République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Ecole Nationale Polytechnique
Département d'Electrotechnique
Laboratoire de Recherche en Electrotechnique

Projet de fin d'études en vue de l'obtention du diplôme d'ingénieur d'état en électrotechnique

Présenté par
ZEKKOUR Fateh
BOUDROUAZ Yacine

Intitulé

Rétrospective sur les méthodes d'Ecoulement de Puissance Optimal

Soutenu le 15 Juin à l'ENP

Membres du jury d'examen

Président	A.Mekhaldi	Professeur à l'ENP
Rapporteur	A.Hellal	Professeur à l'ENP
Examinateurs	M.Teguar	Professeur à l'ENP
	L.Nezli	Professeur à l'ENP

ENP 2015

ملخص

يهدف هذا العمل الى دراسة السريان الأمثل للطاقة الكهربائية بطرق مختلفة حيث نسعى لتخفيض تكلفة انتاج الكهرباء و أيضا للحد من الخسائر أثناء توزيع الكهرباء على الشبكة. نحسب هذا السريان الأمثل بواسطة طرق كلاسيكية و طريقة ذكاء اصطناعي حيث نقارن ونناقش النتائج المتحصل عليها وكذا العوامل المؤثرة سلبا وإيجابا على السير الأمثل للطاقة الكهربائية .

الكلمات المفتاحية : السريان الأمثل للطاقة الكهربائية ، طريقة ذكاء اصطناعي ، طرق كلاسيكية لتخفيض تكلفة انتاج الكهرباء

Résumé

Ce travail traite le problème d'écoulement de puissance optimal en général et particulièrement le cas de la répartition économique de puissance. Pour cela, trois méthodes conventionnelles ont été présenté : méthode de Newton, méthode du gradient 2ème ordre, méthode du point intérieur ainsi qu'une méthode d'intelligence artificielle (PSO) avec adaptation. Des simulations sur différents réseaux standard ont été exécutées et les résultats comparés et analysés.

Mots clés : écoulement de puissance optimal, répartitions économique de puissance , méthodes conventionnelles , méthodes d'intelligence artificiel

Abstract

This work deals with the problem of opf in general and discuss the case of distributions of economic puissance.Pour this we solve this problem with 3-conventional methods which Newton method, 2nd order gradient method, interior point method, and artificial inteli- gence methods. Finally, we represent the different results obtained.

Key words : opf, economical dispatch, conventional methods, artificial inteligence methods.

Remerciements

En premier lieu, nous remercions le Bon Dieu Miséricordieux et Clément, qui nous a guidés dans la bonne voie des sciences et de la connaissance.

Nous tenons à remercier notre promoteur **Pr : A.Hellal** qui nous a fait profiter de son expérience et de ses précieux conseils tout au long de ce travail et dans le domaine de la recherche scientifique. Puisse ce travail vous satisfaire et témoigner notre grande reconnaissance et notre profonde estime.

Nous sommes très sensibles à l'honneur que nous fait **Pr : A.Mekhaldi** en présidant ce jury. Nous tenons à lui exprimer nos remerciements les plus sincères.

Nous exprimons notre profonde reconnaissance au **Pr : M.Teguar** et au **Pr : L.Nezli** d'avoir accepté d'être membres du jury et de juger ce modeste travail.

Nous tenons aussi à remercier notre frère et ami **Islam Ziouani**pour l'aide qu'il nous a donné, pour sa patience, son honnêteté et pour ses qualité humaines.

Nous tenons également à remercier tous les enseignants ayant contribué à notre formation depuis le tronc commun jusqu'à la dernière année de graduation.

Table des matières

Liste des Figures

Liste des Tableaux

Abréviations

EPO **E**coulement de **P**uissance **O**ptimal

MPI **M**éthode de **P**oint **I**nterieur

OEP **O**ptimisation par **E**ssaim de **P**articules

REP **R**épartition **E**conomique de **P**uissance.

Dédicaces

A mes cher parents

A mon cher frère Ahmed et mes cher sœurs Amina et Rania

A toute ma famille

A tout mes amis et collègues

Fateh

Je dédie ce travail à mes amis, mes collègues

Je tiens également à réserver une spéciale dédicace aux êtres les plus tendres à mes yeux et les plus chères à mon cœur, ma mère et mon père.

A ma fiancée, mes sœur et mes frères et à toute ma famille.

Yacine.

Introduction

La gestion d'un réseau de production, distribution ou stockage d'énergie est devenue un enjeu tant économique que technique, devant respecter des contraintes climatiques et économiques. Il apparaît donc la nécessité d'optimiser cette gestion afin de profiter du meilleur fonctionnement du réseau tout en respectant les contraintes imposées.

Le problème de répartition économique, qui est un cas des problème d'écoulement de puissance, dans son traitement à l'état statique et dynamique d'un réseau électrique, occupe dans nos jours une place déterminante dans la stratégie concurrentielle de l'entreprise, face à la libéralisation du secteur d'électricité, donc face à une concurrence acharnée, mais aussi par rapport aux nouvelles restrictions liées à l'environnement qu'il faut respecter.

Dans cette logique, un faible coût de production représente un défi pour les sociétés productrices, vu notamment l e prix des combustibles, et toutes les charges supplémentaires liées à la production , aux quelles on peut ajouter des frais comme celles liées au traitement des déchets nucléaires dans le cas de la production nucléaire.

D'un autre coté, la complexité grandissante des réseaux électriques d'aujourd'hui, vu des tailles avec des centaines de jeux de barres et des milliers de kilomètres de lignes de transmission, ainsi que des structures interconnectées, exige qu'une optimisation de la répartition optimale de puissance active générée constitue une nécessité impérative et un coût minimisé représente un objectif primordial.

Notons qu'une optimisation de cette répartition ne doit pas seulement garantir un faible coût de production mais aussi s'accompagner d'une minimisation des pertes de transport (répartition économique avec pertes), ce qui engendre un problème d'optimisation non linéaire.

Plusieurs méthodes sont proposées pour la résolution du problème d'écoulement de puissance optimal dont des méthodes conventionnelles (méthode de Newton, méthode du gradient, méthode des itérations lambda, programmation linéaire ...etc) et d'autres non conventionnelles (méthodes d'intelligence artificielle) comme les algorithmes génétiques (AG), les algorithmes évolutionnaires, optimisation par essaim de particules (OEP ou en anglais PSO) ..etc

Dans ce mémoire, nous allons appliquer un algorithme d'optimisation par essaim de particules avec apprentissage adaptatif (ALPSO) pour la résolution du problème de répartition économique. Nous appliquerons aussi trois méthodes conventionnelles (méthode de Newton, méthode du gradient 2ème ordre, et méthode du point intérieur) pour le même problème.

Notre mémoire est organisé comme suit :

— un premier chapitre intitulé Formulation et méthodes de résolution du problème de l'EPO donne des définitions de base sur la notion de l'écoulement de puissance optimal, la formulation du problème d'EPO, le dispatching économique et ces différents aspects.
— le deuxième chapitre intitulé *Méthodes conventionnelles de calcul d'écoulement de puissance optimal* explique en détail les méthodes conventionnelles que nous avons étudiées dans notre travail.
— le troisième chapitre intitulé *Méthode d'Optimisation par Essaim de Particules à la REP* décrit l'algorithme d'ALPSO que nous avons utilisé pour la résolution du problème de la REP.
— un quatrième chapitre intitulé *Simulations et discussion de résultats* présente les exemples résolus sur des réseaux standard que nous avons simulé en discutant les différents résultats obtenus par les différentes méthodes.

Chapitre 1

Formulation et méthodes de résolution du problème de l'écoulement de puissance optimal

1.1 Formulation du problème d'écoulement de puissance optimal

En général, l'OPF comprend tout problème d'optimisation qui cherche à optimiser le fonctionnement d'un système de réseau électrique (en particulier, la génération et la transmission de l'électricité), sous réserve de contraintes imposées par les lois électriques et les limites techniques sur les variables de contrôle. Ce cadre général englobe divers problèmes d'optimisation pour la planification et le fonctionnement des réseaux électriques. Il agit sur les variables de contrôle disponibles afin d'optimiser un objectif tout en satisfaisant les équations d'écoulement de puissance de réseau, les contraintes physiques et opérationnelles. L'écoulement de puissance optimal a été présenté au début des années 60 [1], et même avant par d'autres auteurs. Depuis ce temps, l'EPO est devenu un outil indispensable pour le développement, et la commande en temps réel des réseaux électriques. L'EPO est en général un problème non linéaire, avec des variables continues et discrètes. La formulation du problème d'EPO peut s'écrire comme suit :

$$minf(x,u) \tag{1.1}$$

Lié à :

$$g(x,u) = 0 \tag{1.2}$$

$$h(x) \leq 0 \tag{1.3}$$

$$U_{min} \leq U \leq U_{max} \tag{1.4}$$

Avec :

- u : variable de contrôle
- x : variable d'état
- L'équation 1.1 représente la fonction objective.
- L'équation 1.2 représente les contraintes d'égalités.
- L'équation 1.3 représente les contraintes d'inégalités.
- L'équation 1.4 représente les limites des variables de contrôle.

1.2 Signification des éléments de problème

1.2.1 Variables d'état

L'état du système est défini par la matrice admittance du réseau et les tensions aux nœuds.

1.2.2 Variables de contrôle

Les variables de contrôle dans l'EPO en générale sont :

- P_g : puissance active générée
- ϕ : transformateur de phase.
- Q_g : puissance réactive générée
- V : amplitude de tension

1.2.3 Contraintes égalités

Les contraintes d'égalité de l'EPO reflètent l'état du réseau électrique ainsi que la tension souhaitée à chaque nœud du système. L'état du réseau électrique est exprimé par les équations de l'écoulement de puissance qu'ils assurent l'équilibre du système.

1.2.4 Contraintes inégalités

On peut rencontrer deux types de contraintes inégalités :

— **Limites supérieures et inférieures**
Concernent les variables de contrôle car elles sont liées au fonctionnement des générateurs et à la stabilité du réseau.
$Pg_{min} < Pg < Pg_{max}$
$Qg_{min} < Qg < Qg_{max}$

— **Contraintes inégalité fonctionnelle**
On prend par exemple la puissance maximale transitée entre deux nœuds.

1.2.5 Fonction objectif

La fonction objectif de l'EPO peut prendre diverses formes selon l'objectif souhaité. Les objectifs les plus utilisés dans l'EPO sont

— la minimisation du coût de production de la puissance.
— la minimisation des pertes actives.
— la minimisation des pertes réactives
— minimisation de délestage
— maximisation de la limite de capacité de charge
— maximisation du profit
— minimisation des déviations causées par les violations des contraintes ... etc

Parmi les objectifs cités, la minimisation du coût de production de la puissance est l'une des fonctions objectif les plus utilisées pour l'EPO, ce qui a fait l'objet de notre travail dans les sections suivantes.

1.3 Étude de problème de la répartition économique de puissance

Le problème de la répartition économique a eu son début à partir du moment que deux ou plusieurs unités se sont engagés à prendre en charge sur un système de puissance dont la demande total dépassait la génération maximal d'une seule unité. Le problème qui se posait à l'opérateur était exactement comment diviser la génération entre les unités.

La répartition économique alors est fait en temps réel, à cause de la variation du coût et de la puissance demandé pour assigner la génération totale parmi les unités disponibles pour satisfaite la demande. La répartition économique est un processus de calcul selon laquelle la production totale nécessaire est répartie entre les unités de production disponibles de sorte que les contraintes imposées sont remplies et que les besoins en énergie en termes de BTU/h ou $\$/h$ sont minimisés.

1.3.1 Formulation du problème de la REP

Le problème de la répartition économique de puissance peut être formulé de la façon suivante :

$$minf(U = P_g) = min \sum_{i=1}^{N_{gen}} a_i + b_i P_{g_i} + c_i P_{g_i}^2 \tag{1.5}$$

Lié à :

$$\sum_{j \in i}^{N} V_i V_j [G_{ij} cos(\delta_i - \delta j) + B_{ij} sin(\delta_i - \delta j)] - Pg_i + Pd_i = 0 \tag{1.6}$$

$$\sum_{j \in i}^{N} V_i V_j [G_{ij} sin(\delta_i - \delta j) - B_{ij} cos(\delta_i - \delta j)] - Qg_i + Qd_i = 0 \tag{1.7}$$

$$P_{g_{min}} \leq P_g \leq P_{g_{max}} \tag{1.8}$$

tel que :

- N : nombre de nœuds.
- N_{gen} : nombre de nœuds de générations.
- G_{ij} :conductance des lignes.
- B_{ij} : susceptance des lignes.
- $\delta_{i,j}$: déphasage de la tension V_{ij} du nœud i(ou j)
- Pg_i : puissance active générée au nœud i.
- Pd_i : puissance active demandée au nœud i.
- Qg_i : puissance réactive générée au nœud i.
- Qd_i : puissance réactive demandée au nœud i.
- P_g est le variable de contrôle.
- V et θ sont les variables d'état.
- Pg_i est la quantité de la production en MW au générateur i.
- 1.5 représente le coût de production total.
- 1.6 et 1.7 représentent les equations de l'écoulement de puissance.
- 1.8 représente les limites des puissances générées

Note

Généralement, la fonction coût de chaque générateur est de forme quadratique de 2^{eme} ordre.

Le problème de la REP peut être résolu par plusieurs méthodes conventionnelles et d'autres non conventionnelles dites d'intelligence artificielle.

1.3.2 Les méthodes conventionnelles

Les techniques classiques dites aussi mathématiques ou encore conventionnelles appliquées au problème de la répartition économique de puissance, peuvent être classifiées en deux groupes. Le premier représente la famille des méthodes d'optimisation non linéaire (ou programmation non linéaire) [1] qui sont basées sur la théorie du calcul différentiel où le gradient et/ou le Hessien qui sont utilisés pour guider la procédure de recherche afin de localiser la solution optimale. Le deuxième groupe inclut les méthodes de programmation linéaire, qui sont fondées sur les techniques du simplexe et du point intérieur [2]. Le dispatching économique est un problème d'optimisation qui consiste à répartir la production de la

puissance active générée entre les différentes centrales du réseau, de sorte à exploiter ce dernier de la manière la plus économique possible. Cette distribution doit évidemment respecter les limites de production des centrales.

1.3.2.1 Méthode du Gradient

La méthode du Gradient cherche à déterminer la direction de descente qui fait décroitre $f(x + dx)$ le plus vite possible. $\nabla f(X_k)$ indique la direction du taux de décroissement de f au point x_k . Plusieurs façons d'utiliser cette direction de descente existent, dont on peut citer :

— Gradient simple
— Gradient à pas optimal
— Gradient conjugué
— Gradient projeté

1.3.2.2 Méthode de newton

La répartition économique de puissance peut être résolue avec pour but de résoudre le gradient du lagrangien $\nabla L_x = 0$. Puisque c'est une fonction vectorielle, le problème peut être formulé pour trouver le chemin qui conduit le gradient vers zéro. La méthode de Newton est utilisée dans ce sens. La méthode de Newton pour une fonction à plusieurs variables est développée comme suit . Soit une fonction g(x) à minimiser tel que :
Soit une fonction $g(x)$ à minimiser tel que
$g(x + \Delta x) = g(x) + [g'(x)]$ et $\Delta x = 0$.
La construction de la matrice Jacobienne $[G'(x)]$.
l'ajustement en chaque itération se fait suivant :
$\Delta x = -[g'(x)]^{-1} g(x)$. Maintenant si la fonction $g(x)$ est le vecteur de gradient ∇L_x, donc $\nabla x = -\left[\frac{\partial \nabla L_x}{\partial x}\right]^{-1} \nabla L$.
Les éléments de la matrice jacobienne sont les dérivées partielles de second ordre, et la matrice $[G(x)]$ est appelée matrice hessienne.

1.3.2.3 La méthode d'itération de Lambda

La méthode d'itération de Lambda est l'une des méthodes utilisées pour trouver la valeur de Lambda du système et trouver le dispatching économique optimal des générateurs. Contrairement aux autres méthodes d'itération, comme : Gauss-Seidel et Newton-Raphson, la méthode d'itération de Lambda n'utilise pas la valeur précédente de l'inconnue pour trouver la valeur suivante, c'est-à-dire qu'il n'y a pas d'équation qui calcule la valeur suivante en fonction de la valeur précédente. La valeur suivante est prédéfinie par intuition, elle est projetée avec interpolation de la bonne valeur possible jusqu'à ce que le décalage spécifié soit obtenu. On va maintenant discuter comment trouver le dispatching économique optimal utilisant cette dernière.

— La méthode exige qu'il y ait une correspondance entre une valeur *lambda* et l'output (en MW) de chaque générateur.

— — La méthode commence avec des valeurs de lambda en dessous et en-dessus de la valeur optimale (qui est inconnue), puis par itération limite la valeur optimale.

1.3.2.4 Méthode du point intérieur

Les méthodes de point intérieur fonctionnent comme suit : à partir d'une valeur initiale du paramètre de perturbation de la condition de complémentarité du système μ, strictement positif, et d'un point initial x_0 strictement réalisable, on résout de manière approchée le problème barrière. On calcule ensuite un nouveau paramètre de perturbation μ strictement positif et inférieur au précédent. On obtient alors un nouveau problème barrière à résoudre. On le résout de manière approchée à partir de la solution approchée du problème barrière précédent et ainsi de suite. On va donc résoudre, de manière approchée, une suite de problèmes barrières à μ fixé, la suite des paramètres de perturbation tendant vers 0, jusqu'à l'obtention d'une solution du problème initial.

1.3.3 Les méthodes d'intelligence artificielle

1.3.3.1 Algorithmes génétiques

Les algorithmes génétiques (AG) développés par J. Holland présentent des qualités intéressantes pour la résolution de problèmes d'optimisation complexes. Ils tentent de simuler le processus d'évolution des espèces dans leur milieu naturel : soit une transposition artificielle de concepts basiques de la génétique et des lois de survie énoncés par Darwin. Et aussi de la théorie de décision du mathématicien Hadamard du siècle passé.

Rappelons que la génétique représente un individu par un code, c'est-à-dire un ensemble de données (appelées chromosomes), identifiant complètement l'individu. La reproduction est dans ce domaine un mixage aléatoire de chromosomes de deux individus, donnant naissance à des individus enfants ayant une empreinte génétique nouvelle, héritée des parents. La mutation génétique est caractérisée dans le code génétique de l'enfant par l'apparition d'un chromosome nouveau, inexistant chez les individus parents.

Ce phénomène génétique d'apparition de " mutants" est rare mais permet d'expliquer les changements dans la morphologie des espèces, toujours dans le sens d'une meilleure adaptation au milieu naturel.

