République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
UNIVERSITE BLIDA 1

FACULTE DES SCIENCES
DEPARTEMENT DE MATHEMATIQUES

MEMOIRE DE MASTER

En Mathématiques

Spécialité : **Recherche Opérationnelle**

Thème

Construction des matrices d'expériences optimales D par les algorithmes d'échange, de déplacement et d'intelligence artificielle

Présenté par :

BOUFRIDI Amina

BOUTEBAL Abderrahim

Soutenu le 03/10/2013 devant le jury composé de :

Président : Mr TAMI. O
Examinateur : Mr HANNANE. F
Promoteur : Mr EL MOSSAOUI. H

ملخص

ان طريقة المنهج التجريبي هي مجموعة من الطرق و وسائل للتحليل يستعملها الباحثون من اجل تخطيط تجاربهم. الهدف من الطريقة السماح للباحثين في تجويد فعالية تجاربهم مهما كان ميدان تخصصهم كما يساعد علي تحليل المشكل لطرح استراتيجية بحتة ذات جودة عالية بدلالة الأهداف المسطرة و الوسائل المتوفرة لهذه المنهجية.

لهذا العمل هدفان :

- دراسة خوارزميات التبادل و خوارزميات عن طريق الانتقال من اجل إيجاد أحسن مصفوفة و مقارنة المصفوفات لتوجيه اختيار المجرب.

- دراسة جديدة في خوارزمية Donev التي تتمثل في بعض التغييرات مع تحديد سلبياتها و ايجابياتها و مقارنتها مع خوارزمية Donev الأصلية. زيادة على ذلك اقتراح جديد الذي يعتمد على خوارزمية مستعمرة النمل و منها اقتراح مراحل تشكيل هذه التالية لتطبيقها في المنهج التجريبي.

Résumé

La méthodologie de la recherche expérimentale (MRE) est un ensemble de méthodes et de modes de raisonnement destinés à tout expérimentateur désirant faire de la planification expérimentale. Elle a pour objet de lui permettre d'optimiser l'efficacité de sa recherche expérimentale quelle que soit sa branche d'activité. Pour ce faire, elle va l'aider à exprimer aux mieux son problème et lui proposer des stratégies expérimentales optimales en fonction des objectifs qu'il s'est fixé et des moyens dont il dispose.

Ce travail comporte deux objectifs:

- Les algorithmes qui consistent à trouver la meilleure matrice selon le critère D-optimal et il existe deux catégories, les algorithmes d'échanges et ceux par les déplacements, puis on fait une étude comparative sur les deux catégories afin d'aider l'expérimentateur à prendre des décisions.

- Une nouvelle modification sur l'algorithme de Donev tout en précisant ses avantages et ses inconvénients par rapport à l'algorithme de Donev. Une nouvelle proposition qui dépend de l'algorithme de colonie de fourmis dans les plans D-optimaux tout en donnant des étapes à suivre pour construire un tel algorithme.

Abstract

The methodology of experimental research (MRE) is a set of methods and fashion reasoning for any experimenter wishing to make the planning experimental planification. Its purpose is to allow him to optimize effectiveness of its experimental research regardless of its branch activity. To do this, it will help to better express his problem and propose optimal experimental strategies based on objectives that It is fixed and mean it has.

This work has two objectives:

- The algorithms are to find the best matrix according to the D-criterion optimal and there are two categories of algorithms by those exchanges and investments, then a comparative study of two categories to help the experimenter to make decisions.

- A new modification of Donev algorithm while stating its advantages and disadvantages compared to Donev algorithm. A new proposal which depends on ant colony algorithm in the D-optimal designs while giving steps to construct an algorithm.

Remerciement

On remercie tout d'abord notre vénéré **Allah** le tout puissant à qui nous devons tout.

Après avoir passé ces années aussi bénéfiques qu'enrichissantes, il est juste de reconnaitre tous les efforts et l'assistance que nous avons reçu de la part de tous le staff du département de **mathématiques** qui ne cesse de continuer de préserver la fierté de **l'université de Blida** dans son propre domaine.

De ce fait, nous exprimons toute notre gratitude à notre promoteur Monsieur **H. El MOSSAOUI** pour l'effort fourni, les conseils prodigués, sa patience et sa persévérance dans le suivi.

Nous remercions aussi très sincèrement notre président du jury Monsieur **O. TAMI** sans oublier notre examinateur Monsieur **F. HANNANE** pour nous avoir aidés à accomplir ce travail dans les meilleures conditions.

Nous adressons également notre remerciement à tous nos **enseignants** qui nous ont donné les bases des maths.

En fin on tient aussi à remercier ceux qui ont contribué de près et de loin à l'élaboration de ce travail.

Dédicace

Je dédie humblement ce manuscrit à:

Celle qui s'est toujours dévouée et sacrifiée pour moi, celle qui m'a aidée du mieux qu'elle pouvait pour réussir , celle qui m'a accompagnée tout au long de ce parcours, celle qui a toujours été là dans mes moments de détresse , ma très chère **mère**.

Celui qui m'a toujours encouragée et soutenue moralement, mon très cher **père**.

Celle qui m'a toujours ouvert ses bras et soutenue dans tout ce que j'ai entrepris, celle qui a su être bonne, gentille et compréhensive avec moi, ma très chère et adorée **grand-mère**.

Ma très chère sœur **Amel** et mon très cher frère **Billel** ainsi qu'à sa femme **Meriem** qui m'ont énormément aidée et à qui je témoigne mon affection et ma profonde reconnaissance.

Toute ma famille paternelle **BOUFRIDI** et maternelle **ZEMMOUR**.

Mes très chers **cousins** et **cousines** ainsi qu'a ma nièce **Dyna** que j'aime beaucoup.

Mes très chers **amis** que j'apprécie tellement qui ont été toujours à mon côté pour m'encourager et me donner la force pour continuer.

Mes chers professeurs **Mr Hannane** et monsieur **El moussaoui,** ainsi qu'à **Melle Ouezri Leila** qui m'ont toujours encouragés et soutenus depuis le début de mon travail ; ceux qui ont toujours su trouver les mots pour me redonner la force de continuer et d'aller au bout de cette aventure qu'est le mémoire! !

Mon **binôme** et à toute sa famille.

Amina

Dédicace

Je dédie ce mémoire à:

Mon cher **père** et ma chère **mère** qui ont éclairé mon chemin et qui m'ont encouragé et soutenu tout au long de mes études.

Mes **frères** et leurs **femmes**.

Mes **sœurs** et leurs **maris**.

Toute ma famille paternelle **BOUTEBAL** et maternelle **MOUSSOUNI**.

Mes **cousines** et mes **cousins** que j'apprécie beaucoup ainsi que mes **neveux** et mes **nièces**.

Tous mes ami(e)s surtout **Hakim** et **Karima**.

A mon **binôme** et toute sa famille.

Rahim

Table des matiéres

Principaux notions et symboles

k	Nombre des facteurs.
N	Nombre d'essais d'un plan d'expériences.
n	Nombre de points souhaités.
p	Nombre de coefficients d'un modèle mathématique.
χ	Domaine expérimental d'intérêt.
E	Ensemble des matrices d'expériences.
ξ	Matrice d'expériences.
ξ_N	Matrice d'expériences à N expériences.
X	Matrice des effets.
y	Vecteur des réponses mesurées.
$\hat{y}$	Vecteur des réponses calculées.
a_0	Constant d'un modèle.
a_i	Coefficients des termes de premier degré.
a_{ii}	Coefficients des termes carrés.
a_{ij}	Coefficients des termes rectangles.
$\hat{a}_i$	Coefficients de la variable x_i dans un modèle déterminé par la méthode des moindres carrés.
a	Vecteur de tous les coefficients réels du modèle.
$\hat{a}$	Vecteur de tous les coefficients déterminés avec l'hypothèse des moindres carrés.
e	Vecteur des résidus.
(X^tX)	Matrice d'information.
$(X^tX)^{-1}$	Matrice de dispersion.
$E(\hat{a})$	Espérance.
σ	Ecart-type.

t	Variable codée.
T	Variable naturelle.
T_0	Le milieu de l'intervalle du domaine d'étude.
ΔT	La moitié de la largeur du domaine d'étude.
σ^2	Variance.
$N\left(0,\sigma^2\right)$	Loi normale.
$\hat{\sigma}^2$	Variance de regression.
$\hat{\sigma}_u^2$	Variance de regression au point u.
d_u^2	Fonction de variance au point u.
d_u	Fonction d'erreur de prédiction au point u.
s^2	Estimation de la variance expérimentale.
ddl	Nombre de degré de liberté associé à s^2.
F_α	Statistique correspond au test de ficher.
α	Le risque.
v	Niveau de confiance choisi.
λ_i	Les valeurs propres de la matrice de dispersion.
$Eff-D$	Efficacité vis-à-vis du critère D.
$Eff-G$	Efficacité vis-à-vis du critère G.
$Eff-\mathcal{M}$	Efficacité vis-à-vis du critère $\mathcal{M}$.
$Eff-R$	Efficacité vis-à-vis du critère R.
ξ^*	La matrice d'effet optimale.
$x_{(i)}$	Point mauvais.
$x^{(j)}$	Point bon.
$\xi_n^{(0)}$	La matrice d'effet de départ à n expériences.
$\xi_n^{(t)}$	La matrice d'effet à l'instant t.

$\left(X_n^t.X_{Nn}\right)^{(t)}$	La matrice d'information à l'instant t.
$d\left(\xi_n^{(t)}, x_{(i)}\right)$	Fonction d'erreur de prédiction au point $x_{(i)}$.
$d\left(\xi_n^{(t)}, x_{(i)}, x^{(j)}\right)$	Fonction d'erreur de prédiction pour le couple $\left(x_{(i)}, x^{(j)}\right)$.
$\Delta\left(\xi_n^{(t)}, x_{(i)}, x^{(j)}\right)$	La fonction Δ pour le couple $\left(x_{(i)}, x^{(j)}\right)$.
ϵ_{fed}	Le maximum de Δ.
$maxi$	Le nombre d'excursion.
α	Amplitude des déplacements.
c	Température.

Liste des illustrations, graphiques et tableaux

Liste des algorithmes

INTRODUCTION GENERALE

Pour proposer une solution répondant aux objectifs industriels, il est donc parfois nécessaire de chercher l'information manquante en réalisant un ensemble d'expériences.

Les décisions importantes prises à partir des résultats expérimentaux et le coût non négligeable d'une expérimentation interdissent de laisser à la seule intuition de l'expérimentateur la recherche de la solution du problème. Il est nécessaire d'utiliser une approche méthodologique qui permet non seulement de réduire le coût de l'expérimentation, mais aussi d'établir une organisation optimale des expériences.

Le but de la méthodologie de la recherche expérimentale (MRE) est de proposer une ou plusieurs stratégies expérimentales permettant de résoudre un problème particulier posé par cette derniére.

Dans les plans d'expériences il existe plusieurs plans, parmi eux il existe les plans D-optimaux qui aident les expérimentateurs à réaliser plusieurs expériences dans des meilleures conditions. Pour bien faciliter le travail on utilise des algorithmes qui sont des étapes à suivre afin de réaliser l'expérience à un temps limité, parmi ces algorithmes il y a les algorithmes d'échanges dont on fait l'échange du mauvais point par le bon et ceux par le déplacement car on déplace les points jusqu'à l'obtention du bon point et on fait l'échange. A la fin on trouve la matrice d'expérience optimale.

Dans notre travail on va étudier tout ces algorithmes et les programmer sur le logiciel **Maple** afin de pouvoir les comparer.

Nous rappelons dans le premier chapitre les grands principes de la MRE : intérêt et vocabulaire de base des plans d'expériences, leurs différents types ainsi que les étapes de l'étude et le choix des plans.

Le deuxième chapitre est entièrement consacré aux plans D-optimaux tout en parlant de leurs principes et les critères d'optimalité. Le troisième chapitre traite ces derniers plans

mais plus précisément sur les algorithmes d'échanges, où on les applique sur un médicament bien protégé et à la fin on compare ces derniers.

Le quatrième chapitre traite des nouvelles méthodes qui sont inspirés des algorithmes d'échanges mais avec des déplacements, leurs applications est sur une étude explosive et on trouve à la fin aussi la comparaison.

Le cinquième chapitre traite un nouvel algorithme de Donev modifié avec une comparaison entre Donev et Donev modifié, et une proposition des étapes pour construire l'algorithme de colonie de fourmis dans les plans D-optimaux, on termine ce travail avec une conclusion générale.

CHAPITRE I

METHODOLOGIE DE LA RECHERCHE EXPERIMENTALE

Les plans d'expériences sont des manipulations réalisées à des fins scientifiques dans des conditions rigoureusement contrôlées. Ce chapitre traite les plans d'expériences d'une façon générale dont il parle de leurs intérêts, leurs différents types ainsi que les étapes à suivre pour réaliser une expérience et à la fin il parle des choix d'expériences.

1.1 Historique

Réaliser des expériences afin d'étudier et de comprendre un phénomène est une démarche qui remonte à la nuit des temps. Dès le moyen-âge Nicolas Oresme (1325 – 1382) aborde cette question dans ses écrits. Inspirateur de Descartes et Leibnitz, Francis Bacon (1561 – 1626) est un des précurseurs de la méthode expérimentale. En 1627 il fait par exemple macérer des grains de blé dans neuf concoctions différentes afin d'étudier leurs effets sur la rapidité de germination. Arthur Young (1746 – 1820) cherche ensuite à systématiser le procédé et aborde la notion de répétabilité des expériences afin de prendre en compte leurs variabilité. Citons aussi les travaux de Cretté de Palluel (1741 – 1798). C'est ensuite principalement au 19^{ieme} siècle que les méthodes expérimentales se démocratisent.

Les méthodes rigoureuses d'expérimentation basées sur l'utilisation des plans d'expériences sont dues aux travaux de Sir Ronald Fisher (1890 – 1962), ce brillant mathématicien, très productif dans le domaine de la Statistique a été amené à s'intéresser aux techniques d'expérimentation et c'est le point de départ de la méthode théorique des plans d'expériences.

Divers chercheurs ont par la suite marché dans les traces de Fisher afin de promouvoir et développer l'utilisation des techniques de planification expérimentales dans d'autres

domaines que l'agronomie. Dès les années 50 les travaux de Box et ses collaborateurs (principalement sur les surfaces de réponse) ont entrainé un bon nombre d'application pratiques, mais ce sont certainement les travaux de G.Taguchi qui ont permis une vaste diffusion des plans d'expériences, notamment dans le milieu industriel. Taguchi a eu l'idée de réaliser des tables de configurations expérimentales de référence facilement utilisables par des non-spécialistes.

De nombreux chercheurs contemporains ont continué le développement de cette branche de la Statistique dans des voies diverses et variées: adaptation des plans d'expériences pour les problèmes de mélanges, introduction d'effets de blocs, utilisation des modèles non-linéaires, utilisation des modèles contenant des effets de voisinage, plans d'expériences pour expériences simulées, etc ...[9].

1.2 Intérêts des plans d'expériences

Les plans d'expériences sont utilisés dans les études industrielles en recherche-développement. Ils interviennent dans de nombreux domaines industriels [1]:

- L'énergie renouvelable;
- Industries chimiques, pétrochimiques et pharmaceutiques;
- Industries mécaniques et automobiles;
- Industries métallurgiques ...

Leurs utilisation vise aux buts suivants :

- Détermination des facteurs clés dans la conception d'un nouveau produit ou d'un nouveau procédé.
- Optimisation des réglages d'un procédé de fabrication ou d'un appareil de mesure.
- Prédiction par modélisation du comportement d'un procédé.

Les plans d'expériences s'inscrivent dans une démarche générale d'amélioration de la qualité.

Le succès de la démarche originale des plans d'expériences réside dans la possibilité d'interprétation des résultats expérimentaux avec un effort minimal sur le plan expérimental: la minimisation du nombre nécessaire d'expériences permet un gain de temps et en coût financier.

Il faut néanmoins comprendre que les plans d'expériences ne sont pas un outil destiné à priori à la recherche fondamentale car ils ne permettront jamais une explication du phénomène physico-chimique étudié.

1.3 Vocabulaire de base des plans d'expériences

Le scientifique est souvent amené à comprendre comment réagit un système en fonction des facteurs susceptibles et de le modifier. Pour visualiser cette évolution, il mesure une réponse et il va ensuite essayer d'établir des relations de cause à effet entre les réponses et les facteurs[8].

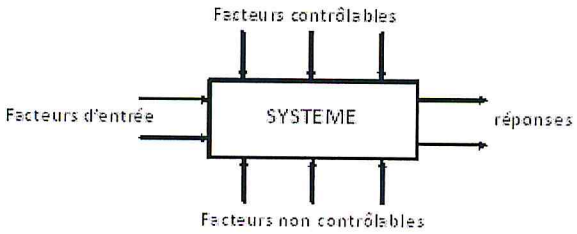


Figure 1.1 : Représentaion du système.

1.3.1 Facteurs

Les variables que l'on désire étudier sont appelées facteurs[2].Parmi ces dérniers on distinguera :

- Les **facteurs d'entrés** dont on cherche à analyser une influence (matière première, vitesse d'agitation, température, rendement ...).
- Les **facteurs contrôlables** qui dépendent directement du choix du technicien (pression, température, matériau ...).
- Les **facteurs non contrôlables** qui varient indépendamment du choix du technicien (conditions climatiques, environnement d'utilisation...).

- Les **facteurs quantitatifs** lorsqu'ils sont naturellement exprimés à l'aide de valeurs numériques.

- Les **facteurs qualitatifs** si les facteurs sont de type modalité.

Les facteurs étudiés dans un plan d'expérience sont bien entendu les facteurs d'entrée. En général un facteur varie entre deux bornes : la borne inférieure (niveau bas que l'on note souvent par -1) et la borne supérieure (niveau haut que l'on note souvent par $+1$). L'ensemble de toutes les valeurs que peut prendre le facteur entre les deux niveaux s'appelle **le domaine de variation** du facteur ou **le domaine d'expérience**.

