit simulé

1) **Déterminer** une configuration aléatoire s
2) **Définir** une notion de voisinage (mécanisme de perturbation d'une configuration)
3) **Initialiser** la température T.
4) **Tant que** le système n'est pas figé ou un critère d'arrêt n'est pas atteint faire
5) **tant que** l'équilibre n'est pas atteint **faire**
6) **Tirer** une nouvelle configuration s' (voisin de s)
7) **Appliquer** la règle de Metropolis (équation (1.2.1))
8) **Poser** avec une probabilité p (équation (1.2.1)) $s \leftarrow s'$
9) **si** alors
10) **Mise à jour** de $s* = s'$
11) **Mise à jour de** $f* = f(s')$
12) **fin si**
13) **fin tant que**
14) **Décroître** la température T
15) **fin tant que**

", qui s'appuient souvent sur une composante stochastique.

1.2.1. Recherches locales

Les méthodes de descente itératives [Hoos04] sont des méthodes d'optimisation locales, dont l'algorithme consiste à choisir un point de départ au hasard. On perturbe légèrement cette solution, ce qui donne une solution voisine. On compare avec la solution de départ. Si la solution est meilleure, on perturbe à nouveau la nouvelle solution. Sinon on revient à la meilleure valeur précédente (généralement, on accepte une solution dégradante suivant la méthode qu'on utilise) et on perturbe à nouveau la solution, jusqu'à atteindre un minimum local. Cette solution peut être améliorée en exécutant plusieurs fois la méthode en partant de solutions initiales différentes. On retiendra le meilleur minimum local obtenu. Dans cette section, nous présentons brièvement quelques recherches locales les plus utilisées dans la littérature.

1.2.1.1. Le recuit simulé

Le recuit simulé est une métaheuristique inspirée d'un processus utilisé en métallurgie. C'est une méthode stochastique qui fait appel à des valeurs aléatoires afin d'éviter d'étudier toutes les possibilités pour trouver le minimum de la fonction objectif (le chemin le plus court, exemple du voyageur de commerce ...). Initialement proposé parallèlement par Kirkpatrick [Kirk83] et par Cerny [Vcer85], l'algorithme du recuit simulé s'appuie sur l'algorithme de Metropolis [Metr53, Hast70] décrit par l'Algorithme (1.1). Le choix des paramètres (essentiellement empirique) pour mettre en œuvre cette méthode est très important et dépend de la nature du problème.

L'algorithme (1.1) représente les principales étapes du recuit simulé sur lequel nous pouvons voir les quatre principaux paramètres de contrôle : la valeur initiale de la température, la fonction de décroissance de la température, l'équilibre thermodynamique et le critère d'arrêt.P

Choix de la température initiale : Pour cela, nous observons la variation moyenne de la fonction objectif. A partir d'une valeur X_0, on génère une valeur aléatoire X'_0 avec une loi uniforme. Puis on calcule . Nous répétons plusieurs fois la même procédure, afin de calculer la variation moyenne de la fonction.

A partir de la probabilité d'acceptation d'une solution dégradante ($|\triangle f| > 0$, équation (1.2.1)) (règle de Métropolis) qui permet d'éviter le blocage dans un minimum local, nous pouvons déterminer la valeur

initiale de T . La valeur initiale T_{init} doit permettre d'accepter une grande proportion p de solutions dégradantes de f. Pour une température initiale " moyenne ", la valeur de p est de $0,5$. Donc, on peut déduire la température initiale avec l'équation (1.2.2).

$$p = e^{-\frac{|\triangle f|}{T}} \tag{1.2.1}$$

$$T_{init} = \frac{-\langle|\triangle f|\rangle}{ln(\langle p\rangle)} \tag{1.2.2}$$

La fonction décroissance de la température : Le rôle de la température est très important. En effet une brutale chute de la température risque de piéger l'algorithme dans un minimum local (plus T diminue rapidement, plus la probabilité d'acceptation tend rapidement vers 0). A contrario, une baisse très lente de la température permet d'éviter le problème et ainsi d'atteindre un minimum global. On utilise le plus souvent une décroissance géométrique comme dans l'équationPl1.2.3) :

$$T_{N+1} = \alpha T \tag{1.2.3}$$

avec $0 < \alpha < 1$

"L'équilibre thermodynamique" : "l'équilibre thermodynamique" est le moment où l'on choisit de changer de palier ou non. Pour cela on compte le nombre de perturbations acceptées et le nombre de perturbations tentées. L'équilibre thermodynamique est alors atteint si le nombre de perturbations acceptées est égal à $12 \times D$ ou si le nombre de perturbations tentées est égal à $100 \times D$ (D étant la dimension du problème).

Le critère d'arrêt : Le critère d'arrêt peut être l'absence d'amélioration au bout de 3 changements de palier de température par exemple, le nombre d'évaluations, ...

Le recuit simulé possède l'avantage de la souplesse quant à son implémentation. Toutefois, son principal inconvénient est le nombre de paramètres qui rend les réglages de l'algorithme assez compliqués et les temps de calcul pouvent devenir très importants. Néanmoins, on trouve des travaux [Cour94] qui s'attachent à régler de manière optimale les paramètres de l'algorithme. Par ailleurs, pour surmonter le problème du temps de calcul, plusieurs méthodes de parallélisation des calculs ont été introduites [Azen92].

1.2.1.2. La recherche tabou

Au cours de ces 25 dernières années, un certain nombre d'études s'appuyant sur l'application de la recherche tabou, une heuristique proposée par F. Glover [Glov86], à différents problèmes combinatoires ont inondé les revues du domaine d'optimisation et de l'ingénierie ([Glov97, Ribe02, Voss99, Osma96, Glov93]). Dans la majorité des cas étudiés, les auteurs mettent en exergue la qualité des solutions obtenues par la recherche tabou, qui sont généralement proches de l'optimum global. Ce succès face à la résolution de divers problèmes combinatoires a fait la réputation de cette méthode. Comme dans le recuit simulé (comme dans toute recherche locale, d'ailleurs), la recherche tabou fonctionne avec une seule configuration courante, qui est actualisée au cours des itérations en explorant N de ses configurations voisines.

La principale originalité de cette méthode réside dans l'utilisation d'une liste mémoire, appelée "liste tabou " et ce, afin d'éviter des mouvements cycliques et donc d'éviter le risque de retour à une configuration et/ou une solution déjà visitée. En d'autres termes, l'introduction d'une liste tabou permet des

Algorithme 1.2 Recherche tabou.

1) **Déterminer** une configuration aléatoire s
2) **Définir une notion de voisinage** (mécanisme de perturbation d'une configuration)
3) **Initialiser** la liste tabou T.
4) **tant qu**'un critère d'arrêt n'est pas satisfait **faire**
5) **Perturbation** de s suivant N mouvements $(t \to s) \notin T$
6) **Évaluation** des N voisins
7) **Sélection** du meilleur voisin $s*$
8) **Insertion** du mouvement $s* \to s$ $(T = T \cup (s* \to s))$
9) **Mise à jour** $s = s*$
10) **fin tant que**

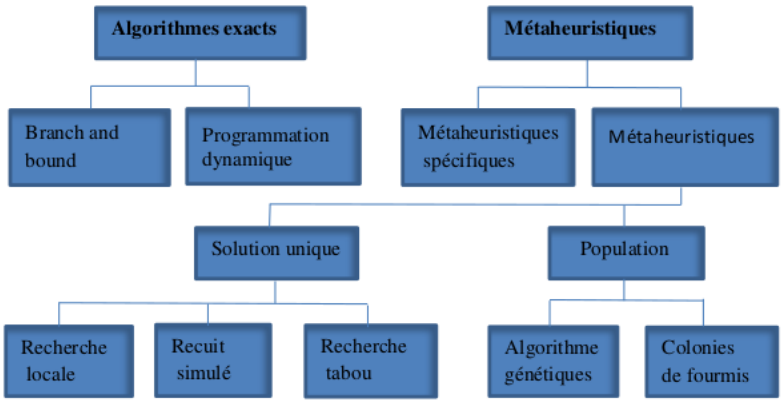


FIGURE 1.2.1.: Classification des métaheuristiques[Elgh01].

mouvements (qui ne sont pas dans la liste "tabou") qui n'améliorent pas la solution courante et donc de continuer la recherche de solutions même lorsqu'un optimum local est rencontré. Cette liste contient un certain nombre de mouvements $(t \to s)$ "temporairement interdits " qui sont, généralement, l'inverse des derniers mouvements $(s \to t)$. Le principe de fonctionnement de cette méthode est décrit dans l'algorithme (1.2).

1.2.2. Autres métaheuristiques

L'optimisation itérative est l'une des plus anciennes méthodes en optimisation. De nombreuses stratégies sont envisageables, en particulier celles inspirées de la nature.

Il n'est donc pas étonnant que, implicitement ou explicitement, plusieurs modèles mathématiques d'optimisation s'inspirent de comportements biologiques (les algorithmes génétiques) et/ou éthologiques (les algorithmes de colonies de fourmis ou l'optimisation par essaim de particules) qui usent abondamment de métaphores et de termes issus de la génétique et la sociologie. Parmi ces modèles, nous pouvons distinguer ceux correspondant à un comportement individuel (un seul agent) et ceux utilisant un comportement collectif (plusieurs agents), figure 1.2.1.

Bien que le concept même des métaheuristiques ait été introduit dans les années 1960 (stratégies d'évolution [Rech65], programmation évolutionnaire [Foge66] et algorithmes génétiques [Holl75]), ce terme métaheuristique n'a pris sens qu'à partir de l'année 1986 avec l'apparition de la toute première métaheuristique, "la recherche tabou" introduite par F. Glover [Glov86]. Une des caractéristiques les plus remarquables des métaheuristiques est leur souplesse et capacité d'adaptation à n'importe quel problème

d'optimisation difficile ou non. En d'autres termes, les métaheuristiques se prêtent naturellement à des extensions. Nous pouvons citer :

- Les métaheuristiques pour l'optimisation multi-objectif [Coll02], où il faut optimiser plusieurs objectifs contradictoires. Le but ne consiste pas ici à trouver un optimum global, mais à trouver un ensemble d'optima appelé " Front de Pareto ", qui forment une surface de compromis pour les différents objectifs du problème.
- Les métaheuristiques pour l'optimisation multimodale [Gold87], où l'on ne cherche plus l'optimum global, mais l'ensemble des meilleurs optima globaux et/ou locaux.
- Les métaheuristiques pour l'optimisation dynamique [Bran02], où la fonction objectif varie dans le temps, ce qui nécessite d'approcher l'optimum à chaque pas de temps.

En outre, cette classe d'algorithmes présente l'avantage d'être intrinsèquement parallélisable. Dans cette partie de chapitre, nous allons nous attacher à expliquer le principe de quelques méthodes d'optimisation qui font partie de la famille des " métaheuristiques ".

1.2.3. Les algorithmes évolutionnaires (AEs)

Introduits dans les années 1950 par A.S. Fraser [Fra57], les algorithmes évolutionnaires (AEs) sont des techniques d'optimisation inspirées principalement par l'évolution biologique des espèces, "la théorie Darwinienne". En effet, ces méthodes d'optimisation sont basées sur les mécanismes de la sélection naturelle où les individus les plus adaptés à leur milieu (héritant de bons gènes de leurs ancêtres) ont tendance à s'affirmer et se reproduire tandis que les individus ayants du mal à s'adapter ont, quant à eux, tendance à s'affaiblir pour disparaître au fil du temps. Les approches évolutionnaires s'appuient, généralement, sur un modèle commun présenté dans l'algorithme (1.3). Au cours des années 1970, la puissance de calcul des ordinateurs a connu un essor considérable et a permis la réalisation de nombreuses approches d'optimisation héritant du concept des algorithmes évolutionnaires. A titre d'exemple, nous pouvons citer :

- Les stratégies d'évolution [Rech65, Beye01], conçues pour résoudre des problèmes à variables continues. Elles sont axées sur la modélisation des paramètres stratégiques qui contrôlent la variation dans l'évolution, autrement dit " l'évolution de l'évolution " ;
- La programmation évolutionnaire [Foge62, Foge66], visant à faire évoluer les structures d'automates à états finis par des successions de croisements et de mutations ;
- Les algorithmes génétiques [Holl75], conçus pour résoudre des problèmes d'optimisation à variables discrètes, en modélisant l'évolution génétique ;
- La programmation génétique [Koza89, Koza90], basée sur les algorithmes génétiques, mais où les individus (ou chromosomes) sont des programmes informatiques, représentés en utilisant une structure d'arbre ;
- L'évolution différentielle [Stor97, Pric05], conçue pour résoudre des problèmes à variables continues. Sa stratégie consiste à biaiser un opérateur de mutation, appliqué à un individu, en fonction des différences calculées avec d'autres individus sélectionnés aléatoirement.

La solution optimale est cherchée à partir d'une population G_n (typiquement 50 à 100) de solutions en utilisant des processus stochastiques. La recherche de la meilleure solution est faite par la production d'une nouvelle génération de population G_{n+1} à partir des anciennes, en appliquant (avec une certaine probabilité) des processus de sélection, de croisement et de mutation sur chaque individu de la génération G_n. L'opération est répétée jusqu'à ce qu'un critère d'arrêt soit atteint. Les individus de la génération

Algorithme 1.3 Modèle commun des AEs

1) **Initialisation** des N individus de la population G_0
2) **Évaluation** des N individus de G_0
3) **tant qu**'un critère d'arrêt n'est pas atteint **faire**
4) **Sélection** de s ($s < N$) individus en vue de la phase de reproduction
5) **Croisement** des s individus sélectionnés
6) **Mutation** de m ($m < s$) des individus croisés (enfants)
7) **Évaluation** des m enfants obtenus
8) **Sélection** de N individus pour représenter la génération G_{n+1}
9) **fin tant que**

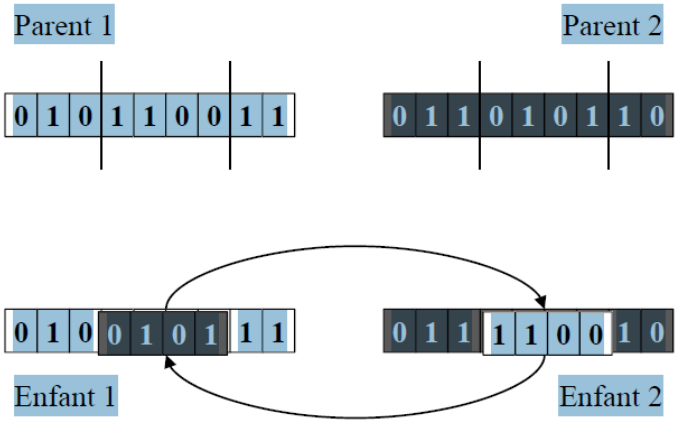


FIGURE 1.2.2.: Exemple d'opérateur de croisement en représentation binaire.

G_n ayant subi ces opérateurs génétiques sont appelés " Parents " et les individus ainsi engendrés à la génération G_{n+1} sont appelés " Enfants ". La population d'individus d'un AE est caractérisée à partir d'un codage (une chaîne, code génétique). En conséquence, le choix des codages qui va être utilisé est très important et doit être adapté au problème afin de limiter la taille de l'espace de recherche (en produisant des solutions valides, le plus souvent possible) lors de l'application des opérateurs de recherche. Il existe plusieurs codages comme le codage binaire qui est souvent utilisé. Néanmoins on peut utiliser le codage réel qui est une alternative du codage binaire.

Un algorithme évolutionnaire classique dispose donc de trois opérateurs principaux :

1. *Un opérateur de sélection* : Comme dans le règne animal, les meilleurs ont le plus de probabilité d'avoir une descendance. En effet, ici les meilleures solutions auront une plus grande probabilité d'être sélectionnées pour produire la génération suivante, ce qui va améliorer les solutions futures. Plusieurs approches existent pour effectuer une sélection telles que la sélection par " roulette biaisée

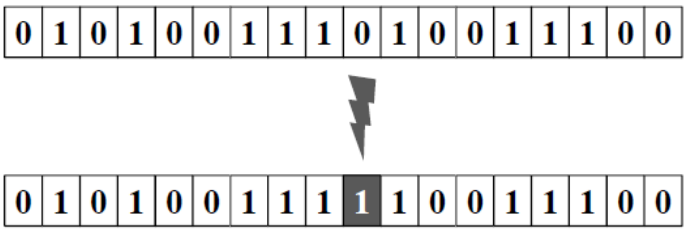


FIGURE 1.2.3.: Exemple d'opérateur de mutation en représentation binaire.

" qui consiste à donner à chaque individu une probabilité d'être sélectionné proportionnellement à sa performance, la sélection par " rang " qui opère comme dans la roulette biaisée dont les secteurs sont proportionnels aux rangs des individus et enfin, la sélection par " tournoi ", qui consiste à tirer, généralement au hasard, des paires d'individus, et choisir les meilleurs de chaque paire (on peut aussi affecter une probabilité élevée pour les meilleurs individus de chaque paire et choisir un suivant cette distribution de probabilité).

2. *Un opérateur de croisement* : Les croisements se font par paires, les parents sélectionnés s'échangent une partie du code d'une façon aléatoire pour générer deux enfants, qui héritent des gènes de leurs parents. Il existe plusieurs méthodes de croisement qui ont été développées telles que le " croisement simple " qui consiste à tirer une position au hasard et à échanger les caractéristiques des deux individus à partir de ce point, le " double croisement " pour lequel l'échange a lieu entre deux positions tirées aléatoirement (figure 1.2.2) et enfin, le " croisement uniforme ", qui introduit un masque croisement généré de manière aléatoire. Généralement, l'opérateur de croisement est appliqué avec une probabilité élevée, typiquement de l'ordre de 80%.
3. *Un opérateur de mutation* : Cela consiste à changer de manière aléatoire, une ou plusieurs valeurs au sein de la chaîne, provoquant soit une amélioration soit une dégradation de la qualité de l'individu (figure 1.2.3). On maintient par ce biais la diversité des solutions. Généralement, l'opérateur de mutation est appliqué avec une faible probabilité, typiquement de l'ordre de 1%. Par contre, ce taux peut augmenter au cours du temps car la diversité de la population diminue.

1.2.4. Optimisation par Essaim Particulaire (OEP)

L'OEP, Optimisation par Essaim Particulaire, est une méthode récente d'optimisation faisant appel à une population d'agents, appelés ici " particules ". Celles-ci se distinguent par leur efficacité collective (collaboration plutôt que compétition, "l'union fait la force "). Cette technique s'appuie sur la technique des algorithmes culturels adaptatifs de R. Reynolds [Reyn94] et a été proposée pour la première fois par R. Eberhart et J. Kennedy en 1995 [Kenn95]. Ces auteurs se sont principalement inspirés des modèles de comportement d'organismes sociaux vivant et agissant réciproquement dans de grands groupes (nuée d'oiseaux, d'abeilles, de poissons ... etc.). En effet, dans certaines espèces animales, par exemple un banc de poisson, des comportements dynamiques (déplacements) relativement complexes peuvent être observés, alors que les individus eux-mêmes n'ont accès qu'à des informations limitées.

Cet algorithme appartient à la famille des algorithmes évolutionnaires qui s'inspirent de la théorie de l'évolution pour résoudre des problèmes divers. Ils font ainsi évoluer un ensemble de solutions à un problème donné, dans l'optique de trouver les meilleurs résultats. Ce sont des algorithmes de nature stochastique, car ils utilisent itérativement des processus aléatoires. L'algorithme OEP peut être classé dans la famille des algorithmes mémétiques car il mime le comportement des animaux dits sociaux. Des scientifiques ont cherché à comprendre le phénomène de déplacement en banc chez les poissons, des nuées d'oiseaux dans le ciel, i.e. comprendre les comportements "sociaux " des animaux. Ils ont aussi tenté de simuler, via des algorithmes, le comportement de ces derniers. Le but de l'algorithme OEP est donc de simuler le comportement de groupes animaliers en déplacements. Le principe de base est simple : faire voler un essaim de particules dans un hyperespace représentant l'espace des solutions en s'inspirant du vol d'un essaim d'oiseaux par exemple, mais en donnant aux individus un but par l'intermédiaire d'une fonction de fitness $\overrightarrow{J}$. Le partage d'informations entre individus au sein d'une espèce peut donc offrir un avantage évolutif, tout comme la transmission de gênes en génétique.

Soit, par exemple, le scénario suivant : un groupe d'oiseaux est en train de chercher de la nourriture

dans un champ. Il y a seulement une partie du champ qui contient de la nourriture. Tous les oiseaux ne savent pas où est la nourriture. Mais ils savent à quelle distance est la nourriture à chaque itération. Quelle est donc la meilleure stratégie pour trouver la nourriture ? Une stratégie efficace est de suivre l'oiseau qui est le plus près de la nourriture. Ainsi, d'une façon générale, lorsqu'un prédateur est repéré à proximité d'un groupe (banc de poissons, nuée d'oiseaux, ...), l'individu le plus proche d'un prédateur (ou nourriture,...) donne l'alerte aux autres en changeant son comportement et ses mouvements par rapport au reste du groupe. C'est l'effet "sentinelle" : l'information passe de proche en proche et les individus ainsi alertés vont devenir plus coopératifs afin de se défendre contre les prédateurs. OEP s'appuie sur un scénario de ce type, scénario de l "union fait la force " afin de résoudre des problèmes d'optimisation.

A cette fin, il est nécessaire de générer initialement et aléatoirement une population d'individus, représentant en fait les solutions potentielles du problème. Puis dans l'optique de trouver la solution optimale du problème posé dans l'espace de recherche considéré, ces individus sont manipulés, à travers plusieurs itérations en temps. Ainsi, à chaque itération, il y a d'abord une mise à jour de la position et de la vitesse (vélocité) de chaque particule. Pour cela, chacune des particules a une mémoire et se souvient de la position de sa meilleure *fitness* (mémoire individuelle). Chaque particule est aussi mise au courant de la position de la particule qui a la meilleure *fitness* parmi toutes les particules avec lesquelles elle est en contact (particules informatrices) (mémoire globale). La nouvelle vitesse (à l'itération $k+1$) de la particule est ensuite déterminée sous forme d'une combinaison linéaire calculée à partir de grandeurs définies à l'itération k, soit :

- la vitesse et la position de la particule
- la meilleure position mémorisée de la particule
- la meilleure position globale mémorisée associée à la particule

Puis, la nouvelle position (à l'itération $k+1$) de la particule est actualisée à partir de la vitesse à l'itération $k+1$.

L'OEP a été initialement mis en œuvre pour des problèmes où l'espace de recherche est continu [Cler05]. Ses performances dans ce contexte ont été démontrées dans de nombreux cas pratiques. Dans le contexte de problèmes d'optimisation dans des espaces de recherche discrets (problèmes d'optimisation combinatoire), plusieurs algorithmes d'OEP discrets ont été proposés [Corr06, Kenn97, Garc08, Pugh06]. Chacun de ces algorithmes s'applique à traiter une classe de problèmes plus particulière. Ainsi, dans le cas où la variable discrète ne peut prendre que 2 valeurs (0 et 1), l'OEP discret binaire (Binary PSO) [Kenn97] présente des performances intéressantes, comme par exemple, dans le cas d'une application de type ilotage optimal d'un réseau électrique [Wenx07].

Cependant, le problème du placement optimal de FACTS (chapitre III) où l'espace de recherche, représenté par l'ensemble des nœuds du réseau, est un espace discret de dimension N, n'entre pas dans cette classe de problèmes. Parmi les quelques algorithmes d'OEP discret adaptés à un espace de recherche discret de dimension N quelconque, la méthode " *Velocity Likehood* " [Corr06] présente l'intérêt de pouvoir travailler avec des particules de tailles différentes. Elle semble donc, a priori, bien adaptée à l'introduction du nombre de FACTS à placer comme paramètre d'optimisation (en associant la taille de la particule avec le nombre de FACTS). De plus, cette approche remplace la notion de vitesse de la particule, dont le sens physique dans un espace de recherche combinatoire n'est pas immédiat, par une notion de "probabilité ou poids de choix" (par maximum de vraisemblance) associée à chacune des positions possibles pour la particule. C'est la raison pour laquelle elle a été retenue dans le cadre du travail présenté dans ce rapport pour traiter du cas du placement optimal de FACTS. Le dimensionnement optimal de FACTS, qui est un problème à espace de recherche continu, est traité par une approche classique d'OEP continu.

Bien entendu, placement optimal et dimensionnement optimal ne peuvent pas être totalement découplés. La méthode mise en œuvre devra donc être adaptée pour tenir compte du caractère hybride (i.e. continu/discret) du problème posé. Dans la suite de cette section (1.2.4), les méthodes d'OEP continu et d'OEP discret "Velocity Likehood " sont brièvement introduites. Puis, une première approche d'"hybridation" de ces deux méthodes est proposée.

1.2.4.1. Formalisation de l'algorithme OEP basique

Étant donné une population de N individus, on appellera désormais les individus *"particules",* chacune d'elles étant une solution potentielle. Par exemple, considérons une nuée d'oiseaux à la recherche d'une source d'eau dans un champ (ce que l'on appellera désormais l'*espace de recherche*).

Les particules (oiseaux) se déplacent dans l'espace de recherche de la manière suivante : initialement positionnée aléatoirement (en désignant la position initiale de la particule i par $\overrightarrow{x_i}(0)$), chaque particule connaît sa distance par rapport à la source d'eau (ce que l'on appellera désormais la *fitness*) et possède une vitesse initiale $\overrightarrow{v_i}(0)$ (souvent appelée *vélocité* et initialisée aléatoirement). La socio-psychologie suggère que les mouvements des individus (dans un diagramme socio-cognitif) soient sous l'influence de leurs derniers comportements (positions) et ceux de leurs voisins (peut-être placés dans un réseau social, et non nécessairement dans l'espace). De là, la mise à jour de la position de chaque particule dépend de la direction de son mouvement à l'instant t, i.e. sa vitesse ou vélocité $\overrightarrow{v_i}(t)$, sa meilleure position précédente $\overrightarrow{p_i}$ et de celle de la meilleure position de toutes ses voisines $\overrightarrow{p_g}$ (on les appellera désormais *informatrices*), figure 1.2.4.

Le déplacement de chaque particule i est donc effectué grâce aux équations (1.2.4) et (1.2.5) suivantes :

$$\overrightarrow{v_i}(t+1) = \omega\overrightarrow{v_i}(t) + \varphi_1(\overrightarrow{p_i} - \overrightarrow{x_i}(t)) + \varphi_2(\overrightarrow{p_g} - \overrightarrow{x_i}(t)) \tag{1.2.4}$$

$$\overrightarrow{x_i}(t+1) = \overrightarrow{x_i}(t) + \overrightarrow{v_i}(t+1) \tag{1.2.5}$$

avec :

- $\overrightarrow{v_i}(t)$ et $\overrightarrow{v_i}(t+1)$, respectivement la vitesse de la particule i aux instants t et $t+1$,
- $\overrightarrow{x_i}(t)$ et $\overrightarrow{x_i}(t+1)$, respectivement la position de la particule i aux instants t et $t+1$,
- ω, l'inertie de la particule i généralement décroissant linéairement de ω_{max} à ω_{min},
- $\overrightarrow{p_i}$ et $\overrightarrow{p_g}$, respectivement la meilleure position précédente et celle de toutes les informatrices de la particule i.

Les paramètres φ_1 et φ_2, appelés *paramètres de confiance,* sont générés aléatoirement suivant une loi uniforme entre 0 et φ_{max} ($U_{[0,\varphi_{max}]}$) et influent sur la vélocité $\overrightarrow{v_i}(t+1)$ de manière à trouver le juste milieu entre les rôles relatifs de l'expérience individuelle (i.e. la meilleure position précédente $\overrightarrow{p_i}$ qui est dirigée par φ_1) et de la communication sociale (i.e. la meilleure position de toutes les informatrices $\overrightarrow{p_g}$ qui est dirigée par φ_2). La sélection aléatoire uniforme de ces deux paramètres est justifiée par le fait qu'il n'y a pas de classification des deux sources d'information, ce qui permet une bonne exploration de l'espace de recherche et surtout, d'éviter une convergence prématurée de l'algorithme.

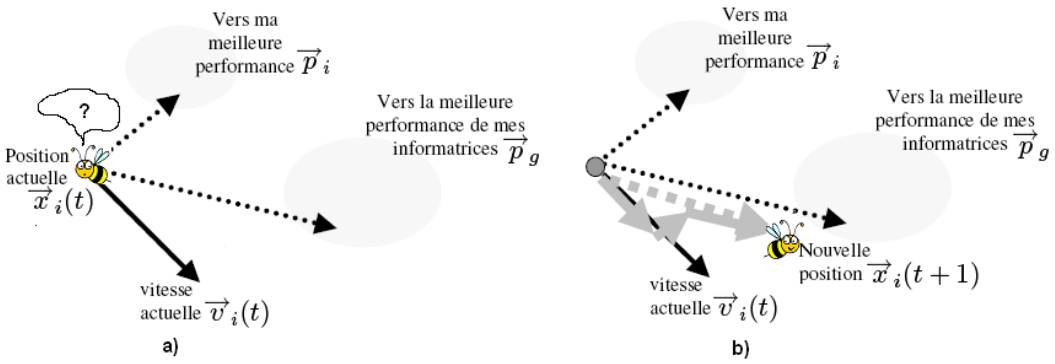


FIGURE 1.2.4.: Principe de déplacement dans l'algorithme OEP ***a)*** avant le déplacement ***b)*** après le déplacement.

L'algorithme emploie aussi un autre paramètre, V_{max}, qui permet de limiter la rapidité du mouvement pour empêcher une "explosion" du système, dans le cas où il y aurait des oscillations de grandes amplitudes. L'algorithme pourrait mettre en œuvre un compromis efficace entre " l'intensification " et " la diversification ". Le seul problème surgit lorsque les points p_i et p_g s'écartent. Dans ce cas, les particules continueront à osciller entre ces deux points sans convergence. Une caractéristique intéressante de cet algorithme est que, lorsqu'on découvre un nouvel optimum (après une phase d'intensification), les particules exploreront l'espace de recherche autour du nouveau point (c'est-à-dire effectueront une phase de diversification).

1.2.4.2. Taille de l'essaim de particule

Le nombre de particules alloué à la résolution du problème dépend essentiellement de deux paramètres, à savoir :

- la taille de l'espace de recherche,
- le rapport entre les capacités de calcul de la machine et le temps maximum de recherche.

Il n'y a pas de règle pour déterminer ce paramètre dans le cas d'un problème d'optimisation continu. Néanmoins, des résultats théoriques ont pu être obtenus concernant la taille minimale de l'essaim de particules dans le cas des problèmes discrets. Faire de nombreux essais permet de se doter de l'expérience nécessaire à l'appréhension de ce paramètre.

Dans la suite de notre travail, nous utiliserons l'équation (1.2.6) [Cler04] pour déterminer la taille minimale de l'essaim de particules, car le problème que nous allons décrire dans le chapitre III (placement et dimensionnement de FACTS) est un problème hybride (présence de variables continues et de variables discrètes).

$$S_{min} = INT(9.5 + 0.124(D - 9)) \tag{1.2.6}$$

Avec :
- S_{min} est la taille minimale de l'essaim de particule,
- $INT(x)$ est la partie entière du nombre réel x,
- D est le nombre de bits des variables binaires.

La preuve détaillée de ce résultat est donnée dans [Cler04, Cler05, Cler07].

1.2.4.3. Le voisinage (informatrices)

Dans la littérature, on trouve trois grandes topologies de voisinage :

1. Topologie en étoile : dans ce type de topologie, chaque particule est reliée à toutes les autres. Par exemple, relier les particules à l'optimum global (l'optimum du voisinage est l'optimum global), relier les particules à d'autres prises aléatoirement (informatrices aléatoires par exemple)...
2. Topologie en anneau : chaque particule est reliée à n particules (en général, n = 3). C'est cette topologie, la plus courante, que nous avons considérée dans le chapitre III.
3. Topologie en rayon : les particules ne communiquent qu'avec une seule particule centrale (prendre par exemple l'optimum Global $\overrightarrow{p_g}$ ou une particule aléatoirement dans l'espace de recherche,... , comme informatrice).

La figure 1.2.5 illustre chaque cas de ces topologies.

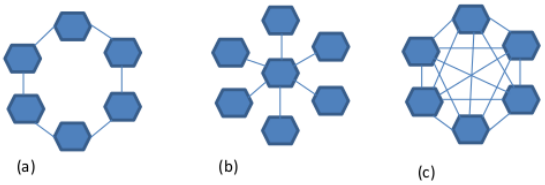


FIGURE 1.2.5.: Principales topologies du voisinage. a) Anneau (avec n=2), b) Rayon, c) Étoile.

A partir des trois topologies, on peut obtenir d'autres types, en les combinant. La notion de voisinage dépend de son sens pris au départ. Il peut être "social" (aléatoirement par exemple) ou "spatial" (i.e., en termes de distance). Il semble être communément admis qu'un voisinage social (un individu x_2 ayant par exemple pour voisins, les individus x_1 et x_3, quelles que soient les localisations spatiales de x_1, x_2, x_3) donne généralement de meilleurs résultats qu'un voisinage spatial (fonction de la proximité des individus dans l'espace de recherche, par exemple). Toutefois, en pratique, il est préférable de tester les deux notions de voisinage.

Dans le cas d'un voisinage spatial, il est nécessaire de définir une métrique (une distance). Un exemple de distance spatiale est celui basé sur une distance euclidienne (utilisé par R. Kennedy et J. Eberhart [Eber01]). Un autre exemple de distance spatiale est celle utilisée par Suganthan [Yuhu98] ; une particule est voisine d'une particule P_a si :

$$\frac{\|x_a - x_b\|}{d_{max}} \leqslant \varepsilon \tag{1.2.7}$$

avec d_{max} la plus grande distance entre deux particules et

$$\varepsilon = \frac{0.3t + 0.6t_{max}}{t_{max}} \tag{1.2.8}$$

avec t l'itération courante et le nombre maximal d'itérations

Dans le type de voisinage utilisé dans ce travail (chapitre III), un réglage (selon les nécessités) est primordial. En effet, prendre un nombre petit de voisins conduit à une division en groupes de l'essaim de particules. Les avantages résident alors essentiellement dans la diversification de la recherche. Par

contre, celle-ci peut engendrer de moins bonnes solutions dans certains cas, en raison de la non solidarité (communication) entre particules.

1.2.4.4. Coefficients de confiance

Les coefficients de confiance, comme décrits dans le paragraphe (1.2.4.1), permettent à la particule de faire le choix entre suivre son chemin (position actuelle) ou suivre celui de son entourage immédiat (informatrices $\overrightarrow{p_g}$ et sa meilleure position $\overrightarrow{p_i}$). Ils sont définis à l'origine comme suit :

$$\begin{cases} \varphi_1 = r_1 c_1 \\ \varphi_2 = r_2 c_2 \end{cases} \tag{1.2.9}$$

avec r_1 et r_2, deux nombres aléatoires suivant une loi uniforme entre $[0,1]$ et c_1, c_2, des constantes positives déterminées de façon empirique.

1.2.4.5. Vitesse maximale

Comme indiqué dans le paragraphe (1.2.4.1), elle permet d'éviter un déplacement trop rapide des particules, qui conduit à un passage à côté de l'optimum. Cependant, fixer la vitesse maximale des particules n'est pas toujours nécessaire. En effet, en introduisant le *coefficient de constriction* k [Cler02] suivant, on peut resserrer l'espace de recherche :

$$k = \frac{2}{|2 - (c_1 + c_2) - \sqrt{(c_1 + c_2)^2 - 4(c_1 + c_2)}|} \quad avec : \; c_1 + c_2 > 4 \tag{1.2.10}$$

L'équation (1.2.4) de la vitesse devient donc :

$$\overrightarrow{v}_i(t+1) = k[\overrightarrow{v}_i(t) + (\overrightarrow{p}_i - \overrightarrow{x}_i(t)) + (\overrightarrow{p}_g - \overrightarrow{x}_i(t))] \tag{1.2.11}$$

Des études ont montré que l'utilisation de ce *coefficient de constriction* k améliorait le taux de convergence. Néanmoins, dans certains cas, il est préférable de fixer la vitesse maximale V_{max} en plus du *coefficient de constriction* k.

1.2.4.6. Facteur d'inertie

Le facteur d'inertie ω permet de contrôler la taille de l'espace de recherche exploré et donc la rapidité de convergence de l'algorithme. Une grande valeur de ω (> 1) permet de grandes amplitudes de mouvement avec une bonne exploration de l'espace de recherche, par opposition à de petites valeurs de ω (< 1) avec de faibles mouvements engendrés. Le facteur d'inertie ω est généralement fixé entre $[0.8, 1.2]$ (ou décroît linéairement entre deux bornes au cours des itérations). Au delà de 1.2, l'algorithme a du mal à converger [Berg02].

Remarque 1. Rappelons (paragraphe (1.2.4.1)) que les positions et les vitesses (vélocités) initiales des particules doivent être initialisées, en général de manière homogène.

Par ailleurs, dans certains cas, l'algorithme mis en œuvre ne converge pas systématiquement. Par conséquent, les critères d'arrêt suivants peuvent être utiles :

- nombre maximum d'itérations,
- variation de vitesse (vaut 0 si la particule concernée ne bouge plus),
- la solution proposée est satisfaisante.

Algorithme 1.4 Colonie de fourmis (ACO) pour le problème du voyageur de commerce

1) $t \leftarrow 1$
Pla**tant** qu'un critère d'arrêt n'est pas atteint **faire**
3) **pour** chaque fourmi $k = 1$ à m **faire**
4) **Choisir** une ville au hasard
5) **pour** chaque ville i non encore visitée **faire**
6) **Choisir** une ville j dans la liste J_i^k des villes restantes selon 1.2.12
7) **fin pour**
8) **Déposer** une piste $\triangle\tau_{ij}^k(t)$ sur le trajet $T^k(t)$ conformément à 1.2.13
9) **fin pour**
10) **Évaporer** les pistes selon 1.2.14
11) $t \leftarrow t + 1$
12)**fin tant que**

Remarque 2. Plusieurs variantes de l'optimisation par essaim de particules ont été proposées. Nous pouvons citer :

- les méthodes OEP coopératives telles que CPSO-S et CPSO-H [Berg04], qui consistent à diviser l'espace de recherche en sous-espaces, où chaque sous-espace est traité séparément (le cas de CPSO-S) ou à utiliser deux phases de recherche, dont la première consiste à utiliser CPSO-S, puis un algorithme de base de PSO (le cas de CPSO-H),
- les méthodes OEP utilisant plusieurs essaims, tels que CONPSO [Bask04], FDR-PSO, PSO-2S/DEPSO-2S [ElDo12a, ElDo12b].

1.2.5. Les algorithmes de colonies de fourmis (ACO)

Cette métaheuristique est inspirée des comportements collectifs de dépôt et de suivi de pistes observés dans les colonies de fourmis. Les fourmis communiquent entre elles indirectement à travers leur environnement et trouvent une solution en tenant compte de leurs expériences collectives. Cet algorithme a été proposé pour la première fois par Dorigo et al, au début des années 80 [Colo91, Dori96]. Le premier algorithme "Ant system" a été spécialement conçu pour résoudre le problème du voyageur de commerce.

Pour illustrer le principe de fonctionnement des algorithmes de colonies de fourmis (ACO), prenons l'exemple de la recherche du plus court chemin entre une fourmilière (nid) et une source de nourriture. Supposons qu'au départ toutes les fourmis suivent une ligne droite (plus court chemin, figure 1.2.6-1). Lors de l'apparition d'un obstacle entre le nid et la source de nourriture (figure 1.2.6-2), les fourmis se séparent en groupes, prenant ainsi des chemins différents, donc non optimaux. En se déplaçant, les fourmis déposent derrière elles des substances chimiques appelées *"phéromones"*. Grâce à une intelligence collective des fourmis, due à un type de communication appelé *"stigmergie"*, les fourmis les plus éloignées du chemin le plus court finissent par prendre ce dernier, laissant ainsi les phéromones déposées sur leur anciennes trajectoires s'évaporer au fil du temps. Par conséquent, le chemin le plus court finira par avoir la densité de phéromones la plus élevée. Plus l'intensité augmentera et plus les fourmis auront tendance à suivre cette trajectoire. Au bout d'un moment, le chemin le plus long n'étant plus utilisé disparaît en même temps que les phéromones déposées sur ce dernier. Finalement, toutes les fourmis prennent le plus court chemin entre la fourmilière et la source de nourriture (figure 1.2.6 3).

Plus formellement, la résolution du problème du voyageur de commerce par l'algorithme de colonies de fourmis peut se faire de la manière suivante : à chaque itération t, chaque fourmi k construit un trajet complet de N noeuds, avec N le nombre de villes à visiter. La trajectoire de chaque fourmi dépendra de

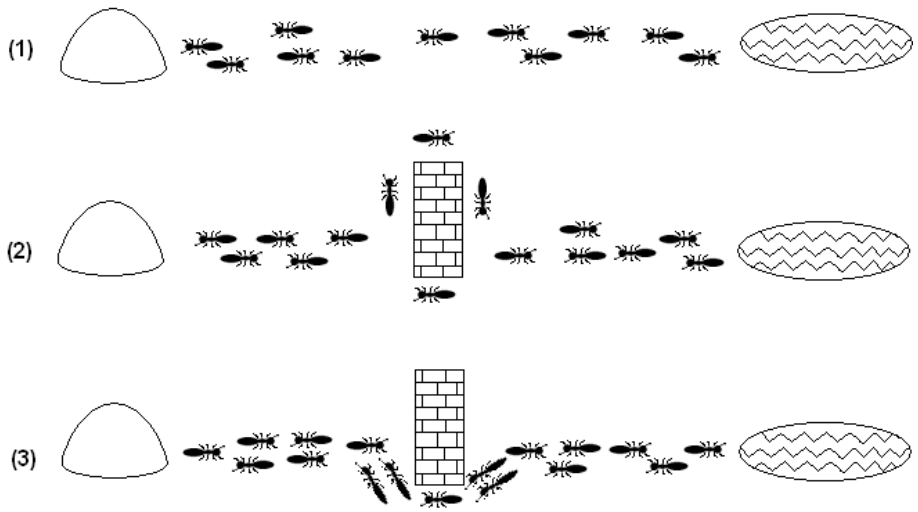


FIGURE 1.2.6.: Faculté d'une colonie de fourmis de retrouver le plus court chemin, obstrué fortuitement par un obstacle [Quin04].

trois points :

- Connaissance d'une liste des villes qu'elle peut visiter à partir de la ville i où elle est située : J_i^k,
- L'inverse de la distance entre les villes $\eta_{ij} = \frac{1}{d_{ij}}$ (visibilité), avec d_{ij} la distance entre la ville i et la ville j. Plus la distance d_{ij} est grande, plus la fourmi aura tendance à aller vers la ville la plus proche,
- Une mémoire collective, qui les informe du chemin le plus utilisé (l'histoire de l'intensité des phéromones) : τ_{ij}.

Le déplacement ce fait alors selon l'équation (1.2.12), appelée la probabilité de transition :

$$P_{ij}^k = \begin{cases} \frac{[\tau_{ij}(t)]^\alpha [\eta_{ij}]^\beta}{\sum_{l \in J_i^k} \left[[\tau_{il}(t)]^\alpha [\eta_{il}]^\beta \right]} & si\ j \in J_i^k \\ 0 & si\ j \notin J_i^k \end{cases} \tag{1.2.12}$$

en désignant par α et β, les paramètres contrôlant l'importance de l'intensité de la piste $\tau_{ij}(t)$, et la visibilité η_{ij} respectivement. Ces deux paramètres sont à régler "judicieusement", afin d'éviter des optima locaux ou la non-convergence de l'algorithme.

A la fin d'une tournée, la quantité de phéromone $\triangle\tau_{ij}^k(t)$ laissée par chaque fourmi dépend de la qualité de la solution trouvée (équation (1.2.13)).

$$\triangle\tau_{ij}^k(t) = \begin{cases} \frac{Q}{L^k(t)} & si\ (i,j) \in T^k(t) \\ 0 & si\ (i,j) \notin T^k(t) \end{cases} \tag{1.2.13}$$

en désignant par $T^k(t)$ et $L^k(t)$ respectivement, le trajet de la fourmi k à l'itération t et la longueur de la tournée. Q est un paramètre de réglage de l'algorithme. Finalement, pour éviter les minima locaux,

on effectue une mise à jour des pistes avec l'équation (1.2.14) :

$$\tau_{ij}(t+1) = (1-\rho)\cdot\tau_{ij}(t) + \triangle\tau_{ij}(t) \tag{1.2.14}$$

avec $\triangle\tau_{ij}(t) = \sum_{k=1}^{m} \triangle\tau_{ij}^{k}(t)$, m le nombre de fourmi et ρ un nombre aléatoire. Les étapes de l'approche par colonie de fourmis du problème du voyageur de commerce sont résumées dans l'algorithme 1.4.

1.3. Systèmes complexes

La transition entre les paragraphes "méthodes d'optimisation" et "systèmes complexes" est certes un peu abrupte, mais nous avons choisi de présenter les éléments significatifs des outils que nous avons développés, afin de les situer.

Les systèmes complexes sont l'une des préoccupations majeures des ingénieurs. A titre d'exemple, les réseaux électriques peuvent être considérés comme des systèmes complexes en condition limite de fonctionnement où l'augmentation régulière de la demande électrique associée à l'amélioration de la qualité du réseau (dimensionnement, entretien, stabilité, ...) peuvent devenir des forces antagonistes, conduisant alors à des perturbations sensibles. L'analyse des travaux issus de la littérature scientifique laisse apparaître différentes voies de modélisation, les unes privilégiant la nature topologique du réseau, les autres s'orientant sur la dynamique des phénomènes mis en jeu.

Ces travaux, essentiellement développés aux États-Unis, bâtissent le pont rapprochant le domaine de la physique statistique du monde électrique. Nous citons à titre d'exemple :

- les modèles HOT (*Highly Optimized Tolerant*), introduits par J. Carlson et J. Doyle en 1999 ([Doyl00, Mann05, Carl99]) qui visent à modéliser des systèmes complexes et robustes dans un environnement incertain,
- les modèles de I. Dobson et al [Dobs01a, Dobs01b] basés sur des processus stochastiques dont les auteurs ont établi les liens étroits entre les systèmes "Self-Organized Critically", SOC ("Criticalité Auto-Organisée", CAO) et le réseau électrique,
- les approches structurelles du problème par la théorie des graphes (abordée dans le chapitre V).

Dans tous les cas, le dénominateur commun devient l'analyse de la complexité pour mieux la servir. Le premier ordre dans cette complexité est l'existence de lois d'échelles explicitant ainsi une dynamique entre ces différents niveaux d'échelles à travers, par exemple, l'apparition de lois de puissance.

1.3.1. Graphes aléatoires

Le concept de graphes aléatoires est introduit pour la première fois par Erdös et Rényi en 1959 [Erdo59]. Leur première approche de construction d'un graphe aléatoire consiste à se fixer un nombre de liens K à connecter, puis à sélectionner aléatoirement deux nœuds à relier à chaque étape de la construction. Ce type de graphes est noté $G_{N,K}^{ER}$. Une autre alternative de cette approche consiste à choisir deux nœuds et les connecter avec une probabilité $0 < p < 1$. Ce type de graphes est noté $G_{N,p}^{ER}$.

Dans le cas de réseaux aléatoires de petite dimension $G_{N,p}^{ER}$, la distribution des degrés $p(k)$ est une loi binomiale de paramètre p contrairement aux réseaux aléatoires de grande dimension $G_{N,p}^{ER}$ qui ont une distribution des degrés de loi de Poisson de paramètre $p(n-1)$.

Dans tout ce qui suit, on désigne les graphes d'Erdös et Rényi $G_{N,K}^{ER}$ et $G_{N,p}^{ER}$ par ER.

1.3.2. Graphes aléatoires généralisés

Bien que les graphes ER soient les modèles les plus étudiés, ils restent néanmoins moins représentatifs des réseaux réels. Il serait donc intéressant de savoir reproduire les caractéristiques (distribution des degrés) d'un certain type de réseau.

Les *réseaux aléatoires généralisés* nous permettent de construire des réseaux ayant une certaine distribution des degrés [Bend78] $p(k)$, en se fixant une séquence d'entiers $D = \{k_1, ..., k_N\}$ de manière à ce que $\sum_{i}^{N} k_i = 2K$.

Pour les différentes méthodes de construction des réseaux ayant une certaine distribution des degrés $p(k)$, on se réfèrera au travaux de Newman et al [Newm02, Newm03].

1.3.3. Réseaux petits mondes (*Small World*)

L'une des expériences les plus surprenantes de Milgram consistait à remettre des lettres à une centaine de personnes dont les expéditeurs ne connaissent pas l'identité des destinataires et évaluer ensuite le nombre d'étapes nécessaires pour que les personnes ciblées reçoivent leur lettre. La remise de la lettre d'une personne à une autre se fait de la manière suivante : si la personne détentrice de la lettre connait l'identité du destinataire, elle est directement remise à ce dernier sinon, elle est remise à une autre personne, qui a une forte chance de le connaître. Milgram constata qu'il suffit de six étapes (en moyenne) pour que les destinataires reçoivent leurs lettre. D'où le théorème des "*six degrés*".

Ce résultat impressionnant peut être expliqué par l'effet "*petit monde*" (petit diamètre) de certains réseaux complexes. De tels réseaux sont appelés *réseaux "petit monde"* (*"Small World"*). Ils sont habituellement caractérisés par :
- leur petit diamètre, qui peut être remplacé par une petite moyenne des chemins géodésiques,
- leur grand coefficient de *clustering* C.

Le modèle de *Watts et Strogatz* (*WS*) [Watt98] est l'une des méthodes produisant des graphes notés $G_{N,K}^{WS}$ prenant en considération ces deux caractéristiques des réseaux petits mondes (petit diamètre et grand coefficient de *clustering* C).

1.3.4. Réseaux libres d'échelle (*Scale-Free*)

Plusieurs réseaux réels (réseaux internet, réseaux d'interaction des protéines, etc) ont une distribution des degrés de loi de puissance de la forme $P(k) \sim ck^{-\alpha}$. Ces réseaux sont appelés *réseaux libres d'échelle* (*scale-free*) et sont caractérisés par la présence d'un nombre très petit de nœuds de fort degré. La notion de "libre d'échelle" vient des lois de puissance qui sont les seules fonctionnelles $f(x)$ restant inchangées (à un facteur multiplicatif près) après changement d'échelle.

Les modèles de *Barabási-Albert* (*BA*) [Albe02] sont des exemples de modèles qui produisent des réseaux libres d'échelle. Le principe de base de la construction de ce genre de réseaux est, qu'en partant de m_0 nœuds isolés, on ajoute successivement $N - m_0$ nœuds ayant des degrés $m_k \leq m_0$, $k = 1...N - m_0$, et que l'on relie à un nœud déjà existant de façon proportionnelle au degré de ce dernier. Ainsi, les nœuds de fort degré auront tendance à se lier d'avantage à un nouveau nœud arrivant. Autrement dit, les nœuds

arrivants se lient aux nœuds existants du graphe de manière *préférentielle*.

Pour plus de détails sur les définitions et les modèles introduits ci-dessus, on se réfèrera à [Bocc06, Doro08, Doro03, Newm03].

1.3.5. Quelques résultats inspirés de la théorie de la percolation

Dans [Cohe00, Cohe01], une condition générale a été définie κ, basée sur la théorie de la percolation, c'est-à-dire sur la fraction de nœuds à enlever avant qu'un réseau complexe ne se désintègre (équation (1.3.1)). Il a été montré en particulier que, pour des réseaux à distribution des degrés de puissance $P(k) \sim ck^{-\alpha}$ (réseaux "libres d'échelle"), la fraction p_c de nœuds à enlever (pannes) s'exprime en fonction de la connectivité k et la quantité f_c (équation (1.3.2)).

Ceci signifie que, pour $\alpha \leq 3$, le réseau ne se désintègre presque jamais, quelle que soit la fraction de nœuds enlevée ($p_c \sim 0.99\%$). En effet, dans ce cas la, κ_0 dans l'équation (1.3.2) converge, proportionnellement à K, vers l'infini. Par conséquent, la quantité $1-p_c$ dans l'équation (1.3.1) converge à son tour vers 0.

On rappelle que la distribution des degrés des réseaux libres d'échelle est donnée par $P(k) \sim ck^{-\alpha}$, en désignant par $k = m, m+1, ..., K$ la connectivité des nœuds, et m, K, respectivement les connectivités minimale et maximale du réseau. La quantité $< k^n >$ désigne le moment d'ordre n de la variable k.

La condition générale d'existence d'un grand composant connexe est :

$$\kappa \equiv \frac{< k^2 >}{< k >} = 2 \Rightarrow f_c = 1 - p_c = \frac{1}{\kappa_0 - 1} \tag{1.3.1}$$

pour les réseaux "libres d'échelles" :

$$\kappa_0 \rightarrow |\frac{2-\alpha}{3-\alpha}| \times \begin{cases} m, & si\, \alpha > 3; \\ m^{\alpha-2}K^{3-\alpha}, & si\, 2 < \alpha < 3; \\ K, & si\, 1 < \alpha < 2. \end{cases} \tag{1.3.2}$$

Dans [Cohe02], R. Cohen & al. ont exprimé le seuil de percolation par la probabilité P_∞ (équation (1.3.3)) qu'un nœud de connectivité k appartienne au plus grand composant connexe qui dépend d'un exposant de criticité β, qui lui même dépend de α dans le cas des réseaux libres d'échelle (équation (1.3.5)). D'une part, les résultats de [Cohe00] sont retrouvés pour $\alpha \leq 3$ et d'autre part, il est montré que pour, $3 < \alpha < 4$, l'exposant de criticité β dépend fortement de α. Par ailleurs, ils ont pu exprimer la moyenne des composants connexes $< s >$ après suppression d'une fraction p_c de nœuds (gouvernée par un exposant γ, équation (1.3.6)), par l'équation (1.3.4) :

$$P_\infty \sim (q - q_c)^{-\beta} \tag{1.3.3}$$

$$< s >= (q - q_c)^{-\gamma} \tag{1.3.4}$$

en désignant, $q = 1 - p$ (p étant la fraction de nœuds enlevés ou en panne).

Ainsi, pour les réseaux "libres d'échelle", les exposants des équations (1.3.3) et (1.3.4) se réduisent à :

$$\beta = \begin{cases} \frac{1}{3-\alpha}, & 2 < \alpha < 3; \\ \frac{1}{\alpha-3}, & 3 < \alpha < 4, \\ 1, & \alpha > 4. \end{cases} \tag{1.3.5}$$

$$\gamma = \begin{cases} 1, & \alpha > 3; \\ -1, & 2 < \alpha < 3. \end{cases} \tag{1.3.6}$$

Dans [Schw02], N. Schartz & al ont effectué les mêmes travaux que dans [Cohe02] pour exprimer les exposants de criticité β et γ sur des graphes orientés et corrélés à loi de puissance (équation (1.3.8)). Ils ont défini un diagramme de phase qui indique l'existence de trois régimes (figure 1.3.1), déterminés par l'exposant de la distribution des degrés α^* :

$$\alpha^* = \alpha_{in} + \frac{\alpha_{in} - \alpha_{out}}{\alpha_{in} - 1} \tag{1.3.7}$$

en désignant par α_{in}, α_{out} respectivement les connectivités des arcs entrants et sortants.

Ils ont conclu que dans ce type de réseaux, l'exposant de criticité β est le même que dans le cas des graphes corrélés, en remplaçant α par α^*.

$$\beta = \begin{cases} \frac{1}{\alpha_{out}-2}, & 2 < \alpha_{out} < 3; \\ 1, & \alpha_{out} > 3. \end{cases} \tag{1.3.8}$$

L'étude du diamètre d'un réseau complexe est l'une des études les plus importantes. En effet, un réseau devient efficace pour l'acheminement de l'énergie (ou d'informations) dès que son diamètre est petit. Le petit diamètre caractérise les réseaux "*Small World*".

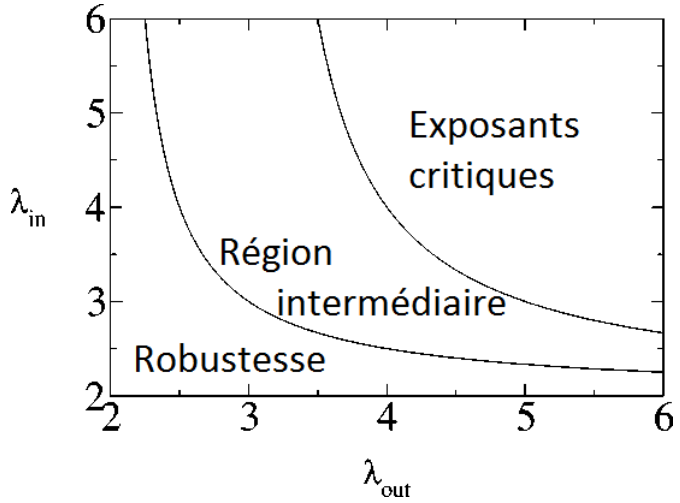


FIGURE 1.3.1.: Diagramme de phase des différents régimes dans le cas des réseaux orientés et corrélés à loi de puissance [Schw02].

Dans [Cohe03], R. Cohen & al ont étudié le diamètre des réseaux aléatoires "libres d'échelle" et montrent

que ce dernier est très petit, en comparaison des réseaux aléatoires réguliers :

- Pour les réseaux aléatoires d'Erdös-Rény(ER), réseaux "*Small World*" :

$$d \sim ln\, N\ .$$

- Pour les réseaux "libres d'échelle" :

$$d \sim \begin{cases} ln\, ln\, N, & 2 < \alpha < 3; \\ ln\, n, & \alpha > 3. \end{cases}$$

en désignant par :
d : le diamètre du réseau,
N : le nombre de nœuds,
α : le paramètre de la loi de puissance.

Dans [Brau03], les auteurs ont étudié la longueur moyenne des chemins optimaux l_{opt} des réseaux aléatoires d'Erdös-Rény (ER), Watts-Strogatz (WS) et des réseaux "*Small World*" en présence d'hétérogénéités (désordre) :

- Dans le cas des réseaux sans corrélations et sans désordre, la longueur moyenne des chemins optimaux est :
  - pour les trois réseaux (ER, WS et "*Small World*") : $l_{opt} \sim ln\, N$
  - pour les réseaux libres d'échelle $l_{opt} \sim \begin{cases} ln\, ln\, N, & 2 < \alpha < 3 \\ ln\, n, & \alpha > 3 \end{cases}$
- Dans le cas des réseaux sans corrélations et avec désordre, la longueur moyenne des chemins optimaux est :
  - pour les réseaux ER et WS : $l_{opt} \sim N^{\frac{1}{3}}$
  - pour les réseaux "libres d'échelle" : $(*)\, l_{opt} \sim \begin{cases} ln^{\alpha-1} N, & 2 < \alpha < 3 \\ N^{\frac{\alpha-3}{\alpha-1}}, & 3 < \alpha < 4 \\ N^{\frac{1}{3}}, & \alpha > 4 \end{cases}$
- Dans le cas des réseaux avec corrélations et avec désordre, la longueur moyenne des chemins optimaux est :
  - pour les réseaux "libres d'échelle" : $l_{opt} \sim \begin{cases} N^{\frac{1}{3}}, & \lambda > 0\, et\, \alpha > 2 \\ N^{\nu_{opt}}, & \lambda \leq 0 \end{cases}$ ν_{opt} étant le même exposant que dans $(*)$.

En présence d'une faible hétérogénéité, la longueur du chemin optimal est la même que dans les réseaux sans désordre. En revanche, dans le cas d'une présence de forte hétérogénéité, la longueur du chemin optimal l_{opt} est influencée par le désordre. Les mêmes auteurs [Zhen06] ont modélisé les corrélations entre les degrés de distribution d'un réseau et ont étudié la longueur moyenne des chemins optimaux d'un réseau libre d'échelle avec corrélation entre les degrés de distribution. Ils concluent que la longueur du chemin optimal l_{opt} croît de manière très importante dans les réseaux à forte hétérogénéité.

B. Wang et al [Kazu08] ont modélisé le désordre d'un réseau pondéré grâce à un coefficient de blocage J :

$$J = << \frac{\sum_{i=1}^{N} [\sum_{j \in \Omega(i)} \omega_{reçu(j)}(i) - C_i]}{\sum_{i=1}^{N} C_i} >_{\nu} >_{réseau} \tag{1.3.9}$$

en désignant par $\Omega(i)$ l'ensemble des voisins du nœud i ,

$$[x] = x \text{ si } x > 0 \text{ , } 0 \text{ sinon,}$$

$\omega_{reçu(j)}(i)$ est le poids du lien (nœud i vers nœud j), C_i le poids total sortant du nœud i, $< ... >_\nu$ est la moyenne sous la distribution ν, $< ... >_{réseau}$ est la moyenne sous les différentes configurations du réseau.

Ils montrent, comme dans [Brau03], que le blocage (i.e. désintégration) d'un réseau augmente avec son hétérogénéité. Y. Wang et al [Wang08] ont étudié la robustesse des réseaux "libres d'échelle" en s'attaquant non pas aux nœuds du réseau, mais aux liens de ce dernier, en partitionnant le graphe en deux pour trouver une coupe minimale. Ils concluent que ce genre d'attaque ne détruit pas forcément le réseau et que ce dernier reste un réseau libre d'échelle avec un diamètre très petit.

L'hétérogénéité peut être mesurée par une entropie du réseau [Canc03, Sole04]. Dans [Wang06], les auteurs ont montré que la maximisation de la robustesse d'un réseau libre d'échelle est équivalente à la maximisation de l'entropie de la distribution des degrés du réseau (équation (1.3.11)). L'entropie d'un réseau est donnée par la quantité (équation (1.3.10)) :

$$H = -\sum_{k=1}^{N-1} p(k) log(p(k)) \tag{1.3.10}$$

$$max\, H(\alpha, m, N)\, sc :< k >= const \tag{1.3.11}$$

en désignant par $p(k)$ la distribution des degrés du réseau et m, la connectivité minimale du réseau.

Ainsi, il a été montré que l'entropie est maximale pour des réseaux exponentiels (la distribution des degrés des réseaux exponentiels est : $p(k) = \frac{e^{-\frac{k}{\gamma}}}{\gamma}$).

L.K. Gallos et al [Gall05] ont introduit une probabilité $w(k_i)$ (équation (1.3.12)) qu'un nœud de k_i liens tombe en panne afin d'étudier le seuil de percolation p_c quand les degrés de plus haute importance ne sont pas connus ou, en d'autres termes, quand les attaquants ne connaissent pas les positions des nœuds vulnérables dans le réseau :

$$w(k_i) = \frac{k_i^\alpha}{\sum_{i=1}^{N} k_i^\alpha}, -\infty < \alpha < +\infty. \tag{1.3.12}$$

Cette modélisation permet de définir une stratégie de défense contre les attaques quand les coûts nécessaires à l'immunisation des nœuds vulnérables sont trop élevés. Les auteurs concluent qu'une faible connaissance du réseau libre d'échelle dégrade rapidement ce dernier en cas d'attaque ($\alpha > 0$). En revanche, une faible connaissance du réseau (libre d'échelle) permet de réduire la propagation des pannes en cas de défense ($\alpha <= 0$) contre les attaques (propagation des virus à travers un réseau social, par exemple). G. Paul & al [Paul07] ont étudié le seuil de percolation p_c des graphes aléatoires réguliers à l'aide de l'algorithme de partitionnement de graphes METIS [Kary8], soit :

$p_c = 1 - \frac{2}{k}$ où k est la connectivité du réseau et p_c la fraction de nœuds enlevée.

Au seuil de percolation p_c, le diamètre et la taille du plus grand composant connexe sont respectivement : $l \sim N^{0.25}$ et $S \sim N^{0.4}$.

Les auteurs concluent que les graphes aléatoires réguliers sont difficiles à attaquer, éventuellement à immuniser, en raison de leur homogénéité.

Il a été prouvé que les réseaux à distribution bimodale des degrés sont optimaux face aux attaques et aux pannes séparément [Tani05].