La disparition de certaines espèces est expliquée par " les lois de survie " selon lesquelles seuls les individus les mieux adaptés auront une longévité suffisante pour générer une descendance. Les individus peu adaptés auront une tendance à disparaître.

C'est une sélection naturelle qui conduit de génération en génération à une population composée d'individus de plus en plus adaptés. Un algorithme génétique est construit de manière tout à fait analogue.

Dans l'ensemble des solutions d'un problème d'optimisation, une population est constituée de solutions convenablement marquées par un codage qui les identifie complètement. Une procédure d'évaluation est nécessaire à la détermination de la force de chaque individu de la population. Viennent ensuite une phase de sélection (en sélectionnant les individus suivant de leur force) et une phase de recombinaison (opérateurs artificiels de croisement et de mutation) qui génèrent une nouvelle population d'individus, qui ont de bonnes chances d'être plus forts que ceux de

la génération précédente. De génération en génération, la force des individus de la population augmente et après un certain nombre d'itérations, la population est entièrement constituée d'individus tous forts, soit de solutions quasi-optimales du problème posé [3].

1.3.3.2 Algorithme à évolution différentielle

Classée parmi les méthodes méta-heuristiques stochastiques d'optimisation, l'algorithme à évolution différentielle (DEA) [4] est une technique relativement récente, conçue pour optimiser des problèmes sur les domaines continus. Cet algorithme est inspiré des algorithmes génétiques et les stratégies évolutionnistes combinées avec une technique géométrique de recherche. Les algorithmes génétiques changent la structure des individus en utilisant la mutation et le croisement, alors que les stratégies évolutionnistes réalisent l'auto adaptation par une manipulation géométrique des individus.

Dans cette approche, chaque variable de décision est représentée dans le chromosome (ou individu) par un nombre réel. Comme dans tout algorithme évolutionnaire, la population initiale du DEA est générée aléatoirement, puis évaluée. Après cela, le processus de sélection prend place. Au cours de la phase de sélection, trois parents sont choisis et ils génèrent un seul enfant (ou descendant) qui est en concurrence avec un parent pour déterminer lequel subsistera à la génération suivante. Le DEA génère un seul enfant (au lieu de deux comme dans les algorithmes génétiques) en ajoutant le vecteur de différence pondérée entre deux parents à un troisième parent. Si le vecteur résultant donne une plus faible valeur de la fonction objectif que celle donnée par un membre prédéterminé de la population, le vecteur nouvellement généré remplace le vecteur auquel il a été comparé.

1.3.3.3 Optimisation par essaim de particules

L'optimisation par essaim de particules est une technique évolutionnaire qui utilise une population de solutions candidates pour développer une solution optimale au problème. Le degré d'optimalité est mesuré par une fonction fitness [5]. Il est inspiré du comportement collectif et de l'intelligence émergente qui existent dans les sociétés à population organisée. Les membres de la population, particules, sont dispersés dans l'espace du problème, ainsi que le comportement de l'essaim peut

être décrit en se plaçant du point de vue d'une particule. Au départ de l'algorithme, un essaim est réparti au hasard dans l'espace de recherche, chaque particule ayant également une vitesse aléatoire. Chaque particule est représentée par une position et une vitesse, qui sont mis à jour comme suit :

$$X_i^{k+1} = V_i^{k+1} + X_i^k \tag{1.9}$$

$$V_i^{k+1} = \omega V_i^k + c_1 rand_1(Pbest_i^k - X_i^k) + c_2 rand_2(Gbest^k - X_i^k) \tag{1.10}$$

Remarque

Notons que les générateurs à combustibles distincts possèdent différents coûts pour fournir le même montant d'énergie électrique. C'est important de se rendre compte que le générateur le plus rentable du système ne peut pas produire de l'électricité au coût le plus bas et qu'un générateur économique ne peut pas être le plus rentable, puisqu'un générateur qui se trouve trop loin du lieu de consommation entraine des pertes de transmission énormes. Cependant, ces pertes varient en fonction de la répartition des puissances entre les centrales et la charge.

Dans ce chapitre, nous avons brièvement discuté de l'écoulement de puissance optimal et la répartition économique de puissance, ensuite, nous avons cité quelques méthodes conventionnelles et intelligentes d'une façon générale. Dans le chapitre suivant, nous présentons en en détails trois méthodes conventionnelles pour la répartition économique de puissance.

Chapitre 2

Méthodes conventionnelles de calcul d'écoulement de puissance optimal

2.1 Méthodes conventionnelles appliquées à la répartition économique de puissance

Le but de la répartition économique est de réduire au minimum le coût de production pour satisfaire la demande (charge) d'un système de puissance tout en maintenant sa sécurité. Dans ce chapitre, nous présenterons quelques méthodes conventionnelles d'optimisation pour atteindre le but désiré. D'abord, la fonction objective, $f(x)$, est présentée. Elle représente l'objectif dont nous nous servirons pour réduire au minimum le coût de la production. Puis les contraintes d'égalités et d'inégalités seront présentées et discutées. Ces contraintes modélisent les lois physiques du système de puissance d'énergie électrique, ainsi que les conditions nécessaires pour le maintenir en sécurité. Généralement, les valeurs initiales de l'OPF sont les solutions de l'écoulement de puissance dont nous allons parler ci dessous.

2.2 Calcul de l'écoulement de puissance

Le calcul de l'écoulement de puissance consiste à résoudre les équations(1.6) et (1.7). Chaque nœud dans le réseau est caractérisé par $Pg_i, Qg_i, V_i, \theta_i$. Nous fixons

deux paramètres et nous devons déterminer les deux autres. Suivant les valeurs spécifiées, les nœuds sont classés en trois types :

— Nœud de référence (slack bus) : V et θ fixes, P et Q inconnues.
— Nœud de génération : P et V fixes, Q et θ inconnues.
— Nœud de charge : P et Q fixes, V et θ inconnues.

trois méthodes classiques sont généralement utilisées

2.2.1 Méthode de Gauss-Seidel

Les premières méthodes étaient basées sur la méthode de $Gauss - Seidel$ relative à la matrice admittance Y . Elle ne nécessite pas beaucoup d'espace mémoire et sa programmation est relativement simple. Pour un système à plusieurs variables, La méthode de $Gauss - Seidel$ utilise à chaque itération la valeur la plus récente calculée. La méthode de $Gauss - Seidel$ se caractérise par sa faible convergence ; elle peut diverger complètement si la valeur initiale est mal choisie. Mais les petits réseaux ne nécessitent que peu d'itérations pour converger. Les grands réseaux, par contre demandent un grand nombre d'itérations si toutefois ils convergent. Ce désavantage a poussé les chercheurs à développer la méthode de $Newton - Raphson$.

2.2.2 Méthode de Newton-Raphson

Cette méthode nécessite plus de temps par itération que celle de Gauss-Seidel, alors qu'elle ne demande que quelques itérations même pour les grands réseaux. Cependant, elle requiert des capacités de stockage ainsi que des puissances de calcul importantes. A cause de la convergence quadratique de la méthode de$Newton - Raphson$, une solution de haute précision peut être obtenue en quelques itérations seulement. Ces caractéristiques font le succès de la méthode découplée rapide de $Newton - Raphson$.

2.2.3 Méthode Découplée rapide (Fast Decoupled Load Flow)

La variation de la puissance active est moins sensible à la variation de la tension V , par contre, elle est sensible à celle de la phase θ. D'autre part, la variation

de la puissance réactive est plus sensible à la variation de la tension V , et est moins sensible à celle de la phase θ. La méthode découplée rapide (FDLF) effectue le même temps d'exécution que celle de $Newton - Raphson$ pour les très petits réseaux. Cependant, elle devient plus rapide pour les réseaux plus importants et pour les tolérances habituelles.

2.3 Méthode de Newton

2.3.1 Définition et généralisation de la méthode

En analyse numérique, la méthode de Newton est un algorithme efficace pour trouver numériquement une approximation précise d'un zéro (ou racine) d'une fonction réelle d'une variable réelle. Cette méthode a été développée par $Isaac\ Newton(1643-1727)$) qui a fait les premiers travaux pour la recherche des zéros d'une équation polynomiale, et aussi $Thomas\ Simpson(1710-1761)$ qui a élargi considérablement le domaine d'application de l'algorithme en montrant, grâce à la notion de dérivée, comment on pouvait l'utiliser pour calculer un zéro d'une équation non linéaire et d'un système formé de telles équations.

Nous allons donc chercher à construire une bonne approximation d'un zéro d'une fonction réelle $f(x)$ en considérant son développement de Taylor au premier ordre. Pour cela, partant d'un point x_0 que l'on choisit de préférence proche du zéro, on approche la fonction au premier ordre, autrement dit, on la considère qu'elle est égale à sa tangente en ce point :

$$f(x) = f(x_0) + f'(x_0)(x - x_0) \tag{2.1}$$

Puis, on résout l'équation $f(x) = 0$
On obtient alors un point x_1 qui a en général de bonnes chances d'être plus proche du vrai zéro de f que le point x_0 précédent. Par cette opération, on peut donc améliorer l'approximation par itérations successives. Cette méthode requiert que la fonction possède une tangente en chacun des points de la suite que l'on construit par itération donc, il suffit que f soit dérivable. Formellement, on part d'un point x_0 appartenant à l'ensemble de définition de la fonction et on construit par récurrence la suite

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} \tag{2.2}$$

le processus itérative se termine lorsque $(x_{k+1} - x_k)$ inférieur à la tolérance.

On peut aussi utiliser la méthode de Newton pour résoudre un système de N équations (non linéaires) à N inconnues $x = [x_1, x_2, ...x_n]$, ce qui revient à trouver un zéro d'une fonction F multivariables , qui devra être différentiable. Dans la formulation donnée ci-dessus, il faut multiplier par l'inverse de la matrice jacobienne $[F(x)]$ au lieu de diviser par $f'(x)$.

2.3.2 Utilisation de la méthode de Newton dans l'optimisation

L'application de la méthode de *Newton* à l'optimisation consiste à minimiser ou maximiser une fonction objective liée à des contraintes d'égalités et d'inégalités. D'après le problème représenté par les équations 1.1, 1.2, 1.3 nous disposons de M contraintes de type égalités et N contraintes de type inégalités. Le nombre de variables est égal à la dimension du vecteur X, tel que $X = [x1, x2, ..., xk]$.

2.3.3 Développement du Lagrangien, du Gradient, et du Hessien

La solution de problème présenté par 1.1, 1.2, 1.3 avec la méthode de Newton exige la création de lagrangien comme ci-dessous

$$L(z) = f(x) + \lambda^t g(x) + \mu^t h(x) \tag{2.3}$$

avec

- $Z = [x, \lambda, \mu]^t$
- λ et μ sont les vecteurs de multiplicateur de Lagrange.

Le gradient et l'hessien donc peuvent s'exprimer sous la forme suivant :

- $Gradient = \nabla L(z) = \dfrac{\partial L(z)}{\partial z_i}$. Vecteur de la première dérivée partiale du lagrangien

$$— \text{ Hessien} = \nabla^2 L(z) = H = \frac{\partial^2 L(z)}{\partial z_i \partial z_j} = \begin{bmatrix} \frac{\partial^2 L(z)}{\partial x_i \partial x_j} & \frac{\partial^2 L(z)}{\partial x_i \partial \mu_j} & \frac{\partial^2 L(z)}{\partial x_i \partial \lambda_j} \\ \frac{\partial^2 L(z)}{\partial \mu_i \partial x_j} & 0 & 0 \\ \frac{\partial^2 L(z)}{\partial \lambda_i \partial x_j} & 0 & 0 \end{bmatrix}$$

H : matrice du deuxième dérivé partial de lagrangien.
Selon la théorie d'optimisation, les conditions nécessaires de Kuhn-Tucker de l'optimalité sont mentionnées comme ci-dessous.

2.3.4 Conditions de Kuhn-Tucker

Soit z^* la solution optimal, alors on aura :

- $\nabla_x L(z^*) = \nabla_x L([x^*, \lambda^*, \mu^*]) = 0$
- $\nabla_\lambda L(z^*) = \nabla_\lambda L([x^*, \lambda^*, \mu^*]) = 0$
- $\nabla_\mu L(z^*) = \nabla_\mu L([x^*, \lambda^*, \mu^*]) = 0$

$\lambda_i^* \geq 0$, si $g(x^*) = 0$ (la contrainte inégalité est active).
$\lambda_i^* = 0$, si $g(x^*) \leq 0$ (la contrainte inégalité est non active).
Le développement du gradient de lagrangien nous donne :

- $\frac{\partial L}{\partial U} = \frac{\partial F}{\partial U} + \lambda \frac{\partial G}{\partial U} + \mu \frac{\partial H}{\partial U} = 0$
- $\frac{\partial L}{\partial X} = \frac{\partial F}{\partial X} + \lambda \frac{\partial G}{\partial X} + \mu \frac{\partial H}{\partial X} = 0$
- $\frac{\partial L}{\partial \lambda} = G(U, X) = 0$
- $\frac{\partial L}{\partial \mu} = H(U, X) = 0$

On peut noter que le Lagrangien inclut seulement les inégalités qui sont imposées. Par exemple, si la tension d'un nœud i est dans la plage de fonctionnement désirée, il n'y a aucun besoin d'activer la contrainte inégalité liée à cette tension. Pour la formulation de la méthode de Newton, les contraintes d'inégalités doivent être manipulées en les séparant deux à deux : actif et inactif [6]. Pour des algorithmes efficaces, la détermination de ces contraintes d'inégalités qui sont actives a une

grande importance. La boucle externe de l'organigramme exécute cette recherche des contraintes obligatoires ou actives [6].

2.3.5 Algorithme de la méthode de Newton

Une fois le gradient et la hessienne sont déterminer on les introduisant dans algorithme.

1. initialisez le vecteur $[z]$ et les inégalités qui sont imposés.
2. création de lagrangien on prend par considération les contraint d'inégalité active.
3. calculez le hessien et le gradient de lagrangien.
4. résoudre l'équation $\Delta z = -[H]^{-1}\nabla L(z)$.
5. calculez le nouveau $z_{new} = z_{old} + \Delta z$.
6. test si $\Delta z < toll$ aller à 7 si non aller à 3.
7. vérifier si les inégalité sont remplie si oui aller à 9 si non aller à 8.
8. Déterminer le nouveau ensemble d'inégalités et aller à 2.
9. fin.

L'organigramme de la méthode est illustré dans la figure suivante

2.3.6 Application de la méthode de Newton à l'EPO

Dans le domaine des réseaux électriques, la méthode de Newton est bien connue comme l'une des méthodes de solution de l'écoulement de puissance optimal. Elle été l'algorithme standard de la solution du problème de l'EPO pendant une longue période. L'approche de Newton est une formulation souple qui peut être adoptée pour développer des algorithmes différents pour l'EPO adaptés aux exigences des différentes applications comme la répartition économique de puissance.

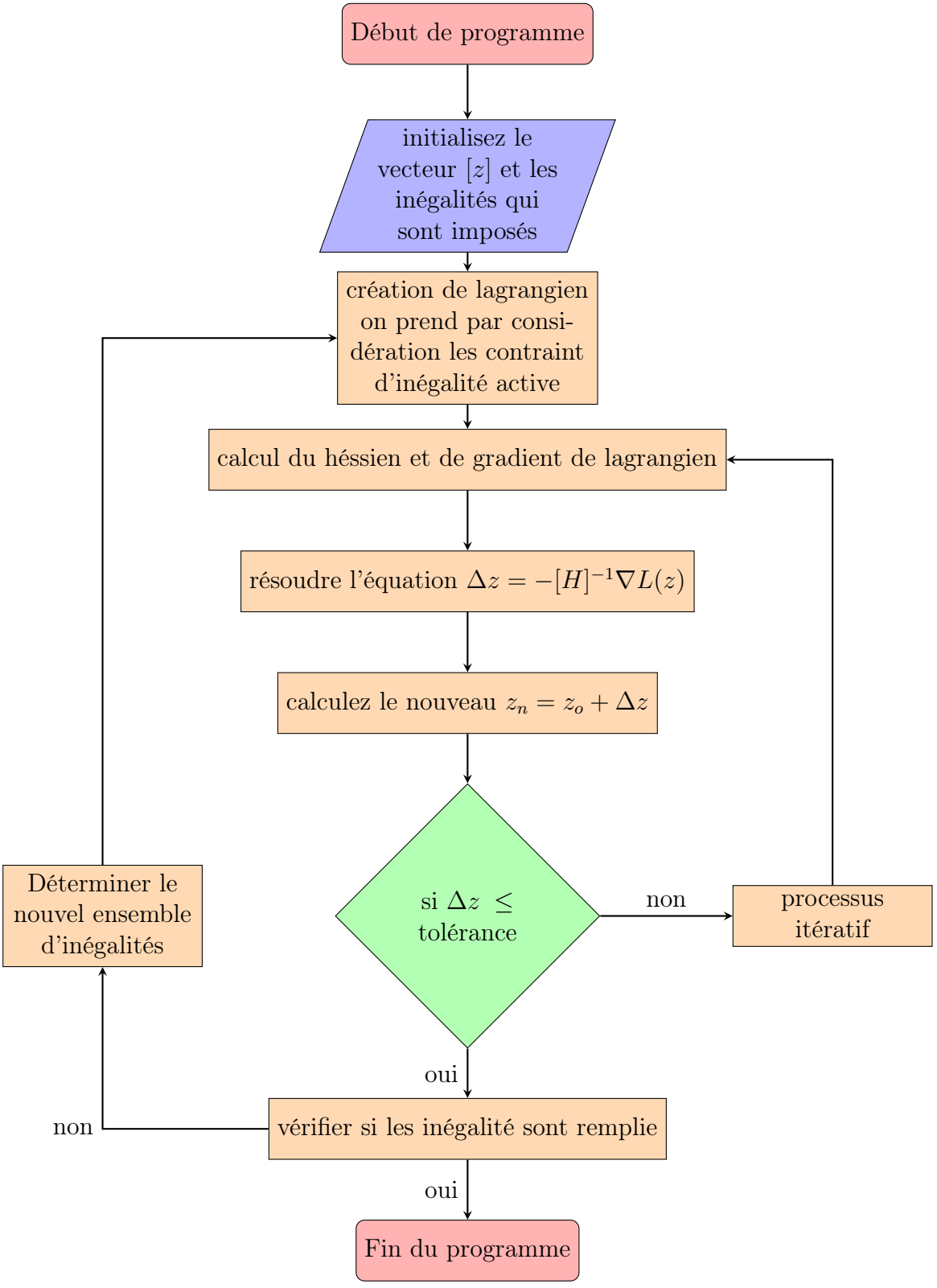

FIGURE 2.1: Organigramme d'algorithme du méthode de Newton

Soit $Z = [Pg_1, ..., Pg_{ng}, V_{ng+1}, ..., V_n, \theta_2, ..., \theta_n, \lambda_{p_1}, ..., \lambda_{p_n}, \lambda_{q_{ng+1}}, ..., \lambda_{q_n}]$.