Un facteur peut prendre plusieurs niveaux à l'intérieur de son domaine de variation (figure 1.2).

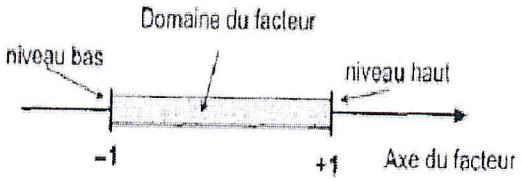


Figure 1.2 : Le domaine de variation du facteur est constitué de toutes les valeurs comprises entre -1 et $+1$.

Les niveaux des facteurs peuvent être considérés comme les coordonnées d'un point de l'espace expérimental. (figure 1.2)

Une expérience est une intervention volontaire dans un système pour observer ou mesurer les effets de cette derniére. Elle consiste donc à modifier certains paramètres afin d'observer les réponses. Elle est représentée par un point dans ce système d'axe qui est appelé **point d'expérience**.

Un plan d'expérience est un ensemble de plusieurs points expérimentaux.

La réunion des domaines de variation de chaque facteur définit **le domaine d'étude** (figure 1.3). Ce dernier est la partie de l'espace expérimental que l'expérimentateur a retenu pour faire ses essais.

Une étude est un ensemble d'expériences bien définies et représentées par une série de points disposés dans le domaine d'étude.(figure 1.4)

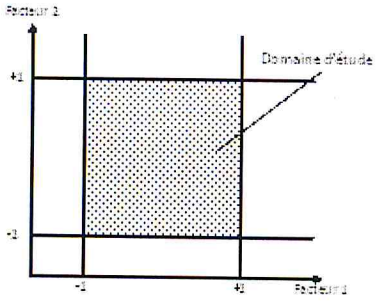


Figure 1.3 : Les domaines de tous les facteurs forment le domaine d'étude

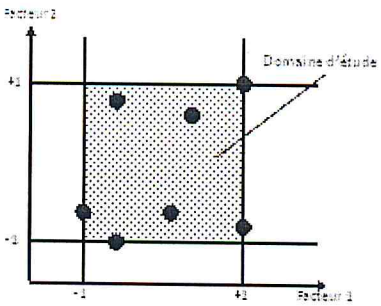


Figure 1.4 : Les points expérimentaux sont disposés dans le domaine d'étude

1.3.1.1 Variables codées ou variables centrées réduites

L'utilisation des variables centrées réduites présente l'intérêt de pouvoir généraliser la théorie des plans d'expériences quels que soient les facteurs ou les domaines d'études retenus. Remplacer les variables naturelles par les variables codées va permettre d'avoir pour chaque facteur le même domaine de variation (entre -1 et $+1$) et de pouvoir ainsi comparer entre eux l'effet des facteurs. Le niveau bas est ainsi codé -1 alors que le niveau haut est codé $+1$[14].

On note ici t la variable codée et T la variable naturelle. On peut les lier par la relation suivante où T_0 est le milieu de l'intervalle du domaine d'étude et $\triangle T$ la moitié de la largeur du domaine d'étude, T_0 et $\triangle T$ étant exprimé en variable naturelle.

$$t = \frac{T - T_0}{\triangle T} \tag{1.1}$$

Les variables centrées réduites sont sans dimension.

1.3.2 Réponses

La réponse est la grandeur mesurée à chaque essai; le plan vise à déterminer quels facteurs l'influencent ou quelle est son évolution en fonction de ceux-ci. Cette grandeur est la plus souvent mesurable mais elle peut également être qualitative[2].

1.4 Les différents types des plans d'expériences

Les plans d'expériences peuvent être classés en trois catégories, les deux premières couvrent les facteurs indépendants et la troisième catégorie est réservée aux facteurs dépendants[2]. Les facteurs indépendants sont des facteurs dont on peut choisir les niveaux comme on le désire donc le choix des niveaux n'entraine aucune contrainte sur le choix des niveaux des autres facteurs, par contre les facteurs dépendants sont des facteurs dont les niveaux sont liés entre eux par la relation : $\sum_{i=1}^{k} x_i = 1$

1.4.1 Plans de criblage

Ces plans permettent de découvrir les facteurs les plus influents sur une réponse donnée. On ne cherche pas vraiment à obtenir une relation précise entre les variations des facteurs et celle de la réponse. Ce dernier a comme type :

- **Plans « un facteur à la fois »** : Ce type de plan peut se révéler très utile lorsqu'il y a beaucoup de facteurs et que le phénomène est compliqué. On n'obtient aucune interaction.
- **Plans factoriels fractionnaires** 2^{k-p} : Ce type permet d'étudier beaucoup de facteurs, si l'on sait bien les interpréter. On peut détecter les interactions entre les facteurs.
- **Plans sursaturés** : Sont des plans qui prennent en compte un grand nombre de facteurs et qui ne demandent que quelques essais.
- **Plans factoriels complets** 2^k **à deux niveaux** : Ces plans sont gourmands en essais dès que l'on dépasse trois facteurs. Ils peuvent servir à faire des criblages mais aussi à faire de la modélisation.

1.4.2 Plans de modélisation

- **Plans factoriels complets à trois niveaux** : Ces plans peuvent être utilisés pour la modélisation mais comme il n'y a que deux niveaux par facteur, on ne peut qu'employer des modèles du premier degré avec interaction.

- **Plan non conventionnel** : c'est tout plan qui s'écarte des plans classiques.

1^{er} cas : un plan factoriel classique a ses points expérimentaux aux sommets du domaine d'étude. Il se peut qu'au cours de l'expérimentation les niveaux prévus n'aient pas été respectés et que les points d'expériences soient décalés par rapport aux sommets. Le réalisé est un plan non conventionnel.

2^{ieme} cas : un plan classique possède un nombre précis d'essais à réaliser. L'ensemble des essais réels est un plan non conventionnel.

- **Plans composites** : permettent une modélisation du second degré. Elle comprend un plan factoriel, un plan en étoile et des points centraux.
- **Plans de Doehlert** : Ces plans ont été imaginés pour interpréter les réponses obtenues avec un modèle du second degré tout en effectuant un minimum d'essais.
- **Plans de Box-Behnken** : Ces plans permettent de modéliser les réponses avec un modèle du second degré tout en respectant certains critères d'optimalité.
- **Plans de Roquemore** : Ces plans ne nécessitent que peu d'essais pour obtenir un modèle du second degré. Ils tentent de respecter plusieurs critères d'optimalités.
- **Plans D-optimaux** : Ces plans permettent de tenir compte de nombreuses contraintes. Ils assurent la précision sur les coefficients du modèle.

1.4.3 Plans de mélanges

Les plans de mélanges sont de plans que l'on utilise lorsqu'on étudie des produits composés de plusieurs constituants, l'objectif est de trouver la loi qui réagit une ou plusieurs réponses en fonction de la composition du mélange.

1.5 Etapes d'une étude par les plans d'expériences

1.5.1 Choix de la réponse et les facteurs d'étude

L'étude doit avant tout avoir un but précis[1]: minimiser un coût de fabrication et chercher les paramètres influents . A ce niveau, il est important de rassembler l'ensemble

Où $a_0, a_1, a_2, \ldots$ sont les coefficients du polynôme.

Les termes produits de type par exemple $a_{ij} \cdot x_i \cdot x_j$ correspondent aux interactions.

1.5.2.1 Matrice d'expériences

La représentation géométrique d'un plan d'expérience est commode pour imaginer la position des points expérimentaux dans le domaine d'étude. Mais elle ne peut plus être employée dés que le nombre de facteurs est supérieur à trois. Pour les espaces multidimensionnels, nous adopterons une représentation en forme de tableau ou matrice d'expériences. La matrice d'expériences ξ comprend une première colonne qui identifie les essais, les colonnes suivantes indiquent les cordonnées des points expérimentaux prévues. Son avantage est d'être utilisable quelque soit le nombre de facteurs.

1.5.2.2 Matrice des effets

La matrice des essais à réaliser pour obtenir le plan d'expérience optimal se déduit en fait des critères permettant d'obtenir les coefficients avec le maximum de précision. Le système d'équations à résoudre doit présenter des coefficients devant les inconnues (qui sont les coefficients du modèle à déterminer) pouvant se mettre sous la forme d'une matrice nommée matrice des effets notée X .

1.5.2.3 Matrices d'information et de dispersion

On appelle matrice d'information la matrice X^tX et si elle est inversible, son inverse est appelée matrice de dispersion $(X^tX)^{-1}$.

1.5.3 Les expériences à réaliser

La méthode des plans d'expériences peut être comparée à la méthodologie traditionnelle dite de "variation facteur par facteur". Pour étudier l'influence des deux facteurs sur une réponse, on peut adopter deux stratégies expérimentales pour la conception des essais.

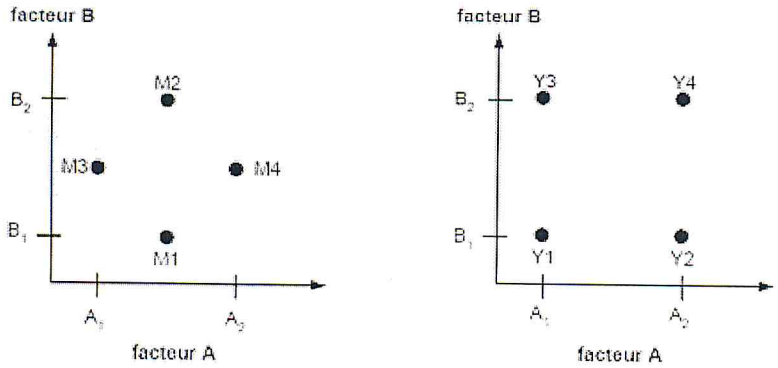


Figure 1.5 : Le choix des expériences à réaliser.

Selon la méthode traditionnelle, on bloque le facteur A au centre du domaine de variation et on fait varier le facteur B aux deux extrémités de son domaine: on obtient les mesures $M1$ et $M2$. Avec le facteur A on réalise la même opération pour obtenir les points $M3$ et $M4$. Dans cette méthode, l'effet de B sera mesuré à partir des mesures $M1$ et $M2$ et celui de A à partir des mesures $M3$ et $M4$. Donc pour chaque facteur la moitié des mesures seulement est utilisée pour rendre compte d'un effet.

La méthode des plans d'expériences consistera à réaliser 4 essais aux extrémités du domaine expérimental. L'effet de A apparaît comme la différence entre la moyenne $\frac{Y2+Y4}{2}$ et la moyenne $\frac{Y1+Y3}{2}$. Le même raisonnement s'applique pour l'effet de B.

Remarque 1 *[1]L'effet de A peut être décrit autrement en écrivant l'expression sous la forme suivante :*

$$\begin{aligned} Effet\ de\ A &= \frac{Y2+Y4}{2} - \frac{Y1+Y3}{2} \\ &= \frac{Y4-Y3}{2} - \frac{Y2-Y1}{2} \end{aligned} \qquad (1.3)$$

Le terme $\frac{Y4-Y3}{2}$ est le demi-effet de A quand B est au niveau haut (c'est-à-dire la demi-variation de la réponse quand A passe du niveau bas A_1 au niveau haut A_2). De même le terme $\frac{Y2-Y1}{2}$ est le demi-effet de A quand B est au niveau bas.

Dans cette deuxième stratégie toutes les mesures sont utilisées pour le calcul d'un effet. On comprend donc que la précision obtenue sera supérieure avec la méthode de plans d'expériences. Un autre avantage de la méthode des plans réside dans un nombre beaucoup

plus faible d'expériences à réaliser que dans la méthode traditionnelle quand le nombre de facteurs augmente.

Il reste néanmoins à connaître la méthode générale pour trouver les "bonnes" expériences à réaliser. La méthode dépend du modèle mathématique choisi.

1.5.4 Réalisation des essais

Un soin tout particulier doit être apporté à l'exécution des essais. Si on ne réalise pas personnellement les essais, il faut notamment vérifier que les facteurs contrôlables mais non étudiés soient bien fixés à des valeurs précises. De même si un des facteurs étudiés est un composé chimique, il est bien sûr préférable de ne pas avoir à changer de lot de matière première durant l'ensemble de l'expérimentation.

1.5.5 Interprétation des résultats des essais

Comme première approche, le plan d'expérience peut être conçu comme un moyen de savoir quels sont les facteurs ou les interactions qui ont une influence statistiquement significative sur la réponse étudiée ?

L'exploitation des résultats expérimentaux est souvent assez rapide, surtout avec un logiciel ! Le principe de l'exploitation est simple : il consiste à calculer les coefficients du modèle polynomial plus la valeur absolue du coefficient est élevée plus le terme correspondant (facteur simple ou interaction) sera influent sur la réponse étudiée. La difficulté est plutôt de pouvoir distinguer une véritable influence et le rôle de l'incertitude entachant inévitablement toute mesure.

En conclusion de l'étude on fournit la liste des facteurs influents la plupart du temps l'expression du modèle en ne retenant que les coefficients jugés statistiquement significatifs.

Il est bon de signaler que le modèle obtenu ne peut être utilisé qu'à l'intérieur du domaine d'étude (d'où l'utilité d'une étude préalable correcte ...) : toute extrapolation est très risquée car elle pourrait apporter des résultats bien différents de ceux attendus. On ne signalera jamais assez que le modèle fourni n'a pas de signification physique et ne saurait être assimilé à une loi physique.

Une dernière étape obligatoire avant l'utilisation du modèle en production sera de tester

par une expérience au centre du domaine expérimental si la valeur prédite par le modèle est proche de la valeur expérimentale. En effet les modèles des plans d'expériences factoriels étudiés ne prennent pas en compte une courbure possible au centre du domaine mais considèrent uniquement un comportement linéaire. Si la vérification n'est pas concluante il importe d'envisager des modèles plus compliqués (modèles avec termes du second degré).

1.6 Calcul des coefficients du modèle

Dans une étude expérimentale quand on cherche à relier une grandeur physique y et une grandeur physique x, on utilise souvent une technique de régression linéaire de y par rapport à x qui consiste à définir une relation du type[1] :

$$y = a.x + b \tag{1.5}$$

On cherche les valeurs des paramètres a et b de façon que la droite passe au mieux par l'ensemble des points expérimentaux. On utilise une méthode dite "**des moindres carrés**" qui minimise la somme des carrés des résidus e_i . Le **résidu** e_i se définit comme l'écart entre la valeur expérimentale y_i obtenue pour une valeur x_i et la valeur calculée à partir du modèle $\hat{y}_i$ soit :

$$e_i = y_i - \hat{y}_i \tag{1.6}$$

Les plans d'expériences exigent l'utilisation de la technique de régression multilinéaire pour déterminer les coefficients d'un modèle polynômial.

Pour effectuer le calcul il faut évidemment un nombre d'expériences au moins égal au nombre de coefficients (nombre d'inconnues). L'objectif est de trouver un ensemble de p coefficients qui résout le mieux possible le système d'équations, on cherche le jeu des coefficients qui minimise la somme des carrés des écarts. C'est l'hypothèse des moindres carrés. La somme des carrés des écarts s'écrit sous forme matricielle $e^t e$, cette somme sera minimale par rapport aux coefficients si :

$$\frac{\partial e^t e}{\partial a} = 0 \tag{1.7}$$

Cette relation matricielle représente p équations, une par coefficient. L'hypothèse des moindres carrés apporte donc les p équations dont on a besoin.

Pour trouver les coefficients, il suffit de résoudre le système suivant de $p+N$ équations à $p+N$ inconnues :

$$\begin{cases} y = X.a + e \\ \frac{\partial e^t e}{\partial a} = 0 \end{cases} \tag{1.8}$$

Avec X matrice des effets.

Dans lequel :

$$\begin{aligned} e^t e &= (y - Xa)^t (y - Xa) \\ &= \left(y^t - a^t X^t\right)(y - Xa) \\ &= y^t y - a^t X^t y - y^t X a + a^t X^t X a \end{aligned}$$

Le second et le troisième terme sont des scalaires, nous pouvons donc écrire :

$$a^t X^t y = y^t X a$$

Alors :

$$e^t e = y^t y - 2a^t X^t y + a^t X^t X a$$

En prenant la dérivée de la fonction $e^t e$ par rapport au vecteur a :

$$\begin{aligned} \frac{\partial e^t e}{\partial a} &= 0 - 2X^t y + 2X^t X a = 0 \\ &\Rightarrow \quad X^t X a = X^t y \end{aligned}$$

La matrice $X^t X$ est une matrice carrée symétrique appelée matrice d'information, si son déterminant est différent de zéro nous pouvons trouver son inverse, d'où la formule :

$$\hat{a} = \left(X^t X\right)^{-1} X^t y \tag{1.9}$$

Cette relation est fondamentale et nous l'utiliserons constamment par la suite, elle est valable pour tous les modèles polynomiaux quelque soit leur degré et quelque soit le nombre de coefficient. Le jeu des coefficients ainsi obtenu est utilisé pour écrire le modèle mathématique qui permet de calculer les réponses dans tout le domaine d'étude.

$$\hat{y} = X\hat{a} \tag{1.10}$$

Les réponses ainsi calculées à l'aide des coefficients des moindres carrés sont souvent appelées les réponses prédites, elles sont trés utiles pour faire des prévisions, tracer les diagrammes d'isoréponses, chercher des optimums ou des valeurs bien précises. C'est un précieux outil de prévision.

1.7 Propriétés statistiques pour les coefficients calculés

1.7.1 L'espérance et la variance mathématique

D'après la formule (1.9), le vecteur $\hat{a}$ est en fonction de y, puisque y est de nature aléatoire alors $\hat{a}$ est aléatoire, donc on peut calculer son espérance et sa variance mathématique[13].

Nous avons alors :

- $E(\hat{a}) = E(a) + E(e)$.
- $var(\hat{a}) = \hat{\sigma}^2 = (X^t X)^{-1}.var(e)$.

Remarque 2 *Selon l'hypothèse des moindres carrées, l'erreur e est distribuée suivant la loi normale* $N(0, \sigma^2)$.

1.7.2 Fonction de variance

L'incertitude sur les coefficients estimés du modèle se reporte naturellement sur les valeurs du modèle lui-même, par la relation fondamentale $\hat{y} = X.\hat{a}$ [3].