T. Tanizawa & al [Tani05] ont étudié la robustesse des réseaux aléatoires notamment, les réseaux "libres d'échelle" et à distribution bimodale des degrés, après une série d'attaques simultanées. Ils ont montré que n attaques successives fragilisent plus le réseau que n attaques simultanées car, les attaquants sont informés des nœuds fragiles (distribution des degrés) après chaque attaque. Ils concluent en outre, comme dans [Tani05], que les réseaux à distribution bimodale restent robustes aux attaques et aux pannes successives.

Les graphes robustes se caractérisent par une fraction r de nœuds de distribution des degrés : $k_2 = (< k > -1 + r)/r$ et les autres nœuds de degré 1. Les réseaux à distribution bimodale sont optimaux pour $0.03 < r < 0.9$ si $p_t/p_r \leq 1$, sinon les réseaux à distribution uniformes sont plus robustes, en désignant par p_t et p_r les fractions de nœuds en panne et attaqués respectivement.
Dans [Shar03], B Shargel et al. construisent, grâce à une algorithmique, une classe de réseaux où les réseaux libres d'échelle et exponentiels sont des cas particuliers et donc héritent des avantages des deux réseaux, à savoir, la robustesse aux attaques comme dans le cas des réseaux exponentiels [Sole08] et la robustesse aux pannes, comme dans le cas des réseaux libres d'échelle [Cohe00, Cohe01].

La distribution des degrés des réseaux exponentiels est $p(k) = \frac{e^{-\frac{k}{\gamma}}}{\gamma}$. Le seuil critique p_c est donné par : $1 - (ln\, p_c - 1) = \frac{1}{2\gamma - 1}$.

En résumé, grâce à leur faible connectivité globale (degré moyen) et leur petit diamètre, les réseaux dits "*Small World*" et les réseaux "libres d'échelle" sont efficaces pour l'acheminement d'informations (globalement et localement). En d'autres termes, tout comme les réseaux purement aléatoires, les graphes "*Small World*" sont statiquement robustes (résistants à des pannes aléatoires). En revanche, ils sont plus vulnérables à des attaques (suppression d'éléments de plus haut degré ou de nœuds plus chargés, par exemple)[Bocc06, Newm03, Doro08]. Pareillement, les réseaux homogènes, tout comme les réseaux exponentiels, semblent être plus résistants à des attaques que les réseaux hétérogènes [ZhiX08, Mott02].

1.3.6. Robustesse dynamique d'un réseau

Il existe dans la littérature plusieurs modèles [Cruc04, Kinn05, Carr02, Watt02] destinés à reproduire le plus fidèlement possible la vie (ou la dynamique) d'un réseau et basés généralement sur un concept itératif. Que le réseau soit pondéré [ZhiX08, Bocc06, Cruc05, Kinn05] ou non [Bocc06], les auteurs tentent de définir des lois de redistribution des flux de puissance en supprimant l'élément défaillant (dans la plupart des cas) ou en pénalisant les flux de puissance des éléments défaillants (par exemple, en fonction de leur degré moyen et leur capacité maximale) [Cruc04, Kinn05]. Cette étude permet, entre autres, de définir un seuil critique de déclenchement en cascade. Il est clair que la dynamique d'un réseau dépend fortement de sa topologie.
M. Rosas-Casas et B. Corominas-Murtra [Rosa12] dans leur étude du réseau électrique européen ont mis en évidence cette relation. Ils ont identifié trois grandeurs qui influent sur la dynamique d'un réseau :

– le degré moyen $< k >$,

– la régularité de la distribution des chemins optima dans le réseau,
– les motifs du graphe (ou sous graphe).

En résumé, l'étude de la robustesse d'un réseau complexe consiste à déterminer la capacité du système à survivre à des perturbations (pannes, attaques ou variation des charges dans le réseau). Pour ce faire, plusieurs indices de vulnérabilité ont été introduits, dont les plus importants sont la distribution des degrés du réseau (ou sa connectivité), l'homogénéité du réseau, le diamètre du réseau, la distribution des chemins géodésiques dans le réseau, qui peut être exprimée par les coefficients de centralité et d'intermédiarité des nœuds ou des liens et les motifs (ou les sous-graphes du réseau). Nous y reviendrons dans le chapitre IV.

1.4. Dispositifs FACTS

Depuis les dernières années, le marché de l'offre de l'énergie électrique est ouvert à la concurrence afin de réduire le prix de l'énergie. Ce changement a traduit la séparation des activités de production, de transport et de distribution. Bien que le client puisse choisir son producteur, le transporteur reste inchangé. En effet, l'activité du transport garde son monopole, et ce réseau est géré par un gestionnaire du réseau indépendant : le TSO (*Transmission System Operator*). Ce gestionnaire a pour fonction principale de coordonner les échanges de puissances afin d'assurer le bon fonctionnement et la pérennité du réseau. La possibilité de construire de nouveaux axes de transport, afin d'accroître les performances et la robustesse du réseau, est de plus en plus difficile, sous la pression des milieux écologistes et la lenteur des administrations. Ceci pousse le gestionnaire du réseau à disposer d'un moyen permettant de contrôler les transits de puissance dans les lignes, afin que le réseau existant puisse être exploité de la manière la plus efficace et la plus sûre possible.

Les dispositifs FACTS (*Flexible Alternative Current Transmission System*) permettent de contrôler les transits de puissance d'une façon continue sur le réseau électrique. Ils agissent en fournissant ou en absorbant de la puissance réactive, en augmentant ou réduisant la tension aux nœuds, en contrôlant l'impédance des lignes ou en modifiant les phases des tensions. Un avantage supplémentaire est qu'ils permettent une extension des limites du réseau.

En régime permanent, ces dispositifs sont utilisés dans deux cas :
– le maintien de la tension, en fournissant de la puissance réactive, lorsque la charge est élevée et que la tension est trop basse ; dans le cas inverse, ils en absorbent si la tension est élevée ;
– le contrôle des transits de puissance, afin d'éviter qu'une ligne soit plus sollicitée qu'une autre, et éviter d'atteindre ses limites. Ils agissent alors en contrôlant la réactance des lignes et en ajustant les déphasages.

En régime transitoire, ils permettent l'amélioration de la stabilité des réseaux (réduction des oscillations de puissances échangées).

Plusieurs types de FACTS ont été développés avec des architectures et des technologies différentes comme :
– Le SVC (*Static Var Compensator*)
– Le STATCOM (*Static Synchronous Compensator*)
– Le TCSC (*Thyristor Controlled Series Capacitor*)
– L'UPFC (*Unified Power Flow Controller*)

Chacun possède ses caractéristiques propres et peut être utilisé pour répondre à des besoins bien précis.

1.4.1. Le SVC

Le SVC est un dispositif *shunt* (compensation en parallèle) pouvant être connecté aux nœuds ou sur un point sectionnant une ligne. Il permet l'accroissement de la puissance transmissible dans les réseaux, en fournissant ou absorbant de la puissance réactive. En régime permanent, la compensation réactive est utilisée pour la sectionnalisation des lignes (ce qui permet d'accroître la puissance transmissible des longues lignes de transport en contrôlant la tension au point de section) et le maintien de la tension aux nœuds. En régime transitoire, les dispositifs *shunt* permettent un contrôle dynamique de la tension pour l'amélioration de la stabilité transitoire et l'amortissement des oscillations de puissance.

Théoriquement, le SVC est sans perte, ce qui veut dire que l'admittance est purement imaginaire et est modélisée par un interrupteur qui commute entre un condensateur et une inductance. Lorsque l'interrupteur est sur la capacité, le courant est déphasé positivement, ce qui a pour effet l'injection de puissance réactive dans le système. En revanche, si l'interrupteur est sur l'inductance, cela engendre un déphasage négatif du courant, la puissance réactive est alors absorbée.

1.4.2. Le STATCOM

Le STATCOM (basé sur la structure d'un convertisseur de tension triphasée) est un dispositif *shunt* utilisé principalement aux nœuds du réseau pour la compensation dynamique, afin de faciliter la tenue en tension, d'accroître la stabilité en régime transitoire et d'amortir les oscillations de puissance. Le STATCOM réagit selon la différence entre la tension de sortie du convertisseur u et la tension de la ligne u_0 :

- si $u_0<u$: le courant circule du convertisseur vers le réseau, le STATCOM produit alors de la puissance réactive (comportement capacitif) ;
- si $u_0>u$: le courant circule du réseau vers le convertisseur, le STATCOM absorbe de la puissance réactive (comportement inductif) ;
- si $u_0=u$: aucun courant ne circule et il n'y a pas d'échange d'énergie réactive.

Sa capacité à soutenir la tension du réseau est meilleure que celle du SVC et il présente certains avantages, comme :
- l'espace nécessaire pour l'installation est réduit, en raison de l'absence d'inductance,
- le recours à des filtres d'harmoniques n'est pas nécessaire,
- les performances en régime dynamique sont meilleures.

1.4.3. Le TCSC

Le TCSC (dispositif de compensation en série) est formé d'une inductance commandée par thyristors en parallèle avec un condensateur. Placé sur une ligne, il permet de modifier sa réactance et a pour principal effet de pouvoir contrôler le flux de puissance circulant dans la ligne où il est placé. La puissance réactive produite par le TCSC est proportionnelle à la puissance circulant dans la ligne. Ce dispositif agissant sur le courant dans la ligne, les puissances apparentes varient de la même manière que les puissances actives.

1.4.4. L'UPFC

L'UPFC est une structure hybride entre les compensations parallèle et série. Il est composé de deux convertisseurs de tension reliés par une liaison à courant continu. Il s'agit de la combinaison d'un STATCOM et d'un SSSC (*Static Synchronous Series Compensator*). Son principe est de dériver une partie du courant circulant dans la ligne pour le réinjecter avec une phase appropriée.

Son côté hybride lui permet de remplir toutes les fonctions des autres dispositifs FACTS :
- le réglage de la tension,
- le contrôle des flux de puissances active et réactive,
- l'amélioration de la stabilité,
- la limitation des courants de court-circuit,
- l'amortissement des oscillations de puissance.

1.5. Conclusion

Dans ce chapitre, un ensemble de techniques d'optimisation "difficile" a été présenté. Cette liste n'est bien évidemment pas exhaustive. Toutefois, le principe de fonctionnement des approches les plus connues et les plus utilisées a été présenté. Dans un premier temps, un éventail de métaheuristiques dont la recherche tabou (TS), le recuit simulé (SA), l'optimisation par essaim de particules (OEP), les algorithmes évolutionnaires (AEs) et les algorithmes de colonie de fourmis (ACO) ont été détaillés. Nous avons vu que la plupart de ces approches s'inspirent de comportement biologiques (les algorithmes évolutionnaires) et/ou éthologique (les algorithmes de colonies de fourmis ou l'optimisation par essaim de particules) ou encore de la physique (le recuit simulé) et usent abondamment de métaphores et de termes issus de la génétique et la sociologie.

Une des méthodes qui a retenu notre attention et, en conséquence, a été détaillée dans ce chapitre est l'optimisation par essaim de particules. En effet, bien que récente (1995), l'optimisation par essaim de particules présente une popularité grandissante quant à la résolution de nombreux problèmes difficiles et, plus particulièrement, à variables continues. C'est pour cette raison que notre choix s'est porté sur ce type d'algorithme d'optimisation. Cependant, en cas de mauvais réglage des paramètres ou méconnaissance du problème à optimiser, cette méthode présente l'inconvénient d'un risque élevé de convergence prématurée. Pour remédier à cette convergence prématurée, plusieurs variantes et confinements ont été proposés dans la littérature. L'une de ces alternatives consiste en l'introduction du concept de distribution de probabilité (GBB-PSO, ...) dans le concept de l'optimisation par essaim de particules. Sur cette base, deux nouveaux algorithmes basés sur le concept des distributions de probabilité de Lévy (α-SLPSO et α-SLS) seront proposés dans le chapitre III.

Par ailleurs, la cohérence du travail nous a amenés à poser les pré-requis de la modélisation des systèmes complexes, issue de la théorie des graphes et de la théorie de percolation. Appliquée au comportement des réseaux électriques, cette modélisation constitue, à notre sens, un apport essentiel à l'études du comportement des réseaux électriques en condition limite de fonctionnement.

CHAPITRE 2

CONTRIBUTION À L'ESTIMATION DES PARAMÈTRES D'UNE LOI $\alpha - STABLE$

2.1. Introduction

Les avantages et les intérêts potentiels des distributions de probabilité $\alpha - stables$ sont aujourd'hui largement reconnus par les chercheurs et la communauté technique. Pour la commodité du raisonnement, J. Lévy-Véhel et C. Walter [Levy02] ont déjà mis en exergue les avantages des lois α-stables dans la modélisation des marchés financiers. En effet, il est maintenant admis la non normalité des distributions de probabilité. B. Mandelbrot [Mand63] et E. Fama [Fama65] ont proposé dans les années 1960 une possible alternative consistant à introduire les lois α-stables. Le deuxième exemple le plus frappant est la preuve d'asymétrie dans les rendements sur les marchés boursiers. En effet, B.D. Fieletz & E.W. Smith [Fiel72] et R.A. Leitch & A.S. Paulson [Leit75] ont fait valoir l'exigence de la relaxation de l'hypothèse de symétrie de Fama&Rolls [Fama65, Fama71]. En raison de leur symétrie et de leur queue trop estompée, les distributions de probabilité gaussiennes échouent dans la modélisation de la plupart des phénomènes physiques à caractère invariant d'échelle. Les lois $\alpha - stables$ généralisent alors le cadre d'analyse.

Le défi actuel est d'obtenir une estimation adéquate de ces distributions de probabilité. La plupart de ces méthodes destinées à estimer les paramètres des lois α-stables (équation ((2.2.1))) sont gourmandes en temps et/ou sont souvent axées sur seulement deux paramètres (α, γ) ou avec restriction des paramètres. A titre d'exemple, citons les estimateurs basés sur les quantiles de la distribution [Fama71, McCu86, Feue81], l'estimateur basé sur une transformation de la fonction caractéristique [Pres72], la technique développée par S.M. Kogon et D.B. Williams [Kogo98] utilisant une estimation empirique de la fonction caractéristique, la méthode de régression développée par I.A. Koutrouvelisv [Kout80] ou encore des méthodes basées sur le maximum de vraisemblance [Brot83, Hold73, Pant92, Wors95]. Dans la littérature, il existe plusieurs études comparatives sur les méthodes d'estimation des paramètres des lois α-stables. La plupart d'entre elles sont concentrées sur certaines classes d'estimateurs les plus cités ([Mitt01], [Kogo98], [Wero95]).

Dans ce chapitre, nous proposons une nouvelle approche pour aborder le problème d'estimation des paramètres d'une distribution de probabilité α-stable. Dans un premier temps, nous esquissons les notions et les propriétés fondamentales des lois α-stables. Notamment, nous donnons deux principales caractéri-

sations de celles-ci et nous insisterons sur le caractère auto-similaire (invariant d'échelle ou encore indéfiniment divisible). Puis nous décrivons très brièvement les outils utilisés pour parvenir à notre estimateur non paramétrique des lois α-stables, en insistant sur le test statistique de Kolmogorov-Smirnov et sur le générateur de lois α-stables. Dans un deuxième temps, nous transformons la problématique d'estimation des paramètres d'une loi α-stable en un problème d'optimisation (équation ((2.1.1))) que nous abordons à l'aide des métaheuristiques et plus particulièrement, l'optimisation par essaim de particules (OEP) [Kenn95, Eber01, Siar03].

$$Prob^1 \begin{cases} \min\limits_{\alpha,\beta,\gamma,\delta} f(\alpha,\beta,\gamma,\delta) \\ g_i \,:\, g_i(\alpha,\beta,\gamma,\delta) \in Support_i \end{cases} \tag{2.1.1}$$

L'estimateur présente l'avantage d'être intrinsèquement parallélisable. Dans la dernière partie de ce chapitre, nous proposons un arsenal de fonctions exploitables sous GP/GPU afin d'améliorer le temps d'exécution de notre estimateur.

2.2. Distribution de probabilité de Lévy

2.2.1. Définitions

Lorsqu'une combinaison linéaire de copies de variables aléatoires indépendantes et identiquement distribuées (iid) est une variable aléatoire de même loi de probabilité, à un paramètre d'échelle et de positionnement près, la variable en question est dite "*stable*". En d'autres termes, la variable aléatoire X est dite *stable* si et seulement si, pour toute constante positive a_i et pour tout jeu de n variables aléatoire indépendantes $X_{i,..,n}$ identiquement distribuées et de même loi de probabilité que X, la variable aléatoire $Y_n = \sum_{i=1}^{n} a_i X_i$ a la même distribution de probabilité que Y, pour certaines constantes et $d_n \in \mathbb{R}$, telles que $Y = c_n X + d_n$.

Pour un certain $\alpha \in]0,2]$, la constante c_n est nécessairement de la forme $n^{1/\alpha}$ d'où le nom de *"α-stabilité"*. Cette famille de distribution de probabilité est appelée "*les distributions de probabilité α-stable de Lévy"*.

Plus formellement, une variable aléatoire X de loi de probabilité α-*stable* a pour fonction caractéristique $\varphi(t)$ (équation (2.2.1)) :

Définition 3. la fonction caractéristique $\varphi(t)$ des lois α-stables est donnée par

$$\begin{cases} \varphi(t,\alpha,\beta,\gamma,\mu) & = e^{it\mu - |\gamma t|^\alpha (1 - i\beta Signe(t)\Phi)} \\ \Phi = -(\frac{2}{\pi}) \log|t| & si\ \alpha = 1, \\ \Phi = \tan(\frac{\pi\alpha}{2}) & sinon \end{cases} \tag{2.2.1}$$

et la distribution de probabilité $f(x)$, qui n'a pas de forme analytique, est implicitement définie par l'équation (2.2.2) :

$$f(x) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} \varphi(t) e^{-ixt} dt \tag{2.2.2}$$

La caractérisation des lois α-stables dans l'équation (2.2.1) a le désavantage d'être non continue pour tous les paramètres α, β, γ et μ notamment en $\alpha = 1$. Pour remédier à ce problème, Zolotarev [Zolo66] a proposé la caractérisation suivante :

Définition 4. Une variable aléatoire X suit une loi α-stable si et seulement si sa fonction caractéristique s'écrit sous la forme :

$$\log(\varphi(t)) = \begin{cases} -\gamma_2^\alpha |t|^\alpha \exp\{-i\beta_2 signe(t)\frac{\pi}{2}K(\alpha)\} + i\mu t, & si\, \alpha \neq 1 \\ -\gamma_2 |t| \exp\{\frac{\pi}{2} + i\beta_2 signe(t)\log|t|\} + i\mu t, & si\, \alpha = 1 \end{cases} \tag{2.2.3}$$

avec :

$$K(\alpha) = \alpha - 1 + signe(1-\alpha) = \begin{cases} \alpha, & si\, \alpha < 1 \\ \alpha - 2, & si\, \alpha > 1 \end{cases} \tag{2.2.4}$$

les paramètres β et γ, définis dans l'équation (2.2.1), sont liés à β_2 et γ_2 par les équations (2.2.5) et (2.2.6) :

$$\alpha \neq 1 \Rightarrow \begin{cases} \tan\left(\beta_2 \frac{\pi K(\alpha)}{2}\right) = \beta \tan(\frac{\pi\alpha}{2}), \\ \gamma_2 = \gamma(1 + \beta^2 \tan^2 \frac{\pi\alpha}{2})^{\frac{1}{2\alpha}} \end{cases} \tag{2.2.5}$$

$$\alpha = 1 \Rightarrow \begin{cases} \beta_2 = \beta \\ \gamma_2 = \frac{2}{\pi}\gamma \end{cases} \tag{2.2.6}$$

La fonction caractéristique $\varphi(t)$ est alors continue en tout point. Une variable aléatoire X de loi α-stable et d'indice de stabilité α, de paramètre d'asymétrie β, de paramètre d'échelle γ et de paramètre de positionnement μ, est généralement notée $S_\alpha(\gamma, \beta, \mu)$.

Pour simuler des variables aléatoires non uniformes, on utilise la distribution de probabilité ou la fonction de répartition F et non la fonction caractéristique. Il est donc nécessaire d'avoir une caractérisation de cette distribution de probabilité la plus fidèle possible. Zolotarev [Zolo66] donne une forme intégrale de la fonction de répartition $F(x, \alpha, \beta_2)$ valable pour tous paramètres α, β, γ et μ. Cette caractérisation des lois α-stable est résumée dans la proposition 5.

Proposition 5. *Étant donné*

$$\epsilon(\alpha) = signe(1-\alpha),$$

$$\gamma_0 = -\frac{\pi}{2}\beta_2 \frac{K(\alpha)}{\alpha},$$

$$C(\alpha, \beta_2) = 1 - \frac{1}{4}\left(1 + \beta_2 \frac{K(\alpha)}{\alpha}\right)(1 + \epsilon(\alpha))$$

$$U_\alpha(\gamma, \gamma_0) = \left(\frac{\sin\alpha(\gamma - \gamma_0)}{\cos(\gamma)}\right) \tag{2.2.7}$$

$$U_1(\gamma, \beta_2) = \frac{\frac{\pi}{2} + \beta_2\gamma}{\cos(\gamma)} \exp\left(\frac{1}{\beta_2}(\frac{\pi}{2} + \beta_2\gamma)\tan(\gamma)\right) \tag{2.2.8}$$

la fonction de répartition $F(x, \alpha, \beta_2)$ d'une variable aléatoire α-stable, ayant comme fonction caractéristique l'équation (2.2.3)*, est donnée par :*

$$\begin{cases} F(x,\alpha,\beta_2) = C(\alpha,\beta_2) + \frac{\epsilon(\alpha)}{\pi} \int_{\gamma_0}^{\frac{\pi}{2}} \exp\left(-x^{\alpha/(\alpha-1)} U_\alpha(\gamma,\gamma_0)\right) d\gamma & si\, x > 0\, et\, \alpha \neq 1 \\ F(x,1,\beta_2) = \frac{1}{\pi} \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \exp\left(-e^{-x/\beta_2} U_1(\gamma,\beta_2)\right) d\gamma & si\, x > 0\, et\, \alpha = 1 \end{cases} \tag{2.2.9}$$

En se référant aux équations (2.2.1) et (2.2.2), la forme des distributions de probabilité α-stables (figure 2.2.1) est complètement déterminée par quatre paramètres principaux :

1. l'indice de stabilité $\alpha \in]0, 2]$, qui détermine la vitesse à laquelle la queue de distribution s'estompe. Pour $\alpha < 2$, l'indice de stabilité α caractérise le comportement asymptotique de la distribution en question.
2. le paramètre d'asymétrie $\beta \in [-1, 1]$, qui contrôle la symétrie de la distribution. Toutefois, le paramètre d'asymétrie usuel n'est pas défini, car les moments d'ordre k pour $k > 1$ n'existent pas. Par conséquent, l'estimation du paramètre d'asymétrie β n'est pas simple dans ce contexte.
3. le paramètre d'échelle $\gamma > 0$, qui concentre, ou disperse les observations autour de la moyenne. En d'autres termes, il contrôle l'épaisseur de la distribution.
4. le paramètre de localisation ou de positionnement $\delta \in \mathbb{R}$, qui contrôle la position de la distribution.

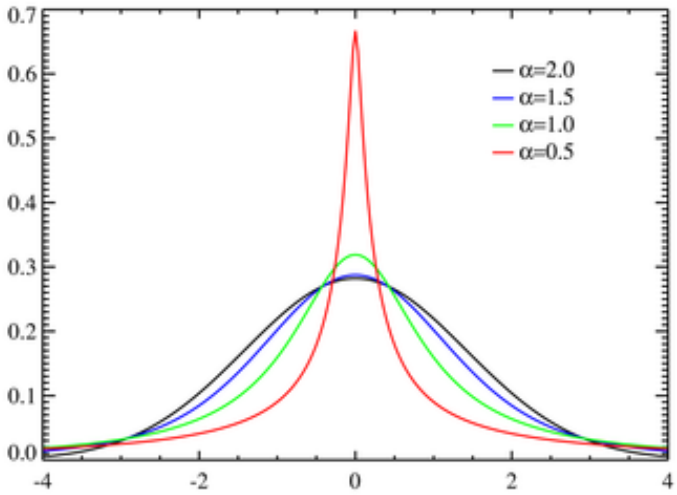


FIGURE 2.2.1.: Influence du paramètre α sur les distributions de probabilité α-stables.

Ce type de distribution de probabilité possède des propriétés intéressantes, à savoir :

i) $X_1 \rightsquigarrow S_{\alpha_1}(\gamma_1, \beta_1, \mu_1)$ et $X_2 \rightsquigarrow S_{\alpha_2}(\gamma_2, \beta_2, \mu_2)$ (iid), alors $X_1 + X_2 \rightsquigarrow S_\alpha(\gamma, \beta, \mu)$ avec

$$\beta = \frac{\beta_1 \gamma_1^\alpha + \beta_2 \gamma_2^\alpha}{\gamma_1^\alpha + \gamma_2^\alpha} \tag{2.2.10}$$

$$\gamma = (\gamma_1^\alpha + \gamma_2^\alpha)^{\frac{1}{\alpha}} \tag{2.2.11}$$

$$\mu = \mu_1 + \mu_2 \tag{2.2.12}$$

ii) $X \rightsquigarrow S_\alpha(\gamma, \beta, \mu)$ et $a \in \mathbb{R}$ alors

$$aX \rightsquigarrow \begin{cases} S_\alpha(|a|\gamma, sgn(a)\beta, a\mu) & si\, \alpha \neq 1 \\ S_\alpha(|a|\gamma, sgn(a)\beta, a\mu - \frac{2}{\pi}a(log|a|)\gamma\beta) & si\, \alpha = 1 \end{cases} \tag{2.2.13}$$

$$X + a \rightsquigarrow S_\alpha(\gamma, \beta, \mu + a) \tag{2.2.14}$$

iii) $X \rightsquigarrow S_\alpha(\gamma, \beta, 0)$ alors

$$\lim_{x\to\infty} x^\alpha P(X > x) \rightsquigarrow C_\alpha(1+\beta)\gamma^\alpha, avec\, C_\alpha\, une\, constante\, en\, fonction\, de\, \alpha$$

$$-X \rightsquigarrow S_\alpha(\gamma, -\beta, 0)) \tag{2.2.15}$$

iv) $X \rightsquigarrow S_\alpha(\gamma, \beta, \mu)$ alors

$$\begin{cases} E|X|^p < \infty & si\, 0 < p \leq 2 \\ E|X|^p\, n'existe\, pas & si\, p > 2 \end{cases} \tag{2.2.16}$$

v) $X \rightsquigarrow S_\alpha(\gamma, \beta, \mu)$ alors, le processus décrit par la v.a. X est auto-similaire en distribution.

vi) $X \rightsquigarrow S_\alpha(\gamma, \beta, \mu)$ alors comme cas particulier, on a :
- $\alpha = 2$, la distribution est réduite à une distribution de probabilité gaussienne $\aleph$ $(\mu = \delta, \sigma = 2\gamma^2)$,
- $\alpha = 1$ et $\beta = 0$, la distribution est réduite à une distribution de probabilité de Cauchy de paramètre d'échelle γ et de positionnement δ,
- $\alpha = 1/2$ et $\beta = 1$, la distribution de probabilité est une loi de probabilité de Lévy de paramètre d'échelle γ et de positionnement μ.

L'objectif de ce chapitre n'est pas de faire une étude approfondie des lois α-stables. Pour plus de détails sur cette famille de distributions de probabilité, on se réfèrera à [Fell66, Niki95].

2.2.2. Générateur des lois $\alpha - stables$ de Mc Culloch

Comme nous l'avons indiqué dans l'introduction, notre algorithme a besoin d'un générateur de loi α-stable. Celui-ci va générer des réalisations d'une loi α-stable $S_\alpha(\gamma, \beta, \mu)$ et permettra de comparer ces dernières à la série de données que l'on veut ajuster. La conception d'un tel outil est loin d'être une tâche facile aux yeux des probabilistes. En effet, en raison de la non-existence de la fonction analytique F^{-1} correspondant à la distribution de probabilité des lois α-stables (sauf cas particuliers), la méthode de la transformée inverse [Devr86] ne peut être utilisée.

Un des premiers simulateurs de lois α-stables consistait à générer des pseudo-nombres indépendants et uniformes entre [0,1] puis à approximer la fonction inverse F^{-1} (équation ((2.2.2))) pour chaque point du support. On obtient alors des réalisations de pseudo-variables aléatoires (iid) de loi α-stable [Fama68, Paul75]. Plusieurs variantes destinées à de grands échantillons ont été proposées par W.H. DuMouchel [Dumo71]. R.N. Mantegna a proposé dans [Mant94] une méthode consistant à faire une série de transformations non linéaires de deux variables gaussiennes pour obtenir des variables aléatoires α-stables par le théorème central limite. Enfin, une autre méthode consiste à approximer l'intégrale dans l'équation (2.2.2) par un autre système de fonctions mutuellement exclusives [Pres92].

	McCulloch		Mantegna		Rejet	
α	1000	100000	1000	100000	1000	100000
0.3	0.01	0.11	0.11	9.20	28.28	31.33
0.6	0.00	0.20	0.10	9.25	3.93	4.71
1	0.00	0.05	0.00	0.44	1.28	1.79
1.3	0.01	0.20	0.02	0.90	1.14	1.58
1.7	0.01	0.22	0.02	0.90	0.90	1.32

TABLE 2.1.: Temps d'exécution (en secondes) des algorithmes McCulloch, Mantegna et de rejet (laptop PC HP®, modèle "Compaq nx5000", 1.5 GHz Intel® Pentium® M processor, 512 MB de RAM et Windows XP *home edition operating system.*

Dans notre estimateur de loi α-stable, nous avons opté pour le générateur codé par McCulloch et Ohio, eux même inspirés des travaux de J.M. Chambers et al [Cham76]. Ce générateur présente l'avantage d'être plus rapide, comme l'indique le tableau (2.1).

Ce générateur développé par Chambers et al, inspiré aussi par les travaux de M. Kanter [Kant75] (génération de lois α-stables positives, $\beta = 1$), se construit en faisant une série de transformations non linéaires de variables uniformes en une variable α-stable. Deux résultats fondamentaux (lemme 6 et théorème 7) ont été démontrés et vont servir par la suite à générer n'importe quelle loi α-stable :

Lemme 6. *Étant donnés γ_0 et $U_\alpha(\gamma, \gamma_0)$ tels que définis dans la proposition 5. Pour $\alpha \neq 1$ et $\gamma_0 < \gamma < \frac{\pi}{2}$, X est une loi $\alpha-$stable $S_\alpha(1, \beta_2, 0)$ si et seulement si pour $x>0$:*

$$\frac{1}{\pi}\int_{\gamma_0}^{\frac{\pi}{2}} \exp\left[-x^{\frac{\alpha}{(\alpha-1)}} U_\alpha(\gamma, \gamma_0)\right] d\gamma = \begin{cases} P(0 < X < x), & si\, \alpha < 1 \\ P(X \geq x), & si\, \alpha > 1 \end{cases} \tag{2.2.17}$$

Théorème 7. *Étant donné γ_0 tel que défini dans la proposition 5. Soient γ et W 2 variables aléatoires (iid) uniformément distribuées sur $[-\frac{\pi}{2}, \frac{\pi}{2}]$ et exponentielle respectivement. Alors :*

$$\begin{cases} X = \frac{\sin\alpha(\gamma-\gamma_0)}{(\cos\gamma)^{1/\alpha}} \left(\frac{\cos(\gamma-\alpha(\gamma-\gamma_0))}{W}\right)^{(1-\alpha)/\alpha} \rightsquigarrow S_\alpha(1, \beta_2, 0), & si\, \alpha \neq 1 \\ X = (\frac{\pi}{2} + \beta_2\gamma)\tan\gamma - \beta_2 \log\left(\frac{W\cos\gamma}{\frac{\pi}{2}+\beta_2\gamma}\right) \rightsquigarrow S_\alpha(1, \beta_2, 0), & si\, \alpha = 1 \end{cases} \tag{2.2.18}$$

Les preuves de la proposition 5, du lemme 6 et du théorème 7 sont données dans [157].

A partir de ce théorème, on peut désormais facilement construire un simulateur de lois $\alpha-$stables (algorithme 2.1). Toutefois, les résultats esquissés dans ces formules ne permettent de traiter que le cas d'une loi α-stable standard $S_\alpha(1, \beta, 0)$. Cependant, en utilisant les propriétés ((2.2.13)) et ((2.2.14)) des loi α-stables, on arrive aisément à recouvrir les autres cas. L'algorithme résume le simulateur qui sera utilisé dans la suite de ce rapport.

2.3. Élaboration de notre estimateur non paramétrique des lois $\alpha - stables$

Dans cette section, nous donnons les outils fondamentaux nécessaires à la construction de l'algorithme proposé. Néanmoins, il convient de noter que cette liste d'outils n'est pas absolue et n'est pas exhaustive. On peut construire d'une autre manière un algorithme qui coïncide avec le squelette de notre approche : l'utilisation des métaheuristiques entre autres. En effet, nous tenons en premier lieu à souligner l'originalité

Algorithme 2.1 Simulateur de McCulloch des loi α-stables.

% on veut simuler N réalisations d'une loi $S_\alpha(\gamma,\beta,\mu)$.
1) **Pour** i allant de 1 à N **faire**
2) Générer une variable aléatoire uniforme .
3) Générer une variable aléatoire exponentielle W de moyenne 1.
4) **Si** $\alpha \neq 1$ **alors**
5) Calculer $B_{\alpha,\beta} = \frac{\arctan(\beta \tan \frac{\pi\alpha}{2})}{\alpha}$.
6) Calculer $S_{\alpha,\beta} = \left[1+\beta^2 \tan^2 \frac{\pi\alpha}{2}\right]^{1/(2\alpha)}$.
7) Calculer $X(i) = S_{\alpha,\beta} \times \frac{\sin(\alpha(V+B_{\alpha,\beta}))}{(\cos(V))^{1/\alpha}} \times \left(\frac{\cos(V-\alpha(V+B_{\alpha,\beta}))}{W}\right)^{(1-\alpha)/\alpha}$.
8) Calculer $X(i) = \gamma X(i) + \mu$.
9) **Sinon faire**
10) Calculer $X(i) = \frac{2}{\pi}\left[(\frac{\pi}{2}+\beta V)\tan V - \beta \log\left(\frac{W\cos V}{\frac{\pi}{2}+\beta V}\right)\right]$.
11) Calculer $X(i) = \gamma X(i) + \frac{2}{\pi}\beta\gamma\log\gamma + \mu$.
12) **Fin Si**
13)**Fin Pour**

Hypothèse H_0	Accepter H_0	Rejeter H_0
Vraie	pas d'erreur	erreur de $1^{ère}$ espèce
Fausse	erreur de $2^{ème}$ espèce	pas d'erreur

TABLE 2.2.: Types d'erreur dans les tests statistiques

de notre approche, par conséquent nous ne sommes pas intéressés à construire le meilleur algorithme utilisant le concept présenté dans ce chapitre. Notamment, l'exploitation de quelques propriétés des lois α-stables, le choix de l'OEP utilisé ou encore le réglage des paramètres de ce dernier peuvent faire l'objet d'une étude approfondie, afin de rendre l'algorithme le plus performant.

2.3.1. Tests statistiques

L'un des outils les plus puissants et les plus utilisés en statistique est le test d'hypothèse. Ce test est basé sur une *"statistique inférentielle"* calculée à partir d'un échantillon de référence afin de *rejeter* ou *ne pas rejeter* une hypothèse statistique en associant à la conclusion un risque d'erreur. En d'autres termes, les tests statistiques traitent des méthodes qui permettent de décider, au vu d'un échantillon, le choix entre deux hypothèses opposées, faites sur la population ou sur les paramètres. Un test statistique est basé sur les données d'une expérience et l'essence d'un test est une règle indiquant au décideur si les données recueillies le conduisent à rejeter ou accepter son hypothèse de départ. Résoudre un problème de test consiste donc à trouver la région de rejet, appelée aussi région critique de l'hypothèse. La décision à prendre (rejeter ou ne pas rejeter l'hypothèse) est liée à la structure de l'échantillon, qui est aléatoire. La décision prise est donc aléatoire (tableau 2.2).

Suivant leur finalité, les tests statistiques se décomposent en quatre classes principales :

- le test de conformité, qui valide des quantités statistiques, la moyenne et la variance par exemple ;
- le test d'adéquation (ou d'ajustement), qui ajuste une loi de probabilité à une autre distribution choisie a priori ;
- le test d'homogénéité (ou de comparaison), qui compare deux échantillons s'ils proviennent de la même population ;
- le test d'association (ou d'indépendance), qui teste l'indépendance de deux variables.

Dans notre problème d'estimation, on dispose d'un échantillon de référence et d'un générateur de loi stable (section 2.2.2), le problème revient donc à comparer cet échantillon de référence à un autre échantillon généré. On est donc en présence d'un test d'homogénéité. L'un des tests les plus utilisés est alors le test de "Kolmogorov Smirnov" [Mass51, Step70].

Le test de Kolmogorov-Smirnov est un test non paramétrique utilisé pour comparer un échantillon à une distribution de probabilité choisie a priori (dans ce cas-là, on parle de test d'adéquation) ou pour comparer deux échantillons distincts (dans ce cas-là, on parle de test d'homogénéité). Ce test non paramétrique est basé sur les propriétés empiriques de la distribution de probabilité de l'échantillon de référence.

Étant donné, $S_n = (x_1, ..., x_n)$, n réalisations d'une variable aléatoire de loi de probabilité f . La fonction de répartition empirique du n-échantillon S est donnée par l'équation (2.3.1) :

$$F_n(x) = \frac{1}{n}\sum_{i=1}^{n} \delta_{x_i \leq x} \tag{2.3.1}$$

avec :

$$\begin{cases} \delta_{x_i \leq x} = 1, & if\, x_i \leq x \\ \delta_{x_i \leq x} = 0, & sinon \end{cases} \tag{2.3.2}$$

En se référant à l'équation (2.3.1), la fonction de répartition empirique $F_n(x)$ décrit un processus qui prend ses valeurs dans l'ensemble des fonctions croissantes dans $[0, 1]$. En conséquence de cette caractéristique, on a la convergence suivante (équation (2.3.3)) :

$$\mathbb{P}\left[\sup_x |F_n(x) - F(x)| > \frac{c}{\sqrt{n}}\right] \overrightarrow{n \to \infty}\ \alpha(c) = 2\sum_{r=1}^{+\infty}(-1)^{r-1} e^{-2r^2c^2} \tag{2.3.3}$$

en désignant par $\alpha(c) \in]0, 1[$ la *p-value* (ou le quantile) du test statistique.

Au premier abord, l'idée principale de fonctionnement du test consiste à calculer :

- une statistique, appelée *la statistique de Kolmogorov-Smirnov* D_n (formule 2.3.4), qui représente la distance entre la fonction de répartition empirique $F_n(x)$ du n-échantillon S_n que l'on veut ajuster et la fonction de répartition théorique fixée a priori $F(x)$ (dans le cas d'un test d'adéquation, figure 2.3.1),

ou

- $D_{n,n'}$ (équation (2.3.5)) qui représente la distance entre les deux fonctions de répartition empiriques $F_n(x)$ et $F_{n'}(x)$ correspondant aux deux échantillons S_n et $S_{n'}$ respectivement (dans le cas d'un test homogénéité).

$$D_n = \sup_x |F_n(x) - F(x)| \tag{2.3.4}$$

$$D_{n,n'} = \sup_x |F_n(x) - F_{n'}(x)| \tag{2.3.5}$$

En général, les statistiques D_n et $D_{n,n'}$ sont calculées sous l'hypothèse nulle H_0 :

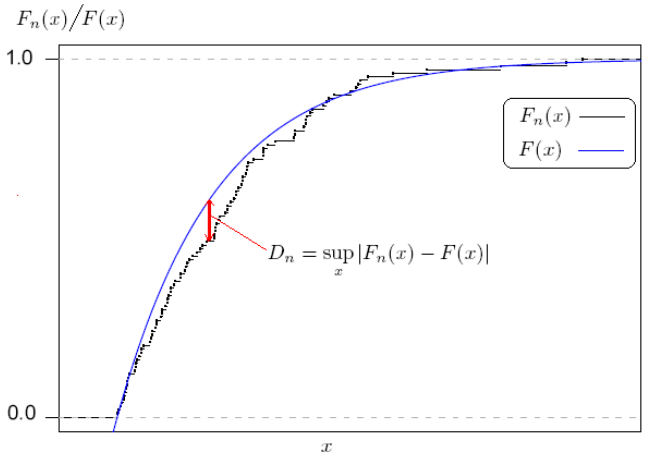


FIGURE 2.3.1.: Le concept basique du test de Kolmogorov-Smirnov

- H_0 : "Les deux échantillons S_n et $S_{n'}$ sont tirés d'une même distribution de probabilité" (dans le cas d'un test homogénéité),
ou
- H_0 : " L'échantillon S_n est tiré de la distribution de probabilité fixée a priori $F(x)$" (dans le cas d'un test d'ajustement).

De surcroît, l'hypothèse nulle H_0 est rejetée si :

- la statistique $\sqrt{n}D_n > c$ (dans le cas d'un test d'ajustement)

ou

- $\sqrt{\frac{nn'}{n+n'}}D_{n,n'} > c$ (dans le cas d'un test homogénéité)

où $c > 0$ est le quantile défini dans [Mill56] lu dans la table tabulée de Kolmogorov-Smirnov.

Dans les deux cas, les distributions de probabilité considérées dans l'hypothèse nulle H_0 sont supposées continues. Par ailleurs, aucune restriction n'est imposée.

2.3.2. Identification du problème d'optimisation & conception de l'estimateur non paramétrique

Étant donné $S_n = (x_1, ..., x_n)$, n réalisations indépendantes d'une variable aléatoire X de loi α-stable d'indice de stabilité α, de paramètre de positionnement μ, de paramètre d'asymétrie β et de paramètre d'échelle γ. On notera désormais la loi $S(\alpha, \beta, \gamma, \mu)$ au lieu de $S_\alpha(\gamma, \beta, \mu)$. On suppose que ces paramètres sont inconnus et que l'on a un générateur de loi α-stable (le générateur de McCulloch 2.2.2 est un bon générateur de départ). Dès lors, on veut construire un bon estimateur pour les paramètres de la variable aléatoire X de loi α-stable $S(\alpha, \beta, \gamma, \mu)$.

De ce fait, les variables décisionnelles du processus d'optimisation sont les paramètres α, β, γ et μ. Chaque particule est représentée par un vecteur de dimension 4.

Soient $\alpha^j(t)$, $\beta^j(t)$, $\gamma^j(t)$ et $\mu^j(t)$ respectivement, l'indice de stabilité α, le paramètre d'asymétrie β, le paramètre d'échelle γ et le paramètre de positionnement μ de la particule j à l'itération t.

L'idée principale de notre approche consiste à tester, à chaque itération t du processus d'optimisation et pour chaque particule j (c'est-à-dire $\alpha^j(t)$, $\beta^j(t)$, $\gamma^j(t)$ et $\mu^j(t)$ déterminés par l'algorithme d'optimisation), l'homogénéité des deux échantillons S_n et $S_n^j(t)$ correspondant respectivement à l'échantillon de référence, dont les paramètres sont à estimer et à l'échantillon tiré d'une loi α-stable de paramètres ($\alpha^j(t)$, $\beta^j(t)$, $\gamma^j(t)$ et $\mu^j(t)$), grâce au générateur de McCulloch.

En d'autres termes, le problème $Prob^1$ défini dans l'équation (2.1.1) devient :

$$Prob^2 \begin{cases} \min\limits_{\alpha,\beta,\gamma,\delta} f(\alpha,\beta,\gamma,\delta) = \underset{x}{Min} D_{n,n'} \\ = Min(\sup_x |F_n(x) - F_{n'}(x)|) \\ g_1 : \alpha \in]0,2], \\ g_2 : \beta \in [-1,1], \\ g_3 : \gamma > 0, \\ g_4 : \delta \in \mathbb{R} \end{cases} \tag{2.3.6}$$

en désignant par :
- $D_{n,n'}$, la fonction objectif de Kolmogorov-Smirnov à minimiser (*fitness* des particules)
- $F_n(x)$ et $F_{n'}(x)$ respectivement, les fonctions de répartition de : l'échantillon S_n de référence que l'on veut ajuster et de $S_n^j(t)$ renvoyé par le processus d'optimisation (utilisant le générateur de McCulloch).

A chaque itération t, les positions de la $j^{ème}$ particule sont mises à jour à l'aide des équations (2.3.8) et (2.3.7) pour évaluer les nouvelles performances (ou la *fitness*) de la $j^{ème}$ particule en calculant à nouveau la statistique de Kolmogorov-Smirnov $D_{n,n'}$. Pour plus d'informations sur la technique d'optimisation utilisée dans ce chapitre notamment, l'optimisation par essaim de particules, on se réfèrera aux chapitres I et III.

$$\overrightarrow{x_i}(t+1) = \overrightarrow{x_i}(t) + \overrightarrow{v_i}(t+1) \tag{2.3.7}$$

en désignant par $\overrightarrow{v_i}(t)$, la vitesse de la $i^{ème}$ particule à l'itération t (équation (2.3.8)) et k, $\varphi_1 = rand(0,1) * k$ et$\varphi_2 = rand(0,1) * k$ respectivement l'inertie et le degré de confiance des particules.

$$\begin{aligned} \overrightarrow{v_i}(t+1) &= k[\overrightarrow{v_i}(t) + rand(0,1) * (\overrightarrow{p_i} - \overrightarrow{x_i}(t)) \\ &\quad + rand(0,1) * (\overrightarrow{p_{best_i}} - \overrightarrow{x_i}(t))] \end{aligned} \tag{2.3.8}$$

avec :

$$k = \frac{2}{|2 - (c_1 + c_2) - \sqrt{(c_1 + c_2)^2 - 4(c_1 + c_2)}|} \tag{2.3.9}$$

et : $c_1 + c_2 > 4$.

Au final, le synoptique de l'approche proposée dans ce chapitre est résumé dans l'algorithme (2.2) et la figure 2.3.2 montre l'organigramme de l'algorithme proposé.

Algorithme 2.2 Estimateur des lois α-stables par OEP.

/* $\alpha^j(t)$, $\beta^j(t)$, $\gamma^j(t)$, *et* $\mu^j(t)$ *sont l'indice de stabilité, le paramètre d'asymétrie,*
/* *le paramètre d'échelle et le paramètre de positionnement de la particule* j
/**à itération* t.
/* $\hat{\alpha}$, $\hat{\beta}$, $\hat{\gamma}$, *et* $\hat{\mu}$ *sont les meilleurs paramètres estimés,*
/*$\overrightarrow{p_j}$ *est la meilleure position rencontrée par la* $j^{ème}$ *particule,*
/*$\overrightarrow{p_{best_j}}$ *est la meilleure position des voisins de la* $j^{ème}$ *particule.*
/*$\overrightarrow{v_j}(t)$ *est la vitesse de la* $j^{ème}$ *particule à l'itération* t,
/*t_{max} *est le nombre maximal d'itérations .*
/*$S^j_{n'}$*contient* n' *réalisations indépendantes d'une loi* α*-stable* $S(\alpha^j(t), \beta^j(t), \gamma^j(t),$
/*$\mu^j(t))$.
1) **Poser** $t \leftarrow 0$,
2) **Pour** chaque particule j **faire**
3) **Initialiser** $\alpha^j(t)$, $\beta^j(t)$, $\gamma^j(t)$, $\mu^j(t)$ et $\overrightarrow{v_j}(t)$
4) **Générer** un échantillon $S^j_{n'}$ à l'aide du générateur McCulloch
5) (algorithme 2.1) avec les paramètres ($\alpha^j(t)$, $\beta^j(t)$, $\gamma^j(t)$, $\mu^j(t)$)
6) **Évaluer** la $j^{ème}$ particule en calculant $D_{n,n'}$(formule 2.3.5)
7) **Poser** $\overrightarrow{p_j}$ comme meilleure position de la particule j rencontrée
8) jusqu'alors
9) **Fin Pour**
10) **Pour** chaque particule j **faire**
11) **poser** $\overrightarrow{p_{best_j}}$ comme la meilleure position des voisins de la particule j.
12) **Fin Pour**
13) **Poser** $\hat{\alpha}$, $\hat{\beta}$, $\hat{\gamma}$, et $\hat{\mu}$ comme la meilleure position (solution) de toutes les particules.
14) **Pour** t allant de 1 à t_{max} **faire**
15) **Pour** chaque particule j **faire**
16) **Mettre à jour** $\overrightarrow{v_j}(t)$ à l'aide de (2.3.8).
17) **Mettre à jour** $\alpha^j(t)$, $\beta^j(t)$, $\gamma^j(t)$, $\mu^j(t)$ à l'aide de (2.3.7).
18) **Mettre à jour** $\overrightarrow{p_j}$.
19) **Fin Pour**
20) **Mettre à jour** $\overrightarrow{p_{best_j}}$ pour toutes les particules.
21) **Mettre à jour** la meilleure estimation des paramètres $\hat{\alpha}$, $\hat{\beta}$, $\hat{\gamma}$,et $\hat{\mu}$
22) du signal original S_n.
23) **Fin Pour**
24) **Retourner** $\hat{\alpha}$, $\hat{\beta}$, $\hat{\gamma}$, et$\hat{\mu}$.
25) **Fin**.

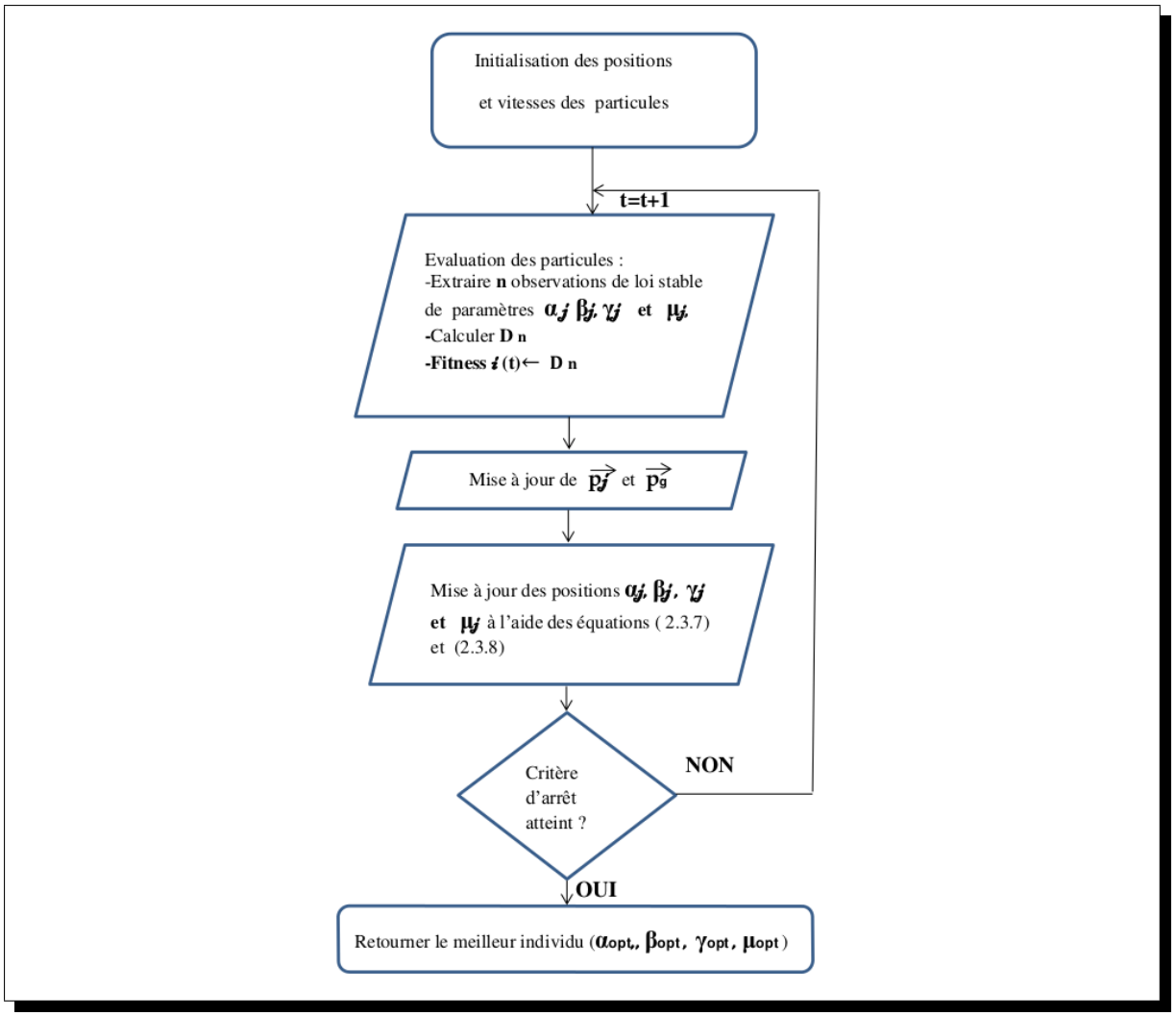


FIGURE 2.3.2.: Organigramme de l'estimateur des paramètres des lois α-stable par OEP.

2.4. Résultats et comparaison sur des *benchmarks*

2.4.1. Validation sur des *benchmarks*

Pour tester l'estimateur, nous avons considéré des *benchmarks* de lois α-stables notamment, en utilisant le générateur des lois $\alpha - stables$ de Mc Culloch décrit dans la section 2.2.2.

Afin d'observer le comportement de l'algorithme, nous l'avons testé sur l'ensemble des différents paramètres $\alpha \in [0.2, 0.5, 0.7, 1.3, 1.5, 1.8]$, $\beta \in [-0.5, 0, 0.5]$, $\gamma \in [1, 5]$, $\mu \in 0$ et sur l'ensemble des différentes tailles d'échantillons $n \in [250, 500, 5000, 10000]$. Les résultats serons comparés à l'estimateur de McCulloch [McCu86], estimateur considéré comme le plus pertinent.

Dans un premier temps, pour chaque ensemble de paramètres (α_i, β_i, γ_i, μ_i) - i étant le *ième benchmark* - on simule 1000 échantillons de même taille n_j et indépendants les uns des autres (iid) où chaque échantillon contient les réalisations de la variable aléatoire α-stable $B_i(\alpha_i, \beta_i, \gamma_i, \mu_i)$. Dans un deuxième temps, on estime, pour chaque échantillon i, les paramètres de la variable aléatoire B_i, à l'aide des deux estimateurs : le notre et celui de McCulloch. Ensuite, on calcule pour chaque *benchmark* i - i.e. les 1000 réalisations - la moyenne et l'écart type des estimations. Sachant que les paramètres à estimer sont connus

à l'avance, soit α_i, β_i, γ_i *et* μ_i, on est en mesure de calculer l'erreur des estimations, et par conséquent, de comparer notre estimateur à celui de McCulloch.

Les résultats des estimations pour chaque jeu de paramètres (α_i, β_i, γ_i, μ_i) et chaque taille d'échantillon n_j sont résumés dans les tableaux 2.3 à 2.10. Ceux ci sont, par convention, organisés en :

- trois principaux groupes avec les *benchmarks* de petite taille ($n = 250$ et $n = 500$), de taille moyenne ($n = 5000$) et de grande taille ($n = 10000$),
- en deux sous-groupes, en particulier, $\alpha < 1$ et $\alpha > 1$.

Dans tous les cas, les paramètres de l'algorithme sont fixés comme suit :

- le coefficient de constriction K défini dans l'équation (2.3.9) est réglé à : 0.7298
- la taille de l'essaim est réglée à : 24
- le nombre de voisins est réglé à : 3
- le nombre d'itérations est réglé à : 400
- l'espace de recherche est : $\alpha \in [0.1, 2]$, $\beta \in [-1, 1]$, $\gamma \in]0, 50]$, $\mu \in [-50, 50]$

Les tableaux 2.3 et 2.4 attestent de la pertinence de notre estimateur lorsque $\alpha \prec 2$ et en particulier, quand $\alpha < 1$ et ceci, pour les différentes valeurs de α, β, γ, μ et pour toutes les tailles d'échantillon n. En effet, bien que les tailles des échantillons n soient petites, les estimations données dans le tableau 2.3 sont très proches des valeurs réelles des paramètres.

Les estimations obtenues à l'aide de l'algorithme de McCulloch pour des valeurs de $\alpha < 1$ (tableaux 2.5 et 2.6) sont très mauvaises, en particulier, pour des valeurs de $\alpha < 0.5$. La méthode de McCulloch est conçue pour des valeurs de $\alpha > 0.6$. Pour des valeurs de $\alpha > 0.6$, la méthode de McCulloch donne des résultats intéressants, mais reste de moindre qualité par rapport à nos estimations. Seul le cas des échantillons symétriques ($\beta = 0$) donne des résultats à peu près similaires.

Les tableaux 2.7 et 2.8 montrent les résultats de notre estimateur pour différents jeux de paramètres α, β, γ, μ et différentes tailles d'échantillon n pour des valeurs de $\alpha > 1$ et les tableaux 2.9, 2.10 et montrent les mêmes résultats obtenus à l'aide de l'estimateur McCulloch.

On remarque que les résultats sont légèrement moins bons par rapport aux résultats donnés dans les tableaux 2.3, 2.4 et 2.5, 2.6, notamment pour les écarts-types. En effet, les écarts-types des deux méthodes donnés dans les tableaux 2.7, 2.8 et 2.9, 2.10 (le cas de $\alpha > 1$), même pour les différentes valeurs de n, sont plus grands que ceux donnés dans les tableaux 2.3, 2.4 et 2.5, 2.6 (cas de $\alpha < 1$).
Cependant, on remarque que notre algorithme donne une meilleure estimation du paramètre de symétrie β, notamment pour des échantillons de grande taille.

De manière générale, la résolution du problème d'optimisation décrit dans la section (2.3.2), notamment la minimisation de la statistique de Kolmogorov-Smirnov (équation (2.3.6)) par notre algorithme est dans tous les cas meilleure que celle de l'algorithme de McCulloch. En effet, la figure 2.4.1 montre que les *box-plots* de l'objectif optimisé sont largement en dessous du seuil d'acceptation et ce, pour toutes les valeurs de α, β, γ etμ et pour toute taille d'échantillon n. Au contraire, la figure (2.4.2) montre que les *box-plots* de la statistique de Kolmogorov-Smirnov calculés pour les estimations de McCulloch sont proches des frontières de rejet, en dépassant parfois même ce seuil.

Un autre point important à noter, au vu de ces résultats, est la convergence asymptotique des deux estimateurs (tableaux 2.4, 2.8, 2.6 et 2.10). La difficulté à estimer les quatre paramètres des lois $\alpha-$stables quand $\alpha > 1$ pour de petites tailles d'échantillon n réside dans le fait que la queue de distribution ne

α	β	γ	$\hat{\alpha}$	$\hat{\beta}$	$\hat{\gamma}$	$\hat{\mu}$	$\hat{\alpha}$	$\hat{\beta}$	$\hat{\gamma}$	$\hat{\mu}$
0.2	-0.5	1	0.20 (0.0150)	-0.49 (0.0659)	1.26 (0.9177)	0.00 (0.0048)	0.20 (0.0100)	-0.49 (0.0458)	1.08 (0.3176)	0.00 (0.0012)
		5	0.20 (0.0145)	-0.49 (0.0628)	5.78 (1.9554)	0.00 (0.0127)	0.20 (0.0107)	-0.49 (0.0451)	5.45 (1.5223)	0.00 (0.0059)
	0.0	1	0.20 (0.0148)	-0.00 (0.0724)	1.20 (0.8452)	0.00 (0.0039)	0.20 (0.0104)	-0.00 (0.0494)	1.09 (0.4644)	0.00 (0.0015)
		5	0.20 (0.0149)	-0.00 (0.0757)	5.71 (2.0174)	0.00 (0.0154)	0.20 (0.0102)	-0.00 (0.0502)	5.44 (1.5519)	0.00 (0.0067)
	0.5	1	0.20 (0.0138)	0.47 (0.0670)	1.21 (0.8178)	0.00 (0.0039)	0.20 (0.0104)	0.48 (0.0472)	1.08 (0.3033)	0.00 (0.0017)
		5	0.20 (0.0147)	0.48 (0.0653)	5.76 (1.9799)	0.00 (0.0157)	0.20 (0.0103)	0.48 (0.0444)	5.50 (1.5389)	0.00 (0.0064)
			n=250				n=500			
0.5	-0.5	1	0.51 (0.0947)	-0.50 (0.0850)	1.04 (0.1894)	-0.08 (0.8496)	0.50 (0.0360)	-0.50 (0.0547)	1.01 (0.1246)	0.01 (0.0949)
		5	0.50 (0.0384)	-0.50 (0.0728)	5.17 (0.8849)	0.03 (0.4231)	0.50 (0.0270)	-0.49 (0.0516)	5.07 (0.6108)	0.00 (0.2897)
	0.0	1	0.50 (0.0567)	-0.00 (0.0828)	1.02 (0.1667)	0.00 (0.1172)	0.50 (0.0272)	-0.00 (0.0569)	1.01 (0.1185)	0.00 (0.0505)
		5	0.50 (0.0522)	-0.00 (0.0835)	5.10 (0.8470)	0.00 (0.6470)	0.50 (0.0270)	-0.00 (0.0547)	5.02 (0.5678)	0.02 (0.2421)
	0.5	1	0.51 (0.0859)	0.49 (0.0893)	1.04 (0.1907)	0.04 (0.6642)	0.50 (0.0306)	0.48 (0.0575)	1.01 (0.1175)	0.00 (0.0812)
		5	0.50 (0.0374)	0.48 (0.0752)	5.18 (0.8672)	0.03 (0.4159)	0.50 (0.0266)	0.49 (0.0519)	5.11 (0.6745)	0.00 (0.2922)
			n=250				n=500			
0.7	-0.5	1	0.70 (0.0570)	-0.50 (0.0882)	1.00 (0.1212)	0.05 (0.3656)	0.70 (0.0546)	-0.50 (0.0625)	1.00 (0.0898)	0.02 (0.2943)
		5	0.70 (0.0645)	-0.50 (0.0900)	5.02 (0.6027)	0.21 (1.7238)	0.70 (0.0478)	-0.50 (0.0609)	5.01 (0.4433)	0.09 (1.2013)
	0.0	1	0.70 (0.0585)	0.00 (0.0893)	1.01 (0.1261)	0.01 (0.2065)	0.70 (0.0449)	-0.00 (0.0662)	1.00 (0.0846)	0.02 (0.4887)
		5	0.70 (0.0590)	-0.016 (0.0932)	5.07 (0.6219)	0.13 (1.0966)	0.70 (0.0415)	-0.00 (0.0631)	5.03 (0.4520)	0.05 (0.6811)
	0.5	1	0.70 (0.0591)	0.48 (0.0876)	1.01 (0.1214)	-0.06 (0.4738)	0.70 (0.0474)	0.49 (0.0662)	1.00 (0.0881)	-0.00 (0.2663)
		5	0.70 (0.0596)	0.48 (0.0885)	5.08 (0.6052)	-0.30 (1.8679)	0.70 (0.0450)	0.49 (0.0611)	5.02 (0.4284)	-0.11 (1.3148)

TABLE 2.3.: n = 250 et n=500, estimation $\alpha-$stable, 0.2 - 0.7

reste pas consistante (i.e. peu de variations), sachant que la pente de la distribution de probabilité en loglog caractérise l'indice de stabilité α des lois $\alpha-$stables.

2.4.2. Parallélisation du processus sur carte GP/GPU

L'algorithme que nous avons développé est par essence parallélisable. Nous avons souhaité l'implémenter afin d'estimer les gains en temps de calcul. Dans cette section, nous présentons directement la procédure utilisée pour réaliser l'implémentation de notre estimateur $\alpha-$stable sur carte GP/GPU.