La fonction lagrangienne

$$\begin{aligned} L(z) = & \sum_{i=1}^{N_{gen}} a_i + b_i P_{g_i} + c_i P_{g_i}^2 + \\ & \sum_{i=1}^{N} \lambda_{pi} \left[\sum_{j=1}^{n} V_i V_j [G_{ij} cos(\delta_i - \delta j) + B_{ij} sin(\delta_i - \delta j)] - Pg_i + Pd_i \right] + \\ & \sum_{i=ng+1}^{N} \lambda_{qi} \left[\sum_{j=1}^{n} V_i V_j [G_{ij} sin(\delta_i - \delta j) - B_{ij} cos(\delta_i - \delta j)] - Qg_i + Qd_i \right]. \end{aligned} \tag{2.4}$$

La condition d'obtention du minimum de coût de génération avec les équations de l'écoulement de puissance sont vérifier. Aussi, on a :

$$\nabla L(z) = \left[\frac{\partial L}{\partial z_i} \right] = \left[\frac{\partial L}{\partial p_g}, \frac{\partial L}{\partial v}, \frac{\partial L}{\partial \theta}, \frac{\partial L}{\partial \lambda_p}, \frac{\partial L}{\partial \lambda_q} \right]^t = 0. \tag{2.5}$$

La construction de la matrice hessienne est obtenue à partir du calcul des dérivés mixte :

$$H = \nabla^2 L(z) = \begin{bmatrix} \frac{\partial^2 L(z)}{\partial P_{g_i} \partial P_{g_j}} & \frac{\partial^2 L(z)}{\partial P_{g_i} \partial V_j} & \frac{\partial^2 L(z)}{\partial P_{g_i} \partial \theta_j} & \frac{\partial^2 L(z)}{\partial P_{g_i} \partial \lambda_{P_j}} & \frac{\partial^2 L(z)}{\partial P_{g_i} \partial \lambda_{Q_j}} \\ \frac{\partial^2 L(z)}{\partial V_i \partial P_{g_j}} & \cdots & \cdots & \cdots & \\ \cdots & \cdots & \cdots & \cdots & \\ \frac{\partial^2 L(z)}{\partial \lambda_i \partial P_{g_j}} & \cdots & \cdots & \cdots & \frac{\partial^2 L(z)}{\partial \lambda_i \partial \lambda_j} \end{bmatrix} \tag{2.6}$$

$$\Delta Z = [\Delta Pg_1, .., \Delta Pg_{ng}, \Delta V_{ng+1}, .., \Delta V_n, \Delta\theta_2, .., \Delta\theta_n, \Delta\lambda_{p_1}, .., \Delta\lambda_{p_n}, \Delta\lambda_{q_{ng+1}}, .., \Delta\lambda_{q_n}] \tag{2.7}$$

Les nouvelles valeurs de $[z]$ sont obtenues comme suit :

$$[z]_{k+1} = [z]_k + [\Delta z] \tag{2.8}$$

Le calcul itératif se poursuivra jusqu'à ce que $[\Delta z]$ soit inférieur à une tolérance fixée.

2.4 Méthode du Gradient

2.4.1 Définition

L'algorithme du gradient désigne un algorithme d'optimisation différentiable. Il est par conséquent destiné à minimiser une fonction réelle différentiable définie sur un espace euclidien. L'algorithme est itératif et procède donc par améliorations successives. Au point courant, un déplacement est effectué dans la direction opposée au gradient, de manière à faire décroître la fonction. Le déplacement le long de cette direction est déterminé par la technique numérique connue sous le nom de recherche linéaire. Cette description montre que l'algorithme fait partie de la famille des algorithmes à directions de descente.

2.4.2 Algorithme de la méthode

On se donne un point initial x_0 et un seuil de tolérance $\epsilon_0 > 0$. l'algorithme du gradient définit une suite d'itération $x_1, x_2, ..., x_k$,jusqu'à ce qu'un test d'arrêt soit satisfait. Il passe de x_k à x_{k+1} par les étapes suivantes :

- Simulation : calcul de $\nabla f(x_k)$.
- Test d'arrêt : si $\|\nabla f(x_k)\| \leq \epsilon_0$, arrêt.
- Calcul du pas $\alpha_k > 0$ par une règle de recherche linéaire sur f en x_k le long de la direction $(-\nabla f(x_k))$.
- Nouvel itéré : $x_{k+1} = x_k - \alpha_k \nabla f(x_k)$.

La direction de descente est exprimée comme suite :
$d = \dfrac{-\nabla f(x)}{\| \nabla f(x) \|^2}$
Orienté donc vers l'opposé du gradient.

2.4.3 Application de la méthode de gradient en EPO

Comme dans la méthode de newton la fonction objective est toujours la fonction coût totale de génération, $C = \sum f(Pg_i)$.
Cette somme contient tous les générateurs du système, y compris le générateur du

nœud référence. Nous commencerons à définir le vecteur X d'état comme suit :

$$X = \begin{bmatrix} \left.\begin{matrix} \theta_i \\ |V_i| \end{matrix}\right\} & \text{pour les nœuds PQ} \\ \theta_i & \text{pour les nœuds PV} \end{bmatrix} \tag{2.9}$$

Un autre vecteur Y est défini comme suit :

$$Y = \begin{bmatrix} \left.\begin{matrix} \theta_k \\ |V_k| \end{matrix}\right\} & \text{pour le nœud de référence} \\ \left.\begin{matrix} P_k^{sp} \\ Q_k^{sp} \end{matrix}\right\} & \text{pour les nœuds PQ} \\ \left.\begin{matrix} P_k^{sp} \\ V_k^{sp} \end{matrix}\right\} & \text{pour les nœuds PV} \end{bmatrix} \tag{2.10}$$

Noter que le vecteur Y se compose de tous les paramètres qui doivent être spécifiés. Certains de ces paramètres sont réglables (par exemple, la puissance générée Pg, et la tension des nœuds PV). D'autres paramètres sont fixes pour ce qui concerne le calcul d'EPO, comme le P et le Q à chaque nœud de charge. Pour faire cette distinction, nous diviserons le vecteur Y en deux parties, u et p : $Y = [u, p]^t$ où u représente le vecteur de contrôle ou les variables réglables, et p représente les variables fixes ou constantes. On note également que nous représentons seulement des contraintes d'égalités en ce moment. Finalement nous définirons l'ensemble des $n1$ équations d'écoulement de puissance .

$$g(x, y) = \begin{bmatrix} \left.\begin{matrix} P_i(|V|, \theta) - P_i^{spe} \\ Q_i(|V|, \theta) - Q_i^{spe} \end{matrix}\right\} & \text{pour les nœuds PQ} \\ P_k(|V|, \theta) - P_k^{spe} & \text{pour les nœuds PV sauf slack bus} \end{bmatrix} \tag{2.11}$$

Nous devons savoir que la production de l'énergie électrique du nœud de référence (slack-bus) n'est pas une variable indépendante, c'est-à-dire, que la génération du nœud de référence change toujours pour équilibrer l'écoulement de puissance ; nous ne pouvons pas la spécifier au début du calcul. Nous souhaitons exprimer le coût ou la fonction objective en fonction des variables de contrôle et des variables d'état. Nous faisons ceci en divisant la fonction de coût comme suit

$$C = \sum_{i=1}^{n-1} f(Pg_i) + f_{ref} Pg_{ref} \tag{2.12}$$

avec : $P_{ref} = P_{ref}(|V|, \theta)$, alors an a :

$$C = \sum_{i=1}^{n-1} f(Pg_i) + f_{ref}[Pg_{ref}(|V|, \theta)] = f(x, u). \tag{2.13}$$

Maintenant on peut construire l'équation lagrangienne pour l'EPO comme suit :

$$L(x, u, p) = f(x, u) + \lambda^t g(x, u, p). \tag{2.14}$$

avec

- x = vecteur d'état
- u = vecteur des variables de contrôle
- p = vecteur des paramètres fixes
- λ = vecteur des multiplicateur de Lagrange
- g = ensemble des contraintes égalités qui représente les équations de l'écoulement de puissance.
- f = la fonction objective

L'équation de Lagrange est peut être plus claire sous la forme suivante :

$$L(x, u, p) = \sum_{i=1}^{n-1} f(Pg_i) + f_{ref}[Pg_{ref}(|V|, \theta)] + [\lambda_1, \lambda_2, ..., \lambda_m] \begin{bmatrix} P_i(|V|, \theta) - P_i^{spe} \\ Q_i(|V|, \theta) - Q_i^{spe} \\ P_k(|V|, \theta) - P_k^{spe} \\ \vdots \end{bmatrix} \tag{2.15}$$

Nous avons maintenant une fonction de Lagrange qui a une fonction objectif simple et m multiplicateurs de Lagrange, un pour chacune des équations de l'écoulement de puissance. Pour réduire au minimum la fonction de coût, lié aux contraintes, on met le gradient de la fonction de Lagrange égale à zéro :
$\nabla L = 0$.
Suivant les trois variables x, u etλ, on a :

$$\nabla L_x = \frac{\partial L}{\partial x} = \frac{\partial f}{\partial x} + \left[\frac{\partial g}{\partial x}\right]^T \lambda \tag{2.16}$$

$$\nabla L_u = \frac{\partial L}{\partial u} = \frac{\partial f}{\partial u} + \left[\frac{\partial g}{\partial u}\right]^T \lambda \tag{2.17}$$

$$\nabla L_\lambda = \frac{\partial L}{\partial \lambda} = g(x, u, p) \tag{2.18}$$

Tel que :

$$\frac{\partial f}{\partial x} = \begin{bmatrix} \dfrac{\partial}{\partial P_{ref}} F_{ref}(P_{ref}) \dfrac{\partial P_{ref}}{\partial \theta_1} \\ \dfrac{\partial}{\partial P_{ref}} F_{ref}(P_{ref}) \dfrac{\partial P_{ref}}{\partial V_1} \\ \vdots \end{bmatrix} \tag{2.19}$$

$$\frac{\partial g}{\partial x} = \begin{bmatrix} \dfrac{\partial P_1}{\partial \theta_1} & \dfrac{\partial P_1}{\partial V_1} & \dfrac{\partial P_1}{\partial \theta_2} & \dfrac{\partial P_1}{\partial V_2} & \cdots \\ \dfrac{\partial Q_1}{\partial \theta_1} & \dfrac{\partial Q_1}{\partial V_1} & \dfrac{\partial Q_1}{\partial \theta_2} & \dfrac{\partial Q_1}{\partial V_2} & \cdots \\ \dfrac{\partial P_2}{\partial \theta_1} & \dfrac{\partial P_2}{\partial V_1} & & \cdots & \\ \dfrac{\partial Q_2}{\partial \theta_1} & \dfrac{\partial Q_2}{\partial V_1} & & \cdots & \end{bmatrix} \tag{2.20}$$

$$\frac{\partial f}{\partial u} = \begin{bmatrix} \dfrac{\partial}{\partial P_1} F_1(P_1) \\ \dfrac{\partial}{\partial P_2} F_2(P_2) \\ \vdots \end{bmatrix} \tag{2.21}$$

La solution par la méthode du gradient se présente comme suit :

1. Donner l'ensemble des paramètres fixes $[p]$
2. Initialiser les variable de contrôle
3. Résoudre le problème de l'écoulement de puissance (2.18)
4. Retirer lambda de l'équation (2.16)
5. Résoudre l'équation (2.17)pour lambda égale à :$\lambda = -\left[\dfrac{\partial g}{\partial x}\right]^{T^{-1}} \dfrac{\partial f}{\partial x}$.

Finalement on obtient :

$$\frac{\partial L}{\partial U} = \frac{\partial F}{\partial U} - \left(\frac{\partial G}{\partial X}\right)^T \left[\left(\frac{\partial G}{\partial X}\right)^T\right]^{-1} \frac{\partial F}{\partial X} \tag{2.22}$$

2.4.4 Application de la méthode du gradient de deuxième ordre dans la REP

La méthode du gradient de deuxième ordre converge rapidement après une ou deux itérations seulement. Si l'une des limites dépasse l'intervalle de fonctionnement, on fixe les puissances qui dépassent leurs limites et on refait le calcul pour les autres puissances. Le problème est de : Minimiser $f(pg)$ Liée à $\sum Pg_i = Pc$
avec Pc : la puissance totale consommé. Pour optimiser le problème précèdent on utilise le gradient de la fonction objective et celui de la contrainte égalité. Le développement en série de Taylor nous donne [7] :

$$\begin{aligned} F + \Delta F = F_1(Pg_1) + F_2(Pg_2) + ... + F_n(Pg_n) + \frac{dF_1}{dPg_1}\Delta Pg_1 + ... + \frac{dF_n}{dPg_n}\Delta Pg_n \\ + \frac{1}{2}\left[\frac{d^2F_1}{dPg_1^2}(\Delta Pg_1)^2 + \frac{d^2F_2}{dPg_2^2}(\Delta Pg_2)^2\right] + ... \end{aligned} \tag{2.23}$$

Dans notre cas, la fonction objective est une fonction quadratique de deuxième ordre avec chaque fonction F_i qui ne dépend que de P_gi , donc les dérivées mixtes de deuxième ordre sont nuls.

$$\frac{\partial^2 F_i}{\partial Pg_i \partial Pg_j} = 0. \tag{2.24}$$

Et à partir de la contraint d'égalité on déduire que :

$$\sum Pgi = Pc \Rightarrow \sum \Delta Pg_i = 0 \Rightarrow \Delta Pg_j = \sum \Delta Pg_i \tag{2.25}$$

Tel que P gj est le nœud dépendant. On remplace 2.25 dans 2.23 on obtient :

$$\begin{aligned} \Delta F = \sum_{i \neq j}\left(\frac{dF_i}{dPg_i} - \frac{dF_j}{dPg_j}\right)\Delta Pg_i + \\ \frac{1}{2}\left\{\sum_{i \neq j}\frac{d^2F_i}{dPg_i^2}(\Delta Pg_i)^2 + \frac{d^2F_j}{dPg_j^2}\left[\sum_{i \neq j}\left(\Delta Pg_i)^2 + 2\Delta Pg_i \sum_{k \neq j}\Delta Pg_k\right)\right]\right\} \end{aligned} \tag{2.26}$$

on fait dérivée la ΔF par rapport à ΔPg pour $i \neq j$. Le résultat est :

$$\frac{\partial \Delta F}{\partial \Delta Pg_i} = 0 \Rightarrow \left(\frac{dF_i}{dPg_i} - \frac{dF_j}{dPg_j}\right) + \frac{d^2F_i}{dPg_i^2}\Delta Pg_i + \frac{d^2F_j}{dPg_j^2}\sum_{i \neq j}\Delta Pg_j = 0. \quad (2.27)$$

on pose :
$F_i^{'} = \dfrac{dF_i}{dPg_i}$
$F_i^{''} = \dfrac{d_i^F}{dPg_i^2}$
Pour tous les équations de système la forme matricielle qui représente 2.27 est le suivant :

$$\begin{bmatrix} F_1^{''} + F_j^{''} & F_j^{''} & \cdots & \cdots & F_j^{''} \\ F_j^{''} & F_2^{''} + F_j^{''} & \ddots & & \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & F_j{''} \\ F_j{''} & \cdots & \cdots & F_j{''} & F_n{''} + F_j{''} \end{bmatrix} \begin{bmatrix} \Delta Pg_1 \\ \vdots \\ \vdots \\ \vdots \\ \Delta Pg_n \end{bmatrix} = - \begin{bmatrix} F_1^{'} - F_j^{'} \\ \vdots \\ \vdots \\ \vdots \\ F_n^{'} - F_j^{'} \end{bmatrix} \quad (2.28)$$

L'algorithme de la méthode est comme suit [7] :

1. Calcule de l'écoulement de puissance et initialisation des Pg_i.
2. Déterminer le nœud dépendant
3. Calculer la matrice F"
4. Résoudre le problème 2.28 et déterminer les ΔPg_i
5. calculer les nouvelles valeurs des Pg_i tel que $Pg_{i_{new}} = Pg_{i_{old}} + \Delta Pg_i$
6. vérifier si l'un des variables est en dehors de la plage désiré
7. si oui fixer le et refaire le calcule pour les autres variable
8. Calculer l'erreur, si l'erreur est inférieur à la tolérance la solution est atteinte si non réitérer.