Ainsi pour un point u quelconque du domaine d'étude possible le modèle prend la valeur:

$$\hat{y}_u = f_X(x_u).\hat{a} \tag{1.11}$$

On suppose ici que le placement des points d'expériences n'introduit pas d'erreur, donc $f_X(x_u)$ est une grandeur constante. Ainsi la variance du modèle au point u :

$$\hat{\sigma}_u^2 = f_X(x_u)^t.\hat{\sigma}^2.f_X(x_u) \tag{1.12}$$

D'où finalement :

$$\hat{\sigma}_u^2 = \hat{\sigma}^2.f_X(x_u)^t.(X^t X)^{-1}.f_X(x_u) \tag{1.13}$$

A la différence de la variance des coefficients, la variance de modèle varie dans le domaine d'étude et est donc une fonction des k facteurs.

On définit **la fonction de variance** d_u^2 au point u, comme suit :

$$d_u^2 = d^2(x_u) = .f_X(x_u)^t . \left(X^t X\right)^{-1} .f_X(x_u) \tag{1.14}$$

Et par conséquent :

$$\hat{\sigma}_u^2 = \hat{\sigma}^2 . d_u^2 \tag{1.15}$$

Enfin, on peut déduire la notion de **fonction d'erreur de prédiction** au point u, comme étant la racine carrée de la fonction de variance.

$$d_u = \sqrt{d_u^2} = \sqrt{f_X(x_u)^t . \left(X^t X\right)^{-1} .f_X(x_u)}. \tag{1.16}$$

On constate ainsi que la fonction d'erreur de prédiction (pour un point u quelconque du domaine d'étude) est un rapport d'écarts-types :

$$d_u = \frac{\hat{\sigma}_u}{\hat{\sigma}} \tag{1.17}$$

CHAPITRE II

LES PLANS D-OPTIMAUX

Ce chapitre traite en général les plans D-optimaux, leurs intérêt, leurs types et à la fin les notions de critère.

2.1 Principe des plans D-Optimaux

Un plan D-optimal est un plan qui permet non seulement d'étudier les effets des différents facteurs mais aussi permet de représenter le phénomène par un modèle mathématique quadratique, delà on peut effectuer une optimisation sans contrainte. Un plan est dit D-optimal s'il minimise le critére :

$$critére - D = \left|\left(X^t X\right)^{-1}\right|$$

2.2 Quand utilise-t-on les plans D-optimaux ?

Les plans D-optimaux sont souvent utilisés lorsque le domaine expérimental n'est pas entièrement accessible. Il existe des contraintes qui interdisent l'accès à certaines régions du domaine d'étude. Par exemple, il est impossible de réaliser certaines combinaisons de niveaux par manque de matériel s'il existe des dangers liés au niveau des facteurs (explorer la plus grande partie possible du domaine expérimental malgré les restrictions dues aux contraintes).

Les plans D-optimaux sont aussi utilisés lorsqu'on veut diminuer le nombre des essais d'un plan classique. On impose un nombre d'essais et l'algorithme de calcul des plans conserve pour un modèle donné les meilleurs points d'expériences.

2.3 *Exemple d'utilisation d'un plan D-optimal*

Soient x_1, x_2 deux facteurs, pour démontrer l'approche D-optimale de façon géométrique, les deux facteurs sont étudiés sur trois niveaux, $-1, 0$ et 1. L'ensemble candidat correspondant est mentionné dans le tableau (2.1).

Nombre	1	2	3	4	5	6	7	8	9
x_1	-1	-1	-1	0	0	0	1	1	1
x_2	-1	0	1	-1	0	1	-1	0	1

Tableau 2.1 : Ensemble candidat à deux facteurs avec trois niveaux.

Si nous voulons afficher l'ensemble candidat à la notation des matrices, la matrice ξ_N doit contenir deux colonnes, une pour chaque facteur et $N = 9$ expériences. La figure (2.2) montre les neuf candidats comme un terrain sur la zone expérimentale.

Compte tenu d'une conception avec seulement trois courses ($n = 3$), nous avons $9!/(3! * 6!) = 84$ sous-ensembles possibles hors de cet ensemble de candidats. Dans cet exemple, nous évaluons quatre conceptions possibles.

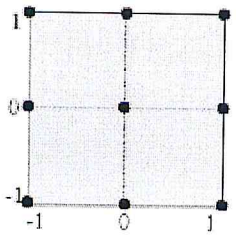


Figure 2.1 : Répartition des candidats sur la région expérimentale.

L'enquête prend en considération deux facteurs avec trois niveaux et donc neuf points candidats de la matrice et de les comparer en fonction des D-critères. L'équation $(*)$ montre les quatre sous-ensembles sélectionnés dans la notation de la matrice et la figure 2.3 affiche la région expérimentale avec les candidats correspondants.

$$\xi_{N_{c_1}} = \begin{bmatrix} -1 & -1 \\ 0 & 0 \\ 1 & 1 \end{bmatrix} \quad \xi_{N_{c_2}} = \begin{bmatrix} 0 & -1 \\ -1 & 0 \\ 0 & 1 \end{bmatrix} \quad \xi_{Nc_3} = \begin{bmatrix} -1 & -1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \xi_{N_{c_4}} = \begin{bmatrix} -1 & -1 \\ 1 & -1 \\ -1 & 1 \end{bmatrix}$$

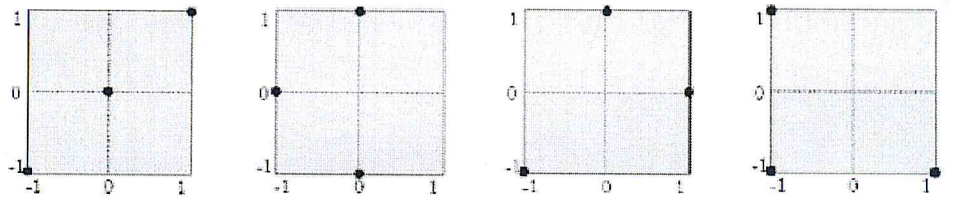


Figure 2.2 : Répartition de la matrice du modèle.

Pour calculer l'optimalité de ces conceptions, nous devons choisir un modèle linéaire pour que l'exemple soit le plus simple possible :

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \varepsilon \qquad (*)$$

En comparant les conceptions selon le D-critère, n'est possible que si nous calculons le déterminant de la matrice d'information $(X^t X)$ pour chacun des quatre motifs.

Expérience	Déterminant
$\xi_{N_{c_1}}$	0
$\xi_{N_{c_2}}$	4
ξ_{Nc_3}	9
$\xi_{N_{c_4}}$	16

Tableau 2.2 : Les différents déterminants des expériences.

2.4 Notion de critère

Il était vu précédemment que la précision des coefficients, et donc la précision des valeurs de la modélisation dépendaient uniquement [3]:

- De la valeur de la variance de régression $\hat{\sigma}^2$;
- Du placement des points d'expériences (matrice d'expériences ξ) ;
- Du type de modèle utilisé (matrice d'expériences ξ).

Les valeurs mesurées de la réponse n'intervenant pas (la terme $\hat{\sigma}^2$ n'étant pas pris en compte), il est donc possible de fixer la précision des coefficients $\hat{a}$ uniquement par la spécification de la matrice d'expériences ξ . Cette étape a donc lieu avant la réalisation des expériences. Pour un type de modèle donné, on recherchera donc le placement **optimal** des points d'expériences pour lequel l'erreur su sur les réponses prédites est la plus faible possible. On gardera à l'esprit que dans tous les cas un critère d'optimalité ne s'applique que pour une situation bien définie, c'est-à-dire pour laquelle on a fixé [3]:

- Le nombre d'expériences du plan ;
- Le modèle d'interpolation choisi.

En réalité, il existe plusieurs types d'objectifs finaux recherchés qui s'avèrent être très proches dans leurs finalités. En particulier, on peut ainsi être amené à [3]:

- Minimiser la somme des variances des coefficients ;
- Minimiser la variance globale de tous les coefficients ;
- Minimiser la plus grande variance de coefficient ;
- etc.

2.4.1 Définition d'un critère

E étant l'ensemble des matrices d'expériences exactes constituées de N expériences, un critère est une mesure $C(\xi)$ sur une matrice d'expériences ξ appartenant à E, cette dèrniere prend des valeurs réelles ou booléennes (oui/non); dans le premier cas le critère est dit quantitatif, dans le second qualitatif.

Pour un critère quantitatif, nous recherchons généralement la valeur la plus petite possible; dans certains cas cependant, nous sommes intéressés par la valeur la plus grande [6].

2.4.2 Remarques sur les critères

Plusieurs auteurs ont montré que la limite du domaine de confiance des coefficients est donnée par la relation[4] :

$$(a - \hat{a})^t \left(X^t X\right) (a - \hat{a}) = p s^2 F_{\alpha,p,ddl} \tag{2.2}$$

Où :

s^2 est une estimation de la variance expérimentale.

F est la statistique correspond au test de ficher.

α est le niveau de confiance choisi.

X est la matrice des effets.

p est ombre de coefficients d'un modèle mathématique.

$\hat{a}$ est le vecteur calculé.

ddl est le nombre de degré de liberté associé à s^2.

Ce domaine est un hyper ellipsoïde dans l'espace des coefficients, centré sur le vecteur calculé $\hat{a}$. Avec un risque α, on peut dire, que les valeurs vraies des coefficients sont dans cet hyper ellipsoïde. Pour deux coefficients, l'hyper ellipsoïde est représenté graphiquement par :

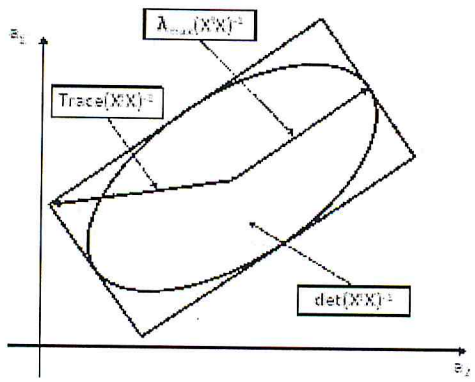


Figure 2.3 : Ellipsoïde de confiance

2.4.2.1 Volume

Le volume de l'ellipsoïde est égal à $\pi \left(\lambda_1 \lambda_2\right)^{1/2} = \pi \left(\det \left(X^t X\right)^{-1}\right)^{1/2}$, ce dernier est lié au déterminant de la matrice de dispersion $\left(X^t X\right)^{-1}$. Plus le volume sera petit, plus

l'hyper ellipsoïde tend vers un point et plus on se rapprochera de la solution vraie A.

2.4.2.2 Forme

Lorsque la forme est très allongée, il existe alors une grande disparité sur la précision des coefficients. Si $\lambda_1 = \lambda_2$ l'ellipsoïde devient une hyper sphère, et donc tous les coefficients sont déterminés avec la même précision.

2.4.2.3 Orientation

Si les axes principaux de l'hyper ellipsoïde sont parallèles aux axes des coefficients alors la valeur calculée d'un coefficient sera indépendante des valeurs calculées des autres coefficients. Cet hyper ellipsoïde représente les performances des ensembles d'observations pour un modèle donné.

D'après la formule (2.2), l'hyper ellipsoïde dépend de la matrice de (X^tX), donc les critères sont fondés sur les matrices d'information (X^tX) et de dispersion $(X^tX)^{-1}$.

2.4.3 Critères d'optimalité

Il existe aujourd'hui de nombreux critères d'optimalité. Il serait difficile d'être exhaustif, mais on essaiera toutefois d'en présenter une liste suffisamment conséquente pour bien apprécier la variété des choix possibles. Après une présentation générale et un certain aperçu géométrique d'un certain type de critère, on les exposera tout en les partitionnant en quatre classes [7]:

- **Classe 1** : critère pour l'estimation de paramètre et leur fonction.
- **Classe 2** : critère dans l'espace des observations.
- **Classe 3** : critère pour la discrimination des modèles.
- **Classe 4** : critère d'erreur quadratique moyenne.

Les principaux criteres d'optimalités sont [4]:

2.4.3.1 Critere A

Une matrice d'experiences est dite **A-optimale** si elle conduit à la trace minimale de $(X^tX)^{-1}$.

2.4.3.2 *Critere D*

Le critere-D est défini par :

$$D(\xi) = \left|(X^t X)^{-1}\right| \tag{2.5}$$

Une matrice d'experiences ξ^{*D} est dite **D-optimale** si elle conduit au déterminant minimal pour sa matrice de dispersion.

$$\xi^{*D} = \arg_{\xi \in \Xi} \min D(\xi) \tag{2.6}$$

2.4.3.3 *Efficacité-D*

L'efficacité de la matrice d'expériences ξ^1 par rapport à la matrice d'expériences ξ^2 est définie par :

$$\begin{aligned} Eff - D(\xi^1, \xi^2) &= \left(\frac{\left|(X_1^t X_1)^{-1}\right|}{\left|(X_2^t X_2)^{-1}\right|}\right).100 \\ &= \left(\frac{\left|X_2^t X_2\right|}{\left|X_1^t X_1\right|}\right).100 \end{aligned} \tag{2.7}$$

2.4.3.4 *Critere E*

Une matrice d'experiences est dite **E-optimale** si sa valeur propre maximale de $(X^t X)^{-1}$ est la plus faible possible .

2.4.3.5 *Critère G*

Ce critère prend en compte la plus grande valeur sur tout le domaine expérimental χ, de la fonction de variance $d^2(y_u)$ engendré par la matrice ξ_N.

$$d_{max}(\xi_N) = \max_{u \in \chi}(d^2(y_u)) \tag{2.8}$$

La meilleure matrice d'expériences vis à vis de ce critère étant celle qui a la plus petite valeur :

$$\xi_N^{*G} = \arg_{\xi_N \in \Xi} \min d_{max}(\xi_N) \tag{2.9}$$

2.4.3.6 *Efficacité-G*

L'efficacité-G de la matrice d'expériences ξ^2 par rapport à la matrice d'expériences ξ^1 est définie par :

$$Eff - G\left(\xi^1, \xi^2\right) = \left[\frac{d_{max}\left(\xi^1\right)}{d_{max}\left(\xi^2\right)}\right].100 \tag{2.10}$$

Avec : $d_{max}\left(\xi^1\right) \leq d_{max}\left(\xi^2\right)$.

2.4.3.7 *Critère M*

le critère-$\mathcal{M}$ permet de tenir compte de la qualité d'information apportée par l'expérience. Ce critère est indépendant du nombre des essais du plan d'expériences.

On appelle matrice des moments M, la matrice définie par :

$$M\left(\xi_N\right) = \frac{X^t X}{N} \tag{2.11}$$

Pour une matrice d'expériences $\xi_N \in \Xi$, le critère-$\mathcal{M}$ est défini par :

$$\mathcal{M}\left(\xi_N\right) = \left|M\left(\xi_N\right)\right| = \frac{\left|X^t X\right|}{N^p} \tag{2.12}$$

Une matrice d'experiences $\xi^{*\mathcal{M}}$ est dite **$\mathcal{M}$-optimale** si :

$$\xi_N^{*R} = \arg_{\xi \in \Xi} \max \mathcal{M}\left(\xi_N\right) \tag{2.13}$$

2.4.3.8 *Efficacité-$\mathcal{M}$*

Soient M_1 et M_2 deux matrices des moments associés à deux matrices d'expériences ξ^1 et ξ^2 constituées respectivement de N_1 et N_2 expériences.

L'efficacité ξ^1 par rapport à ξ^2 est définie par :

$$\begin{aligned} Eff - \mathcal{M}\left(\xi^1, \xi^2\right) &= \left(\frac{\left|M\left(\xi^1\right)\right|}{\left|M\left(\xi^2\right)\right|}\right)^{1/p}.100 \\ &= \frac{N_2}{N_1}.\left(\frac{\left|X_1^t X_1\right|}{\left|X_2^t X_2\right|}\right).100 \end{aligned} \tag{2.14}$$

2.4.3.9 *Critère-R (Critère de redondance)*

Pour une matrice d'expériences ξ_N, le critère-R est défini par :

$$R\left(\xi_N\right) = N \tag{2.15}$$

La minimisation de ce critère n'est pas une fin en soi. Il s'agit plutôt de rechercher le nombre minimal d'expériences permettant d'atteindre les objectifs fixés par l'expérimentateur.

2.4.3.10 Efficacité-R

Soit $\xi_{N_1} \in \Xi$ et $\xi_{N_2} \in \Xi$, telles que $N_1 < N_2$. Alors, l'efficacité-R de la matrice d'expériences ξ_{N_1} vis-à-vis de la matrice d'expériences ξ_{N_2} est définie par :

$$Eff - R\left(\xi_{N_1}, \xi_{N_2}\right) = \frac{N_1}{N_2}.100 \tag{2.16}$$

Plus $Eff - R$ est faible, meilleure est la matrice d'expériences ξ_{N_1} par rapport à la matrice d'expériences ξ_{N_2}, suivant ce critère.

L'efficacité-R de ξ_N par rapport à la matrice d'expériences optimale-R ξ^{*R} est :

$$Eff - R\left(\xi^{*R}, \xi_N\right) = \frac{p}{N}.100 \tag{2.17}$$

Avec $p =$ nombre de coefficients du modèle, i.e. nombre minimal d'expériences.

D'où :

$$\xi^{*R} = \xi_p \tag{2.18}$$

2.4.3.11 Critère d'orthogonalité

Une matrice d'expériences est orthogonale si elle permet d'obtenir des estimations des coefficients indépendants. Cela se caractérise par des axes de l'ellipsoïde parallèles aux axes des coefficients. Cette propriété est obtenue quand $(X^tX)^{-1}$ est diagonale donc quand les covariances des coefficients sont nulles.

2.4.3.12 Critère de Presque Orthogonalité

Si la matrice $(X^tX)^{-1}$ obtenue en retirant sa première ligne et sa première colonne est diagonale, le critère de presque orthogonalité est respecté.