La fonction "*kstest2.m*" de Matlab réalise un test d'hypothèse statistique de Kolmogorov-Smirnov à 2 séries indépendantes d'échantillons X_1 et X_2. Le résultat du test détermine si X_1 et X_2 proviennent de la même loi de distribution (booléen H). La série X_1 est la série de référence dont on veut déterminer les paramètres de la loi α-stable (ce peut être, par exemple, la distribution des pannes d'un réseau électrique). On suppose que cette dernière suit une loi α-stable. Les séries X_2 sont générées par les paramètres de la loi α-stable des particules de l'OEP. Cette fonction est la fonction objectif de l'OEP développée dans la section (2.3). L'idée initiale est de calculer en parallèle, de manière synchrone, pour toutes les particules, la fonction *kstest2 :*

$$[H,P,KSSTAT] = kstest2(X_1, X_2, ALPHA, TYPE)$$

avec, en entrée :
- X_1 série d'échantillons,
- X_2 série d'échantillons,
- *Alpha* : niveau de confiance,

α	β	γ	$\hat{\alpha}$	$\hat{\beta}$	$\hat{\gamma}$	$\hat{\mu}$	$\hat{\alpha}$	$\hat{\beta}$	$\hat{\gamma}$	$\hat{\mu}$
0.2	-0.5	1	0.20 (0.0036)	-0.49 (0.0159)	1.02 (0.0946)	0.00 (0.0002)	0.20 (0.0030)	-0.50 (0.0118)	1.01 (0.0828)	0.00 (0.0002)
		5	0.20 (0.0039)	-0.49 (0.0156)	5.17 (0.5255)	0.00 (0.0013)	0.20 (0.0030)	-0.49 (0.0124)	5.12 (0.4233)	0.00 (0.0010)
	0.0	1	0.20 (0.0036)	-0.00 (0.0168)	1.01 (0.0985)	0.00 (0.0002)	0.20 (0.0027)	0.00 (0.0120)	1.02 (0.0777)	0.00 (0.0002)
		5	0.20 (0.0038)	-0.00 (0.0167)	5.13 (0.5897)	0.00 (0.0012)	0.20 (0.0031)	0.00 (0.0129)	5.07 (0.3784)	0.00 (0.0009)
	0.5	1	0.20 (0.0035)	0.49 (0.0159)	1.01 (0.0943)	0.00 (0.0002)	0.20 (0.0025)	0.49 (0.0154)	1.02 (0.0839)	0.00 (0.0002)
		5	0.20 (0.0040)	0.49 (0.0160)	5.15 (0.5576)	0.00 (0.0013)	0.20 (0.0030)	0.49 (0.0126)	5.11 (0.4289)	0.00 (0.0012)
			n=5000				n=10000			
0.5	-0.5	1	0.50 (0.0111)	-0.49 (0.0201)	1.00 (0.0411)	0.00 (0.0234)	0.50 (0.0087)	-0.49 (0.0161)	1.00 (0.0339)	0.00 (0.0177)
		5	0.50 (0.0101)	-0.49 (0.0180)	5.02 (0.2021)	0.00 (0.0974)	0.49 (0.0081)	-0.49 (0.0149)	5.01 (0.1651)	0.00 (0.0759)
	0.0	1	0.50 (0.0101)	0.00 (0.0195)	1.00 (0.0409)	0.00 (0.0171)	0.50 (0.0084)	0.00 (0.0156)	1.00 (0.0316)	0.00 (0.0140)
		5	0.50 (0.0103)	0.00 (0.0199)	5.02 (0.2037)	0.00 (0.0848)	0.50 (0.0080)	0.00 (0.0151)	5.01 (0.1545)	0.00 (0.0637)
	0.5	1	0.50 (0.0116)	0.49 (0.0196)	1.00 (0.0425)	0.00 (0.0234)	0.50 (0.0086)	0.49 (0.0147)	1.00 (0.0317)	0.00 (0.0170)
		5	0.49 (0.0102)	0.49 (0.0187)	5.01 (0.1986)	0.00 (0.1003)	0.50 (0.0083)	0.49 (0.0152)	5.01 (0.1571)	0.00 (0.0836)
			n=5000				n=10000			
0.7	-0.5	1	0.70 (0.0220)	-0.49 (0.0259)	1.00 (0.0350)	0.01 (0.1270)	0.69 (0.0164)	-0.49 (0.0215)	1.00 (0.0289)	0.00 (0.0883)
		5	0.70 (0.0178)	-0.50 (0.0238)	5.01 (0.1673)	0.06 (0.4675)	0.70 (0.0157)	-0.49 (0.0205)	5.01 (0.1360)	0.02 (0.4150)
	0.0	1	0.70 (0.0179)	0.00 (0.0242)	1.00 (0.0344)	0.00 (0.0513)	0.70 (0.0156)	0.00 (0.0201)	1.00 (0.0277)	0.00 (0.0435)
		5	0.70 (0.0163)	0.00 (0.0225)	5.01 (0.1648)	0.00 (0.2378)	0.70 (0.0132)	0.00 (0.0196)	5.01 (0.1327)	0.00 (0.2044)
	0.5	1	0.70 (0.0204)	0.49 (0.0271)	1.00 (0.0341)	0.00 (0.1138)	0.69 (0.0173)	0.49 (0.0213)	1.00 (0.0285)	0.00 (0.0930)
		5	0.70 (0.0183)	0.49 (0.0245)	5.01 (0.1640)	0.01 (0.5005)	0.70 (0.0160)	0.49 (0.0208)	5.01 (0.1363)	0.04 (0.4201)

TABLE 2.4.: n = 5000 et n=10000, estimation $\alpha-$stable, 0.2 - 0.7

α	β	γ	$\hat{\alpha}$	$\hat{\beta}$	$\hat{\gamma}$	$\hat{\mu}$	$\hat{\alpha}$	$\hat{\beta}$	$\hat{\gamma}$	$\hat{\mu}$
0.2	-0.5	1	0.51 (0.0131)	-0.96 (0.1403)	8.46 (6.74)	19.2 (16.62)	0.51 (0.0068)	-0.99 (0.0671)	7.83 (4.69)	18.6 (11.53)
		5	0.51 (0.0136)	-0.96 (0.1492)	44.1 (34.37)	100 (85.5)	0.51 (0.0052)	-0.99 (0.0504)	39.2 (18.27)	94.2 (45.43)
	0.0	1	0.56 (0.0259)	0.01 (0.5768)	2.78 (2.0169)	-0.05 (2.3619)	0.57 (0.0191)	-0.05 (0.4278)	2.75 (1.3159)	0.17 (1.5591)
		5	0.56 (0.0258)	-0.01 (0.5872)	12.5 (8.927)	0.78 (11.277)	0.57 (0.0196)	0.00 (0.4452)	13.3 (6.424)	-0.13 (8.1083)
	0.5	1	0.51 (0.0101)	0.97 (0.1033)	9.05 (8.7198)	-20.3 (16.749)	0.51 (0.0045)	0.99 (0.0360)	7.15 (3.7307)	-17.2 (9.09)
		5	0.51 (0.0135)	0.96 (0.1621)	41.4 (29.28)	-93.8 (72.15)	0.51 (0.0057)	0.99 (0.0515)	36.6 (16.643)	-87.2 (41.36)
			n=250				n=500			
0.5	-0.5	1	0.55 (0.0382)	-0.73 (0.2653)	0.89 (0.3917)	0.44 (0.4138)	0.54 (0.0237)	-0.77 (0.2139)	0.80 (0.2613)	0.41 (0.2894)
		5	0.56 (0.0454)	-0.74 (0.2563)	4.37 (1.8325)	2.26 (2.0338)	0.54 (0.0217)	-0.77 (0.2120)	4.00 (1.4385)	1.98 (1.2949)
	0.0	1	0.58 (0.0135)	0.01 (0.2391)	1.01 (0.2387)	-0.01 (0.3383)	0.58 (0.0062)	0.00 (0.1543)	1.04 (0.1599)	-0.01 (0.2307)
		5	0.58 (0.0135)	-0.00 (0.2517)	4.96 (1.1959)	0.05 (1.6594)	0.58 (0.0054)	0.00 (0.1517)	5.19 (0.8011)	-0.03 (1.1531)
	0.5	1	0.55 (0.0429)	0.74 (0.2561)	0.88 (0.3613)	-0.46 (0.4178)	0.54 (0.0260)	0.76 (0.2116)	0.81 (0.2884)	-0.39 (0.2432)
		5	0.55 (0.0388)	0.74 (0.2692)	4.47 (2.0433)	-2.18 (1.9647)	0.55 (0.0248)	0.74 (0.2159)	4.20 (1.4839)	-1.95 (1.3542)
			n=250				n=500			
0.7	-0.5	1	0.71 (0.0849)	-0.59 (0.2133)	0.93 (0.2504)	0.31 (0.9265)	0.71 (0.0659)	-0.58 (0.1663)	0.93 (0.2190)	0.23 (0.6952)
		5	0.71 (0.0884)	-0.58 (0.2149)	4.65 (1.3515)	1.65 (4.9961)	0.71 (0.0608)	-0.58 (0.1667)	4.69 (1.0349)	1.07 (2.1052)
	0.0	1	0.70 (0.0711)	-0.01 (0.1743)	0.96 (0.1616)	0.02 (0.3709)	0.70 (0.0494)	-0.00 (0.1242)	0.95 (0.1043)	0.01 (0.2638)
		5	0.70 (0.0699)	0.01 (0.1828)	4.79 (0.8228)	-0.13 (1.9261)	0.70 (0.0442)	0.00 (0.1289)	4.84 (0.5662)	-0.10 (1.3533)
	0.5	1	0.71 (0.0886)	0.59 (0.2199)	0.94 (0.2583)	-0.34 (0.9045)	0.70 (0.0640)	0.58 (0.1663)	0.92 (0.2077)	-0.20 (0.5451)
		5	0.71 (0.0855)	0.59 (0.2120)	4.62 (1.3248)	-1.72 (5.1308)	0.70 (0.0644)	0.57 (0.1719)	4.71 (1.0425)	-0.87 (2.4869)

TABLE 2.5.: n = 250 et n=500, estimation Mc Culloch, 0.2 - 0.7

α	β	γ	$\hat{\alpha}$	$\hat{\beta}$	$\hat{\gamma}$	$\hat{\mu}$	$\hat{\alpha}$	$\hat{\beta}$	$\hat{\gamma}$	$\hat{\mu}$
0.2	-0.5	1	0.51 (0.0005)	-1.00 (0.000)	6.54 (0.977)	15.8 (2.417)	0.51 (0.0003)	-1.00 (0.000)	6.55 (0.699)	15.8 (1.727)
		5	0.51 (0.0005)	-1.00 (0.000)	32.9 (4.835)	79.8 (11.949)	0.51 (0.0003)	-1.00 (0.000)	32.60 (3.326)	78.9 (8.215)
	0.0	1	0.58 (0.0037)	0.00 (0.129)	2.93 (0.337)	-0.01 (0.585)	0.58 (0.0024)	0.00 (0.090)	2.97 (0.224)	-0.00 (0.421)
		5	0.58 (0.0039)	-0.00 (0.132)	14.6 (1.685)	0.15 (2.969)	0.58 (0.0022)	-0.00 (0.088)	14.8 (1.173)	0.07 (2.065)
	0.5	1	0.51 (0.0005)	1.00 (0.000)	6.55 (0.953)	-15.8 (2.356)	0.51 (0.0003)	1.00 (0.000)	6.52 (0.675)	-15.7 (1.664)
		5	0.51 (0.0005)	1.00 (0.000)	33.0 (5.016)	-80.0 (12.411)	0.51 (0.0003)	1.00 (0.000)	32.9 (3.418)	-79.7 (8.442)
			n=5000				n=10000			
0.5	-0.5	1	0.54 (0.0049)	-0.81 (0.0822)	0.70 (0.0818)	0.38 (0.0680)	0.54 (0.0033)	-0.81 (0.0554)	0.70 (0.0550)	0.39 (0.0457)
		5	0.54 (0.0052)	-0.81 (0.0878)	3.55 (0.4408)	1.95 (0.3552)	0.54 (0.0032)	-0.81 (0.0545)	3.53 (0.2760)	1.97 (0.2272)
	0.0	1	0.59 (0.0014)	-0.00 (0.0502)	1.07 (0.0512)	0.00 (0.0828)	0.59 (0.0011)	-0.00 (0.0375)	1.07 (0.0351)	0.00 (0.0626)
		5	0.59 (0.0014)	0.00 (0.0525)	5.32 (0.2381)	-0.00 (0.4327)	0.59 (0.0011)	-0.00 (0.0375)	5.36 (0.1805)	0.00 (0.3155)
	0.5	1	0.54 (0.0051)	0.80 (0.0863)	0.71 (0.0877)	-0.38 (0.0694)	0.54 (0.0034)	0.81 (0.0573)	0.70 (0.0566)	-0.39 (0.0453)
		5	0.54 (0.0048)	0.80 (0.0800)	3.56 (0.3993)	-1.94 (0.3348)	0.54 (0.0032)	0.81 (0.0539)	3.51 (0.2765)	-1.96 (0.2245)
			n=5000				n=10000			
0.7	-0.5	1	0.71 (0.0211)	-0.54 (0.0596)	0.96 (0.0768)	0.13 (0.1268)	0.71 (0.0145)	-0.53 (0.0423)	0.97 (0.0523)	0.12 (0.0901)
		5	0.71 (0.0204)	-0.54 (0.0614)	4.83 (0.3694)	0.67 (0.5940)	0.71 (0.0145)	-0.53 (0.0437)	4.86 (0.2689)	0.62 (0.4311)
	0.0	1	0.70 (0.0156)	0.00 (0.0412)	0.98 (0.0316)	-0.00 (0.0901)	0.70 (0.0119)	-0.00 (0.0312)	0.98 (0.0250)	0.00 (0.0682)
		5	0.70 (0.0167)	-0.00 (0.0434)	4.92 (0.1756)	0.01 (0.4728)	0.70 (0.0114)	-0.00 (0.0294)	4.94 (0.1200)	-0.00 (0.3203)
	0.5	1	0.71 (0.0211)	0.54 (0.0573)	0.97 (0.0734)	-0.14 (0.1311)	0.71 (0.0147)	0.53 (0.0396)	0.97 (0.0520)	-0.12 (0.0909)
		5	0.71 (0.0202)	0.54 (0.0587)	4.82 (0.3624)	-0.60 (0.6217)	0.71 (0.0145)	0.53 (0.0433)	4.86 (0.2725)	-0.60 (0.4599)

TABLE 2.6.: n = 5000 et 10000, estimation Mc Culloch, 0.2 - 0.7

α	β	γ	$\hat{\alpha}$	$\hat{\beta}$	$\hat{\gamma}$	$\hat{\mu}$	$\hat{\alpha}$	$\hat{\beta}$	$\hat{\gamma}$	$\hat{\mu}$
1.3	-0.5	1	1.395 (0.150)	-0.557 (0.258)	1.017 (0.079)	0.081 (0.434)	1.373 (0.119)	-0.539 (0.1928)	1.013 (0.0627)	0.086 (0.3036)
		5	1.360 (0.129)	-0.572 (0.205)	5.037 (0.414)	0.105 (2.252)	1.345 (0.090)	-0.543 (0.1434)	5.038 (0.2902)	0.136 (1.6287)
	0.0	1	1.373 (0.171)	-0.013 (0.229)	1.008 (0.084)	-0.012 (0.277)	1.353 (0.138)	-0.010 (0.1745)	1.008 (0.0645)	-0.018 (0.1911)
		5	1.345 (0.133)	-0.011 (0.195)	4.992 (0.408)	-0.132 (1.551)	1.321 (0.089)	-0.013 (0.1225)	5.003 (0.2925)	-0.113 (1.0061)
	0.5	1	1.387 (0.138)	0.542 (0.232)	1.012 (0.084)	-0.135 (0.372)	1.366 (0.116)	0.529 (0.1939)	1.012 (0.0638)	-0.097 (0.2994)
		5	1.359 (0.125)	0.538 (0.211)	5.042 (0.389)	-0.390 (2.033)	1.347 (0.092)	0.526 (0.1493)	5.045 (0.2893)	-0.343 (1.4487)
			n=250				n=500			
1.5	-0.5	1	1.543 (0.148)	-0.571 (0.272)	1.003 (0.074)	-0.023 (0.241)	1.540 (0.113)	-0.553 (0.2283)	1.002 (0.0557)	-0.001 (0.1666)
		5	1.530 (0.141)	-0.567 (0.274)	4.970 (0.373)	-0.162 (1.303)	1.534 (0.112)	-0.559 (0.1968)	5.011 (0.2824)	-0.031 (0.8027)
	0.0	1	1.557 (0.168)	0.000 (0.325)	1.003 (0.077)	-0.016 (0.197)	1.546 (0.130)	-0.014 (0.2080)	1.005 (0.0596)	-0.012 (0.1330)
		5	1.543 (0.166)	-0.021 (0.322)	4.992 (0.390)	-0.106 (1.018)	1.526 (0.114)	-0.012 (0.1823)	4.994 (0.2682)	-0.058 (0.6505)
	0.5	1	1.545 (0.143)	0.514 (0.290)	1.001 (0.078)	-0.036 (0.224)	1.534 (0.115)	0.525 (0.2354)	1.001 (0.0556)	-0.022 (0.1577)
		5	1.527 (0.137)	0.520 (0.285)	5.009 (0.375)	-0.082 (1.138)	1.525 (0.106)	0.524 (0.2013)	4.998 (0.2775)	-0.124 (0.7442)
			n=250				n=500			
1.8	-0.5	1	1.801 (0.139)	-0.345 (0.543)	0.9938 (0.064)	-0.011 (0.151)	1.800 (0.117)	-0.421 (0.4453)	0.995 (0.0512)	-0.003 (0.1030)
		5	1.792 (0.138)	-0.394 (0.524)	4.9609 (0.335)	-0.058 (0.763)	1.811 (0.116)	-0.468 (0.4117)	4.986 (0.2439)	-0.002 (0.5173)
	0.0	1	1.807 (0.136)	-0.028 (0.556)	0.9935 (0.068)	-0.021 (0.142)	1.839 (0.118)	0.001 (0.5093)	1.001 (0.0516)	-0.010 (0.0925)
		5	1.811 (0.137)	-0.072 (0.592)	4.9551 (0.340)	-0.141 (0.742)	1.832 (0.114)	-0.047 (0.4960)	5.008 (0.2352)	-0.043 (0.4690)
	0.5	1	1.797 (0.135)	0.247 (0.533)	0.9972 (0.065)	-0.042 (0.138)	1.808 (0.112)	0.359 (0.4740)	0.999 (0.0500)	-0.025 (0.0976)
		5	1.797 (0.133)	0.2895 (0.530)	4.9690 (0.336)	-0.201 (0.693)	1.803 (0.109)	0.395 (0.4451)	4.991 (0.2464)	-0.103 (0.5062)

TABLE 2.7.: n = 250 et n=500, estimation $\alpha-$stable, 1.3 - 1.8

α	β	γ	$\hat{\alpha}$	$\hat{\beta}$	$\hat{\gamma}$	$\hat{\mu}$	$\hat{\alpha}$	$\hat{\beta}$	$\hat{\gamma}$	$\hat{\mu}$
1.3	-0.5	1	1.333 (0.0471)	-0.517 (0.0584)	1.0083 (0.0232)	0.064 (0.1292)	1.328 (0.0385)	-0.511 (0.0473)	1.007 (0.0187)	0.063 (0.1055)
		5	1.326 (0.0393)	-0.515 (0.0516)	5.033 (0.1083)	0.234 (0.5904)	1.320 (0.0349)	-0.510 (0.0446)	5.023 (0.0945)	0.215 (0.5007)
	0.0	1	1.311 (0.0394)	0.001 (0.0441)	1.004 (0.0228)	-0.000 (0.0728)	1.310 (0.0326)	0.000 (0.0362)	1.003 (0.0180)	0.000 (0.0601)
		5	1.308 (0.0355)	0.000 (0.0424)	5.013 (0.1057)	-0.008 (0.3670)	1.306 (0.0307)	0.001 (0.0356)	5.012 (0.0861)	0.010 (0.2987)
	0.5	1	1.331 (0.0462)	0.517 (0.0574)	1.006 (0.0234)	-0.059 (0.1269)	1.328 (0.0406)	0.513 (0.0492)	1.008 (0.0198)	-0.057 (0.1151)
		5	1.324 (0.0412)	0.514 (0.0556)	5.029 (0.1122)	-0.215 (0.6159)	1.320 (0.0342)	0.509 (0.0445)	5.023 (0.0905)	-0.214 (0.5044)
			n=5000				n=10000			
1.5	-0.5	1	1.520 (0.0488)	-0.523 (0.0770)	1.004 (0.0201)	0.007 (0.0581)	1.516 (0.0406)	-0.512 (0.0611)	1.003 (0.0163)	0.011 (0.0475)
		5	1.517 (0.0419)	-0.521 (0.0708)	5.017 (0.0999)	0.032 (0.2701)	1.513 (0.0363)	-0.515 (0.0574)	5.014 (0.0786)	0.026 (0.2285)
	0.0	1	1.513 (0.0449)	-0.000 (0.0621)	1.003 (0.0204)	-0.000 (0.0474)	1.509 (0.0347)	-0.002 (0.0503)	1.002 (0.0165)	-0.002 (0.0375)
		5	1.510 (0.0411)	0.002 (0.0585)	5.013 (0.0982)	0.001 (0.2296)	1.508 (0.0331)	-0.001 (0.0492)	5.009 (0.0780)	-0.004 (0.1930)
	0.5	1	1.518 (0.0475)	0.514 (0.0806)	1.004 (0.0200)	-0.009 (0.0590)	1.516 (0.0412)	0.514 (0.0607)	1.004 (0.0169)	-0.008 (0.0485)
		5	1.515 (0.0446)	0.516 (0.0736)	5.015 (0.0966)	-0.037 (0.2742)	1.514 (0.0357)	0.513 (0.0566)	5.014 (0.0764)	-0.043 (0.2343)
			n=5000				n=10000			
1.8	-0.5	1	1.818 (0.0554)	-0.532 (0.2453)	1.002 (0.0185)	0.005 (0.0344)	1.813 (0.0462)	-0.542 (0.1949)	1.002 (0.0149)	0.004 (0.0275)
		5	1.815 (0.0504)	-0.563 (0.1971)	5.010 (0.0895)	0.014 (0.1673)	1.816 (0.0415)	-0.563 (0.1672)	5.012 (0.0714)	0.010 (0.1319)
	0.0	1	1.840 (0.0674)	-0.001 (0.2554)	1.006 (0.0189)	0.000 (0.0297)	1.836 (0.0569)	0.003 (0.1988)	1.006 (0.0164)	-0.001 (0.0264)
		5	1.833 (0.0612)	0.008 (0.2028)	5.029 (0.0962)	0.003 (0.1537)	1.824 (0.0483)	-0.000 (0.1384)	5.025 (0.0745)	-0.000 (0.1250)
	0.5	1	1.818 (0.0567)	0.545 (0.2692)	1.003 (0.0197)	-0.005 (0.0336)	1.813 (0.0461)	0.535 (0.1902)	1.001 (0.0144)	-0.005 (0.0263)
		5	1.813 (0.0504)	0.556 (0.2004)	5.012 (0.0877)	-0.013 (0.1644)	1.815 (0.0414)	0.560 (0.1710)	5.015 (0.0705)	-0.015 (0.1306)

TABLE 2.8.: n = 5000 et n=10000, estimation $\alpha-$stable, 1.3 - 1.8

α	β	γ	$\hat{\alpha}$	$\hat{\beta}$	$\hat{\gamma}$	$\hat{\mu}$	$\hat{\alpha}$	$\hat{\beta}$	$\hat{\gamma}$	$\hat{\mu}$
1.3	-0.5	1	1.317 (0.1339)	-0.545 (0.1906)	0.991 (0.0904)	-0.135 (0.8843)	1.305 (0.0891)	-0.524 (0.1262)	0.988 (0.0698)	-0.112 (0.4938)
		5	1.314 (0.1424)	-0.551 (0.1826)	4.968 (0.5083)	-1.020 (3.7072)	1.306 (0.0917)	-0.541 (0.1403)	4.961 (0.3475)	-0.636 (2.6845)
	0.0	1	1.304 (0.1135)	-0.007 (0.1984)	0.993 (0.0901)	-0.014 (0.4547)	1.303 (0.0797)	-0.006 (0.1259)	0.998 (0.0624)	-0.006 (0.2808)
		5	1.314 (0.1162)	0.015 (0.1886)	4.963 (0.4098)	0.116 (2.2671)	1.302 (0.0738)	-0.006 (0.1349)	4.966 (0.2943)	-0.047 (1.4243)
	0.5	1	1.308 (0.1305)	0.545 (0.1707)	0.983 (0.0941)	0.171 (0.8818)	1.310 (0.0947)	0.534 (0.1262)	0.990 (0.0644)	0.099 (0.4584)
		5	1.306 (0.1357)	0.519 (0.1818)	4.941 (0.4687)	0.745 (4.3048)	1.304 (0.0920)	0.526 (0.1287)	4.970 (0.3406)	0.569 (2.5334)
			n=250				n=500			
1.5	-0.5	1	1.525 (0.1452)	-0.579 (0.2481)	0.992 (0.0866)	-0.042 (0.2473)	1.505 (0.1005)	-0.545 (0.1781)	0.993 (0.0579)	-0.027 (0.1696)
		5	1.502 (0.1582)	-0.554 (0.2526)	4.927 (0.4251)	-0.245 (1.3573)	1.512 (0.1093)	-0.575 (0.1879)	4.983 (0.2943)	-0.215 (0.8817)
	0.0	1	1.508 (0.1386)	0.015 (0.2768)	0.999 (0.0786)	0.001 (0.2290)	1.502 (0.0923)	-0.006 (0.1710)	0.992 (0.0579)	-0.000 (0.1446)
		5	1.515 (0.1465)	-0.007 (0.2891)	4.984 (0.4197)	-0.057 (1.0431)	1.506 (0.0898)	-0.006 (0.1677)	4.998 (0.2882)	-0.049 (0.7007)
	0.5	1	1.512 (0.1456)	0.564 (0.2479)	0.993 (0.0838)	0.047 (0.2459)	1.514 (0.1081)	0.566 (0.1949)	0.999 (0.0617)	0.031 (0.1769)
		5	1.523 (0.1516)	0.583 (0.2418)	4.977 (0.4348)	0.296 (1.3701)	1.513 (0.1000)	0.549 (0.1814)	5.000 (0.2843)	0.142 (0.7790)
			n=250				n=500			
1.8	-0.5	1	1.814 (0.1524)	-0.518 (0.4560)	1.003 (0.0824)	0.010 (0.1350)	1.815 (0.1191)	-0.559 (0.3724)	0.998 (0.0566)	0.003 (0.1011)
		5	1.810 (0.1595)	-0.458 (0.4969)	5.010 (0.3936)	0.043 (0.7217)	1.823 (0.1280)	-0.601 (0.3530)	5.008 (0.2700)	0.018 (0.5138)
	0.0	1	1.814 (0.1488)	0.039 (0.5932)	1.005 (0.0782)	-0.004 (0.1307)	1.809 (0.1255)	0.036 (0.4558)	0.995 (0.0593)	-0.000 (0.0884)
		5	1.822 (0.1591)	-0.010 (0.5706)	5.036 (0.4100)	0.049 (0.6431)	1.809 (0.1225)	0.033 (0.4797)	5.008 (0.2850)	0.005 (0.4810)
	0.5	1	1.813 (0.1546)	0.444 (0.5020)	0.996 (0.0807)	-0.016 (0.1366)	1.802 (0.1184)	0.537 (0.3828)	0.995 (0.0551)	-0.002 (0.0987)
		5	1.814 (0.1490)	0.467 (0.5021)	5.019 (0.3878)	-0.045 (0.7065)	1.820 (0.1223)	0.575 (0.3795)	5.010 (0.2600)	-0.026 (0.5314)

TABLE 2.9.: n = 250 et n=500, estimation Mc Culloch, 1.3 - 1.8

α	β	γ	$\hat{\alpha}$	$\hat{\beta}$	$\hat{\gamma}$	$\hat{\mu}$	$\hat{\alpha}$	$\hat{\beta}$	$\hat{\gamma}$	$\hat{\mu}$
1.3	-0.5	1	1.303 (0.0300)	-0.533 (0.0402)	0.991 (0.0217)	-0.041 (0.1207)	1.306 (0.0217)	-0.533 (0.0290)	0.992 (0.0150)	-0.027 (0.0861)
		5	1.306 (0.0305)	-0.533 (0.0406)	4.958 (0.1058)	-0.149 (0.5967)	1.305 (0.0208)	-0.533 (0.0294)	4.959 (0.0722)	-0.142 (0.4197)
	0.0	1	1.303 (0.0239)	0.000 (0.0427)	0.997 (0.0188)	0.001 (0.0768)	1.303 (0.0173)	-0.000 (0.0309)	0.998 (0.0133)	-0.001 (0.0553)
		5	1.303 (0.0242)	0.002 (0.0427)	4.983 (0.0958)	0.015 (0.3818)	1.303 (0.0181)	0.000 (0.0303)	4.990 (0.0692)	0.002 (0.2693)
	0.5	1	1.305 (0.0292)	0.533 (0.0394)	0.992 (0.0199)	0.032 (0.1184)	1.305 (0.0211)	0.531 (0.0277)	0.991 (0.0152)	0.028 (0.0859)
		5	1.305 (0.0302)	0.532 (0.0399)	4.965 (0.1122)	0.179 (0.6173)	1.305 (0.0216)	0.533 (0.0287)	4.959 (0.0747)	0.150 (0.4249)
			n=5000				n=10000			
1.5	-0.5	1	1.503 (0.0339)	-0.525 (0.0548)	0.995 (0.0186)	-0.013 (0.0517)	1.503 (0.0239)	-0.526 (0.0385)	0.996 (0.0133)	-0.013 (0.0357)
		5	1.503 (0.0329)	-0.529 (0.0542)	4.978 (0.0905)	0.081 (0.2584)	1.503 (0.0232)	-0.529 (0.0391)	4.982 (0.0645)	-0.071 (0.1787)
	0.0	1	1.501 (0.0290)	-0.001 (0.0502)	0.997 (0.0183)	-0.004 (0.0426)	1.504 (0.0210)	0.000 (0.0351)	0.999 (0.0131)	-0.000 (0.0316)
		5	1.504 (0.0289)	-0.000 (0.0513)	4.992 (0.0903)	0.002 (0.2182)	1.502 (0.0201)	-0.000 (0.0359)	4.991 (0.0647)	-0.004 (0.1573)
	0.5	1	1.504 (0.0321)	0.529 (0.0557)	0.996 (0.0180)	0.014 (0.0490)	1.502 (0.0233)	0.525 (0.0372)	0.995 (0.0125)	0.014 (0.0363)
		5	1.503 (0.0317)	0.530 (0.0531)	4.984 (0.0910)	0.081 (0.2376)	1.502 (0.0242)	0.527 (0.0403)	4.979 (0.0642)	0.074 (0.1822)
			n=5000				n=10000			
1.8	-0.5	1	1.806 (0.0443)	-0.602 (0.1879)	0.998 (0.0168)	-0.010 (0.0278)	1.805 (0.0311)	-0.591 (0.1371)	0.998 (0.0120)	-0.011 (0.0203)
		5	1.807 (0.0475)	-0.610 (0.1955)	4.995 (0.0935)	-0.055 (0.1402)	1.803 (0.0317)	-0.582 (0.1422)	4.991 (0.0605)	-0.056 (0.1010)
	0.0	1	1.804 (0.0431)	0.003 (0.1335)	1.000 (0.0186)	0.000 (0.0288)	1.801 (0.0305)	-0.000 (0.0908)	0.998 (0.0127)	0.000 (0.0207)
		5	1.802 (0.0433)	-0.002 (0.1352)	4.994 (0.0866)	-0.006 (0.1462)	1.803 (0.0314)	0.002 (0.0984)	4.996 (0.0640)	0.002 (0.1054)
	0.5	1	1.805 (0.0455)	0.597 (0.1905)	0.998 (0.0176)	0.007 (0.0296)	1.804 (0.0312)	0.587 (0.1434)	0.998 (0.0119)	0.011 (0.0205)
		5	1.803 (0.0438)	0.589 (0.1857)	4.991 (0.0854)	0.054 (0.1539)	1.803 (0.0317)	0.579 (0.1350)	4.990 (0.0615)	0.053 (0.0991)

TABLE 2.10.: n = 5000 et n=10000, estimation Mc Culloch, 1.3 - 1.8

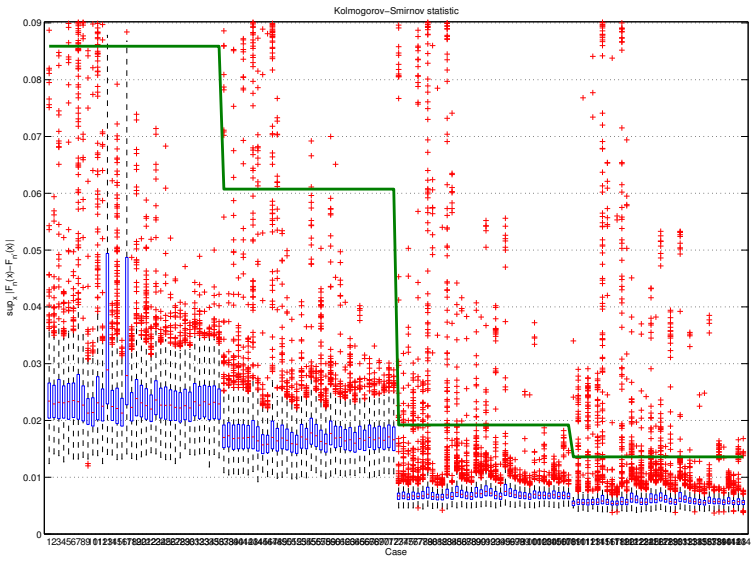


FIGURE 2.4.1.: Application de l'algorithme au *benchmark* avec : $\alpha = 0.2$, $\beta = [-0.5, 0, 0.5]$, $\gamma = [1, 5]$, $\mu = 0$.

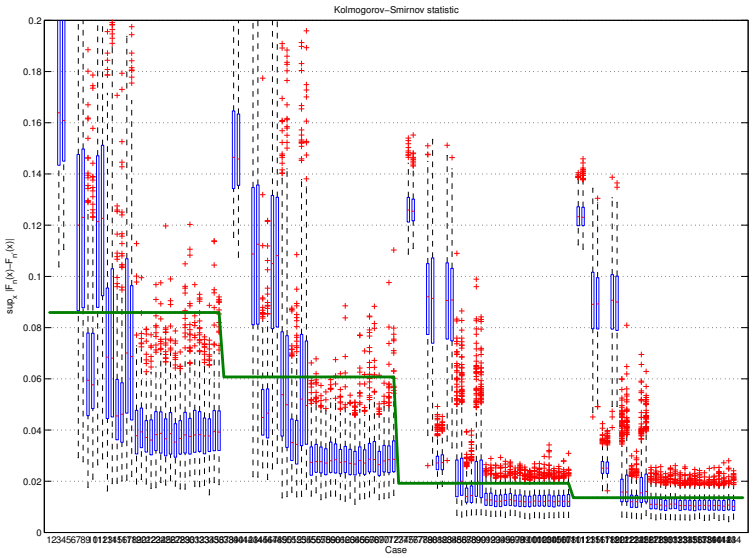


FIGURE 2.4.2.: Application de l'algorithme au *benchmark* avec : $\alpha = 0.2$, $\beta = [-0.5, 0, 0.5]$, $\gamma = [1, 5]$, $\mu = 0$.

– *Type* : 'unequal', 'larger', and 'smaller',

et en sortie :
– $KSSTAT$: statistiques du test,
– P : valeur asymptotique,
– H : un nombre binaire d'acceptation ou de rejet de l'hypothèse.

L'analyse du code Matlab *kstest2.m* montre que cette fonction peut être décomposée séquentiellement en tâches parallélisables :

1. Tri,
2. Histogrammes,
3. Sommes cumulatives,
4. Détermination de la valeur maximale d'une série.

Ces dernières sont codées dans un langage proche du C sous forme de *kernels* (noyaux) que l'on va pouvoir compiler et ensuite télécharger sur la cible (GPU) à partir du programme Python exécuté sur l'hôte (CPU). Il ne s'agit pas, d'un point de vue algorithmique, d'une simple traduction des algorithmes séquentiels existants pour ces opérations. Il est nécessaire de trouver ou de développer des algorithmes les plus intrinsèquement parallèles possible. Il s'agit du premier niveau de parallélisme.

Le deuxième niveau de parallélisme concerne les données suivant le paradigme logiciel SIMD (*Single Instruction Multiple Data*) qui est le paradigme utilisé par les GPU. En effet, la même instruction *kstest2* s'applique aux données des particules de OEP. Afin d'utiliser ce niveau de parallélisme, les séries générées par les paramètres (loi α-stable) des particules sont concaténées dans un même tableau. Le traitement de ce dernier est alors réalisé de manière synchrone. Un mécanisme de bas niveau sous-jacent permet de répartir ces tâches sur les SM (*Streaming Processor*) du GPU suivant un principe de file d'attente

(“*queueing*”).

Ces opérations de bas niveau (compilation à l'exécution et téléchargement sur la cible) sont réalisées par les commandes avancées du module Python pyOpenCL et ont lieu au tout début du programme. Le module pyOpenCL réalise l'interfaçage entre Python et OpenCL (standard ouvert de spécifications concernant la programmation et le calcul parallèle sur des plateformes hétérogènes). Le code Python est exécuté par le CPU (hôte), les *kernels* compilés et transférés sont exécutés par le GPU (cible).

Il est impératif de limiter au maximum les transferts de données entre l'hôte et la cible. Pour cela, il convient d'allouer des ressources mémoire a priori dans le GPU à l'initialisation du programme. Dans le cas de la fonction *kstest*2, les variables auxquelles il faut allouer un espace mémoire dans le GPU sont :

- les séries X_1 et X_2,
- les paramètres α_i, β_i, γ_i et μ_i déterminant les paramètres de la distribution de la série X_2 pour la particules i,
- des tableaux de stockage et de traitement intermédiaires nécessaires pour les algorithmes de tri, d'histogramme et de somme cumulative.

Les tableaux 2.11 et 2.12 donnent respectivement la description des fichiers Python et OpenCL utilisés pour les opérations précédentes.

Fichiers	*Descriptions*
copy.cl	*Mise en forme des allocations mémoire dans le GPU (copie)*
fill.cl	*Mise en forme des allocations mémoire dans le GPU (remplissage)*
stab_rnd1.cl	*Générateur de nombres aléatoires suivant une loi alpha-stable*
stab_rnd2.cl	*Générateur de nombres aléatoires suivant une loi alpha-stable*
pyopencl-ranluxcl.cl	*Générateur de nombres aléatoires suivant une loi uniforme*
radix.cl	*Algorithme de tri “radix” modifié (origine Nvidia)*
histogram.cl	*Algorithme parallélisé de calcul d'histogrammes*
cumsum_large.cl	*Algorithme parallélisé de somme cumulative (origine Nvidia)*
KSstatistic.cl	*Détermination des maxima et de deltaCDF. Mise en forme des résultats*

TABLE 2.11.: Liste des fichiers OpenCL

Fichiers	Fonctions	Descriptions
main.py		Programme principal
kernel_compilation.py	kernel_compilation	Compilation des *kernels*
radixSortv2.py	radixSort	Appel de l'algorithme de tri “radix”
histogram.py	histogram	Appel de l'algorithme de calcul d'histogrammes
KSstatistic.py	deltaCDF, maximum1, maximum2 et Qfunction	Appel des fonctions

TABLE 2.12.: Liste des fichiers Python

2.4.2.1. Générateur de distribution suivant une loi α-stable

Il est réalisé en s'inspirant de la fonction Matlab *stabrnd.m* (algorithme 2.1). Le générateur de base est un générateur de nombres aléatoires suivant une loi de distribution uniforme codée en C sous forme de noyaux et directement exécutée sur le GPU. Il est basé sur l'algorithme RANLUX (RANdom LUXury) [Jame94, Mart94]. Il n'y a aucun transfert massif de données entre l'hôte et la cible. Les espaces mémoire des séries X2 correspondant aux particules de l'algorithme OEP sont directement remplis par les noyaux

OpenCL (librairies pyopencl-ranluxcl.cl, stab_rnd.cl, fill.cl et copy.cl). Les seules variables faisant l'objet d'un transfert entre l'hôte et la cible et réciproquement sont les paramètres α_i, β_i, γ_i et μ_i déterminant les paramètres de la loi α-stable de la série X_2 pour la particule i.

2.4.2.2. Tri

Cette fonction repose sur un algorithme de tri " Radix " ou tri par base. Le principe de base se comprend grâce à l'exemple suivant :

Trier la liste : 170, 45, 75, 90, 2, 24, 802, 66.

1. tri par le chiffre le moins significatif (unités) : 170, 90, 2, 802, 24, 45, 75, 66.
2. tri par le chiffre suivant (dizaines) : 2, 802, 24, 45, 66, 170, 75, 90.
3. tri par le chiffre le plus significatif (centaines) : 2, 24, 45, 66, 75, 90, 170, 802.

Ces opérations reposent sur des opérations de masquage binaire (utilisation des opérateurs AND, OR et XOR) et non sur des opérateurs de comparaison.

Cet algorithme destiné initialement au tri des entiers positifs a été adapté aux réels (type FLOAT32 mantisse+exposant+signe codés sur 32 bits) par des modifications mineures . La référence [Duva10] décrit dans le détail l'algorithme Radix. L'implantation en OpenCL repose très largement sur un exemple tiré du SDK (*Sample Development Kit*) OpenCL de Nvidia [NVDIA]. Ce dernier est codé en C++ (à la place de Python). Il a fallu effectuer un travail de technique long afin de réaliser l'interfaçage avec pyOpenCL (identification des différents noyaux et de leurs paramètres, des variables et des tableaux utilisés).

2.4.2.3. Histogramme

Au niveau GPU, les histogrammes sont massivement utilisés en traitement d'image [Free11] car ils servent à évaluer le contraste d'une image (surexposition ou sous-exposition par exemple). Ils utilisent des méthodes d'indexation directe. Les classes de ces histogrammes sont de largeur fixe (codage des niveaux de gris sur 8 bits par exemple) et utilisent une méthode d'indexation directe.

Il s'agit d'un algorithme original basé sur l'utilisation de la dichotomie et des fonctions dites "atomiques" du standard OpenCL [OpenCL]. Pour chaque valeur de la série à trier, on applique d'abord un algorithme de dichotomie. La valeur est comparée aux bornes des classes de l'histogramme et à la valeur médiane. Le résultat de ces comparaisons permet de restreindre l'intervalle auquel appartient la classe de la valeur de la série en question. De proche en proche, lorsque l'intervalle est suffisamment réduit (le nombre de classes de l'intervalle est divisé par 2 à chaque itération), la classe est déterminée. L'index de l'histogramme correspondant à la classe déterminée précédemment est alors incrémenté de un. Plusieurs instances de ce processus peuvent être lancées en parallèle. En effet, la détermination de la classe à laquelle appartient une valeur de la série ne dépend pas de celle des autres valeurs. L'utilisation d'une fonction dite "atomique" pour réaliser les opérations d'incrémentation permet d'assurer que toutes les incrémentations seront prises en compte et qu'il n'y aura pas de masquage (absence de prise en compte), par exemple dans le cas où deux instances de l'algorithme incrémentent en même temps le même index (même adresse mémoire).

2.4.2.4. Somme cumulative

Les références [Hill86] et [Blel90] décrivent dans le détail les algorithmes de somme cumulative parallélisée.

Algorithme 2.3 Somme cumulative.

```
% on veut calculer la somme cumulative de N éléments d'un vecteur.
1) Pour d allant de 1 à log2(N) faire
2)     Pour k en parallèle faire
3)         Si k > 2^d alors
4)             x[out][k] ← x[in][k − 2^(d−1)] + x[in][k]
5)         Sinon
6)             x[out][k] := x[in][k]
7)         Fin Si.
8)     Fin Pour.
9)     Échanger (in,out)
10) Fin Pour
```

Soit une séquence de nombres x_0, x_1, x_2,... , x_n l'algorithme de somme cumulative parallélisé peut être décrit par le pseudo-code 2.3, où N est le nombre d'éléments de la séquence dont on veut calculer la somme cumulative.

La figure 2.4.3 montre graphiquement l'évolution du processus de calcul de la somme cumulative pour une série de 16 échantillons.

L'implantation en OpenCL repose très largement sur un exemple tiré du SDK (*Sample Development Kit*) OpenCL de Nvidia [NVDIA]. Ce dernier est codé en C++ (à la place de Python). De par, les optimisations en termes d'accès mémoire (globale/locale), des variables de stockage intermédiaires, le code est nettement plus complexe que le pseudo-code présenté précédemment. Il a fallu effectuer un travail de rétroingénierie long afin de réaliser l'interfaçage avec pyOpenCL (identification des différents noyaux, des variables et des tableaux utilisés).

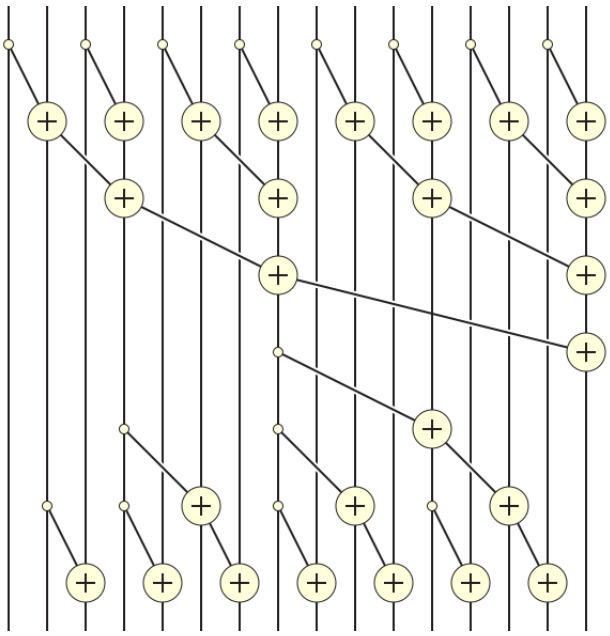


FIGURE 2.4.3.: Représentation d'un circuit à 16 éléments

2.4.2.5. Maximum

La valeur maximale d'une série se détermine à l'aide d'un algorithme parallélisé de "réduction". La série est découpée en sous-sections. Une instance du *kernel* compilé et téléchargé est affectée à chaque sous-

section, afin d'en déterminer la valeur maximale suivant un processus séquentiel. Les valeurs maximales locales ainsi déterminées sont rassemblées dans un nouveau tableau de dimension plus réduite (d'où le terme "réduction"). Puis on détermine la valeur maximale de ce dernier, ou on reprend un processus de découpage en sous-sections. On peut optimiser le nombre de sous-sections et le nombre d'étapes de réduction en fonction du GPU. De multiples instances de cet algorithme peuvent être lancées en fonction du nombre de particules.

2.4.2.6. Résultats

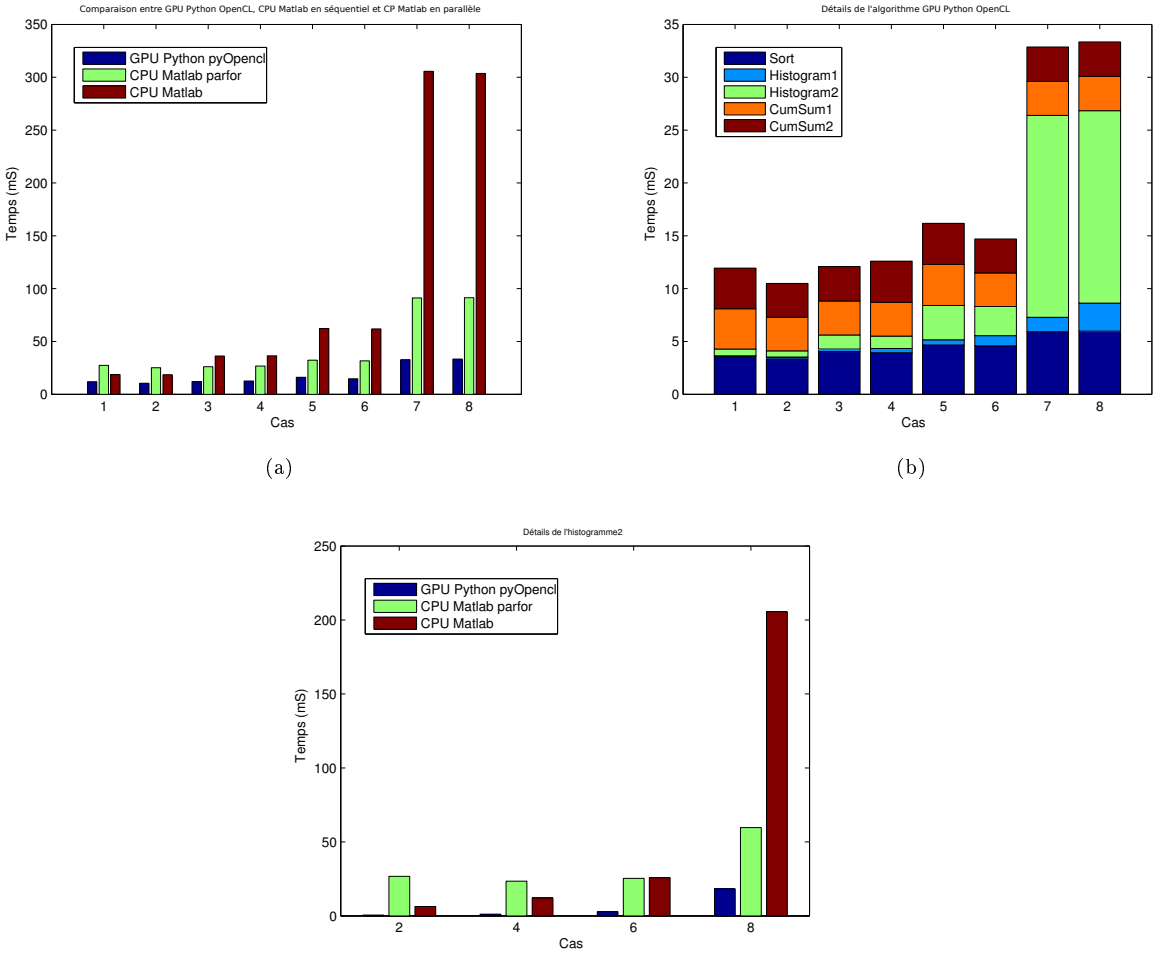
Le tableau (2.13) présente la comparaison des temps d'exécution GPU python+OpenCL/CPU Matlab de la fonction *kstest2* pour différentes tailles des séries X_1 et X_2. Le nombre de particules simulées est de 32. La carte GPU équipe les 8 stations de travail. La carte GTX 570 est plus puissante (2 fois plus de SM (*Streaming Multiprocessor*) et les fréquences du cœur et de la mémoire plus élevées). Ses résultats sont donnés à titre indicatif. L'analyse des résultats montre que, dans tous les cas, le GPU est nettement plus rapide que le CPU (de 4 à 16 fois), d'autant plus que les séries sont longues.

Cas	1	2	3	4	5
Série X1 (Nb échantillons N1)	1023	2047	4095	8191	16383
Série X2(Nb échantillons N2)	7167	14335	28671	57343	114686
N1+N2	8190	16382	32766	65534	131070
GPU Quadro 4000 Python/OpenCL (Temps d'exécution en mS)	30,2	27,0	37,4	100	144
GPU GTX 570 Python/OpenCL (Temps d'exécution en mS)	28,5	22,1	26,0	45,5	60,0
CPU X5690 Matlab (Temps d'exécution en mS)	53,9	106	195	419	959
Accélération GPU Quadro 4000	1,78	3,93	**5,21**	**4,19**	**6.66**
Accélération GPU GTX 570	1,89	4,79	**7,5**	**9,2**	**15.98**

TABLE 2.13.: Temps d'exécution de kstest2. Comparaison GPU Python+pyopencl/CPU Matlab.

L'implémentation de l'algorithme sur Python GPU/OpenCL est comparée à sa version séquentielle sur Matlab et sa version parallèle CPU de Matlab (*parfor*). Le mécanisme de "cores pool" de Matlab est similaire à celui de OpenCL. Dans la version parallèle de Matlab, à chaque itération de l'algorithme, la fonction objectif (2.3.5) est calculée en une seule passe pour les 32 particules (on remplace la boucle "*for*" par "*parfor*"). Le processeur utilisé est un 3.46 GHz 6-core INTEL XEON X5690. La figure (2.4.4a) montre le temps d'exécution d'une itération de l'algorithme (2.2), pour différentes versions. Dans tous les cas, l'implémentation sur Python GPU/OpenCL est meilleure que les deux autres et ceci d'autant plus que les longueurs des deux séries X_1 et X_2 sont grandes. En outre, pour les séries de petite taille, la version parallèle de Matlab est plus lente que la version séquentielle correspondante. cette figure montre aussi que la sensibilité de l'algorithme implémenté sur Python GPU/OpenCL par rapport à la longueur de la série X_1 ($N1$) est faible.

La figure 2.4.4b donne les détails des temps d'exécution de l'algorithme mis en œuvre sur GPU/OpenCL. Excepté pour l'histogramme, le temps de calcul de l'algorithme est à peine dépendant de la longueur de la série X_1. Cette faible dépendance peut être expliquée par le fait que les limites du processus de file d'attente de GPU/OpenCL ne sont pas atteintes. la figure 2.4.4c donne les détails des temps de calcul de l'algorithme "Histogram2". Cette figure montre que l'algorithme d'histogramme parallèle GPU/OpenCL surpasse très largement à la fois les versions séquentielle et parallèle de la fonction "histc" de Matlab, avec des accélérations de $11,3$ et $3,3$ respectivement pour le cas 8.



(a) (b)

(c)

FIGURE 2.4.4.: Temps d'exécution de l'algorithme : a) comparaison entre GPU Python OpenCL, CPU Matlab en séquentiel et CP Matlab en parallèle, b) détails de l'algorithme GPU Python OpenCL, c) détails de l'histogramme2. Les tests sont effectués sur les différentes tailles des séries X_1 et X_2 décrites dans le tableau 2.14.

Case	1	2	3	4	5	6	7	8
N1	256	512	1024	2048	2048	4096	4096	8192
N2	3840	3584	7168	6144	14336	12288	61440	57344
N1+N2	4096	4096	8192	8192	16384	16384	65536	65536

TABLE 2.14.: Description des cas étudiés : N_1 et N_2 correspondent à la taille des séries X_1 et X_2 respectivement.

2.5. Conclusion

Un nouvel estimateur non paramétrique de lois α-stables a été proposé. Cet estimateur présente deux particularités :

1. son originalité, liée à l'utilisation des métaheuristiques dans ce domaine statistique,
2. il ne fait aucune restriction sur les paramètres à estimer.

L'algorithme présenté dans ce chapitre obtient de très bons résultats et, plus particulièrement, pour des valeurs de l'indice de stabilité $\alpha < 1$. Il dépasse, de ce fait, l'un des estimateurs les plus efficaces et les plus utilisés, à savoir, l'estimateur de McCulloch.

Un autre avantage de notre estimateur est qu'il est intrinsèquement parallélisable. En effet, dans la deuxième partie des résultats, nous avons proposé des fonctions parallélisées (tri, histogramme, somme cumulative et valeur maximale d'une série) effectuant les calculs sur des cartes GPU. Ces fonctions parallélisées développées ou adaptées sont plus que réutilisables et granulaires, car il s'agit de fonctions de base extrêmement courantes dans le domaine des probabilités et des statistiques, entre autres.

CHAPITRE 3

PLACEMENT DE FACTS

3.1. Introduction

Ces dernières années, l'interconnexion des réseaux et la dérégulation des marchés de l'énergie ont fortement modifié les conditions d'opération des réseaux de transport de l'énergie électrique. Les opérateurs de réseaux (TSO pour *Transmission System Operator*) doivent gérer au mieux des échanges d'énergie (flux de puissance) entre des producteurs de plus en plus nombreux et variés (introduction de production décentralisée) et des consommateurs de plus en plus éloignés des centres de production (réseaux interconnectés), tout en tenant compte des contraintes physiques de réseaux exploités de plus en plus près de leurs limites de fonctionnement. Une telle gestion nécessite de disposer de moyens de plus en plus performants pour contrôler les transits de puissance dans les lignes de transport afin, d'une part, d'optimiser les flux de puissance (e.g. réduire les nœuds de congestion) et, d'autre part, d'accroître la stabilité du réseau. A ce titre, les FACTS (*Flexible Alternative Current Transmission System*), en agissant principalement sur la répartition de la puissance réactive dans le réseau, apparaissent comme des dispositifs offrant une grande souplesse de contrôle. Ainsi, ils permettent, d'une part, en régime permanent, une "optimisation" des flux de puissance et une meilleure maîtrise de la tension aux nœuds du réseau et, d'autre part, en régime transitoire, une amélioration sensible de la stabilité du réseau.

L'objectif est alors de déterminer la répartition optimale de la puissance réactive, en relation avec la localisation et le dimensionnement optimaux de FACTS, afin d'améliorer les performances d'un réseau électrique. La puissance réactive agissant localement, peut-elle avoir une influence sur le comportement global du réseau ? Peut-on alors en déduire la zone d'influence d'un nœud réactif ? Plus avant, localisation et dimensionnement de la puissance réactive doivent permettre de déterminer si sa gestion peut être distribuée ou si elle doit être localisée. En d'autres termes, les questions posées sont :

- où placer des FACTS dans le réseau ?
- combien de FACTS ?
- quelle puissance (taille) attribuer à ces FACTS ?
- quel(s) type(s) de FACTS ?
- a quel prix ?

Ce chapitre apporte des éléments de réponse à ces questions. On ne considèrera ici que le cas du fonctionnement du réseau en régime statique (permanent), sans prise en compte du régime transitoire (i.e stabilité transitoire du réseau). Pour ce faire, trois grandes nouveautés ont été apportées :

- la première réside dans le choix libre (effectué par l'algorithme) des quatre variables de décision (le nombre de FACTS, leur emplacement, leur valeur de consigne et leur type). En effet, à notre connaissance, peu (ou pas) de travaux traitent ces quatre variables en même temps ;
- la seconde réside dans l'hybridation de l'algorithme à travers le codage proposé ;
- enfin, la dernière contribution consiste en l'application d'un nouvel algorithme (α-SLPSO).

Le présent chapitre s'organise en 4 parties. La première partie fait un bref rappel sur le modèle *Power Flow* et sa résolution. Elle présente également les modifications du modèle du réseau introduites par implantation de FACTS. Elle se termine par une revue rapide de quelques critères à optimiser par placement de FACTS, en relation notamment avec la sécurité du réseau et l'augmentation de ses capacités de transport d'énergie.

La deuxième partie du chapitre passe en revue quelques méthodes d'OEP discrètes mises en œuvre pour résoudre le problème de placement et dimensionnement optimal de FACTS. Nous donnons également dans cette partie les codages des particules utilisés pour y parvenir.

Dans la troisième partie de ce chapitre, nous proposons un nouvel algorithme à base de population d'agents (α-SLPSO : $\alpha-$*Stable Lévy Particle Swarm Optimization*) et une recherche locale de Lévy (α-SLLS : α-*Stable Lévy Local Search*) basés sur les propriétés d'invariance d'échelle des lois α-stables de Lévy. Nous testons ces deux algorithmes sur des fonctions test en les comparant à d'autres variantes de l'OEP.

Enfin, la dernière partie du chapitre présente les résultats de simulation obtenus sur une application aux cas des réseaux IEEE 30 et IEEE 57, respectivement, dans les cas mono-objectif et multi-objectif.

3.2. Le modèle *Power Flow* et sa résolution

L'approche proposée s'articule sur l'association d'un calcul de réseau de type *Power Flow* et de techniques d'optimisation numérique. Cette section présente brièvement le problème *Power Flow* et sa résolution. Plus de détails sont donnés dans le document interne [Arevab].

3.2.1. Le modèle *Power Flow*

Le modèle *Power Flow* donne une représentation d'un réseau électrique en régime permanent de fonctionnement. Ses équations s'appuient sur les lois de Kirschoff (loi des nœuds) à partir d'une description du réseau en termes de nœuds (générateurs et/ou charges) et de liens entre ces nœuds (lignes de transport de l'énergie). Elles formalisent un état d'équilibre du réseau sous certaines conditions d'entrée (e.g. puissances consommées et générées) et certaines contraintes d'opération (e.g. flux de puissance maximal sur les lignes de transport).

L'état du réseau est complètement décrit par les variables d'état que sont :

- l'amplitude de la tension aux nœuds, soit pour le nœud i, U_i
- la phase de la tension aux nœuds, soit pour le nœud i, θ_i

Le réseau est soumis à différentes entrées, soient :

- les puissances active et réactive consommées aux nœuds "charge", soit respectivement, pour le nœud i : P_{Di} et Q_{Di}

– les puissances active et réactive générées aux nœuds "générateur", soit respectivement, pour le nœud i : P_{Gi} et Q_{Gi}

L'équilibre du réseau est obtenu lorsque les variables d'état U_i et θ_i satisfont les équations d'équilibre de puissance active et réactive en chacun des nœuds du réseau. Pour le nœud i, ces équations sont respectivement données par les relations (3.2.1) et (3.2.2).

$$P_{Gi} - P_{Di} = U_i \sum_{j \in \Omega_i} U_j \left(G_{ij} \cos(\theta_i - \theta_j) + B_{ij} \sin(\theta_i - \theta_j)\right) \tag{3.2.1}$$

$$Q_{Gi} - Q_{Di} = U_i \sum_{j \in \Omega_i} U_j \left(G_{ij} \sin(\theta_i - \theta_j) - B_{ij} \cos(\theta_i - \theta_j)\right) \tag{3.2.2}$$

où Ω_i est l'ensemble des nœuds connectés au nœud i, y compris le nœud i. G_{ij} et B_{ij} sont les éléments de la matrice d'admittance Y tels que $Y_{ij} = G_{ij} + jB_{ij}$.

La résolution des équations d'équilibre de puissance (3.2.1) et (3.2.2) est soumise à un ensemble de contraintes de fonctionnement du réseau, correspondant à ses limites physiques. Ces contraintes se classent en deux familles :

– Des contraintes d'inégalité :

Les contraintes d'inégalité se rapportent aux limites de flux maximal de puissance sur les lignes de transport :

$$S_{ij}(U_i, \theta_i) \leq S_{ij_{max}} \tag{3.2.3}$$

avec S_{ij}, le flux de puissance sur la ligne reliant les nœuds i et j et $S_{ij_{max}}$, le flux maximal autorisé sur la même ligne.

– Des contraintes de bornes sur les variables d'état et d'entrée, avec :

  – des limites basse et haute sur les puissances active et réactive générées aux nœuds "générateurs" :

$$P_{Gi_{min}} \leq P_{Gi} \leq P_{Gi_{max}} \tag{3.2.4}$$

$$Q_{Gi_{min}} \leq Q_{Gi} \leq Q_{Gi_{max}} \tag{3.2.5}$$

  – des limites basse et haute sur l'amplitude de la tension aux différents nœuds du réseau :

$$U_{i_{min}} \leq U_i \leq U_{i_{max}} \tag{3.2.6}$$

3.2.2. Résolution des équations du réseau

Deux approches existent pour la résolution des équations du modèle *Power Flow* :

– une approche dite "*Power Flow* (PF)", où le problème consiste à résoudre les équations d'équilibre de puissance (3.2.1) et (3.2.2) pour certaines variables (d'état ou d'entrée) imposées ;

– une approche, dite "*Optimal Power Flow* (OPF)", où le problème consiste à minimiser une fonction-nelle donnée, généralement en relation avec les coûts des puissances active et/ou réactive générées, tout en respectant, d'une part, les équations d'équilibre de puissance (3.2.1) et (3.2.2) et, d'autre

part, les contraintes d'inégalité et de bornes (3.2.3), (3.2.4), (3.2.5) et (3.2.6).