L'organigramme de la méthode est comme suit :

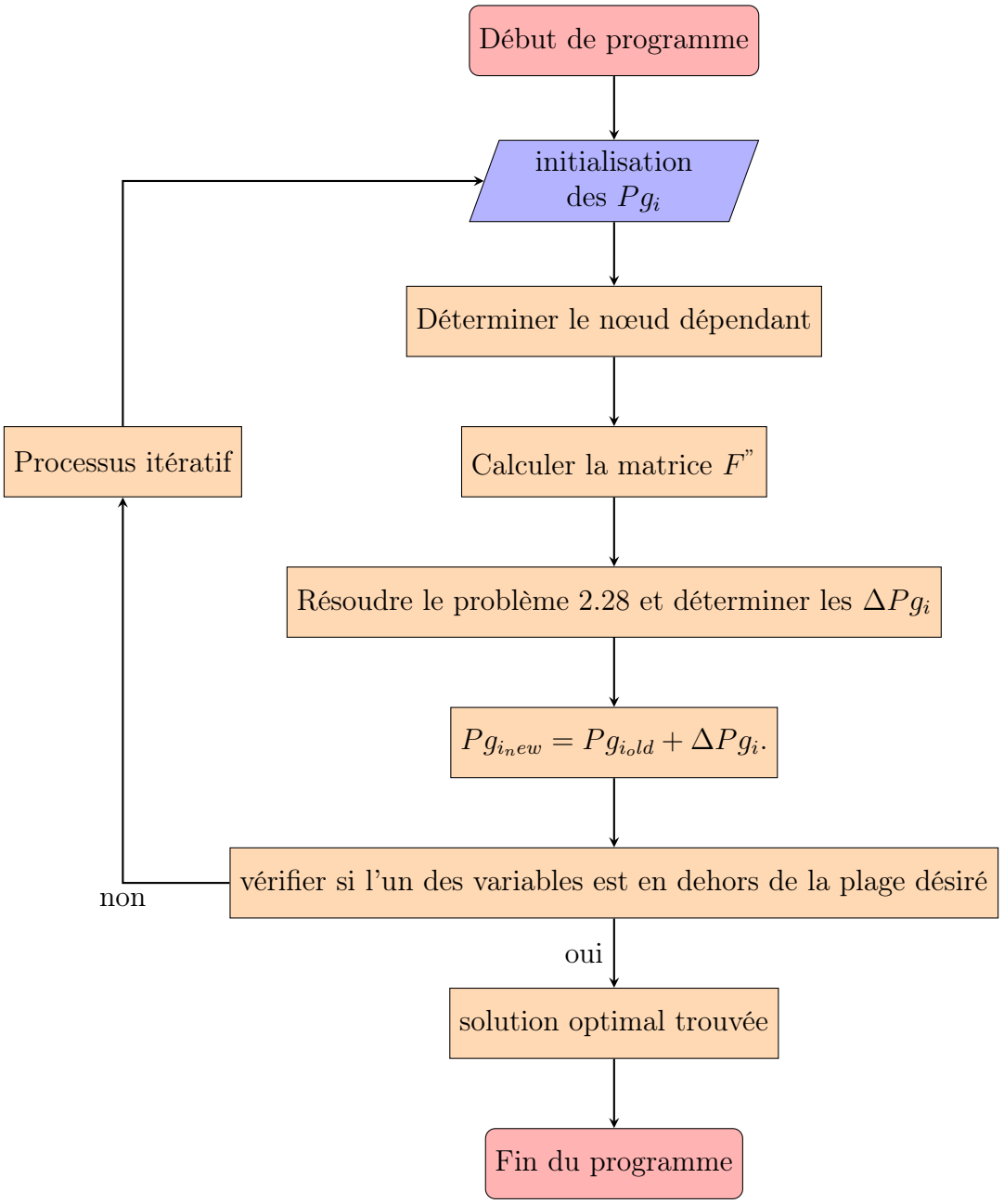

FIGURE 2.2: Organigramme de la méthode de gradient

2.5 Méthode d'optimisation du point intérieur (MPI)

La méthode du point intérieur est l'une des méthodes les plus utilisées pour le calcul d'EPO. Elle est apparue dans les années 50 et développée par Fiacco et McCormick [8] et devenu très populaire après la publication de Karmarkar [9]. Ce dernier travail a démontré que la méthode du point intérieur est plus performante que la méthode classique du simplex surtout pour les grands problèmes de programmation linéaire et surtout de programmation non linéaire. La MPI a plusieurs applications dans les réseaux électriques comme : la minimisation des coûts de production de l'énergie électrique, la minimisation des pertes actives, la minimisation de délestage, maximisation de sécurité social. Le primaire-dual prédicateur-correcteur algorithme de point intérieur proposé par Merhotra [10] est devenu la référence de la MPI.

Les principales caractéristiques de la MPI concernent sa convergence rapide et son traitement adéquat des contraintes inégalités par les fonctions logarithme de barrière. Comme il y a des avantages, il y a aussi des inconvénients : l'heuristique de réduire le paramètre de barrière et le fait que les variables d'écart et leurs variables duales correspondantes doivent rester positifs à chaque itération qui peut raccourcir considérablement la durée du pas de Newton. Des approches ponctuelles non-intérieures de l'EPO ont proposées, telles que : l'algorithme de point illimité, la méthode de complémentarité, le lissage jacobien. Ces dernières méthodes prouvent des propriétés de convergence comparables avec les meilleurs algorithmes de points intérieurs.

2.5.1 Formulations

La formulation du problème d'EPO peut s'écrire sous la forme suivante :

$$\begin{aligned} & \min\ f(x) \\ \text{Lié à :} \quad & g(x) = 0 \\ & h_l \leqslant h(x) \leqslant h_u \end{aligned}$$

tel que f est la fonction objective dont la somme des fonction de coût de production de chaque centrale.

$$min \sum_{i \in G} c_{0i} + c_{1i} P_{gi} + c_{2i} P_{gi}^2 \tag{2.29}$$

où c_{0i},c_{1i}, et c_{2i} sont les coefficients qui décrivent l'allure de cout de production de chaque centrale. La fonction g est l'ensemble des contraintes égalités tel que :

$$P_{gi} - P_{ci} - P_i = 0, i \in N \tag{2.30}$$

$$Q_{gi} - Q_{ci} - Q_i = 0, i \in N \tag{2.31}$$

P_{gi} : La puissance générée.
P_{ci} : La puissance consommée.
P_i : La puissance injectée à chaque nœud.
Les puissances actives et réactives injectée s'écrivent sous la forme suivante :

$$P_i = \sum_{j \in i} V_i V_j [G_{ij} cos(\delta_i - \delta j) + B_{ij} sin(\delta_i - \delta j)] \tag{2.32}$$

$$Q_i = \sum_{j \in i} V_i V_j [G_{ij} sin(\delta_i - \delta j) - B_{ij} cos(\delta_i - \delta j)] \tag{2.33}$$

dont G_{ij}, B_{ij} sont les éléments de la matrice admittance Y_{ij}, ($Y_{ij} = G_{ij} + jB_{ij}$), et δ_i est le déphasage de tension V_i.
La fonction h est l'ensemble des contraintes inégalités. Il y a deux types de contraintes inégalités :

- Contraintes opérationnelles qui assurent un fonctionnement sécurisé du système.
- Contraintes physiques de chaque élément de réseaux électrique (les valeurs minimales et maximal de fonctionnement de chaque élément).

Dans notre cas, on peut considérer les contraintes suivantes :

$$(G_{ij}^2 + B_{ij}^2)[(e_i - e_j)^2 + (f_i - f_j)^2] \leq (I_j^{max})^2, j \in N \tag{2.34}$$

$$P_{gi}^{min} \leq P_{gi} \leq P_{gi}^{max}, i \in G \tag{2.35}$$

$$Q_{gi}^{min} \leq Q_{gi} \leq Q_{gi}^{max}, i \in G \tag{2.36}$$

N : nombre de nœuds.
G : nombre de générateurs.

2.5.2 Conditions d'optimalité et mise à jour des variables

Le problème d'EPO peut s'écrit comme suit :

$$minf(x) \tag{2.37}$$

Lié a :

$$g(x) = 0 \tag{2.38}$$

$$h_l \leq h(x) \leq h_u \tag{2.39}$$

$$x_l \leq x \leq x_u \tag{2.40}$$

où les dimensions du vecteur inconnu x, de la fonction g, et de la fonction h sont n, m, et p respectivement.
Afin de simplifier la présentation, nous englobons les contraintes de (2.40) dans les contraintes d'inégalité (2.39). L'algorithme de point intérieur comprend quatre étapes. D'abord, nous transformons la contrainte inégalité en une contrainte égalité en ajoutant des nombre positifs s_l et s_u aux deux bornes de l'inégalité (2.39). Ensuite, les conditions de non-négativité sont implicitement traitées en les ajoutant à la fonction objectif comme termes de barrière logarithmiques. Nous transformons ensuite le problème d'optimisation avec contraintes d'égalité en un problème d'optimisation sans contraintes. Finalement, nous résolvons le système des conditions d'optimalités de $Karush-KuhnTucker$ (KKT) du premier ordre avec la méthode de Newton.

Après l'application des transformations précédentes, notre système à résoudre est devenu comme suit :

$$minf(x) \tag{2.41}$$

Lié a :

$$g(x) = 0 \tag{2.42}$$

$$h(x) - h_l - s_l = 0 \tag{2.43}$$

$$-h(x) + h_u - s_u = 0 \tag{2.44}$$

$$s_l, s_u \geq 0 \tag{2.45}$$

A présent, les conditions de non-négativité (2.45) sont ajoutées à la fonction objectif comme termes de barrière logarithmiques, entraînant le problème d'optimisation suivant :

$$min[f(x) - \mu(lns_u + lns_l)] \tag{2.46}$$

Lié a :

$$g(x) = 0 \tag{2.47}$$

$$h(x) - h_l - s_l = 0 \tag{2.48}$$

$$-h(x) + h_u - s_u = 0 \tag{2.49}$$

où μ est un scalaire positif appelé paramètre de barrière qui est progressivement réduit à zéro lors de la progression itératif. Notons que le théorème de Fiacco & McCormick [8], démontre que lorsque μ tend vers zéro, $x(\mu)$ approche la solution du problème x^*. Le lagrangien du problème d'optimisation avec les contraintes égalités ci-dessus s'écrit :

$$L_\mu = f(x) - \mu(lns_l + lns_u) - \lambda^T g(x) - \pi_l^T(h(x) - h_l - s_l) - \pi_u^T(-h(x) + h_u - s_u) \tag{2.50}$$

où λ est le vecteur de multiplicateur de Lagrange, π_l et π_u étant appelées variables duales.

Les conditions nécessaires d'optimalité de Karush-Kuhn-Tucker (KKT) du premier ordre sont :

$$\nabla_{s_l} L_\mu = -\mu S_l^{-1} e + \pi_l = 0 \tag{2.51}$$

$$\nabla_{s_u} L_\mu = -\mu S_u^{-1} e + \pi_u = 0 \tag{2.52}$$

$$\nabla_{\pi_l} L_\mu = -h(x) + h_l + s_l = 0 \tag{2.53}$$

$$\nabla_{\pi_u} L_\mu = h(x) - h_u + s_u = 0 \tag{2.54}$$

$$\nabla_\lambda L_\mu = -g(x) = 0 \tag{2.55}$$

$$\nabla_x L_\mu = \nabla f(x) - \nabla g(x)\lambda^T - \nabla h(x)(\pi_l^T - \pi_u^T) = 0 \tag{2.56}$$

où $e = [1, ..., 1]^T$, $S_l = diag(s_{l1}, ..., s_{lp})$, et $S_u = diag(s_{u1}, ..., s_{up})$.

Le système d'équations est donc finalement résolu avec la méthode de Newton. Le

système symétrique linéaire suivant est obtenu :

$$\begin{bmatrix} \mu S_l^{-2} & 0 & I & 0 & 0 & 0 \\ 0 & \mu S_u^{-2} & 0 & I & 0 & 0 \\ I & 0 & 0 & 0 & 0 & -\nabla h(x) \\ 0 & I & 0 & 0 & 0 & \nabla h(x) \\ 0 & 0 & 0 & 0 & 0 & -\nabla g(x) \\ 0 & 0 & -\nabla h(x)^T & \nabla h(x)^T & -\nabla g(x)^T & \nabla_x^2 L_\mu \end{bmatrix} \begin{bmatrix} \Delta s_l \\ \Delta s_u \\ \Delta \pi_l \\ \Delta \pi_u \\ \Delta \lambda \\ \Delta x \end{bmatrix} = - \begin{bmatrix} \nabla_{s_l} L_\mu \\ \nabla_{s_u} L_\mu \\ \nabla_{\pi_l} L_\mu \\ \nabla_{\pi_u} L_\mu \\ \nabla_\lambda L_\mu \\ \nabla_x L_\mu \end{bmatrix} \tag{2.57}$$

Où :

$$\nabla_x^2 L_\mu = \nabla_x^2 f(x) - \nabla_x^2 g(x)\lambda^T - \nabla_x^2 h(x)(\pi_l^T - \pi_u^T) \tag{2.58}$$

Afin de réduire la dimension du système précédent, et pour accélérer la vitesse de calcul, nous pouvons réécrire le système réduit suivant juste pour Δx et $\Delta\lambda$:

$$\begin{bmatrix} 0 & -\nabla g(x) \\ -\nabla g(x)^T & H_d \end{bmatrix} \begin{bmatrix} \Delta\lambda \\ \Delta x \end{bmatrix} = - \begin{bmatrix} \nabla_\lambda L_\mu \\ \zeta \end{bmatrix} \tag{2.59}$$

Où :

$$H_d = \nabla_x^2 L_\mu + \mu \nabla_x h(x)^T (S_l{-}2 + S_u{-}2) \nabla_x h(x) \tag{2.60}$$

et

$$\zeta = \nabla_x L_\mu + \nabla_x h(x)^T [\mu(S_u^{-2}\nabla_{\pi_u} L_\mu - S_l^{-2}\nabla_{\pi_l} L_\mu) + \nabla_{s_l} L_\mu - \nabla_{s_u} L_\mu] \tag{2.61}$$

ensuite , nous calculons :

$$\Delta s_l = \nabla h(x)\Delta x - \nabla \pi_l L\mu \qquad \Delta \pi_l = -\mu S_l^{-2}\Delta s_l - \nabla_{s_l} L_\mu \tag{2.62}$$

$$\Delta s_u = -\nabla h(x)\Delta x - \nabla \pi_u L\mu \qquad \Delta \pi_u = -\mu S_u^{-2}\Delta s_u - \nabla_{s_u} L_\mu \tag{2.63}$$

Mise à jour des variables

A la k^{ime} itération , les variables primaires et duales sont misent à jour comme suit :

$$s_l^{k+1} = s_l^k + \alpha_p^k \Delta s_l^k \qquad \pi_l^{k+1} = \pi_l^k + \alpha_d^k \Delta \pi_l^k \tag{2.64}$$

$$s_u^{k+1} = s_u^k + \alpha_p^k \Delta s_u^k \qquad \pi_u^{k+1} = \pi_u^k + \alpha_d^k \Delta \pi_u^k \tag{2.65}$$

$$X^{k+1} = X^k + \alpha_p^k \Delta X^k \qquad \lambda^{k+1} = \lambda^k + \alpha_d^k \Delta \lambda^k \tag{2.66}$$

avec $\alpha_p, \alpha_d \in (0, 1]$ pas primaires et duales. Le pas maximum doit être choisit de tel façon que les variables s_l, et s_u avec leurs variables duales soient toujours positives.

2.5.3 Réduction du paramètre de barrière μ

L'écart de complémentarité entre primaire et duales est défini comme le résidu de contraintes de complémentarité :

$$\rho^k = s_l^T \pi_l + s_u^T \pi_u. \tag{2.67}$$

Une heuristique inspirée de la programmation linéaire (LP) et de la programmation quadratique (QP) est de réduire le paramètre de barrière proportionnellement à l'écart de la complémentarité.[11]

$$\mu^{k+1} = \sigma^k \frac{\rho^k}{2(m+p)} \tag{2.68}$$

Avec : $\sigma \in [0, 1]$ paramètre assurant à la fois l'optimalité et la faisabilité.
Les valeurs extrêmes de σ (0 ou bien 1) assure soit l'optimalité ou bien la faisabilité. Une valeur initiale typique de σ est 0.4 et avec le processus itératif, la MPI atteint ces meilleurs performances lorsque $\sigma \in [0.1, 0.2]$ [11]
Le choix de μ^0 est très important pour les performances de l'algorithme. Il n'est y a pas de règle générale pour choisir ce dernier, cela dépends de la nature de problème à traiter.
En générale , μ est choisit entre $[0.01, 1000]$.

2.5.4 Critères de convergence

La convergence est atteinte et le processus itératif s'arrête dès que les expressions ci-dessous deviennent inférieures à certaines tolérances :

$$max\{max\{h_l - h(x)\}, \{h(x) - h_u\}, \| g(x) \|_\infty\} \leq \epsilon_1 \tag{2.69}$$

$$\frac{\| \nabla f(x) - \nabla g(x)\lambda^T - \nabla h(x)(\pi_l^T - \pi_u^T) \|_\infty}{1 + \| x \|_2 + \| \lambda \|_2 + \| \pi_l \|_2 + \| \pi_u \|_2} \leq \epsilon_1 \tag{2.70}$$

$$\frac{\rho}{1 + \| x \|_2} \leq \epsilon_1 \tag{2.71}$$

$$\frac{|f(x^k) - f(x^{k-1})|}{1 + |f(x^k)|} \leq \epsilon_2 \tag{2.72}$$

De plus, une tolérance supplémentaire concerne le paramètre de barrière μ tel que : $\mu \leq \epsilon_\mu$. Enpratique, les tolérances suivantes sont choisies :
$\epsilon_1 = 10^{-4}$
$\epsilon_2 = 10^{-6}$
$\epsilon_\mu = 10^{-6}$

2.5.5 Choix d'un point initiale

Un grand avantage des algorithmes de point intérieur est qu'un point réalisable de départ n'est pas nécessaire. Seules les conditions de non-négativité $s_l, s_u, \pi_l, \pi_u \geq 0$ doivent être satisfaites à chaque itération. Il est recommandé de prendre la solution x^0 de l'écoulement de puissance comme point initial de l'algorithme de point intérieur. Les variables d'écart s_l et s_u sont initialisées comme suit :

$$s_l^0 = min\{max\{\delta h^\Delta, h(x^0 - h_l\}, (1-\delta)h^\Delta\} \tag{2.73}$$

$$s_u^0 = min\{max\{\delta h^\Delta, h_l - h(x^0\}, (1-\delta)h^\Delta\} \tag{2.74}$$

tel que : $h^\Delta = h_u - h_l$ et on choisit $\delta \in [0.1, 0.3]$, ce qui donne de très bonnes résultats.
Les variables duales sont calculées par :

$$\pi_l^0 = \mu_0 S_l^{-1} e \qquad \pi_u^0 = \mu_0 S_u^{-1} e. \tag{2.75}$$

Finalement, les multiplicateurs de Lagrange relatifs au contraintes égalités (2.42) sont initialiser à 0 ($\lambda = 0$)

2.5.6 Organigramme de l'algorithme du point intérieur

Un résumé de l'algorithme du point intérieur est illustré comme suit :