2.4.3.13 Critère d'iso variation par rotation

On désire que les réponses calculées avec le modèle issu du plan d'expériences aient une erreur de prévision identique pour des points situés à la même distance du centre du domaine d'étude. Dans ce cas on parle de plan iso variant par rotation.

2.4.3.14 Autres critères

Il existe de nombreux autres critères d'optimalités, souvent dérivés de ceux présentés précédemment. Dans tous les cas, ils se basent sur des considérations de précision des coefficients de modèle ou bien des valeurs estimées par ce modèle.

CHAPITRE III

CONSTRUCTION ALGORITHMIQUE DES PLANS D-OPTIMAUX PAR LES ALGORITHMES D'ÉCHANGES

La recherche du meilleur plan n'est pas une méthode exacte, mais plutôt une procédure algorithmique utilisant certaines stratégies, pour cela il existe un grand nombre d'algorithmes pour la génération des matrices d'expériences qui utilisent le critère D-optimal. Dans ce chapitre nous allons présenter ces algorithmes et les appliquer sur la fabrication pharmaceutique d'un médicament avec une comparaison entre eux.

Qu'est ce qu'un algorithme d'échange ?

Un algorithme d'échange consiste à sélectionner les matrices ξ^* optimales par un échange d'un ou plusieurs points de plan de départ généré et répéter cette procédure d'échange jusqu'à trouver la meilleure matrice selon les critères D-optimale.

Les algorithmes peuvent être dévisés en deux groupes [10]:

- Le premier sert à ajouter ou supprimer les points d'une manière séquentielle.
- Le second consiste à réaliser un échange par un ajout et une suppression simultanée des points.

Dans les algorithmes d'échanges, on peut traiter deux cas : le 1^{er} cas est une **recherche épuisante** qui évite les points doubles dans le plan donc on travaille qu'avec $(N-n)$ points candidats et le 2^{ieme} c'est une **recherche non épuisante** où la sélection d'une expérience est deux fois possible donc on travaille avec les N point candidats.

3.1 Les Algorithmes avec points candidats

Ils réduisent le domaine d'intérêt a un sous ensemble fini de points, cette discrétisation du domaine définit le sous ensemble de point que l'expérimentateur est prêt à réaliser. C'est dans ce dernier qu'on trouve la matrice d'expérience à suivre si elle existe.

Ces algorithmes sont une succession de remplacement des mauvais points de la matrice par les bons points pris parmi les points candidats (on dit qu'un point $x_{(i)}$ est mauvais, et le point $x^{(j)}$ est bon si le point $x_{(i)}$ entraine un déterminant de la matrice de dispersion plus élevé que celui obtenu avec le point $x^{(j)}$) . A chaque étape on observe la diminution du déterminant de la matrice de dispersion.

Pour augmenter la probabilité d'atteindre la solution optimale, il est conseillé d'effectuer plusieurs essais en changeant la matrice de départ $\xi_n^{(0)}$. Cette dernière est choisie aléatoirement parmi les points candidats ou fixée par l'expérimentateur à condition que la matrice de dispersion existe ou que le déterminant de la matrice d'information soit positif.

La principale différence entre les algorithmes suivants est le processus des échanges de points que l'on rencontre à chaque itération :

3.1.1 Algorithme DETMAX (Mitchell 1974)

Cet algorithme a été publié par Mitchell en 1974 et est un exemple typique de l'algorithme de catégorie 1, avec une matrice de départ aléatoire de n points. L'algorithme cherche à améliorer le déterminant de la matrice d'information en ajoutant ou en supprimant un point. L'expérience d'ajouter $x^{(j)}$ est celui avec la plus grande variance de prédiction $d(x^{(j)})$. Cette expérience conduit à l'augmentation du déterminant. Le résultat d'un tel procédé d'échange est un déterminant qui est supérieur ou égal au précédent. Il comporte deux étapes pour effectuer l'échange :

- Ajouter un point $x^{(j)}$ qui maximise le déterminant.
- Supprimer le point $x_{(i)}$ qui entraîne la diminution minimale.

Les itérations s'arrêtent à partir du moment où le critère choisi n'augmente plus pour l'excursion maximum[10].

Remarque 3 *Pour gagner du temps pendant les calculs, on évalue le déterminant associe à chaque échange par le calcul de la fonction de variance $d(\xi_n, x)$ en chaque point candidat.*

Théorème 4 *[5]Les matrices d'informations $\left(X_n^t.X_n\right)^{(t)}$ et $\left(X_{n+1}^t.X_{n+1}\right)^{(t+1)}$ sont liées par la relation suivant après l'ajout de l'expèrience j qui provoque l'augmentation maximale*

du déterminant :

$$\left(X_{n+1}^{t}.X_{n+1}\right)^{(t+1)} = \left(X_{n}^{t}.X_{n}\right)^{(t)} + f\left(x^{(j)}\right).f^{t}\left(x^{(j)}\right) \tag{3.1}$$

De même les déterminants sont liés par :

$$\left|X_{n+1}^{t}.X_{n+1}\right|^{(t+1)} = \left|X_{n}^{t}.X_{n}\right|^{(t)} * \left|1 + d\left(\xi_{n}, x^{(j)}\right)\right| \tag{3.2}$$

Avec :

$$d\left(\xi_{n}, x^{(j)}\right) = f^{t}\left(x^{(j)}\right).\left(\left(X_{n}^{t}.X_{n}\right)^{(t)}\right)^{-1}.f\left(x^{(j)}\right) \tag{3.3}$$

Après la supperssion de l'expèrience i qui provoque la diminution minimale du déterminant, les matrices d'informations $\left(X_{n+1}^{t}.X_{n+1}\right)^{(t+1)}$ *et* $\left(X_{n}^{t}.X_{n}\right)^{(t+2)}$ *sont liées par la relation:*

$$\left(X_{n}^{t}.X_{n}\right)^{(t+2)} = \left(X_{n+1}^{t}.X_{n+1}\right)^{(t+1)} - f\left(x_{(i)}\right).f^{t}\left(x_{(i)}\right) \tag{3.4}$$

De même les déterminants sont liés par :

$$\left|X_{n}^{t}.X_{n}\right|^{(t+2)} = \left|X_{n+1}^{t}.X_{n+1}\right|^{(t+1)} * \left|1 - d\left(\xi_{n+1}, x_{(i)}\right)\right| \tag{3.5}$$

Avec :

$$d\left(\xi_{n+1}, x_{(i)}\right) = f^{t}\left(x_{(i)}\right).\left(\left(X_{n+1}^{t}.X_{n+1}\right)^{(t+1)}\right)^{-1}.f\left(x_{(i)}\right) \tag{3.6}$$

Preuve. *D'après l'ajout de la ligne* $x^{(j)}$ *à la matrice d'expèrience* X_n *, on trouve une matrice à* $n+1$ *lignes donc :*

$$X_{n+1} = \begin{bmatrix} X_{n} \\ f^{t}\left(x^{(j)}\right) \end{bmatrix}$$

Alors :

$$\begin{aligned} X_{n+1}^{t}.X_{n+1} &= \begin{bmatrix} X_{n} \\ f^{t}\left(x^{(j)}\right) \end{bmatrix}^{t} . \begin{bmatrix} X_{n} \\ f^{t}\left(x^{(j)}\right) \end{bmatrix} \\ &= \begin{bmatrix} X_{n}^{t} & f\left(x^{(j)}\right) \end{bmatrix} . \begin{bmatrix} X_{n} \\ f\left(x^{(j)}\right) \end{bmatrix} \\ &= X_{n}^{t}.X_{n} + f\left(x^{(j)}\right).f^{t}\left(x^{(j)}\right) \end{aligned}$$

D'où la formule (3.1) *.D'après cette derniére on a :*

$$\begin{aligned}\left(X_{n+1}^t.X_{n+1}\right)^{(t+1)} &= \left(X_n^t.X_n\right)^{(t)} + f\left(x^{(j)}\right).f^t\left(x^{(j)}\right) \\ &= \left[I_p + f\left(x^{(j)}\right).f^t\left(x^{(j)}\right).\left(\left(X_n^t.X_n\right)^{(t)}\right)^{-1}\right].\left(X_n^t.X_n\right)^{(t)}\end{aligned}$$

Donc :

$$\begin{aligned}\left|X_{n+1}^t.X_{n+1}\right|^{(t+1)} &= \left|\left[I_p + f\left(x^{(j)}\right).f^t\left(x^{(j)}\right).\left(\left(X_n^t.X_n\right)^{(t)}\right)^{-1}\right].\left(X_n^t.X_n\right)^{(t)}\right| \\ &= \left|\left(X_n^t.X_n\right)^{(t)}\right|.\left|I_p + f\left(x^{(j)}\right).f^t\left(x^{(j)}\right).\left(\left(X_n^t.X_n\right)^{(t)}\right)^{-1}\right| \\ &= \left|\left(X_n^t.X_n\right)^{(t)}\right|.\Big|I_p + f\left(x^{(j)}\right).f^t\left(x^{(j)}\right).\left(\left(X_n^t.X_n\right)^{(t)}\right)^{-1} \\ &\quad .\left(f\left(x^{(j)}\right).f^t\left(x^{(j)}\right)\right).\left(f\left(x^{(j)}\right).f^t\left(x^{(j)}\right)\right)^{-1}\Big| \\ &= \left|\left(X_n^t.X_n\right)^{(t)}\right|.\Big|I_p + f\left(x^{(j)}\right).\left[f^t\left(x^{(j)}\right).\left(\left(X_n^t.X_n\right)^{(t)}\right)^{-1}.f\left(x^{(j)}\right)\right] \\ &\quad .f^t\left(x^{(j)}\right).\left(f\left(x^{(j)}\right).f^t\left(x^{(j)}\right)\right)^{-1}\Big|\end{aligned}$$

D'après (3.3) *on trouve :*

$$\begin{aligned}\left|X_{n+1}^t.X_{n+1}\right|^{(t+1)} &= \left|\left(X_n^t.X_n\right)^{(t)}\right|.\Big|I_p + f\left(x^{(j)}\right).d\left(\xi_n, x^{(j)}\right).f^t\left(x^{(j)}\right) \\ &\quad .\left(f\left(x^{(j)}\right).f^t\left(x^{(j)}\right)\right)^{-1}\Big| \\ &= \left|\left(X_n^t.X_n\right)^{(t)}\right|.\Big|\left[I_p + f\left(x^{(j)}\right).d\left(\xi_n, x^{(j)}\right).f^t\left(x^{(j)}\right)\right] \\ &\quad .\left(f\left(x^{(j)}\right).f^t\left(x^{(j)}\right)\right)^{-1}\Big| \\ &= \left|\left(X_n^t.X_n\right)^{(t)}\right|.\Big|\left[I_p + d\left(\xi_n, x^{(j)}\right).f\left(x^{(j)}\right).f^t\left(x^{(j)}\right)\right] \\ &\quad .\left(f\left(x^{(j)}\right).f^t\left(x^{(j)}\right)\right)^{-1}\Big| \\ &= \left|\left(X_n^t.X_n\right)^{(t)}\right|.\left|\left[1 + d\left(\xi_n, x^{(j)}\right)\right].I_p\right| \\ &= \left|\left(X_n^t.X_n\right)^{(t)}\right|.\left|1 + d\left(\xi_n, x^{(j)}\right)\right|.|I_p| \\ &= \left|\left(X_n^t.X_n\right)^{(t)}\right|.\left|1 + d\left(\xi_n, x^{(j)}\right)\right|\end{aligned}$$

D'où la formule (3.2) . ■

On fait la même chose pour montrer les formules (3.4) *et* (3.5) .

Cet algorithme serait l'algorithme de Wynn-Mitchell de 1972, mais il a modifié en 1974 cette approche pour obtenir une plus grande flexibilité. Dans ce cas, une excursion signifie que le plan de $n+1$ points ne doit pas être réduit immédiatement à un plan de n points, mais peut devenir un modèle à $n+2$ points. Par conséquent, le remplacement de plus d'un point dans le plan de départ peut être possible au cours d'une itération. La limite des excursions est définie par Mitchell (1974) pour être $k = 6$. En considérant un plan avec n meilleurs points actuels, l'algorithme ajoute ou supprime jusqu'à ce que la limite de déplacement soit atteinte. La taille du plan créé varie de $n - k$ à $n + k$. Si aucune amélioration dans le déterminant n'est trouvée au cours de cette excursion, tous les plans créés sont enregistrés dans une liste F qui contient les plans d'échec. Cet ensemble F est utilisé pour la prochaine excursion où l'on considère deux règles différentes définies par Mitchell, soit D le plan actuel, à tout moment au cours d'une excursion, les règles à utiliser sont les suivantes[15]:

1. Si le nombre de points dans D est supérieur à n, il faut soustraire un point si D n'est pas dans F et ajouter un point autrement.

2. Si le nombre de points dans D est inférieur à n, ajouter un point si D n'est pas dans F et soustraire une autre point.

L'algorithme suivant montre que l'utilisation de ces règles d'excursions et la visualisation de flux de l'algorithme avec une notion de programmation simple et abstraite.

Remarque 5 *Si l'ensemble F est vide, l'algorithme ajoute simplement et soustrait des points, tels que décrit dans le premier paragraphe. Considérant les plans d'échecs des itérations précédentes, l'algorithme procède toujours dans le sens d'un plan à n points dans F, qui a déjà conduit à l'échec, puis il change de direction (Mitchell 1974). Si une excursion améliore le plan, les plans d'échec dans F sont supprimés et un nouveau départ est effectué (d'Aguiar et al. 1995, Mitchell 1974).*

Algorithm 6 *[10]*

Générer un plan aléatoire avec un nombre de points n désiré ;

Tant que la limite de déplacement n'est pas atteinte **faire**

Si le nombre de candidats $(N-n)$ est égale au nombre de points souhaités n **alors**

ajouter ou supprimer un point aléatoirement ;

Si non Si le nombre de candidat $(N-n)$ est supérieur aux nombres de points souhaités n **alors**

Si le nouveaux plan n'est pas à l'intérieur de l'ensemble de plan d'échec **alors**

supprimer le point candidat dont sa variance de prédiction est la plus petite ;

Si non

ajouter le point candidat dont sa variance de prédiction est plus grande ;

Si non Si le nombre de candidats $(N-n)$ est inférieur au nombre de points souhaités n **alors**

Si le nouveaux plan n'est pas à l'intérieur de l'ensemble de plan d'échec **alors**

ajouter le point candidat dont sa variance de prédiction est plus grande ;

Si non

supprimer le point candidat dont sa variance de prédiction est la plus petite ;

Si il n y a aucune amélioration sur le déterminant **alors**

sauvegarder le plan dans la liste d'échec ;

Si non

garder la liste d'échec ;

Algorithme 3.1 : Algorithme DETMAX (Mitchell-1974).

3.1.2 Algorithme de FEDOROV 1972

C'est un algorithme d'échange simultané qui garde toujours la taille n du plan désiré sans excursion. Après la génération d'un plan de départ aléatoire, l'algorithme sélectionne un point $x_{(i)}$ du plan et il doit être éliminé par un point $x^{(j)}$ de l'ensemble candidat. La procédure d'ajout et la suppression d'un point est faite en une seule étape et peut être démontrée par l'utilisation de la matrice d'information. En référence à l'équation (3.1) et

(3.4), un échange est une addition et une soustraction simultanée d'un point[10].

Théorème 7 *[5]Les matrices d'informations* $\left(X_n^t.X_n\right)^{(t+1)}$ *et* $\left(X_n^t.X_n\right)^{(t)}$ *sont liées par la relation :*

$$\left(X_n^t.X_n\right)^{(t+1)} = \left(X_n^t.X_n\right)^{(t)} + f\left(x^{(j)}\right).f^t\left(x^{(j)}\right) - f\left(x_{(i)}\right).f^t\left(x_{(i)}\right) \tag{3.7}$$

De même pour les déterminants :

$$\left|\left(X_n^t.X_n\right)^{(t+1)}\right| = \left|\left(X_n^t.X_n\right)^{(t)}\right| * \left|1 + \Delta\left(\xi_n^{(t)}, x^{(j)}, x_{(i)}\right)\right| \tag{3.8}$$

Avec :

$$\begin{aligned}\Delta\left(\xi_n^{(t)}, x^{(j)}, x_{(i)}\right) &= d\left(\xi_n^{(t)}, x^{(j)}\right) - \left(d\left(\xi_n^{(t)}, x_{(i)}\right) * d\left(\xi_n^{(t)}, x^{(j)}\right) - d^2\left(\xi_n^{(t)}, x^{(j)}, x_{(i)}\right)\right) \\ &\quad -d\left(\xi_n^{(t)}, x_{(i)}\right)\end{aligned} \tag{3.9}$$

$$d\left(\xi_n^{(t)}, x^{(j)}\right) = f^t\left(x^{(j)}\right).\left(\left(X_n^t.X_n\right)^{(t)}\right)^{-1}.f\left(x^{(j)}\right) \tag{3.10}$$

$$d\left(\xi_n^{(t)}, x_{(i)}\right) = f^t\left(x_{(i)}\right).\left(\left(X_n^t.X_n\right)^{(t)}\right)^{-1}.f\left(x_{(i)}\right) \tag{3.11}$$

$$d\left(\xi_n^{(t)}, x^{(j)}, x_{(i)}\right) = f^t\left(x^{(j)}\right).\left(\left(X_n^t.X_n\right)^{(t)}\right)^{-1}.f\left(x_{(i)}\right) \tag{3.12}$$

L'idée de base de l'algorithme de Fedorov est de calculer la valeur de Δ pour tous les couples possibles $\left(x_{(i)}, x^{(j)}\right)$ et sélectionner l'une avec la plus grande valeur. Le point $x_{(i)}$ est pris à partir d'une sélection et $x^{(j)}$ peut être soit pris à partir des points restants qu'on appelle une recherche épuisante qui évite les points doubles dans le plan $n*(N-n)$ ou bien avec une recherche non épuisante, la sélection d'une expérience deux fois est possible $n*(N-1)$. Comme l'équation (3.8) montre, les points avec la plus haute Δ-valeurs augmentent la valeur du déterminant. De plus si plus un couple avec la même Δ-valeur est trouvé, l'algorithme choisit parmi eux d'une manière aléatoire[11].