3.2.2.1. Approche *Power Flow*

L'examen des équations d'équilibre de puissance (3.2.1) et (3.2.2), en termes de nombre d'équations ($2N$, où N est le nombre total de nœuds) et de nombre de variables d'état et d'entrée ($4N$), montre qu'il est nécessaire de fixer certaines variables, afin de pouvoir résoudre ces équations.

Le choix entre variables imposées et variables à déterminer en chaque nœud va dépendre de la nature du nœud considéré, c'est-à-dire :

- Pour les nœuds "charge" :
  - Variables imposées : P_{Di} et Q_{Di}
  - Variables à déterminer : U_i et θ_i
- Pour les nœuds "générateur", à l'exception du nœud "bilan" (ou *swing bus*) :
  - Variables imposées : P_{Gi} et U_i
  - Variables à déterminer : Q_{Gi} et θ_i
- Pour le nœud "générateur bilan" :
  - Variables imposées : U_i et θ_i
  - Variables à déterminer : P_{Gi} et Q_{Gi}

Une fois les variables imposées définies, la solution du problème PF non linéaire (i.e. des équations d'équilibre de puissance (3.2.1) et (3.2.2)) est déterminée en utilisant une méthode numérique itérative (e.g. méthode de Newton-Raphson). Cette solution donne l'état du réseau, pour des conditions de fonctionnement données (variables imposées), qui assure l'équilibre des puissances en chacun des nœuds du réseau. Cependant, une telle solution ne sera valide que si elle respecte les contraintes d'inégalité et de bornes imposées, ce qui peut ne pas être le cas. En cas de solution non valide, il est alors nécessaire de modifier les variables imposées en chaque nœud et de calculer une nouvelle solution.

3.2.2.2. Approche *Optimal Power Flow*

L'approche OPF se distingue de l'approche PF en ce sens qu'elle recherche une solution qui, d'une part, respectera toutes les contraintes d'inégalité et de bornes du réseau et, d'autre part, optimisera (souvent minimisera) une fonctionnelle donnée.

Comme pour le problème PF, la détermination d'une solution passe par l'imposition de certaines variables d'entrée ou d'état, c'est-à-dire :

- Pour les nœuds "charge" :
  - Variables imposées : P_{Di} et Q_{Di}
  - Variables à déterminer : U_i et θ_i
- Pour les nœuds "générateur", à l'exception du nœud "bilan" (ou *swing bus*) :
  - Variables imposées : Aucune
  - Variables à déterminer : P_{Gi}, Q_{Gi}, U_i et θ_i
- Pour le nœud "générateur bilan" :
  - Variables imposées : θ_i
  - Variables à déterminer : P_{Gi}, Q_{Gi} et U_i

Ici, le nombre de variables à déterminer est plus élevé que pour le problème PF. Afin de pouvoir résoudre le problème, il est nécessaire d'ajouter des contraintes supplémentaires. Ces contraintes représentent,

d'une part, les contraintes d'inégalité (3.2.3) et de bornes (3.2.4), (3.2.5), (3.2.6) et, d'autre part, la fonctionnelle à optimiser.

Par exemple, une fonctionnelle relative au coût de la puissance générée peut s'exprimer comme :

$$J_{eco} = \sum_{i=1}^{N_G} f_i(P_{Gi}, Q_{Gi}, U_i, \theta_i) \tag{3.2.7}$$

où N_G est le nombre total de nœuds "générateur".

La solution du problème OPF, si elle existe, détermine un état du réseau qui respecte toutes les contraintes du réseau (i.e. d'égalité, d'inégalité et de bornes). Elle présente l'intérêt de pouvoir intégrer des contraintes de coût d'énergie dans le problème de placement et de dimensionnement de FACTS dans un réseau. Bien entendu, la solution de ce problème dépendra fortement des critères et des paramètres de coût retenus et ne sera "optimale" que pour ces critères et paramètres donnés, ce qui peut apparaître comme une limitation de la solution du problème de placement et dimensionnement de FACTS obtenue. Cependant, des contraintes du même ordre existent dans le cas d'une résolution de type PF. En effet, la solution du problème PF dépend du choix des valeurs imposées (et plus particulièrement des amplitudes de tension imposées aux nœuds "générateurs") et la solution du problème de placement et dimensionnement de FACTS sera ici dépendante de la distribution de ces variables imposées.

En d'autres termes, que ce soit pour une résolution des équations du réseau par une approche PF ou par une approche OPF, la solution du problème de placement et dimensionnement de FACTS sera dépendante des variables imposées, i.e. d'une certaine configuration du réseau en termes de puissances demandées, de puissances générées,... Passer au delà de cette limitation impose la prise en compte de variations, dans certaines limites, des entrées imposées aux problèmes PF ou OPF. Le problème d'optimisation associé devra alors faire appel à des techniques d'optimisation dans un environnement stochastique, ce qui n'entre pas dans le cadre du présent manuscrit.

Dans le cadre du travail présenté ici, la recherche d'une solution au problème de placement et de dimensionnement de FACTS est effectuée sur la base d'une résolution des équations du réseau par une approche OPF, avec des paramètres de coût constants.

3.2.3. Implantation de FACTS et modification du réseau

Implanter un FACTS sur un nœud ou une ligne du réseau électrique revient principalement à modifier la répartition de la puissance réactive dans le réseau. Cette modification dépend notamment de la nature du FACTS (e.g. série ou parallèle) implanté. En régime permanent, pour la plupart des FACTS, elle se traduit généralement, au niveau de la description du réseau, par une modification, soit de la matrice d'admittance Y, soit de demandes de puissance réactive en certains nœuds.

La liste suivante, sans être exhaustive, donne des indications sur ces modifications pour quelques FACTS assez souvent utilisés :

- FACTS parallèle :
  - STATCOM : modification de la puissance réactive au nœud de connexion du STATCOM ;
  - SVC : modification de l'impédance parallèle au nœud de connexion du SVC, modification de la matrice Y ;

- FACTS série :
  - TCSC : modification de l'impédance de la ligne où est implanté le TCSC, modification de la matrice Y ;
  - Transformateur déphaseur : modification de l'impédance de la ligne où est implanté le transformateur, modification de la matrice Y.

Plus de détails sur la modélisation associée à ces différents FACTS (et quelques autres) peuvent être trouvés dans la thèse de S. Gerbex [Gerb03].

Dans le cadre de ce rapport, seuls les cas de STATCOM, SVC et TCSC sont traités. La modification du réseau est donc faite sous forme d'une modification de la puissance réactive demandée aux nœuds sélectionnés, ou sous forme de contrôle de transit des puissances dans les lignes sélectionnées.

3.2.4. Choix des critères à optimiser par placement et dimensionnement de FACTS

Les dispositifs FACTS étant modélisés en régime permanent, les critères à optimiser seront principalement en relation directe avec l'effet des FACTS sur les transits de puissance (flux de puissance) et les tensions aux nœuds du réseau étudié.
Ces critères se présentent sous forme de fonctions objectifs à minimiser, ou maximiser selon les cas considérés, c'est-à-dire :

- la maximisation de la charge desservie ,

- la maximisation de la sécurité du système, avec :
  - l'équilibrage de la charge des lignes,
  - l'équilibrage des tensions aux nœuds,
  - la sécurité d'exploitation,

- La minimisation des pertes actives.

Maximisation de la charge desservie
La maximisation de la charge desservie (*system loadability*) consiste à déterminer la puissance maximale pouvant être desservie aux consommateurs, tout en respectant les contraintes du réseau. La fonctionnelle associée, à maximiser, s'exprime comme :

$$J_1 = \sum_{i=1}^{N_L} P_{L_i} \tag{3.2.8}$$

où P_{L_i} est la puissance consommée au nœud i, et N_L est le nombre total de nœuds du réseau.

Maximisation de la sécurité du système
Dans ce cas, les FACTS doivent permettre d'exploiter le réseau avec une marge maximale de sécurité. A cette fin, deux critères peuvent être retenus, à savoir, l'équilibrage de la charge des lignes et l'équilibrage des tensions aux nœuds. Ces deux critères peuvent également être associés pour définir un critère de sécurité d'exploitation.

Équilibrage de la charge des lignes
Cet équilibrage consiste à répartir de façon la plus uniforme possible les charges entre les différentes lignes du réseau. La fonctionnelle associée, à minimiser, s'exprime alors comme :

$$J_2 = \sum_{j=1}^{N_B} (\frac{S_j}{S_{j_{max}}})^2 \tag{3.2.9}$$

où S_j est la puissance transitant sur la ligne j, $S_{j_{max}}$ est la puissance maximale autorisée sur la ligne j et N_B est le nombre total de lignes.

Équilibrage des tensions aux nœuds
Cet équilibrage consiste à répartir de façon la plus uniforme possible les tensions aux différents nœuds du réseau. La fonctionnelle associée, à minimiser, s'exprime alors comme :

$$J_3 = \sum_{i=1}^{N_L} (\frac{U_i - U_{i_N}}{U_{i_N}})^2 \tag{3.2.10}$$

où U_i est la tension au nœud i, U_{i_N} est la tension nominale au nœud i et N_L est le nombre total de nœuds du réseau.

Dans le cas où les tensions s'expriment en p.u. (*per unit*), la tension nominale $U_{i_N} = 1$ et le critère J_3 s'exprime comme :

$$J_3 = \sum_{i=1}^{N_L} (U_i - 1)^2 \tag{3.2.11}$$

Sécurité d'exploitation
La sécurité d'exploitation combine l'équilibrage de la charge des lignes et l'équilibrage des tensions aux nœuds. La fonctionnelle associée, à minimiser, est donnée par :

$$J_4 = \sum_{j=1}^{N_B} \omega_j (\frac{S_j}{S_{j_{max}}})^{2p} + \sum_{i=1}^{N_L} \omega_i (\frac{U_i - U_{i_N}}{U_{i_N}})^{2q} \tag{3.2.12}$$

où ω_j et ω_i sont des poids permettant de donner de l'importance à certaines lignes ou certains nœuds critiques du réseau. p et q sont des exposants permettant de pondérer la pénalisation entre surcharges de ligne et écarts de tension.

Minimisation des pertes actives
Il s'agit ici de minimiser les pertes actives sur les lignes du réseau. Au delà d'une dégradation des lignes due à leur échauffement, par dissipation d'énergie Joule, réduire les pertes actives revêt un intérêt économique certain, le coût associé étant facturé aux consommateurs. La fonctionnelle associée, à minimiser, est donnée par :

$$J_5 = \sum_{j=1}^{N_B} R_j I_j^2 \tag{3.2.13}$$

où R_j est la résistance de la ligne j et I_j est le courant circulant dans la même ligne.

Dans le cadre du présent rapport, le critère retenu concerne l'équilibrage des tensions aux nœuds du réseau. La fonctionnelle associée, à minimiser, est alors donnée par l'équation (3.2.11).

3.3. Description des différents algorithmes utilisés

Nous présentons les différentes techniques utilisées pour résoudre le problème de placement et dimensionnement de FACTS. En raison de sa capacité à résoudre une large classe de problèmes difficiles, l'optimisation par essaim de particules (OEP) sera appliquée et adaptée au problème de placement et dimensionnement de FACTS. Comme nous allons voir par la suite, ce dernier est abordé sous forme d'un problème hybride (variables continues/discrètes). Nous commençons par donner la manière dont les particules sont codées dans les deux cas d'études :

- a) Cas particulier : le nombre de FACTS est fixé à 2, de type STATCOM. Ce cas d'étude est justifié par la volonté d'une bonne validation des algorithmes appliqués (ainsi que le paramétrage) dans la section 3.4. Pour ce faire, la variante standard du PSO 2007 (chapitre I, paragraphe 1.2.4.5) et la méthode "*Velocity Likelihoods*" (section 3.3.2) sont, respectivement, appliquées sur les variables continues et discrètes.

- b) Cas général : le nombre et le type des FACTS sont laissés libres. Les variables discrètes sont traitées par la méthode "Velocity Likelihoods". A titre de comparaison, les variables continues sont traitées à l'aide de deux algorithmes : le PSO standard 2007 et l'algorithme α-SLPSO (cf 3.3.3.1). La motivation principale de l'introduction de la nouvelle approche OEP (α-SLPSO) réside dans le désir d'amélioration des résultats obtenus par le PSO standard 2007.

Nous détaillons à la suite la méthode "Velocity Likelihoods" appliquée pour aborder le caractère discret (les emplacements des FACTS) de notre problème d'optimisation. Cette méthode est légèrement modifiée par nos soins, dans le but d'apporter des améliorations dans les résultats obtenus.

Enfin, nous présentons, dans la dernière partie de cette section, un nouvel algorithme (α-SLPSO), qui sera validé sur 15 fonctions test (indiquées en annexe A.1).

3.3.1. Codages des solutions

La prise en considération de toutes les variables de décision dans l'algorithme de l'OEP nécessite une réflexion approfondie sur la manière de représenter au mieux une solution (bon codage des particules). En effet, la manipulation de nombreuses variables, considérées séparément, conduit à des résultats médiocres, en raison de la non synchronisation des convergences des différentes variables de décision (i.e, une variable de décision peut converger lentement ou converger vers un optimum local), ce qui conduit à des optima locaux et éventuellement à la non convergence de l'algorithme.

Pour remédier à ce problème, deux types de codages ont été proposés. Le premier est adapté dans le cas du placement et dimensionnement de deux FACTS du même type (cas particulier) et le second est adapté dans le cas du placement et dimensionnement de plusieurs FACTS de types différents (cas général).

Cas du placement et dimensionnement de deux FACTS du même type

Dans ce type de codage, une particule i est représentée par un vecteur

$$\overrightarrow{X_i} = (e_1 ... e_{n_f}; t_1 ... t_{n_f}; v_1 ... v_{n_f})$$

en désignant par :

- $e_i \in \{1, .., Nombre\,de\,noeuds\}$, $e_i \neq e_j$ pour tout $j \neq i$: les emplacements des FACTS.
- $t_i \in \{STATCOM, SVS, TCS\}$: les types des FACTS.
- $v_i \in [Q_{min}, Q_{max}]$: les valeurs de consigne des FACTS.

– $n_f \in N$: le nombre de FACTS (fixé) à placer.

Une solution (ou particule) peut donc être représentée comme suit :

v_1	v_2	...	v_{n_f}
t_1	t_2	...	t_{n_f}
e_1	e_2	...	e_{n_f}

Comme nous l'avons déjà souligné, ce type de codage est adapté dans le cas de placement et dimensionnement optimal de n_f FACTS du même type, c'est-à-dire :

- le type de FACTS est fixé (ici STATCOM),
- le nombre de FACTS n_f est fixé (dans notre cas d'étude à 2).

Cas du placement et dimensionnement de plusieurs FACTS de types différents (cas général)

Dans ce type de codage, la forme de la particule est modifiée comme ci-dessous de façon à expliciter des variables de décision. Ce type de codage est adapté dans le cas général où le type des FACTS est libre (STATCOM, SVC et TCSC), le nombre de FACTS est libre mais également dans le cas particulier où le type des FACTS est fixé.

Une particule est représentée par un vecteur :

$$V = (e_1...e_n e_{n+1}...e_{2n} e_{2n+1} e_{2n+b}; v_1...v_n v_{n+1}...v_{2n} v_{2n+1} v_{2n+b})$$

en désignant par :

– $e_i \in \{0, 1\}$, 0 s'il n'y a pas de FACTS placé sur le nœud i et 1 sinon,
– n, le nombre de nœuds dans le réseau,
– b, le nombre de branches dans le réseau,
– $v_i \in [Q_{min}, Q_{max}]$, les valeurs de consigne des FACTS.

Soit, sous forme de tableau :

e_1	e_2	...	e_{2n+b}
v_1	v_2	...	v_{2n+b}

Remarque. Les deux variables de décision "type" et "nombre" sont explicites dans ce type de codage. En effet, pour connaître le nombre de FACTS à placer, il suffit de faire la somme sur les e_i et pour connaître le type du FACTS placé au nœud i on procède de la manière suivante :

– si $i \in \{1, .., n\}$, le FACTS est de type STATCOM,
– si $i \in \{n + 1, .., 2n\}$, le FACTS est de type SVC,
– si $i \in \{2n + 1, .., 2n + b\}$, le FACTS est de type TCSC.

3.3.2. Optimisation par essaim de particules binaire (PSO méthode *Velocity Likelihoods*)

La méthode d'OEP discret, dite "Velocity Likehood", a été introduite par E.S. Correa, A.A. Freitas et C.G. Johnson [Corr06]. Le principe de base est brièvement présenté ici et illustré par un exemple simple. Une modification de la méthode sera ensuite introduite pour une meilleure exploration de l'ensemble du domaine de recherche.

3.3.2.1. Principe de la méthode

Définition des variables

Soit une particule di dont la position $\overrightarrow{X_{di}}$, de dimension M, est définie à l'itération k par :

$$\overrightarrow{X_{di}}(k) = [x_{di1}\, x_{di2} \ldots x_{diM_i}] \tag{3.3.1}$$

avec $x_{dim} \in [1 \cdots N]\,, \quad m = 1 \cdots M$ et $x_{dim} \neq x_{diq}\ \forall m, q$ et N, le nombre total de positions possibles pour la particule di.

Dans le cas d'un problème de placement de FACTS, la dimension M de la particule s'identifie au nombre de FACTS à implanter dans le réseau. En choisissant des particules de dimensions différentes, il sera alors possible d'introduire le nombre de FACTS à implanter comme un paramètre d'optimisation. Toutefois, une autre approche d'hybridation prenant en compte le nombre et les types des FACTS comme variables d'optimisation sera présentée plus loin.

Les éléments x_{dim} étant identifiés aux nœuds du réseau, choisir $x_{dim} \neq x_{diq}\ \forall m, q$ revient à interdire que plusieurs FACTS soient positionnés sur un même nœud. A l'itération (k-1), la meilleure position de la particule di est donnée par :

$$\overrightarrow{P_{best}^{di}}(k-1) = [p_{di1}\, p_{di2} \ldots p_{diM}] \tag{3.3.2}$$

avec $b_{dim} \in [1 \cdots N]\,, \quad m = 1 \cdots M_i$ et $b_{dim} \neq b_{diq}\ \forall m, q$

Chaque particule di a un ensemble $\Omega_{inform_{di}}$, de dimension $N_{inform_{di}}$, de particules informatrices qui communiquent avec elle, en lui transmettant leurs meilleures positions respectives. Soit alors la meilleure position de l'ensemble des particules informatrices de la particule di définie à l'itération (k-1) par :

$$\overrightarrow{G_{best}^{di}}(k-1) = [g_{di1}\, g_{di2} \ldots g_{diL}] \tag{3.3.3}$$

avec $g_{dil} \in [1 \cdots N]\,, \quad l = 1 \cdots L$ et $g_{dil} \neq g_{diq}\ \forall l, q$

Soit la vitesse de déplacement de la particule di définie, à l'itération k, comme :

$$\overrightarrow{V_{di}}(k) = \begin{bmatrix} v_{di1} & v_{di2} & \cdots & v_{diN} \\ p_{di1} & p_{di2} & \cdots & p_{diN} \end{bmatrix} \tag{3.3.4}$$

avec $p_{dij} \in [1 \cdots N]\,, \quad j = 1 \cdots N$ et $p_{dij} \neq p_{diq}\ \forall j, q$.

La première ligne de $\overrightarrow{V_{di}}$ contient des poids associés aux différentes positions contenues dans la seconde ligne de $\overrightarrow{V_{di}}$.

Description de l'algorithme

L'algorithme OEP discret " *Velocity Likehood* " s'appuie sur les étapes suivantes :

1. A partir de la position $\overrightarrow{X_{di}}(k)$, on calcule la fonction de *fitness*, à l'itération k, $J(\overrightarrow{X_{di}}(k))$,
2. Connaissant $J(\overrightarrow{X_{di}}(k))$, on actualise, si nécessaire $\overrightarrow{P_{best}^{di}}(k)$ et $\overrightarrow{G_{best}^{di}}(k)$,
3. Connaissant $\overrightarrow{V_{di}}(k)$, $\overrightarrow{X_{di}}(k)$, $\overrightarrow{P_{best}^{di}}(k)$ et $\overrightarrow{G_{best}^{di}}(k)$, on actualise la vitesse $\overrightarrow{V_{di}}(k+1)$,
4. Connaissant $\overrightarrow{V_{di}}(k+1)$ et $\overrightarrow{X_{di}}(k)$, on actualise la position $\overrightarrow{X_{di}}(k+1)$.

Algorithme 3.1 Mise à jour des vitesses des particules par la méthode "Velocity Likelihoods".

Pour $j = 1 \cdots N$:
- Pour $m = 1 \cdots M$:
  - Si $\overrightarrow{V_{di}}(k)\,[2,j] = \overrightarrow{X_{di}}(k)\,[m]$ alors :

$$\overrightarrow{V_{di}}(k)\,[1,j] = \overrightarrow{V_{di}}(k)\,[1,j] + \alpha_{di}(k) \tag{3.3.5}$$

  - Fin Si
  - Si $\overrightarrow{V_{di}}(k)\,[2,j] = \overrightarrow{P_{best}^{di}}(k)\,[m]$ alors :

$$\overrightarrow{V_{di}}(k)\,[1,j] = \overrightarrow{V_{di}}(k)\,[1,j] + \beta_{di}(k) \tag{3.3.6}$$

  - Fin Si
- Fin Pour
- Pour $l = 1 \cdots L$:
  - Si $\overrightarrow{V_{di}}(k)\,[2,j] = \overrightarrow{G_{best}^{di}}(k)\,[l]$ alors :

$$\overrightarrow{V_{di}}(k)\,[1,j] = \overrightarrow{V_{di}}(k)\,[1,j] + \gamma_{di}(k) \tag{3.3.7}$$

  - Fin Si
- Fin Pour

Fin Pour

Calcul de $J(\overrightarrow{X_{di}}(k))$

La détermination de la fonction objectif $J(\overrightarrow{X_{di}}(k))$ est effectuée à partir d'une résolution de type OPF du modèle du réseau étudié. Les variables d'état ainsi obtenues permettent alors de déterminer $J(\overrightarrow{X_{di}}(k))$. Pour rappel, les équations du modèle *Power Flow* (les équations (3.2.1), (3.2.2), (3.2.3), (3.2.4), (3.2.5) et (3.2.6)) forment un système d'équations non linéaire.

Actualisation de $\overrightarrow{P_{best}^{di}}$ et $\overrightarrow{G_{best}^{di}}$

A l'instar de l'OEP standard décrite dans la section 1.2.4.1 du chapitre I, les meilleures positions personnelles $\overrightarrow{P_{best}}$ et des informatrices des particules $\overrightarrow{G_{best}}$ sont mises à jour selon la manière classique de fonctionnement d'un algorithme d'optimisation par essaim de particules. En d'autres termes, la mise à jour de $\overrightarrow{P_{best}^{di}}(k)$ et $\overrightarrow{G_{best}^{di}}(k)$ de la particule di à l'itération k se fait comme suit :

- Si $J(\overrightarrow{X_{di}}(k)) < J(\overrightarrow{P_{best}^{di}}(k-1))$, alors $\overrightarrow{P_{best}^{di}}(k) = \overrightarrow{X_{di}}(k)$, sinon, $\overrightarrow{P_{best}^{di}}(k) = \overrightarrow{P_{best}^{di}}(k-1)$
- Soit $J(\overrightarrow{P_{best}^{dr}}(k)) = \min_{j \in \Omega_{inform_{di}}} J(\overrightarrow{P_{best}^{dj}}(k)$:

  Si $J(\overrightarrow{P_{best}^{dr}}(k)) < J(\overrightarrow{G_{best}^{di}}(k-1))$, alors $\overrightarrow{G_{best}^{di}}(k) = \overrightarrow{P_{best}^{dr}}(k)$, sinon, $\overrightarrow{G_{best}^{di}}(k) = \overrightarrow{G_{best}^{di}}(k-1)$

Actualisation de la vitesse de la particule di

Les différentes étapes du processus d'actualisation de la vitesse V_{di} sont décrites dans l'algorithme 3.1 où $\alpha_{di}(k) > 0$, $\beta_{di}(k) > 0$ et $\gamma_{di}(k) > 0$ sont des réels positifs.

Les paramètres $\alpha_{di}(k)$, $\beta_{di}(k)$ et $\gamma_{di}(k)$ traduisent le "degré d"attractivité" donné respectivement à la position $\overrightarrow{X_{di}}$, à la meilleure position $\overrightarrow{P_{best}^{di}}$ et à la meilleure position globale $\overrightarrow{G_{best}^{di}}$, associées à la particule di, dans le processus d'actualisation de la vitesse. Dans la version de base de l'algorithme, ils sont choisis constants. Ils jouent des rôles similaires à ceux joués par les paramètres $w_{ci}(k)$, $c_{i1}(k)$ et $c_{i2}(k)$ dans le cas de l'OEP continu (chapitre I, section *états de l'art*). Ils viennent accroître le poids relatif de certaines positions contenues dans le vecteur de vitesse $\overrightarrow{V_{di}}$, selon que ces positions appartiennent ou non à $\overrightarrow{X_{di}}$,

$\overrightarrow{P_{best}^{di}}$ et $\overrightarrow{G_{best}^{di}}$.

Une composante stochastique est introduite dans le processus d'actualisation de la vitesse, de la manière suivante :

$$\overrightarrow{V_{di}}(k)\,[1, 1:N_i] = \overrightarrow{V_{di}}(k)\,[1, 1:N_i] * \varphi_{di}(k) \tag{3.3.8}$$

où $*$ représente le produit d'Hadamard (i.e. composante par composante) de deux matrices et $\varphi_{di}(k)$ est un vecteur de N variables aléatoires, uniformément distribuées entre 0 et 1.

Soit la composante élémentaire $\overrightarrow{V_{di_j}}(k)$ de la vitesse définie comme :

$$\overrightarrow{V_{di_j}}(k) = \begin{bmatrix} v_{di_j}(k) \\ p_{di_j}(k) \end{bmatrix} = V_{di}(k)\,[1:2, j]$$

La vitesse $\overrightarrow{V_{di}}(k+1)$ est alors déterminée par une réorganisation de la vitesse $\overrightarrow{V_{di}}(k)$ selon un tri décroissant sur les poids $v_{di_j}(k)$ associés aux composantes élémentaires $\overrightarrow{V_{di_j}}(k)$:

$$\overrightarrow{V_{di}}(k+1) = tri_{v_{di_j}(k)\,decroissant}\left[\overrightarrow{V_{di}}(k)\,[1:2, j]\right] \; pour\; j = 1 \cdots N \tag{3.3.9}$$

Actualisation de la position de la particule di

L'actualisation de la position $\overrightarrow{X_{di}}$ consiste à sélectionner, comme nouvelle position, les M composantes de la seconde ligne du vecteur de vitesse $\overrightarrow{V_{di}}(k+1)$ associées aux M plus forts poids de la première ligne du même vecteur, soient les M premières composantes de la seconde ligne de $\overrightarrow{V_{di}}(k+1)$:

$$\overrightarrow{X_{di}}(k+1) = \overrightarrow{V_{di}}(k+1)\,[2, 1:M] \tag{3.3.10}$$

Exemple : Soit un cas d'étude où le nombre de nœuds $N = 5$. Soit une particule di de dimension $M = 3$. La position et la vitesse de la particule, à l'itération k, sont respectivement données par :

$$\overrightarrow{X_{di}}(k) = \begin{bmatrix} 3 & 4 & 1 \end{bmatrix}$$

$$\overrightarrow{V_{di}}(k) = \begin{bmatrix} 1.25 & 1.10 & 1.02 & 0.95 & 0.87 \\ 3 & 4 & 1 & 5 & 2 \end{bmatrix}$$

Supposons, qu'après actualisation, la meilleure position et la meilleure position globale associées à la particule di soient respectivement données par :

$$\overrightarrow{P_{best}^{di}}(k) = \begin{bmatrix} 2 & 5 & 3 \end{bmatrix}$$

$$\overrightarrow{G_{best}^{di}}(k) = \begin{bmatrix} 1 & 2 \end{bmatrix}$$

Soient alors les paramètres de poids $\alpha_{di} = 0.1$, $\beta_{di} = 0.2$ et $\gamma_{di} = 0.4$, la première étape d'actualisation de la vitesse consiste à déterminer les modifications de poids associées à chaque position selon (3.3.5), (3.3.6) et (3.3.7), c'est-à-dire :

$$\overrightarrow{V_{di}}(k) = \begin{bmatrix} (1.25+0.1+0.2) & (1.10+0.1) & (1.02+0.1+0.4) & (0.95+0.2) & (0.88+0.2+0.4) \\ 3 & 4 & 1 & 5 & 2 \end{bmatrix}$$

$$\overrightarrow{V_{di}}(k) = \begin{bmatrix} 1.55 & 1.20 & 1.52 & 1.15 & 1.48 \\ 3 & 4 & 1 & 5 & 2 \end{bmatrix}$$

Introduisant le vecteur :

$$\varphi_{di}(k) = \begin{bmatrix} 0.71 & 0.5 & 0.95 & 0.2 & 0.7 \end{bmatrix}$$

la vitesse est modifiée, selon (3.3.8) :

$$\overrightarrow{V_{di}}(k) * \varphi_{di}(k) = \begin{bmatrix} 1.1 & 0.6 & 1.444 & 0.23 & 1.036 \\ 3 & 4 & 1 & 5 & 2 \end{bmatrix}$$

En tenant compte de (3.3.9), la vitesse est alors actualisée comme :

$$\overrightarrow{V_{di}}(k+1) = \begin{bmatrix} 1.444 & 1.1 & 1.036 & 0.6 & 0.23 \\ 1 & 3 & 2 & 4 & 5 \end{bmatrix}$$

La position de la particule di est alors actualisée, selon (3.3.10), en sélectionnant les $M = 3$ premières composantes de la seconde ligne de $\overrightarrow{V_{di}}(k+1)$, soit :

$$\overrightarrow{X_{di}}(k+1) = \overrightarrow{V_{di}}(k+1)\,[2, 1:3] = \begin{bmatrix} 1 & 3 & 2 \end{bmatrix}$$

3.3.2.2. Modification de la méthode

Afin de mieux explorer l'ensemble du domaine de recherche et d'éviter ainsi un blocage éventuel sur un optimum local, l'algorithme initial a été modifié de la façon suivante :

– **Une variation des paramètres $\alpha_{di}(k)$, $\beta_{di}(k)$ et $\gamma_{di}(k)$**

Ici, en modulant ces paramètres, l'idée sous-jacente est d'éviter que la position de la particule di ne se bloque trop rapidement sur sa meilleure position, $\overrightarrow{P_{best}^{di}}$, ou la meilleure position de ses informatrices, $\overrightarrow{G_{best}^{di}}$. A cette fin, les paramètres $\alpha_{di}(k)$, $\beta_{di}(k)$ et $\gamma_{di}(k)$ sont choisis dépendants de l'itération k de la façon suivante :

$$\alpha_{di}(k) = \alpha_{dii} + (\alpha_{dif} - \alpha_{dii})\frac{k}{N_{iter}} \tag{3.3.11}$$

$$\beta_{di}(k) = \beta_{dii} + (\beta_{dif} - \beta_{dii})\frac{k}{N_{iter}} \tag{3.3.12}$$

$$\gamma_{di}(k) = \gamma_{dii} + (\gamma_{dif} - \gamma_{dii})\frac{k}{N_{iter}} \tag{3.3.13}$$

où N_{iter} est le nombre total d'itérations, α_{dii} et α_{dif} sont les valeurs initiale (à $k = 0$) et finale (à $k = N_{iter}$) de $\alpha_{di}(k)$, β_{dii} et β_{dif} sont les valeurs initiale et finale de $\beta_{di}(k)$ et γ_{dii} et γ_{dif} sont les valeurs initiale et finale de $\gamma_{di}(k)$.

– **L'introduction d'une contribution supplémentaire aléatoire $\varphi_{di_3}(k) \cdot va_{di}$ dans la phase d'actualisation de la vitesse de la particule di**

Cette contribution, en modifiant aléatoirement le poids respectif de chacune des positions possibles, doit pouvoir permettre à certaines positions, de poids plus faible, d'être "revalorisées" et, ainsi, d'être retenues de nouveau. Afin de donner une importance pouvant être variable à cette contribution aléatoire, elle est définie par

$$\varphi_{di_3}(k) \cdot va_{di}(k) = \varphi_{di_3}(k) \cdot (va_{dii} + (va_{dif} - va_{did})\frac{k}{N_{iter}}) \tag{3.3.14}$$

où va_{dii} et va_{dif} sont les valeurs initiale et finale de $va_{di}(k)$ et $\varphi_{di_3}(k)$ est un vecteur de N_i variables aléatoires uniformément distribuées entre 0 et 1. Alors, compte tenu de ces modifications, la première ligne du vecteur de vitesse de la particule di est définie par :

$$\overrightarrow{V_{di}}(k) = \varphi_{di_1}(k) * \overrightarrow{V_{di}}(k) + \varphi_{di_2}(k) * f(\alpha_{di}(k), \beta_{di}(k), \gamma_{di}(k)) + \varphi_{di_3}(k) \cdot va_{di}(k) \tag{3.3.15}$$

où $f(\alpha_{di}(k), \beta_{di}(k), \gamma_{di}(k))$, vecteur de dimension N, représente la somme des **« seules »** contributions de $\alpha_{di}(k)$, $\beta_{di}(k)$ et $\gamma_{di}(k)$ à l'actualisation de la vitesse $\overrightarrow{V_{di}}(k)$, respectivement déterminées par les équations (3.3.5), (3.3.6) et (3.3.7). $\varphi_{di_1}(k)$ et $\varphi_{di_2}(k)$ sont des vecteurs de N variables aléatoires uniformément distribuées entre 0 et 1.

On remarquera que l'imposition de $\varphi_{di_1}(k) = \varphi_{di_2}(k)$ et $va_{di}(k) = 0$ dans l'équation (3.3.15) revient à appliquer l'équation (3.3.8) lors du processus d'actualisation de la vitesse de la particule di, i.e. à appliquer la méthode non modifiée.

Remarque importante. On admettra qu'en fixant les positions et le type des FACTS, il est possible de lancer un algorithme OEP pour optimiser la variable "consignes des FACTS" (i.e. on transforme le problème de placement et dimensionnement de FACTS en un problème "Bi-Level"). Toutefois, le temps de calcul explose en fonction du nombre de particules considéré. En effet, à chaque itération et pour chaque particule, on doit résoudre un sous-problème d'optimisation.

3.3.3. Élaboration d'un algorithme d'optimisation : stable-Lévy-PSO ($\alpha-$SLPSO) et sa recherche locale de Lévy ($\alpha-$SLLS)

Les deux mécanismes fondamentaux qui caractérisent les métaheuristiques sont le mécanisme de diversification et le mécanisme d'intensification. Ces deux processus "non-contradictoires" en amont doivent permettre au processus d'optimisation en aval d'atteindre un état d'équilibre (qui est généralement un optimum d'une fonctionnelle à atteindre). Cette dynamique, qui diffère d'une métaheuristique à une autre, est d'autant plus délicate qu'elle peut entraîner le processus d'optimisation vers un optimum local. L'optimisation par essaim de particules ne fait pas exception à cette règle [Elab09].

Cette caractéristique de convergence prématurée de l'algorithme OEP a incité plusieurs chercheurs de différents domaines à élaborer différentes théories pour réconcilier ces deux dynamiques et donc éviter une convergence prématurée de l'algorithme OEP. Un grand nombre de ces travaux consiste à modifier les règles de mise à jour de la vitesse $\overrightarrow{v}$, de la position $\overrightarrow{x}$ et/ou la topologie des particules informatrices [DasS08].

Dans le cadre de l'amélioration de l'algorithme d'optimisation par essaim de particules de base initié par M. Clerc et al. [Cler07], nous proposons un nouvel algorithme d'optimisation par essaim de particules désigné par $\alpha-$Stable-Lévy-PSO ($\alpha-$SLPSO). Cet algorithme a été conçu à la base pour résoudre le problème de placement et dimensionnement de FACTS. Néanmoins, après avoir constaté que $\alpha-$SLPSO rivalisait bien avec le PSO standard 2007, nous avons proposé d'étendre son champ d'application en le comparant à d'autres algorithmes OEP.

Par ailleurs, une recherche locale de Lévy, désignée par *Stable Lévy Local Search* (SLLS) ou par analogie à $\alpha-$SLPSO, $\alpha-$Stable Lévy Local Search ($\alpha-$SLLS), sera également présentée dans cette partie du travail. Son principe de fonctionnement hérite du même mécanisme que l'algorithme $\alpha-$SLPSO, à l'exception que cette recherche locale ne s'appuie que sur un seul agent (i.e. une seule évaluation par itération), plutôt qu'une population d'agents. Nous avons tenu à la présenter car, même si elle est décorrellée de l'objectif "placement optimal de FACTS", elle constitue une excellente méthode d'optimisation et ouvre de nouvelles perspectives.

3.3.3.1. Élaboration d'un algorithme $\alpha-$Stable-Lévy-PSO ($\alpha-$SLPSO)

Quelle est la façon la plus rapide de trouver une cible cachée? Cette question pertinente est d'une importance vitale pour les animaux qui cherchent à survivre. En effet, la recherche d'un objet caché est l'une des tâches les plus fréquentes des espèces vivantes, comme par exemple trouver de la nourriture, un partenaire sexuel ou un abri, etc. Dans ces exemples, le temps de recherche est généralement un facteur de limitation qui doit être optimisé pour la survie de l'espèce. Ainsi, déterminer la meilleure stratégie de recherche est un problème crucial, qui a inspiré de nombreux travaux expérimentaux et théoriques dans différents domaines (mathématiques appliquées, processus stochastiques, biologie moléculaire...). Un problème d'optimisation n'échappe pas à ces règles. En effet, par analogie, la fonction objectif à optimiser $\overrightarrow{f}$ (l'objet recherché) peut être gloutonne en temps de calcul, ce qui pousse de plus en plus les ingénieurs et les décideurs à définir des algorithmes moins gourmands en temps de calcul.

Des travaux expérimentaux et des observations sur la stratégie de recherche des espèces vivantes (y compris l'être humain) révèlent que celle-ci est généralement intermittente [Beni05], c'est-à-dire que des phases de recherche actives (locales) alternent aléatoirement avec des phases de recherche balistique rapide (sauts), (Fig. 3.3.1).

Cela signifie, dans les termes des métaheuristiques et en particulier dans les techniques d'optimisation par essaim de particules, qu'à n'importe quelle itération t du processus d'optimisation, chaque particule doit être capable, au dépourvu de ses informatrices, d'exploiter d'autres zones prometteuses de l'espace de recherche que celles qui lui ont été promises par la phase d'intensification. En d'autres termes, chaque particule $\overrightarrow{P_i}$ doit effectivement alterner des phases de recherche intensives et des sauts balistiques en intermittence. Il faudra donc transmettre formellement et fidèlement ce comportement à chaque particule de l'essaim. L'un des moyens les plus efficaces et le plus utilisé pour reproduire de l'intermittence consiste à introduire le concept de loi $\alpha-$stable (appelée aussi loi α-stable de Lévy). En effet, on distingue les

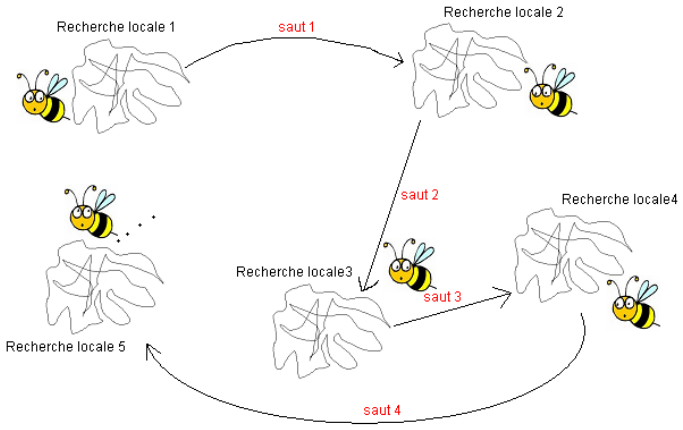


FIGURE 3.3.1.: Principe de fonctionnement de l'algorithme Stable-Lévy-PSO (α-SLPSO)..

mouvements browniens des vols de Lévy par l'absence d'une échelle caractéristique dans ces derniers : le second moment d'ordre de la distribution de Lévy est infini. Cela signifie que des évènements rares sont possibles ou, dans notre cas, de très grands sauts sont possibles. Pour plus de détails sur cette famille de distribution, on se réfèrera au chapitre II.

Ainsi, nous proposons de mettre à jour les positions des particules à l'aide de la loi α-stable de la manière suivante :

$$\overrightarrow{x_i}(t+1) = S_\alpha(\beta,\, \gamma,\, \Delta) \tag{3.3.16}$$

avec α, β, γ et Δ respectivement, l'indice de stabilité, le paramètre d'asymétrie, le paramètre d'échelle et le paramètre de positionnement de la loi α-stable de Lévy. Bien que le paramètre α puisse être adapté à chaque problème d'optimisation, on le fixera dans notre algorithme d'optimisation à $\frac{1}{2}$. En d'autres termes, on mettra à jour les positions des particules à l'aide de la loi de Lévy qui correspond au cas où $\alpha = \frac{1}{2}$. Il est à noter que la loi de Lévy est un cas particulier des lois α-stable de Lévy (chapitre II).

En se référant à l'équation (3.3.16) de [Cham76], le calcul de la nouvelle position de la particule $\overrightarrow{x_i}(t+1)$ à l'itération $t+1$ s'effectue à l'aide des équations (3.3.17) (3.3.18) en posant $\gamma = K \times |r_1\overrightarrow{G_{best_i}}(t) - r_2\overrightarrow{P_{best_i}}(t)|$ et $\Delta = \left(\frac{\overrightarrow{G_{best_i}}(t)+\overrightarrow{P_{best_i}}(t)}{2}\right)$.

$$\overrightarrow{x_i(t+1)} = \left(\frac{\overrightarrow{G_{best_i}}(t) + \overrightarrow{P_{best_i}}(t)}{2}\right) + K \times |r_1\overrightarrow{G_{best_i}}(t) - r_2\overrightarrow{P_{best_i}}(t)| \times y \tag{3.3.17}$$

$$y = \frac{\pi}{2} * \left[bphi * tan(\phi) - \beta * log\left[\frac{\pi}{2} * w * \left(\frac{cos(\phi)}{bphi}\right)\right]\right] + z \tag{3.3.18}$$

en désignant par :

$$w = -log(u)$$

$$\phi = (u - 0.5) * \pi$$

$$z = \beta * tan(\pi * \frac{\alpha}{2})$$

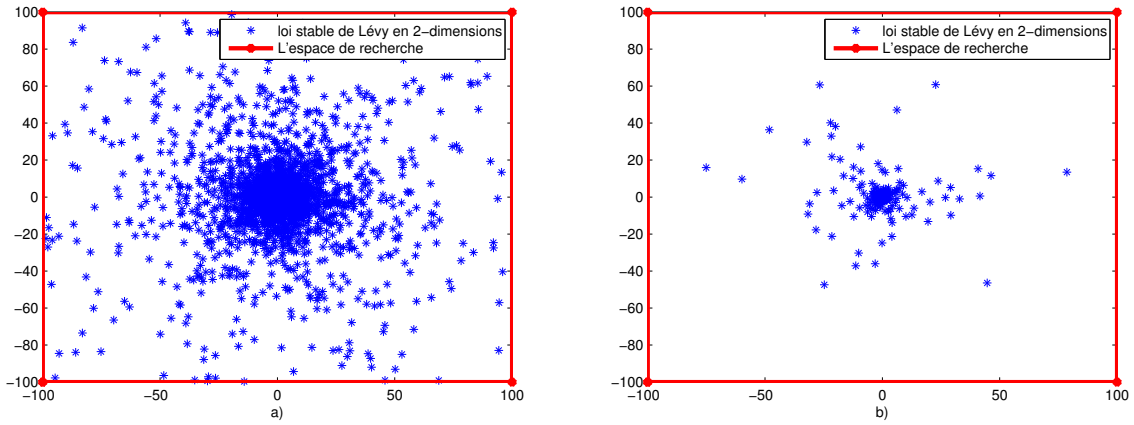


FIGURE 3.3.2.: Loi α-stable de Lévy en 2-dimension pour : a) $\gamma = 1$, b) $\gamma = 0.001$, les autres paramètres sont fixés comme suit : $\alpha = 0.5$, $\beta = 0$ et $\triangle = 50$. L'espace de recherche est fixé à $[-50, +50]$.

$$bphi = \frac{\pi}{2} + \beta * \phi$$

et r_1, r_2 et u sont des variables uniformes entre 0 et 1, $K \in]0, 1]$ un réel positif, $\overrightarrow{P_{best_i}}(t)$ et $\overrightarrow{G_{best_i}}(t)$ sont respectivement la meilleure position de la particule $\overrightarrow{x_i}$ et celle de sa meilleure informatrice à l'instant t. Les équations (3.3.17) et (3.3.18) sont issues, par analogie avec la simulation des lois $\alpha-$stables de Lévy, de la référence [Cham76].

Contrairement aux méthodes classiques OEP, cette méthode n'utilise pas de vitesse de façon explicite. Cependant, la vitesse est implicitement prise en compte dans le paramètre γ de la variable aléatoire S_α c'est-à-dire le terme $|r_1\overrightarrow{G_{best_i}}(t) - r_2\overrightarrow{P_{best_i}}(t)|$ de l'équation (3.3.17). En effet, plus la meilleure position de la particule i, $\overrightarrow{P_{best_i}}(t)$, s'éloigne de la meilleure position de ses informatrices $\overrightarrow{G_{best_i}}(t)$, plus le paramètre γ sera grand ce qui, de fait, augmentera la dispersion des particules (la vélocité de la particule en question sera plus grande). Pour éviter qu'une particule soit figée lorsque γ vaut 0 et pour augmenter la diversité du processus d'optimisation, nous avons ajouté deux composantes aléatoires, r_1 et r_2. Ainsi, la quantité $|r_1\overrightarrow{G_{best_i}}(t) - r_2\overrightarrow{P_{best_i}}(t)|$ converge plus lentement vers zéro et le blocage de la particule est évité lorsque $\overrightarrow{P_{best_i}}(t) \rightsquigarrow \overrightarrow{G_{best_i}}(t)$.

Comme nous le constatons, l'algorithme α-SLPSO possède deux paramètres principaux, K et β. Le paramètre K permet de renormaliser le paramètre d'échelle γ par rapport à l'espace de recherche. Ce paramètre possède une importante propriété, qui consiste à réguler la phase d'intensification. En effet, $|r_1\overrightarrow{G_{best_i}}(t) - r_2\overrightarrow{P_{best_i}}(t)|$ tend vers 0 quand $K \rightsquigarrow 0$. Par conséquent, les réalisations de la variable aléatoire S_α seront concentrées sur un point de l'espace de recherche. En revanche, lorsque $K \succ 0$, les réalisations de la variable aléatoire S_α seront plus éparses (figure 3.3.2). En d'autres termes, le paramètre K contrôle la longueur des sauts effectués par les particules.

Comme nous l'avons déjà indiqué, ce paramètre dépend des valeurs extrêmes X_{max} et X_{min} de l'espace de recherche. En résumé, nous proposons de calculer le paramètre K^j, relatif à la dimension D^j de la particule, de la manière suivante (équation (3.3.19)) :

$$K^j = (X^j_{max} - X^j_{min})/100 \tag{3.3.19}$$

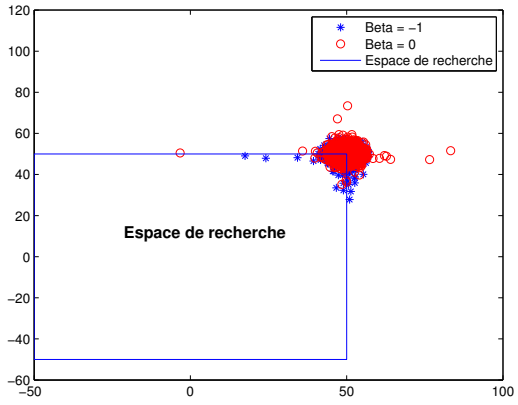


FIGURE 3.3.3.: Loi α-stable de Lévy en 2-dimension pour : $\alpha = 1.95$, $\beta = -1$ (en étoile) et $\beta = 0$ (en rond), le paramètre d'échelle $\gamma = 15$ et le paramètre de positionnement $\triangle = 50$. L'espace de recherche est fixé à $[-50, +50]$.

où X^j_{max} et X^j_{min} sont, respectivement, les valeurs maximale et minimale de la composante D^j de la particule j.

La figure 3.3.3 montre qu'il est judicieux de simuler une variable S_α asymétrique (i.e, $\beta \neq -1$) aux bordures de l'espace de recherche. En effet, il est préférable, pour les particules, de rester dans l'espace de recherche plutôt que d'en sortir. Par conséquent, pour éviter que les particules sortent plus souvent de l'espace de recherche, nous proposons de calculer (à chaque itération du processus d'optimisation) le paramètre $\beta^j_i(t+1)$ correspondant à la composante j de la particule i à l'itération $t+1$, de la manière suivante (équation (3.3.20)) :

$$\beta^j_i(t+1) = -\frac{m}{2(X^j_{max} - X^j_{min})} \tag{3.3.20}$$

où :

- m est le paramètre de positionnement Δ qui vaut : $\left(\frac{\overrightarrow{G_{best_i}} + \overrightarrow{P_{best_i}}}{2}\right)$
- X^j_{max} et X^j_{min} sont respectivement les valeurs maximale et minimale de la composante D^j de la particule j.

Pour assurer que le paramètre d'asymétrie β soit compris entre $[-1, 1]$, nous avons divisé m par deux.

3.3.3.2. Élaboration d'une recherche locale de Lévy (α−SLLS)

Dans presque tous les domaines, que ce soit en sciences de l'ingénieur (biologie, mathématiques, physique, ...) ou en sciences humaines (histoire, sociologie, ...), le concept de lois d'échelle pourrait apporter beaucoup dans la compréhension de certains phénomènes (le chaos, par exemple). Ce concept de loi d'échelle (ou relativité d'échelle, [Nott11]) introduit un espace des échelles, qui est l'espace des résolutions et de leurs transformations (c'est-à-dire des transformations entre échelles).

Nous nous concentrons dans cette section sur la recherche locale de Lévy (α-Stable Lévy Local Search) : α−SLLS.

Une méthode de recherche locale se caractérise essentiellement par une notion de voisinage et un mécanisme d'exploitation de ce dernier [Siar03]. Ces méthodes sont dites *"méthodes de descente"*.

En général, le concept d'un algorithme de recherche locale est le suivant : l'algorithme commence sa recherche à partir d'un point arbitraire dans l'espace des solutions possibles. Puis, dans une zone "locale" autour de ce point de départ, il recherche une ou plusieurs solutions. En d'autres termes, l'algorithme améliore la solution du problème en se déplaçant dans l'espace de recherche d'un voisin à l'autre et ce, en modifiant, souvent légèrement, la solution courante, jusqu'à ce qu'il ne puisse plus améliorer la solution et/ou qu'un certain critère d'arrêt soit atteint (Algorithme 3.2).

Algorithme 3.2 Recherche locale.

```
/* j, j' : solution courante et solution voisine sélectionnées */
/* S : Espace de recherche */
/* V(j) : Voisin de j dans S */
/* val(j) : Fitness de j */
1) Poser j ←solution initiale dans S
2) Poser i ← j
3) Tant que les conditions d'arrêt ne sont pas atteintes faire
4)         Choisir j' = V(j)
5)         Poser j ← j'
6)         Si val(j) < val(i)
7)                 Poser i ← j'
8)         Fin Si
9) Fin Tant que
10) Retourner i
```

Pour construire notre algorithme α-SLPSO, nous avons supposé que la stratégie optimale de recherche des espèces vivantes est généralement intermittente et que des phases de recherche actives (locales) alternent aléatoirement avec des phases de recherche balistiques rapides (sauts).

En s'inspirant de ce qui a été décrit dans la section 3.3.3.1, on peut mettre en œuvre la méthode de recherche locale de Lévy $\alpha-$SLLS de la manière suivante (algorithme 3.3) :

- Initialement, on génère aléatoirement un point s dans l'espace de recherche. Ce point s constitue la solution du problème posé à l'itération 0. En outre, comme décrit dans le chapitre II, la distribution de probabilité de Lévy possède 4 paramètres principaux que l'on initialise comme suit :
  1) $\alpha = \alpha_{min}$ (paramètre d'échelle),
  2) pour chaque dimension D^j de s, on pose $\beta_i^j(0) = -\frac{s(j)}{2(X_{max}^j - X_{min}^j)}$ (paramètre de symétrie),
  3)$\gamma^j = \frac{(X_{max}^j - X_{min}^j)}{2}$ (le paramètre d'échelle) avec X_{max}^j et X_{min}^j les bornes de l'espace de recherche correspondant à la dimension D^j,
  4) le paramètre de positionnement de la distribution $\Delta = s$ (la solution initiale) et enfin $T = T_{max}$ (température de la recherche par analogie avec la température du recuit simulé).

- A l'itération t, on génère une variable aléatoire de Lévy de paramètres :

  -$\alpha = \alpha_{min} + \triangle\alpha * t$, avec $\triangle\alpha = \frac{\alpha_{max} - \alpha_{min}}{Nombre\, d'itérations}$,

  -$\gamma = \left(\frac{x_{max} - x_{min}}{2}\right) e^{\left(\frac{-\triangle f}{T_{max} - \triangle T * t}\right)}$ avec $\triangle f = f(s') - f(s)$ et $f(s')$, la *fitness* obtenue par s' à l'itération t, $\triangle T = \frac{T_{max} - T_{min}}{Nombre\, d'itérations}$. Cette formule est connue sous le nom de règle de Metropolis.

- On arrête l'algorithme si un ou plusieurs critères d'arrêt sont satisfaits. On donne dès lors la meilleure solution s.

Algorithme 3.3 Recherche Locale de Lévy (LLS).

/ * $S_\alpha(\beta, \gamma, s)$: *variable aléatoire de Lévy de paramètres* α, β, γ *et* s * /
/ * s, s' : *solution courante et solution voisine sélectionnées* * /
/ * $f(s)$ *et* $f(s')$: *Fitness de* s *et* s' *respectivement* * /
/ * x_{min}, x_{max} : *l'espace de recherche* * /
/ * t : *l'itération* t * /
1) **Générer** une solution initiale s
2) **Poser**$\alpha \leftarrow \alpha_{max}$;
3) **Poser**$\gamma \leftarrow \left(\frac{x_{max}-x_{min}}{2}\right)$;
4) **Poser**$\beta \leftarrow -\frac{s}{2(X_{max}-X_{min})}$;
5) **Poser**$\triangle\alpha \leftarrow \frac{\alpha_{max}-\alpha_{min}}{Nombre\,d'itérations}$;
6) **Poser**$\triangle T \leftarrow \frac{T_{max}-T_{min}}{Nombre\,d'itérations}$;
6) **Tant que** les conditions d'arrêt ne sont pas atteintes **faire**
7) **Générer** aléatoirement une solution $s' = S_\alpha(\beta, \gamma, s)$
8) **Si** $f(s') < f(s)$ **alors**
9) **Poser** $s \leftarrow s'$,
10) **Poser**$\beta \leftarrow -\frac{s'}{2(X_{max}-X_{min})}$
11) **Fin si**
12) **Poser**$\alpha \leftarrow \alpha_{min} + \triangle\alpha * t$
13) **Poser**$\gamma \leftarrow \left(\frac{x_{max}-x_{min}}{2}\right) e^{\left(\frac{-\triangle f}{T_{max}-\triangle T*t}\right)}$
14) **Poser**$t \leftarrow t + 1$;
15) **Fin tant que**
16) **Retourner** s

Comme dans le recuit simulé [Kirk83], la variation de la température T de T_{max} à T_{min} assure que l'équilibre du système, décrit dans l'algorithme 3.3, ne pourra être atteint que lorsque $\frac{-\triangle f}{T_{max}-\triangle T*t} \rightsquigarrow -\infty$. Par ailleurs, contrairement à l'algorithme $\alpha-$SLPSO plutôt destiné à proposer des solutions optimales globales, $\alpha-$SLLS est vouée à proposer des solutions optimales locales. Pour cette raison, nous avons décidé de faire varier le paramètre α (plutôt que de le fixer à $\frac{1}{2}$) de α_{min} à α_{max}, de manière à ce que les dernières itérations du processus d'optimisation se concentrent plutôt sur la phase d'intensification. En effet, lorsque $\alpha \rightsquigarrow 2$, la loi de la variable aléatoire S_α devient une gaussienne. En d'autres termes, les grands sauts deviennent de plus en plus improbables.

La recherche locale $\alpha-$SLLS diffère du principe général de fonctionnement d'une recherche locale (algorithme 3.2) de deux points de vue :

1. Les voisins s' de la solution courante s sont choisis aléatoirement, suivant une loi α-stable de Lévy.
2. Une solution qui dégrade la *fitness* de la fonction objectif n'est pas retenue et ce, contrairement au recuit simulé, quelle que soit la valeur de cette *fitness*.

En procédant ainsi, nous espérons ne pas nous bloquer sur un optimum local et permettre à l'algorithme d'effectuer de grands sauts (i.e. visiter des zones de recherche prometteuses).

3.3.3.3. Analyse et résultats

Après avoir étayé le principe de fonctionnement de l'algorithme $\alpha-$SLPSO et de sa recherche locale $\alpha-$SLLS, nous allons présenter dans cette section les performances obtenues par ces deux algorithmes. Bien que l'algorithme $\alpha-$SLPSO ait été conçu à l'origine pour résoudre le problème de placement et dimensionnement de FACTS dans des réseaux électriques, il était pertinent de le comparer à d'autres

algorithmes OEP, afin de quantifier ses capacités à résoudre différents problèmes. Pour tester les algorithmes $\alpha-$SLPSO et $\alpha-$SLLS, nous utiliserons les fonctions tests introduites dans l'annexe A.1.

Dans un premier temps, nous validons l'algorithme $\alpha-$SLPSO, en le comparant à cinq autres algorithmes OEP, à savoir PSO-2S, EPUS-PSO, CLPSO, CPSO-H et le standard SPO2007. Les résultats de l'algorithme $\alpha-$SLPSO et de la recherche locale $\alpha-$SLLS sont obtenus par des programmes codés en Python. Les résultats de toutes les autres approches sont tirés de la littérature [ElDo12a, ElDo12b] et [Hsie09]. Par ailleurs, il est à noter que chaque algorithme est exécuté 30 fois sur chaque fonction test. La moyenne et l'écart-type de chaque algorithme sont dès lors évalués sur chaque fonction test. Enfin, nous réalisons la même analyse expérimentale sur la recherche locale $\alpha-$SLLS et comparons avec les algorithmes $\alpha-$SLPSO et SPO2007.

Performances de l'algorithme $\alpha-$SLPSO Afin de comparer l'algorithme $\alpha-$SLPSO aux autres variantes d'OEP, 30 exécutions de chaque algorithme sont effectuées en 10-D et 30-D. Nous enregistrons la moyenne et l'écart-type pour chaque méthode, sur chaque fonction test, dans un tableau. Un critère d'arrêt commun à tous les algorithmes est fixé, à savoir que les processus d'optimisation s'arrêtent lorsqu'un nombre maximal d'évaluations Max_{Fes} est atteint. En se référant aux standards de la conférence CEC2005, Max_{Fes} est en fonction de la dimension du problème à résoudre, à savoir, $Max_{Fes} = 10000 \times D$. Par conséquent, le critère d'arrêt Max_{Fes} est fixé à 40000 et 150000, en 10-D et 30-D, respectivement.

Les paramètres des algorithmes PSO-2S, EPUS-PSO, CLPSO, CPSO-H et standard PSO2007 sont issus de [Hsie09]. Les paramètres de notre algorithme $\alpha-$SLPSO sont fixés comme suit :

- le nombre de particules $Nb_{particules}$ est fixé à 20 (souvent utilisé dans la littérature),
- le nombre d'informatrices $N_{informatrices}$ est fixé à 3 (souvent utilisé dans la littérature),
- la topologie des particules est un modèle social. C'est-à-dire qu'à chaque itération de l'algorithme, on choisit aléatoirement les informatrices de chaque particule (on espère une meilleure phase de diversification),
- le paramètre α est fixé à $\frac{1}{2}$ (cf section 3.3.3.1),
- le paramètre K est pris selon l'équation (3.3.19) (cf section 1.2.4.5),
- en cas de sortie de l'espace de recherche, les particules sont remises aléatoirement dans ce dernier (pour une meilleure phase de diversification).

Résultats des problèmes en 10-D Les tableaux 3.1 et 3.2 présentent les résultats obtenus par α-SLPSO, PSO-2S, EPUS-PSO, CLPSO, CPSO-H et le standard PSO2007, en termes d'erreurs moyennes et d'écarts-types pour 30 exécutions de 11 fonctions tests parmi les 15 présentées en Annexe A.1. La dimension des fonctions utilisée est, dans tous les cas, égale à 10. Les meilleurs résultats parmi les six algorithmes sont présentés en rouge. Par ailleurs, l'algorithme α-SLPSO est testé sur 3 fonctions supplémentaires, mais pas comparées, faute de résultats sur les autres algorithmes. La performance dans la dernière ligne de chaque tableau indique le nombre de fois (de fonctions) où l'algorithme α-SLPSO surpasse chaque autre algorithme.

L'examen de ces résultats expérimentaux, dans le cas des problèmes en 10-D, amène les commentaires suivants :

- Les algorithmes $\alpha-$SLPSO, PSO-2S et EPUS-PSO semblent obtenir de meilleurs résultats que le reste des variantes d'OEP. En effet, $\alpha-$SLPSO, PSO-2S et EPUS-PSO surpassent le reste des algorithmes OEP sur la majorité des fonctions tests. En particulier, α-SLPSO, PSO-2S et EPUS-PSO obtiennent des résultats significativement meilleurs que le reste des algorithmes sur f_5, f_6 et f_{14}.

f	α-SLPSO	PSO-2S	EPUS-PSO	CLPSO	CPSO-H	SPSO2007
	$2,73e-113$	$1,05e-086$	$5,55e-153$	$4,02e-021$	$9,01e-010$	$4,00e-101$
f_1	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$
	$1.23e-112$	$3,10e-086$	$2,44e-134$	$6,49e-021$	$1,38e-009$	$1,80e-100$
	$7,66e-003$	$6,18e-028$	$9,32e-012$	$5,04e+002$	$1,62e+003$	$1,83e-027$
f_2	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$
	$1,24e-002$	$2,13e-027$	$4,16e-011$	$1,99e+002$	$9,35e+002$	$4,03e-027$
	$1.41e+000$	$2,35e-002$	$9,13e-002$	$2,84e+000$	$1,08e+000$	$3,10e-001$
f_3	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$
	$3,95e-001$	$1,56e-002$	$1,05e-001$	$1,05e+000$	$1,18e+000$	$9,95e-001$
	$0,00e+000$	$4,12e-015$	$2,72e-015$	$1,78e-011$	$7,02e-006$	$3,85e-001$
f_4	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$
	$0,00e+000$	$1,21e-015$	$1,50e-015$	$1,63e-011$	$5,77e-006$	$2,07e-001$
	$3.19e-014$	$0,00e+000$	$0,00e+000$	$2,35e-009$	$3,83e-010$	$5,11e+000$
f_5	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$
	$1.46e-013$	$0,00e+000$	$0,00e+000$	$2,72e-009$	$7,36e-010$	$2,29e+000$
	$0,00e+000$	$5,85e-003$	$3,21e-002$	$2,06e+000$	$4,12e+000$	$4,04e-002$
f_6	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$
	$0,00e+000$	$2,35e-002$	$8,11e-002$	$4,80e-001$	$1,72e+000$	$1,78e-001$
	$6,68e-002$	$-$	$-$	$-$	$-$	$-$
f_8	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$
	$5,47e-002$	$-$	$-$	$-$	$-$	$-$
Performance		$\frac{3}{6}$	$\frac{2}{6}$	$\frac{6}{6}$	$\frac{5}{6}$	$\frac{4}{6}$

TABLE 3.1.: Comparaison des différents algorithmes en 10-D : fonctions standards.