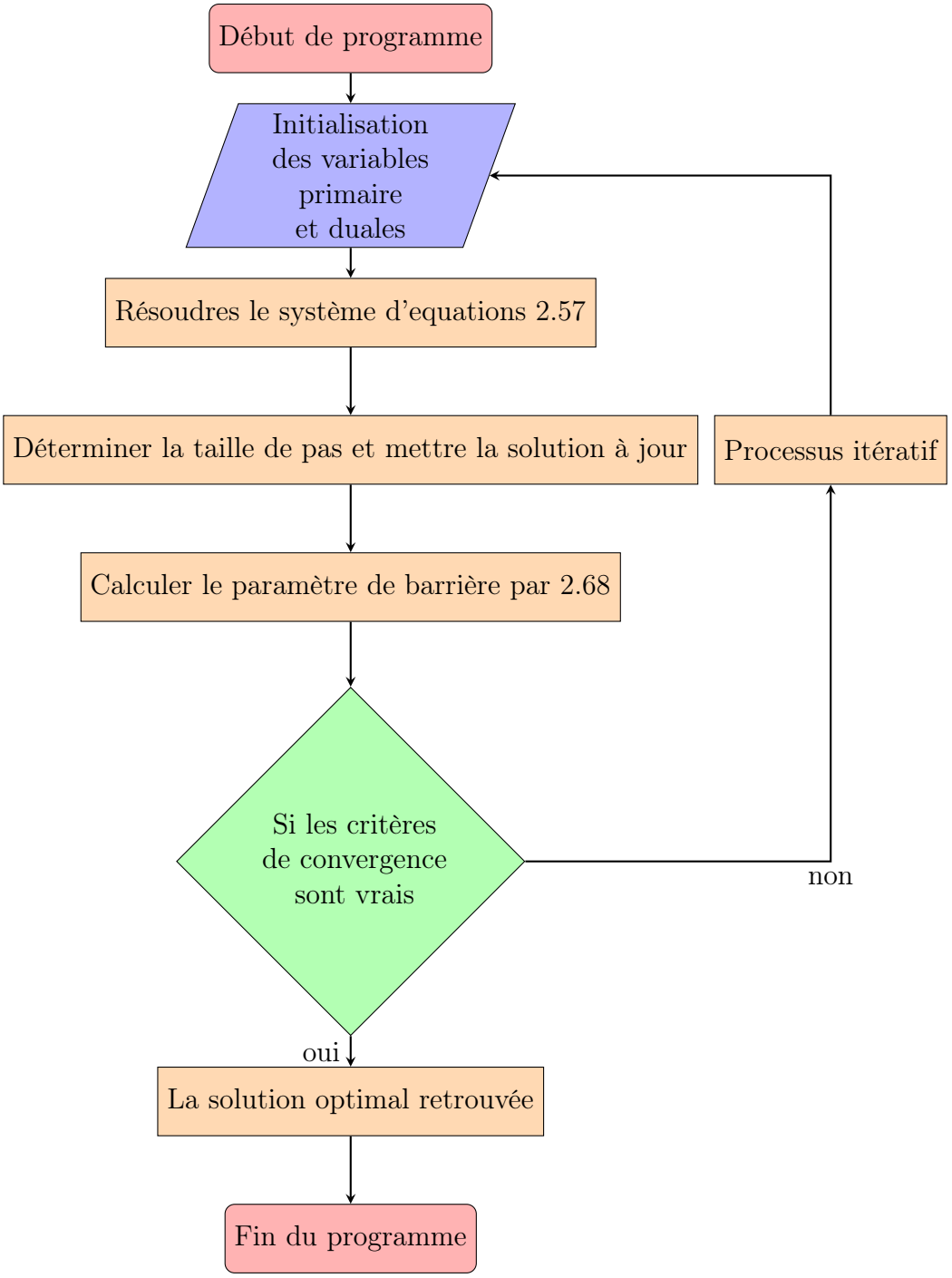

FIGURE 2.3: Organigramme de la méthode du point intérieur

2.5.7 Application de la MPI à la REP

Pour l'application de la MPI à la répartition économique de puissance, nous avons utilisé $MATPOWER$ qui est un paquet de MATLAB® M-files pour résoudre l'écoulement de puissance et l'écoulement de puissance optimal. Il est conçu comme un outil de simulation pour les chercheurs et les éducateurs qui est facile à utiliser et modifier. $MATPOWER$ est conçu pour donner la meilleure performance possible tout en gardant le code simple à comprendre et à modifier. Il a été initialement développé dans le cadre du $PowerWebproje$.[12] .

Chapitre 3

Méthode d'Optimisation par Essaim de Particules à la REP

3.1 Définition

L'optimisation par essaim de particules (Particle Swarm Optimization) est une méthode d'optimisation stochastique, pour les fonctions non-linéaires, basée sur la reproduction d'un comportement social et développée par Eberhart et Kennedy [13] en 1995.

L'origine de cette méthode vient des observations faites lors des simulations informatiques de vols groupés d'oiseaux et de bancs de poissons. Ces simulations ont mis en valeur la capacité des individus d'un groupe en mouvement à conserver une distance optimale entre eux et à suivre un mouvement global par rapport aux mouvements locaux de leur voisinage.

D'autre part, ces simulations ont également révélé l'importance du mimétisme dans la compétition qui oppose les individus à la recherche de la nourriture. En effet, les individus sont à la recherche de sources de nourriture qui sont dispersés de façon aléatoire dans un espace de recherche, et dès lors qu'un individu localise une source de nourriture, les autres individus vont alors chercher à le reproduire.

Ce comportement social basé sur l'analyse de l'environnement et du voisinage constitue alors une méthode de recherche d'optimum par l'observation des tendances des individus voisins. Chaque individu cherche à optimiser ses chances en suivant une tendance qu'il modère par ses propres vécus.

Dans notre travail, nous utiliserons un algorithme d'optimisation par essaim de particules appelé ALPSO-II (Adaptive Learning Particle Swarm Optimizer-II) ou bien optimisation par essaim de particule avec apprentissage adaptatif II [14]

Dans la méthode d'OEP, un essaim de particules vole à travers l'espace de recherche. Chaque particule suit la meilleure position précédente trouvé par ses particules voisines et la précédente meilleure position trouvée par lui-même. Chaque particule est représentée par une position et une vitesse, qui sont mis à jour comme suit :

$$X_i^{k+1} = V_i^{k+1} + X_i^k \tag{3.1}$$

$$V_i^{k+1} = \omega V_i^k + c_1 rand_1(Pbest_i^k - X_i^k) + c_2 rand_2(Gbest^k - X_i^k) \tag{3.2}$$

Où :

- X_i^k : position de l'individu i à l'itération k.
- V_i^k : vitesse de l'individu i à l'itération k.
- ω : est un facteur d'inertie qui détermine de combien la vitesse précédente est conservée.
- $rand_1, rand_2$: des nombres aléatoires entre 0 et 1.
- c_1, c_2 : sont des constantes d'accélération.
- $Pbest_i$: meilleur position de l'individu i à l'iteration k.
- $Gbest^k$: meilleure position du groupe d'individus jusqu'à l'itération k.

Deux principaux modèles d'algorithmes d'OEP existent : *gbest* (meilleure position globale) et *lbest* (meilleure locale), leur différence étant dans la manière de définir le voisinage de chaque particule. Dans le modèle de *gbest*, le voisinage d'une particule se compose des particules de tout l'essaim, qui partagent des informations entre eux. Par contre dans le modèle *lbest*, le voisinage d'une particule est défini par plusieurs particules fixes. Le modèle de *gbest* a une vitesse de convergence plus rapide avec plus de chances de rester coincé en un optimum local que *lbest* [13].

Afin d'améliorer la performance de l'OEP, nous présentons une optimisation par essaim de particule avec apprentissage adaptatif (ALPSO), qui utilise une nouvelle stratégie d'apprentissage où chaque particule peut ajuster sa stratégie de recherche selon les rapports de sélection des quatre opérateurs d'apprentissage dans différents milieux. Pour la meilleure particule, nous introduisons une méthode d'apprentissage qui peut soustraire l'information de toutes les particules améliorés.

3.2 Stratégie d'apprentissage dans ALPSO

Dans la procédure d'optimisation avec ALPSO, l'information apprise par chaque particule provient de quatre sources :

— de *gbest*
— du propre *pbest*
— du *pbest* de la particule la plus proche
— et et une position aléatoire sur lui-même (l'individu).

Les equations d'apprentissage sont les suivantes :

$$exploitation : v_k^d = \omega v_k^d + \eta.r_k^d.(pbest_k^d - x_k^d) \tag{3.3}$$

$$exploration : v_k^d = \omega v_k^d + \eta.r_k^d.(pbest_{k-nearest}^d - x_k^d) \tag{3.4}$$

$$Le\ saut x_k^d = x_k^d + v_{avg}^d.N(0,1) \tag{3.5}$$

$$convergence : V_k^d = \omega V_k^d + \eta.r_k^d.(gbest^d - X_k^d) \tag{3.6}$$

Tel que :

— $pbest_{k-nearest}$: est la *pbest* de la particule la plus proche de la particule k.
— V_{avg} : est la vitesse moyenne de tout les particules.
— $N(0,1)$: est un nombre aléatoire d'une distribution normal avec une moyenne 0 et une variance 1.

L'apprentissage à partir du voisin le plus proche permet à une particule d'explorer la région des optimum locaux autour d'elle. Les particules qui sont près d'un optimum local se rapprocheront de plus en plus de cette région parce que le *pbest* est remplacé uniquement lorsque une meilleure position est trouvée. Cette stratégie peut aider l'essaim à trouver plusieurs optimums locaux plutôt qu'un seul comme le PSO basique, et surtout pour les problèmes complexes.

Une fois que les particules convergent vers un optimum local ou bien, s'il y a une région plus prometteuse à proximité sans particules recouvrantes, les particules doivent avoir une probabilité pour accéder à cette région prometteuse. Ainsi, l'apprentissage d'une position aléatoire autour de lui même est nécessaire. Dans l'ALPSO, chaque particule a quatre choix différents pour ajuster son comportement. Les quatre choix permettent à chaque particule de se déplacer vers une position prometteuse avec une probabilité plus élevée que l'EOP de base.

3.2.1 Mécanisme d'apprentissage adaptatif

Dans notre travail, nous introduisons une structure adaptative utilisant les quatre opérateurs d'apprentissage précités, chacun affecté d'un rapport de sélection.

Ces opérateurs d'apprentissage jouent les rôles de convergence, d'exploitation, d'exploration et de saut sur les bassins d'attraction des optimums locaux.

Afin de permettre aux particules de choisir automatiquement un opérateur d'apprentissage approprié au cours du processus de la recherche, un mécanisme de sélection adaptative, basé sur l'hypothèse de l'opérateur le plus utilisé avec succès dans les récentes dernières itérations, peut être aussi efficaces dans les itérations suivantes. Pour chaque particule, l'un des quatre opérateurs d'apprentissage est sélectionné en fonction de leur rapports de sélection. L'opérateur qui résulte à la meilleure performance relative aura son rapport de sélection augmenté. L'opérateur le plus approprié sera choisi automatiquement pour une particule et va contrôler le comportement de recherche de la particule en fonction de son paysage de fitness locale au stade évolutif correspondant.

Pour toutes les particules, le rapport de sélection de chaque opérateur est également initialisé à 1/4 (à l'exception de la particule *gbest*, dans lequel le rapport de sélection initiale est 1/3) et est mis à jour en fonction de sa performance relative.

Les rapports de sélection des opérateurs d'une particule sont mis à jour s'ils ne s'améliorent pas pour une fréquence de mise à jour U_f pendant des itérations successives. Au cours de la période de mise à jour pour chaque particule, la valeur de progression et la valeur de la récompense de l'opérateur i sont calculés comme suit.
La valeur de progression $p_i^k(t)$ de l'opérateur i pour la particules k à l'itération t est défini comme :

$$p_i^k(t) = \begin{cases} |f(\overrightarrow{x}_k(t)) - f(\overrightarrow{x}_k(t-1))| & si \;\; l'operateur\ i\ est\ choisit\ par\ \overrightarrow{x}_k(t) \\ & et\ ce\ dernier\ est\ mieux\ que\ \overrightarrow{x}_k(t-1) \\ 0 & sinon. \end{cases} \tag{3.7}$$

et l'expression de la valeur de récompense $r_i^k(t)$ est :

$$r_i^k(t) = \frac{p_i^k(t)}{\sum_{j=1}^{M} p_j^k(t)}\alpha + \frac{g_i^k}{G_i^k}(1-\alpha) + c_i^k s_i^k(t). \tag{3.8}$$

La valeur de récompense $r_i^k(t)$comporte trois éléments, qui sont la valeur de progression normalisée, le taux de réussite, et le rapport de sélection précédent avec :

- g_i^k : compteur qui enregistre le nombre de fois d'apprentissage réussis de la particule k, où son enfant est plus en forme que particule k en appliquant l'opérateur i depuis la dernière mise à jour du rapport de sélection
- G_i^k : nombre total d'itérations où l'opérateur i est sélectionné par la particule k depuis la dernier mise à jour du rapport de sélection
- $\frac{g_i^k}{G_i^k}$: taux de réussite de l'opérateur i pour la particule k
- α : nombre aléatoire entre 0 est 1
- M : nombre d'opérateurs
- c_i^k : facteur de pénalité de l'opérateur i pour la particule k
- et s_i^k : rapport de sélection de l'opérateur i pour la particule k à l'itération courante.

Le rapport de sélection de l'opérateur i de la particule k à l'itération suivante $t+1$ est écrit comme suit :

$$s_i^k(t+1) = \frac{r_i^k(t)}{\sum_{j=1}^{M} r_j^k(t)}(1 - M * \gamma) + \gamma. \tag{3.9}$$

avec γ rapport de sélection minimum pour chaque opérateur affecté à la valeur 0.01. Pour la particule *gbest* dans ALPSO, elle sera mise à jour tant qu'une particule va mieux au fil du temps par l'extraction d'informations de cette particule améliorée.

Nous observons deux paramètres clés dans l'ALPSO : la fréquence de mise à jour (U_f) et la probabilité d'apprentissage (P_l). Les valeurs de U_f et P_l affectent de manière significative la performance de l'ALPSO. Cette dernière (ALPSO) introduit quelques méthodes pour choisir les valeurs optimales des deux paramètres. Pour le paramètre de la fréquence de mise à jour (U_f), chaque particule est affectée à une valeur de U_f au lieu d'utiliser une même valeur toute les particules. La valeur de U_f est définie pour chaque particule k comme suit :

$$U_f^k = max(10 * exp(-(1.6.k/N)^4, 1) \tag{3.10}$$

avec N : la taille de population et U_f^k est la fréquence de mise à jour de la particule K. Même chose pour le paramètre P_l, chaque particule K a son propre paramètre. P_l est écrit comme suit :

$$P_l^k = max(1 - exp(-(1.6.k/N)^4, 0.05) \tag{3.11}$$

Pour ajuster la valeur de P_l d'une façon adaptative, ALPSO a besoin de calculer le rapport d'amélioration de particule définit par :

$$IMPR_k(t) = max(\frac{f(\overrightarrow{x_k}(t-1)) - f(\overrightarrow{x_k}(t))}{f(\overrightarrow{x_k}(t-1))}, 0) \tag{3.12}$$

avec $IMPR_k$ est le rapport d'amélioration de particule k entre les deux itérations $t-1$, et t.

3.3 ALPSO-II

Dans cette partie, nous nous penchons en détails sur la technique ALPSO-II. D'abord, deux opérateurs d'apprentissage de l'ALPSO sont remplacés par deux nouveaux opérateurs d'apprentissage. Deuxièmement, un mécanisme de contrôle est introduit pour sur- veiller l'état des particules. Enfin, une approche visant à contrôler le nombre de particules qui apprennent à partir de la meilleure position globale (nommée position de abest) est ajouté dans ALPSO-II. Le but de toutes ces améliorations est d'augmenter la diversité de sorte que ALPSO- II peut chercher beaucoup plus de meilleures solutions dans le paysage complexe de remise en forme.

3.3.1 Opérateurs d'apprentissage dans ALPSO-II

Dans ALPSO-II, nous utilisons également quatre opérateurs d'apprentissage, mais l'opérateur "d'apprentissage de la *pbest* de la particule la plus proche" (opérateur d'exploration) est remplacer par "un opérateur d'apprentissage de la *pbest* d'une particule aléatoire ", et chaque particules apprend à partir d'une position archive de *gbest*, (*abest*) à la place de l'apprentissage à partir de la particule *gbest*. Les deux opérateurs d'apprentissage mises à jour sont définis comme suit :
Opérateur c' : apprentissage à partir de *pbest* d'une particule aléatoire.

$$exploration : v_k^d = \omega v_k^d + \eta.r_k^d.(pbest_{rand}^d - x_k^d) \tag{3.13}$$

Opérateur d' : apprentissage à partir de la position *abest*

$$convergence : v_k^d = \omega v_k^d + \eta.r_k^d.(abest^d - x_k^d) \tag{3.14}$$

où la position de abest est utilisée pour sauvegarder la meilleure position trouvée par ALPSO-II jusque là. Dans ALPSO-II, une seule particule apprend d'une position de $pbest_{rand}$ une position meilleure que sa propre meilleure pbest de position historique. Grâce à cette stratégie, plus de ressources de calcul sont données aux particules les moins performantes pour améliorer tout l'essaim.

En introduisant le nouvel opérateur d'exploration, ALPSO-II permet à une particule d'explorer le paysage non recherché remis en forme avec une probabilité supérieure à celle d'ALPSO où cette particule va apprendre d'une particule aléatoire au lieu de son voisinage le plus proche [15]

3.3.2 Surveillance de l'état des particules

D'une manière générale, la réinitialisation est une méthode pour augmenter la diversité de la population dans les algorithmes évolutionnaires. Cependant, il y a des situations où en utilisant cette méthode, la protection des individus réinitialisés n'est pas assurée et ces individus sont éliminés. Nous disposons de plusieurs façons de vérifier le moment où nous devons effectuer une réinitialisation.

La première méthode est de vérifier la diversité de la population. Si la diversité est inférieure à un seuil, nous effectuons une réinitialisation. La seconde méthode consiste à surveiller la particule *gbest*, si elle ne s'améliore pour un certain nombre d'itérations, la réinitialisation peut être lancée. Pour les algorithmes E O P , nous pouvons surveiller la vitesse des particules. Si l'amplitude de la vitesse d'une particule est inférieure à une valeur de seuil, on peut réinitialiser cette particule.

Quelle que soit la méthode que nous utilisons, nous devons définir une valeur de seuil pour effectuer cette opération. Cependant, il est très difficile d'obtenir une valeur de seuil optimale pour un problème particulier. En outre, les valeurs de seuil pour les différents problèmes peuvent être différentes.

Le problème commun des approches ci-dessus, c'est qu'elles ne peuvent pas savoir si une particule est à l'étape de l'évolution ou à l'étape de convergence. Si cette particule converge, nous pouvons effectuer une réinitialisation. Afin de surveiller l'état de particules, nous introduisons un mécanisme pour vérifier si une particule est à l'état de convergence.