Tandis que si un couple avec un effet Δ-valeur positif est trouvé, l'algorithme change les points et met à jour la matrice d'information et de dispersion. Parfois, on se trouve devant un couple qui augmente le déterminant mais pas d'une manière significative. Pour cela Fedorov a défini une valeur seuil ϵ_{fed}, si le maximum Δ-valeur est plus petit que ϵ_{fed} alors on arrête l'échange.En général on prend comme ϵ_{fed} la valeur 10^{-6}(Aguitar et All).

Algorithm 8 *[10]*

Générer un plan aléatoire avec un nombre de points désiré n ;

Tant que des couples avec Δ positive sont trouvés **alors**

Pour $i = 1$ à n **faire**

Calculer la variance de prédiction $d\left(\xi_n^{(t)}, x_{(i)}\right)$ pour le point d'appui $x_{(i)}$de plan ;

Pour $j = 1$ à N **faire**

Calculer la variance de prédiction $d\left(\xi_n^{(t)}, x^{(j)}\right)$ pour le point d'appui $x^{(j)}$;

Calculer la fonction de prédiction $d\left(\xi_n^{(t)}, x^{(j)}, x_{(i)}\right)$ pour le couple $\left(x_{(i)}, x^{(j)}\right)$;

Calculer la fonction $\Delta\left(\xi_n^{(t)}, x^{(j)}, x_{(i)}\right)$ pour le couple $\left(x_{(i)}, x^{(j)}\right)$;

Vérifier si le delta est maximum et enregistrer le couple ;

Si le maximum des deltas est positif **alors**

Si plus qu'un couple avec le même delta maximal **alors**

|Sélectionner aléatoirement un couple ;

Changer le point $x_{(i)}$ sélectionné par $x^{(j)}$;

Mettre à jour la matrice d'information et la matrice de dispersion ;

Réinitialiser delta maximal ;

Algorithme 3.2 : Algorithme de Fedorov (1972).

3.1.3 Algorithme de Fedorov Modifié

Cook et Nachtsheim 1980 font une comparaison de différents algorithmes pour construire des plans D-optimaux exacts et ils ont inventé un algorithme propre en fonction de l'algorithme de Fedorov. Ce dernier détermine la valeur de Δ pour tous les couples d'échanges possibles pendant une itération mais n'utilise qu'une seule valeur pour effectuer un échange. Ce calcul est couteux lors de son utilisation . Avec la version modifiée par Cook et Nachtsheim, chaque itération de l'algorithme de Fedorov est décomposée en n étapes, une pour chaque point d'appui dans le plan au début de l'itération. Avec une matrice de conception d'un ordre aléatoire, l'algorithme commence avec le premier point d'appui $x_{(i)}$ et calcule les Δ-valeurs pour tous les couples possibles avec ce point d'appui. Après avoir

trouvé le meilleur échange de ce point, le plan est mis à jour et le point d'appui prochain est proposé pour un échange. Ce comportement est visualisé dans l'algorithme (3.3).

La différence entre les deux approches peut être montré par l'exemple suivant : considérons un plan souhaité avec $n = 5$ points et un ensemble de candidats $N = 20$ expériences, $n * N = 100$ couples possibles peuvent être proposés pour un échange. A chaque itération de l'algorithme de Fedorov on calcule ces 100 valeurs de Δ pour tous les couples possibles et on utilise un seul d'entre eux pour réaliser l'échange. En revanche, la version modifiée de l'algorithme commence par le premier point de plan et on ne calcule que 20 Δ-valeurs pour les couples possibles y compris ce point. Après cela, un échange est effectué et la matrice de dispersion doit être mise à jour. L'algorithme se poursuit avec le point du plan suivant et on calcule à nouveau les 20 valeurs du Δ. En général, à la fois l'algorithme calcule les 100 valeurs au cours d'une itération, la version d'échange calcule jusqu' à 5 points au cours de cette procédure.

Une étude réalisée par Cook et Nachtsheim montre que l'approche modifiée est peut être deux fois plus vite que l'algorithme de Fedorov[10].

Algorithm 9 *[10]*

Générer un plan aléatoire avec un nombre de points désiré n ;

Tant que des couples avec Δ positif sont trouvés **alors**

  Pour $i = 1$ à n **faire**

    Calculer la variance de prédiction $d\left(\xi_n^{(t)}, x_{(i)}\right)$ pour le point d'appui $x_{(i)}$;

    Pour $j = 1$ à N **faire**

      Calculer la variance de prédiction $d\left(\xi_n^{(t)}, x^{(j)}\right)$ pour le point $x^{(j)}$;

      Calculer la fonction de prédiction $d\left(\xi_n^{(t)}, x^{(j)}, x_{(i)}\right)$ pour le couple $\left(x_{(i)}, x^{(j)}\right)$;

      Calculer la fonction $\Delta\left(\xi_n^{(t)}, x^{(j)}, x_{(i)}\right)$ pour le couple $\left(x_{(i)}, x^{(j)}\right)$;

      Sélectionner le couple avec un delta maximal ;

    Si le maximum des deltas est positif **alors**

      Si plus qu'un couple avec le même delta maximal **alors**

      [Sélectionner aléatoirement un couple ;

Changer le point $x_{(i)}$ sélectionné par $x^{(j)}$;

Mettre à jour la matrice d'information et la matrice de dispersion ;

Réinitialiser delta maximal ;

Algorithme 3.3 : Algorithme de Fedorov modifié par Cook & Nachtsheim (1980).

3.1.4 Algorithme k-échange

Au cours d'une comparaison de l'algorithme de Fedorov et l'algorithme de Mitchell, Johnson et Nachtsheim 1983 ont montré que les points sélectionnés par l'algorithme de suppression de Fedorov ne sont normalement pas les uns avec la plus faible variance de prédiction, mais la fréquence des rangs de points supprimés pourrait être caractérisée comme biaisé en faveur des rangs de la variance inferieures. Simplement dit, au lieu de considérer tous les points candidats du plan ou seulement l'un avec la plus petite variance de prédiction, un ensemble de k points ayant la plus faible variance devait être sélectionnés. Similaire à l'algorithme de Fedorov modifié, une itération est maintenant décomposée en k étapes. A l'intérieur de chacune des ces k étapes les Δ-valeurs sont calculées et le couple correspondant serait échangé si le déterminant augmente.

La similitude avec les algorithmes précédemment décrits peuvent reconnue si nous définissons k-valeurs. Avec $k = 1$ l'algorithme est de Mitchell et $k = n$ il devient celui de Fedorov modifié.

La sélection des k-valeurs est difficile et dans la plupart des cas dépendent de problème proposé. Une valeur commune qui est également suggéré par Johnson et Natchtsheim, est $k = 3$ ou $k = 4$. Meyer et Nachtsheim 1995 plus tard ont conseillé de sélectionner la valeur tel que[10]:

$$k \leq \frac{n}{4}$$

Considérons l'exemple avec $n = 5$ et $N = 20$, le k-échange accélère le processus de sélection en réduisant le nombre compté pour le calcul de Δ. Similaire à la procédure de Fedorov modifiée, chaque itération de l'algorithme de base Fedorov est décomposé en Δ-calculs pour chaque point de plan. Mais en fonction de k-valeurs définis, l'algorithme tient

compte que certains points de plan. Avec $k = 3$ l'algorithme k-échange calcule trois fois les 20 Δ-valeur et effectue jusqu'à trois échanges.

Cette approche a une plus grande efficacité de calcul par rapport à la procédure de Fedorov modifiée. En particulier pour les données de taille n grande, la définition de k-échange a un impact. La quantité du plan n'est pas toujours aussi bonne que celle de l'algorithme de Fedorov normale, mais en ce qui concerne le temps de calcul nécessaire, l'algorithme donne de bons résultats[10].

Algorithm 10 *[10]*

Générer un plan aléatoire avec un nombre de points désiré n ;

Tant que des couples avec Δ positive sont trouvés **alors**

- Calculer la variance de prédiction $d\left(\xi_n^{(t)}, x_{(i)}\right)$ pour tous les points de plan ;
- Sélectionner k points de plan avec la variance de prédiction la plus faible ;
- **Pour** $i = 1$ à k **faire**
  - Calculer la variance de prédiction $d\left(\xi_n^{(t)}, x_{(i)}\right)$ pour le point $x_{(i)}$;
  - **Pour** $j = 1$ à N **faire**
    - Calculer la variance de prédiction $d\left(\xi_n^{(t)}, x^{(j)}\right)$ pour le point $x^{(j)}$;
    - Calculer la fonction de prédiction $d\left(\xi_n^{(t)}, x^{(j)}, x_{(i)}\right)$ pour le couple $\left(x_{(i)}, x^{(j)}\right)$;
    - Calculer la fonction $\Delta\left(\xi_n^{(t)}, x^{(j)}, x_{(i)}\right)$ pour le couple $\left(x_{(i)}, x^{(j)}\right)$;
    - Sélectionner le couple avec un delta maximal ;
  - **Si** le maximum des deltas est positif **alors**
    - **Si** plus qu'un couple avec le même delta maximal **alors**
    - [Sélectionner aléatoirement un couple ;
    - Changer le point $x_{(i)}$ sélectionné par $x^{(j)}$;
    - Mettre à jour la matrice d'information et la matrice de dispersion ;
    - Réinitialiser delta maximal ;

Algorithme 3.4 : Algorithme k-échange (Johnson & Nachisheim (1980)).

3.1.5 Algorithme kl-échange

C'est une modification de l'algorithme de Fedorov et donc un type du groupe 2 développé par Atkinson et Donev en 1989, cet algorithme permet de réduire la liste des points d'appuis et d'échanges avant de calculer la valeur Δ pour tous les couples possibles. L'utilisation du k points ayant la plus faible variance de prédiction est similaire à la procédure de k-échange. En plus de cela, nous ne considérons que les l candidats avec la plus grande variance de prédiction entre les points d'appuis. En définissant $k = n$ et $l = N$, cet algorithme est la procédure de Fedorov. A l'initialisation normale de l'algorithme avec $k < n$ et $l < N$ réduit la quantité de calculer Δ et d'accélérer l'algorithme. Comme l'algorithme de Fedorov, le kl-échange sélectionne les points dont la valeur Δ est la plus grande et effectue un seul échange, nous devons observer que les kl procédures s'appuient sur l'algorithme de Fedorov et n'effectue pas l'échange multiple au cours d'une itération. Seule la notion de choisir le point k de la plus faible variance de prédiction est similaire à l'algorithme de k-échange.

Atkinson et Donev 1989 ont défini deux modifications de cet algorithme : la première est semblable à l'idée de base de l'algorithme de Fedorov modifié par Cook et Nashtcheim et réalise plus d'un échange au cours de chaque itération. Avec la sélection de $l = 1$, cela a changé kl-échange et devient soit l'algorithme de Fedorov modifie avec $k = n$ ou k-échange avec $k < n$. La seconde modification modifie les critères de sélection pour les points k et l. Au lieu d'un choix dépend de la variance, l'algorithme sélectionne de façon aléatoire parmi les points de plan et les points d'appuis.

Dans l'algorithme (3.5), la version de base est montrée sans aucune modification. Par rapport à l'algorithme de Fedorov, le kl-échange réduit la quantité de calcul de Δ entre chaque itération. Pour notre exemple $n = 5, N = 20, k = l = 3$ conduit au calcul de seulement $k*l = 3*3 = 9$ valeurs de Δ. En comparaison avec les 100 couples de l'algorithme de Fedorov, cette approche kl accélère le calcul d'une manière efficace. Certes nous devons observer que la quantité d'itérations effectué dans kl échange peut être plus élevée par ce que tous les couples sont pris en considération pour un échange, mais en général l'algorithme kl échange crée des modèles très rapides et dans la plupart des cas donnent des résultats acceptables compte tenu de la D-optimalité[10].

Algorithm 11 *[10]*

Générer un plan aléatoire avec un nombre de points désiré n ;

Tant que des couples avec Δ positif sont trouvés **alors**

Calculer la variance de prédiction $d\left(\xi_n^{(t)}, x_{(i)}\right)$ pour tous les points de plan ;

Sélectionner k points de plan avec la variance de prédiction la plus faible ;

Sélectionner l points de plan avec la variance de prédiction la plus grande ;

Pour $i = 1$ à k **faire**

Calculer la variance de prédiction $d\left(\xi_n^{(t)}, x_{(i)}\right)$ pour le point $x_{(i)}$;

Pour $j = 1$ à l **faire**

Calculer la variance de prédiction $d\left(\xi_n^{(t)}, x^{(j)}\right)$ pour le point d'appui $x^{(j)}$;

Calculer la fonction de prédiction $d\left(\xi_n^{(t)}, x^{(j)}, x_{(i)}\right)$ pour le couple $\left(x_{(i)}, x^{(j)}\right)$;

Calculer la fonction $\Delta\left(\xi_n^{(t)}, x^{(j)}, x_{(i)}\right)$ pour le couple $\left(x_{(i)}, x^{(j)}\right)$;

Vérifier si le delta est le maximum et enregistrer le couple ;

Si le maximum des deltas est positif **alors**

Si plus qu'un couple avec le même delta maximal **alors**

Sélectionner aléatoirement un couple ;

Changer le point $x_{(i)}$ sélectionné par $x^{(j)}$;

Mettre à jour la matrice d'information et la matrice de dispersion ;

Réinitialiser delta maximal ;

Algorithme 3.5 : Algorithme kl-échange (Atkinson & Donev (1989)).

3.1.6 Algorithme kl-échange modifié

Une modification de l'algorithme kl-échange est développée. Le changement dans l'algorithme est d'étendre la valeur k si les k points sélectionnés ont la même variance de prédiction et d'un analogue $(k+1)^{ième}$ valeurs existent. En utilisant une sélection aléatoire si plus de k candidats ont la même variance, alors elle peut aussi être une solution, mais elle n'est pas aussi infiable. La deuxième modification tente d'empêcher l'algorithme kl-échange de l'échange de certains couples qui ne sont pas les meilleurs choix en réduisant la liste des

points d'appuis.Commençant par $l = N$ points dans l'ensemble de points d'appuis possibles, l'algorithme supprime les points en fonction de critères différents. En règle générale, un échange ne doit être effectué si la variance de prédiction $d\left(\xi_n^{(t)}, x^{(j)}\right)$ du point de support $x^{(j)}$ est supérieure à la variance de prédiction $d\left(\xi_n^{(t)}, x_{(i)}\right)$ du point de conception $x_{(i)}$ supprimé. Un deuxième critère de réduire la liste des points de support est développé dépend de la variance moyenne de la prédiction à partir de la dernière itération. Seuls les points d'appuis avec une variance plus élevée de prédiction que la variance moyenne est prise en compte. Dans ce cas, la variance moyenne de prédiction est calculée avec l'emploi de tous les candidats possibles. Si l'algorithme mène à une valeur maximale parmi ces meilleurs points d'appuis possibles, le support restant des points est vérifié. Cet algorithme trouve des difficultés pour comparer les approches précédentes parce que le montant du point de conception et de soutien compte pour un échange est dépendant de problème. Nous ne faisons pas valeurs pour l et k et les critères pour la sélection des points peuvent changer au cours du processus, mais en général, la modification kl-échange est comparable avec kl-échange normal. Les calculs doivent un peu plus de temps parce que nous changeons parfois les critères de la variance et on calcule le meilleur échange. Mais dans l'ensemble, l'algorithme est comparable au kl-échange normal. Compte tenu de la qualité des motifs générés, cet algorithme conduit aux meilleurs résultats dans la plupart des cas, vaut également la procédure k-échange.

Bien sûr, les résultats n'ont pas la même qualité que les dessins créés avec l'Algorithme de Fedorov, mais avec un temps de calcul recevable, nous créons une bonne D-optimalité.

Algorithm 12 *[10]*

Générer un plan aléatoire avec un nombre de points désiré n ;

Tant que des couples avec Δ positive sont trouvés **alors**

Calculer la variance de prédiction $d\left(\xi_n^{(t)}, x_{(i)}\right)$ pour tous les points de plan ;

Si plus de k points de plan ont la même faible variance de prédiction **alors**

⌊étant donné k;

sélectionner k points de conception avec la plus basse variance de prédiction ;

calculer la variance de prédiction $d\left(\xi_n^{(t)}, x^{(j)}\right)$ pour tous les points d'appuis ;

calculer la variance moyenne de prédiction sur l'ensemble des points candidats ;

Sélectionner l points d'appuis avec une plus grande variance de prédiction que la moyenne valeur ;

Pour $i = 1$ à k **faire**

Calculer la variance de prédiction $d\left(\xi_n^{(t)}, x_{(i)}\right)$ pour ce point de conception ;

Pour $j = 1$ à l **faire**

calculer la variance de prédiction $d\left(\xi_n^{(t)}, x^{(j)}\right)$ pour ce point d'appui ;

Si la variance du point d'appui est supérieure à la variance du point de conception **alors**

calculer la fonction de prédiction $d\left(\xi_n^{(t)}, x^{(j)}, x_{(i)}\right)$ pour ce couple ;

calculer la fonction delta $\Delta\left(\xi_n^{(t)}, x^{(j)}, x_{(i)}\right)$ pour ce couple ;

Si delta maximal est négatif **alors**

Sélectionner $N - l$ points d'appui qui n'ont pas été vérifiés ;

Pour $i = 1$ à k **faire**

calculer la variance de prédiction $d\left(\xi_n^{(t)}, x_{(i)}\right)$ pour ce point de conception;

Pour $j = 1$ à $N - l$ **faire**

calculer la variance de prédiction $d\left(\xi_n^{(t)}, x^{(j)}\right)$ pour ce point d'appui ;

calculer la fonction de variance $d\left(\xi_n^{(t)}, x^{(j)}, x_{(i)}\right)$ pour ce couple ;

calculer la fonction delta $\Delta\left(\xi_n^{(t)}, x^{(j)}, x_{(i)}\right)$ pour ce couple ;

Si delta maximal est positif **alors**

Si plus qu'un couple avec le même delta maximal **alors**

Sélectionner aléatoirement un couple ;

Changer le point $x_{(i)}$ sélectionné par $x^{(j)}$;

Mettre à jour la matrice d'information et la matrice de dispersion ;

Réinitialiser delta maximal ;

Algorithme 3.6 : Algorithme kl-échange modifié.