- L'algorithme PSO-2S semble surpasser tous les autres algorithmes. En effet, le score de ce dernier est de 6 /11 et 8/11 par rapport à $\alpha-$SLPSO, et EPUS-PSO respectivement.
- PSO-2S et EPUS-PSO semblent surpasser notre algorithme $\alpha-$SLPSO. Le score du $\alpha-$SLPSO est de 5/11 par rapport à PSO-2S et de 4/11 par rapport à EPUS-PSO.
- Enfin, nous remarquons que α-SLPSO est surpassé par PSO-2S et EPUS-PSO sur l'ensemble des fonctions test unimodales ($f1$, f_2 et f_{10}).

Résultats des problèmes en 30-D Les résultats expérimentaux pour les problèmes en 30-D sont présentés dans les tableaux 3.3 et 3.4. Comme en 10-D, nous comparons les algorithmes par rapport à l'erreur moyenne et l'écart-type sur 30 exécutions, et ceci, en utilisant les mêmes fonctions que dans le cas de 10-D.

L'examen de ces résultats expérimentaux amène les commentaires suivants :

- Comme dans le cas de 10-D, les algorithmes PSO-2S et EPUS-PSO semblent obtenir de meilleurs résultats que le reste des variantes du PSO. En outre, PSO-2S surpasse tous les algorithmes, excepté $\alpha-$SLPSO, dans la majorité des fonctions tests.
- Contrairement au cas précédent (10-D), $\alpha-$SLPSO obtient de meilleurs résultats que tous les autres algorithmes. En effet, son score est de 7/11 par rapport aux deux algorithmes PSO-2S et EPUS-PSO.
- Comme dans le cas précédent, α-SLPSO est surpassé par PSO-2S et EPUS-PSO sur l'ensemble des fonctions tests unimodales ($f1$, f_2 et f_{10}).
- Enfin, α-SLPSO semble être moins sensible à l'augmentation de la dimension D, notamment, sur les fonctions multimodales f_4, f_5, f_6, f_8, f_{10}, f_{13}, et f_{15} .

En résumé, α-SLPSO donne de très bons résultats sur l'ensemble des fonctions tests en 10-D et 30-D (tableau 3.5). Ces résultats nous montrent que l'algorithme est plus performant lorsque la dimension D de la fonction à optimiser augmente. En d'autres termes, α-SLPSO semble être plus adapté aux problèmes de grandes dimensions (tableau 3.6), ce qui est le cas du placement et dimensionnement de FACTS.

f	α-SLPSO	PSO-2S	EPUS-PSO	CLPSO	CPSO-H	SPSO2007
	$9.67e-113$	-	-	-	-	-
f_s	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$
	$2.70e-112$	-	-	-	-	-
	$2,99e-003$	$3,38e-009$	$3,99e-007$	$8,27e+002$	$2,15e+003$	$1,02e-007$
f_{10}	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$
	$4,79e-003$	$9,02e-009$	$1,63e-006$	$3,35e+002$	$1,19e+003$	$4,12e-007$
	$3.73e+000$	$4,21e-001$	$8,96e-001$	$6,44e+000$	$3,01e+000$	$5,73e-001$
f_{11}	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$
	$1.65e+000$	$7,77e-002$	$2,15e+000$	$4,56e+000$	$2,52e+000$	$1,05e+000$
	$4.62e+000$	$1,56e+001$	$2,61e-015$	$8,45e-008$	$1,31e+000$	$2,03e+001$
f_{12}	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$
	$4.17e+000$	$2,28e+000$	$1,57e-015$	$2,97e-007$	$9,68e-001$	$8,54e-002$
	$1.3e+000$	$2,65e-001$	$7,26e+000$	$8,85e+000$	$2,81e+001$	$1,26e+001$
f_{13}	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$
	$4.60e+000$	$9,93e-001$	$4,31e+000$	$3,07e+000$	$1,39e+001$	$7,00e+000$
	$4,32e-001$	$6,43e-001$	$1,12e+000$	$1,98e+000$	$3,48e+000$	$4,20e+000$
f_{14}	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$
	$3,56e-001$	$3,45e-001$	$1,36e+000$	$5,38e-001$	$1,41e+000$	$1,50e+000$
	$4,53e-001$	–	-	-	-	-
f_{15}	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$
	$2,51e-001$	–	–	-	-	-
Performance		$\frac{2}{5}$	$\frac{2}{5}$	$\frac{4}{5}$	$\frac{3}{5}$	$\frac{3}{5}$

TABLE 3.2.: Comparaison des différents algorithmes en 10-D : fonctions pivotées.

f	α-SLPSO	PSO-2S	EPUS-PSO	CLPSO	CPSO-H	SPSO2007
	$2.36e-135$	$3,02e-119$	$8,50e-263$	$1,34e-025$	$1,57e-010$	$3,36e-107$
f_1	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$
	$1.19e-134$	$1,56e-118$	$1,02e-262$	$1,71e-025$	$2,99e-10$	$1,79e-106$
	$1,20e-001$	$6,18e-011$	$1,97e+002$	$2,41e+004$	$8,36e+004$	$8,25e-008$
f_2	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$
	$2,71e-001$	$2,63e-011$	$4,69e+000$	$6,78e+003$	$6,58e+004$	$9,00e-008$
	$5.50e+000$	$9,75e+000$	$2,55e+001$	$2,01e+001$	$1,03e+001$	$1,23e+001$
f_3	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$
	$7.77e+000$	$9,59e-001$	$3,53e-001$	$3,37e+000$	$1,37e+001$	$1,47e+000$
	$3.12e-015$	$1,63e-001$	$3,91e-015$	$9,90e-014$	$3,90e-006$	$1,36e+000$
f_4	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$
	$8.61e-016$	$3,85e-001$	$1,07e-015$	$3,80e-014$	$5,98e-006$	$1,13e+000$
	$5.92e-017$	$1,00e+000$	$0,00e+000$	$1,87e-009$	$3,15e-010$	$4,88e+001$
f_5	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$
	$3.18e-016$	$1,18e+001$	$0,00e+000$	$5,34e-009$	$1,05e-009$	$1,44e+001$
	$0.00e+000$	$3,63e-001$	$4,08e-001$	$1,49e+001$	$1,36e+000$	$1,55e+001$
f_6	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$
	$0.00e+000$	$2,79e-001$	$1,22e+000$	$1,69e+000$	$3,38e+000$	$1,09e+000$
	$6,97e-003$	-	-	-	-	-
f_8	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$	$\pm$
	$8,99e-003$	-	-	-	-	-
Performance		$\frac{5}{6}$	$\frac{4}{6}$	$\frac{6}{6}$	$\frac{6}{6}$	$\frac{5}{6}$

TABLE 3.3.: Comparaison des différents algorithmes en 30-D : fonctions standard.

f	SLPSO	PSO-2S	EPUS-PSO	CLPSO	CPPSO-H	SPSO2007
f_s	$6.87e-072$ ± $3.28e-071$	- ± -	- ± -	- ± -	- ± -	- ± -
f_{10}	$1.95e+000$ ± $2.73e+000$	$7,80e-005$ ± $9,25e-005$	$1,20e+001$ ± $2,46e+001$	$2,38e+004$ ± $6,49e+003$	$3,87e+004$ ± $1,71e+004$	$1,09e-001$ ± $1,80e-001$
f_{11}	$1,56e+001$ ± $1.02e+000$	$1,39e+001$ ± $1,48e-001$	$1,96e+001$ ± $2,91e+001$	$3,04e+001$ ± $1,32e+001$	$3,03e+001$ ± $2,59e+001$	$2,77e+001$ ± $1,31e+001$
f_{12}	$3.12e+000$ ± $3.33e+000$	$1,69e+001$ ± $1,66e+000$	$6,81e-001$ ± $1,02e+000$	$2,22e-006$ ± $8,29e-006$	$1,52e+000$ ± $8,30e-001$	$2,09e+000$ ± $1,31e+000$
f_{13}	$3.19e-010$ ± $1.68e-009$	$1,40e+000$ ± $2,73e-001$	$3,76e+001$ ± $1,48e+001$	$4,70e+001$ ± $6,85e+000$	$9,52e+001$ ± $2,93e+001$	$9,04e+001$ ± $4,03e+001$
f_{14}	$5.00e+000$ ± $3.13e+000$	$3,34e+000$ ± $1,22e+000$	$4,26e+000$ ± $2,97e+000$	$1,51e+001$ ± $1,93e+000$	$1,36e+001$ ± $3,56e+000$	$3,27e+001$ ± $3,19e+000$
f_{15}	$1,83e-002$ ± $2,41e-002$	- ± -	- ± -	- ± -	- ± -	- ± -
Performance		$\frac{2}{5}$	$\frac{3}{5}$	$\frac{4}{5}$	$\frac{4}{5}$	$\frac{3}{5}$

TABLE 3.4.: Comparaison des différents algorithmes en 30-D : fonctions pivotées.

f	PSO-2S	EPUS-PSO	CLPSO	CPPSO-H	SPSO2007
10D	$5/11$	$4/11$	$10/11$	$8/11$	$8/11$
30D	$7/11$	$7/11$	$10/11$	$10/11$	$8/11$

TABLE 3.5.: Comparaison des différents algorithmes.

Pour consolider ces analyses expérimentales et éviter d'en tirer des conclusions hâtives, nous proposons d'effectuer une analyse multicritère des différents résultats obtenus dans les tableaux 3.1, 3.2, 3.3 et 3.4. En effet, si l'on suppose que l'algorithme α-SLPSO surpasse l'algorithme PSO-2S dans 6 fonctions tests sur 11, que PSO-2S surpasse EPUS-PSO dans 8 fonctions sur 11 et que EPUS-PSO surpasse à son tour α-SLPSO dans 6/11, il est difficile de trancher entre les trois méthodes.

Pour ce faire, la méthode d'aide à la décision ELECTRE II est appliquée sur les résultats obtenus par chaque algorithme pour chaque fonction test. Son principe de fonctionnement consiste à calculer une matrice de concordance et discordance pour ensuite "surclasser" (faiblement et fortement) les alterna-

f	f_{21}	f_{22}	f_{23}	f_{24}	f_{25}
50-D	$1,32e-004$ ± $4,43e-005$	$5,16e+01$ ± $1,78e+001$	$2,10e-003$ ± $4,07e-004$	$2,30e+001$ ± $4,76e+000$	$2,67e-002$ ± $4,68e-002$
100-D	$5,52e-003$ ± $1.27e-002$	$1,62e+002$ ± $4,54e+001$	$8,22e-003$ ± $3,37e-003$	$6,11e+001$ ± $1,40e+001$	$1,27e-002$ ± $1,80e-002$
200-D	$1.35e-001$ ± $6.04e-001$	$2,89e+002$ ± $6,28e+001$	$2,86e-002$ ± $9,20e-002$	$1,52e+002$ ± $1,76e+001$	$7,75e-003$ ± $1,47e-002$

TABLE 3.6.: Résultats en dimensions 50-D, 100-D et 200-D.

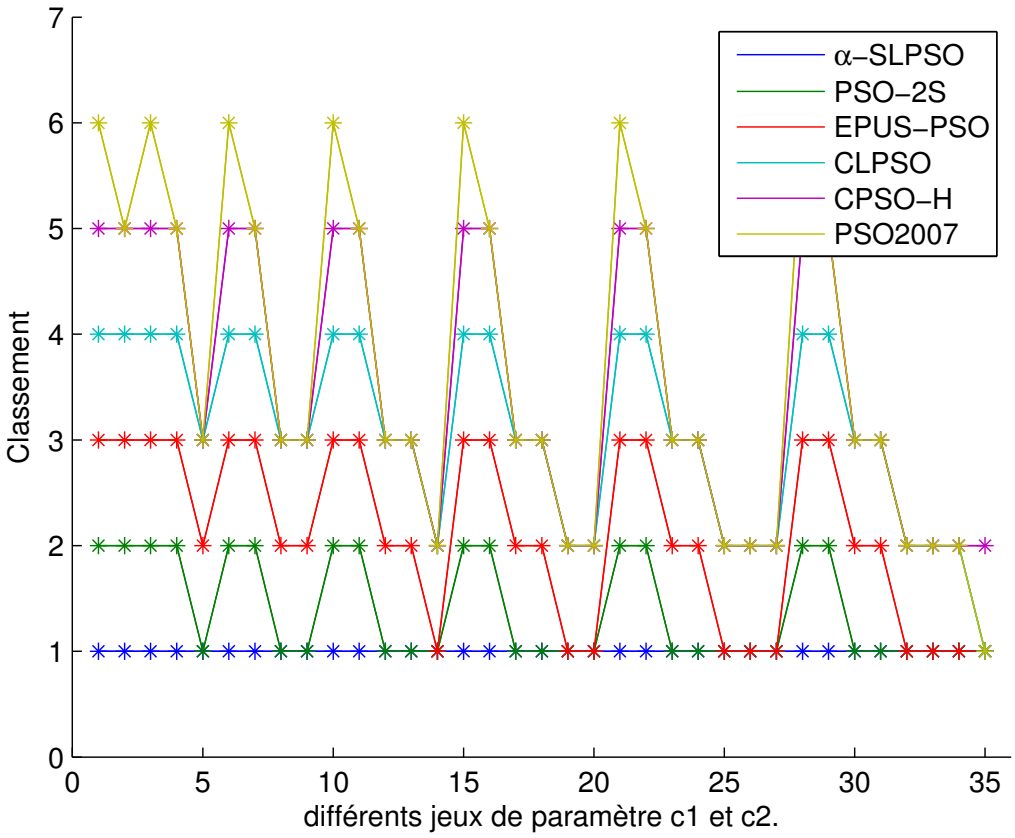


FIGURE 3.3.4.: Classement des variantes α-SLPSO, PSO-2S, EPUS-PSO, CLPSO, CPSO-H et standard PSO2007, pour un jeu de paramètres de concordance c_1 et c_2.

tives entre elles [RoyB68]. Par manque de données sur les résultats des algorithmes PSO-2S, EPUS-PSO, CLPSO, CPSO-H et le standard PSO2007, nous ne retiendrons que la moyenne des *fitness* comme critère de classement, soit, 22 critères (6 fonctions standards et 5 fonctions pivotées en dimensions 10 et 30). Par ailleurs, bien que toutes les fonctions n'aient pas la même complexité, nous avons décidé d'affecter à chacune d'entre elles le même poids.

Afin de classer les différentes variantes d'OEP, nous avons programmé la méthode ELECTRE II sous Matlab et nous l'avons appliquée à notre problème de classement pour différents jeux de paramètres c_1 et c_2 ($c_1 > c_2$ sont des seuils de concordance dans la méthode ELECTRE II). La figure 3.3.4 montre les résultats du classement obtenus pour les différents jeux de paramètres. Cette figure montre clairement que α-SLPSO surclasse - dans la majorité des cas - toutes les autres méthodes. Pour certaines valeurs de c_1 et c_2 , cette figure montre aussi que α-SLPSO, PSO-2S et EPUS-PSO sont indiscernables.

Performances de la recherche locale $\alpha-$SLLS Afin de présenter les résultats de notre recherche locale $\alpha-$SLLS et de les comparer à l'algorithme standard PSO2007, 100 exécutions de la recherche locale sont effectuées en 10-D et 30-D. Comme dans le cas précédent, nous enregistrons la moyenne et l'écart-type de chaque méthode sur chaque fonction test dans un tableau. Le critère d'arrêt commun à tous les algorithmes Max_{Fes} est toujours le même : 40 000 et 150 000 en 10-D et 30-D respectivement.

Les paramètres de notre algorithme de recherche locale $\alpha-$SLLS sont fixés comme suit :

– les températures maximale T_{max} et minimale T_{min} sont fixées à 10 et 0 respectivement,

	10-D			30D		
f	α-SLLS	α-SLPSO	SPSO2007	α-SLLS	α-SLPSO	SPSO2007
f_1	$6.54e-030$ $\pm$ $2.10e-029$	$2,73e-113$ $\pm$ $1.23e-112$	$4,00e-101$ $\pm$ $1,80e-100$	$4.95e-043$ $\pm$ $1,95e-042$	$2.36e-135$ $\pm$ $1.19e-134$	$3,36e-107$ $\pm$ $1,79e-106$
f_2	$4,3e-004$ $\pm$ $2,15e-004$	$7,66e-003$ $\pm$ $1,24e-002$	$1,83e-027$ $\pm$ $4,03e-027$	$7,41e+000$ $\pm$ $1,01e+000$	$1,20e-001$ $\pm$ $2,71e-001$	$8,25e-008$ $\pm$ $9,00e-008$
f_3	$3.61e+000$ $\pm$ $2,11e+000$	$1.41e+000$ $\pm$ $3,95e-001$	$3,10e-001$ $\pm$ $9,95e-001$	$2,59e+000$ $\pm$ $5,69e+000$	$5.50e+000$ $\pm$ $7.77e+000$	$1,23e+001$ $\pm$ $1,47e+000$
f_4	$1,72e+000$ $\pm$ $8,43e-001$	$0,00e+000$ $\pm$ $0,00e+000$	$3,85e-001$ $\pm$ $2,07e-001$	$4.81e+000$ $\pm$ $2.09e+000$	$3.12e-015$ $\pm$ $8.61e-016$	$1,36e+000$ $\pm$ $1,13e+000$
f_5	$3,15e+001$ $\pm$ $1,19e+001$	$3.19e-014$ $\pm$ $1.46e-013$	$5,11e+000$ $\pm$ $2,29e+000$	$1,23e+002$ $\pm$ $2,42e+001$	$5.92e-017$ $\pm$ $3.18e-016$	$4,88e+001$ $\pm$ $1,44e+001$
f_6	$5.11e+000$ $\pm$ $1,60e+000$	$0,00e+000$ $\pm$ $0,00e+000$	$4,04e-002$ $\pm$ $1,78e-001$	$2,81e+001$ $\pm$ $3,46e+000$	$0.00e+000$ $\pm$ $0.00e+000$	$1,55e+001$ $\pm$ $1,09e+000$
f_8	$4,85e-001$ $\pm$ $3,36e-001$	$6,68e-002$ $\pm$ $5,47e-002$	- $\pm$ -	$1,37e-002$ $\pm$ $1,31e-002$	$6,97e-003$ $\pm$ $8,99e-003$	- $\pm$ -

TABLE 3.7.: Résultats obtenus par la recherche locale α-SLLS en 10D et 30-D sur les fonctions standard.

- les paramètres d'échelle α_{max} et α_{min} sont fixés à 2 et $\frac{1}{2}$ respectivement.
- en cas de sortie de l'espace de recherche, une remise aléatoire à l'intérieur de ce dernier est effectuée.

Résultats des problèmes en 10-D et 30-D Les résultats expérimentaux pour les problèmes en 10-D et 30-D de notre recherche locale de Lévy α-SLLS sont présentés dans le tableau 3.7. Ces résultats nous montrent que la recherche locale α-SLLS obtient des résultats très encourageants, plus particulièrement pour les fonctions unimodales f_1 et f_2 et dans toutes les fonctions multimodales, exceptée la fonction f_5. Par ailleurs, nous remarquons que la recherche locale α-SLLS surpasse les deux algorithmes dans la fonction f_3 en 30-D.

3.3.4. "Hybridation" des algorithmes OEP continu et discret pour application au positionnement et au dimensionnement de FACTS

Placement et dimensionnement de FACTS ne peuvent être décorrélés. En effet, le choix de la position optimale d'un FACTS est nécessairement couplé à celui de sa dimension et réciproquement. Il apparaît donc pertinent de faire évoluer de façon coordonnée les particules associées à la recherche de la dimension, i.e. les particules *ci* continues et celles associées à la recherche de la position, i.e. les particules *di* discrètes.

L'approche retenue ici est d'établir une sorte de "hiérarchie" entre ces deux types de particules, en donnant la priorité à la recherche de la position optimale. La procédure retenue repose alors sur les principes suivants :

- la dimension d'une particule *di* est égale au nombre de FACTS, Nb_{FACTS}, à implanter ;
- la dimension de la particule *ci*, associé à la particule *di*, est égale au nombre total de nœuds, Nb_{noeud}, où un FACTS peut être placé ;
- la modification des particules suit la procédure suivante :
  - la position des Nb_{FACTS} FACTS associés à la particule *di* est déterminée ;

Algorithme 3.4 Algorithme hybride pour le placement de FACTS.

1. **Initialisation** des différents paramètres associés au problème d'optimisation
2. **Initialisation**, de façon aléatoire, dans tout l'espace de recherche, des vitesses et positions associées aux $Nb_{particule}$ particules, où $Nb_{particule}$ est le nombre total de particules
3. **Pour** chaque itération k **faire**
   a) **Mise à jour**, si nécessaire, des paramètres de réglage associés au problème d'optimisation
   b) **Pour** chaque particule i (i.e. particules di et ci) **faire**
      - A partir des positions $X_{di}(k)$ et $X_{ci}(k)$, **calcul** de la fonction de *fitness* $J_i(k)$)
      - **Actualisation** de $B_{di}(k)$ et $B_{ci}(k)$
   c) **Fin Pour**
   d) **Pour** chaque particule i **faire**
      - **Actualisation** de $G_{di}(k)$ et $G_{ci}(k)$
      - **Actualisation** de la vitesse $V_{di}(k+1)$
      - **Actualisation** de la position $X_{di}(k+1)$
      - **Actualisation** de la vitesse $V_{ci}(k+1)$
      - **Actualisation** de la position $X_{ci}(k+1)$
   e) **Fin Pour**
4. **Fin Pour**
5. **Détermination** de la meilleure solution, parmi les $Nb_{particule}$ $G_{di}(k)$ et $G_{ci}(k)$

– seuls les éléments de la particule *ci* (i.e. les nœuds) désignés par la particule *di* sont modifiés.

En liant ainsi une variable "dimension de FACTS" à chaque nœud possible, et en ne modifiant cette variable que lorsque la position du nœud correspondant est sélectionnée, une telle approche revient à fixer un FACTS, dont la dimension va évoluer en fonction des positions retenues, en chacun des nœuds possibles du réseau et à ne retenir en final que Nb_{FACTS} nœuds (i.e. Nb_{FACTS} FACTS). En résumé, l'algorithme OEP "hybride" reposant sur les principes décrits ci-dessus s'appuie sur les étapes décrites dans l'algorithme 3.4.

3.4. Application dans le cas mono objectif

3.4.1. Placement et dimensionnement de deux FACTS

Cette section est consacrée à la mise en œuvre de l'algorithme OEP hybride proposé, dans le cas de deux réseaux électriques tests :

– le réseau *benchmark* IEEE 30 Bus (ou 30 nœuds),
– le réseau *benchmark* IEEE 57 Bus (ou 57 nœuds).

Dans un premier temps, l'étude sur le réseau 30 nœuds permet de valider l'algorithme mis en œuvre. Cette validation est effectuée en comparant les optima retenus par l'OEP avec un "optimum" calculé par une exploration systématique de tout l'espace de recherche, basée sur une discrétisation de ce dernier.

L'application au réseau 57 nœuds complète cette validation sur un réseau plus étendu. L'influence de quelques paramètres de réglage de l'algorithme OEP sur les performances obtenues en termes de convergence est illustrée sur ce réseau. De même, l'influence, sur la solution optimale, des paramètres économiques associés à la résolution du modèle du réseau par une méthode OPF est illustrée.

Pour les deux réseaux, les paramètres de simulation et de réglage initiaux de l'algorithme OEP sont :
- $\alpha_{ii} = 0.2$, $\alpha_{if} = 0.1$ pour toute particule i
- $\beta_{ii} = 0.1$, $\beta_{if} = 0.2$ pour toute particule i
- $\gamma_{ii} = 0.1$, $\gamma_{if} = 0.5$ pour toute particule i
- $va_{ii} = 0.5$, $va_{if} = 0.5$ pour toute particule i
- $c_{i1}(k) = c_{i2}(k) = 0.975$ pour toute particule i et toute itération k (souvent utilisés)
- $w_i(k) = w_{ii} + (w_{if} - w_{cii})\frac{k}{N_{iter}}$ avec $w_{ii} = 1.5$ et $w_{if} = 0.5$ pour toute particule i
- Limites sur la puissance des FACTS (position X_i des particules i) :
  - $X_{cim_{min}} = -50\,MVAR$ pour toute composante m et toute particule i
  - $X_{cim_{max}} = \ \ 50\,MVAR$ pour toute composante m et toute particule i
  - En cas de dépassement des limites haute ou basse sur x_{cim} : imposition de $v_{cim} = 0$ et réinitialisation aléatoire x_{cim} pour toute composante m et toute particule i.
- Limites sur la vitesse $\overrightarrow{V_{ci}}$ des particules ci :
  - $V_{cim_{min}} = \frac{(X_{cim_{min}} - X_{cim_{max}})}{2\,\Delta t} = -50$ pour toute composante m et toute particule i ($\Delta t = 1$)
  - $V_{cim_{max}} = \frac{(X_{cim_{max}} - X_{cim_{min}})}{2\,\Delta t} = \ \ 50$ pour toute composante m et toute particule i ($\Delta t = 1$)
- Nombre de particules : $Nb_{particule} = 50$
- Nombre de particules informatrices : $N_{inform_{ci}} = N_{inform_{di}} = 10$ pour toute particule i
- Nombre d'itérations : $N_{iter} = 200$
- Nombre de FACTS à implanter : $Nb_{FACTS} = 2$
- Type de FACTS à implanter : STATCOM, c'est-à-dire modification de la demande de puissance réactive au nœud de connexion du STATCOM (pas de modification de la matrice d'admittance Y)
- Critère économique (équation (3.2.7)) sur la puissance active P lors de la résolution OPF des équations du réseau.

Le choix des paramètres de poids associés à α_{di}, β_{di} et γ_{di} est effectué afin de donner un poids relatif plus important à la position de la particule (paramètre α_{di}) en début d'itérations et à la meilleure position "globale" de la particule (paramètre γ_{di}) en fin d'itérations. De plus, une composante aléatoire, d'amplitude maximale $va_{di} = 0.5$, est introduite dans le processus de mise à jour de la vitesse $\overrightarrow{V_{di}}$ des particules discrètes $\overrightarrow{X_{di}}$.

3.4.1.1. Application au réseau IEEE 30 nœuds

La topologie du réseau 30 nœuds est donnée sur la figure 3.4.1. Il est composé de 30 nœuds dont 6 nœuds “générateur”. Ayant déjà la possibilité de faire varier la puissance réactive injectée sur ces nœuds “générateur” par l'intermédiaire des alternateurs, ces nœuds sont éliminés de l'espace de recherche. Cet espace est donc restreint à 24 nœuds “charge” parmi lesquels il convient de déterminer ceux où seront implantés les deux FACTS. Le critère à minimiser retenu est le critère J_3 (i.e. équilibrage des tensions aux nœuds, équation (3.2.10)).

Les paramètres et limites associés aux différentes particules $\overrightarrow{X_{ci}}$ et $\overrightarrow{X_{di}}$ sont :
- $Nb_{noeud} = 24$,
- Dimension d'une particule $\overrightarrow{X_{ci}}$: $Z = Nb_{noeud} = 24$ pour toute particule i,
- Dimension d'une particule $\overrightarrow{X_{di}}$: $M = Nb_{FACTS} = 2$ pour toute particule i.

Afin de déterminer, de façon systématique, une solution “optimale”, l'espace de recherche est discrétisé de la façon suivante :

- Examen de tous les couples possibles de positions, soit 276 couples,
- Exploration entre $X_{cim_{min}} = -50\,MVAR$ et $X_{cim_{max}} = 50\,MVAR$ par pas de $2\,MVAR$, soit 51 pas.

La recherche de "l'optimum" systématique repose donc sur $276 * 51 * 51 = 717\,876$ évaluations du critère à optimiser (i.e. résolutions du modèle du réseau par la méthode OPF).

L'OEP étant une approche de type stochastique, l'évaluation de ses performances est faite sur un jeu de plusieurs essais (passages de l'algorithme). La validation de l'algorithme proposé se fera alors aussi à travers sa capacité à reproduire l'obtention de la solution optimale ou d'une solution suffisamment "proche" de la solution optimale.

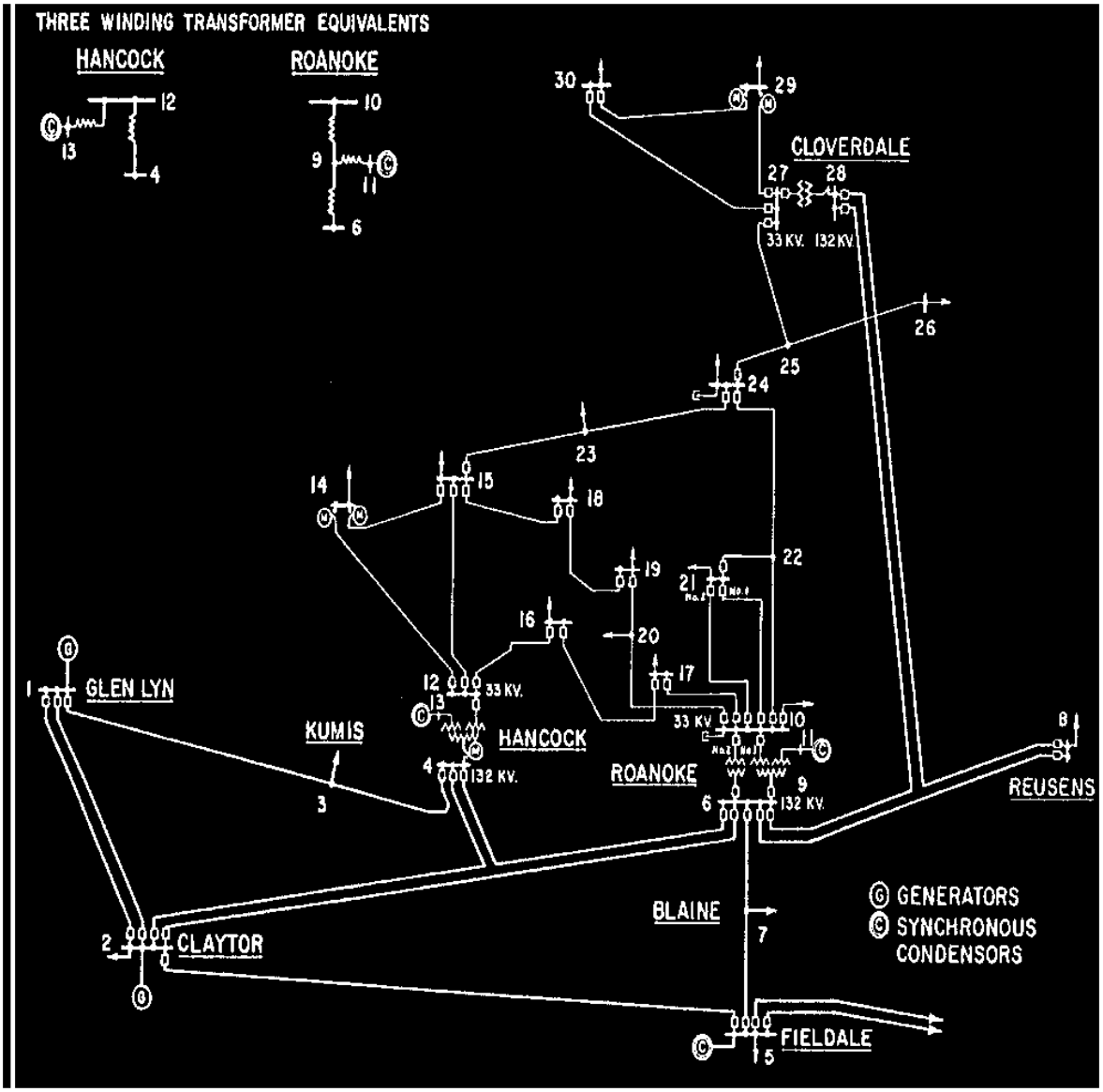


FIGURE 3.4.1.: Topologie du réseau IEEE 30 Bus [IEEE30].

Les figures 3.4.2 et 3.4.3 montrent respectivement l'optimum obtenu, les nœuds sélectionnés et la puissance réactive associée à ces nœuds pour 30 passages de l'algorithme OEP proposé. A titre de comparaison, la valeur du critère obtenue sans FACTS est également donnée sur la figure 3.4.2. L'examen de

ces courbes amène les commentaires suivants :

- L'introduction de FACTS dans le réseau permet bien d'améliorer l'équilibrage des tensions aux nœuds, et ainsi d'accroître la sécurité du système.

- Les optima trouvés par l'algorithme OEP sont tous très proches de "l'optimum" déterminé par l'approche systématique. En montrant ainsi une convergence de façon très répétitive vers la solution "optimale", ce résultat met en exergue la consistance de l'algorithme OEP. Cet aspect est confirmé par l'examen de la figure 3.4.3 où les positions sélectionnées par l'approche OEP sont très souvent celles correspondant à la solution "optimale" systématique. En particulier, le nœud N°8 est sélectionné par OEP pour tous les passages. Quand ce n'est pas le cas, les nœuds sélectionnés par OEP sont "proches" des nœuds "optimaux", ce qui, compte tenu de la numérotation "géographique" des nœuds (figure 3.4.1), indique une action dans la même zone d'influence. Bien que moins nettement "identiques" aux valeurs de puissance trouvées par l'approche systématique, les puissances sélectionnées par l'algorithme OEP varient autour des valeurs "optimales" (figure 3.4.3), ce qui indique une bonne répétitivité de l'algorithme.

- L'algorithme OEP nécessite $N_{iter} * Nb_{particule}$, soit ici $50 * 200 = 10\,000$ évaluations du critère à optimiser pour obtenir, à chaque passage, une solution très proche de la solution "optimale" systématique. Ce nombre est à mettre en relation avec les 717 876 évaluations du critère qu'a nécessité l'approche systématique, mettant ainsi en exergue l'intérêt d'avoir recours à un algorithme d'optimisation, ici de type OEP, pour la détermination d'une solution satisfaisante avec une forte réduction (ici de $98,6\%$) du nombre d'évaluations du critère, et donc du temps de calcul correspondant.

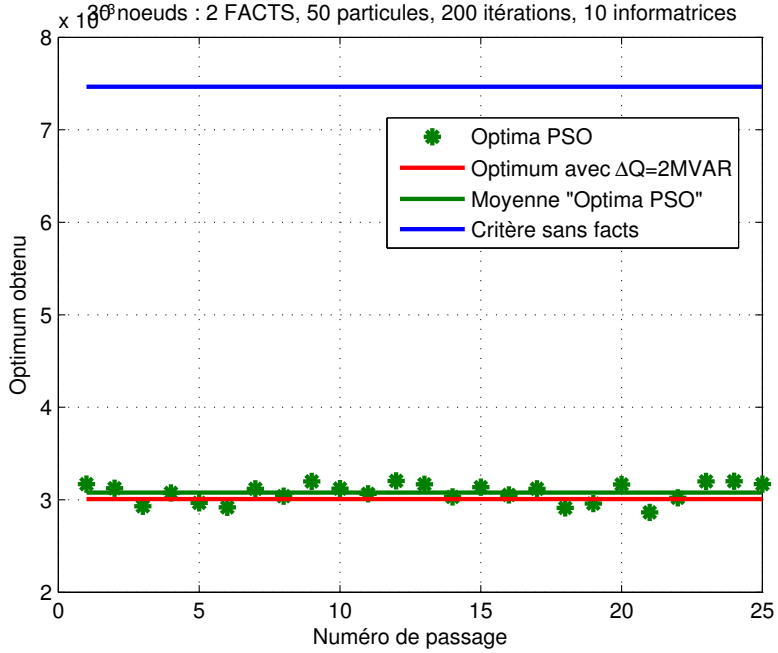


FIGURE 3.4.2.: Réseau IEEE 30 nœuds, 50 particules, 200 itérations, 10 informatrices. Comparaison du critère : paramètres de simulation et de réglage initiaux

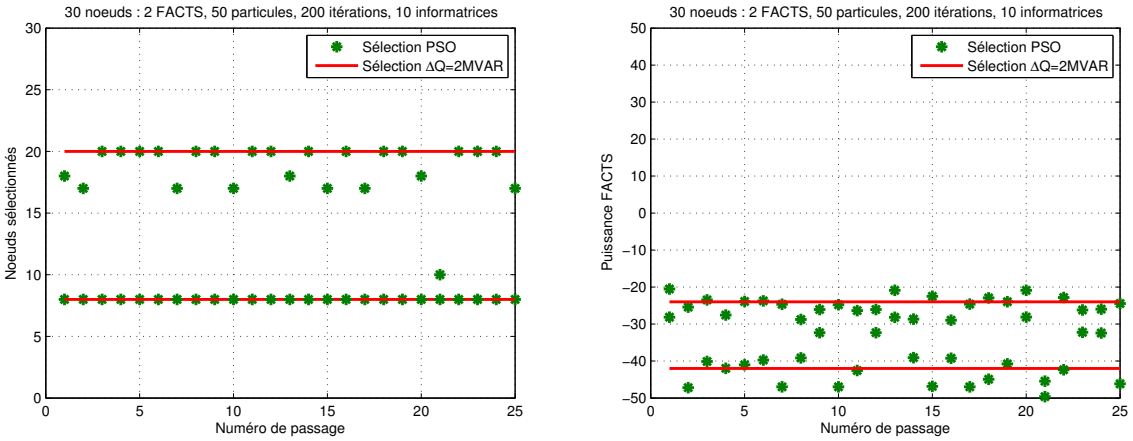


FIGURE 3.4.3.: Réseau IEEE 30 nœuds, 50 particules, 200 itérations, 10 informatrices. Comparaison des positions et puissances sélectionnées : paramètres de simulation et de réglage initiaux

3.4.1.2. Application au réseau IEEE 57 nœuds

La topologie du réseau 57 nœuds est donnée en figure 3.4.4. A l'instar du cas du réseau 30 nœuds, les nœuds "générateur" (ici au nombre de 7) sont retirés de l'espace de recherche.

Les paramètres et limites associés aux différentes particules $\overrightarrow{X_{ci}}$ et $\overrightarrow{X_{di}}$ sont :
- $Nb_{noeud} = 50$,
- dimension d'une particule $\overrightarrow{X_{ci}}$: $Z = Nb_{noeud} = 50$ pour toute particule i,
- dimension d'une particule $\overrightarrow{X_{di}}$: $M = Nb_{FACTS} = 2$ pour toute particule i.

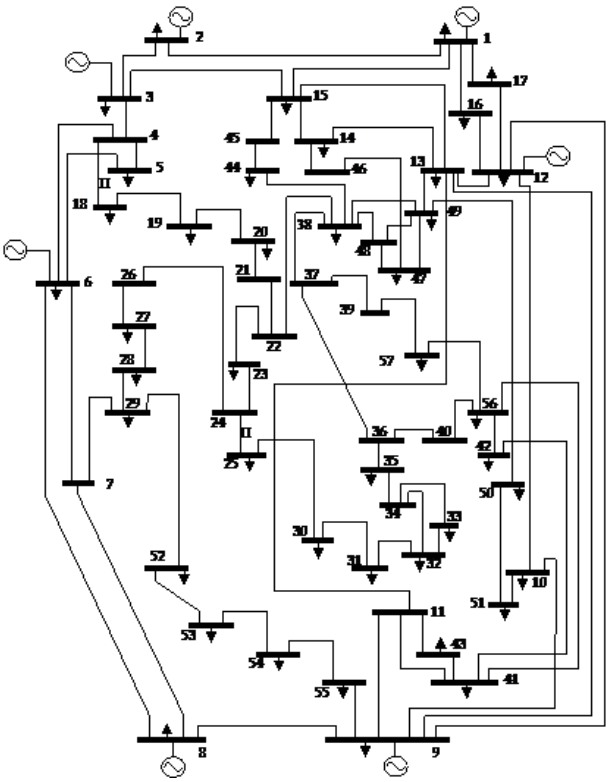


FIGURE 3.4.4.: Topologie du réseau IEEE 57 Bus [Gasb09].

Ce cas d'étude est intéressant car le nombre de nœuds possible est le double du cas précédent. Il en résulte un nombre de couples de positions beaucoup plus élevé (i.e. 1 225) et donc, un espace de recherche combinatoire de plus grande dimension. Les résultats obtenus doivent donc qualifier, entre autres, l'aptitude de l'algorithme OEP proposé à la recherche d'un optimum dans un espace plus large. Comme dans le cas du réseau 30 nœuds, les résultats obtenus sont comparés à ceux déterminés par une approche systématique.

Afin de déterminer cette solution "optimale" systématique, l'espace de recherche est discrétisé de la façon suivante :

- examen de tous les couples possibles de positions, soit 1 225 couples,
- exploration entre $X_{cim_{min}} = -50\,MVAR$ et $X_{cim_{max}} = 50\,MVAR$, par pas de $5\,MVAR$, soit 21 pas.

La recherche de "l'optimum" systématique repose donc sur $1225 * 21 * 21 = 540\,225$ évaluations du critère à optimiser (i.e. résolution du modèle du réseau par la méthode OPF).

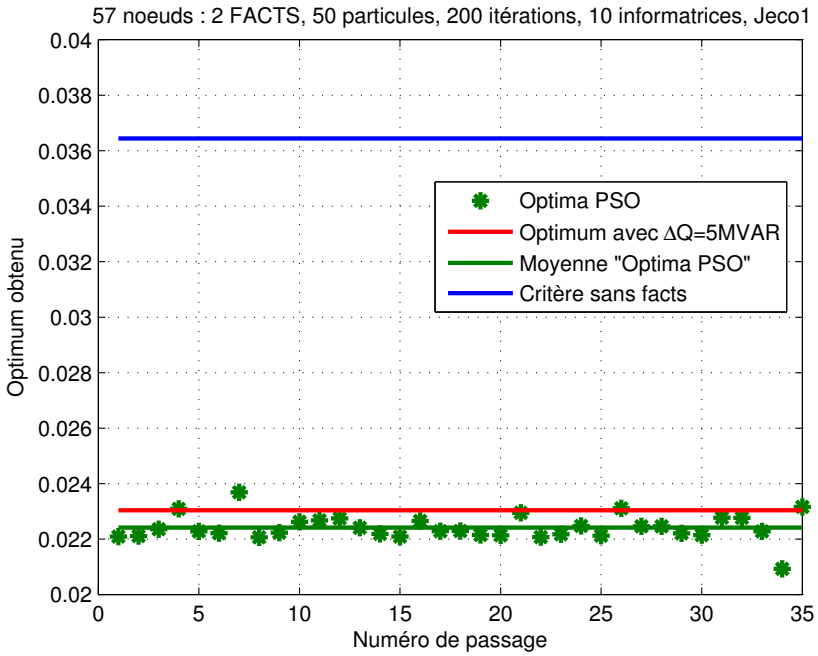


FIGURE 3.4.5.: Réseau IEEE 57 nœuds, 50 particules, 200 itérations, 10 informatrices. Comparaison du critère : paramètres de simulation et de réglage initiaux

Les figures 3.4.5 et 3.4.6 montrent respectivement l'optimum obtenu, les nœuds sélectionnés et la puissance réactive associée à ces nœuds pour 30 passages de l'algorithme OEP proposé. A titre de comparaison, la valeur du critère obtenue sans FACTS est également donnée sur la figure 3.4.5. L'examen de ces courbes amène les commentaires suivants :

- L'algorithme OEP propose des optima très proches, voire "meilleurs" (67% des cas), que "l'optimum" systématique, qui ne correspond qu'à une "approximation", par une approche discrétisée et par "pas" assez élevés de $5\,MVAR$, d'un optimum réel (figure 3.4.5). Un tel résultat vient confirmer les performances de l'algorithme OEP, déjà validées dans le cas du réseau 30 nœuds.

- La consistance de l'algorithme OEP proposé est encore mise en exergue par la figure 3.4.6 à travers

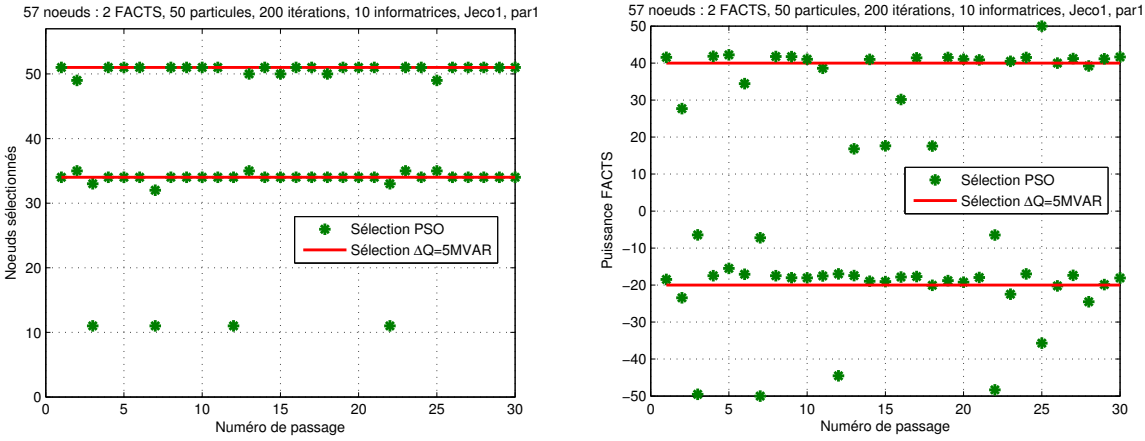


FIGURE 3.4.6.: Réseau IEEE 57 nœuds, 50 particules, 200 itérations, 10 informatrices
Comparaison des positions et puissances sélectionnées : paramètres de simulation et de réglage initiaux

une sélection de nœuds et de puissance associées "quasi-constante", autour de la solution "optimale" systématique, pour les 30 passages effectués, et ce dans un espace de recherche plus étendu.

– L'algorithme OEP nécessite ici 10 000 évaluations du critère pour obtenir une solution "optimale". Il est intéressant ici de mettre en parallèle ce nombre avec les 540 225 évaluations du critère qu'a nécessité l'approche systématique, à travers une évaluation des temps de calcul respectifs. Ainsi, compte tenu de la durée moyenne d'une évaluation du critère égale à 0,6 s, la recherche de la solution systématique a nécessité 324 135 sec, soit environ 90 heures (3,75 jours ! !), à rapprocher du temps de calcul pour un passage OEP égal à 6 000 s, soit 1h40mn (i.e. une réduction d'environ 98% du temps de calcul).

Pertinence de la méthode "Velocity Likehood" modifiée (section 3.3.2.2)

Si l'approche OEP continue (associée ici à la recherche de la puissance optimale des FACTS) est une technique reconnue et souvent utilisée, l'application d'une méthode OEP discrète "Velocity Likehood" à la recherche de la position optimale des FACTS peut poser question, notamment quant à l'assurance d'une exploration suffisamment complète de l'espace de recherche. Afin d'accroître l'étendue de cette exploration, une composante aléatoire va_i a été ajoutée lors de la mise à jour de la vitesse de la particule $\overrightarrow{V_{di}}$ (équation (3.3.15)). Afin de juger de l'influence de la composante va_i sur les performances de l'algorithme OEP proposé, elle a été supprimée pour un ensemble de 30 passages de l'algorithme.

Lors de ces essais, les paramètres initiaux de réglage et de simulation sont modifiés, à savoir $va_{ii} = 0$, $va_{if} = 0$, pour toute particule i.

Les figures 3.4.7 et 3.4.8 montrent respectivement l'optimum obtenu, les nœuds sélectionnés et la puissance réactive associée à ces nœuds pour les 30 passages réalisés.

L'examen des figures 3.4.7 et 3.4.8 montre une plus grande dispersion des optima obtenus, avec seulement 23% des optima "meilleurs" que l'optimum systématique. Une même dispersion se retrouve sur les positions et puissances sélectionnés. L'ajout d'une composante aléatoire va_i lors de la mise à jour de la vitesse $\overrightarrow{V_{di}}$ apporte donc une amélioration significative des performances de l'algorithme OEP mis en œuvre. En permettant une meilleure exploration de l'ensemble de l'espace de recherche, la composante va_i évite un blocage trop rapide sur un optimum "local" et ainsi une meilleure recherche d'un optimum

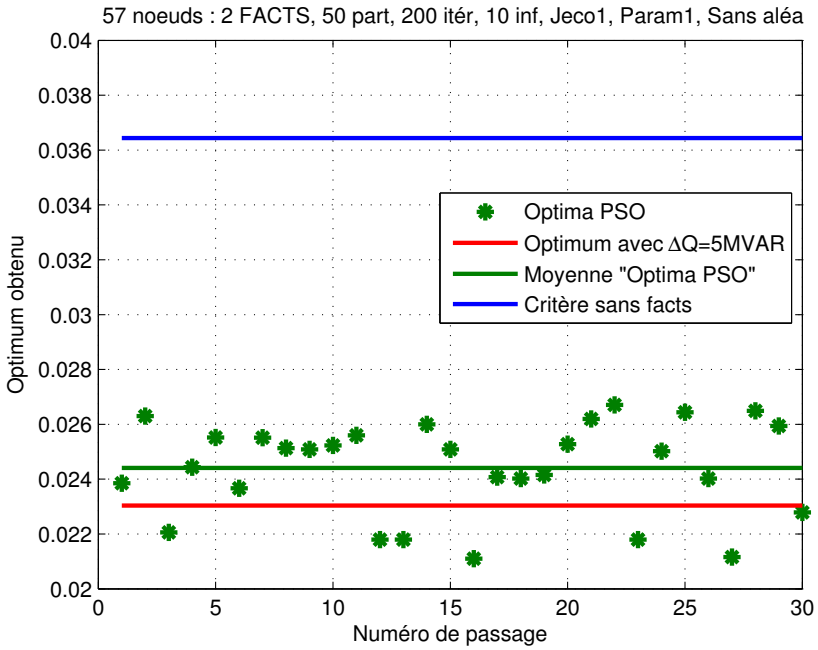


FIGURE 3.4.7.: Réseau IEEE 57 nœuds, 50 particules, 200 itérations, 10 informatrices
Comparaison du critère : annulation de la composante aléatoire va_{di}

global.

Influence des limites hautes et basses sur la vitesse $\overrightarrow{V_{ci}}$ des particules ci

Dans la section précédente, il a été montré l'importance d'une "bonne" exploration de l'ensemble de recherche associé à la position des FACTS. Il en va de même pour la recherche de la puissance optimale à associer aux positions sélectionnées. Le déplacement de la position $\overrightarrow{X_{ci}}$ des particules ci dépend essentiellement de la vitesse $\overrightarrow{V_{ci}}$ de ces particules. Plusieurs paramètres entrent en jeu lors de l'évolution de cette vitesse. Plus particulièrement, l'amplitude des limites haute et basse sur cette vitesse (i.e. $V_{cim_{max}}$ et $V_{cim_{min}}$) peut modifier la façon dont l'espace de recherche est exploré. Ainsi, une amplitude limite trop grande pourrait générer des gradients de déplacement trop élevés dont la conséquence serait une exploration assez "discontinue" de l'espace de recherche. A contrario, une amplitude limite trop faible pourrait confiner le déplacement d'une particule dans un espace "trop" restreint autour de sa position initiale. L'influence de ces limites de vitesse est illustrée à travers un ensemble de 30 passages de l'algorithme avec une réduction de l'amplitude des limites sur la vitesse $\overrightarrow{V_{ci}}$.

Lors de ces essais, les paramètres initiaux de réglage et de simulation ont été modifiés, c'est-à-dire les limites sur la vitesse $\overrightarrow{V_{ci}}$ des particules ci :

- $V = \frac{(X_{cim_{min}} - X_{cim_{max}})}{5\,\Delta t} = -20$ pour toute composante m et toute particule i ($\Delta t = 1$),
- $V = \frac{(X_{cim_{max}} - X_{cim_{min}})}{5\,\Delta t} = \quad 20$ pour toute composante m et toute particule i ($\Delta t = 1$).

Les figures 3.4.9 et 3.4.10 montrent respectivement l'optimum obtenu, les nœuds sélectionnés et la puissance réactive associée à ces nœuds pour les 30 passages réalisés. L'examen de ces courbes montre clairement une amélioration significative des performances de l'algorithme OEP. Ainsi, l'optimum OEP est meilleur que l'optimum systématique dans 90% des cas (à comparer avec les 67% du premier test (cf figure 3.4.5)). Plus remarquable encore est la consistance de l'algorithme quant à la sélection des posi-

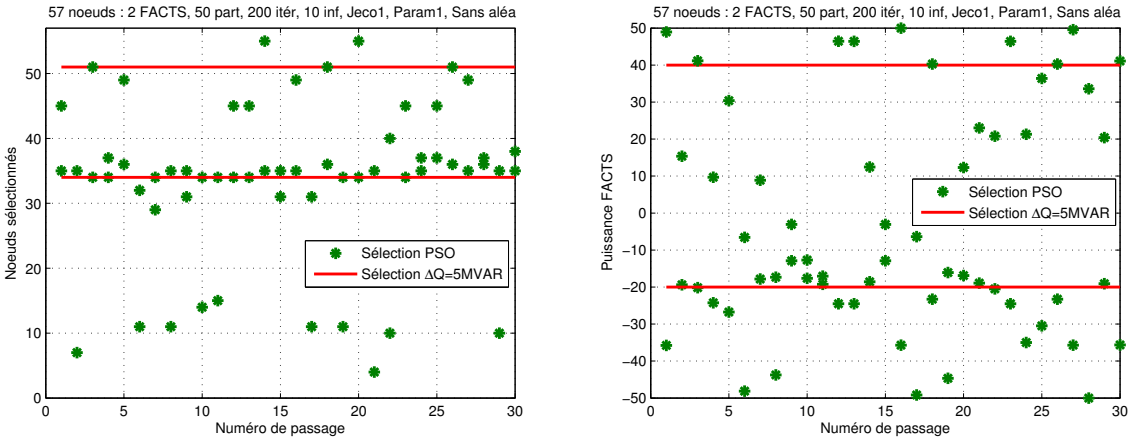


FIGURE 3.4.8.: Réseau IEEE 57 nœuds, 50 particules, 200 itérations, 10 informatrices
Comparaison des positions et puissances sélectionnées : annulation de la composante aléatoire va_{di}.

tions de FACTS (cf figure 3.4.10). Une amélioration du même type peut être observée sur le choix des puissances associées aux nœuds sélectionnés.

Les deux exemples précédents de modification du réglage des paramètres d'optimisation mettent en lumière l'importance du choix des dits paramètres. Ils mettent aussi en avant la question du réglage "optimal" de ces paramètres. Cette question est, à ce jour, encore ouverte et devra faire l'objet de travaux complémentaires. En règle générale, cette problématique est récurrente dans la mise en œuvre d'un grand nombre de techniques d'optimisation de type évolutionnaire (OEP, Algorithme Génétique,...) dont le comportement se fonde sur une exploration stochastique "dirigée" de l'espace de recherche.

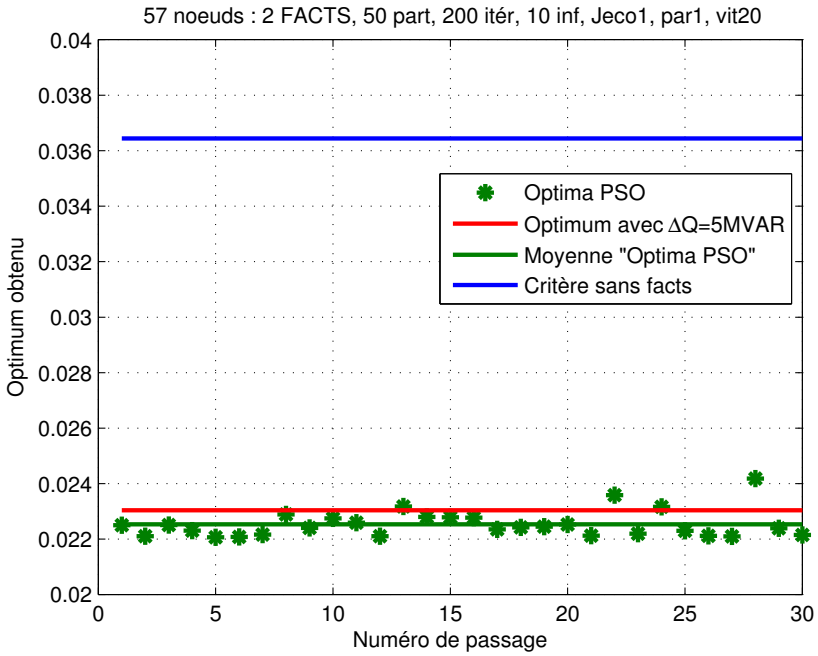


FIGURE 3.4.9.: Réseau IEEE 57 nœuds, 50 particules, 200 itérations, 10 informatrices
Comparaison du critère : réduction des limites max et min sur la vitesse $\overrightarrow{V_{ci}}$

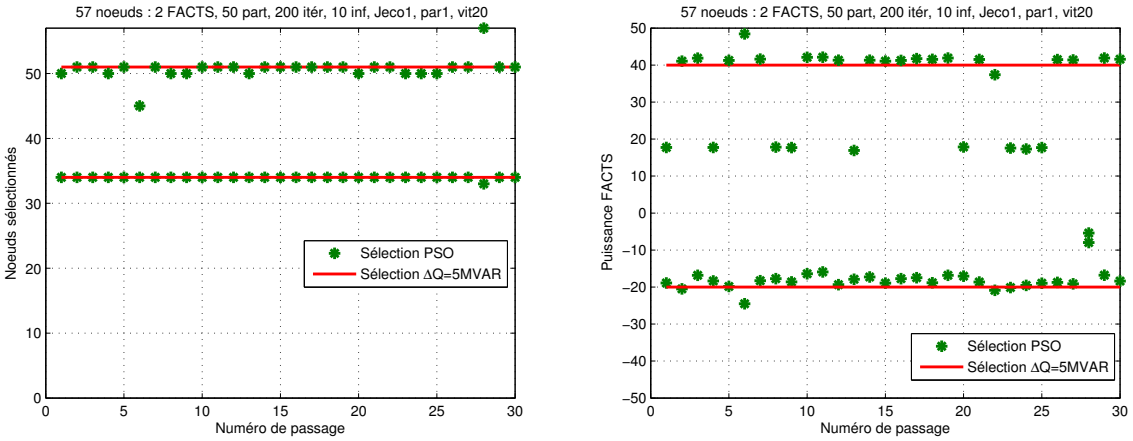


FIGURE 3.4.10.: Réseau IEEE 57 nœuds, 50 particules, 200 itérations, 10 informatrices
Comparaison des positions sélectionnées : réduction des limites max et min sur la vitesse $\overrightarrow{V_{ci}}$

Modification des conditions de fonctionnement du réseau électrique

La résolution OPF des équations du réseau donne une solution en régime permanent du réseau. Cette solution satisfait la minimisation d'une fonctionnelle de coût de puissance (équation (3.2.7)), tout en respectant les contraintes du réseau. Elle représente ainsi une "image" du réseau dans des conditions de fonctionnement données. Le résultat du placement optimal et du dimensionnement de FACTS dépendra donc de ces conditions de fonctionnement. Afin d'illustrer cette dépendance, un nouveau critère de coût est retenu pour un ensemble de 30 passages de l'algorithme OEP, à savoir un critère économique (équation (3.2.7)) sur les puissances active P et réactive Q lors de la résolution OPF des équations du réseau.

Les figures 3.4.11 et 3.4.12 montrent respectivement l'optimum obtenu, les nœuds sélectionnés et la puissance réactive associée à ces nœuds pour les 30 passages réalisés. A titre de comparaison, la valeur du critère obtenue sans FACTS est également donnée sur la figure 3.4.11. Ces résultats montrent que la solution optimale, i.e. nœuds 11 et 33, est différente de celle du cas précédent, mettant ainsi clairement en évidence la dépendance de cette solution vis-à-vis des conditions de fonctionnement du réseau. L'analyse des performances de l'algorithme OEP montre que celles-ci ne sont pas modifiées par le changement de conditions de fonctionnement. En effet, avec les mêmes réglages des paramètres d'optimisation, ces performances sont du même niveau que celles observées lors d'essais précédents (cf figures 3.4.5 et 3.4.6), ceci en termes de réalisation d'un optimum OEP meilleur que "l'optimum" systématique (67% des cas réalisés) et de reproductibilité des résultats (i.e. consistance de l'algorithme OEP).

3.4.2. Placement et dimensionnement de plusieurs FACTS de types différents (cas général)

Nous souhaitons maintenant placer et dimensionner des FACTS dont le nombre et le type sont inconnus dans le réseau 57 nœuds, afin de minimiser la somme des différences de tension des nœuds (critère J_3). Dans tous les cas, sauf indication contraire, les paramètres de l'algorithme sont fixés comme suit :

- $Taille\,de\,l'essaim$: 20,
- $Nombre\,d'informatrices$: 3,
- $Nombre\,d'itérations$: 500.

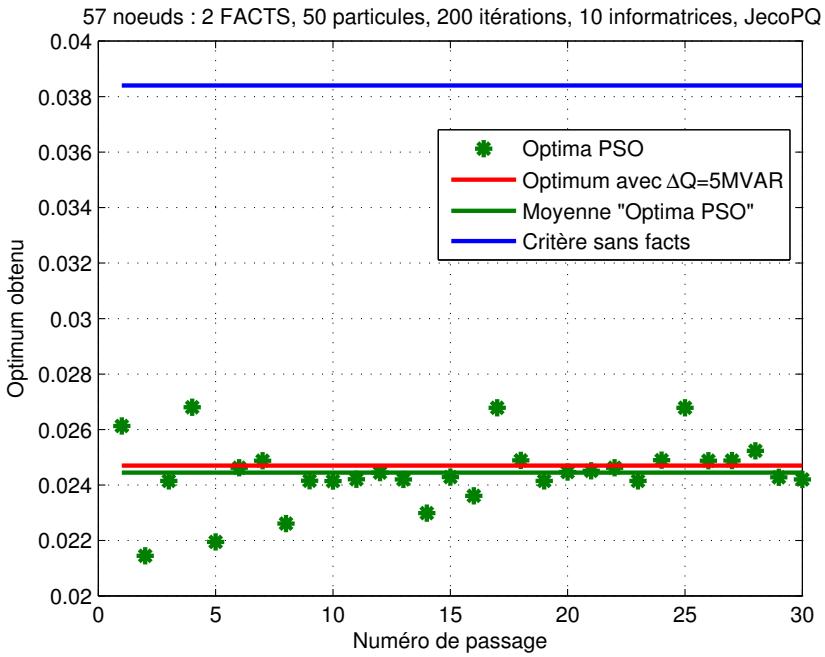


FIGURE 3.4.11.: Réseau IEEE 57 nœuds, 50 particules, 200 itérations, 10 informatrices
Comparaison du critère : modification du critère économique sur les puissances active P et réactive Q

Les résultats de 15 réalisations de l'algorithme PSO2007 et $\alpha-$SLPSO sont donnés dans les figures 3.4.13 a-b) et 3.4.13 c-d) respectivement. Ces dernières montrent que les deux algorithmes mis en œuvre sont consistants (convergent souvent vers une solution unique). On notera néanmoins une meilleure consistance pour l'algorithme $\alpha-$SLPSO. En effet, les variances obtenues par $\alpha-$SLPSO et PSO2007 sont respectivement de $1,39 \times 10^{-7}$ et $8,19 \times 10^{-7}$.

On remarque que les résultats obtenus par $\alpha-$SLPSO et PSO2007 sont légèrement meilleurs que dans le cas où l'on fixe le type des FACTS à 2 STATCOM (figure 3.4.9). En effet, la moyenne des *fitness* est de $0,0132$ dans le cas général (i.e. le type et le nombre de FACTS sont libres), soit une réduction de 65% du critère J_3, contre 0.0225 quand on fixe les FACTS à 2 STATCOM, soit une réduction de près de 40% du critère J_3. On peut penser en effet qu'une plus grande liberté en termes de nombre, position et type des FACTS permet une meilleure répartition spatiale de la puissance réactive dans le réseau. Cette puissance agissant principalement de façon locale, il en résulte une meilleure uniformisation du plan de tension.