Dans ALPSO-II, il existe un mécanisme de contrôle de la performance des quatre i opérateurs d'apprentissage. L'approche consiste à surveiller les rapports des quatre opérateurs d'apprentissage de sélection. Une fois qu'une particule converge vers un optimum local, et qu'aucun des quatre opérateurs ne peut l'aider à sauter hors de ce dernier, leur rapport de sélection va revenir à l'étape initiale où ils ont des valeurs égales à 1/4. Par conséquent, nous pouvons utiliser cette information pour examiner si une particule a convergé ou pas. En utilisant cette approche, nous pouvons facilement éviter les problèmes cités précédemment pour la réinitialisation des particules. Pour atteindre cet objectif, en calculant les rapports de sélection normale comme dans ALPSO, nous avons besoin de créer un rapport de sélection de surveillance pour chaque opérateur d'apprentissage. Dans ALPSO-II, toute définition, fonctionnement de calcul, et mise à jour des rapports de sélection de suivi sont les mêmes que dans ALPSO pour calculer et mettre à à jour les rapports de sélection normale, sauf le calcul de $p_i^{'k}$ valeur de progression de l'opérateur i pour particule k à l'itération t, qui est défini comme suit :

$$p_i^{'k}(t) = \begin{cases} |f(\overrightarrow{x}_k(t)) - f(\overrightarrow{x}_k^{pbest})| & \textit{si l'operateur } i \textit{ est choisit par } \overrightarrow{x}_k(t) \\ & \textit{et ce dernier est mieux que } \overrightarrow{x_k}^{pbest} \\ 0 & \textit{sinon.} \end{cases} \tag{3.15}$$

Pour distinguer les définitions relatives à la mise à jour des rapports de sélection de contrôle dans les deux algorithmes différents, nous mettons un symbole prime après chaque définition, par exemple, $p'_k(t)$ et $p_k(t)$ représentent la valeur de la surveillance de progrès et de la valeur de progrès commune de l'opérateur i pour la particule k à l'itération t, respectivement.

Dans ALPSO-II, les rapports de sélection communs et les rapports de sélection de surveillance sont mis à jour en même temps et une fois qu'ils sont mis à jour, tous les paramètres de composants sont remis à l'état initial : les valeurs de progrès, les valeurs de récompenses, les taux de réussite sont tous misent à 0.

La ré-initialisation d'une particule est effectuée une fois la variance des rapports de sélection de contrôle est inférieur à $0,05$.

3.3.3 Contrôle du nombre de particules qui apprennent de la position de abest

Dans l'algorithme d'ALPSO, tout en accomplissant la recherche locale pour une particule dépend de la performance des opérateurs de recherche locaux (par exemple l'opérateur de l'exploitation et l'opérateur d'exploration), il y a encore une chance de faire une très bonne recherche globale.

Comme nous le savons, les particules, qui sont loin de la position de *abest*, ne peuvent pas obtenir des avantages en apprenant de lui surtout pour les problèmes multi-modales. Dans ALPSO-II, pour faire équilibrer la recherche globale et la recherche locale, nous permettons seulement un certain nombre de particules (Q), qui sont à proximité de la position de *abest*, à apprendre de ce dernier.

En fait, ALPSO-II permet juste aux Q particules d'utiliser les quatre opérateurs d'apprentissage et les autres particules n'utilisent pas l'opérateur de la convergence.

Chapitre 4

Simulations et discussion de résultats

Ce présente chapitre représente la partie simulation de ce mémoire. Trois cas d'études ont été réalisés pour solutionner notre problème et ce par utilisation d'une méthode heuristique que nous avons présenté en détail dans le troisième chapitre à savoir : l'algorithme d'optimisation par essaims de particules (ALPSO), et trois méthodes conventionnelles : méthode de Newton, méthode du gradient et méthode dU point intérieur détaillés au deuxième chapitre.

Notre étude consiste à trouver la solution de la répartition économique de puissance pour trois réseaux standard (9 nœuds A,30 nœuds B, 57 nœuds C) et de comparer les différentes résultats.

4.1 Analyse des résultats obtenus par la méthode du gradient

La méthode du gradient a été appliquée sur le système 9 nœuds qui contient trois nœuds de générations et six nœuds de charge.

TABLE 4.1: Valeurs de P_g optimal calculés par la méthode de gradient

Numéro de nœud	Valeurs initiales de P_g(MW)	P_g après OPF(MW)	autres valeurs de P_g(MW)	P_g après OPF (MW)
1	71.95	88.11	100	88.115
2	163	136.38	100	136.38
3	85	95.45	119.95	95.45
P_g total (MW)	le coût initiale ($/h)	le coût optimisé ($/h)	le coût initiale ($/h)	le coût optimisé ($/h)
319.95	5438.32	5336.004	5537.5	5336.004

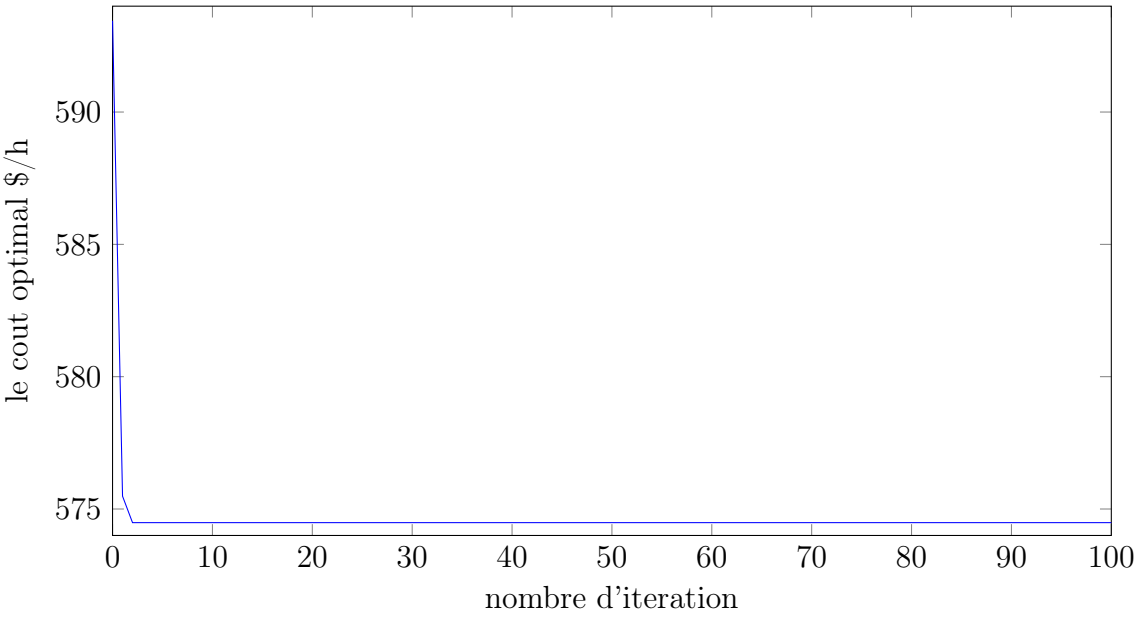

FIGURE 4.1: Méthode de gradient

Le tableau 4.1 représente les valeurs de Pg optimal pour deux valeurs initiales différentes. On remarque bien d'après les résultats dans le tableau que les valeurs de P_g optimal ne dépends pas des valeurs initiales. Dans la 3^{me} colonne du tableau, les valeurs initiales sont différentes de celles de la 1^{re} colonne et on a obtenu les même résultats pour Pg optimal. Nous remarquons aussi une diminution du coût de production, avec un profit de $102,319\$/h$.

TABLE 4.2: Différentes valeurs de P_g en chaque itération

Numéro de nœud	P_g du 1^{iter} (MW)	P_g du 2^{iter} (MW)	P_g du 100^{iter} (MW)
1	88.11	88.11	88.11
2	136.38	136.38	136.38
3	95.45	95.45	95.45
Le coût optimisé ($/h)	5336.004	5336.004	5336.004

Le tableau 4.2 représente les valeurs de Pg optimal durant l'exécution de processus d'optimisation. Nous remarquons que les valeurs optimisées restent les mêmes après la 1^{ere} itération. Donc, le processus converge après une seule itération.

4.2 Analyse des résultats obtenus par la méthode de Newton

Ensuite, nous avons appliqué la méthode de Newton sur le même système 9 nœuds. Les résultats obtenus sont illustrés au tableau 4.3

TABLE 4.3: Valeurs de P_g optimal calculés par la méthode de Newton

Numéro de nœud	Valeurs initiales de P_g(MW)	P_g après OPF(MW)	autres valeurs de P_g(MW)	P_g après OPF (MW)
1	71.95	90.74	100	90.752
2	163	134.69	100	134.691
3	85	94.51	119.95	94.511
P_g total (MW)	le coût initiale ($/h)	le coût optimisé ($/h)	le coût initiale ($/h)	le coût optimisé ($/h)
319.95	5438.32	5316.84	5537.5	5316.90

Le tableau 4.3 représente les valeurs de Pg optimal pour deux valeurs initiales différentes. Nous remarquons d'après le tableau que l'optimisation par la méthode de newton ne dépend pas des valeurs initiales comme celle du gradient.

Pour mieux comprendre le comportement de la méthode de newton pour l'EPO pour des différents systèmes, nous appliquons cette dernière sur un réseau de trente nœuds (30 bus). On obtient les résultats suivantes :

TABLE 4.4: Résultats obtenus par la méthode de newton dans un système 30-bus

Numéro de nœuds	P_g du 1^{iter} (MW)	P_g du 2^{iter} (MW)	P_g du 18^{iter} (MW)	P_g du 100^{iter} (MW)
1	26.2	30.6	42.85	42.85
2	60.83	58.43	57.24	57.24
13	36.38	28.46	22.56	22.56
22	21.53	21.59	35.43	35.43
23	19.23	19.39	16.46	16.46
27	27.44	33.16	17.08	17.08
Le coût optimisé ($/h)	592.58	582.84	574.69	574.69

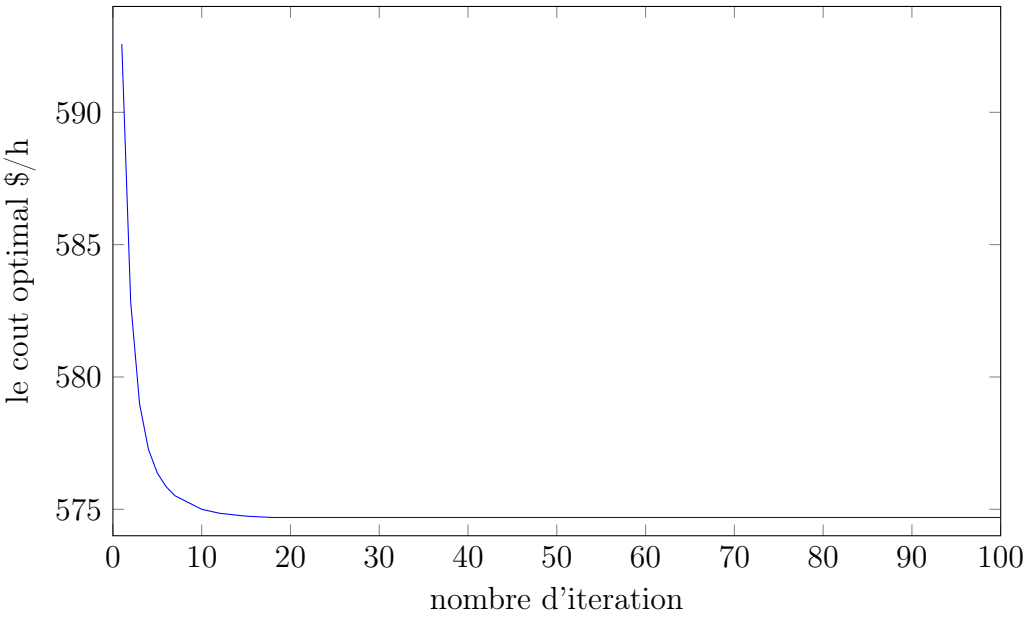

FIGURE 4.2: Méthode de Newton

D'après le tableau, nous voyons que la méthode de newton converge après 18 itérations. Si on compare avec le système (9-bus), on voie bien que le nombre d'itération est augmenté. Donc, le nombre d'itération nécessaire pour que la méthode de newton converge est augmenté suivant la taille de réseau.

Nous remarquons aussi que pour les nœuds 23 et 27 sont deux nœuds de générations qui ont la même fonction coût mais des valeurs de sorties de P_g différentes. Cela est du au fait que les deux centrales ne sont pas dans la même position. Alors , la centrale la plus proche à la charge donne des valeurs de P_g plus que la centrale la plus éloignée. Après le calcul d'EPO sur ce système (30-bus), on réduit le coût de production d'une valeur de : 18.75 $\$/h$.

4.3 Analyse des résultats par la méthode MPI

Le tableau suivant illustre les resultats obtenu par la MPI en utilisant le logiciel MATPAOWER [12] sur un reseau standard de 30 nœuds. Les résultat sont dans l'annexe (D). Nous remarquons que cette méthode converge très vite après quelque itérations. La MPI est la méthode qui donne le temps de calcul le plus petit parmis les méthodes conventionnelles étudiées.

4.4 Analyse des résultats par la méthode ALPSO-II

Le tableau suivant nous montre les résultats obtenus par la méthode d'ALPSO-II sur le réseau 30-bus.

TABLE 4.5: Les valeurs de Pg obtenu par ALPSO-II

Numéro de nœud	P_g (MW)
1	44.79
2	58.30
13	22.31
22	32.31
23	15.86
27	15.81
Pg total(MW)	189.38
Le cout ($\$/h$)	565.97

La puissance totale générée est : 189.38 MW et la puissance demandée est : 189.20 MW. Par conséquence, les pertes sont 0.2 MW. En comparaison, la puissance générée totale par la méthode de Newton est égale 191.62MW. Donc, on conclut que ALPSO-II réduit les pertes mieux que la méthode de newton.

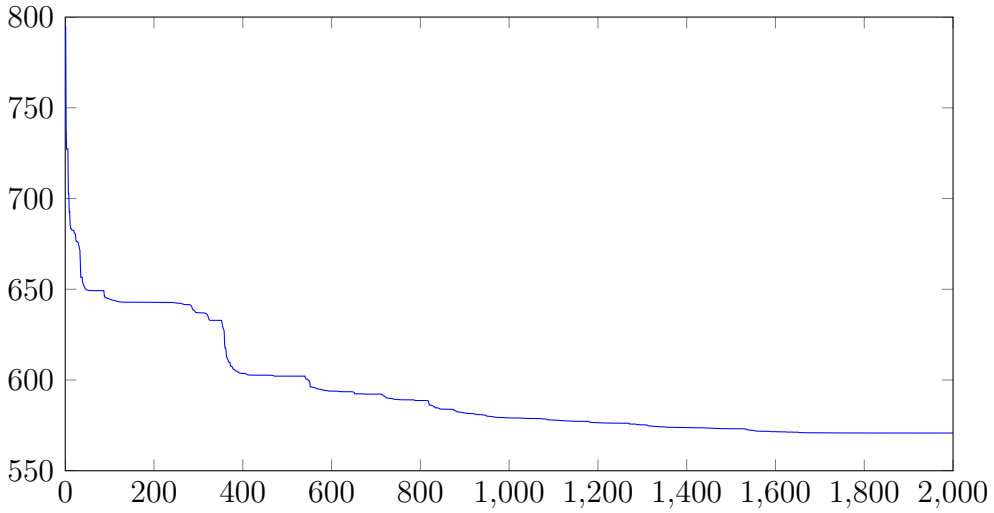

FIGURE 4.3: Méthode ALPSO

4.5 Comparaison entre les résultats obtenus par les différentes méthodes appliquées pour le réseau 30 nœuds

Nous comparons les résultats obtenus avec tout les méthodes qu'on a utiliser dans notre travail. Le tableau suivant illustre ces résultats.

TABLE 4.6: Comparaison entres les différentes méthodes

Numéro de nœud	Méthode de gradient	Méthode de Newton	Méthode de point intérieur	Méthode de ALPSO
1	45.108	44.833	41.54	44.79
2	58.695	58.154	55.40	58.30
13	16.087	22.475	16.20	22.31
22	22.435	33.470	22.74	32.31
23	16.087	15.896	16.27	15.86
27	33.23	16.4267	39.91	15.81
Le coût optimisé	574.48	573.02	576.89	565.97
Profit ($\$/h$)	18.9677	20.42	16.56	27.47

Le coût obtenu par la méthode ALPSO-II est visiblement meilleur que ceux des trois autres méthodes mais l'ALPSO est un peu lente à cause du nombre d'itérations élevé. La méthode du gradient présente une convergence rapide vers l'optimum.

Afin de confirmer nos résultats, le tableau suivant présente les résultats obtenus pour le système 57 nœuds pour les différentes méthodes.

TABLE 4.7: Les résultats obtenus pour le réseaux 57 nœuds C

Numéro de nœuds	Méthode de Newton	Méthode de gradient	Méthode de point intérieur	Méthode de ALPSO
1	163	140.37	142.63	139.22
2	100	89.02	87.81	73.43
3	45.65	43.56	45.07	43.09
6	100	89.02	72.89	100
8	480.35	490.06	459.82	482.07
9	58.09	89.02	97.55	100
12	331.55	337.59	361.54	333.10
Le cout optimal ($/h)	42225.01	42168.9	41837.79	41753
Profit ($/h)	9123.20	9176.29	9495.2	9610.4

On remarque que le meilleur profit est celui de la méthode ALPSO. Si on calcule le profit généré pendant une année, nous remarquons que c'est considérable. L'algorithme ALPSO a fait une très bonne optimisation qui permet de gagner 84.18 millions de dollars par années.

Conclusion

Ce mémoire nous a permis de traiter le problème d'écoulement de puissance optimal qui est un des problèmes les plus en vue dans le domaine du fonctionnement des réseaux électriques, surtout que les besoins en énergie électrique augmentent continuellement avec les besoins socio-économiques croissants dans toutes les sociétés du monde. Ce mémoire a essayé d'être une courte rétrospective de méthodes ayant eu à résoudre ce problème, que ce soit des méthodes conventionnelles basées sur des techniques analytiques de fonctions issues de la modélisation des systèmes d'énergie de puissance ou de méthodes non conventionnelles basées principalement sur des techniques d'intelligence artificielle. Notons que nous nous sommes concentrés sur le cas de la répartition économique, qui est un des problèmes les plus importants de l'écoulement de puissance optimal.

Nous avons ainsi porté notre choix sur trois méthodes conventionnelles : méthode du gradient, méthode de Newton et méthode du point intérieur, que nous avons détaillé, programmé et simulé. Notre étude nous a permis, de constater que ces méthodes convergent généralement vite, mais deviennent difficiles à utiliser lorsque le problème physique devient :

- fortement non linéaire (charges, régulation, autres dispositifs).
- la fonction à optimiser n'est pas différentiable.
- la fonction à optimiser comporte plusieurs objectifs simultanés (optimisation multi-objectifs).

Cette difficulté se traduit souvent par :

- leurs convergences vers des optimums locaux .
- difficulté majeure liée à leur programmation et leur mise en œuvre.