3.1.7 Algorithme FDOP (Yonchev)

Suivant l'échange proposé par FEDOROV on remarque que le point choisi n'est pas toujours le point ayant la fonction de variance la plus élevée car il cherche le meilleur couple. Pour cette raison Yonchev évite la première étape en ajoutant aléatoirement M points candidats pour en extraire par la suite successivement les M mauvais points[5].

- $\xi_n^{(t)}$ (ajout de M points) $\rightarrow \xi_{n+M}^{(t+1)}$.
- $\xi_{n+M}^{(t+1)}$ (Suppression séquentielles de M points donnant la diminution minimale) $\rightarrow$ $\xi_n^{(t+2)}$.

Le critère d'arrêt est le nombre $maxi$ d'excursion n'ayant pas modifié la structure de ξ_n. L'expérience montre que prendre $maxi \leq 4$ suffit pour obtenir des résultats comparables à ceux obtenus par les algorithmes de Mitchell et Fedorov.

L'amélioration des temps de calcul dépend du choix de M. Plus M est petit, plus le traitement d'un lot est rapide, mais plus il y a de lot à traiter.

L'étude de l'influence de M montre que les meilleurs résultats sont obtenus pour des valeurs de M comprises dans l'intervalle $[p/2, 3p]$ où p est le nombre de paramètre du modèle.

Pour cet algorithme, nous notons $X_k^{(i)j}$: la matrice du modèle comportant k expériences testées durant l'excursion longue i et l'excursion courte j.

D'où l'algorithme proposé par Yonchev :

Algorithm 13 *[5]*

Initialiser la matrice de départ $X_n^{(0)0}$

Tant que (nombre d'excursion $< maxi$) **alors**

Initialiser des lots

Créer R lots contenant aléatoirement M points de la liste des expériences candidates.

Exécution courte

Pour $j = 1$ à R lots faire

Calculer le déterminant et la matrice de dispersion de $X_n^{(t)0}$

Ajouter les M points du $j^{ème}$ lot $\rightarrow X_{n+M}^{(t)j}$

Caractériser la nouvelle matrice

Calculer le déterminant et la matrice de dispersion

Eliminer les M points mauvais

Tant que (nombre de points dans la matrice $> n$) **alors**

Calculer les variances de prédiction

Le point dont la variance de prédiction est minimale est retiré de la matrice.

Validation d'une excursion courte

Si $\left(\left|\left(X_n^t.X_n\right)^{(t)0}\right| < \left|\left(X_n^t.X_n\right)^{(t)j}\right|\right)$ **alors**

$X_n^{(t)0} = X_n^{(t)j}$

Validation d'une excursion

Si $\left(\left|\left(X_n^t.X_n\right)^{(t-1)0}\right| < \left|\left(X_n^t.X_n\right)^{(t)0}\right|\right)$ **alors**

Incrémenter le nombre d'excursions

Sinon

Le nombre d'excursions $= 0$

Algorithme 3.7 : Algorithme de FDOP.

3.2 Application des algorithmes d'échanges sur la fabrication pharmaceutique d'un médicament bien protégé

3.2.1 Préparation de l'experimentation

Une société pharmaceutique fabrique un médicament très actif, mais malheureusement sensible à la température et à l'humidité. Pour augmenter la durée de conservation, on ajoute un desséchant au conditionnement du médicament. Un technicien est chargé d'optimiser la quantité de desséchant qu'il faut introduire. Le matériel dont il dispose pour réaliser son étude lui impose des contraintes. Toutes les combinaisons de niveaux ne sont pas possibles. Il ne peut pas, non plus, faire trop d'essais car la fabrication attend des résultats pour démarrer. Pour surmonter toutes ces contraintes, il choisit de conduire ses expériences selon un plan D-optimal.

3.2.2 Description des contraintes

Le médicament à étudier se décompose et perd son activité à la fois sous l'influence de la chaleur mais aussi sous l'influence de l'humidité. Ces deux facteurs ont des actions néfastes qui parfois se renforcent et qui parfois s'annihilent l'une l'autres suivant les conditions de conservation. Le médicament doit être étudié entre 30° à 50° pour la température et entre 50% et 90% pour l'humidité. Le technicien possède deux étuves pour réaliser ces essais. La première étuve couvre les températures de 30° à 40° et les humidités de 50% à 90%. Le second couvre les températures de 40° à 50° et les humidités de 50% à 80%. Il n'est pas possible d'atteindre 90% d'humidité avec cette dernière étuve.

Le médicament se décompose en une substance facile à déterminer et à doser. Cette substance n'est pas toxique, mais elle est inerte. La disparition du médicament principal diminue l'activité thérapeutique du produit vendu et c'est ce qu'il faut éviter.

3.2.3 Choix des facteurs

Le technicien retient 3 facteurs :

1^{er} : La température en $°C$

2^{ieme} : Le pourcentage d'humidité

3^{ieme} : La masse de desséchant en g

3.2.4 Domaine d'étude

Le tableau suivant indique les niveaux hauts et bas de chaque facteur. On sait que le point 50° et 90% d'humidité ne peut pas êtres réalisé avec le matériel disponible.

Facteur	**Niveau -1**	**Niveau 0**	**Niveau +1**
Température (1)	$30°C$	$40°C$	$50°C$
Humidité (2)	50%	70%	90%
Desséchant (3)	$1g$	$2g$	$3g$

Tableau 3.1 : Un médicament bien protégé. Domaine d'étude.

3.2.5 Choix des réponses

La réponse choisie par le technicien est la masse de la substance issue de la décomposition du médicament. Cette substance est facile à isoler et peser avec une bonne précision.

Etant donné la complexité des processus de dégradation du médicament, les modèles des réponses ne sont pas probablement de premier degré. Il faut envisager des modelés du second degré.

3.2.6 Choix du plan expérimental

Le domaine d'étude est tronqué puisqu'il n'est pas possible de réaliser tous les points d'expériences. Le technicien décide de couvrir le domaine accessible avec un grand nombre de points d'expériences et d'utiliser l'algorithme D-optimal pour choisir les plus pertinents.

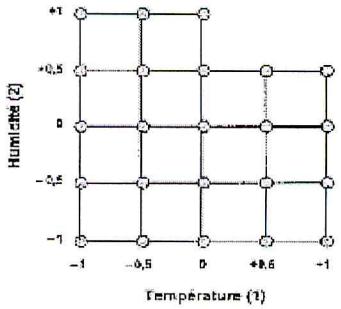


Figure 3.1 : Disposition des 23 points d'experiences candidats pour un niveau du facteur 3. Comme il y a trois niveaux pour ce facteur, il y a 69 points candidats.

Le technicien attribue trois niveaux au facteur 3 (masse de desséchant) sur le quel ne pèse aucune contrainte experimentale. Les facteurs 1 et 2 auront les niveaux $-1, -0.5, 0, +0.5, +1$ à chaque fois que cela sera possible. Pour un niveau du facteur 3 les points d'expériences candidats seront disposés comme l'indique la figure précédente. Il y a 23 points d'expériences candidats par niveau du facteur 3. Il y a donc tout $N = 69$ essais possibles. Mais, comme le travail doit être réalisé le plus vite possible, le technicien fait tourner l'algorithme de calcul des plans D-optimaux afin de choisir les expériences les plus favorables à l'établissement du modèle du second degré. Ce modèle est le modèle complet qui possède dix coefficients. Le technicien demande douze points d'expériences donc $n = 12$.

Pour déterminer la matrice d'expérience X, on va utiliser les algorithmes précédents et puis on calcule le déterminant de $\left(X^tX\right)$ dont les résultats sont dans le tableau (3.2).

ALGORITHMES	$\det\left(X^tX\right)$
DETMAX	430988.5030
FEDOROV	656099.9986
FEDOROV MODIFIE	598181.8757
K-ECHANGE	549042.0004
KL-ECHANGE	542666.4617
FDOP	591258.8599

Tableau 3.2 : Le déterminant de $\left(X^tX\right)$ suivant les algorithme .

Dans cet exemple on remarque que par l'algorithme de Fedorov on trouve le déterminant très élevé d'où l'optimalité.

3.3 Comparaison entre les algorithmes d'échanges

En règle générale, une comparaison des algorithmes qui doit être fait en fonction de deux critères principaux :

Le temps de calcul utilisé pour créer un plan D-optimal, et le second est la qualité de la conception. Dans cette analyse on va comparer les algorithmes suivant l'application précédente, nous utilisons la D-optimalité où son analogue, le facteur déterminant pour évaluer les plans optimaux.

La base des données fournies dans ce travail est une analyse des six algorithmes mis en œuvre avec **MAPEL**. Pendant le déroulement du programme pour calculer le déterminant et le temps de calcul est mesurée, les données recueillies peuvent être trouvés dans le tableau (3.3).

ALGORITHMES	det (X^tX)	Temps d'exécution / s
DETMAX	430988.5030	12.3
FEDOROV	656099.9986	287.2
FEDOROV MODIFIE	598181.8757	100.7
K-ECHANGE	549042.0004	55.3
KL-ECHANGE	542666.4617	15.6
FDOP	591258.8599	34.5

Tableau 3.3 : Comparaison des algorithmes d'échanges.

L'algorithme de Fedorov, par exemple, calcule la valeur pour tous les couples possibles et n'effectue qu'un seul échange en fonction de ces données, avec un grand nombre de facteurs, cette approche est très coûteuse. La modification qui était faite par Cook & Nachtsheim (1980) permet de réduire le nombre de calcul de chaque échange. Le temps de calcul moyen utilisé pour l'algorithme Fedorov modifié peut être rapide jusqu'à 50% que l'algorithme normale, surtout avec beaucoup de données, la différence entre les deux algorithmes devient évidente.

D'autres algorithmes comme le k-échange par Johnson & Nachtsheim (1983) ou kl-échange sont beaucoup plus rapide dans la création des plans D-optimaux et peut être utilisés pour un très grand nombre de facteurs.

Le k-échange est trois fois plus rapide que la procédure de Fedorov. L'algorithme de Mitchell ou l'algorithme DETMAX a un temps de calcul qui est comparable à la k-échange normal. Certes, un algorithme rapide n'est pas utile si le plan créé n'a pas le choix de qualité ou de D-optimalité. Par conséquent, nous devons comparer les algorithmes en fonction du temps utilisé et la qualité de la conception créée.

Tous les algorithmes peuvent piéger dans un optimum dit local et arrêter le processus d'échange, même si la meilleure conception n'est pas encore trouvée. Avec un nombre croissant de facteurs ou termes de modèle la plupart des plans D-optimaux sont différents même si elles reposent sur le même algorithme.

Le Fedorov et Fedorov modifié créent normalement les meilleurs plans, mais en général, la sélection du meilleur algorithme est dépendante de problème. Les meilleures conceptions

sont créées avec le Fedorov ou Fedorov modifié, mais l'algorithme de Wynn-Mitchell doit être utilisé si une génération rapide est plus importante.

Par contre, l'algorithme de FDOP est meilleur que FEDORV, FEDOROV modifié et k-échange par rapport au temps d'excursion et si on compare par rapport au déterminant on trouve qu'il est aussi meilleur que celui de MITCHELL , k-échange et le kl-échange.

CHAPITRE IV

CONSTRUCTION ALGORITHMIQUE DES PLANS D-OPTIMAUX PAR DÉPLACEMENT

Avec le temps, de nouvelles méthodes sont découvertes pour construire les plans D-optimaux après les algorithmes d'échanges dont le principe est d'utiliser les matrices proches de l'optimalité (c.à.d. une matrice déjà générée par un algorithme d'échange) comme une solution de départ (réalisable). Dans ce chapitre on va présenter trois méthodes qui sont les plus utilisables.

4.1 Les algorithmes sans point candidat

Ils utilisent le domaine expérimental dans sa totalité. Les points testés sont générés aléatoirement au cours de la recherche qui est effectuée de plus en plus finement jusqu'à la précision recherchée.

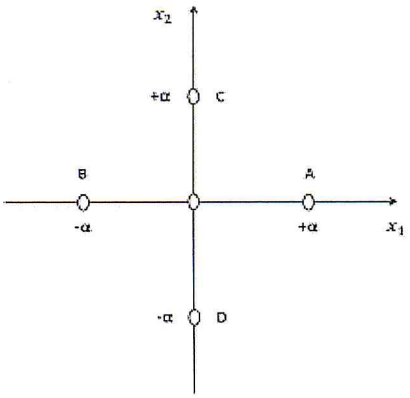
4.1.1 Algorithme de DONEV (Donev et Atkinson)

Cet algorithme propose un moyen d'améliorer des matrices quasi optimales, obtenues par exemple par les algorithmes d'échanges. Il explore les alentours de chacun des points de la matrice, afin d'obtenir une solution plus performante en occupant des positions plus précises.

A partir d'une matrice proche de l'optimalité ils réalisent une amélioration de la position des points sans bouleverser la configuration.

Ils effectuent sur chacun des points de la matrice des variations pour déterminer lesquels des n points doivent être modifié et dans quelle direction. Le principe de cet algorithme consiste à evaluer la variation du déterminant de la matrice d'information en faisant varier une coordonnée d'un point de la matrice, ce qui revient à perturber la matrice d'expérience en modifiant une valeur et en évaluant les effets sur le déterminant de la matrice d'information[5].

Exemple 14 *Dans le cas où notre domaine est défini par deux variables indépendantes, soit x_i le point qui nous désirons faire varier. Nous testerons les quatre points A, B, C et D tels qu'ils sont représentés sur le schéma. Le point x_i est déplacé sur la position qui donne le plus grand déterminant de $\left(X^tX\right)$.*



$A\ (x_{i1}+\alpha, x_{i2})$

$B\ (x_{i1}-\alpha, x_{i2})$

$C\ (x_{i1}, x_{i2}+\alpha)$

$D\ (x_{i1}, x_{i2}-\alpha)$

Figure 4.1 : Déplacements de Donev en 2 dimensions.

L'algorithme calcule ainsi $(2.k)$ déplacements et donc $(2.k)$ fonctions « delta » par étape.

Nous pouvons encore réduire le temps de calcul en utilisant une méthode rapide pour recalculer les termes de $\left(X^tX\right)^{-1}$.

L'algorithme proposé par Donev et Atkinson, très performant sur des matrices proches de l'optimalité-D, est le suivant :

Algorithm 15 *[5]*

Choix d'une amplitude de déplacement α

Calculer la matrice $\left(X^tX\right)$

Tant que $(\alpha \geq \alpha_{\min})$ **alors**

$\Delta_{\max} = 0$

Essayer pour tous les points

Pour $i = 1$ à n **faire**

Essayer tous les changements sur les axes

Pour $j = 1$ à nombre de variables indépendantes **faire**

Essayer *sens* négatif puis positif

Pour $sens = -1$ puis $+1$ **faire**

$D = (d_1, d_2, \ldots, d_n)$ avec $d_j = sens$ et $d_m = 0$ (pour $m \neq j$)

Calculer le nouveau point $x' = x_i + \alpha.D$

Calculer la fonction delta de Fedorov : Δ

Si $(\Delta > \Delta_{\max})$ **alors**

x' est le changement sélectionné

$\Delta_{\max} = \Delta$

Changement ou convergence

Si $(\Delta_{\max} > 1)$ **alors**

Effectuer le meilleur changement sélectionné

Calculer de la nouvelle matrice $\left(X^t X\right)^{-1}$

Sinon

$\alpha = \alpha/2$

Algorithme 4.1 : Algorithme de DONEV.

4.1.2 Algorithme de FDOP modifié

Cette méthode est basée sur le principe de Yonchev et inspirée des méthodes à déplacements aléatoires. Elle provoque des perturbations sur une partie des points pour détecter si une modification aléatoire peut améliorer la solution[5].

Algorithm 16 *[5]*

Initialiserde la matrice de départ $X_n^{(1)0}$

Tant que (nombre d'excutions < 4) **alors**

Créer R lots contenant M numéros de points.

Excursion courte

Pour $j = 1$ à R lots **faire**

Matrice de dispersion de la matrice actuelle $\rightarrow X_n^{(t)0}$

Genérer les M nouveaux points

Pour $i = 1$ à M **faire**

$l =$ numéro du $i^{ème}$ point dans le $j^{ème}$ lot

$D =$ choix du déplacement unitaire est aléatoire

$x_{n+l} = x_l + \alpha.D$

Eliminer les M mauvais points

Tant que $(i \neq 0)$ **alors**

Matrice de dispersion de la nouvelle matrice $\rightarrow X_{n+i}^{(t)j}$

Caculer les variances de prédiction $d\left(x_{n+i}\right)$

Eliminer le point ayant la variance de prédiction minimale

$i = i - 1$

Validation d'une excursion courte

Si $\left(\left|\left(X_n^t.X_n\right)^{(t)0}\right| < \left|\left(X_n^t.X_n\right)^{(t)j}\right|\right)$ **alors**

$X_n^{(t)0} = X_n^{(t)j}$

Validation d'une excursion

Si $\left(\left|\left(X_n^t.X_n\right)^{(t-1)0}\right| < \left|\left(X_n^t.X_n\right)^{(t)0}\right|\right)$ **alors**

Incrémenter nombre d'excursions

Sinon

nombre d'excursions $= 0$

Algorithme 4.2 : Algorithme de FDOP modifié.

4.2 Les algorithmes Méta-heuristiques

Une méthode approchée ou heuristique est un algorithme qui a pour but de trouver une solution réalisable sans garantie d'optimalité[16].

4.2.1 La méthode du recuit simulé

Cette technique provient de l'observation de la formation d'une structure cristalline quand un métal refroidit. Le recuit simulé est une technique permettant de trouver au bout d'un temps raisonnable une solution pour des problèmes de complexité élevées, elle est considérée comme une technique stochastique et ne garantie pas de solution optimale.