De plus, on remarque que les résultats obtenus par PSO2007 sont légèrement moins bons que dans l'approche $\alpha-$SLPSO. En effet, la moyenne de 15 réalisations du PSO2007 est de 0.0151, soit une réduction de 59% du critère J_3, contre 65% de réduction du critère J_3, pour l'algorithme $\alpha-$SLPSO.

Au delà des performances légèrement supérieures en terme de *fitness*, la différence majeure entre les algorithmes $\alpha-$SLPSO et PSO2007 concerne la convergence de ces deux algorithmes. En effet, la figure 3.4.14 montre que, dans les premières itérations du processus d'optimisation (les 120 premières itérations), les deux approches convergent avec la même vitesse. Au delà de la $120^{ème}$ itération, l'approche $\alpha-$SLPSO converge plus vite que PSO2007. Par ailleurs, l'approche $\alpha-$SLPSO converge en moyenne après 300 itérations, alors que le PSO2007 quant à lui converge après 500 itérations, ce qui représente un gain de 40% sur la vitesse de convergence de l'algorithme.

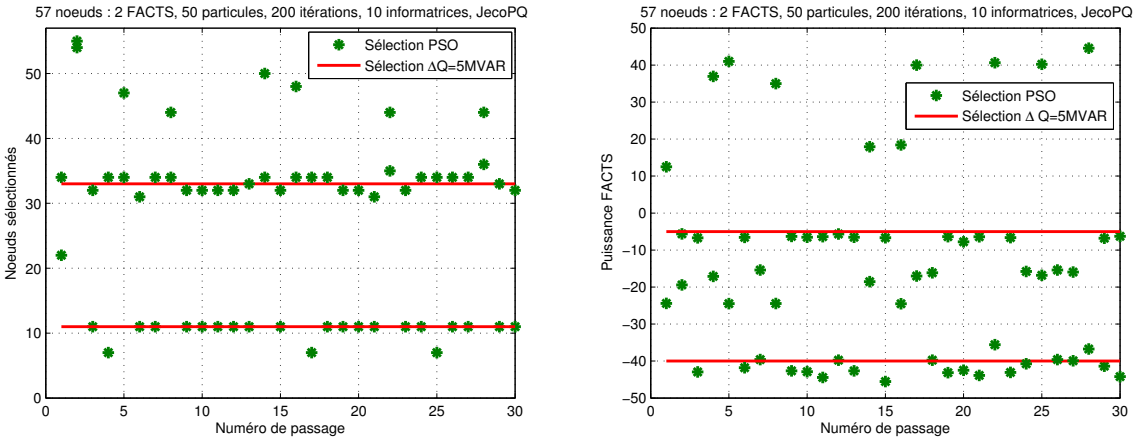


FIGURE 3.4.12.: Réseau IEEE 57 nœuds, 50 particules, 200 itérations, 10 informatrices
Comparaison des positions et puissances sélectionnées : modification du critère économique sur les puissances active P et réactive Q

	STATCOM	*SVC*	*TCSC*
LPD-PSO	*85.54*	*138.48*	*7.52*
PSO standard 2007	*76.31*	*96.29*	*12.19*

TABLE 3.8.: Moyenne sur les 15 passages de la puissance installée pour chaque type de FACTS..

Le tableau 3.9 montre que nous utilisons plus de FACTS de type TCSC. En revanche, la somme de la puissance injectée par l'ensemble des TCSC dépasse à peine $7.52\,MVar$. Ce qui représente une valeur de puissance réactive par FACTS implanté assez faible. Il en est de même pour les FACTS de type STATCOM et SVC. Ce résultat confirme l'action "locale" de la puissance réactive sur le comportement du réseau. Par ailleurs, le tableau 3.10 montre que la somme de puissance installée est assez élevée. Le coût économique associé peut donc s'avérer très important. Pour ces raisons, il est intéressant de traiter le problème de placement de FACTS sous une forme multi-objectif associant un critère économique au critère physique. Cette perspective est initiée et présentée en annexe A.2.

3.5. Conclusion

Les FACTS (*Flexible Alternative Current Transmission System*), en agissant principalement sur la répartition de la puissance réactive dans le réseau, se présentent comme des dispositifs permettant, d'une part, en régime permanent, une "optimisation" des flux de puissance et une meilleure maîtrise de la tension aux nœuds du réseau et, d'autre part, en régime transitoire, une amélioration sensible de la stabilité du réseau. Savoir choisir, dimensionner et placer ces FACTS dans un réseau électrique est un problème encore ouvert aujourd'hui.

Dans ce mémoire, des techniques d'optimisation par essaim de particules (OEP) sont appliquées pour

	STATCOM	*SVC*	*TCSC*
LPD-PSO	*10.33*	*15*	*20.73*
PSO standard 2007	*8.46*	*10.46*	*17.86*

TABLE 3.9.: Moyenne sur les 15 passages du nombre de FACTS utilisés pour chaque type de FACTS.

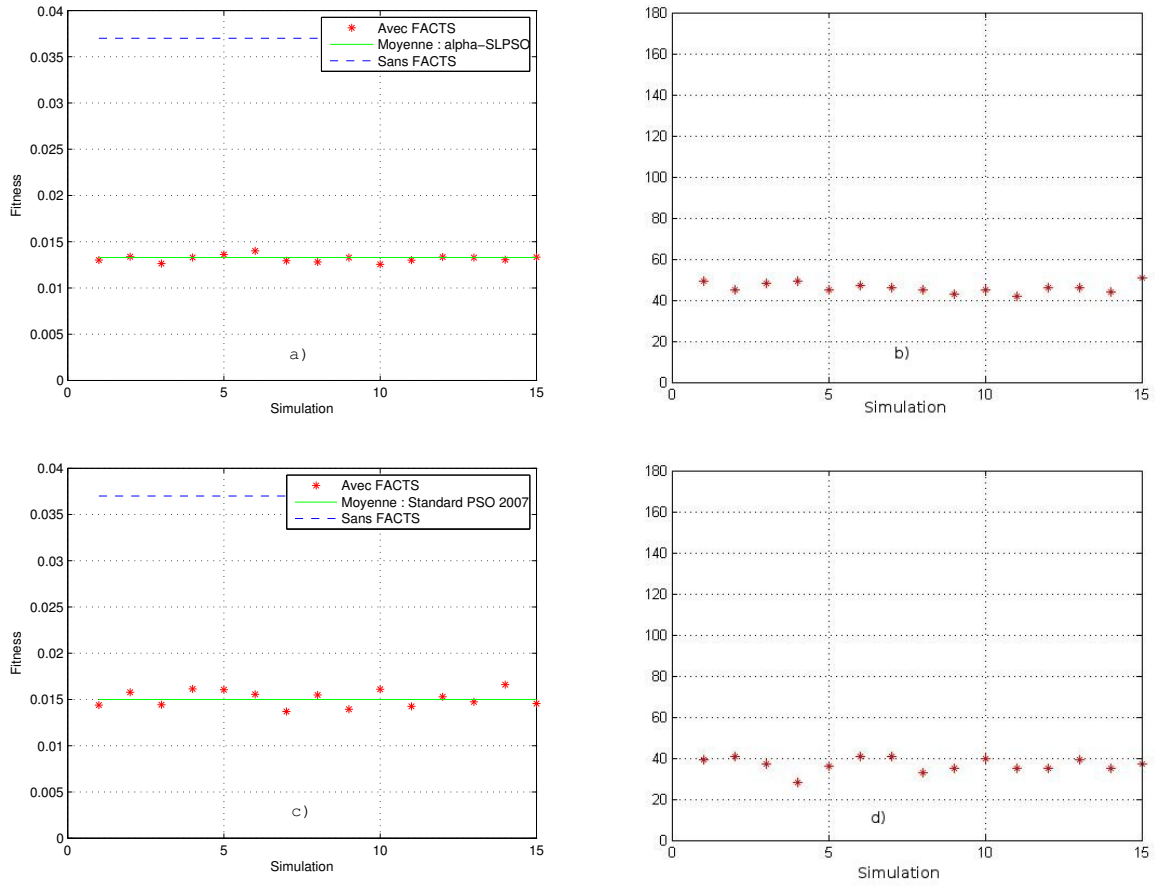


FIGURE 3.4.13.: a) *Fitness* optimaux trouvés en fonction des simulations par $\alpha-$SLPSO. b) Nombre de FACTS optimaux trouvés en fonction des simulations par $\alpha-$SLPSO, c) *Fitness* optimaux trouvés en fonction des simulations par PSO2007 et d) Nombre de FACTS optimaux trouvés en fonction des simulations par PSO2007 .

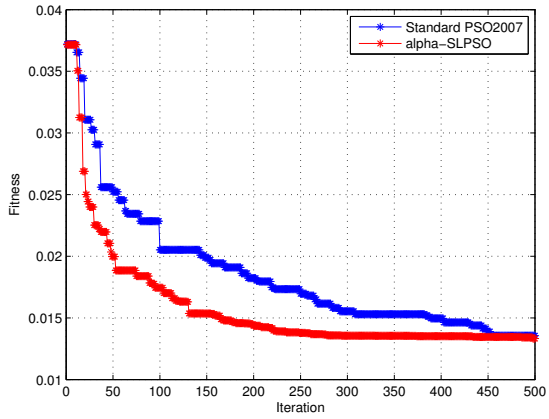


FIGURE 3.4.14.: Évolution de la *fitness* de l'algorithme standard PSO2007 (en bleu) et de l'algorithme OEP α-SLPSO (en rouge).

	Fitness	*Somme des puissance injectées*	*Nombre de FACTS*
LPD-PSO	*0.0132*	*231.54*	*46.06*
PSO standard 2007	*0.0151*	*184.80*	*36.80*

TABLE 3.10.: Somme totale des puissances injectées ($MVar$), Nombre de FACTS installés et moyenne des *fitness* obtenus avec les algorithmes PSO standard de Clerk 2007 et α-SLPSO.

apporter une solution à ce type de problématique, dans un contexte de régime permanent de fonctionnement du réseau (les problèmes de stabilité transitoire ne sont pas considérés ici). Dans le cadre des applications présentées dans ce rapport, les FACTS (ici des STATCOM, SVC et TCSC) sont placés et dimensionnés afin d'optimiser une fonctionnelle relative à la maximisation de la sécurité du réseau électrique (équilibrage des tensions aux nœuds).

L'approche proposée associe une résolution des équations du réseau par une méthode de type *Optimal Power Flow* (OPF) et une recherche de la solution optimale de placement et dimensionnement des FACTS par des techniques OEP. Le caractère hybride du problème d'optimisation (de par la dualité des espaces de recherche, i.e. continu pour le dimensionnement et combinatoire pour le placement) est résolu en associant deux techniques OEP spécifiques. L'aspect continu induit une approche d'OEP classique, très largement utilisée dans de nombreuses applications. L'aspect discret du problème est traité par la méthode de type "Velocity Likehood".

Une meilleure exploration de l'ensemble de l'espace de recherche discret, pouvant ainsi éviter un éventuel blocage trop rapide sur un optimum "local", est obtenue à travers une amélioration de la méthode "Velocity Likehood". Cette amélioration s'appuie sur l'introduction d'une composante aléatoire supplémentaire dans la phase d'actualisation de la vitesse des particules (et donc de la définition de nouvelles positions).

Afin de tenir compte du couplage naturel (de par la topologie du réseau) entre placement et dimensionnement de FACTS (i.e. du caractère hybride du problème d'optimisation), une première approche "d'hybridation" des deux techniques OEP utilisées est proposée. La méthode ainsi mise en œuvre s'appuie sur une hiérarchisation entre ces deux techniques, en ce sens que le placement est prioritaire devant le dimensionnement.

L'algorithme OEP a été appliqué et validé dans le cas de deux réseaux *benchmark IEEE* : le réseau 30 nœuds et le réseau 57 nœuds. De par sa dimension supérieure, le réseau 57 nœuds présente un premier cas d'étude significatif quant à la difficulté du problème de recherche optimale, en ce sens que les espaces de recherche combinatoire et continu sont déjà assez étendus. Les études ont été menées pour un nombre et un type de FACTS fixés, soit 2 STATCOM à placer et dimensionner de manière optimale. Les résultats de simulation obtenus sur ces deux réseaux mettent en valeur les performances de l'algorithme proposé, en termes de recherche d'une solution optimale et de consistance de l'algorithme.

Des modifications des paramètres de réglage de l'algorithme montrent que les performances obtenues sont significativement sensibles à ces réglages. Une telle sensibilité n'est pas spécifique à l'algorithme proposé ici et est souvent récurrente dans la mise en œuvre de techniques d'optimisation de type évolutionnaire, telles que l'OEP ou les algorithmes génétiques. Le problème du réglage optimal de ces algorithmes reste encore un problème ouvert.

Dans ce chapitre, un algorithme à base de population d'agents (α-*Stable Lévy Particle Swarm Opti-*

mization : α-SLPSO) et une recherche locale (α-*Stable Lévy Local Search* : α-SLLS) ont été proposés. De très bons et très encourageants résultats sont présentés dans la section 3.3.3.3. En effet, une méthode de classement (ELECTRE II), appliquée sur un ensemble de fonctions tests, met en exergue les performances de $\alpha-$SLPSO par rapport à cinq autres variantes d'OEP (PSO-2S, EPUS-PSO, CLPSO, CPSO-H et standard SPO2007).

L'algorithme α-SLPSO a été utilisé dans le cadre du placement et dimensionnement de FACTS de types différents. Les résultats obtenus montrent que l'algorithme α-SLPSO présente des performances très proches, voire meilleures, que celles obtenues avec le PSO2007 standard. De plus, l'algorithme α-SLPSO s'avère très performant en terme de vitesse de convergence (gain de l'ordre de 40%). Cette performance est à mettre en relation avec la nature de l'algorithme α-SLPSO, qui s'appuie sur l'utilisation de lois α-stables de Lévy, qui autorisent une stratégie de recherche introduisant de l'intermittence (alternance de recherches locales et de "sauts" balistiques importants).

CHAPITRE 4

ÉTUDE TOPOLOGIQUE DES RÉSEAUX ÉLECTRIQUES

4.1. Introduction

Les réseaux de transmission d'énergie électrique, compte tenu des nombreuses interconnexions, présentent une complexité grandissante. Pour des raisons économiques, ils sont souvent exploités près de leurs conditions limites de fonctionnement. La majeure partie des déclenchements en cascade ou des pannes électriques de ces systèmes ont de sérieuses conséquences [Areva]. Bien que chaque panne électrique puisse être attribuée à une cause particulière (défaillance d'équipements, erreurs humaines, ...), le réseau électrique doit être considéré comme un système complexe et une attention particulière doit être accordée à son comportement global, tant statique que dynamique. C'est l'objectif principal de la théorie des réseaux complexes [Bocc06]. En ce sens, l'étude des réseaux électriques a connu et connaît un essor considérable [Albe04, Carr02, Cruc05, Kinn05].

Comme beaucoup de systèmes, un réseau électrique doit faire face à des pannes qui, compte tenu de sa grande connectivité, peuvent s'étendre à des régions entières : on parle alors de "blackout" (phénomène d'avalanche), c'est-à-dire ayant des conséquences à grande échelle. La taille des réseaux électriques et leur complexité rendent difficile la compréhension de ces phénomènes qui émergent localement. En ce qui concerne les problèmes tels que l'effondrement de la tension, la congestion de réseaux ou le phénomène de *blackout*, la complexité topologique du réseau électrique est l'une des causes qui peuvent amener le réseau dans ses conditions limites de fonctionnement. Un certain nombre de travaux existent et se fondent sur un usage intensif des outils de physique statistique. L'adaptation de méthodes de percolation, les réseaux sans degré caractéristique, les systèmes critiques auto-organisés sont autant d'outils de choix pour décrire les propriétés statistiques et topologiques d'un réseau.

L'objectif principal du présent chapitre est de déterminer la capacité des réseaux électriques à surmonter des perturbations ou défaillances (variations de charge, coupure de lignes, ...) à travers une étude de robustesse statique (i.e., sans prise en compte d'un flux de puissance circulant dans le réseau). Pour ce faire, nous commençons dans un premier temps par une étude "non exhaustive" des caractéristiques phénoménologiques, ou encore d'observables, qui sont extraites du corpus de mesures traditionnellement

employées pour la description des réseaux complexes. Nous étudierons également la vulnérabilité du réseau à travers une étude de robustesse aux attaques et aux pannes. Afin de compléter cette analyse statique, nous étudierons dans un deuxième temps la fractalité du réseau en question, en appliquant trois algorithmes pour la détermination de la dimension fractale associée. De surcroît, nous proposons dans la dernière partie du chapitre un modèle, qui peut être représentatif de la dynamique des réseaux électriques.

4.2. Analyse topologique du réseau colombien

Dans cette partie de chapitre, un arsenal d'indicateurs topologiques et de robustesse des réseaux électriques est présenté. Dans un premier temps, nous commençons par esquisser quelques observables extraits du lexique de la théorie des graphes [Berg73] et des systèmes complexes [Bocc06, Newm03], en les appliquant au réseau électrique colombien. La raison du choix de ce cas d'étude est que, dans la collaboration avec le partenaire colombien, il est implicitement entendu que les outils développés sont à appliquer à leur réseau. Dans un deuxième temps, nous nous intéressons à l'étude de robustesse et, de fractalité du réseau et en particulier, trois algorithmes seront présentés.

4.2.1. Caractéristiques phénoménologiques

4.2.1.1. Quelques indicateurs topologiques

Avant d'introduire quelques définitions (caractéristiques) d'un graphe, on définit ce dernier comme suit :

Un graphe G, souvent noté $G = (V, E)$, est un couple d'ensembles (V, E) où $V = \{v_1, ..., v_N\}$ et $E = \{e_1, ..., e_K\}$ désignent respectivement les ensembles de N nœuds et K liens $(i, j) \in V^2$ du graphe G.

Le graphe G peut être décrit de manière matricielle de deux façons :
- *Matrice d'adjacence* A : c'est une matrice carrée $N \times N$ d'éléments $a_{ij}\,(i, j = 1, ..., N)$ où $a_{ij} = 1$ si le nœud i est adjacent au nœud j et 0 sinon.

- *Matrice d'incidence* B : c'est une matrice de dimension $N \times K$ d'éléments $b_{ik}\,(i = 1, ..., N\,;\, k = 1, ..., K)$ où $b_{ik} = 1$ si le nœud $i \in V$ est adjacent au lien $k \in E$ et 0 sinon.

Degré d'un nœud, degré moyen d'un graphe, homogénéité et corrélations

Le degré, parfois désigné par connectivité, d'un nœud $i \in V$ noté k_i est le nombre de liens $(i, j) \in V^2$ adjacents au nœud i (i.e., le nombre de voisins de i). Plus formellement, $k_i = \sum_{i \in V} a_{ij}$ en désignant par a_{ij} les éléments de la matrice d'adjacence A correspondant au graphe G. Ainsi, le degré moyen du graphe G est le moment d'ordre 1 (espérance) $< k >$ de la distribution des degrés $P(k)$:

$$< k >= \sum_k kP(k) = \sum_{i \in V} \frac{k_i}{N} \tag{4.2.1}$$

Dans le cas où la distribution des degrés $P(k)$ n'est pas régulière, on dit que le graphe G est *hétérogène*. Autrement, le graphe est dit *homogène* (régulier).

La distribution des degrés $P(k)$ nous fournit une photographie de la connectivité des nœuds tels qu'ils sont réellement connectés dans le graphe. Cependant, des corrélations entre les degrés des nœuds peuvent se cacher dans le graphe, à savoir, la probabilité qu'un nœud de degré k se connecte à un autre nœud de degré k' dépend de k. En d'autres termes, il peut y avoir une relation d'attachement préférentiel entre les

nœuds. Les corrélations des degrés du nœud i, $k_{nn,i}$, peuvent être mesurées en introduisant la *moyenne des degrés des voisins les plus proches* du nœud i, définie comme suit :

$$k_{nn,i} = \frac{1}{k_i} \sum_{j \in \eta_i} k_j = \frac{1}{k_i} \sum_{j=1}^{N} a_{ij} k_j$$

où η_i et k_i désigne respectivement, l'ensemble des voisins du nœuds i et son degré dans le graphe.

De là, la *moyenne des degrés des voisins les plus proches*,$k_{nn}(k)$, de n'importe quel nœud de degré k, peut être exprimée par la quantité [Past01] :

$$k_{nn}(k) = \sum_{k'} k' P(k'|k) \tag{4.2.2}$$

où $P(k'|k)$ désigne la probabilité conditionnelle qu'un nœud de degré k' se connecte à un autre nœud de degré k.

Dans le cas où $k_{nn}(k)$ est croissant, la corrélation est dite *assortative*. Elle est dite *diassortative* lorsque $k_{nn}(k)$ est décroissant. En l'absence de corrélations, la quantité $k_{nn}(k)$ se réduit à $\frac{<k^2>}{<k>}$, c'est à dire, $k_{nn}(k)$ est indépendant de k.

Dans la réalité, la formule (4.2.2) est difficilement calculable. Pour remédier à ce problème, le coefficient de corrélation de Pearson r constitue une autre alternative pour calculer la quantité $k_{nn}(k)$. Il est défini comme suit [Newm02] :

$$r = \frac{K^{-1} \sum_i j_i k_i - [K^{-1} \sum_i \frac{1}{2}(j_i + k_i)]^2}{K^{-1} \sum_i \frac{1}{2}(j_i^2 + k_i^2) - [K^{-1} \sum_i \frac{1}{2}(j_i + k_i)]^2} \tag{4.2.3}$$

en désignant par j_i, k_i respectivement, les degrés des deux nœuds constituant le lien i, avec $i = 1, ..., K$.

Moyenne des chemins géodésiques, diamètre et efficacité d'un graphe

Le *chemin géodésique* est l'un des paramètres les plus importants d'un graphe. De fait, l'acheminement d'une information dans un graphe G passe nécessairement par la manipulation des chemins géodésiques de ce dernier. Formellement, le chemin géodésique entre un nœud i et un nœud j, noté d_{ij}, est le plus court chemin existant dans le graphe reliant les deux nœuds. La valeur maximale de tous les d_{ij} est appelée le *diamètre du graphe* ($Diam(G)$). De là, on peut définir *la moyenne des chemins géodésiques* L, d'un graphe G comme suit :

$$L = \frac{1}{N(N-1)} \sum_{i,j \in V, i \neq j} d_{ij} \tag{4.2.4}$$

Dans le cas où le graphe G n'est pas connexe (c'est-à-dire qu'il existe des nœuds inatteignables à partir d'autres nœuds du graphe), *la moyenne des chemins géodésiques* L et le diamètre du graphe divergent. Par ailleurs, ces derniers sont l'une des mesures d'efficacité d'un réseau, qui peuvent être utilisées comme indicateurs de robustesse d'un réseau. Pour remédier à ce problème de divergence, on définit *l'efficacité globale* E du graphe G comme étant la moyenne harmonique des chemins géodésiques, à savoir :

$$E = \frac{1}{N(N-1)} \sum_{i,,j \in V, i \neq j} \frac{1}{d_{ij}} \tag{4.2.5}$$

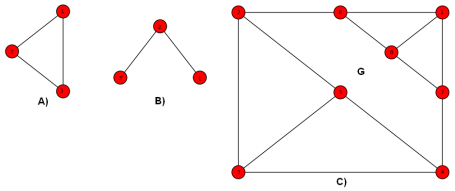


FIGURE 4.2.1.: a) Triangle b) Triplet c) Graphe illustratif.

Coefficient de centralité et d'intermédiarité (*betweenness*)

Le *coefficient de centralité et d'intermédiarité* (*betweenness*) des nœuds (éventuellement des liens) est l'une des caractéristiques les plus répandues d'un graphe. En effet, cette mesure nous permet de déterminer le rôle ou l'importance de chaque nœud (éventuellement d'un lien) du graphe. A l'origine, le coefficient de centralité et d'intermédiarité a été introduit pour mesurer l'importance d'un individu dans les réseaux sociaux [Wass94]. Il est défini comme le nombre de fois qu'un nœud (lien) est apparu dans la construction des chemins géodésiques du graphe. Plus précisément, le coefficient de centralité et d'intermédiarité qu'on notera b_i d'un nœud i est défini par la relation (4.2.6).

$$b_i = \sum_{j,k \in V, j \neq k} \frac{n_{jk}(i)}{n_{jk}} \tag{4.2.6}$$

en désignant par n_{jk}, le nombre de chemins géodésiques entre le nœud j et le nœud k et $n_{jk}(i)$, le nombre de chemins géodésiques entre les nœuds j et k, passant par le nœud i.

A titre d'illustration sur le graphe G de la figure (4.2.1), on a :

$$\{b_1, ..., b_8\} = \{0.12,\ 0.12,\ 0.12,\ 0.13,\ 0.12,\ 0.13,\ 0.13,\ 0.12\}.$$

Coefficient de regroupement (*clustering*)

"L'ami de mon ami est mon ami" dit-on. Cette expression n'a rien de banal. En effet, les amis d'un individu ont une forte chance de se connaître, en raison de leur similarité (centres d'intérêt, lieux de rencontre, proximité,...). Cette probabilité de lien entre les amis d'un individu peut être mesurée par le *coefficient de regroupement* ("*clustering* "ou *transitivité*) de ce dernier. Plus formellement, le coefficient de regroupement c_i d'un nœud i est défini comme suit :

$$c_i = \frac{2e_i}{k_i(k_i - 1)} \tag{4.2.7}$$

en désignant par e_i le nombre maximal de liens dans le sous-graphe des voisins du nœud i.

A titre d'illustration sur le graphe G de la figure (4.2.1-c), $c_1 = \frac{2\times 2}{3\times(3-1)} = \frac{2}{3}$ et $c_2 = \frac{2\times 1}{3\times(3-1)} = \frac{1}{3}$.

La quantité c_i n'est pas une mesure commune à tout les nœuds du graphe G. Par conséquent, le comportement global de ce dernier est difficilement cernable en utilisant seulement l'équation (4.2.7). On peut alors introduire une autre grandeur appelée *"coefficient de regroupement moyen"* d'un graphe G, qui est défini comme la moyenne de tous les coefficients de regroupement des nœuds de G :

$$C =< c >= \frac{1}{N} \sum_{i \in V} c_i \tag{4.2.8}$$

La formule (4.2.8) mesure la densité des triangles présents dans le graphe G (figure (4.2.1)) pour la définition d'un triangle et d'un triplet). Par conséquent, la mesure c_i peut être remplacée par la *transitivité* T, définie comme suit :

$$T = \frac{3 \times \textit{Le nombre de triangles présents dans } G}{\textit{Le nombre de triplets présents dans } G} \tag{4.2.9}$$

L'examen des équations (4.2.7), (4.2.9) et (4.2.8) nous amène aux remarques suivantes :

- $0 \leq c_i \leq 1$, $0 \leq C \leq 1$.
- On peut définir le coefficient de regroupement des liens de la même manière que pour les nœuds.
- On peut définir le coefficient de regroupement $c(k)$ d'une classe de nœuds de degré k.

A titre d'illustration sur sur le graphe G de la figure (4.2.1).c, on a :

- $C = \frac{1}{8}(\frac{2}{3} + \frac{1}{3} + \frac{2}{3} + \frac{1}{3} + \frac{2}{3} + \frac{1}{3} + \frac{1}{3} + \frac{2}{3}) = \frac{1}{2}$.
- $T = \frac{3 \times 4}{20} = \frac{3}{5}$.

Comme indiqué tout au début du chapitre, cette liste d'indicateurs est loin d'être exhaustive. Pour d'autres caractéristiques d'un graphe, on se réfèrera à [Bocc06, Newm03, Doro03, Doro08].

4.2.1.2. Résultats

Nous nous sommes intéressés au cas du réseau électrique colombien. Les informations dont nous disposons concernent la structure du réseau, son plan de charge et l'historique des incidents. Dans les données communiquées par l'opérateur colombien, figurent un plan a priori complet de son réseau et une base de données des incidents survenus sur son réseau depuis 1996. Le plan se présente sous la forme d'un fichier de sauvegarde d'un cas d'étude sur le logiciel *DigSilent*. Ce logiciel est un environnement complet d'études de *Power Flow* édité par une entreprise allemande. Par ailleurs, les coordonnées GPS du réseau colombien ont été fournies sous forme de fichier KML (*Keyhole Markup Language*).

Il a fallu, pour rendre ces données brutes exploitables par nos algorithmes, les extraire de leur environnement logiciel (DigSilent). La procédure a été d'exporter dans un format propriétaire DigSilent (format ASCII .dgs) la base de données de l'opérateur et d'écrire des scripts Python pour lire le fichier texte pour en extraire les données pertinentes. Celles qui nous intéressent dans cette partie de chapitre concernent la topologie (lignes, stations et sous stations) du réseau. Il a ainsi été possible de rendre l'exploitation et l'interaction des résultats sous forme de langage KML (*Google Earth*), grâce à des programmes développés sous Python. Un exemple d'un fichier KML du réseau colombien obtenu est présenté dans la figure (4.2.2). Bien que cette présentation Google Earth soit plus élégante et pratique, pour des raisons de visualisation, nous ferons appel dans la suite à un module de la bibliothèque de traitement de graphes en Python : *NetworkX*.

Une fois obtenue une version exploitable du réseau colombien, nous avons rassemblé l'ensemble des codes traitant de la caractérisation de graphes pour en faire une application cohérente et complète, capable d'exécuter tous les traitements sur un graphe cible. Ce programme a été nommé *"BTGT"*. Il est composé d'un noyau principal, de modules de traitement unitaire et d'un gestionnaire de graphes.

Il a été conçu de manière modulaire pour accepter, via l'héritage Python, l'adjonction de futurs modules. Les modules intégrés sont les modules de calcul des "betwennesses", des indices de "clustering" et des distributions de degrés. Par la suite, ont été adjoints des tests de vulnérabilité aux attaques et aux pannes, un module de calcul de dimension fractale de graphes. Par ailleurs, le programme Python "*BTGT*" mis

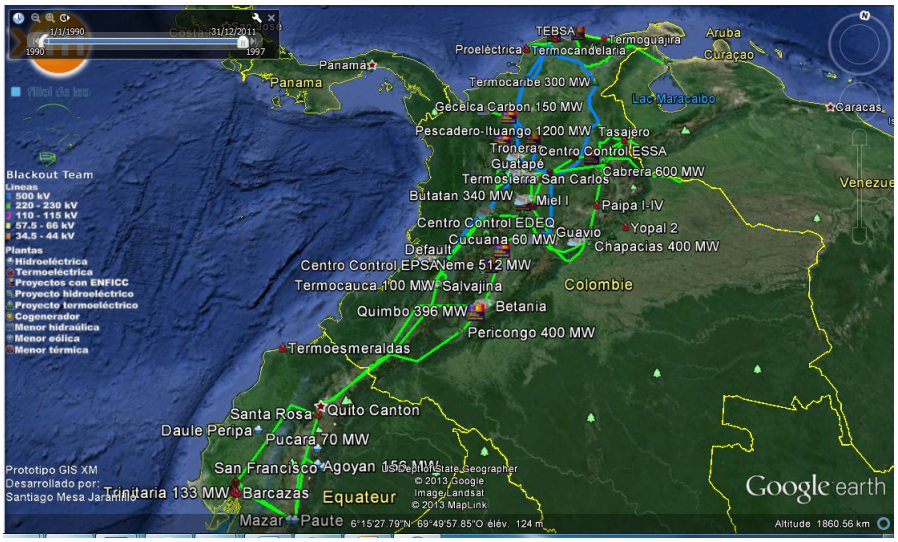


FIGURE 4.2.2.: Réseau colombien (lignes 500 KV et 220 KV) en fichier KML (Google Earth)

en œuvre génère un fichier texte contenant les principales caractéristiques phénoménologiques d'un réseau.

Un exemple de fichier obtenu à l'aide du programme appliqué au réseau colombien se présente comme suit :

```
Moyenne des chemins geodesique: L=9.9353400223
Efficacite glbale: E=0.133844784944
Diametre: D=24
Rayon: R=13
Assortativite:-0.220627605881
Coefficient de Pearson: r=-0.220627605881
Transitivite: T=0.0960451977401
Moyenne des coefficients de regroupement: C=0.085632034632
La densite : Densite=0.00911928651059
```

Réseau réel à 300 noeuds

```
Moyenne des chemins geodesique: L=8.25182269022
Efficacite glbale: E=0.148346370898
Diametre: D=21
Rayon: R=11
Assortativité:-0.14107292377
Coefficient de Pearson:-0.14107292377
Transitivite: T=0.107775590551
Moyenne des coefficients de regroupement: C=0.112945084997
La densite : Densite=0.0061727805223
```

Réseau réel colombien

Les résultats obtenus sur le réseau colombien sont représentés sur les figures (4.2.3, 4.2.5 et 4.2.4). Les figures (4.2.3-a), (4.2.3-b), (4.2.4-c), (4.2.5) et (4.2.3-f) représentent, respectivement, les coefficients de centralité des liens, des nœuds, les coefficients de regroupement des nœuds, la distribution des degrés en évolution linéaire et en diagramme loglog du réseau colombien.

Les couleurs atténuées des nœuds ou des liens sur les figures (4.2.3-a), (4.2.3-b), (4.2.4-a) indiquent respectivement que ces derniers ont des coefficients de centralité ou des coefficients de regroupement (*clusters)* faible. Par opposition, les nœuds ou liens de coefficients plus grands sont de couleurs plus foncées.

Sur la figure (4.2.4-b), les nœuds ayant un symbole carré représentent le centre du réseau et ceux ayant un symbole triangulaire représentent sa périphérie et les couleurs foncées indiquent l'éloignement des nœuds du centre du réseau. A l'inverse, les nœuds proches du centre sont représentés par des couleurs atténuées.

L'examen de ces courbes amène les commentaires suivants :

– Le réseau colombien n'est pas efficace localement. En effet, les indices de *clustering* indiquent l'efficacité locale du réseau. En d'autres termes, si un nœud d'indice de *clustering* élevé (couleur foncée) tombe en panne, sa forte connectivité permet d'acheminer l'énergie par ses plus proches voisins. Or,

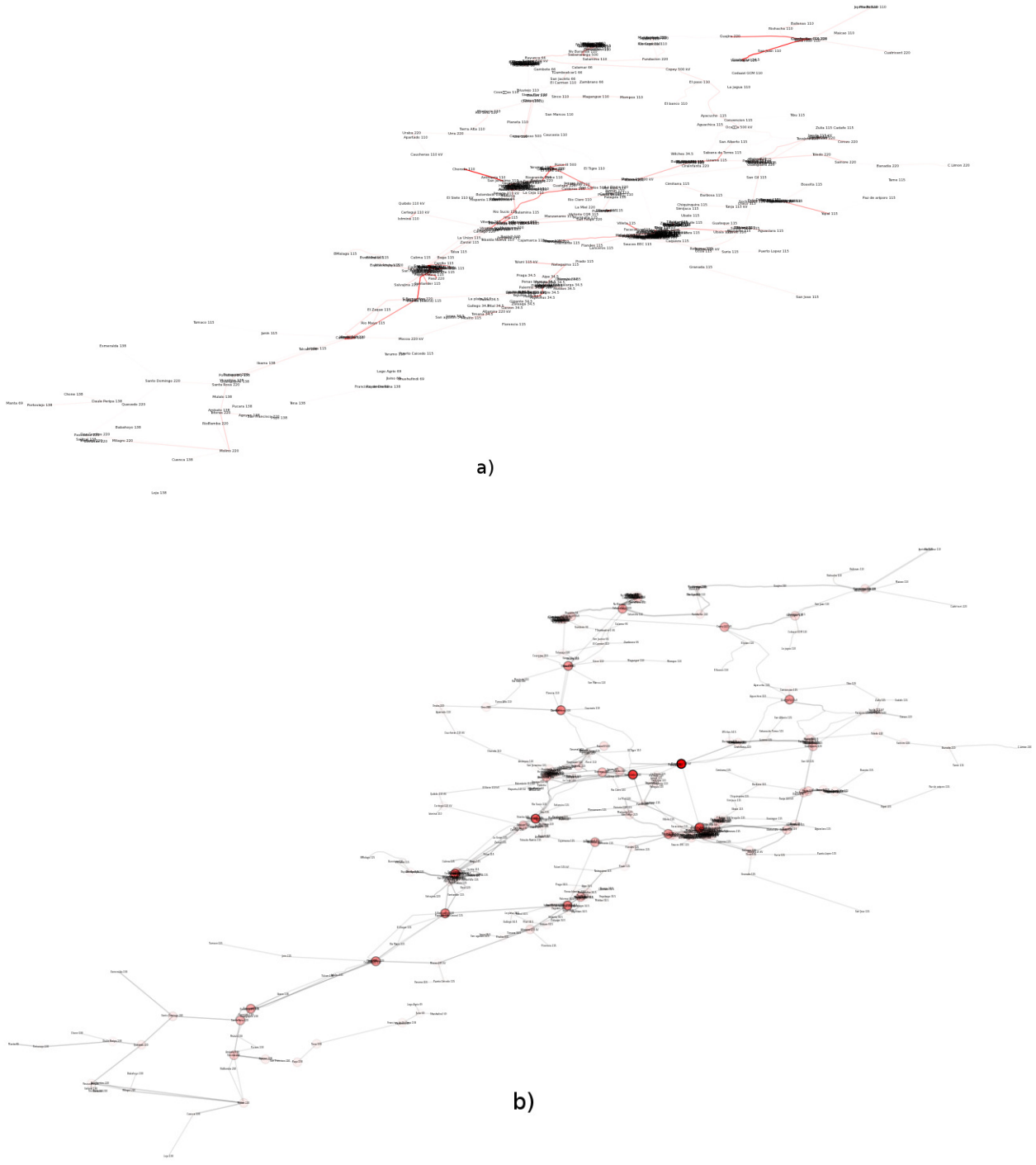


FIGURE 4.2.3.: Résultats sur le réseau colombien : a) coefficients de centralité (betwennesses) des liens b) coefficients de centralité des nœuds

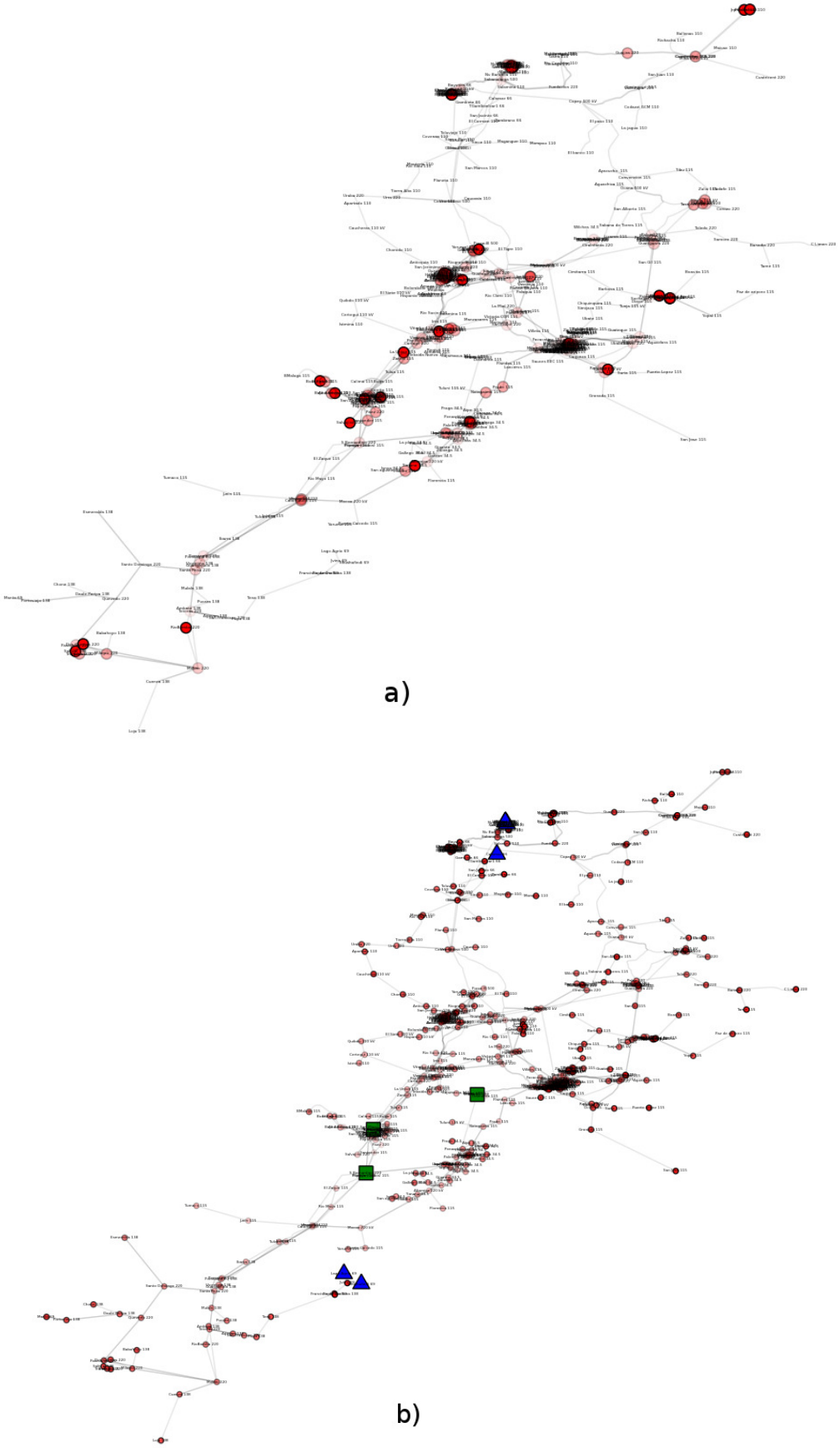
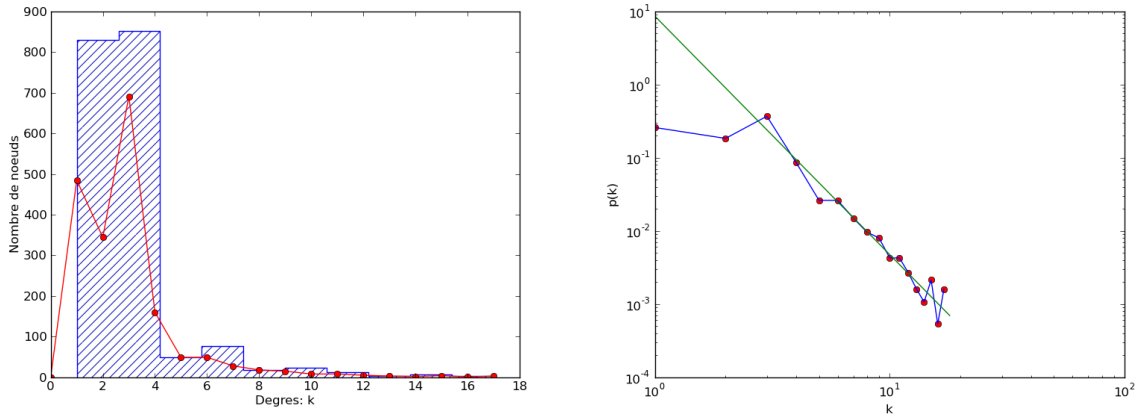


FIGURE 4.2.4.: Résultats sur le réseau colombien : a) coefficient de regroupement (*clustering*) b) centre et périphérie



a) b)

FIGURE 4.2.5.: Résultats sur le réseau colombien : a) distribution des degrés b) distribution des degrés dans un diagramme loglog

la figure 4.2.3-c) montre clairement qu'il y a peu de régions avec un coefficient de regroupement élevé.

– Le réseau colombien peut être considéré comme libre d'échelle, avec comme exposant : $\gamma = 3.26$. En effet, la figure (4.2.4-b) fait apparaître un régime linéaire en diagramme loglog de la distribution des degrés du réseau colombien. Par ailleurs, le réseau ne peut être considéré comme "*Small World*". En effet, le diamètre et la moyenne des chemins géodésiques du réseau sont trop élevés pour qu'on puisse le considérer comme tel.

– La valeur négative du coefficient de corrélation de Pearson r laisse suggérer que le réseau colombien est dis-assortatif. En d'autres termes, des attachements préférentiels peuvent exister dans le réseau.

4.2.2. Dimension fractale

La dimension fractale, notée D, est une grandeur qui donne une indication sur la manière dont un ensemble fractal remplit l'espace. Il existe de nombreuses définitions à la dimension fractale [Thei90]. Les plus importantes notions de dimensions fractales rencontrées dans la littérature et définies de façon formelle sont la dimension de Rényi et la dimension de Hausdorff.

Les dimensions fractales sont utilisées dans de nombreux domaines de recherche, tels que la physique [Zhan10], le traitement d'image [Soil96, Toll03], l'acoustique [Efte04], la chimie [Efte04]. Outre l'aspect formel et théorique, de nombreux organismes et phénomènes du monde réel présentent des propriétés fractales (végétation, organismes cellulaires, sismicité, etc), à savoir des propriétés d'auto-similarité (déterministes ou stochastiques).

Ce que nous retiendrons pour ce chapitre est la nécessité de pouvoir disposer de l'estimation de la dimension fractale, car les phénomènes mis en jeu dans la dynamique des réseaux électriques laissent entrevoir des propriétés d'échelle :

1. propriétés d'échelle au niveau des graphes des réseaux électriques, sachant que la définition de la dimension fractale des graphes se fait par extension, la notion de métrique étant difficilement appréhendable [Tani05, Thei90, Song07],
2. propriétés d'échelle dans la dynamique même de ces réseaux, dès lors que le modèle de *criticalité*

auto-organisée se révèle pertinent pour ceux ci [Carr02] . L'objet en question est alors une série temporelle, ou un ensemble de données échantillonnées.

Pour ce qui est de l'estimation des dimensions fractales des graphes, nous présentons trois des algorithmes les plus utilisés à savoir :

1. l'algorithme de coloriage de graphe (GC),
2. l'algorithme *Compact-Box-Burning* (CBB),
3. l'algorithme *Maximum-Excluded-Mass-Burning* (MEMB).

L'intérêt de la description de ces algorithmes est lié directement à l'étude des marches aléatoires dans un graphe, l'apparition des pannes sur un nœud, éventuellement un lien, et la topologie du graphe. Autrement dit, il serait intéressant de détecter des propriétés d'auto-similarité régissant le graphe (panne, marche aléatoire dans le graphe, topologie du graphe, etc).

Avant de présenter ces méthodes, il est nécessaire de définir formellement ce qu'est une "boîte", définition 1 et ce qu'est une dimension "fractale", définition 2 :

Définition 8. Dans un graphe $G = (V, E)$, on appelle une boîte de taille l_B (respectivement de rayon r_B), un sous-graphe $H = (V', E')$ de G tel que $\nabla(i, j) \in V'^2$, $l_{ij} < l_B$ (respectivement, $l_{ij} \leq 2 * r_B$).

Définition 2. Soit K un sous-ensemble compact de $\mathbb{R}^n$, on désigne par $N_\varepsilon(K)$ le nombre minimum de boules fermées de rayon ε nécessaires pour recouvrir K. On définit alors la *dimension fractale* (si elle existe), notée D_f , par :

$$D_f = \lim_{\varepsilon \to 0} \frac{logN_\varepsilon(K)}{log\frac{1}{\varepsilon}} \tag{4.2.10}$$

En se référant à la définition 8, la dimension fractale du sous-ensemble compact K correspond alors à la pente dans le diagramme *loglog* de $N_\varepsilon(K)$ *et* $\frac{1}{\varepsilon}$.

Bien que, dans le domaine des mathématiques, la notion de dimension fractale soit bien précise, elle reste néanmoins moins évidente quant à sa définition dans un graphe. En effet, on peut définir autant de métriques que l'on veut dans un graphe. Dans ce qui suit et en se référant aux définitions 1 et 2, nous définissons une dimension fractale d'un graphe $G = (V, E)$ de deux manières [Cost07] :

- La première, issue de la méthode des boîtes, couvre le réseau avec N_B boîtes de côté (diamètre) ℓ_B, i.e. la distance entre deux nœuds dans une boîte est inférieure ou égale à ℓ_B. En notant la dimension fractale d_B, un graphe fractal suivra alors la loi (4.2.11) :

$$N_B \sim \ell_B^{-d_B} \tag{4.2.11}$$

- La seconde, issue de la méthode "Cluster Growing", choisit un nœud graine aléatoirement et forme un "cluster" à partir de ce nœud, i.e. prend tous les nœuds disponibles à une distance inférieure ou égale à ℓ, puis recommence l'opération jusqu'à épuisement des nœuds disponibles. En notant M_c la masse moyenne de l'ensemble des "clusters" résultant et d_C la dimension fractale du graphe associé, un graphe fractal suivra alors la loi (4.2.12) :

$$M_C \sim \ell^{d_C} \tag{4.2.12}$$

Algorithme 4.1 "*Greedy Coloring*", GC

i) Sans affecter aucune couleur, donner un identifiant $Id(i) \in \{1, 2, ..., N\}$ unique pour chaque nœud,
ii) Pour toutes les valeurs de l_B, affecter la couleur 0 au nœud d'identifiant $Id = 1$,
iii) Poser $i = 2$ et répéter ce qui suit tant que $i \leq N$:
a) pour chaque nœud d'identifiant $Id(j) < Id(i)$, calculer la distance l_{ij},
b) Poser $l_B = 1$,
c) Affecter au nœud d'identifiant $Id(i)$ une couleur C_{il_B} qui n'est pas prise par les autres nœuds d'identifiants $Id(j) < Id(i)$ tels que : $l_{ij} \geq l_B$,
d) Poser $l_B = l_B + 1$ et répéter c) tant que $l_B \leq l_B^{max}$,
e) Poser $i = i + 1$,

Bien que ces deux définitions soient exclusives dans certains cas, c'est-à-dire qu'il existe des situations où un graphe est fractal uniquement suivant une seule de ces deux définitions, ces cas restent néanmoins rares. Par la suite, on privilégiera la première définition, car elle est plus consensuelle dans la littérature.

4.2.2.1. L'algorithme des couleurs ("Greedy Coloring", GC)

Dans [Song07], les auteurs ont montré que le problème de couverture optimale d'un graphe G à l'aide de boîtes de tailles maximales l_B est équivalent à la résolution du problème de coloriage du graphe dual G' associé. En partant du graphe initial G et en fixant la taille maximale des boîtes à l_B, deux nœuds i et j du graphe G' sont connectés s'ils sont à distance géodésique, dans le graphe initial G, $l_{ij} \geq l_B$. Sur la figure (4.2.6-a), les nœuds 2, 3 et 8 sont, par rapport au nœud 1, à distance inférieure ou égale à 3. Par conséquent, ils ne peuvent être connectés au nœud 1 dans le graphe dual G' (figure (4.2.6-b)).

L'algorithme proposé dans [Song07] utilise une méthode constructive classique de coloriage appelée (*Greedy*) et construit en parallèle le graphe dual G'. Le principe général de fonctionnement de l'approche est décrit dans l'algorithme (4.1).

On commence tout d'abord par l'indexation des nœuds et, ensuite, on affecte la couleur 0 au nœud d'identifiant $Id = 1$ et ce, pour toutes les valeurs de l_B. En effet, chaque nœud du graphe aura autant de couleurs que le nombre de variations de l_B. A titre d'exemple, si l_B prend ses valeurs dans {1,2,...,M}, chaque nœud aura par conséquent M couleurs. A chaque itération i, les calculs des distances l_{ij} ne sont effectués que pour les nœuds j d'identifiant $Id(j) < Id(i)$. A la fin de l'algorithme, la dimension fractale du graphe est identifiée comme la pente en diagramme *loglog* du nombre de boîtes nécessaires pour couvrir tout le graphe en fonction des tailles maximales de ces dernières (i.e., l_B).

4.2.2.2. L'algorithme "Compact-Box-Burning" (CBB)

Le principe général de fonctionnement de l'algorithme CBB se décline comme suit :
- Au départ, tous les nœuds sont susceptibles d'être sélectionnés pour appartenir à la boîte courante. On sélectionne aléatoirement un nœud p dans le graphe (figure (4.2.7-a)), ce nœud représente le centre de la boîte en question (les nœuds en noir de la figure (4.2.7)).
- On écarte ensuite tous les nœuds i du graphe (les nœuds en rouge de la figure (4.2.7)) qui sont à distance géodésique $l_{ip} \geq l_B$.
- On réitère le processus de sélection d'un nouveau centre de la boîte dans le sous graphe résultant de chaque étape de sélection, jusqu'à ce qu'aucun nœud ne soit susceptible d'appartenir à la boîte (figures (4.2.7-b) et (4.2.7-c)).

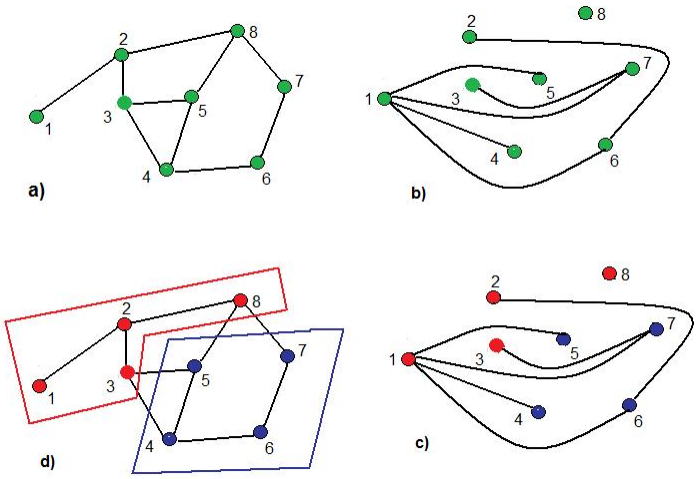


FIGURE 4.2.6.: Exemple d'application de l'algorithme de coloriage, $l_B = 3$: a) le graphe initial G b) le graphe dual G' c) résolution du problème de coloriage du graphe dual G' d) représente le résultat final (nombre de boîtes $N_B = 2$)

- A la fin de ces étapes, on construit une boîte contenant des nœuds à distance géodésique au plus $l_B - 1$ (figure (4.2.7-c)).
- Il ne reste qu'à répéter la procédure de construction des boîtes jusqu'à ce que tous les nœuds soient affectés à une boîte (figures (4.2.7-d), (4.2.7-e) et (4.2.7-f) pour l'itération 2, figure (4.2.7-g), (4.2.7-h) et (4.2.7-i) pour l'itération 3 et enfin, figure (4.2.7-j) pour l'itération 4).

Le résumé de la technique CBB est présenté dans l'algorithme 4.2.

Algorithme 4.2 "*Compact-Box-Burning*", CBB

i) Poser $C = \{l'ensemble\ des\ noeuds\ non\ affectés\ à\ une\ boîte\}$,
ii) Choisir aléatoirement un nœud $p \in C$ et poser $C = C \backslash \{p\}$,
iii) Poser $C = C \backslash \{i\ tel\ que\ l_{pi} \geq l_b\}$,
iv) Répéter ii) et iii) tant que $C \neq \varnothing$

4.2.2.3. L'algorithme "*Maximum-Excluded-Mass-Burning*" (MEMB)

L'algorithme MEMB vise à améliorer l'aspect aléatoire de la sélection des nœuds centraux des boîtes de l'algorithme CBB. L'idée de l'algorithme est basée sur le choix des *hubs* (des nœuds de plus fort degré) comme centres des boîtes, ce qui nous rapproche, a priori, de la solution optimale.

Pour ce faire, on définit tout d'abord la notion de "*masse exclue*" comme suit :

- pour un rayon de recouvrement r_B donné, la "*masse exclue*" d'un nœud i est égale au nombre de nœuds j (y compris le nœud i en question) qui sont à distance géodésique $l_{ij} \leq r_B$ et qui ne sont pas encore affectés à une boîte (visités). Sur la figure (4.2.8-a), le nœud 3 regroupe 3 nœuds (les nœuds 2, 4 et 5) qui sont à distance géodésique inférieure à $r_B = 1$, sa "*masse exclue*" vaut donc 4.
- Il suffit alors de calculer les "*masses exclues*" de tous les nœuds non centraux à chaque étape de l'algorithme et de choisir le nœud p de "*masse exclue*" la plus importante comme le nouveau centre de la nouvelle boîte et ensuite, marquer tous les nœuds j à distance géodésique $l_{jp} \leq r_B$ comme étant "*marqués*".

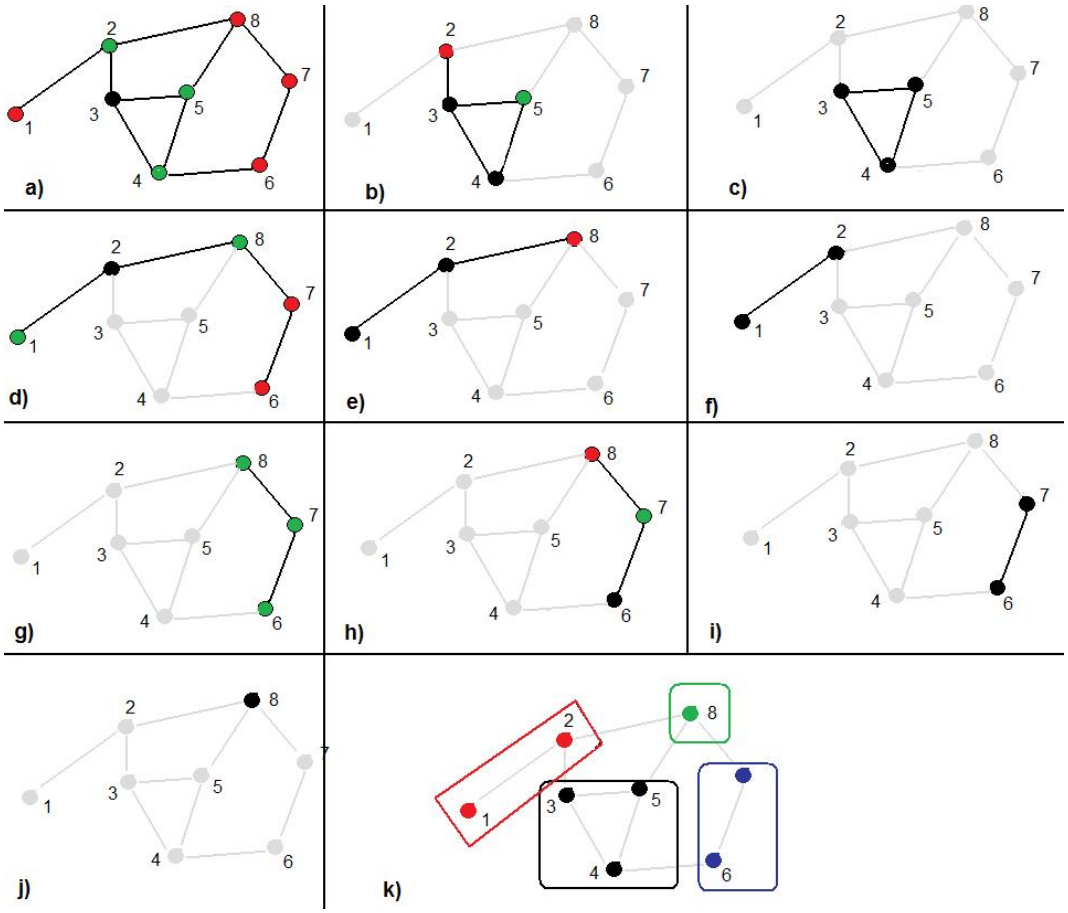


FIGURE 4.2.7.: Exemple d'application de l'algorithme CBB, $l_B = 2$: les nœuds en rouge correspondent aux nœuds "*écartés*" (distance géodésique $\geq l_B$), en noir aux nœuds *"centraux"*, en vert aux nœuds qui sont à distance géodésique $< l_B$. a) Sélection aléatoire du nœud 3 et écartement des nœuds à distance $\geq l_B$ du nœud 3. b) Sélection aléatoire du nœud 4 et écartement des nœuds à distance $\geq l_B$ du nœud 2. c) sélection du nœud restant 5. *Itération 1* : a), b) et c). *Itération 2* : d), e) et f). *Itération 3* : g), h) et i. *Itération 4* : j). La figure d) représente le résultat final (nombre de boîtes $N_B = 4$).

Le résumé de la technique MEMB est présenté dans l'algorithme 4.3 et la figure (4.2.8) illustre l'application de l'algorithme MEMB dans le cas de $r_B = 1$.

Algorithme 4.3 "Maximum-Excluded-Mass-Burning", MEMB

i) **Initialiser** tous les nœuds à "non_marqués",
ii) **Calculer** "la masse exclue" pour tous les nœuds "non_centrés" et "non_marqués", et choisir le nœud p de masse "exclue maximale" comme centre et poser le nœud p comme étant "marqué",
iii) **Poser** tous les nœuds i qui vérifient la relation $l_{ip} \leq r_b$ comme étant "marqués",
iv) **Répéter** les étapes ii) et iii) tant qu'il existe des nœuds "non_marqués" .

4.2.2.4. Résultats

Les figures (4.2.9-a), (4.2.9-b) et (4.2.9-c) représentent respectivement les résultats obtenus par application des algorithmes GC, CBB et MEMB aux *benchmarks* des réseaux IEEE 57 nœuds, 118 nœuds et 300 nœuds.

Les figures (4.2.10-a) et (4.2.10-b), quant à elles, représentent la comparaison des dimensions fractales

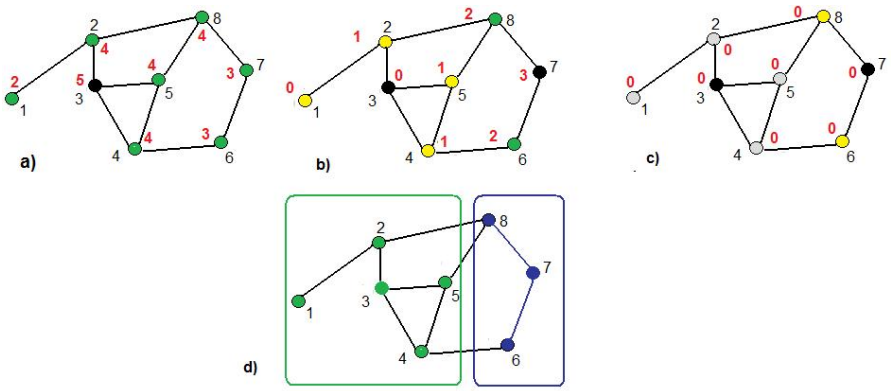


FIGURE 4.2.8.: Exemple d'application de l'algorithme MEMB, $r_B = 1$: les nœuds en vert correspondent aux nœuds *"non marqués"*, en jaune aux *"nœuds marqués"* et les nœuds noirs aux "*nœuds centraux*". Les nœuds en gris sont des nœuds qui appartiennent déjà à une boîte. La figure d) représente le résultat final (nombre de boîtes $N_B = 2$ = nombre de nœuds centraux).