Pour les méthodes d'intelligence artificielles, nous nous sommes intéressés à la technique d'optimisation par essaim de particules avec un paramètre d'adaptation

pour améliorer l'efficacité de la technique de base. Cette technique a donné de meilleurs résultats que les méthodes conventionnelles mais le temps de calcul est un peu élevé.

Nous proposons comme éventuelles perspectives :

— l'amélioration et la diminution du temps de calcule des méthodes intelligentes
— combiner entre les méthodes conventionnelles et non conventionnelles par le développement d'une sorte d'algorithmes hybrides

Références

[1] M. JA, E.-H. ME, and A. R, "A review of selected optimal power flow literature to 1993,part i :nonlinear quadratic programming approach," *IEEE trans Power Sys*, pp. 96–104, 1999.

[2] H. Atun and T.yalcinoz, "Implementing soft computing techniques to solve economic dispatch problem in power systems," *xpert Systems with Applications*, vol. 35, 2008.

[3] O. Guenounou, *Méthodologie de conception de contrôleurs intelligents par l'approche génétique- application à un ioprocédé.* PhD thesis, Université Toulouse III, Paul Sabatier.

[4] M.Basu, "Economic enviremental dispatch using multiobjective differential evolution," *Applied soft Computing*, vol. 11, pp. 2845–2853, 2011.

[5] M. Clerc and J. Kennedy, "The particle swarm : Explosion, stability and convergence in a multi-dimensional complex space," *IEEE Transactions on Evolutionary Computation*, vol. 6, 2001.

[6] *Optimal Power Flow Problem & Solution Methodologies.*

[7] B. H and L. M, "Projet de fin d'etude : Analyse de base des methodes d'ecoulement de puissance optimal et applications," *Ecole Nationale Polytechnique d'Alger*, 1993.

[8] A. Fiacco and G. McCormick, *Nonlinear Programming : Sequential Unconstrained Minimization Techniques.* John Willey & Sons, 1968.

[9] Karmarkar, "A new polynomial-time algorithm for linear programming," *Combinatorica*, vol. 4, pp. 373–395, 1984.

[10] S. Mehrotra, "On the implementation of a primal-dual interior point method," *SIAM Journal on Optimization*, vol. 2, pp. 575–601, 1992.

[11] S. Wright, "Primal-dual interior-point methods," *SIAM*, 1997.

[12] R. D. Zimmerman, C. E. Murillo-Sánchez, and R. J. Thomas, "Matpower : Steady-state operations, planning and analysis tools for power systems research and education," *Power Systems, IEEE Transactions*, vol. 26, pp. 12–19, Feb 2011.

[13] Y. S. Russell C. Eberhart and J. Kennedy, *Swarm Intelligence*. The Morgan Kaufmann Series in Artificial Intelligence, San Francisco, CA, USA : Morgan Kaufmann, 2001.

[14] C. Li and S. Yang, "Adaptive learning particle swarm optimizer-ii for global optimization," *Congress on Evolutionary Computation (CEC), IEEE*, pp. 1–8, 2010.

[15] C. Li and S. Yang, "An adaptive learning particle swarm optimizer for function optimization," *Proc. of the 2009 IEEE Congress on Evolutionary Computation*, pp. 381–388, 2009.

Annexe A

Système 9 nœuds

Ci dessous les données du système 30 nœuds

```
function mpc = case9
%CASE9    Power flow data for 9 bus, 3 generator case.
%   Please see CASEFORMAT for details on the case file format.
%
%   Based on data from Joe H. Chow's book, p. 70.

%   MATPOWER
%   $Id: case9.m 2408 2014-10-22 20:41:33Z ray $

%% MATPOWER Case Format : Version 2
mpc.version = '2';

%%-----  Power Flow Data  -----%%
%% system MVA base
mpc.baseMVA = 100;

%% bus data
% bus_i type Pd Qd Gs Bs area Vm Va baseKV zone Vmax Vmin
mpc.bus = [
1 3 0 0 0 0 1 1 0 345 1 1.1 0.9;
2 2 0 0 0 0 1 1 0 345 1 1.1 0.9;
3 2 0 0 0 0 1 1 0 345 1 1.1 0.9;
```

```
4 1 0 0 0 0 1 1 0 345 1 1.1 0.9;
5 1 90 30 0 0 1 1 0 345 1 1.1 0.9;
6 1 0 0 0 0 1 1 0 345 1 1.1 0.9;
7 1 100 35 0 0 1 1 0 345 1 1.1 0.9;
8 1 0 0 0 0 1 1 0 345 1 1.1 0.9;
9 1 125 50 0 0 1 1 0 345 1 1.1 0.9;
];

%% generator data
% bus Pg Qg Qmax Qmin Vg mBase status Pmax Pmin Pc1 Pc2 Qc1min Qc1max
Qc2min Qc2max ramp_agc ramp_10 ramp_30 ramp_q apf
mpc.gen = [
1 0 0 300 -300 1 100 1 250 10 0 0 0 0 0 0 0 0 0 0 0;
2 163 0 300 -300 1 100 1 300 10 0 0 0 0 0 0 0 0 0 0 0;
3 85 0 300 -300 1 100 1 270 10 0 0 0 0 0 0 0 0 0 0 0;
];

%% branch data
% fbus tbus r x b rateA rateB rateC ratio angle status angmin angmax
mpc.branch = [
1 4 0 0.0576 0 250 250 250 0 0 1 -360 360;
4 5 0.017 0.092 0.158 250 250 250 0 0 1 -360 360;
5 6 0.039 0.17 0.358 150 150 150 0 0 1 -360 360;
3 6 0 0.0586 0 300 300 300 0 0 1 -360 360;
6 7 0.0119 0.1008 0.209 150 150 150 0 0 1 -360 360;
7 8 0.0085 0.072 0.149 250 250 250 0 0 1 -360 360;
8 2 0 0.0625 0 250 250 250 0 0 1 -360 360;
8 9 0.032 0.161 0.306 250 250 250 0 0 1 -360 360;
9 4 0.01 0.085 0.176 250 250 250 0 0 1 -360 360;
];

%%-----  OPF Data  -----%%
%% generator cost data
% 1 startup shutdown n x1 y1 ... xn yn
% 2 startup shutdown n c(n-1) ... c0
mpc.gencost = [
```

```
2 1500 0 3 0.11 5 150;
2 2000 0 3 0.085 1.2 600;
2 3000 0 3 0.1225 1 335;
];
```

Annexe B

Système 30 nœuds

```
function mpc = case30
%CASE30    Power flow data for 30 bus, 6 generator case.
%   Please see CASEFORMAT for details on the case file format.
%
%   Based on data from ...
%     Alsac, O. & Stott, B., "Optimal Load Flow with Steady State Security",
%     IEEE Transactions on Power Apparatus and Systems, Vol. PAS 93, No. 3,
%     1974, pp. 745-751.
%   ... with branch parameters rounded to nearest 0.01, shunt values divided
%   by 100 and shunt on bus 10 moved to bus 5, load at bus 5 zeroed out.
%   Generator locations, costs and limits and bus areas were taken from ...
%     Ferrero, R.W., Shahidehpour, S.M., Ramesh, V.C., "Transaction analysis
%     in deregulated power systems using game theory", IEEE Transactions on
%     Power Systems, Vol. 12, No. 3, Aug 1997, pp. 1340-1347.
%   Generator Q limits were derived from Alsac & Stott, using their Pmax
%   capacities. V limits and line |S| limits taken from Alsac & Stott.

%   MATPOWER
%   $Id: case30.m 2408 2014-10-22 20:41:33Z ray $

%% MATPOWER Case Format : Version 2
mpc.version = '2';

%%-----  Power Flow Data  -----%%
```

```
%% system MVA base
mpc.baseMVA = 100;

%% bus data
% bus_i type Pd Qd Gs Bs area Vm Va baseKV zone Vmax Vmin
mpc.bus = [
1 3 0 0 0 0 1 1 0 135 1 1.05 0.95;
2 2 21.7 12.7 0 0 1 1 0 135 1 1.1 0.95;
3 1 2.4 1.2 0 0 1 1 0 135 1 1.05 0.95;
4 1 7.6 1.6 0 0 1 1 0 135 1 1.05 0.95;
5 1 0 0 0 0.19 1 1 0 135 1 1.05 0.95;
6 1 0 0 0 0 1 1 0 135 1 1.05 0.95;
7 1 22.8 10.9 0 0 1 1 0 135 1 1.05 0.95;
8 1 30 30 0 0 1 1 0 135 1 1.05 0.95;
9 1 0 0 0 0 1 1 0 135 1 1.05 0.95;
10 1 5.8 2 0 0 3 1 0 135 1 1.05 0.95;
11 1 0 0 0 0 1 1 0 135 1 1.05 0.95;
12 1 11.2 7.5 0 0 2 1 0 135 1 1.05 0.95;
13 2 0 0 0 0 2 1 0 135 1 1.1 0.95;
14 1 6.2 1.6 0 0 2 1 0 135 1 1.05 0.95;
15 1 8.2 2.5 0 0 2 1 0 135 1 1.05 0.95;
16 1 3.5 1.8 0 0 2 1 0 135 1 1.05 0.95;
17 1 9 5.8 0 0 2 1 0 135 1 1.05 0.95;
18 1 3.2 0.9 0 0 2 1 0 135 1 1.05 0.95;
19 1 9.5 3.4 0 0 2 1 0 135 1 1.05 0.95;
20 1 2.2 0.7 0 0 2 1 0 135 1 1.05 0.95;
21 1 17.5 11.2 0 0 3 1 0 135 1 1.05 0.95;
22 2 0 0 0 0 3 1 0 135 1 1.1 0.95;
23 2 3.2 1.6 0 0 2 1 0 135 1 1.1 0.95;
24 1 8.7 6.7 0 0.04 3 1 0 135 1 1.05 0.95;
25 1 0 0 0 0 3 1 0 135 1 1.05 0.95;
26 1 3.5 2.3 0 0 3 1 0 135 1 1.05 0.95;
27 2 0 0 0 0 3 1 0 135 1 1.1 0.95;
28 1 0 0 0 0 1 1 0 135 1 1.05 0.95;
29 1 2.4 0.9 0 0 3 1 0 135 1 1.05 0.95;
30 1 10.6 1.9 0 0 3 1 0 135 1 1.05 0.95;
```

```
];

%% generator data
% bus Pg Qg Qmax Qmin Vg mBase status Pmax Pmin Pc1 Pc2 Qc1min Qc1max
Qc2min Qc2max ramp_agc ramp_10 ramp_30 ramp_q apf
mpc.gen = [
1 23.54 0 150 -20 1 100 1 80 0 0 0 0 0 0 0 0 0 0 0 0;
2 60.97 0 60 -20 1 100 1 80 0 0 0 0 0 0 0 0 0 0 0 0;
22 21.59 0 62.5 -15 1 100 1 50 0 0 0 0 0 0 0 0 0 0 0 0;
27 26.91 0 48.7 -15 1 100 1 55 0 0 0 0 0 0 0 0 0 0 0 0;
23 19.2 0 40 -10 1 100 1 30 0 0 0 0 0 0 0 0 0 0 0 0;
13 37 0 44.7 -15 1 100 1 40 0 0 0 0 0 0 0 0 0 0 0 0;
];

%% branch data
% fbus tbus r x b rateA rateB rateC ratio angle status angmin angmax
mpc.branch = [
1 2 0.02 0.06 0.03 130 130 130 0 0 1 -360 360;
1 3 0.05 0.19 0.02 130 130 130 0 0 1 -360 360;
2 4 0.06 0.17 0.02 65 65 65 0 0 1 -360 360;
3 4 0.01 0.04 0 130 130 130 0 0 1 -360 360;
2 5 0.05 0.2 0.02 130 130 130 0 0 1 -360 360;
2 6 0.06 0.18 0.02 65 65 65 0 0 1 -360 360;
4 6 0.01 0.04 0 90 90 90 0 0 1 -360 360;
5 7 0.05 0.12 0.01 70 70 70 0 0 1 -360 360;
6 7 0.03 0.08 0.01 130 130 130 0 0 1 -360 360;
6 8 0.01 0.04 0 32 32 32 0 0 1 -360 360;
6 9 0 0.21 0 65 65 65 0 0 1 -360 360;
6 10 0 0.56 0 32 32 32 0 0 1 -360 360;
9 11 0 0.21 0 65 65 65 0 0 1 -360 360;
9 10 0 0.11 0 65 65 65 0 0 1 -360 360;
4 12 0 0.26 0 65 65 65 0 0 1 -360 360;
12 13 0 0.14 0 65 65 65 0 0 1 -360 360;
12 14 0.12 0.26 0 32 32 32 0 0 1 -360 360;
12 15 0.07 0.13 0 32 32 32 0 0 1 -360 360;
12 16 0.09 0.2 0 32 32 32 0 0 1 -360 360;
```

```
14 15 0.22 0.2 0 16 16 16 0 0 1 -360 360;
16 17 0.08 0.19 0 16 16 16 0 0 1 -360 360;
15 18 0.11 0.22 0 16 16 16 0 0 1 -360 360;
18 19 0.06 0.13 0 16 16 16 0 0 1 -360 360;
19 20 0.03 0.07 0 32 32 32 0 0 1 -360 360;
10 20 0.09 0.21 0 32 32 32 0 0 1 -360 360;
10 17 0.03 0.08 0 32 32 32 0 0 1 -360 360;
10 21 0.03 0.07 0 32 32 32 0 0 1 -360 360;
10 22 0.07 0.15 0 32 32 32 0 0 1 -360 360;
21 22 0.01 0.02 0 32 32 32 0 0 1 -360 360;
15 23 0.1 0.2 0 16 16 16 0 0 1 -360 360;
22 24 0.12 0.18 0 16 16 16 0 0 1 -360 360;
23 24 0.13 0.27 0 16 16 16 0 0 1 -360 360;
24 25 0.19 0.33 0 16 16 16 0 0 1 -360 360;
25 26 0.25 0.38 0 16 16 16 0 0 1 -360 360;
25 27 0.11 0.21 0 16 16 16 0 0 1 -360 360;
28 27 0 0.4 0 65 65 65 0 0 1 -360 360;
27 29 0.22 0.42 0 16 16 16 0 0 1 -360 360;
27 30 0.32 0.6 0 16 16 16 0 0 1 -360 360;
29 30 0.24 0.45 0 16 16 16 0 0 1 -360 360;
8 28 0.06 0.2 0.02 32 32 32 0 0 1 -360 360;
6 28 0.02 0.06 0.01 32 32 32 0 0 1 -360 360;
];

%%-----  OPF Data  -----%%
%% generator cost data
% 1 startup shutdown n x1 y1 ... xn yn
% 2 startup shutdown n c(n-1) ... c0
mpc.gencost = [
2 0 0 3 0.02 2 0;
2 0 0 3 0.0175 1.75 0;
2 0 0 3 0.0625 1 0;
2 0 0 3 0.00834 3.25 0;
2 0 0 3 0.025 3 0;
2 0 0 3 0.025 3 0;
];
```