Les premiers algorithmes, leurs objectif est de positionner une série de points les uns

par rapport aux autres dans le domaine expérimental en fonction du modèle par contre cet algorithme permet une convergence rapide dés que la correspondance a été obtenue et aussi elle conduit à la meilleure matrice d'expérience[5].

Le principe de cette méthode est que chaque point de la matrice est perturbé par des déplacements aléatoires. Si un déplacement améliore le déterminant il est accepté. En revanche s'il n'entraine pas d'amélioration il n'est pas rejeté et il est accepté avec une probabilité P particulière.

Récuit simulé donne trés souvent les résultats corrects et la matrice est très proche de la D-optimalité.

Algorithm 17 *[5]*

Générer n points aléatoires, choix de α et c

Calculer la matrice $\left(X^t X\right)^{-1}$

Tant que (Amplitude des déplacements ≥ 0.001) **alors**

  Tant que (nombre de boucles $\leq 3 * n$) **alors**

    Déterminant actuel et matrice inverse

    Modification successives des points

    Pour $i = 1$ à n **faire**

      Tant que ($Nbreconvergence < 3$) **alors**

        Tant que (les déplacements testés < 20) **alors**

          Choix d'un déplacement aléatoire D

          $x' = x_i + \alpha.D$

          $\Delta = 1 + d\left(x'\right) - d\left(x_i\right) - d\left(x'\right) * d\left(x_i\right) + d^2\left(x', x_i\right)$

          Si ($\Delta \geq 0$) **alors**

          [Probabilité $= 1$

          Sinon

          [Probabilité $= \exp\left(\Delta / c\right)$

          Si (Probabilité $\geq$ valeur aléatoire) **alors**

          [Le changement effectué

Si (la moyenne du déterminant reste constante) **alors**

[Incrémenter ($Nbreconvergence$)

Sinon

[$Nbreconvergence = 0$

Tester le Pourcentage de changement accepté

De 0% à 20% : Température fortement augmentée ($c * 3$)

De 20% à 40% : Température augmentée ($c * 2$)

De 40% à 60% : Amplitude des déplacements réduite ($\alpha/2$)

De 60% à 80% : Température réduite ($c/2$)

De 80% à 100% : Température fortement réduite ($c/3$)

Algorithme 4.3 : Algorithme du Recuit Simulé.

4.3 Application des algorithmes sur l'étude explosive

Un explosif ne doit pas présenter de risque d'explosion en cas de choc accidentel. Il doit pouvoir être transporté et conservé en toute sécurité. Mais les moyens mis en œuvre pour le rendre sur ne doivent pas diminuer sa puissance explosive. Le formulateur doit trouver le meilleur compromis entre sécurité et efficacité. C'est l'un de ses soucis majeurs. Les compositions explosives ont été classées en plusieurs catégories dont la plus recherché est MDEPS (matière détonante extrêmement peu sensible). De telles compositions sont obtenues en associant judicieusement des composés actifs. L'objectif de l'étude est de trouver une composition moins sensible que la composition H6 réputée trop sensible. Une composition est sensible si elle présente des risques élévés d'explosion spontanée lorsqu'elle est soumise à des agressions telles qu'une variation de température ou un léger choc. Les essais sur les matières explosives demandent une préparation précise et une organisation sans failles. Les compositions sont délicates à réaliser et elles nécessitent des protocoles de fabrication longs et rigoureux. Les compagnes d'évaluation des propriétés explosives et les mesures mettent en œuvre d'importants moyens. Le nombre des expériences doit être limité au strict nécessaire. L'économie d'un seul essai entraine un gain de temps et une économie appréciable. C'est l'une des situations ou les plans D-optimaux sont précieux.

4.3.1 Description de la composition explosive

La composition H6 contient quatre constituants

- L'aluminium.
- RDX, explosif très réactif.
- La cire.
- Un explosif fusible, le TNT (trinitrotoluéne).

Cette composition est satisfaisante comme explosif, mais elle ne passe pas les tests permettant de la classer en MDEPS. Pour diminuer la sensibilité du produit, les expérimentateurs explorent plusieurs voies :

- Un meilleur équilibre entre les proportions des trois premiers constituants, l'aluminium, l'explosif RDX et la cire.
- Le remplacement du RDX, explosif réputé trop sensible, par le couple HMX/ONTA qui l'est beaucoup moins.
- L'utilisation d'un nouvel explosif fusible, le TNMA à la place du TNT.

4.3.2 Choix des facteurs

Les pyrotechniciens retiennent quatre facteurs :

Facteur 1 : la proportion d'aluminium ;

Facteur 2 : le rapport HMX sur ONTA ;

Facteur 3 : la proportion de cire ;

Facteur 4 : le type d'explosif fusible.

4.3.3 Domaine d'étude

Le tableau (4.1) indique les niveaux haut et bas de chaque facteur.

Facteur	Niveau -1	Niveau +1
Aluminium (1)	8%	20%
Rapport HMX sur ONTA (2)	0/100	50/50
Cire (3)	4%	10%
Explosif fusible (4)	TNMA	TNT

Tableau 4.1 : Une étude explosive. Domaine d'étude.

4.3.4 Choix des réponses

Deux réponses sont choisies par les responsables de l'étude :

- La vitesse de détonation en mètres par seconde. Cette grandeur caractérise la force explosive du produit. On vise une vitesse de détonation d'au moins $7400 m/s$.
- Le coefficient de sensibilité qui caractérise la susceptibilité du produit aux sollicitations extérieures. Cette sensibilité doit être faible, inférieure à 95 si possible.

4.3.5 Choix du plan expérimental

Il y a trois facteurs continus et un facteur discontinu. Les expérimentateurs pensent que les réponses suivent une loi du second degré pour les facteurs continus. Il faut donc choisir un plan permettent d'obtenir ce modèle. Un plan de Doehlert leur semble approprié. Treize essais doivent être envisagés pour ces trois facteurs.

Quant au facteur discontinu (type d'explosif fusible), il n'a que deux niveaux.

Le plan complet entraine la réalisation de 26 essais. C'est beaucoup trop. Les expérimentateurs décident donc de choisir un petit nombre d'essais parmi les 26 possibles. Ce choix est réalisé grâce au critère de D-optimalite, qui oblige à choisir, avant les essais, un modèle mathématique.

4.3.6 Modèle postulé

Les expérimentateurs adoptent un modèle du second degré pour les trois facteurs continus. Ils n'envisagent pas d'interaction entre ces facteurs. Le modèle possède donc un terme

constant, des termes du premier degré et des termes du second degré. Il n'y a pas de termes rectangles.

Le facteur discontinu ne prend que deux niveaux. On le modélise avec un terme du premier degré et on ne prévoit pas d'interaction entre ce facteur et les trois autres.

Le modèle à priori est le même pour les deux réponses, vitesse de détonation et sensibilité. Le modèle est le suivant :

$$y = a_0 + a_1x_1 + a_2x_2 + a_3x_3 + a_4x_4 + a_{11}x_1^2 + a_{22}x_2^2 + a_{33}x_3^2 + e$$

Il y a huit coefficients à déterminer. Si l'on veut ménager quelques degrés de liberté pour évaluer le résidu, on réalisera entre 10 et 12 expériences, soit moins de la moitié du plan prévu initialement.

Essai n°	Aluminium (1)	HMX sur ONTA (2)	Cire (3)	Explosif fusible (4)
1	0	0	0	−1
2	+1	0	0	−1
3	+0.5	+0.866	0	−1
4	−0.5	+0.866	0	−1
5	−1	0	0	−1
6	−0.5	−0.866	0	−1
7	+0.5	−0.866	0	−1
8	+0.5	+0.289	+0.816	−1
9	−0.5	+0.289	+0.816	−1
10	0	+0.577	+0.866	−1
11	+0.5	−0.289	−0.816	−1
12	−0.5	−0.289	−0.816	−1
13	0	+0.577	−0.816	−1
14	0	0	0	+1
15	+1	0	0	+1
16	+0.5	+0.866	0	+1
17	−0.5	+0.866	0	+1
18	−1	0	0	+1
19	−0.5	−0.866	0	+1
20	+0.5	−0.866	0	+1
21	+0.5	+0.289	+0.816	+1
22	−0.5	+0.289	+0.816	+1
23	0	−0.577	+0.816	+1
24	+0.5	−0.289	−0.816	+1
25	−0.5	−0.289	−0.816	1
26	0	+0.577	−0.816	1

Tableau 4.2 : Une étude explosive. Points candidats du projet initial.

4.3.7 Calcul du plan D-optimal

Les expérimentateurs possèdent un logiciel de plan d'expériences pourvu de l'algorithme de calcul des plans D-optimaux. Ils commencent par entrer les 26 points candidats indiqués par le tableau (4.2) . Ils indiquent ensuite au logiciel, le modèle qu'ils ont retenu et ils lancent l'algorithme de calcul en imposant un nombre de $n = 12$ expériences. Apres plusieurs passages, ils retiennent les expériences données par la machine et écrivent le tableau (4.3) . C'est le plan D-optimal qui vas être réalisé. Ces douze essais permettant de déterminer les huit coefficients du modèle, on va utiliser les algorithmes précédents et puis on calcule le déterminant de (X^tX) dont les résultats sont dans le tableau (4.3).

ALGORITHMES	det (X^tX)
DONEV	7109.512325
FDOP modifié	8545.111665

Tableau 4.3 : Le déterminant de (X^tX) suivant les algorithmes .

4.4 Comparaison des algorithmes

Les algorithmes qui sont traités dans ce chapitre obligent d'effectuer plusieurs fois la même recherche pour augmenter la probabilité d'obtenir la solution idéale. Nous effectuons donc pour toutes les méthodes 10 fois la même recherche et comparons les meilleures solutions obtenues.

En pratique, il est impossible de considérer l'infinité des niveaux donc dans tous les cas nous accepterons une incertitude relative sur la position d'un point expérimental de l'ordre de 0.005.

Le tableau suivant montre les différents déterminants calculés par les algorithmes ainsi que leurs temps d'excusions.

ALGORITHMES	det (X^tX)	**Temps d'exécution / s**
DONEV	7109.512325	18.8
FDOP modifié	8545.111665	12.3

Tableau 4.4 : Comparaison des algorithmes.

CHAPITRE V

LES NOUVELLES PROPOSITIONS

Ce chapitre traite un nouveau algorithme qui est une modification de l'algorithme de Donev avec une comparaison entre les deux algorithmes et terminant le travail par une proposition des étapes pour construire l'algorithme de colonie de fourmis dans le cas des plans D-optimaux.

5.1 L'algorithme de DONEV modifié

5.1.1 Principe de l'algorithme

Donev modifié propose un moyen d'améliorer l'algorithme de Donev à partir d'une matrice presque optimale où il fait des déplacements. Cet algorithme déplace le 1^{er} point de la matrice après avoir choisi le meilleur point du plan tout en utilisant la fonction de Fedorov Δ et on répète cette procédure plusieurs fois et à la fin on fait l'échange. Puis on répéte la méme procédure pour le point suivant, à la fin on obtient une matrice d'expériences dont le déterminant est meilleur que le précèdent d'où l'optimalité.

5.1.2 Algorithme proposé

Choix d'un amplitude de déplacement α

Calculer la matrice (X^tX)

Pour $i = 1$ à n **faire**

$t = i$

$\Delta_{\max} = 0$

Initialiser α

Tant que $(\alpha \geq \alpha_{\min})$ **alors**

Essayer tous les changements sur les axes

Pour $j = 1$ à nombre de variables indépendantes **faire**

- **Donev** : améliore de façon significative les solutions obtenues par l'algorithme d'échange. Cependant la convergence vers la solution finale peut s'avérer parfois trés longue.
- **Fdop modifié** : utilisant des déplacements aléatoires proposer pour un faible cout en temps de calcul des solutions très proche de la solution optimale. N'est utilisable que par ce qu'il est possible de hiérarchiser les points de la matrice (le déterminant de la nouvelle matrice peut être calculé rapidement) il n'existe pas de telles possibilités pour les autres critères.

Essayer *sens* négatif puis positif

Pour $sens = -1$ puis $+1$ **faire**

$D = (d_1, d_2, \ldots, d_n)$ avec $d_j = sens$ et $d_m = 0$ (pour $m \neq j$)

Calculer le nouveau point $x_q = x_t + \alpha.D$

Calculer la fonction delta de Fedorov : $\Delta(i, q)$

Si $(\Delta(i, q) > \Delta_{\max})$ **alors**

x_q est le changement sélectionné

$\Delta_{\max} = \Delta(i, q)$

Si $(\Delta_{\max} > 1)$ **alors**

$\alpha = \alpha$

Sinon

$\alpha = \alpha/2$

Si $(\Delta(i, s) = \Delta_{\max})$ **alors**

$t = s$

Changement ou convergence

Si $(\Delta_{\max} > 1)$ **alors**

Effectuer le meilleur changement sélectionné

Calculer la nouvelle matrice $(X^t X)^{-1}$

Algorithme 5.1 : Algorithme de DONEV modifié.

5.1.3 Etude comparative entre l'algorithme de Donev et Donev modifié

Les expérimentations ont été réalisées sur un PC équipé d'un processeur Pentium 3,1333 MHz avec une mémoire de 4 GB de RAM sous Windows 7. Les algorithmes sont implémentés en **Maple**. Les programmes prennent comme données d'entrée des matrices d'expériences contenant des points et doivent être changées à la fin de l'algorithme pour arriver à l'optimalité.

5.1.3.1 Instances numériques utilisées.

Un ensemble d'instances aléatoires a été élaboré. Cinq instances de problèmes-tests ont été générées de manière aléatoire.

Le nombre de facteurs considéré dans ces instances est $2, 3, 4, 5, 6$.

5.1.3.2 Résultats de la résolution

Une étude statistique des résultats générés est effectuée pour déterminer la performance des deux méthodes de résolution. À l'instant initial, on démarre avec une matrice presque optimale ensuite on fait le déplacement par une amplitude précise (0.05).

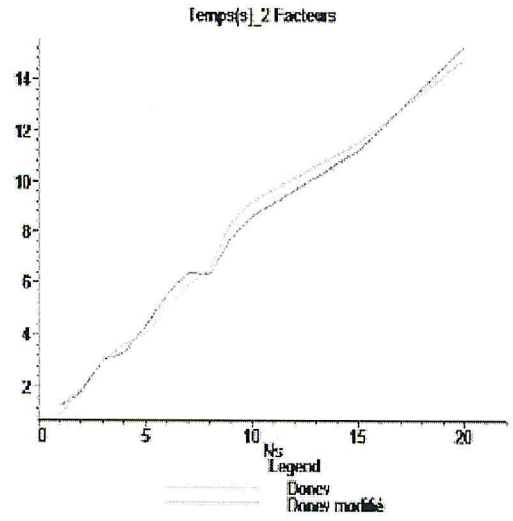
5.1.3.3 Critères de validité du modèle

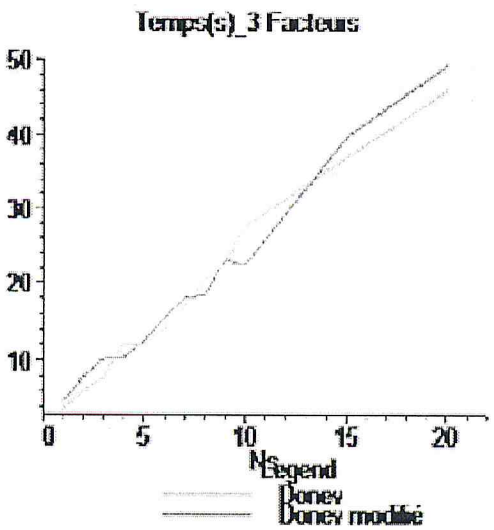
Deux critères de comparaisons sont utilisés :

- **Le temps d'exécution**

Il est représenté dans un repère orthonormé dont l'abscisse est le nombre d'itérations et l'ordonnée est le temps d'exécution. À chaque fois qu'on augmente le nombre d'itérations le temps d'exécution augmente et il n y a pas une grande différence de temps d'exécution entre les deux algorithmes.

Les résultats de nos différents essais, et les comportements des deux méthodes de résolution sont représentés dans les graphes ci-dessous.





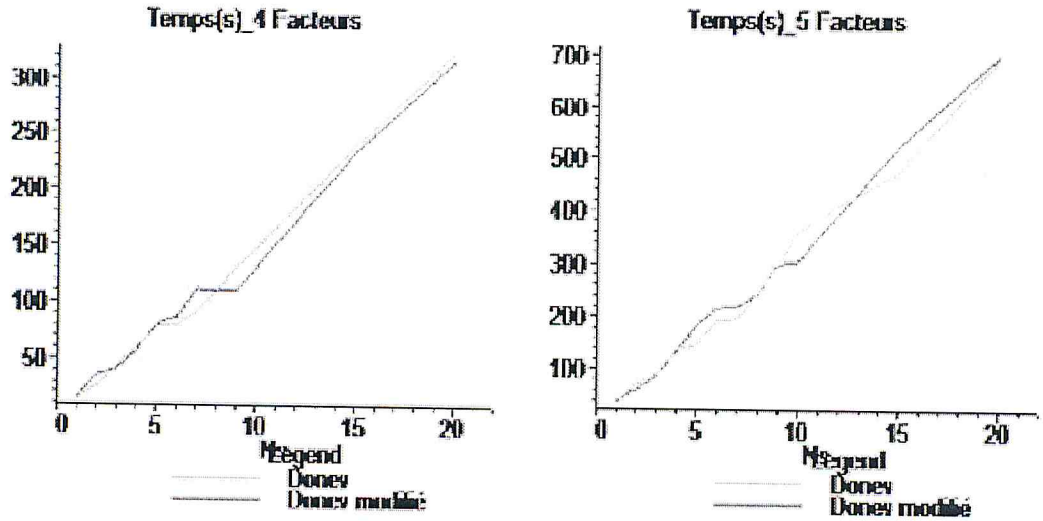
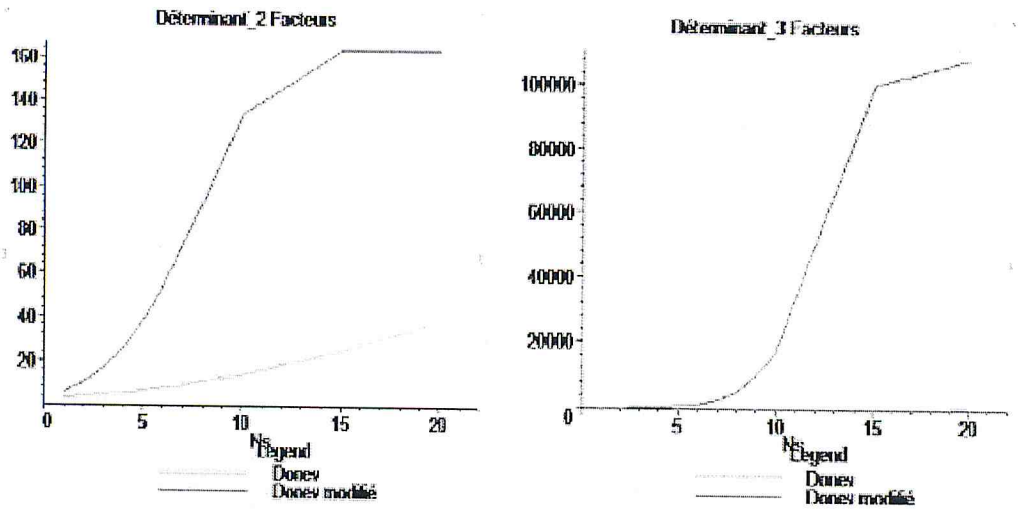


Figure 5.1 : Courbes comparatives par rapport au temps d'exécution.