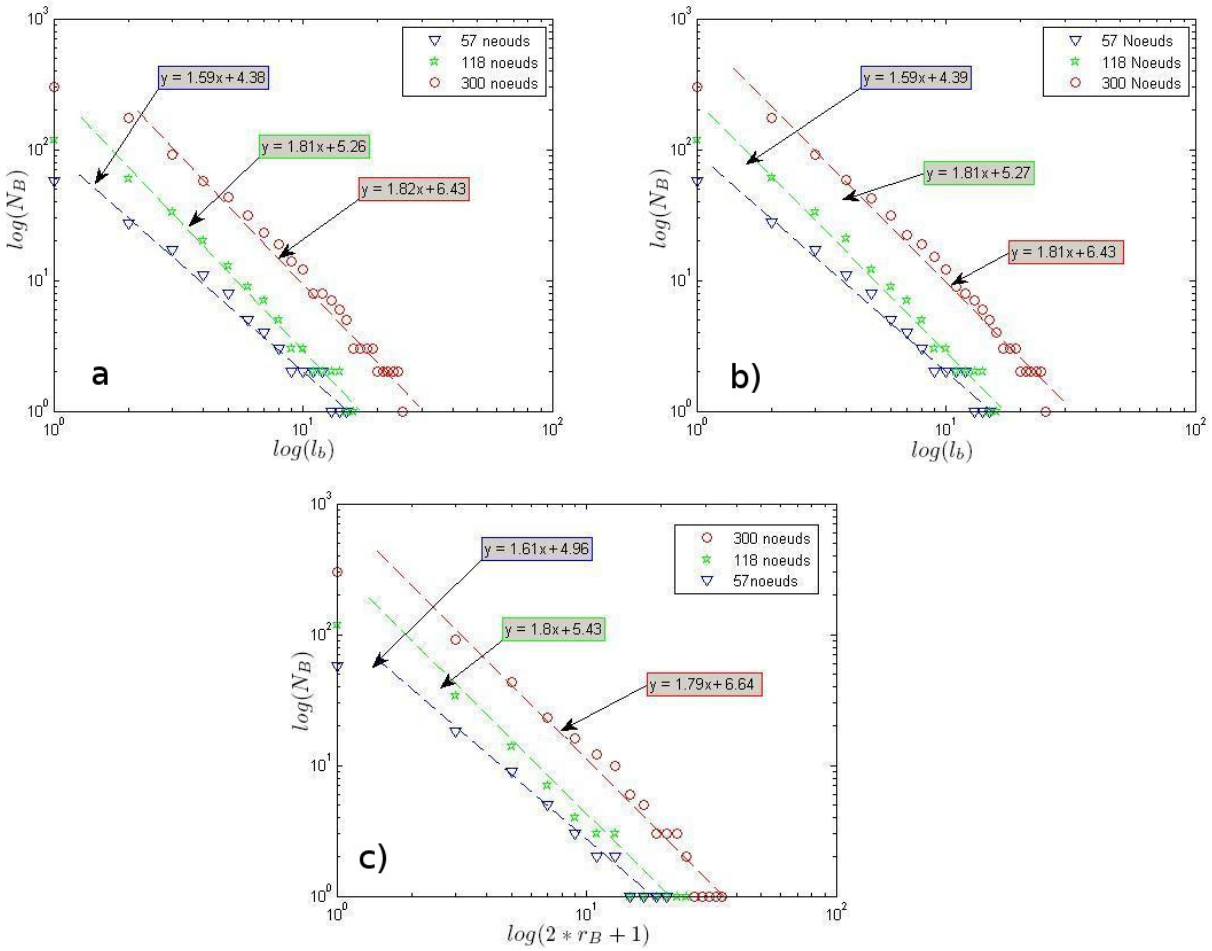
obtenues par l'application des 3 algorithmes au réseau IEEE 57 nœuds (le nombre de réalisations de chaque algorithme est de 10^2).

Enfin, la figure (4.2.11) représente les résultats obtenus par l'application des algorithmes CBB et MEMB au réseau colombien.

Les mesures faites sur les réseaux électriques considérés comme des *benchmarks* (IEEE à 57, 118 et 300 nœuds) laissent apparaître que la notion de dimension fractale peut avoir une signification. On trouve en effet un comportement linéaire sur environ deux décades et ce, dans tous les cas de figure. Bien que les trois algorithmes (GC, CBB et MEMB) donnent des résultats quasi similaires sur l'estimation de la dimension fractale sur les différents *benchmarks* (figure 4.2.10-a), on remarque néanmoins que la dimension fractale du réseau IEEE 57 nœuds (qui est d'environ 1.6) n'est pas la même que celles des réseaux IEEE 118 et 300 nœuds (qui sont d'environ 1.8). Toutefois, en prenant en considération que les trois sous-réseaux *benchmarks* proviennent du même réseau principal (le réseau Nord-Est américain), la figure (4.2.9) laisse à suggérer que la dimension fractale de ceux-ci est d'environ de 1.8. D'un autre côté, l'examen de la figure (4.2.11) indique que la détermination de la dimension fractale du réseau colombien, quelle que soit la méthode utilisée, amène à une valeur de $d = 2.1$.

Par ailleurs, la figure (4.2.10-b) montre que l'algorithme MEMB est légèrement moins sensible aux réalisations. En effet, les résultats obtenus après 1000 réalisations de l'algorithme MEMB sont plus proches de la moyenne que les deux autres algorithmes. Cela signifie que l'algorithme MEMB nécessite moins de réalisations pour nous fournir une "bonne" solution proche de l'optimum ; de plus, la complexité en temps de calcul de l'algorithme MEMB est meilleure que celles des deux autres algorithmes.

En conclusion, il apparaît que la notion de dimension fractale de graphe des réseaux électriques peut avoir une existence légitime. Nous n'allons pas plus loin dans l'interprétation, car des tests complémentaires doivent être établis pour renforcer la signification physique. En particulier, une des pistes que nous envisageons est de se placer dans le cadre de la relativité d'échelle développée par L. Nottale [Nott11], où la notion de dimension fractale devient relative et, en conséquence, n'est plus une constante.



c)

FIGURE 4.2.9.: Résultats des trois algorithmes appliqués aux réseaux IEEE 57 nœuds, 118 nœuds et 300 nœuds : a) Algorithme des couleurs (Greedy Algorithm), b) Algorithme CBB, c) Algorithme MEMB.

4.2.3. Robustesse du réseau

Dans les sections précédentes, quelques indicateurs de la topologie des réseaux électriques ont été esquissés. Ainsi, la détermination des caractéristiques générales d'un réseau, c'est-à-dire, la distribution des degrés, le coefficient de regroupement, le diamètre, ... nous a permis d'avoir un point de vue global (en statique) sur la robustesse d'un réseau, en le comparant à des modèles déjà existants (graphes "*Small World*", "libres d'échelle", aléatoires, ...). Dans cette section, nous allons nous intéresser à la robustesse des réseaux électriques ,en simulant des pannes sur ces derniers.

La sûreté d'un réseau se décompose en deux sous-problèmes, statique et dynamique :

- une étude de robustesse statique fait référence au fait qu'aucune considération de flux de puissance, et éventuellement de flux d'informations, n'est prise en compte.
- une étude de robustesse dynamique nécessite la prise en considération des flux de puissance circulant dans le réseau électrique. En d'autres termes, lorsqu'un nœud ou un lien du réseau électrique tombe en panne, une redistribution des flux de puissance est effectuée, afin de déterminer l'état $t+1$ du réseau.

Dans ce qui suit, on s'intéressera principalement à la robustesse statique des réseaux complexes.

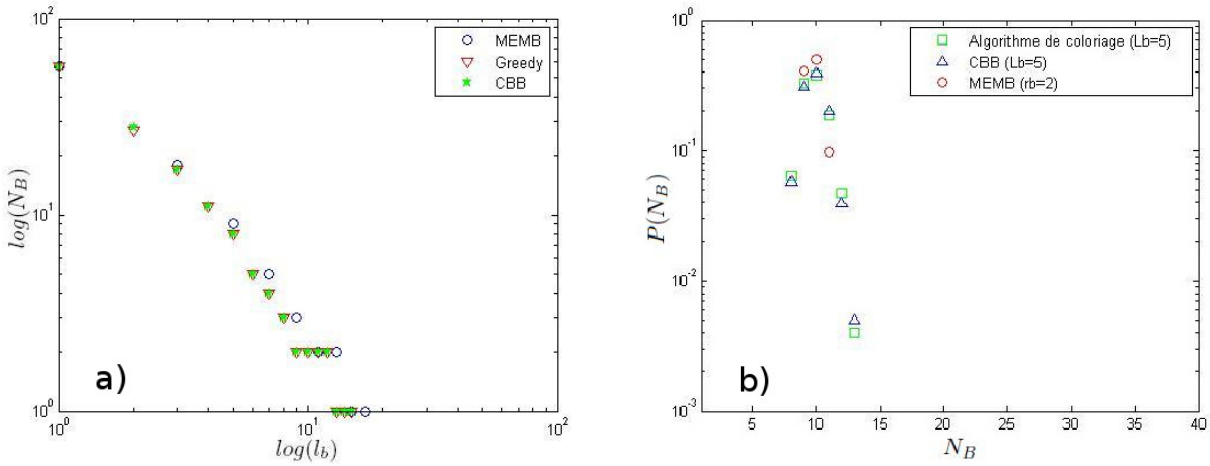


FIGURE 4.2.10.: a) Comparaison des dimensions fractales obtenues par l'application des 3 algorithmes au réseau IEEE 57 nœuds (le nombre de réalisations de chaque algorithme est de 10^2) b) Comparaison des distributions du nombre de boîtes N_B nécessaires pour couvrir le réseau IEEE 57 nœuds (le nombre de réalisations de chaque algorithme est de 10^3).

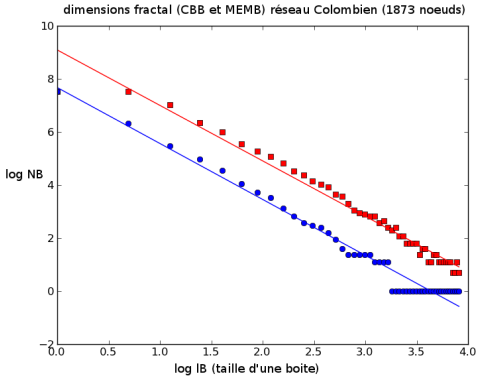


FIGURE 4.2.11.: Dimension fractale du réseau colombien (en rouge, algorithme CBB et en bleu algorithme MEMB).

Plusieurs mesures d'efficacité peuvent être utilisées comme indicateur de robustesse d'un réseau [Newm03, Doro08, Bocc06]. On peut par exemple étudier un réseau en quantifiant le plus grand composant (ou sous-réseau) connexe, après suppression volontaire et aléatoire d'une fraction d'éléments du système, *ce que l'on appelle panne*, ou après suppression d'une fraction d'éléments de plus haute importance (i.e. les nœuds de fort degré, de forte charge, etc), *ce que l'on appelle attaque*. Plus la taille du plus grand composant connexe est grande, plus le réseau reste fiable. Par ailleurs, la moyenne des chemins géodésiques L (équation (4.2.4)), qui peut être remplacée par l'efficacité globale E ou locale E_{loc} (qui sont les moyennes harmoniques des chemins géodésiques entre chaque nœud, équation (4.2.5)), est un autre indicateur de robustesse d'un réseau [Cruc05, Mott02]. Un réseau hautement robuste possède donc une efficacité globale proche de 1.

Une autre approche pour mesurer la robustesse d'un réseau consiste tout simplement à compter le nombre d'éléments défectueux dans le système [Carv09]. D'autres indicateurs de robustesse peuvent être définis suivant le type du réseau, comme par exemple la perte de connectivité C_L, utilisée dans [Albe04] :

$$C_L = 1 - < \frac{N_g^i}{N_g} > \tag{4.2.13}$$

où N_g est le nombre total de générateurs et N_g^i le nombre de générateurs connectés au nœud i.

L'étude de la robustesse d'un réseau en supprimant un élément du système s'appelle "étude N-1" (ou contingence N-1) [Bocc06, Roz09].

Les simulations de pannes (figure 4.2.12-a) et d'attaques (figure 4.2.12-b) ont été faites sur notre réseau test réel, le réseau électrique colombien. Les indicateurs tels que l'efficacité globale, la moyenne des chemins géodésiques, le diamètre et la taille du composant géant indiquent que le réseau colombien est vulnérable aux attaques et moyennement robuste aux pannes. En effet, on constate qu'il suffit d'enlever un ratio de 10% des nœuds les plus importants (cas d'une attaque) pour que le réseau " tombe ". En revanche, dans le cas de pannes, il faut un ratio de 50% des nœuds pour que celui ci s'affaisse.

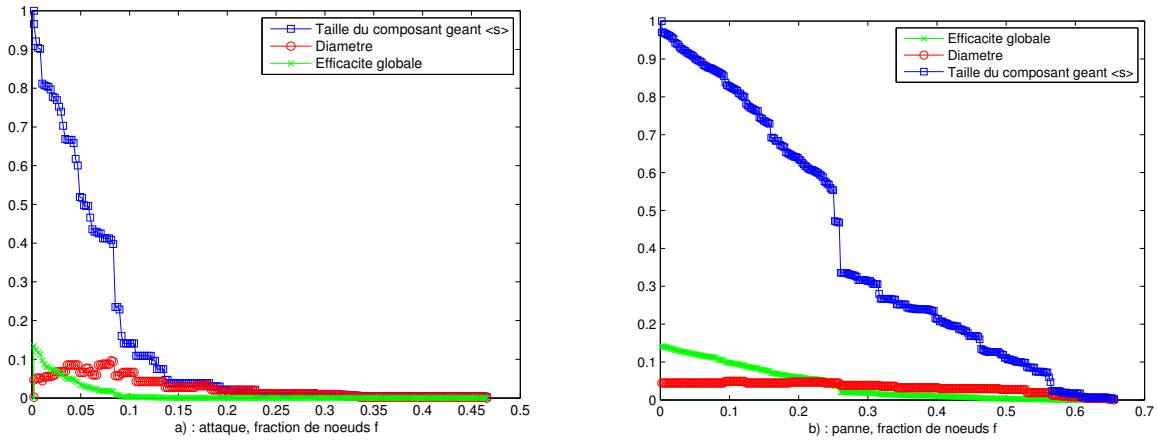


FIGURE 4.2.12.: Robustesse du réseau colombien : a) aux pannes, b) aux attaques

Toutefois, bien que le réseau soit libre d'échelle, il est néanmoins légèrement plus robuste aux attaques qu'un modèle de graphe libre d'échelle de type Barabàsi [Albe02]. En effet, il suffit d'enlever 2% des nœuds de plus fort degré du réseau Barabàsi pour que celui-ci s'affaisse.

Cette légère différence peut être expliquée par le caractère fractal du réseau colombien, que nous avons développé dans la section 4.2.2. Celui peut accroître la robustesse vis-à-vis des attaques intentionnelles si on la compare à la robustesse des réseaux libre d'échelle ayant le même exposant $\gamma = 3.26$. En effet, la propriété fractale fournit une meilleure protection lorsque les "hubs" sont retirés du système, par effet de dispersion (isolement des "hubs" entre eux).

4.3. Conclusion

Ce chapitre est consacré à l'étude topologique des réseaux électriques, notamment le réseau colombien. Dans un premier temps, quelques indicateurs et observables, qui nous ont servi de guide pour la description des réseaux électriques, ont été présentés puis appliqués au réseau électrique colombien.

Après examen des différents résultats obtenus, on peut tirer quatre conclusions principales concernant le réseau colombien :

- Le réseau colombien se révèle particulièrement vulnérable aux attaques (pannes intentionnelles) et moyennement robuste aux pannes aléatoires. Ainsi, il suffit que 10% des nœuds de plus fort degré tombent en panne pour que le réseau tout entier soit hors d'usage.
- Le réseau colombien présente des propriétés "libre d'échelle", ce qui expliquerait sa vulnérabilité vis-à-vis des attaques intentionnelles.
- Cependant, le réseau colombien - ce qui est également le cas du réseau nord-est américain - présente a priori des propriétés fractales, ce qui lui permettrait d'être un peu plus robuste aux attaques que les réseaux libres d'échelle.
- Enfin, en raison de son faible coefficient de regroupement, le réseau colombien n'est pas efficace "localement", d'autant plus qu'il n'appartient pas à la famille des réseaux "*Small Word*".

CHAPITRE 5

CONCEPTION D'UNE MÉTAHEURISTIQUE POUR L'OPTIMISATION DE LA TOPOLOGIE INTERNE D'UN PARC ÉOLIEN

5.1. Introduction

Une installation éolienne consiste à utiliser le vent pour faire tourner une hélice de grande taille. Lorsque le vent arrive sur l'hélice, la forme particulière des pales permet de la mettre aisément en mouvement. En hauteur, le vent est plus fort car libéré des rugosités du sol. L'hélice est donc placée en haut d'une tour de plusieurs dizaines de mètres. L'ajout d'un système automatique permet de placer l'éolienne face au vent, ce qui augmente son efficacité. Lorsque l'hélice se met à tourner, elle acquiert l'énergie du mouvement transmise par le vent. Cette dernière est ensuite transformée en énergie électrique par effet dynamo et envoyée dans le réseau.

Plusieurs éoliennes peuvent être regroupées sur un même site pour former un parc éolien. Dans ce cas, la maintenance est plus simple et la facturation est plus facile. En revanche, la stratégie d'exploitation et la gestion de la complexité et des risques encourus d'un tel parc deviennent délicates. Parmi ces difficultés, le raccordement des éoliennes au réseau n'est pas une tâche facile pour les opérateurs. En effet, compte tenu du coût élevé des câbles, du coût des départs et de leur éventuelle indisponibilité, une mauvaise interconnexion d'un parc éolien peut engendrer des pertes pécuniaires et sécuritaires très importantes.

Dans ce chapitre, l'objectif principal du travail effectué était de répondre à un besoin spécifique, à savoir la mise en place d'un outil de calcul permettant la détermination « optimale » de l'interconnexion d'un parc éolien. Cette étude a été volontairement réalisée sur une durée de temps limitée afin de valider, d'une part, l'apport des techniques d'optimisation, notamment celui des Algorithmes Génétiques et, d'autre part, de développer un outil de calcul exploitable directement par l'utilisateur.

Section (mm^2)	R (Ω/km)	L (mH/km)	C($\mu F/km$)	I admissible (A)	Coût (€/m)
50	0.82	0.48	0.14	119.07	3.65
95	0.41	0.41	0.17	174.15	4.59
150	0.26	0.38	0.21	221.13	5.1
240	0.16	0.35	0.27	290.79	6.5
240 copper	0.10	0.35	0.27	372.6	15.4

TABLE 5.1.: Paramètres de référence par phase pour câbles enterrés 33kV (ferme *onshore*)

5.2. Position du problème

L'énergie éolienne est l'une des composantes principales des sources d'énergie renouvelable. Outre la difficulté du design des parcs éoliens (position face au vent, hauteur des éoliennes, ...), le raccordement de ces dernières au réseau électrique s'avère être une tâche difficile aux yeux des experts.

5.2.1. Contexte

Les promoteurs de projets éoliens peuvent faire appel à des outils d'aide à la décision lors de la conception de leur parc, en particulier pour l'optimisation de l'architecture électrique interne. Leur besoin prend, notamment, en considération la définition :

1. de la topologie du réseau interne à la " ferme " (20kV ou 33kV),
2. du type de câble de ce réseau interne (section),
3. de l'emplacement du poste électrique (MT/HT) dans le cas d'un parc *offshore* avec plateforme,
4. de la configuration du poste électrique (MT/HT),
5. des besoins en compensation.

Les trois premiers points peuvent être formulés sous la forme d'un problème d'optimisation et résolus via une métaheuristique.

Dans cette phase de conception de la ferme, les promoteurs ont connaissance :

1. du schéma d'implantation des mâts (coordonnées X,Y des éoliennes),
2. de l'emplacement du poste (MT/HT) de raccordement au réseau électrique, et donc, du niveau de tension de raccordement (HT),
3. du type d'aérogénérateur (diagramme P, Q, U),
4. des paramètres des câbles (section, impédance, coût,...).

Les paramètres des câbles (section, impédance, coût,...) sont normalisés. En guise d'application de notre outil d'aide à la décision, les valeurs du tableau (5.1) seront utilisées pour des câbles enterrés. Néanmoins, ce type de tableau est dédié à un projet particulier.

La possibilité de placer deux câbles en parallèle est envisageable, si la puissance à évacuer l'exige. Dans ce cas-là, un facteur de déclassement est appliqué au courant maximal admissible dans les lignes en parallèle (typiquement 0.83).

Dans ce chapitre, un ensemble d'éoliennes raccordées en antenne auprès d'un poste électrique est appelé "grappe" ou "cluster". La figure (5.2.1) représente une topologie à 4 grappes. Le coût d'un départ MT est estimé à 20k€.

Il est à noter que le présent problème peut être modélisé sous forme de problème multi-objectif. En effet, l'augmentation du nombre de *clusters* :

- accroît le nombre de départs (dégradation du coût),
- réduit la section des câbles (réduction du coût),
- augmente la disponibilité de la ferme (la perte d'un départ est moins critique) et la facilité de maintenance.

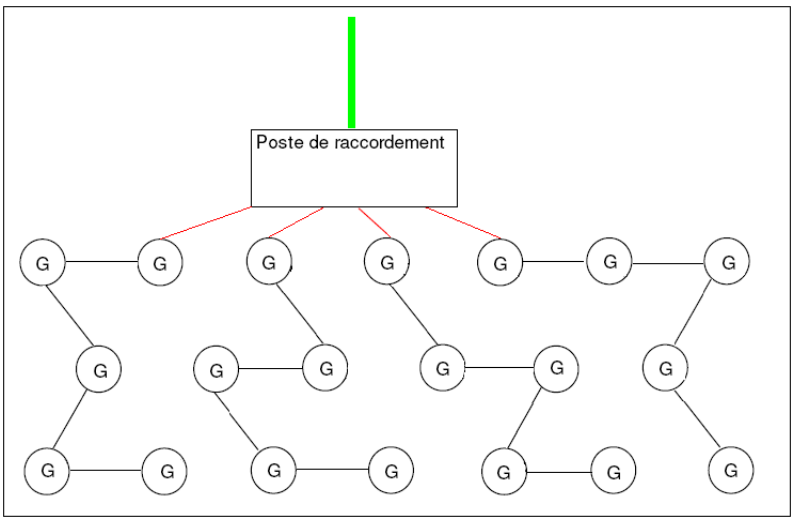


FIGURE 5.2.1.: Solution à 4 clusters

Dans ce cas, le nombre de départs (*clusters*) n'est plus une variable de décision, mais plutôt un critère à optimiser, ce qui réduira, sans aucun doute, le nombre de possibilités d'interconnexions (tableau (5.3)). En revanche, cela compliquera la résolution du problème. C'est pour cette raison que, dans un premier temps, nous nous sommes attachés à résoudre le problème sous sa forme mono-objectif.

Dans le cas d'une ferme en mer, il est possible qu'un poste MT/HT soit installé sur une plateforme. Le positionnement de cette plateforme est libre (moyennant une distance de sécurité de 100 m par rapport à l'éolienne la plus proche). Le poste en mer sur plateforme est ensuite raccordé en HT au poste d'atterrage du réseau électrique (via un câble sous-marin HT).

Le critère à minimiser est un critère de "coût" constitué de 4 coûts élémentaires, c'est-à-dire :

1. le coût des câbles (en fonction de leur longueur et de leur section),
2. le coût des pertes Joule (en fonction de la résistance et du courant traversant le câble),
3. le coût des départs (en fonction du nombre de grappes),
4. le coût de l'énergie non distribuée (en fonction notamment du nombre de grappes).

Comme dans la majorité des problèmes d'optimisation réels, les objectifs à minimiser sont soumis à un ensemble de contraintes, ce qui complique la résolution du problème. Dans le cas présent, la minimisation du « coût » doit tenir compte des contraintes suivantes :

1. Il convient de s'assurer que le nombre de sections différentes utilisées pour les câbles reste inférieur à une certaine valeur (typiquement, pas plus de 5 types de sections).
2. Les types de câble utilisés doivent être ceux du tableau (5.1).

3. Le croisement des câbles est à éviter.
4. La chute de tension le long d'une grappe doit être inférieure à 2%.
5. Chaque éolienne ne peut avoir qu'un ou deux départs (voir figure (5.2.2)).
6. Toute les éoliennes doivent être raccordées (directement ou via d'autres éoliennes) au poste de raccordement.
7. Le positionnement des mâts est non révisable.
8. Le courant admissible pour chaque câble doit être supérieur au courant circulant dans le câble en question.

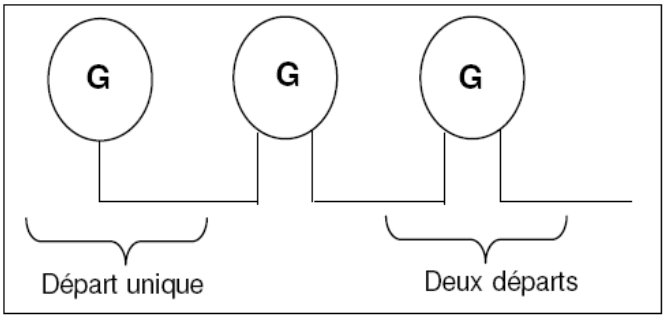


FIGURE 5.2.2.: Raccordement d'une éolienne (1/2 départs).

5.2.2. Calcul des flux de puissance dans les câbles de liaison des éoliennes

Pour les calculs de flux de puissance, on se place dans le cas le plus critique vis-à-vis des pertes, du courant maximum permanent, et de la chute de tension, à savoir :

1. un fonctionnement à P_{max}, Q_{max} pour chaque éolienne,
2. une tension minimale au point de livraison (poste électrique MT/HT).

Par ailleurs, on notera que les câbles sont modélisés par un modèle électrique en « π ».

5.2.2.1. Calcul de l'énergie non distribuée

L'algorithme d'optimisation a besoin, entre autres, d'un critère de coût (ou fonction objectif). Ce dernier est basé sur le coût des pertes Joule dans les câbles et sur le coût de l'investissement initial lié aux longueurs et aux types de câble utilisé. Il est également nécessaire de rappeler que les tensions calculées en chacun des nœuds vérifient les contraintes imposées (la chute de tension le long d'une grappe doit être inférieure à 2%).

Les pertes Joule sont déterminées par le calcul du courant dans chaque câble (par l'intermédiaire d'un calcul de type *AC Power Flow*, cf section 3.2). Ce dernier fournit également la tension en chaque nœud. Les pertes et les tensions ainsi obtenues seront utilisées pour le calcul de la fonction objectif décrite ci-dessous.

L'objectif est de déterminer le coût de l'énergie non distribuée actualisé sur 20 ans. En effet, si défaillance des disjoncteurs ou des transformateurs il y a, l'énergie produite par les éoliennes peut ne pas être distribuée. Cela représente un manque à gagner qu'il faut chiffrer.

A chaque départ est associé un temps moyen d'indisponibilité μ par an (typiquement $\mu_c = 5h/an$). Le temps moyen d'indisponibilité μ_f de la ferme est donc :

Prix de rachat MWh(€)	180
Facteur de charge	0.34246575
Taux d'actualisation	2%
Temps d'indisponibilité d'un *cluster* (μ_c *en* h/an)	5

TABLE 5.2.: Paramètres à utiliser pour le calcul de l'énergie non distribuée.

$$\mu_f = N_d \times \mu_c \tag{5.2.1}$$

De là, le coût de l'énergie non distribuée Σ_e s'exprime comme :

$$\Sigma_e \quad = \quad \sum_{i=1}^{20} \frac{P_{max} \times \kappa \times \mu_f}{(1+\tau)^i} \times \psi \tag{5.2.2}$$

Au final, la formule du coût de l'énergie non distribuée est :

$$\Sigma_e \quad = \quad P_{max} \times \kappa \times \mu_f \times \frac{\left(1 - \left(\frac{1}{1+\tau}\right)^{20}\right)}{\tau} \times \psi \tag{5.2.3}$$

en désignant par :

- N_d, le nombre de départs (*clusters*),
- P_{max}, la puissance maximale (puissance nominale) de la ferme éolienne,
- κ, le facteur de charge (i.e. puissance moyenne réellement produite sur la puissance nominale),
- τ, le taux d'actualisation,
- ψ, le prix de rachat du KW/h.

Le calcul de l'énergie non distribuée est réalisé sur la base des données du tableau (5.2).

5.2.2.2. Calcul du coût des pertes

Les pertes Joule sont évaluées sur la base de la formule $\varphi = RI^2$.

Le prix de rachat ψ du MWh permet de chiffrer le coût des pertes (tableau (5.2)). Ces pertes doivent également être actualisées sur 20 ans. Ce qui revient à dire que les coûts des pertes Σ_p sont calculés comme suit (équation (5.2.4)) :

$$\Sigma_p \quad = \quad \varphi \times \kappa \times \frac{\left(1 - \left(\frac{1}{1+\tau}\right)^{20}\right)}{\tau} \times \psi \times N_h \tag{5.2.4}$$

A ce stade de formalisme, tous les ingrédients nécessaires à la compréhension, aux calculs des coûts de l'installation et à la modélisation du problème de l'interconnexion d'un parc éolien sont esquissés. En résumé, l'outil d'aide à la décision, qui sera détaillé dans les paragraphes suivants, prend comme paramètres et variables d'entrée :

- Schéma d'implantation des mâts sous forme de coordonnées (X, Y),
- Emplacement du poste d'atterrage sous forme de coordonnées (X, Y),
- Type d'aérogénérateur (P_{max}, Q_{max}),

- Paramètres des câbles à utiliser (section, R, L, C, I admissible, Coût, $U_{nominal}$),
- Coût d'un départ,
- Prix de rachat du MWh,
- Facteur de charge prévisionnel de la ferme,
- Temps d'indisponibilité d'un cluster,
- Taux d'actualisation.

et en sortie, le programme d'optimisation qui sera mis en œuvre renvoie, comme variables de sortie :

- La topologie du réseau interne à la ferme,
- Les types des câbles de ce réseau interne (sections),

minimisant les pertes Σ dans le réseau interne du parc (équation (5.2.5)) :

$$\Sigma = \Sigma_e + \Sigma_p \tag{5.2.5}$$

où Σ_e et Σ_p sont, respectivement, le coût de l'énergie non distribuée et le coût des pertes, définis ci-dessus.

Bien que le problème décrit dans cette section soit purement combinatoire et non hybride, il présente néanmoins une complexité explosant très rapidement avec le nombre d'éoliennes à raccorder. En effet, le nombre de possibilités (de configurations) est de l'ordre de :

$$(n-1) \times n! \times (Nb\_Type)^n$$

où n désigne le nombre d'éoliennes présentes dans la ferme et Nb_Type désigne le nombre de types de câbles disponibles, soit $(n-1) \times (Nb\_Type)^n$ fois plus complexe que le problème du voyageur de commerce !

Le tableau 5.3 montre l'explosion combinatoire du problème. On suppose qu'une configuration est évaluée en $1\,\mu s$.

Nombre d'éoliennes	Nombre de possibilités	Temps de calcul
5	1500000	1.5 secondes
10	3.1894e+014	10 ans
15	5.5870e+023	17716 milliards d'années
20	4.4084e+033	1.3979e+020 d'années
30	7.1640e+054	2.2717e+041 d'années

TABLE 5.3.: Nombre de possibilités pour 5 types de câble en fonction du nombre d'éoliennes.

5.3. Algorithmes génétiques et adaptation à notre problème

5.3.1. Codage de la solution

Le codage des individus est une phase cruciale dans la conception d'un algorithme génétique. Il doit couvrir entièrement l'espace de recherche et respecter, entre autres, le maximum de contraintes. Les opérateurs de croisement et de mutation doivent être facilement adaptables au type de codage choisi.

Dans le cadre de notre problème, un individu (chromosome) est représenté comme suit :

$$(E_1, E_2, ..., E_n|1, l_1, l_2, ..., l_{n-1}) \tag{5.3.1}$$

où $E_i \in \mathbb{N}$, $(E_1, E_2, ..., E_n)$ forment une permutation, $l_i \in \{1, -i\}$.

- Lorsque l_i vaut 1 , cela veut dire que l'élément E_{i+1} de la permutation est connecté au poste de raccordement.

- Lorsque l_i vaut $-i$, cela veut dire que l'élément E_{i+1} de la permutation est connecté à l'élément E_i de la même permutation.

Un tel codage assure que toutes les éoliennes soient connectées et que chaque éolienne ne soit connectée au plus qu'à deux éléments (éolienne ou poste de raccordement).

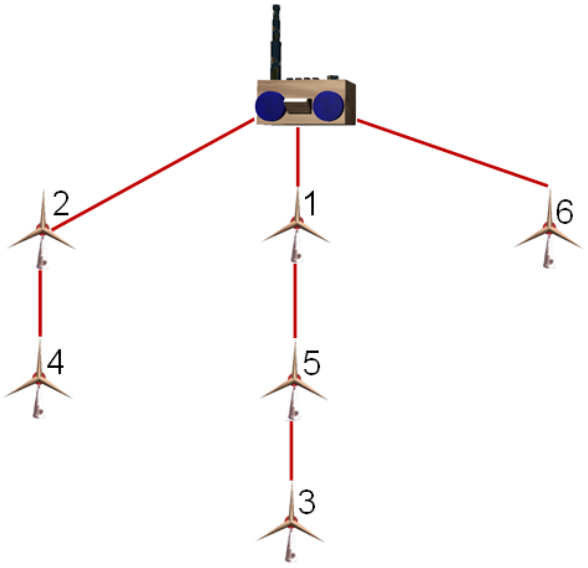


FIGURE 5.3.1.: Graphe correspondant à l'individu *Exemple_Codage*.

Exemple : Le graphe correspondant à l'individu *Exemple_Codage* suivant est représenté sur la figure (5.3.1) :

$$Exemple\_Codage = (2, 4, 1, 5, 3, 6|1, -1, 1, -3, -4, 1)$$

Dans cet exemple, les éléments de la permutation E_i, $i \in \{1, ..., 6\}$ correspondent respectivement aux éoliennes 2, 4, 1, 5, 3 et 6 et les éléments l_j, $j \in \{1, ..., 6\}$ valent respectivement 1, -1, 1, -3, -4 et 1. Les éléments l_1, l_3 et l_6 valent 1, ce qui signifie que les éoliennes 2, 1 et 6 sont connectées au poste de raccordement. Par ailleurs, l'élément l_4 vaut -3, ce qui signifie que l'élément E_4 de la permutation, soit l'éolienne 5, va être raccordée à l'élément E_3 de la permutation, soit l'éolienne 1 (figure (5.3.1)).

On remarquera que l'élément E_1 dans l'équation (5.3.1) vaut toujours 1, ce qui signifie que l'éolienne correspondant au premier élément E_1 de la permutation est, par défaut, connectée au poste de raccordement. En effet, une topologie qui respecte les contraintes prédéfinies dans 5.2.1 doit avoir au moins une éolienne connectée au poste.

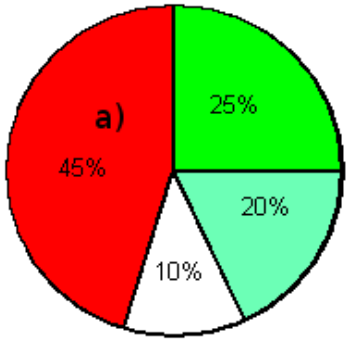


FIGURE 5.3.2.: Exemple de roulette biaisée.

5.3.2. Opérateur de sélection

Bien que cette phase ne présente aucune difficulté d'adaptation d'un quelconque opérateur de sélection, elle fait partie des phases cruciales des algorithmes génétiques. En effet, une mauvaise sélection des individus conduit systématiquement à une convergence prématurée de l'algorithme ou, le cas échéant, à la divergence de l'algorithme.

En règle générale, les parents sont sélectionnés pour la phase de reproduction (croisement) en fonction de leur performance (équation (5.3.2)) où $f(A_i)$ et $P[A_i]$ sont, respectivement, la performance et la probabilité avec laquelle le chromosome A_i sera réintroduit dans la nouvelle population.

$$P[A_i] = \frac{f(A_i)}{\sum_{j=1}^{N} f(A_j)} \tag{5.3.2}$$

Dans notre cas, l'opérateur de sélection mis en œuvre est celui de la roulette biaisée. Pour ce faire, on procède de la manière suivante :

1. On trie les individus de la population par ordre de performance à l'aide de l'équation (5.3.2).
2. On attribue une section de surface pour chaque individu, du meilleur au plus mauvais
3. On génère un nombre aléatoire dans $[0, 1]$.
4. On cherche à quel intervalle appartient le nombre généré et on sélectionne l'individu correspondant.

Par exemple, sur 4 individus :

Si la somme des performances vaut 10 et que les performances de chaque individu sont respectivement 2.5, 2, 1 et 4.5, l'état de la roulette biaisée correspondant sera représenté comme sur la figure 5.3.2 :

L'individu correspondant à la surface (a) a 45% de chances d'être sélectionné alors que l'individu correspondant à la surface blanche n'a que 10% de chances d'être sélectionné.

Pour garantir l'élitisme de notre algorithme, on gardera à chaque génération le meilleur individu. Par ailleurs, pour éviter une convergence prématurée de l'algorithme, on gardera systématiquement quelques mauvais individus (typiquement 2).

5.3.3. Croisement

L'opérateur de croisement utilisé dans la section (5.4) est celui décrit dans [Mora04, KimY06]. Son principe est le suivant :

1. Étant donné deux parents $P_1 = (v_1^1, ..., v_n^2)$ et $P_2 = (v_1^2, ..., v_n^2)$,
2. Sélectionner, aléatoirement, 2 positions de croisement. Soient Pos_1 et Pos_2 ces positions,
3. Recopier la partie $[v_{Pos_1}^1, .., v_{Pos_2}^1]$ du parent P_1 dans la partie $[v_{Pos_1}^2, ..., v_{Pos_2}^2]$ du parent P_2 ,
4. Supprimer les éléments $v_{Pos_1}^1, .., v_{Pos_2}^1$

L'opérateur est alors défini en 2 phases et procède comme suit :

PHASE 1 (Recherche des cycles)

- Étant donné deux parents P_1 et P_2.
- Considérer les positions des éléments correspondant aux permutations des deux parents comme étant non marquées.
- Poser $i = 1$.

1. Générer aléatoirement une position non marquée (soit Pos_i, cette position et Elt_i, l'élément correspondant dans la permutation de l'individu P_1). Marquer la position Pos_i par i.

2. Soit y l'élément de la permutation du parent P_2 correspondant à la position Pos_i.

   - **Si** $y = Elt_i$ poser $i = i + 1$ et aller à 1 (s'il reste des positions non marquées)

   - **sinon** poser Pos_i comme la position correspondant à l'élément y de la permutation du parent P_1 et marquer la position Pos_i par i et aller à 2.

3. A la fin des étapes précédentes, on obtient une suite de chiffres indiquant à quel cycle appartient chaque élément des deux permutations de P_1 et P_2. Noter $Cycle$, cette suite de chiffres.

PHASE 2 (mixer les cycles obtenus)

1. Créer aléatoirement un masque M de croisement dont les éléments correspondants indiquent de quel parent l'enfant va hériter dans chaque cycle. Il y a donc autant d'éléments dans le masque que de cycles détectés (exemple : s'il y a 3 cycles détectés, le masque sera $M = P_1P_2P_1$).
2. Convertir la suite de chiffres $Cycle$ en une permutation à partir du masque de la manière suivante : $Cycle(i) \leftarrow M(i)$. Par exemple, si l'élément i de la suite de chiffres $Cycle$ vaut 2 et que $M = P_1P_2P_1$, alors $Cycle(i) \leftarrow P2$. Ce qui veut dire que l'élément 2 de la permutation de l'enfant obtenu héritera l'élément 2 de la permutation du parent P_2.

Exemple : Étant donné deux permutations $A = (5, 6, 2, 3, 1, 4)$ et $B = (6, 5, 2, 1, 3, 4)$. L'application du *crossover* géométrique à ces deux parents se déroule de la manière suivante (figure 5.3.3) :

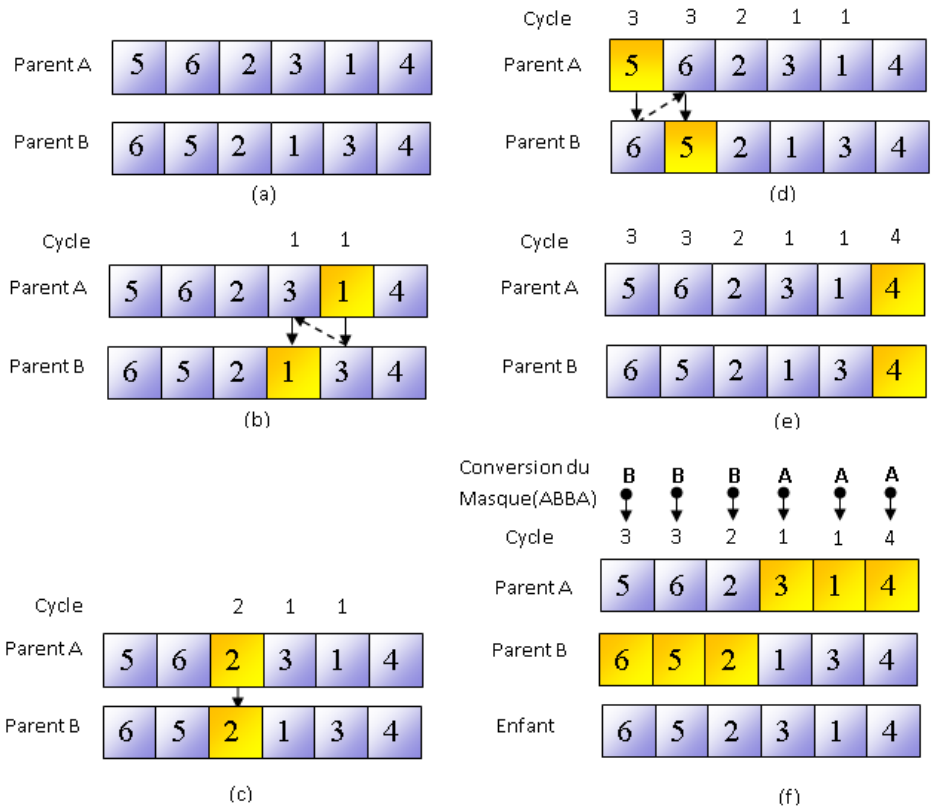


FIGURE 5.3.3.: Étapes du *crossover géométrique* appliqué aux deux parents A et B.

Étant donnés les deux parents sur la figure (5.3.3-a). Posons $i = 1$ et générons une position entre 1 et 6 (6 étant le nombre de gènes ou d'éléments dans notre codage), soit $Pos_1 = 5$ cette position. On attribue alors le cycle $i = 1$ à cette position. La valeur de l'élément Elt_1 correspondant au parent A vaut 1 et celle correspondant au parent B vaut 3. On cherche le gène qui prend comme valeur 3 dans le parent A, ce gène a comme position 4. On attribue alors le cycle $i = 1$ à la position 4. L'élément correspondant à la position 4 du parent B vaut 1, qui correspond aussi à l'élément de départ Elt_1, on obtient alors un cycle (figure 5.3.3-b). On pose $i = 2$, c'est-à-dire on cherche un 2ème cycle, et on refait la même procédure jusqu'à obtenir tous les cycles (toutes les positions sont marquées).

Supposons maintenant que l'on ait identifié tous les cycles (figure 5.3.3.e). La séquence des cycles identifiés vaut alors, dans l'ordre, {3, 3, 2, 1, 1, 4}. Le nombre de cycles obtenus est de 4, on génère alors un masque de 4 valeurs dans $\{A, B\}$, soit $ABBA$ le masque ainsi généré. Ce qui signifie que l'on attribue le premier cycle au parent A, le second et le troisième au parent B et enfin le quatrième au parent A. Pour convertir la séquence {3, 3, 2, 1, 1, 4} en une permutation, il suffit donc de remplacer les valeurs 1, 2, 3 *et* 4 par A, B, B et A respectivement, ce qui donne : $\{3, 3, 2, 1, 1, 4\} \Longrightarrow \{B, B, B, A, A, A\}$.

La figure (5.3.4.c) représente le graphe de l'enfant obtenu par le croisement des parents A et B de la figure (5.3.3). Cette figure montre que l'enfant obtenu a bel et bien hérité des deux parents. En effet, les éoliennes 1 et 4 de l'enfant ne sont connectées qu'au poste de raccordement ce qui est la signature du parent A. Par ailleurs, le raccordement des éoliennes 6, 5 et 3 porte la même signature que le parent B.

Remarque. Dans notre cas d'application, le *crossover* géométrique est appliqué sur les permutations des individus avec une probabilité élevée, alors que le *crossover* classique décrit dans la section 2.2 est

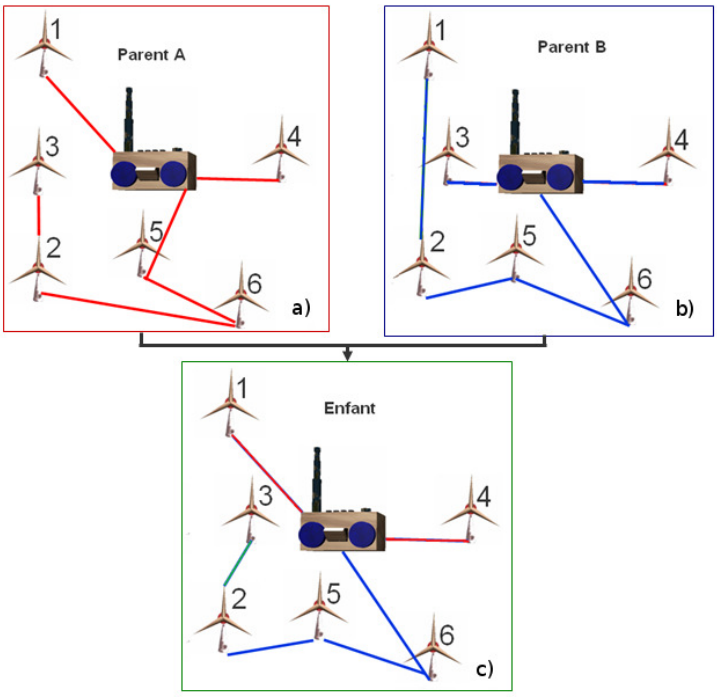


FIGURE 5.3.4.: Représentation graphique de l'exemple du croisement de la figure 5.3.3 : a) parent **A**, b) parent **B**, c) l'enfant obtenu par le croisement du parent **A** et **B**.

appliqué sur la deuxième partie (les liaisons l_i) des individus, avec une probabilité faible.

5.3.4. Mutation

Contrairement au croisement, les opérateurs de mutation standards peuvent - généralement - être adaptés à n'importe quel type de codage. Dans le cas de notre codage, la mutation des individus s'effectue de la manière suivante :

1. Générer une position Pos_1 dans laquelle la mutation peut avoir lieu.

2. **Si** la position Pos_1 correspond à la partie « permutation » de l'individu alors (figure 5.3.5-a) :

   a) Générer une autre position Pos_2, correspondant à la partie « permutation » de l'individu.

   b) Permuter les deux éléments correspondant aux deux positions Pos_1 et Pos_2.

3. **Sinon**, avec des probabilités équitables :

   a) Augmenter, si possible, de 1 le nombre d'éoliennes connectées au poste de raccordement, c'est-à-dire :

      i. générer une position dans la deuxième partie de l'individu dont les éléments sont négatifs,

      ii. muter l'élément correspondant en 1 (figure 5.3.5-b).

   b) Diminuer, si possible, de 1 le nombre d'éoliennes connectées au poste de raccordement, c'est-à-dire :

i. générer une position dans la deuxième partie de l'individu dont les éléments valent 1, soit la position Pos_i,

ii. muter l'élément correspondant en $-i$ (figure 5.3.5-c).

c) Changer aléatoirement la position de l'un des éléments valant 1 de la deuxième partie de l'individu (ne pas toucher au premier 1, car le premier élément de la permutation est toujours connecté au poste de raccordement) (figure 5.3.5-d).

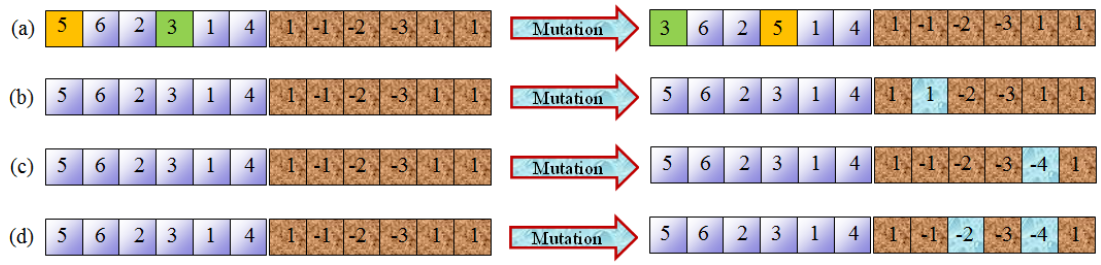


FIGURE 5.3.5.: Exemples de mutation d'un individu.

Ainsi, l'opérateur de mutation tel que défini permet d'atteindre n'importe quelle topologie, à savoir, la modification de l'ordre des éoliennes d'une grappe (figure 5.3.6-b), l'ajout d'une grappe (figure 5.3.6-c), la suppression d'une grappe (figure 5.3.6-d) et enfin, la modification du nombre d'éoliennes de chaque grappe (figure 5.3.6-e).

En outre, la gestion des contraintes dans le paragraphe ((5.2.1)) devient plus simple, à savoir :

1. Le codage ainsi défini dans le paragraphe (5.3.1) prend en considération les contraintes 5 et 6, discutées dans le paragraphe (5.2.1).

2. La contrainte 2 est une variable d'entrée ; elle est donc considérée a priori et ne peut pas être violée, de même pour la contrainte 7.

3. La contrainte 4 est prise en considération en pénalisant la fonction objectif en cas de violation.

4. La mise en parallèle de 2 câbles, éventuellement plusieurs câbles, permet de satisfaire la contrainte 8 en cas de surcharge (i.e., courant trop élevé).

5. La contrainte 3 n'est pas prise en compte explicitement dans le processus d'optimisation. Cependant, en raison du coût élevé des câbles, une solution optimale comportera nécessairement peu de croisements, voire aucun (dans un quadrilatère, la somme des deux diagonales est supérieure à la somme des deux côtés). De plus, les solutions comportant des câbles trop longs seront pénalisées, ce qui favorisera la connexion des éoliennes avec leurs plus proches voisines.

5.4. Application

Bien qu'une ferme ne comporte - en moyenne - qu'une vingtaines d'éoliennes, elle fait partie néanmoins du monde des réseaux électriques. Par conséquent, comme tout réseau électrique, elle est régie par les

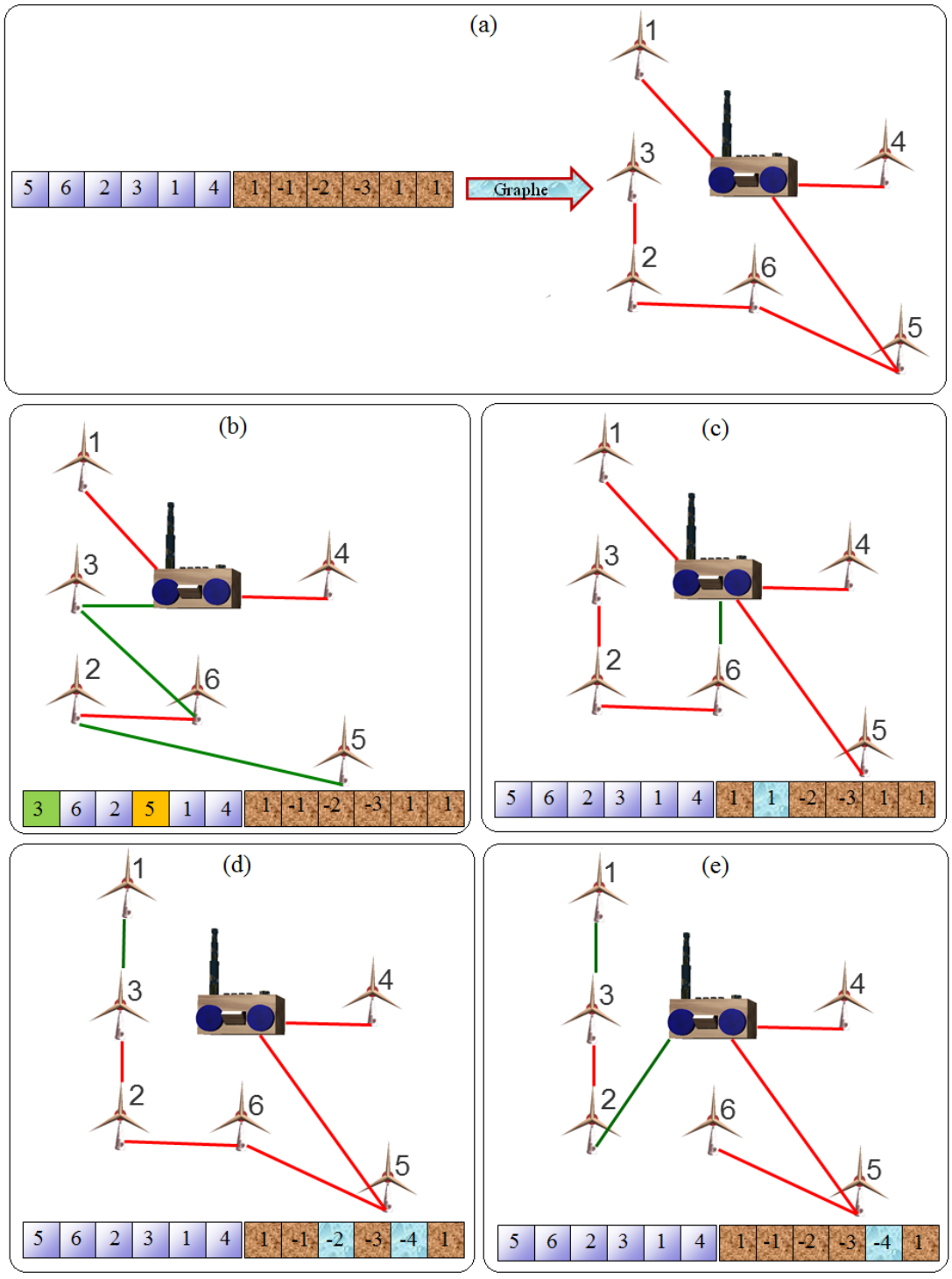


FIGURE 5.3.6.: Exemples de mutation : a) le graphe de l'individu de la figure 5.3.5, b), modification de l'ordre des éoliennes d'une grappe, c) ajout d'une grappe, d) suppression d'une grappe, d) modification du nombre d'éoliennes d'une grappe.

équations et les contraintes du modèle *Power Flow* décrit dans la section 3.2.1. Ainsi, les flux de puissance circulant dans une topologie donnée ne sont valides que s'ils satisfont les équations de la section 3.2.1. Le système non linéaire de ce modèle peut aisément être résolu par des méthodes de descente, notamment des méthodes du type Newton Raphson.

Le calcul du courant dans chaque câble et de la tension en chaque nœud est réalisé à partir d'une fonction (NewtonPF) du module Pylon, qui est une traduction en langage Python 2.6 du programme *Matlab Matpower* de calcul de *Power Flow*.

Cette fonction a pour entrée la description (topologie) du réseau de câbles interconnectant les éoliennes, soit :

1. Définition des bus (type de bus, demande de puissance active et réactive,...),
2. Définition des branches entre les bus (numéros des bus connectés pour chaque branche, résistance p.u., réactance p.u., susceptance p.u.,...),
3. Définition des générateurs (numéros des bus auxquels sont connectés les générateurs, puissances actives et réactives générées,...).

La fonction *NewtonPF* fournit, en sortie, la tension en p.u. en chaque nœud (phase et amplitude) et la matrice d'admittance.

Les courants en p.u. dans les branches (phase et amplitude) sont calculés grâce à la matrice d'admittances déterminées par *NewtonPF*. Un test est également effectué pour savoir si les courants dans les branches sont compatibles avec le courant admissible par le type de câble de chaque branche. On corrige, le cas échéant, en tenant compte d'un facteur de déclassement, le nombre de câbles en parallèle, de façon à respecter cette contrainte. Dans ce cas-là, si le nombre de câbles en parallèle a été modifié sur au moins une branche, on relance un calcul de *Power Flow*.

Tout le long des applications qui suivent, les paramètres de l'algorithme seront fixés comme suit :

1. Nombre d'itérations : 1000 (fixé pour un temps de calcul " raisonnable "),
2. Taille de la population : 50,
3. Probabilité de mutation : $P_m = 0.1$,
4. Probabilité de croisement : $P_c = 0.8$,
5. P_{max} : 1 MW,
6. Q_{max} : 0.283 MW,
7. Nombre d'individus régénérés : 20% de la population initiale,
8. Types des câbles : ceux du tableau (5.1) du paragraphe (5.2.1),
9. Prix de rachat, Facteur de charge, Taux d'actualisation et Temps d'indisponibilité d'un *cluster* : ceux du tableau (5.2) du paragraphe (5.2.1),
10. Coût d'un départ : 20 k€.

5.4.1. Application à des fermes de 15 et 20 éoliennes

La figure (5.4.1) montre le câblage proposé par l'algorithme d'une ferme comprenant 15 éoliennes. Le résumé des *outputs* est présenté dans le tableau (5.4).

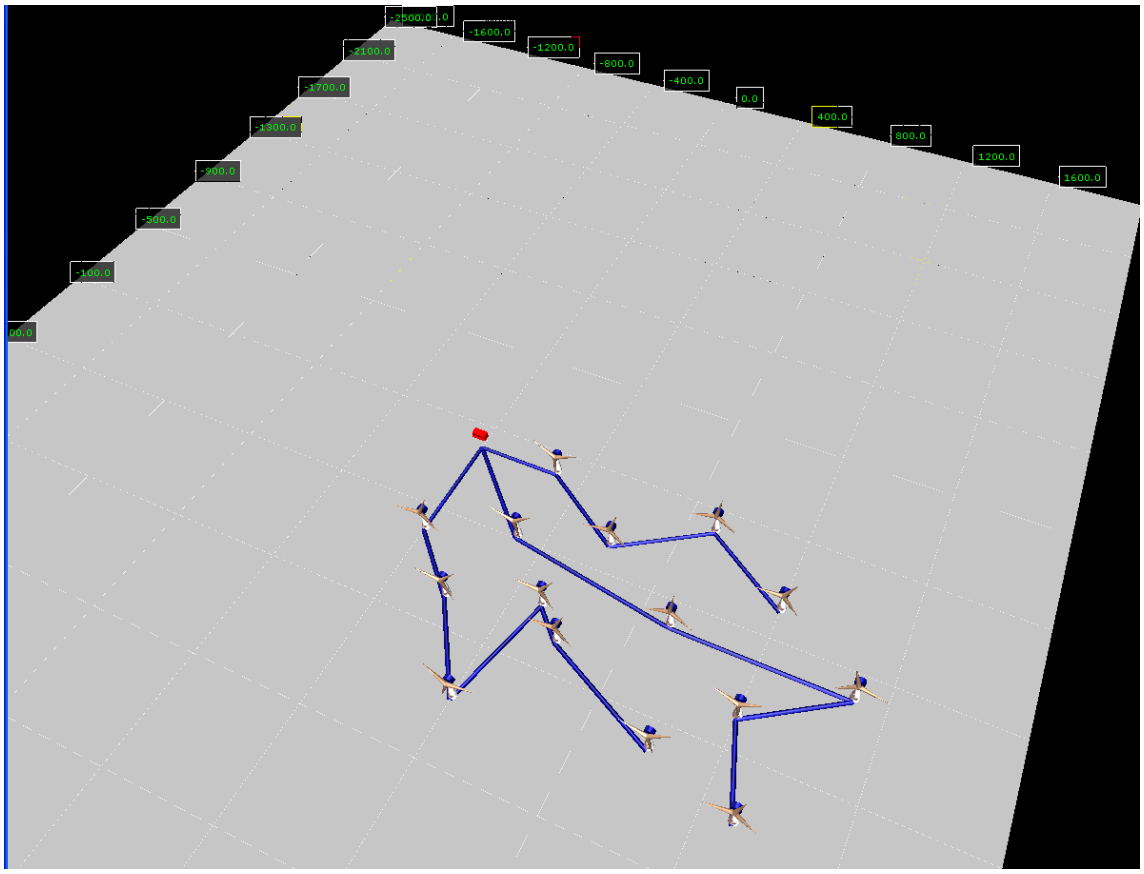


FIGURE 5.4.1.: Topologie d'une ferme de 15 éoliennes obtenue par l'algorithme génétique.

Coût_energie_nd (€)	Coûts des pertes (€)	Coûts des Câbles (€)	Coûts totaux (€)
226 792	132 911	106 425	466 128

Puissance perdue (Joule)	Tension max (p.u)	Tension min (p.u)	Nombre de départs	Itér
0.015	0.952	0.95	3	998

TABLE 5.4.: Résumé des *outputs* d'une solution correspondant à une ferme de 15 éoliennes

La figure (5.4.2) montre le câblage proposé par l'algorithme d'une ferme comprenant 20 éoliennes. Le résumé des *outputs* est présenté dans le tableau (5.5).

5.4.2. Application à une ferme de 30 éoliennes

La figure (5.4.3) montre le câblage proposé pour une ferme comprenant 30 éoliennes. Le résumé des *outputs* est présenté dans le tableau (5.6). La figure en haut et à gauche représente l'évolution des des trois critères économiques (le coût de l'énergie non distribuée, le coût des câbles et le coût des pertes Joule) définis dans le paragraphe 5.2.1. La figure au centre et à gauche représente l'évolution de la fonction objectif du problème en question (la somme des trois coûts économiques définis dans 5.2.1). La figure à gauche et en bas représente les chutes de tension aux bords des éoliennes (en pourcentage par rapport à la tension nominale). Enfin, la figure à droite représente la topologie de la ferme éolienne proposée par l'algorithme génétique.

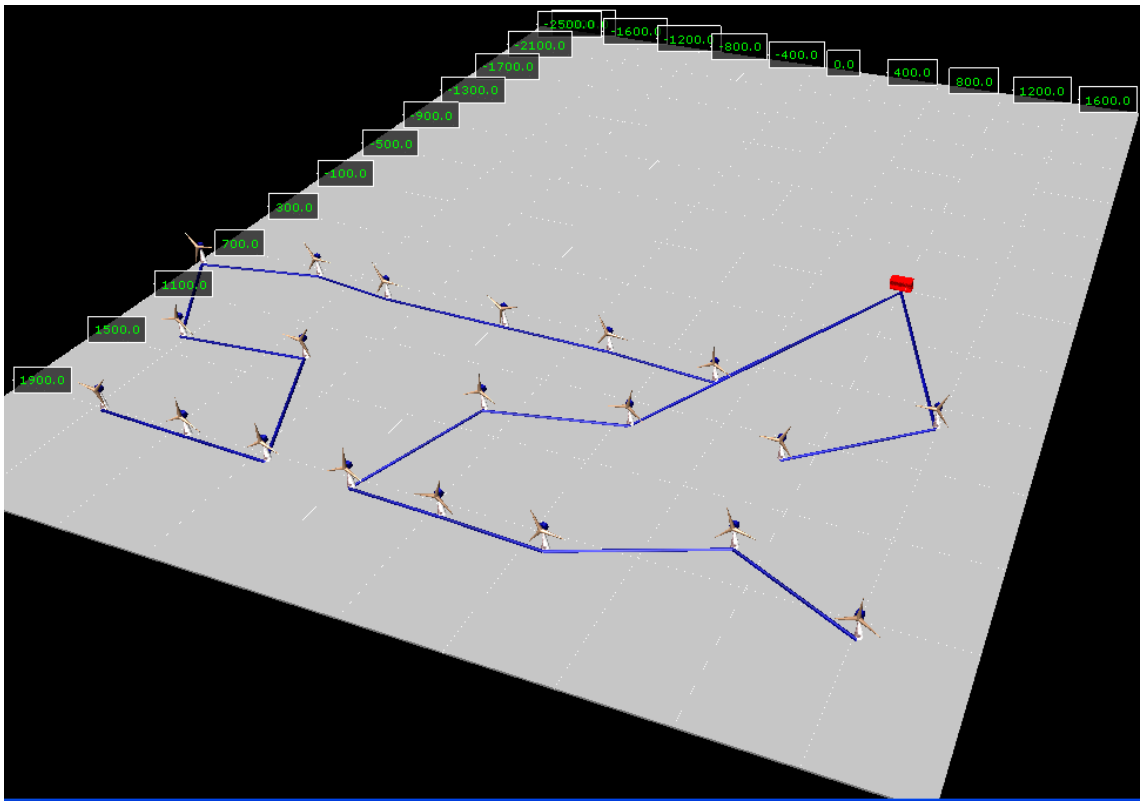


FIGURE 5.4.2.: Topologie d'une ferme de 20 éoliennes obtenue par l'algorithme génétique

Coût_energie_nd (€)	Coûts des pertes (€)	Coûts des Câbles (€)	Coûts totaux (€)
302 389	534 951	316 303	1 213 644

Puissance perdue (pertes Joule) (MW)	Tension max (p.u)	Tension min (p.u)	Nombre de départs	Itér
0.060	0.955	0.95	3	-

TABLE 5.5.: Résumé des *outputs* d'une solution proposée correspondant à une ferme de 20 éoliennes

Le tableau (5.7) représente le détail des *outputs* de la solution correspondant à la figure 5.4.3.