Annexe C

Système 57 nœuds

```
function mpc = case57
%CASE57    Power flow data for IEEE 57 bus test case.
%   Please see CASEFORMAT for details on the case file format.
%   This data was converted from IEEE Common Data Format
%   (ieee57cdf.txt) on 15-Oct-2014 by cdf2matp, rev. 2393
%   See end of file for warnings generated during conversion.
%
%   Converted from IEEE CDF file from:
%       http://www.ee.washington.edu/research/pstca/
%
%   Manually modified Qmax, Qmin on generator 1 to 200, -140, respectively.
%
%  08/25/93 UW ARCHIVE           100.0  1961 W IEEE 57 Bus Test Case

%   MATPOWER
%   $Id: case57.m 2394 2014-10-15 20:39:39Z ray $

%% MATPOWER Case Format : Version 2
mpc.version = '2';

%%-----  Power Flow Data  -----%%
%% system MVA base
mpc.baseMVA = 100;
```

```
%% bus data
% bus_i type Pd Qd Gs Bs area Vm Va baseKV zone Vmax Vmin
mpc.bus = [
1 3 55 17 0 0 1 1.04 0 0 1 1.06 0.94;
2 2 3 88 0 0 1 1.01 -1.18 0 1 1.06 0.94;
3 2 41 21 0 0 1 0.985 -5.97 0 1 1.06 0.94;
4 1 0 0 0 0 1 0.981 -7.32 0 1 1.06 0.94;
5 1 13 4 0 0 1 0.976 -8.52 0 1 1.06 0.94;
6 2 75 2 0 0 1 0.98 -8.65 0 1 1.06 0.94;
7 1 0 0 0 0 1 0.984 -7.58 0 1 1.06 0.94;
8 2 150 22 0 0 1 1.005 -4.45 0 1 1.06 0.94;
9 2 121 26 0 0 1 0.98 -9.56 0 1 1.06 0.94;
10 1 5 2 0 0 1 0.986 -11.43 0 1 1.06 0.94;
11 1 0 0 0 0 1 0.974 -10.17 0 1 1.06 0.94;
12 2 377 24 0 0 1 1.015 -10.46 0 1 1.06 0.94;
13 1 18 2.3 0 0 1 0.979 -9.79 0 1 1.06 0.94;
14 1 10.5 5.3 0 0 1 0.97 -9.33 0 1 1.06 0.94;
15 1 22 5 0 0 1 0.988 -7.18 0 1 1.06 0.94;
16 1 43 3 0 0 1 1.013 -8.85 0 1 1.06 0.94;
17 1 42 8 0 0 1 1.017 -5.39 0 1 1.06 0.94;
18 1 27.2 9.8 0 10 1 1.001 -11.71 0 1 1.06 0.94;
19 1 3.3 0.6 0 0 1 0.97 -13.2 0 1 1.06 0.94;
20 1 2.3 1 0 0 1 0.964 -13.41 0 1 1.06 0.94;
21 1 0 0 0 0 1 1.008 -12.89 0 1 1.06 0.94;
22 1 0 0 0 0 1 1.01 -12.84 0 1 1.06 0.94;
23 1 6.3 2.1 0 0 1 1.008 -12.91 0 1 1.06 0.94;
24 1 0 0 0 0 1 0.999 -13.25 0 1 1.06 0.94;
25 1 6.3 3.2 0 5.9 1 0.982 -18.13 0 1 1.06 0.94;
26 1 0 0 0 0 1 0.959 -12.95 0 1 1.06 0.94;
27 1 9.3 0.5 0 0 1 0.982 -11.48 0 1 1.06 0.94;
28 1 4.6 2.3 0 0 1 0.997 -10.45 0 1 1.06 0.94;
29 1 17 2.6 0 0 1 1.01 -9.75 0 1 1.06 0.94;
30 1 3.6 1.8 0 0 1 0.962 -18.68 0 1 1.06 0.94;
31 1 5.8 2.9 0 0 1 0.936 -19.34 0 1 1.06 0.94;
32 1 1.6 0.8 0 0 1 0.949 -18.46 0 1 1.06 0.94;
33 1 3.8 1.9 0 0 1 0.947 -18.5 0 1 1.06 0.94;
```

```
34 1 0 0 0 0 1 0.959 -14.1 0 1 1.06 0.94;
35 1 6 3 0 0 1 0.966 -13.86 0 1 1.06 0.94;
36 1 0 0 0 0 1 0.976 -13.59 0 1 1.06 0.94;
37 1 0 0 0 0 1 0.985 -13.41 0 1 1.06 0.94;
38 1 14 7 0 0 1 1.013 -12.71 0 1 1.06 0.94;
39 1 0 0 0 0 1 0.983 -13.46 0 1 1.06 0.94;
40 1 0 0 0 0 1 0.973 -13.62 0 1 1.06 0.94;
41 1 6.3 3 0 0 1 0.996 -14.05 0 1 1.06 0.94;
42 1 7.1 4.4 0 0 1 0.966 -15.5 0 1 1.06 0.94;
43 1 2 1 0 0 1 1.01 -11.33 0 1 1.06 0.94;
44 1 12 1.8 0 0 1 1.017 -11.86 0 1 1.06 0.94;
45 1 0 0 0 0 1 1.036 -9.25 0 1 1.06 0.94;
46 1 0 0 0 0 1 1.05 -11.89 0 1 1.06 0.94;
47 1 29.7 11.6 0 0 1 1.033 -12.49 0 1 1.06 0.94;
48 1 0 0 0 0 1 1.027 -12.59 0 1 1.06 0.94;
49 1 18 8.5 0 0 1 1.036 -12.92 0 1 1.06 0.94;
50 1 21 10.5 0 0 1 1.023 -13.39 0 1 1.06 0.94;
51 1 18 5.3 0 0 1 1.052 -12.52 0 1 1.06 0.94;
52 1 4.9 2.2 0 0 1 0.98 -11.47 0 1 1.06 0.94;
53 1 20 10 0 6.3 1 0.971 -12.23 0 1 1.06 0.94;
54 1 4.1 1.4 0 0 1 0.996 -11.69 0 1 1.06 0.94;
55 1 6.8 3.4 0 0 1 1.031 -10.78 0 1 1.06 0.94;
56 1 7.6 2.2 0 0 1 0.968 -16.04 0 1 1.06 0.94;
57 1 6.7 2 0 0 1 0.965 -16.56 0 1 1.06 0.94;
];

%% generator data
% bus Pg Qg Qmax Qmin Vg mBase status Pmax Pmin Pc1 Pc2 Qc1min Qc1max
Qc2min Qc2max ramp_agc ramp_10 ramp_30 ramp_q apf
mpc.gen = [
1 128.9 -16.1 200 -140 1.04 100 1 575.88 0 0 0 0 0 0 0 0 0 0 0 0;
2 0 -0.8 50 -17 1.01 100 1 100 0 0 0 0 0 0 0 0 0 0 0 0;
3 40 -1 60 -10 0.985 100 1 140 0 0 0 0 0 0 0 0 0 0 0 0;
6 0 0.8 25 -8 0.98 100 1 100 0 0 0 0 0 0 0 0 0 0 0 0;
8 450 62.1 200 -140 1.005 100 1 550 0 0 0 0 0 0 0 0 0 0 0 0;
9 0 2.2 9 -3 0.98 100 1 100 0 0 0 0 0 0 0 0 0 0 0 0;
```

```
12 310 128.5 155 -150 1.015 100 1 410 0 0 0 0 0 0 0 0 0 0 0 0;
];

%% branch data
% fbus tbus r x b rateA rateB rateC ratio angle status angmin angmax
mpc.branch = [
1 2 0.0083 0.028 0.129 0 0 0 0 0 1 -360 360;
2 3 0.0298 0.085 0.0818 0 0 0 0 0 1 -360 360;
3 4 0.0112 0.0366 0.038 0 0 0 0 0 1 -360 360;
4 5 0.0625 0.132 0.0258 0 0 0 0 0 1 -360 360;
4 6 0.043 0.148 0.0348 0 0 0 0 0 1 -360 360;
6 7 0.02 0.102 0.0276 0 0 0 0 0 1 -360 360;
6 8 0.0339 0.173 0.047 0 0 0 0 0 1 -360 360;
8 9 0.0099 0.0505 0.0548 0 0 0 0 0 1 -360 360;
9 10 0.0369 0.1679 0.044 0 0 0 0 0 1 -360 360;
9 11 0.0258 0.0848 0.0218 0 0 0 0 0 1 -360 360;
9 12 0.0648 0.295 0.0772 0 0 0 0 0 1 -360 360;
9 13 0.0481 0.158 0.0406 0 0 0 0 0 1 -360 360;
13 14 0.0132 0.0434 0.011 0 0 0 0 0 1 -360 360;
13 15 0.0269 0.0869 0.023 0 0 0 0 0 1 -360 360;
1 15 0.0178 0.091 0.0988 0 0 0 0 0 1 -360 360;
1 16 0.0454 0.206 0.0546 0 0 0 0 0 1 -360 360;
1 17 0.0238 0.108 0.0286 0 0 0 0 0 1 -360 360;
3 15 0.0162 0.053 0.0544 0 0 0 0 0 1 -360 360;
4 18 0 0.555 0 0 0 0 0.97 0 1 -360 360;
4 18 0 0.43 0 0 0 0 0.978 0 1 -360 360;
5 6 0.0302 0.0641 0.0124 0 0 0 0 0 1 -360 360;
7 8 0.0139 0.0712 0.0194 0 0 0 0 0 1 -360 360;
10 12 0.0277 0.1262 0.0328 0 0 0 0 0 1 -360 360;
11 13 0.0223 0.0732 0.0188 0 0 0 0 0 1 -360 360;
12 13 0.0178 0.058 0.0604 0 0 0 0 0 1 -360 360;
12 16 0.018 0.0813 0.0216 0 0 0 0 0 1 -360 360;
12 17 0.0397 0.179 0.0476 0 0 0 0 0 1 -360 360;
14 15 0.0171 0.0547 0.0148 0 0 0 0 0 1 -360 360;
18 19 0.461 0.685 0 0 0 0 0 0 1 -360 360;
19 20 0.283 0.434 0 0 0 0 0 0 1 -360 360;
```

```
21 20 0 0.7767 0 0 0 0 1.043 0 1 -360 360;
21 22 0.0736 0.117 0 0 0 0 0 0 1 -360 360;
22 23 0.0099 0.0152 0 0 0 0 0 0 1 -360 360;
23 24 0.166 0.256 0.0084 0 0 0 0 0 1 -360 360;
24 25 0 1.182 0 0 0 0 1 0 1 -360 360;
24 25 0 1.23 0 0 0 0 1 0 1 -360 360;
24 26 0 0.0473 0 0 0 0 1.043 0 1 -360 360;
26 27 0.165 0.254 0 0 0 0 0 0 1 -360 360;
27 28 0.0618 0.0954 0 0 0 0 0 0 1 -360 360;
28 29 0.0418 0.0587 0 0 0 0 0 0 1 -360 360;
7 29 0 0.0648 0 0 0 0 0.967 0 1 -360 360;
25 30 0.135 0.202 0 0 0 0 0 0 1 -360 360;
30 31 0.326 0.497 0 0 0 0 0 0 1 -360 360;
31 32 0.507 0.755 0 0 0 0 0 0 1 -360 360;
32 33 0.0392 0.036 0 0 0 0 0 0 1 -360 360;
34 32 0 0.953 0 0 0 0 0.975 0 1 -360 360;
34 35 0.052 0.078 0.0032 0 0 0 0 0 1 -360 360;
35 36 0.043 0.0537 0.0016 0 0 0 0 0 1 -360 360;
36 37 0.029 0.0366 0 0 0 0 0 0 1 -360 360;
37 38 0.0651 0.1009 0.002 0 0 0 0 0 1 -360 360;
37 39 0.0239 0.0379 0 0 0 0 0 0 1 -360 360;
36 40 0.03 0.0466 0 0 0 0 0 0 1 -360 360;
22 38 0.0192 0.0295 0 0 0 0 0 0 1 -360 360;
11 41 0 0.749 0 0 0 0 0.955 0 1 -360 360;
41 42 0.207 0.352 0 0 0 0 0 0 1 -360 360;
41 43 0 0.412 0 0 0 0 0 0 1 -360 360;
38 44 0.0289 0.0585 0.002 0 0 0 0 0 1 -360 360;
15 45 0 0.1042 0 0 0 0 0.955 0 1 -360 360;
14 46 0 0.0735 0 0 0 0 0.9 0 1 -360 360;
46 47 0.023 0.068 0.0032 0 0 0 0 0 1 -360 360;
47 48 0.0182 0.0233 0 0 0 0 0 0 1 -360 360;
48 49 0.0834 0.129 0.0048 0 0 0 0 0 1 -360 360;
49 50 0.0801 0.128 0 0 0 0 0 0 1 -360 360;
50 51 0.1386 0.22 0 0 0 0 0 0 1 -360 360;
10 51 0 0.0712 0 0 0 0 0.93 0 1 -360 360;
13 49 0 0.191 0 0 0 0 0.895 0 1 -360 360;
```

```
29 52 0.1442 0.187 0 0 0 0 0 0 1 -360 360;
52 53 0.0762 0.0984 0 0 0 0 0 0 1 -360 360;
53 54 0.1878 0.232 0 0 0 0 0 0 1 -360 360;
54 55 0.1732 0.2265 0 0 0 0 0 0 1 -360 360;
11 43 0 0.153 0 0 0 0 0.958 0 1 -360 360;
44 45 0.0624 0.1242 0.004 0 0 0 0 0 1 -360 360;
40 56 0 1.195 0 0 0 0 0.958 0 1 -360 360;
56 41 0.553 0.549 0 0 0 0 0 0 1 -360 360;
56 42 0.2125 0.354 0 0 0 0 0 0 1 -360 360;
39 57 0 1.355 0 0 0 0 0.98 0 1 -360 360;
57 56 0.174 0.26 0 0 0 0 0 0 1 -360 360;
38 49 0.115 0.177 0.003 0 0 0 0 0 1 -360 360;
38 48 0.0312 0.0482 0 0 0 0 0 0 1 -360 360;
9 55 0 0.1205 0 0 0 0 0.94 0 1 -360 360;
];

%%-----  OPF Data  -----%%
%% generator cost data
% 1 startup shutdown n x1 y1 ... xn yn
% 2 startup shutdown n c(n-1) ... c0
mpc.gencost = [
2 0 0 3 0.077579519 20 0;
2 0 0 3 0.01 40 0;
2 0 0 3 0.25 20 0;
2 0 0 3 0.01 40 0;
2 0 0 3 0.0222222222 20 0;
2 0 0 3 0.01 40 0;
2 0 0 3 0.0322580645 20 0;
];
```

Annexe D

Résultats obtenus avec le logiciel MATPOWER

```
MATPOWER Version 5.1, 20-Mar-2015 -- AC Optimal Power Flow
MATLAB Interior Point Solver -- MIPS, Version 1.2, 20-Mar-2015
 (using built-in linear solver)
Converged!

Converged in 1.12 seconds
Objective Function Value = 576.89 $/hr
================================================================================
|     System Summary
================================================================================

How many?                How much?              P (MW)            Q (MVAr)
---------------------    -------------------  -------------  -----------------
Buses             30     Total Gen Capacity     335.0         -95.0 to 405.9
Generators         6     On-line Capacity       335.0         -95.0 to 405.9
Committed Gens     6     Generation (actual)    192.1             105.1
Loads             20     Load                   189.2             107.2
  Fixed           20       Fixed                189.2             107.2
  Dispatchable     0       Dispatchable          -0.0 of -0.0       -0.0
Shunts             2     Shunt (inj)            -0.0               0.2
Branches          41     Losses (I^2 * Z)        2.86             13.33
Transformers       0     Branch Charging (inj)     -              15.2
```

Inter-ties	7	Total Inter-tie Flow	51.0 58.1
Areas	3		

	Minimum	Maximum
Voltage Magnitude	0.961 p.u. @ bus 8	1.069 p.u. @ bus 27
Voltage Angle	-5.69 deg @ bus 19	0.00 deg @ bus 1
P Losses (I^2*R)	-	0.30 MW @ line 2-6
Q Losses (I^2*X)	-	2.39 MVAr @ line 28-27
Lambda P	3.66 $/MWh @ bus 1	5.38 $/MWh @ bus 8
Lambda Q	-0.06 $/MWh @ bus 29	1.40 $/MWh @ bus 8

Bus	Voltage		Generation		Load		Lambda($/MVA-hr)	
#	Mag(pu)	Ang(deg)	P (MW)	Q (MVAr)	P (MW)	Q (MVAr)	P	Q
1	0.982	0.000*	41.54	-5.44	-	-	3.662	-
2	0.979	-0.763	55.40	1.67	21.70	12.70	3.689	-
3	0.977	-2.390	-	-	2.40	1.20	3.754	-0.016
4	0.976	-2.839	-	-	7.60	1.60	3.771	-0.021
5	0.971	-2.486	-	-	-	-	3.744	-0.001
6	0.972	-3.229	-	-	-	-	3.779	-0.020
7	0.962	-3.491	-	-	22.80	10.90	3.801	0.003
8	0.961	-3.682	-	-	30.00	30.00	5.383	1.405
9	0.990	-4.137	-	-	-	-	3.823	0.020
10	1.000	-4.600	-	-	5.80	2.00	3.846	0.039
11	0.990	-4.137	-	-	-	-	3.823	0.020
12	1.017	-4.498	-	-	11.20	7.50	3.810	-
13	1.064	-3.298	16.20	35.93	-	-	3.810	-
14	1.007	-5.040	-	-	6.20	1.60	3.868	0.018
15	1.009	-4.814	-	-	8.20	2.50	3.856	0.018
16	1.003	-4.839	-	-	3.50	1.80	3.849	0.031
17	0.995	-4.887	-	-	9.00	5.80	3.862	0.047
18	0.993	-5.484	-	-	3.20	0.90	3.911	0.047

19	0.987	-5.688	-	-	9.50	3.40	3.926	0.058
20	0.990	-5.472	-	-	2.20	0.70	3.910	0.055
21	1.009	-4.621	-	-	17.50	11.20	3.854	0.017
22	1.016	-4.503	22.74	34.20	-	-	3.843	-
23	1.026	-3.756	16.27	6.96	3.20	1.60	3.813	-
24	1.017	-3.885	-	-	8.70	6.70	3.884	0.028
25	1.044	-2.072	-	-	-	-	3.932	0.022
26	1.027	-2.476	-	-	3.50	2.30	3.999	0.067
27	1.069	-0.715	39.91	31.75	-	-	3.916	-
28	0.982	-3.215	-	-	-	-	4.106	0.250
29	1.050	-1.849	-	-	2.40	0.90	3.966	-0.059
30	1.039	-2.643	-	-	10.60	1.90	4.051	-0.012
			--------	--------	--------	--------		
		Total:	192.06	105.08	189.20	107.20		

Branch Data								
Brnch	From	To	From Bus Injection		To Bus Injection		Loss (I^2 * Z)	
#	Bus	Bus	P (MW)	Q (MVAr)	P (MW)	Q (MVAr)	P (MW)	Q (MVAr)
1	1	2	21.04	-2.34	-20.95	-0.27	0.092	0.28
2	1	3	20.50	-3.10	-20.28	2.02	0.220	0.84
3	2	4	18.63	-5.85	-18.40	4.60	0.232	0.66
4	3	4	17.88	-3.22	-17.84	3.36	0.035	0.14
5	2	5	14.36	-0.69	-14.25	-0.78	0.108	0.43
6	2	6	21.66	-4.21	-21.36	3.21	0.300	0.90
7	4	6	17.58	5.68	-17.54	-5.54	0.036	0.14
8	5	7	14.25	0.96	-14.15	-1.64	0.109	0.26
9	6	7	8.70	8.46	-8.65	-9.26	0.049	0.13
10	6	8	23.82	21.37	-23.71	-20.93	0.108	0.43
11	6	9	7.27	-8.27	-7.27	8.54	0.000	0.27
12	6	10	4.15	-4.73	-4.15	4.96	0.000	0.23
13	9	11	0.00	-0.00	0.00	-0.00	0.000	0.00
14	9	10	7.27	-8.54	-7.27	8.68	0.000	0.14
15	4	12	11.06	-15.24	-11.06	16.21	0.000	0.97

16	12	13	-16.20	-34.01	16.20	35.93	0.000	1.92
17	12	14	4.68	2.08	-4.65	-2.01	0.030	0.07
18	12	15	6.07	3.18	-6.04	-3.12	0.032	0.06
19	12	16	5.31	5.04	-5.26	-4.94	0.047	0.10
20	14	15	-1.55	0.41	1.55	-0.41	0.006	0.01
21	16	17	1.76	3.14	-1.75	-3.12	0.010	0.02
22	15	18	7.20	3.75	-7.13	-3.60	0.071	0.14
23	18	19	3.93	2.70	-3.92	-2.67	0.014	0.03
24	19	20	-5.58	-0.73	5.59	0.75	0.010	0.02
25	10	20	7.85	1.58	-7.79	-1.45	0.058	0.13
26	10	17	7.27	2.73	-7.25	-2.68	0.018	0.05
27	10	21	-4.43	-11.56	4.47	11.67	0.046	0.11
28	10	22	-5.06	-8.39	5.13	8.54	0.067	0.14
29	21	22	-21.97	-22.87	22.07	23.07	0.099	0.20
30	15	23	-10.92	-2.72	11.04	2.97	0.124	0.25
31	22	24	-4.46	2.59	4.49	-2.54	0.031	0.05
32	23	24	2.03	2.39	-2.01	-2.37	0.012	0.03
33	24	25	-11.18	-1.75	11.41	2.16	0.235	0.41
34	25	26	3.54	2.36	-3.50	-2.30	0.042	0.06
35	25	27	-14.96	-4.52	15.20	4.99	0.246	0.47
36	28	27	-11.45	-21.09	11.45	23.48	0.000	2.39
37	27	29	6.16	1.65	-6.08	-1.50	0.078	0.15
38	27	30	7.10	1.63	-6.95	-1.36	0.149	0.28
39	29	30	3.68	0.60	-3.65	-0.54	0.030	0.06
40	8	28	-6.29	-9.07	6.36	7.41	0.069	0.23
41	6	28	-5.05	-14.50	5.09	13.68	0.047	0.14
							--------	--------
						Total:	2.860	13.33