5.4.3. Solution d'une ferme de 30 éoliennes proposée par une expertise humaine

La figure (5.4.4) montre le câblage proposé par une expertise humaine pour la même ferme comprenant 30 éoliennes. Le résumé des *outputs* est présenté dans le tableau (5.8).

On remarque d'après les tableaux (5.6) et (5.8) que l'algorithme génétique réduit de 22.5% les coûts par rapport à la solution proposée par l'expertise humaine.

Le tableau (5.9) présente le détail des *outputs* de la solution correspondante à la figure (5.4.4), que l'on peut alors comparer au tableau (5.7).

5.4.4. Validation

Les solutions obtenues par, d'une part, un expert et, d'autre part, numériquement par "algorithme génétique" sont tout à fait cohérentes. Un avantage est trouvé pour la solution numérique, qui permet de réduire les coûts d'environ 22%. Ce résultat est en conséquence significatif.

Le choix porté sur les algorithmes génétiques était conditionné par la nature discrète du problème.

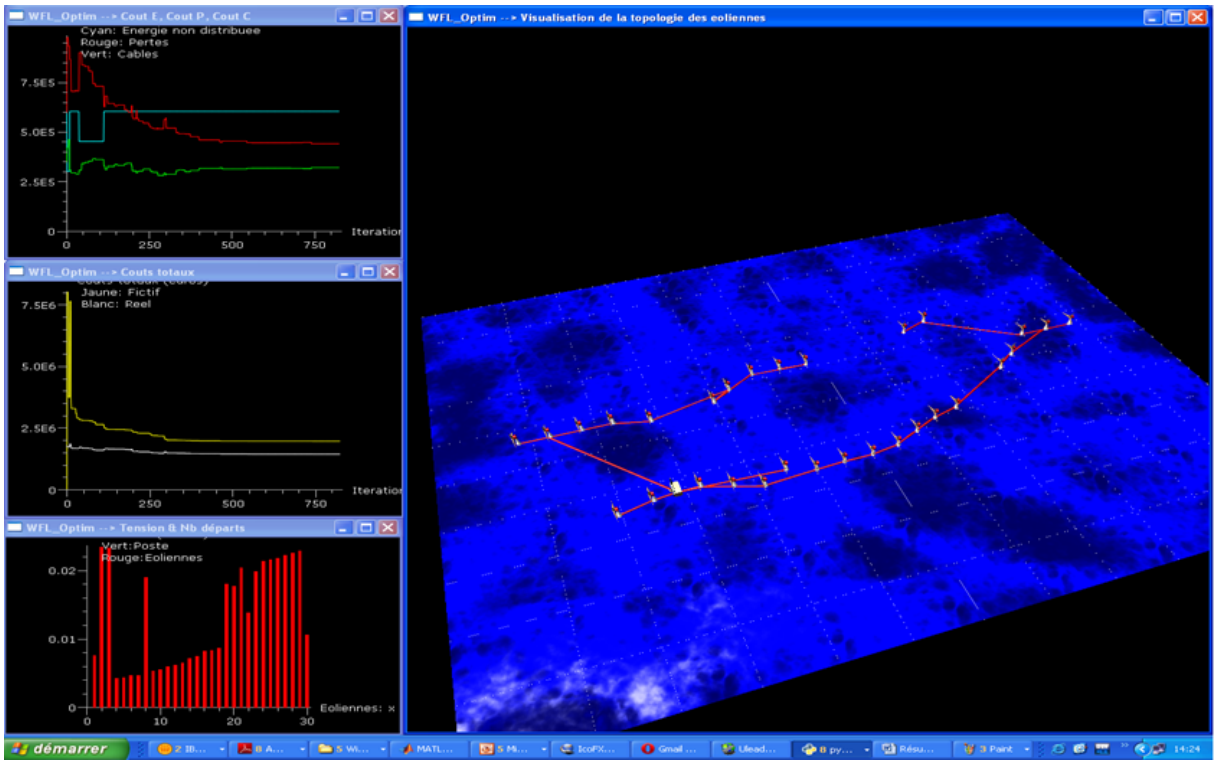


FIGURE 5.4.3.: Topologie d'une ferme de 30 éoliennes obtenue par l'algorithme génétique

Coût_energie_nd (€)	Coûts des pertes (€)	Coût 3 *clusters* (€)	Coûts des Câbles (€)	Coûts totaux (€)
453 584	494 474	60 000	294 366	1 302 425

Puissance perdue (pertes Joule) (MW)	Tension max (p.u)	Tension min (p.u)	Nombre de départs	Itér
0.056	0.954	0.95	3	978

TABLE 5.6.: Résumé des *outputs* d'une solution proposée par algorithme génétique, correspondant à une ferme de 30 éoliennes

Néanmoins, il semble naturel de vouloir leur substituer les techniques à base de population (essaim de particules, colonies de fourmis, ...).

5.5. Conclusion

Dans ce chapitre, un outil d'aide à la décision par Algorithme Génétique a été proposé. Cet outil peut être utilisé par les opérateurs de réseaux électriques afin d'optimiser la topologie interne de leurs parcs éoliens. Dans un premier temps, une modélisation - codage - efficace pour ce genre de problème a été proposée. Nous avons vu, par des exemples, que ce type de codage donne une flexibilité importante quant à la gestion des contraintes et l'implémentation de notre Algorithme Génétique. Dans un deuxième temps, l'implantation d'une fonction Python réalisant un calcul *AC Power Flow* a été réalisée. Elle ne repose que sur du code Python dont les sources sont accessibles et gratuits (module Pylon). On s'affranchit donc totalement de l'environnement *MATLAB/ MATPOWER* dont l'utilisation induit des coûts qui peuvent être élevés. Bien que ceci ne soit pas mentionné dans ce chapitre, une validation de ce code par rapport aux résultats donnés par un programme similaire *MATLAB/ MATPOWER* a été conduite avec succès.

N° de grappe	Éolienne de départ	Éolienne d'arrivée	Distance m	Section mm^2
1	Cluster	1	385	240
	1	19	973	50
Totaux	-	-	1 359	-
2	Cluster	2	159	240
	2	3	290	240 c
	3	4	198	240 c
	4	5	185	240 c
	5	6	187	240
	6	7	187	240
	7	8	187	240
	8	9	189	240
	9	10	186	240
	10	11	186	240
	11	12	187	150
	12	13	187	240 c
	13	14	577	240
	14	15	190	240
	15	16	282	240
	16	18	400	240
	18	17	200	150
	17	30	894	95
Totaux	-	-	4 878	-
3	Cluster	21	893	240
	21	20	200	240
	20	22	400	240 c
	22	23	245	150
	23	24	437	240
	24	25	164	240
	25	26	227	240
	26	27	200	240
	27	28	192	95
	28	29	820	95
Totaux	-	-	3 781	-

TABLE 5.7.: Détails des *outputs* d'une solution proposée par algorithme génétique, correspondant à une ferme de 30 éoliennes

Ainsi, développé entièrement à partir de la plateforme Python, l'exécutable à disposition (version 1.0) autorise la détermination de solutions pertinentes quant au raccordement interne d'un parc éolien. Une seconde version permettra de réduire le temps de calcul nécessaire à l'obtention d'une solution optimale.

Les éléments suivants sont à prendre en considération :

1. Sur la base des choix effectués, le nombre d'itérations et la taille de la population peuvent être paramétrés.

2. L'initialisation de l'algorithme génétique qui est faite permet de réduire le champ d'exploration de manière significative. L'introduction de considérations physiques devrait permettre une accélération de la convergence.

3. L'interface graphique développée ne constitue en rien un absolu. Elle peut être remaniée à souhait, en fonction des désidérata de l'utilisateur.

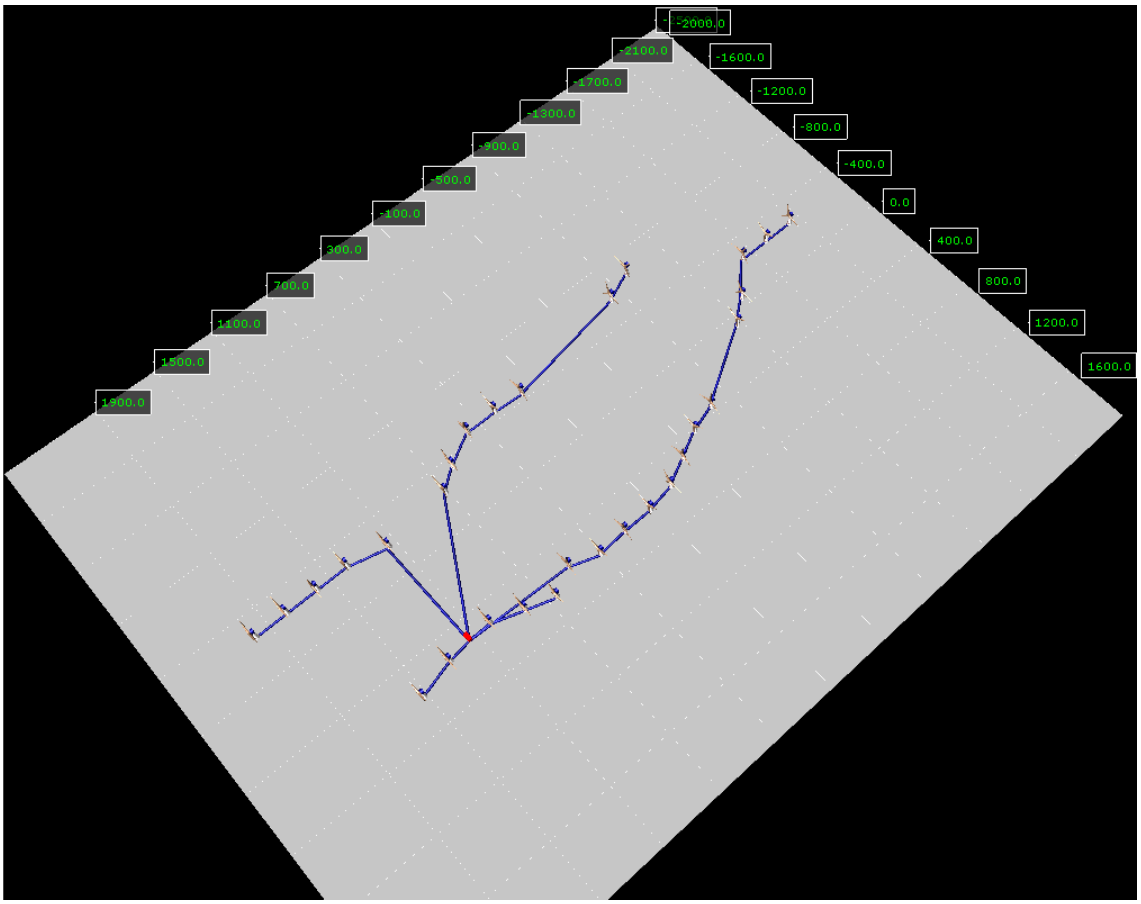


FIGURE 5.4.4.: Topologie d'une ferme de 30 éoliennes proposée par une expertise humaine

La conception de l'algorithme génétique (sélection, croisement, mutation) peut faire l'objet d'une réflexion plus approfondie. Une pré-étude sera alors nécessaire afin de dégager des voies pérennes, quant à l'optimalité de la solution.

Coût_energie_nd (€)	Coûts des pertes (€)	Coût 5 *clusters* (€)	Coûts des Câbles (€)	Coûts totaux (€)
755 973	651 615	100 000	173 150	1 680 739

Puissance perdue (pertes Joule) (MW)	Tension max (p.u)	Tension min (p.u)	Nombre de départs	Itér
0.0737	0.955	0.95	5	-

TABLE 5.8.: Résumé des *outputs* d'une solution proposée par une expertise humaine, correspondant à une ferme de 30 éoliennes

N° de grappe	Éolienne de départ	Éolienne d'arrivée	Distance m	Section mm^2
1	Cluster	2	159	50
	2	1	226	50
Totaux	-	-	385	-
2	Cluster	3	131	50
	3	4	198	50
	4	5	185	50
Totaux	-	-	514	-
3	Cluster	6	658	240
	6	7	188	240
	7	8	188	150
	8	9	189	150
	9	10	187	150
	10	11	187	95
	11	12	187	95
	12	13	187	95
	13	14	577	95
	14	15	190	50
	15	16	282	50
	16	17	200	50
	17	18	200	50
Totaux	-	-	3 420	-
4	Cluster	23	714	50
	23	22	245	50
	22	21	200	50
	21	20	200	50
	20	19	200	50
Totaux	-	-	1 559	-
5	Cluster	24	873	95
	24	25	164	50
	25	26	227	50
	26	27	200	50
	27	28	192	50
	28	29	820	50
	29	30	215	50
Totaux	-	-	2 693	-

TABLE 5.9.: Détails des *outputs* d'une solution proposée par une expertise humaine, correspondant à une ferme de 30 éoliennes

CONCLUSION GÉNÉRALE

Dans un cadre prédéfini, élargi par nécessité, nous avons souhaité tout au long de ce travail vérifier la pertinence d'hypothèses scientifiques. Celles ci ont surtout été guidées par des impératifs techniques quant au positionnement d'Alstom Grid et ont pu être étayées par le support académique dont nous avons bénéficié.

Nous pensons que chaque chapitre a pu remplir ses objectifs, avec des impacts différents, a permis de se positionner et d'envisager des perspectives :

- Chapitre 1

  L'analyse bibliographique effectuée sur les thématiques développées permet de quantifier l'intérêt des sujets, leur pérennité et leur efficacité :

  - le nombre de travaux publiés dans le domaine des métaheuristiques atteste de l'activité et du dynamisme de la discipline. En ce qui concerne les techniques d'intensification et de diversification, ou, en d'autres termes, les techniques de recherche locale et de recherche globale, l'$\alpha-$stabilité a été encore peu publiée. Sous-jacent, le cadre de l'invariance d'échelle est un thème encore émergent.
  - la modélisation des systèmes complexes reste une thématique en plein essor et une première étape a été probablement atteinte sur la description de leurs propriétés topologiques. L'aspect dynamique est en développement. Les considérations sur la signification de dimensions fractales des graphes doit être renforcée.
  - Le problème du positionnement optimal de FACTS s'est bien développé pendant le travail de thèse.

- Chapitre 2
  - Nous pensons que l'usage des lois $\alpha-$stables présente un intérêt dans l'optimisation des recherches locale / globale des informatrices. Les deux techniques (recherche locale / globale et recherche locale) que nous avons proposées doivent être complétées et les premiers résultats obtenus sont tout à fait encourageants.
  - Nous pensons qu'il faut cadrer les perspectives dans la thématique de l'invariance d'échelle. Les travaux initiés par L. Nottale [Nott11] sur le cadre relativiste de l'échelle sont un point de départ.

- L'estimateur non paramétrique de lois $\alpha-$stables proposé est original par l'utilisation des métaheuristiques et par sa capacité à estimer les 4 paramètres sans restriction. Il est par ailleurs facilement parallélisable ce qui, pour les applications, est une étape importante à considérer. Bien que les résultats exposés dans ce chapitre soient très encourageants, l'estimateur présenté dans cette thèse peut être largement amélioré, en exploitant les propriétés des lois α-stables, ou en appliquant d'autres métaheuristiques par exemple. En effet, quoique l'objectif du travail était de donner des résultats pérennes quant à l'estimation des paramètres des lois α-stables, nous voulons, avant tout, disséminer un nouveau point de vue dans le domaine des statistiques : l'introduction des métaheuristiques, entre autre.

- Chapitre 3
  - Ce chapitre a été dédié à la mise en œuvre d'une solution de placement et dimensionnement de FACTS dans un réseau électrique. Rappelons le, le choix s'est porté sur les algorithmes d'essaim de particules avec l'intégration d'une stratégie de recherche basée sur l'$\alpha-$stabilité. Les résultats obtenus par l'algorithme α-SLPSO sont cohérents avec ceux obtenus par d'autres variantes d'optimisation par essaim de particules. Ce dernier peut être amélioré en appliquant d'autres topologies de l'optimisation par essaim de particules et surtout, en proposants une étude, plus rigoureuse, des paramètres de l'algorithme notamment, du paramètre d'échelle γ des lois α-stables, afin d'améliorer la phase d'intensification. Néanmoins, les objectifs ont été atteints tant en approche mono-objectif qu'en approche multi-objectif. L'étape prochaine résidera dans la mise à disposition de ce module dans une démarche logicielle. Par ailleurs, la recherche locale α-SLS proposée dans cette thèse semble être, en vue des résultats obtenus, très prometteuse. Par conséquent, il est intéressant de mieux travailler le concept ce cette dernière et de la valider sur des cas d'études réels.
  - Une étape devra être franchie par la prise en compte du comportement du réseau électrique en régime transitoire.

- Chapitre 4
  - L'étude et la caractérisation topologique ont été développées sur le cas du réseau électrique colombien. Il s'agit d'un point important, car nous avons pu avoir accès aux données d'un opérateur de réseau. Ce fait, assez rare pour le signaler, ouvre des perspectives tout à fait intéressantes pour la validation d'outils d'analyse, mais aussi et surtout, pour appréhender les phénomènes tels que les opérateurs les considèrent, dans leur abstraction et leur simplification.
  - Les premières estimations de dimension fractale de graphes sont pertinentes, mais nous devons encore progresser sur la compréhension physique. Là encore, des extensions sont envisageables et se rapportent à la définition de structures invariantes, qui pourraient caractériser les réseaux électriques et leur dynamique.

- Chapitre 5

  Il s'agissait, dans ce dernier chapitre, de mettre en œuvre, en un temps limité (2 mois) un outil d'aide à la décision relatif à l'optimisation de la topologie interne d'un parc éolien. L'exercice devait aboutir à une première version d'un prototype logiciel. Nous pensons avoir mené à terme cette mission à la fois sur les plans scientifique et opérationnel. Si la mise en œuvre d'un algorithme génétique n'est pas en soi originale, son utilisation a néanmoins permis de réduire les coûts de 20 % sur un exemple type de ferme d'éoliennes.

ANNEXE A

APPLICATION DANS LE CAS DU MULTI-OBJECTIF

A.1. Fonctions Tests (*benchmarks*)

Dans cette section, on présentera les fonctions *benchmarks* utilisées dans l'expérimentation des algorithmes. Les principales fonctions tests sont les suivantes [Suga05] :

Fonction Sphère

$$f_1(x) = \sum_{i=1}^{n} x_i^2. \tag{A.1.1}$$

Fonction de *Rosenbrock*

$$f_2(x) = \sum_{i=1}^{n} (100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2). \tag{A.1.2}$$

Fonction de *Akley*

$$f_3(x) = -20exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - exp\left(\frac{1}{n}\sum_{i=1}^{n} cos(2\pi x_i)\right) + 20 + e. \tag{A.1.3}$$

Fonction de *Rastringin*

$$f_4(x) = \sum_{i=1}^{n} (x_i^2 - 10cos(2\pi x_i) + 10). \tag{A.1.4}$$

Fonction de *Weierstrass*

$$f_5(x) = \sum_{i=1}^{n} \left(\sum_{k=0}^{k_{max}} \left[a^k cos(2\pi b^k (x_i + 0.5)) \right] \right) - D \sum_{k=0}^{k_{max}} \left[a^k cos(2\pi b^k .0.5) \right] \tag{A.1.5}$$

$$a = 0.5, \;\; b = 3, \;\; k_{max} = 20.$$

Fonction de *Griewank*

$$f_6(x) = \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} cos\left(\frac{x_i}{\sqrt{i}}\right) + 1. \tag{A.1.6}$$

La tableau 1 représente les paramètres des fonctions *de benchmarks*.

f	Nom	Optimum global x^*	$f(x^*)$	Espace de recherche
f_1	Sphere function	(0, 0, ..., 0)	0	$[-100, 100]^N$
f_2	Quadric function	(0, 0, ..., 0)	0	$[-100, 100]^N$
f_3	Rosenbrock's function	(1, 1, ..., 1)	0	$[-2.048, 2.048]^N$
f_4	Ackley's function	(0, 0, ..., 0)	0	$[-32.768, 32.768]^N$
f_5	Rastrigin's function	(0, 0, ..., 0)	0	$[-5.12, 5.12]^N$
f_6	Weierstrass's function	(0, 0, ..., 0)	0	$[-0.5, 0.5]^N$
f_8		(0, 0, ..., 0)	0	$[-600, 600]$

Table 1. Fonctions *de benchmarks*.

A.2. Application dans le cas du multi-objectif

L'introduction d'un critère économique sur le coût de la puissance réactive installée nécessite une estimation de ce coût. Celle-ci sera basée sur les relations A.2.1. La figure A.2.1 présente l'évolution de ce coût en fonction de la puissance installée, S, $\tilde{S}$ *et* $\hat{S}$, pour différents types de FACTS.

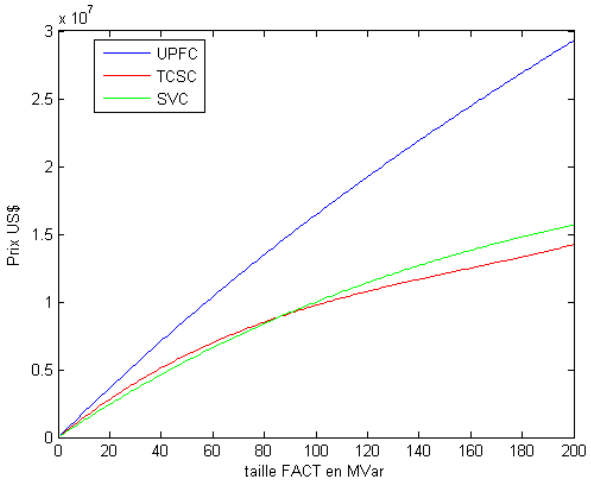


FIGURE A.2.1.: Coût des FACTS en fonction de leurs dimensions et de leurs types.

$$Prix_{SVC} = Prix_{STATCOM} = (0.0003.S^2 - 0.3051.S + 127.38) \times S.10^3$$

$$prix_{TCSC} = (0.0015.\tilde{S}^2 - 0.7130.\tilde{S} + 153.75) \times \tilde{S}.10^3 \tag{A.2.1}$$

$$prix_{UPFC} = (0.0003.\hat{S}^2 - 0.2691.\hat{S} + 188.22) \times \hat{S}.10^3$$

avec respectivement S_i, $\hat{S}_i$ *et* $\tilde{S}_i$ les puissances installées du STATCOM i ou SVC i, du TCSC i et de l'UPFC i.

Les coûts engendrés par le placement et dimensionnement de chaque type de FACTS sera donc (par interpolation de la figure A.2.1) :

$$\begin{aligned} P_1 &= \sum_{i=1}^{N_1}(0.0003.S_i^2 - 0.3051.S_i + 127.38) \times S_i.10^3 \\ P_2 &= \sum_{i=1}^{N_2}(0.0015.\tilde{S}_i^2 - 0.7130.\tilde{S}_i + 153.75) \times \tilde{S}_i.10^3 \\ P_3 &= \sum_{i=1}^{N_3}(0.0003.\hat{S}_i{}^2 - 0.2691.\hat{S}_i + 188.22) \times \hat{S}_i.10^3 \end{aligned} \tag{A.2.2}$$

avec respectivement $N_i, N_2\, et\, N_3$ le nombre total de STATCOM et SVC, TCSC et UPFC installés dans le réseau.

Finalement, le critère de coût à optimiser sera :

$$J_6 = P_1 + P_2 + P_3 \tag{A.2.3}$$

En guise de tests et de comparaison avec certaines méthodes d'optimisation par essaim de particules multi-objectif (notamment l'approche par $\epsilon - contrainte$, l'algorithme H-GA-PSO et H- PSO définis dans le chapitre I), on se fixe deux critères à optimiser, à savoir :

- la minimisation des différences de tension nodales (J_3, équation (3.2.10)),
- la minimisation du coût total des FACTS installés dans le réseau (J_6, équation (A.2.3)).

Sauf indication contraire, les paramètres de réglage initiaux communs des trois méthodes sont fixés comme suit :

- L'algorithme validé : α-SLPSO (même paramètres que dans le cas mono objectif).
- Taille de l'essaim : $Nb_{particule} = 50$,
- Nombre d'itérations : 1000,
- Choix du réseau : IEEE 57 nœuds.

Approche par $\epsilon - contrainte$

Ici, la dimension de $\overrightarrow{\epsilon}$ est de 2 (i.e. deux objectifs). La figure A.2.2-a montre la région de Pareto (en bleu) trouvée en faisant varier ϵ_2 (correspondant à l'objectif coût) de $0.20 * 10^6$ à $5 * 10^7$ (chapitre I).

L'algorithme $\alpha-$SLPSO est exécuté plusieurs fois, ici 400 fois, (variation de ϵ_2) pour retrouver la région de Pareto, mais en termes de recherche de cette dernière, cela correspond à une seule exécution.

L'algorithme a trouvé 72 solutions sur le front de Pareto, soit $7.2{*}10^{-4}$ solutions/itération, ce qui n'est pas du tout satisfaisant, compte tenu du temps de calcul très élevé. En revanche, on remarque une bonne diversité des solutions, résultats attendus, vu que l'on a déjà validé l'algorithme $\alpha-$SLPSO dans le cas mono objectif.

H-GA-PSO (*Hybrid Genetic Algorithm PSO*)

La figure A.2.2-b montre la région de Pareto obtenue correspondant à une simulation avec comme paramètre la taille de l'archive (voisins) = 3.

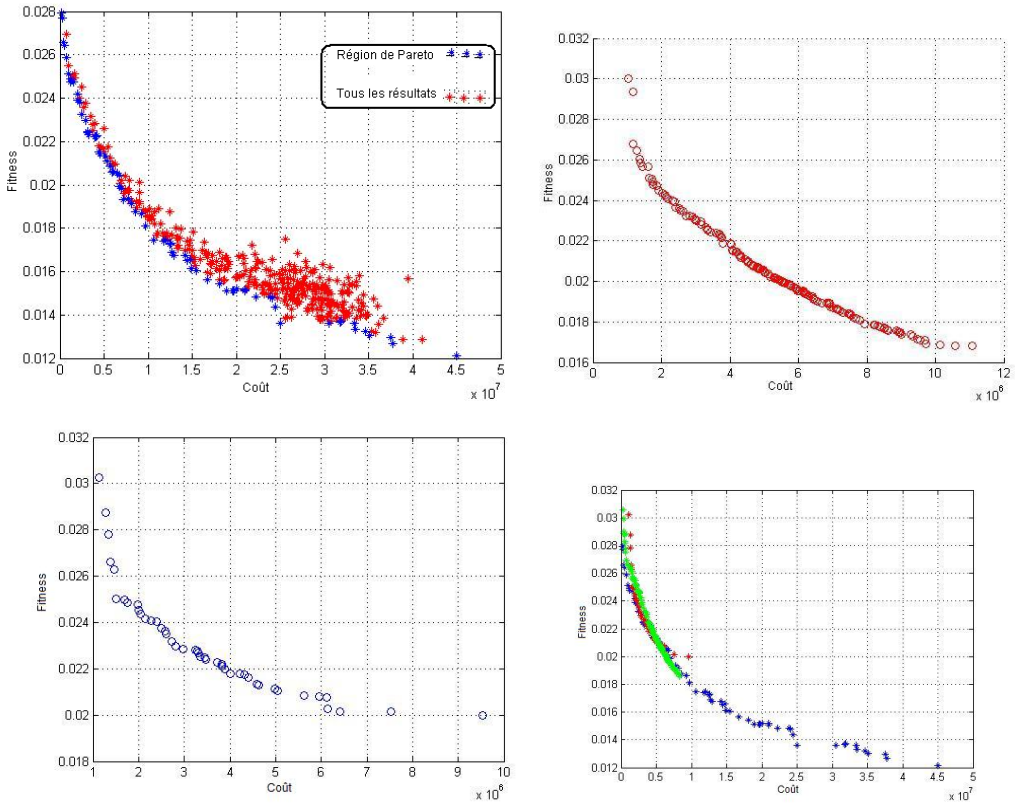


FIGURE A.2.2.: a)Région de Pareto, approche ϵ-contrainte, 35 particules 500 itérations. b) Région de Pareto, approche H-GA-PSO, 50 particules 1000 itérations. c) Région de Pareto, approche H-PSO (50, particules 1000 itérations). d) Région de Pareto, approche H-PSO (50 particules 1000 itérations) , approche H-GA-PSO (50 particules, 1000 itérations) et approche ϵ-contrainte (35 particules, 500 itérations).

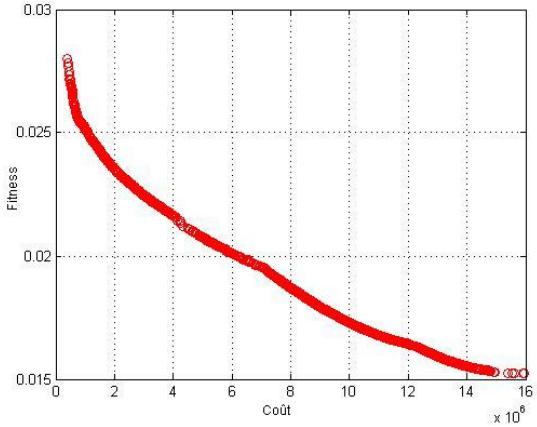


FIGURE A.2.3.: Région de Pareto, approche H-GA-PSO (50 particules, 90000 itérations).

L'algorithme nous propose 161 solutions du front de Pareto, soit 0.161 solutions/itération, ce qui est nettement mieux que l'approche $\epsilon-$contrainte pour un temps de calcul près de 140 fois moins élevé. En outre, les solutions proposées sont, dans la plupart des cas, meilleures, ce qu'illustre la figure (A.2.2-d). En revanche, on remarque une diversité des solutions moins bonne que dans l'approche ϵ- contrainte.

Approche H-PSO (*Homogeneous PSO*)

La figure A.2.2-c montre la région de Pareto obtenue en exécutant H- PSO avec comme paramètre, le nombre de voisin (l'archive) = 3 .

L'algorithme nous propose 45 solutions du front de Pareto, soit 0.045 solutions/itération. Ce qui est déjà beaucoup plus par rapport à l'approche ϵ-contrainte (si l'on tient compte du facteur temps de calcul). En revanche, on remarque que cette méthode présente une faiblesse dans la diversification (i.e. propose seulement une partie de la région de Pareto).

On remarque, d'après la figure A.2.2-d, que la qualité des solutions des deux méthodes H-PSO et H-GA-PSO est pratiquement identique.

D'après les figures A.2.2-b et A.2.2-d et les résultats présentés, la méthode H-GA-PSO semble être la meilleure approche pour notre problème d'optimisation, compte tenu du temps de calcul demandé et du nombre de solutions q proposé. Néanmoins, sa difficulté à nous proposer des solutions nouvelles (i.e. garantir la diversité des solutions) reste une faiblesse. En laissant tourner l'algorithme un plus longtemps, il propose de plus en plus de nouvelles solutions, ce qu'illustre la figure A.2.3. L'algorithme propose 1590 solutions du front de Pareto, soit 0.0159 solutions/itération, ce qui prouve que l'algorithme possède la propriété d'ergodicité (capacité à proposer toute la région de Pareto).

La figure A.2.4 indique sur un même graphe la solution obtenue par placement et dimensionnement de FACTS (cf 3.4.1) et le front de Pareto obtenu dans le cas du placement et dimensionnement de plusieurs FACTS de type différent (3.4.2). L'examen de cette figure met en exergue l'intérêt d'une approche multi-objectif. En effet, pour une performance "physique" (critère J_3) identique, le coût proposé par l'approche multi-objectif est sensiblement moins élevé (3 millions \$, à comparer avec 8 millions \$, pour 2 FACTS).

...

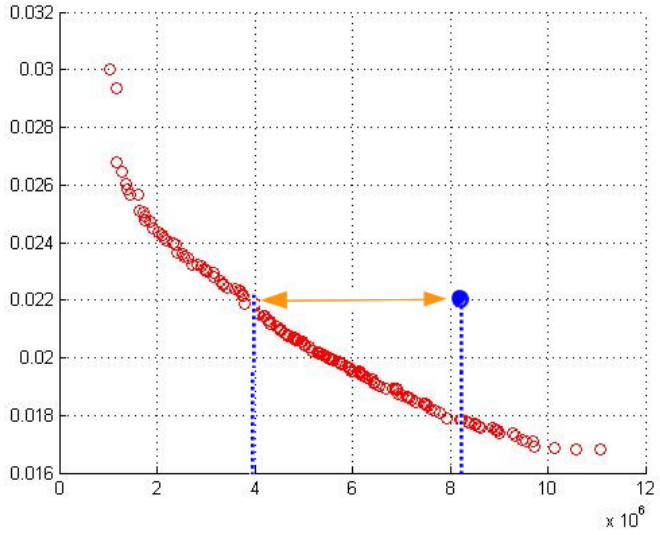


FIGURE A.2.4.: Comparaison des solutions à 2 FACTS et plusieurs FACTS.

RÉFÉRENCES BIBLIOGRAPHIQUES

[Albe02] R. Albert and A-L. Barabási. Statistical mechanics of complex networks. *Rev. Mod. Phys.*, 74 :47–97, Jan 2002. 22, 116

[Albe04] R. Albert, I. Albert, and G.L. Nakarado. Structural vulnerability of the north american power grid. *Phys. Rev. E*, 69 :025103, Feb 2004. 100, 115

[Areva] Blackout Team. Blackout dans les réseaux électriques : Analyse de la dynamique (document interne AREVA), Jul 2006. 100

[Arevab] Blackout Team. Power flow model (document interne AREVA), Jul 2006. 59

[Avri12] M. Avriel. *Nonlinear Programming : Analysis and Methods.* Dover Books on Computer Science. Dover Publications, Englewood Cliffs, NJ, 2012. 7

[Azen92] R. Azencott. *Simulated annealing : parallelization techniques.* Wiley-Interscience series in discrete mathematics. Wiley, New York, 1992. 9

[Bask04] S. Baskar and P.N. Suganthan. A novel concurrent particle swarm optimization. In *Evolutionary Computation, 2004. CEC2004. Congress on*, volume 1, pages 792–796, Portland, OR, USA, 2004. IEEE. 19

[Bend78] E.A. Bender and E.R. Canfield. The asymptotic number of labeled graphs with given degree sequences. *Journal of Combinatorial Theory, Series A*, 24(3) :296–307, May 1978. 22

[Beni05] O. Bénichou, M. Coppey, M. Moreau, P.H. Suet, and R. Voituriez. Optimal search strategies for hidden targets. *Physical Review Letters*, 94(19) :198101, May 2005. 72

[Berg02] F. Van Den Bergh. *An Analysis of Particle Swarm Optimizers.* PhD thesis, Pretoria, South Africa, 2002. AAI0804353. 18

[Berg04] F. Van den Bergh and A.P. Engelbrecht. A cooperative approach to particle swarm optimization. *Evolutionary Computation, IEEE Transactions on*, 8(3) :225–239, 2004. 19

[Berg73] C. Berge. *Graphes et hypergraphes.* Dunod Université. 1973. 101

[Bert95] D. Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena Scientific Belmont, 1995. 7

[Beye01] H.G. Beyer. *The Theory of Evolution Strategies.* Springer, Berlin, Heidelberg, New York, New York, NY, USA, 2001. 11

[Blel90] G.E. Blelloch. Prefix sums and their applications. Technical Report CMU-CS-90-190, School of Computer Science, Carnegie Mellon University, nov 1990. 52

[Bocc06] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, and D-U. Hwang. Complex networks : Structure and dynamics. *Phys. Rep.*, 424 :175–308, Feb 2006. 23, 27, 100, 101, 104, 115, 116

[Bran02] J. Branke. *Evolutionary Optimization in Dynamic Environments.* Kluwer Academic Publishers, Norwell, MA, USA, 2001. 11

[Brau03] L.A. Braunstein, S.V. Buldyrev, R. Cohen, S. Havlin, and H.E. Stanley. Optimal paths in disordered complex networks. *Physical Review Letters*, 91(16) :168701, Aug 2003. 25, 26

[Brot83] K.M. Brothers, W.H. DuMouchel, and A.S. Paulson. ractiles of the stable laws. Technical report, Rensselaer Polytechnic Institute, Troy, NY, USA, 1983. 32

[Canc03] R.F. i Cancho and R. Solé. Optimization in complex networks. In *Statistical mechanics of complex networks*, pages 114–126. Springer, 2003. 26

[Carl99] J.M. Carlson and J. Doyle. Highly optimized tolerance : A mechanism for power laws in designed systems. *Physical Review E*, 60(2) :1412–27, Aug 1999. 21

[Carr02] B.A. Carreras, V.E. Lynch, I. Dobson, and D.E. Newman. Critical points and transitions in an electric power transmission model for cascading failure blackouts | Browse - Chaos. *Chaos*, 12 :985–994, Dec 2002. 27, 100, 109

[Carv09] R. Carvalho, L. Buzna, F. Bono, E. Gutiérrez, W. Just, and D. Arrowsmith. Robustness of trans-european gas networks. *Phys. Rev. E*, 80 :016106, Jul 2009. 115

[Cham76] J.M. Chambers, C.L. Mallows, and B.W. Stuck. A method for simulating stable random variables. *Journal of the American Statistical Association*, 71(354) :340–344, Jun 1976. 3, 37, 73, 74

[Cler02] M. Clerc and J. Kennedy. The particle swarm-explosion, stability, and convergence in a multidimensional complex space. *Evolutionary Computation, IEEE Transactions on*, 6(1) :58–73, Feb 2002. 18

[Cler04] M. Clerc. Discrete particle swarm optimization, illustrated by the traveling salesman problem. In *New Optimization Techniques in Engineering*, volume 141 of *Studies in Fuzziness and Soft Computing*, pages 219–239. Springer Berlin Heidelberg, 2004. 16

[Cler05] M. Clerc. *L'optimisation par essaims particulaires.* Hermes Science Publications, London, UK, 2005. 14, 16

[Cler07] M. Clerc. Binary particle swarm optimisers : toolbox, derivations, and mathematical insights, Jan 2007. 16, 72

[Cohe00] R. Cohen, K. Erez, D. ben Avraham, and S. Havlin. Resilience of the internet to random breakdowns. *Phys. Rev. Lett.*, 85 :4626–4628, Nov 2000. 23, 27

[Cohe01] R. Cohen, K. Erez, D.B. Avraham, and S. Havlin. Breakdown of the Internet under Intentional Attack. *Physical Review Letters*, 86 :3682–3685, Apr 2001. 23, 27

[Cohe02] R. Cohen, D. ben Avraham, and S. Havlin. Percolation critical exponents in scale-free networks. *Phys. Rev. E*, 66 :036113, Sep 2002. 23, 24

[Cohe03] R. Cohen and S. Havlin. Scale-free networks are ultrasmall. *Phys. Rev. Lett.*, 90 :058701, Feb 2003. 24

[Coll02] Y. Collette and P. Siarry. *Optimisation multiobjectif.* Algorithmes (Paris). Eyrolles, 2002. 11

[Colo91] A. Colorni, M. Dorigo, V. Maniezzo, et al. Distributed optimization by ant colonies. In *Proceedings of the first European conference on artificial life*, volume 142, pages 134–142, Paris, France, 1991. MIT Press. 19

[Corr06] E.S. Correa, A.A. Freitas, and C.G. Johnson. A new discrete particle swarm algorithm applied to attribute selection in a bioinformatics data set. In *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, GECCO '06, pages 35–42, New York, NY, USA, 2006. ACM. 14, 66

[Cost07] L.F. Costa, F.A. Rodrigues, G. Travieso, and P.R.V. Boas. Characterization of complex networks : A survey of measurements. *Advances in Physics*, 56(1) :167–242, Feb 2007. 109

[Cour94] J.P. Courat, G. Raynaud, I. Mrad, and P. Siarry. Electronic component model minimization based on log simulated annealing. *Circuits and Systems I : Fundamental Theory and Applications, IEEE Transactions on*, 41(12) :790–795, 1994. 9

[Cruc04] P. Crucitti, V. Latora, and M. Marchiori. Model for cascading failures in complex networks. *Phys. Rev. E*, 69 :045104, Apr 2004. 27

[Cruc05] P. Crucitti, V. Latora, and M. Marchiori. Locating critical lines in high-voltage electrical power grids. *Fluctuation and Noise Letters*, 5(02) :L201–L208, 2005. 27, 100, 115

[DasS08] S. Das, A. Abraham, and A. Konar. Particle swarm optimization and differential evolution algorithms : Technical analysis, applications and hybridization perspectives. In Y. Liu, A. Sun, H. Loh, W. Lu, and E-P. Lim, editors, *Advances of Computational Intelligence in Industrial Systems*, volume 116 of *Studies in Computational Intelligence*, pages 1–38. Springer Berlin Heidelberg, 2008. 72

[Devr86] L. Devroye. *Non-uniform random variate generation.* Springer-Verlag, 1986. 36

[Dobs01a] I. Dobson, B.A. Carreras, V.E. Lynch, and D.E. Newman. An initial model fo complex dynamics in electric power system blackouts. In *System Sciences, 2001. Proceedings of the 34th Annual Hawaii International Conference on*, pages 710–718, Jan 2001. 3, 21

[Dobs01b] B.A Carreras, V.E Lynch, M.L. Sachtjen, I. Dobson, and D.E Newman. Modeling blackout dynamics in power transmission networks with simple structure. In *hicss*, volume 1, page 2018, 2001. 21

[Dori96] M. Dorigo, V. Maniezzo, and A. Colorni. Ant system : optimization by a colony of cooperating agents. *Systems, Man, and Cybernetics, Part B : Cybernetics, IEEE Transactions on*, 26(1) :29–41, 1996. 19

[Doro03] S.N. Dorogovtsev and J.F.F. Mendes. *Evolution of Networks : From Biological Nets to the Internet and WWW (Physics).* Oxford University Press, Inc., New York, NY, USA, 2003. 23, 104

[Doro08] S.N. Dorogovtsev, A.V. Goltsev, and J.F.F. Mendes. Critical phenomena in complex networks. *Rev. Mod. Phys.*, 80 :1275–1335, Oct 2008. 23, 27, 104, 115

[Doyl00] J. Doyle and J.M. Carlson. Power laws, highly optimized tolerance, and generalized source coding. *Physical Review Letters*, 84(24) :5656–9, Jun 2000. 21

[Dumo71] V.J. Duvanenko. *Parallel In-Place N-bit-Radix Sort.* PhD thesis, New Haven, CT, 1971. 36

[Duva10] W. Feller. An introduction to probability theory and its applications. Vol. II. Dr. Dobb's Journal, Aug 2010. 52

[Eber01] R.C. Eberhart, Y. Shi, and J. Kennedy. *Swarm Intelligence (The Morgan Kaufmann Series in Evolutionary Computation).* Morgan Kaufmann, 1 edition, April 2001. 17, 33

[Efte04] Ali Eftekhari. Fractal dimension of electrochemical reactions. *Journal of the Electrochemical Society*, 151(9) :E291–E296, Aug 2004. 108

[ElDo12a] A. El Dor, M. Clerc, and P. Siarry. A multi-swarm pso using charged particles in a partitioned search space for continuous optimization. *Comput. Optim. Appl.*, 53(1) :271–295, Sep 2012. 19, 78

[ElDo12b] A. El Dor, M. Clerc, and P. Siarry. Hybridization of differential evolution and particle swarm optimization in a new algorithm : Depso-2s. In *Proceedings of the 2012 International Conference on Swarm and Evolutionary Computation*, SIDE'12, pages 57–65, Berlin, Heidelberg, 2012. Springer-Verlag. 19, 78

[Elab09] M. El-Abd. Preventing premature convergence in a pso and eda hybrid. In *Evolutionary Computation, 2009. CEC'09. IEEE Congress on*, pages 3060–3066. IEEE, May 2009. 71

[Elgh01] E-G. Talbi. Méthodes d'optimisation avancées, Apr 2001. 10

[Erdo59] P. Erdõs and A. Rényi. On random graphs i. *Publ. Math. Debrecen*, 6 :290–297, 1959. 21

[Fama65] E.F. Fama. The Behavior of Stock-Market Prices. *The Journal of Business*, 38(1) :34–105, Jan 1965. 32

[Fama68] E.F. Fama and R. Roll. Some properties of symmetric stable distributions. *J. Am. Stat. Assoc.*, 63(323) :817–836, Sep 1968. 36

[Fama71] E.F. Fama and R. Roll. Parameter estimates for symmetric stable distributions. *Journal of the American Statistical Association*, 66(334) :331–338, Jun 1971. 32

[Fell66] W. Feller. An introduction to probability theory and its applications. Vol. II. New York-London-Sydney : John Wiley and Sons, Inc. XVIII, 626 p. (1966)., 1966. 36

[Feue81] A. Feuerverger and P. McDonnough. On efficient inference in symmetric stable laws and processes. *Canadian Journal of Statistics*, 18(2) :109–122, 1981. 32

[Fiel72] B.D. Fielitz and E.W. Smith. Asymmetric stable distributions of stock price changes. *Journal of the American Statistical Association*, 67(340) :813–814, Jan 1972. 32

[Foge62] L.J. Fogel. Autonomous automata. *Industrial Research*, 4(2) :14–19, 1962. 11

[Foge66] L.J. Fogel, A.J. Owens, and M.J. Walsh. *Artificial Intelligence through Simulated Evolution.* John Wiley, 1966. 10, 11

[Fra57] A.S. Fraser. Simulation of genetic systems by automatic digital computers. I. Introduction. *Australian Journal of Biological Science*, 10(2) :484–491, 1957. 11

[Free11] M. Freeman. *Digital Slr Handbook.* Ilex, 3rd edition, 2011. 52

[Gall05] L.K. Gallos, R. Cohen, P. Argyrakis, A. Bunde, and S. Havlin. Stability and topology of scale-free networks under attack and defense strategies. *Physical Review Letters*, 94(18) :188701, May 2005. 26

[Garc08] F.J.M. Garcia and J.A.M. Perez. Jumping frogs optimization : a new swarm method for discrete optimization. *Documentos de Trabajo del DEIOC*, (3), 2008. 14

[Gasb09] B. Gasbaoui and B. Allaoua. Ant colony optimization applied on combinatorial problem for optimal power flow solution. *Leonardo Journal of Sciences*, (14) :1–16, Jan 2009. 88

[Gerb03] G. St'ephane. *Métaheuristiques appliquées au placement optimal de dispositifs FACTS dans un réseau électrique.* PhD thesis, Lausanne, Suisse, 2003. 002742. 63

[Glov86] F. Glover. Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research*, 13(5) :533–549, 1986. 9, 10

[Glov93] F. Glover and E. Taillard. A user's guide to tabu search. *Annals of operations research*, 41(1) :1–28, 1993. 9

[Glov97] F. Glover and M. Laguna. *Tabu Search.* Kluwer Academic Publishers, Norwell, MA, USA, 1997. 9

[Gold87] D.E. Goldberg and J. Richardson. Genetic algorithms with sharing for multimodal function optimization. In *Proceedings of the Second International Conference on Genetic Algorithms on Genetic Algorithms and Their Application*, pages 41–49, Hillsdale, NJ, USA, 1987. L. Erlbaum Associates Inc. 11

[Hast70] W.K. Hastings. Monte carlo samping methods using markov chains and their applications. *Biometrika*, 57 :97–109, 1970. 8

[Hill86] W.D. Hillis and G.L. Steele Jr. Data parallel algorithms. *Commun. ACM*, 29(12) :1170–1183, Dec 1986. 52

[Hold73] D. Hold and E. Crow. Tables and graphs of the stable probability functions. *Journal of Rechearch of the National Bureau of Standards*, 3-4(77b) :143–198, Jul 1973. 32

[Holl75] J.H. Holland. *Adaptation in natural and artificial systems : an introductory analysis with applications to biology, control, and artificial intelligence.* University of Michigan Press, 1975. 10, 11

[Hoos04] H.H. Hoos and T. Stützle. *Stochastic Local Search : Foundations & Applications.* The Morgan Kaufmann Series in Artificial Intelligence. Elsevier Science, 2004. 8

[Hsie09] S-T. Hsieh, T-Y. Sun, C-C. Liu, and S-J. Tsai. Efficient population utilization strategy for particle swarm optimizer. *Systems, Man, and Cybernetics, Part B : Cybernetics, IEEE Transactions on*, 39(2) :444–456, Apr 2009. 78

[IEEE30] University of Washington. Power Systems Test Case Archive. http ://www.ee.washington.edu/research/pstca/. 86

[Jame94] F. James. Ranlux : A fortran implementation of the high-quality pseudorandom number generator of lüscher. *Computer Physics Communications*, 79(1) :111–114, Sep 1994. 51

[Kant75] M. Kanter. Stable densities under change of scale and total variation inequalities. *The Annals of Probability*, 3(4) :697–707, Aug 1975. 37

[Kary8] G. Karypis and V. Kumar. *METIS : A Software Package for Partitioning Unstructured Graphs, Partitioning Meshes, and Computing Fill-Reducing Orderings of Sparse Matrices*, Sep 1998. 26

[Kazu08] B. Wang, A. , Kazuyuki, and C. Luonan. Traffic jamming in disordred flow distribution networks. In *Operations Research and Its Applications : The 7th International Symposium, ISORA '08, Lijiang, China, October 31 November 3, 2008 : Proceedings*, Lecture notes in operations research, pages 465–469, Lijiang, China, 2008. World Publishing Corporation. 25

[Kenn95] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Neural Networks, 1995. Proceedings., IEEE International Conference on*, volume 4, pages 1942–1948, Perth, WA, Australia, 1995. IEEE. 13, 33

[Kenn97] J. Kennedy and R.C. Eberhart. A discrete binary version of the particle swarm algorithm. In *Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on*, volume 5, pages 4104–4108, Washington, DC, USA, 1997. IEEE Computer Society. 14

[KimY06] Yong-Hyuk Kim, Yourim Yoon, Alberto Moraglio, and Byung-Ro Moon. Geometric crossover for multiway graph partitioning. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 1217–1224. ACM, 2006. 126

[Kinn05] R. Kinney, P. Crucitti, R. Albert, and V. Latora. Modeling cascading failures in the north american power grid. *The European Physical Journal B - Condensed Matter and Complex Systems*, 46(1) :101–107, Jul 2005. 27, 100

[Kirk83] S. Kirkpatrick, C. D. Gelatt, Jr., and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220 :671–680, 1983. 8, 77

[Kogo98] S.M. Kogon and D.B. Williams. A practical guide to heavy tails. pages 311–335, 1998. 32

[Kout80] I.A. Koutrouvelis. Regression-type estimation of the parameters of stable laws. *Journal of the American Statistical Association*, 75(372) :918–928, Dec 1980. 32

[Koza89] J.R. Koza. Hierarchical genetic algorithms operating on populations of computer programs. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence - Volume 1*, IJCAI'89, pages 768–774, San Francisco, CA, USA, 1989. Morgan Kaufmann Publishers Inc. 11

[Koza90] J.R. Koza. Genetic programming : A paradigm for genetically breeding populations of computer programs to solve problems. Technical report, Stanford, CA, USA, 1990. 11

[Leit75] R.A. Leitch and A.S. Paulson. Estimation of stable law parameters : stock price behavior application. *Journal of the American Statistical Association*, 70(351a) :690–697, Sep 1975. 32

[Levy02] J. Lévy Véhel and C. Walter. *Les Marchés fractals.* Finance. Puf, 2002. 32

[Mand63] B. Mandelbrot. The variation of certain speculative prices. *The Journal of Business*, 36 :394–419, 1963. 3, 32

[Mann05] M. Manning, J.M. Carlson, and J. Doyle. Highly optimized tolerance and power laws in dense and sparse resource regimes. *Physical Review E*, 72(1) :016108, Jul 2005. 21

[Mant94] R.N. Mantegna. Fast, accurate algorithm for numerical simulation of lévy stable stochastic processes. *Phys. Rev. E*, 49 :4677–4683, May 1994. 36

[Mart94] M. Lüscher. A portable high-quality random number generator for lattice field theory simulations. *Computer physics communications*, 79(1) :100–110, Feb 1994. 51

[Mass51] F.J. Massey Jr. The kolmogorov-smirnov test for goodness of fit. *Journal of the American statistical Association*, 46(253) :68–78, Mar 1951. 39

[McCu86] J.H. McCulloch. Simple consistent estimators of stable distribution parameters. *Communications in Statistics-Simulation and Computation*, 15(4) :1109–1136, 1986. 32, 43

[Metr53] N. Metropolis, W.R. Arianna, N.R. Marshall, H.T. Augusta, and T. Edward. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21 :1087–1092, 1953. 8

[Mill56] L.H. Miller. Table of percentage points of kolmogorov statistics. *Journal of the American Statistical Association*, 51(273) :111–121, Mar 1956. 40

[Mitt01] S. Mittnik. *Stable Non-gaussian Models in Finance and Econometrics*, volume 34 of *Mathematical and computer modelling*. Pergamon Press, NewYork, NY, USA, 2001. 32

[Mora04] Alberto Moraglio and Riccardo Poli. Topological interpretation of crossover. In *Genetic and Evolutionary Computation–GECCO 2004*, pages 1377–1388. Springer, 2004. 126

[Mott02] A.E. Motter and Y-C. Lai. Cascade-based attacks on complex networks. *Phys. Rev. E*, 66 :065102, Dec 2002. 27, 115

[NVDIA] CUDA. NVIDIA OpenCL SDK Code Samples. https ://developer.nvidia.com/opencl. 52, 53

[Neld65] J.A. Nelder and R. Mead. A simplex method for function minimization. *The computer journal*, 7(4) :308–313, 1965. 7

[Newm02] M.E.J. Newman. Assortative mixing in networks. *Phys. Rev. Lett.*, 89 :208701, Oct 2002. 22, 102

[Newm03] M.E.J. Newman. The Structure and Function of Complex Networks. *SIAM Review*, 45 :167–256, Aug 2003. 22, 23, 27, 101, 104, 115

[Niki95] C.L. Nikias and M. Shao. *Signal Processing with Alpha-stable Distributions and Applications*. Wiley-Interscience, New York, NY, USA, 1995. 36

[Noce00] J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer series in operations research and financial engineering. Springer, New York, NY, USA, 2000. 7

[Nott11] L. Nottale. *Scale Relativity and Fractal Space-Time : A New Approach to Unifying Relativity and Quantum Mechanics*. Imperial College Press, 2011. 75, 113, 138

[OpenCL] The Khronos Group. OpenCL API 1.1 Quick Reference Card. http ://www.khronos.org. 52

[Osma96] I.H. Osman and J.P. Kelly. *Meta-Heuristics : Theory and Applications*. Kluwer Academic Publishers, Norwell, MA, USA, 1996. 9

[Pant92] D.B. Panton. Cumulative distribution function values for symmetric standardized stable distributions. *Communications in Statistics-Simulation and Computation*, 21(2) :485–492, 1992. 32

[Past01] R. Pastor-Satorras, A. Vázquez, and A. Vespignani. Dynamical and correlation properties of the internet. *Phys. Rev. Lett.*, 87 :258701, Nov 2001. 102

[Paul07] G. Paul, R. Cohen, S. Sreenivasan, S. Havlin, and H.E. Stanley. Graph partitioning induced phase transitions. *Phys. Rev. Lett.*, 99 :115701, Sep 2007. 26

[Paul75] A.S. Paulson, E.W. Holcomb, and R.A. Leitch. The estimation of the parameters of the stable laws. *Biometrika*, 62(1) :163–170, Apr 1975. 36

[Pres72] S.J. Press. Estimation in univariate and multivariate stable distributions. *Journal of the American Statistical Association*, 67(340) :842–846, 1972. 32

[Pres92] W.H. Press. *Numerical Recipes in C : The Art of Scientific Computing*. Number 4 in Numerical recipes in C : the art of scientific computing / William H. Press. Cambridge University Press, 1992. 36

[Pric05] K. Price, R.M. Storn, and J.A. Lampinen. *Differential Evolution : A Practical Approach to Global Optimization (Natural Computing Series)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005. 11

[Pugh06] J. Pugh and A. Martinoli. Discrete multi-valued particle swarm optimization. In *Proceedings of IEEE swarm intelligence symposium*, volume 1, pages 103–110, Indianapolis, Indiana, USA, 2006. 14

[Quin04] J. Quinqueton. Aspects socio-organisationnels dans les systÃšmes multi-agentsÂ : l'intelligence artificielle en essaim, Dec 2004. 20

[Rech65] I. Rechenberg. Cybernetic solution path of an experimental problem. Technical report, Royal Air Force Establishment, 1965. 10, 11

[Reyn94] R.G. Reynolds. An introduction to cultural algorithms. In *Proceedings of the third annual conference on evolutionary programming*, pages 131–139, San Diego, California, USA, 1994. World Scientific Press. 13

[Ribe02] P. Festa and M.G.C. Resende. GRASP : An annotated bibliography. In C.C. Ribeiro and P. Hansen, editors, *Essays and surveys in metaheuristics*, pages 325–367. Kluwer Academic Publishers, 2002. 9

[Rosa12] M. Rosas Casals, B. Corominas Murtra, et al. Assessing european power grid reliability by means of topological measures. 2012. 27

[RoyB68] B. Roy. Classement et choix en présence de points de vue multiples. *RAIRO - Operations Research - Recherche Opérationnelle*, 2(V1) :57–75, 1968. 82

[Roz09] B. Rozel. *La sécurisation des infrastructures critiques : recherche d'une méthodologie d'identification des vulnérabilités et modélisation des interdépendances*. PhD thesis, Paris, France, 2009. 00540312. 116

[Schw02] N. Schwartz, R. Cohen, D. Ben-Avraham, A-L. Barabási, and S. Havlin. Percolation in directed scale-free networks. *Physical Review E*, 66(1) :015104, Aug 2002. 24

[Shar03] B. Shargel, H. Sayama, I.R. Epstein, and Y. Bar-Yam. Optimization of robustness and connectivity in complex networks. *Phys. Rev. Lett.*, 90 :068701, Feb 2003. 27

[Siar03] J. Dreo, A. Petrowski, É.D. Taillard, and P. Siarry. *Métaheuristiques pour l'optimisation difficile*. Eyrolles (Editions), nov 2003. 33, 75

[Soil96] P. Soille and J.F. Rivest. On the Validity of Fractal Dimension Measurements in Image Analysis. *Journal of Visual Communication and Image Representation*, 7(3) :217–229, Sep 1996. 108

[Sole04] R. Solé and S. Valverde. Information Theory of Complex Networks : OnÂ Evolution and Architectural Constraints. In *Complex Networks*, volume 650 of *Lecture Notes in Physics*, pages 189–207. Springer Berlin / Heidelberg, 2004. 26

[Sole08] R.V. Solé, M. Rosas-Casals, B. Corominas-Murtra, and S. Valverde. Robustness of the european power grids under intentional attack. *Phys. Rev. E*, 77 :026102, Feb 2008. 27

[Song07] C. Song, L.K. Gallos, S. Havlin, and H.A. Makse. How to calculate the fractal dimension of a complex network : the box covering algorithm. *Journal of Statistical Mechanics : Theory and Experiment*, 2007(03) :P03006, Jan 2007. 108, 110

[Step70] M.A. Stephens. Use of the kolmogorov-smirnov, cramér-von mises and related statistics without extensive tables. *Journal of the Royal Statistical Society. Series B (Methodological)*, 32(1) :115–122, 1970. 39

[Stor97] R. Storn and K. Price. Differential evolution : A simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11(4) :341–359, 1997. 11

[Suga05] P.N. Suganthan, N. Hansen, J.J. Liang, K. Deb, Y-P Chen, A. Auger, and S. Tiwari. Problem definitions and evaluation criteria for the cec 2005 special session on real-parameter optimization. *KanGAL Report*, 2005005, 2005. 140

[Tani05] T. Tanizawa, G. Paul, R. Cohen, S. Havlin, and H.E. Stanley. Optimization of network robustness to waves of targeted and random attacks. *Phys. Rev. E*, 71 :047101, Apr 2005. 27, 108

[Thei90] J. Theiler. Estimating fractal dimension. *JOSA A*, 7(6) :1055–1073, Jun 1990. 108

[Toll03] C.R. Tolle, T.R. McJunkin, and D.J. Gorisch. Suboptimal minimum cluster volume cover-based method for measuring fractal dimension. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25(1) :32–41, 2003. 108

[Vcer85] V. Černỳ. Thermodynamical approach to the traveling salesman problem : An efficient simulation algorithm. *Journal of optimization theory and applications*, 45(1) :41–51, 1985. 8

[Voss99] S. Voss, I.H. Osman, and C. Roucairol. *Meta-heuristics : Advances and trends in local search paradigms for optimization*. Kluwer Academic Publishers, 1999. 9

[Wang06] B. Wang, H. Tang, C. Guo, and Z. Xiu. Entropy optimization of scalefree networksśobustness to random failures. *Physica A : Statistical Mechanics and its Applications*, 363(2) :591–596, May 2006. 26

[Wang08] Y. Wang, S. Xiao, G. Xiao, X. Fu, and T.H. Cheng. Robustness of complex communication networks under link attacks. In *Proceedings of the 2008 International Conference on Advanced Infocomm Technology*, ICAIT '08, pages 61 :1–61 :7, New York, NY, USA, 2008. ACM. 26

[Wass94] S. Wasserman and K. Faust. *Social Network Analysis : Methods and Applications*, volume 8 of *Structural Analysis in the Social Sciences*. Cambridge University Press, Cambridge, MA, USA, 1994. 103

[Watt02] D.J. Watts. A simple model of global cascades on random networks. *Proceedings of the National Academy of Sciences of the United States of America*, 99(9) :5766, May 2002. 27

[Watt98] D.J. Watts and S.H. Strogatz. Collective dynamics of 'small-world'networks. *nature*, 393(6684) :440–442, Apr 1998. 22

[Wenx07] W. Liu, L. Liu, D.A. Cartes, and G.K. Venayagamoorthy. Binary particle swarm optimization based defensive islanding of large scale power systems. *IJCSA*, 4(3) :69–83, 2007. 14

[Wero95] R. Weron. Performance of the estimators of stable law parameters. HSC Research Reports HSC/95/01, Hugo Steinhaus Center, Wroclaw University of Technology, 1995. 32

[Wors95] G.J. Worsdale. Tables of cumulative distribution functions for symmetric stable distributions. *Applied Statistics*, 24(1) :123–131, Nov 1975. 32

[Yuhu98] Y. Shi and R. Eberhart. A modified particle swarm optimizer. In *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, pages 69–73, May 1998. 17

[Zhan10] Z. Zhang, S. Gao, L. Chen, S. Zhou, H. Zhang, and J. Guan. Mapping koch curves into scale-free small-world networks. *Journal of Physics A : Mathematical and Theoretical*, 43(39) :395101, 2010. 108

[Zhen06] Z. Wu, L.A. Braunstein, V. Colizza, R. Cohen, S. Havlin, and H.E. Stanley. Optimal paths in complex networks with correlated weights : The worldwide airport network. *Phys. Rev. E*, 74 :056104, Nov 2006. 25

[ZhiX08] Z-X. Wu, G. Peng, W-X. Wang, S. Chan, and E.W-M. Wong. Cascading failure spreading on weighted heterogeneous networks. *of Statistical Mechanics : Theory and Experiment*, 2008 :P05013, Mar 2008. 27

[Zolo66] V.M. Zolotarev. On representation of stable laws by integrals. *Selected Translations in Mathematical Statistics and Probability*, 6(1) :84–88, 1966. 34

[schr98] A. Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, Inc., New York, NY, USA, 1998. 7