- **La valeur du déterminant**

Elle est représentée dans un repère orthonormé dont l'abscisse est le nombre d'itérations et l'ordonnée est la valeur du déterminant. On remarque que la valeur du déterminant de Donev modifié est plus meilleure que celle de Donev et on voit la différence dès la première itération.

Ces résultats sont regroupés dans le graphique ci-dessous :



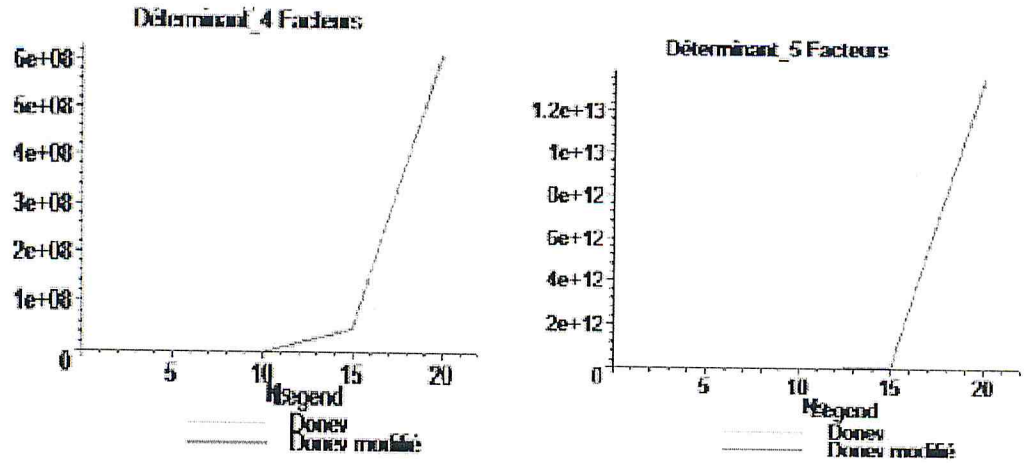


Figure 5.2 : Courbes comparatives par rapport aux déterminants.

5.2 Proposition d'un algorithme de Colonie de Fourmis (OCF)

5.2.1 Principe de l'algorithme

Cet algorithme est une méta heuristique dont le comportement est inspiré de celui des fourmières réelles, il s'inspire d'une expérience mené en 1989 par Goss et Al, des fourmis avaient été mis en contact avec une source de nourriture reliée à la fourmière par un pont ayant deux branches de longueur différentes après une durée de temps elle trouve le plus court chemin.

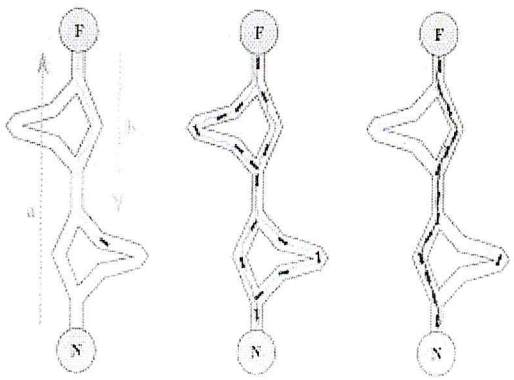


Figure 5.3 : Représentation de système Fourmi.

Cette dernière partage des informations à l'aide d'une substance chimique appelé phéromone. Elles disposent par terre une certaine quantité lorsqu'elles se déplacent et puis les autres

suivent le chemin où il y a plus de phéromone. Comme la phéromone s'évapore avec le temps alors sur le long chemin sa trace disparait.

Des fourmis qui vont se déplacer à la recherche de solutions et qui vont sécréter des phéromones pour indiquer à leurs congénères si un chemin est intéressant ou non. Si un chemin se retrouve fortement phéromoné, cela signifiera que beaucoup de fourmis l'ont jugé comme faisant partie d'une solution intéressante et que les fourmis suivantes devront la considérer avec intérêt.

Un risque apparaît lorsqu'un chemin non optimal est marqué. En effet, les fourmis qui s'en trouveront à proximité seront tentées d'y passer augmentant encore le niveau de phéromone de ce chemin, pour diminuer le risque d'enfoncer la colonie dans un minimum local du problème, on pourra prendre soin de diminuer automatiquement le niveau de phéromone de tout le système, pour réhausser l'intérêt des autres chemins qui pourraient faire partie de la solution optimale. Ce paramètre, correspondant au taux d'évaporation des phéromones, est l'un des paramètres principaux de l'algorithme.

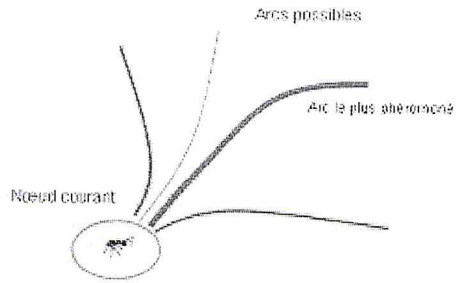


Figure 5.4 : Graphe qui montre la quantité du phéromone posé par les fourmis sur chaque chemin.

De la même manière, aucun chemin ne devra être inondé de phéromones et aucun chemin ne devra être totalement invisible, on pourra donc aussi contrôler le niveau de phéromone de chaque chemin pour le maintenir entre des bornes minimum et maximum. Un chemin inondé de phéromones masquerait tous les autres à proximité et un chemin où il n y a pas de phéromones ne serait jamais choisi par une fourmi, en conséquence nous devons conserver ces chemins avec des valeurs raisonnables. Ces bornes min et max sont aussi des paramètres de l'algorithme [16]...[22].

5.2.2 L'utilisation de l'algorithme de colonies de fourmis dans les plans D-optimaux

A partir de la matrice des algorithmes d'échanges, on essaye d'appliquer l'algorithme de colonie de fourmis avec un déplacement et un échange au même temps. On positionne sur chaque sommet une fourmi qui va se déplacer et choisir son chemin aléatoirement au début et elle posent une quantité du phéromone et avec le temps cette quantité change car une fois que la fourmi passe elle laisse la phéromone d'où le changement de quantité et puis les autres qui arrivent choisissent le chemin qui contient beaucoup de phéromone et c'est celui qui représente le plus court chemin d'où la solution presque optimale. En revanche les autres sommets non choisis on les met dans une liste tabou afin de les comparer avec les autres pour ne pas tomber dans le minimum local.

5.2.2.1 Les étapes de l'algorithme

1^{ere} étape : Positionner une fourmi sur chaque sommet.

2^{ieme} étape : Initialiser la phéromone $\tau_0 = \Delta$.

3^{ieme} étape : Prendre la matrice d'expérience obtenue par les algorithmes d'échanges et faire un déplacement pour le 1^{er} point dont la fourmi part au point qui a Δ_{max}.

4^{ieme} étape : Construire une liste taboue L qui contient les sommets non choisis et de les comparer avec les autres points choisis à chaque itération pour éviter le minimum local.

5^{ieme} étape : Passer au point qui a Δ_{max} et refaire N_s déplacements mais suivant une probabilité p_{ij} qui dépend de la phéromone Δ , la visibilité α et β qui sont des paramètres limités et la variance de prédiction d_i, d_j .

6^{ieme} étape : Si le test d'arrêt est atteint alors faire le changement du mauvais point par le bon sinon retourner à l'étape précédente.

7^{ieme} étape : Passer au point suivant de la matrice et réappliquer les étapes précédentes.

CONCLUSION GENERALE

Les plans D-optimaux sont des plans très utilisés par les expérimentateurs car ces derniers leurs permettent d'optimiser leurs expériences. Pour trouver la matrice optimale dans ces plans, on a utilisé dans ce travail les algorithmes d'échanges : DETMAX, FEDOROV, FEDOROV MODIFIE, K-ECHANGE, KL-ECHANGE, KL-ECHANGE MODIFIE, FDOP dont leur principe est de faire changer les mauvais points par les bons en utilisant les variances de prédiction. Aussi on a utilisé les algorithmes de déplacements : FDOP MODIFIE, DONEV qui ont pour principe de déplacer les points du plan suivant avec un pas précis afin d'arriver au bon point pour faire l'échange.

Nous avons programmé tous ces algorithmes avec le logiciel **MAPLE** afin de nous aider à faire une comparaison entre eux par rapport au déterminant et au temps d'exécution. On remarque que le meilleur algorithme est celui de Fedorov par rapport au déterminant, et par rapport au temps on préfère Detmax, mais pour les algorithmes de déplacements on utilise les matrices presque optimales obtenues par les algorithmes d'échanges et on constate que Fdop est le meilleur.

En revanche, on a fait une modification sur l'algorithme de Donev et cela nous a donné de très bons résultats ainsi nous proposons une amélioration de l'algorithme de colonie de fourmis dans le cas des plans D-optimaux.

Comme tout travail scientifique, des perspectives sont proposées afin d'améliorer l'étude faite dans ce mémoire pour l'algorithme de colonie de fourmis, à trouver la probabilité du choix du déplacement.

Les Programmes Sur MAPLE

Ces algorithmes sont codés et exécutés sur un PC équipé d'un processeur Pentium 3,1333 MHz avec une mémoire de 4 GB de RAM sous Windows 7. Les algorithmes sont implémentés en **Maple**.

I. Les programmes de l'application du chapitre 3

```
> restart:
> with(LinearAlgebra):
> with(linalg):
> for i from 1 by 1 to 23 do  x[i][4]:=-1;  end do:
> for i from 24 by 1 to 46 do x[i][4]:=0;  end do:
> for i from 47 by 1 to 69 do x[i][4]:=1;   end do:
> for i from 1 by 1 to 5 do  x[i][3]:=-1;  x[i+23][3]:=-1; x[i+46][3]:=-1; end do:
> for i from 6 by 1 to 10 do x[i][3]:=-0.5; x[i+23][3]:=-0.5; x[i+46][3]:=-0.5; end do:
> for i from 11 by 1 to 15 do x[i][3]:=0; x[i+23][3]:=0; x[i+46][3]:=0; end do:
> for i from 16 by 1 to 20 do x[i][3]:=0.5; x[i+23][3]:=0.5; x[i+46][3]:=0.5; end do:
> for i from 21 by 1 to 23 do x[i][3]:=1; x[i+23][3]:=1; x[i+46][3]:=1; end do:
> for i in {1,6,11,16,21,24,29,34,39,44,47,52,57,62,67} do x[i][2]:=-1; x[i+1][2]:=-0.5;
    x[i+2][2]:=0; x[i+3][2]:=0.5; x[i+4][2]:=-1; end do:
> for i from 1 by 1 to 69 do
> x[i]:= <1 |x[i][2]| x[i][3]|x[i][4] | (x[i][2])*(x[i][3])| (x[i][2]) *(x[i][4])|
(x[i][3])*(x[i][4])|(x[i][2])^2|(x[i][3])^2|(x[i][4])^2>;
> end do:
> n:=69; N:=12; p:=10;
> S:= {1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,
28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51,52,53,
54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69};
> X:= <seq(x[i],i=S)>:
> SB:= {12,13,16,17,18,19,24,58,60,61,65,69};
```

```
> SH := {1,2,3,4,5,6,7,8,9,10,11,14,15,20,21,22,23,25,26,27,28,29,30,31,32,33,34,
35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51,52,53,54,55,56,57,59,62,63,64,
66,67,68};
> X:= <seq(x[i],i=SB)>:
> det:= det(multiply(transpose(X),X));
> w:=det:
> pas:=1:
> delta :=10:
```

L'algorithme de Detmax

```
> for t from 1 by 1 while (pas > 0) do
> v:=-100000:
> for i in SH do
> di:=multiply(x[i], inverse(multiply(transpose(X),X)) ,x[i] ):
> if (di>v) then bon:=i: v:=di: end if: end do:
> SB := {op(SB),bon};  SH := SH minus {bon};
> X:= <seq(x[i],i=SB)>:
> v:=100000:
> for j in SB do
> di:=multiply(x[j], inverse(multiply(transpose(X),X)) ,x[j] ):
> if (di<v) then mauv:=j: v:=di: end if: end do:
> SB := SB minus {mauv}; SH := {op(SH),mauv};
> X:= <seq(x[i],i=SB)> ;
> with(linalg):
> det:= det(multiply(transpose(X),X));
> pas:=det-w;
> if (pas>0) then w:=det: end if
```

L'algorithme de Fedorov

```
> for t from 1 by 1 while ( delta > 0 ) do
> for j in SH do
> dj:=multiply(x[j], inverse(multiply(transpose(X),X)) ,x[j] ):
> for i in SB do
> di:=multiply(x[i], inverse(multiply(transpose(X),X)) ,x[i] ):
> dij:=multiply(x[i], inverse(multiply(transpose(X),X)) ,x[j] ):
> deltaij:=evalf(dj-((dj)*(di)-(dij)^2)-di):
> if (deltaij>delta) then bon:=j: mauv:=i: delta:=deltaij:
> end if: end do: end do:
> if ( delta>0) then
> SB := {op(SB minus {mauv}),bon};
    SH := {op(SH),mauv} minus {bon}; end if:
> X:= <seq(x[i],i=SB)>:
> with(linalg): det:= det(multiply(transpose(X),X)); end do:
```

L'algorithme de Fedorov modifié

```
> for t from 1 by 1 while (pas > 0) do
> for i in SB do
> di:=multiply(x[i], inverse(multiply(transpose(X),X)) ,x[i] ):
> delta:=-1000000:
> for j in SH do
> dj:=multiply(x[j], inverse(multiply(transpose(X),X)) ,x[j] ):
> dij:=multiply(x[i], inverse(multiply(transpose(X),X)) ,x[j] ):
> deltaij:=evalf(dj-((dj)*(di)-(dij)^2)-di):
> if (deltaij>delta) then bon:=j: mauv:=i: delta:=deltaij: end if: end do:
> if (delta >0 ) then
> SB := {op(SB minus {mauv}),bon}:
    SH := {op(SH),mauv} minus {bon}:
```

```
> X:= <seq(x[i],i=SB)>: end if:
> with(linalg): det:= det(multiply(transpose(X),X)); end do:
> pas:=det-w: if (pas >0 ) then w:=det: end if: end do:
```

L'algorithme de K-échange

```
> for t from 1 by 1 while (pas>0) do
> for i in SB do
> d[i]:=multiply(x[i], inverse(multiply(transpose(X),X)) ,x[i] ): end do:
> a:={}: v:=0: sk:=SB:
> while v<k do
> z:=sort([seq(d[i],i=sk)]);
> for i in sk do if ((d[i]=z[1])) then r1:=i: end if: end do;
> a:={op(a),r1};
> sk:=sk minus a;
> v:=nops(a); end do;
> SV:=a;
> for i in SV do
> di:=multiply(x[i], inverse(multiply(transpose(X),X)) ,x[i] ):
> delta:=-10000000000000:
> for j in SH do
> dj:=multiply(x[j], inverse(multiply(transpose(X),X)) ,x[j] ):
> dij:=multiply(x[i], inverse(multiply(transpose(X),X)) ,x[j] ):
> deltaij:=evalf(dj-((dj)*(di)-(dij)^2)-di):
> if (deltaij>delta) then bon:=j: mauv:=i: delta:=deltaij: end if: end do:
> if (delta >0 ) then SB := {op(SB minus {mauv}),bon}:
    SH := {op(SH),mauv} minus {bon}:
> X:= <seq(x[i],i=SB)>: end if:
> with(linalg): det:= det(multiply(transpose(X),X));
> end do; pas:=det-w: if (pas >0 ) then w:=det: end if: end do:
```

L'algorithme de KL-échange

```
> for t from 1 by 1 while (pas > 0) do
> for i in S do d[i]:=multiply(x[i], inverse(multiply(transpose(X),X)) ,x[i] ): end do:
> a:={}: v:=0: sk:=SB:
> while v<k do
> z:=sort([seq(d[i],i=sk)]):
> for i in sk do
> if ((d[i]=z[1])) then r1:=i: end if: end do:
> a:={op(a),r1}: sk:=sk minus a: v:=nops(a): end do: SK:=a; a:={}: v:=0: sl:=SH:
> while v<l do
> z:=sort([seq(d[i],i=sl)]):
> for i in sl do
> if ((d[i]=z[nops(z)])) then r1:=i: end if: end do:
> a:={op(a),r1}: sl:=sl minus a: v:=nops(a): end do: SL:=a;
> delta:=-1000000:
> for j in SL do
> dj:=multiply(x[j], inverse(multiply(transpose(X),X)) ,x[j] ):
> for i in SK do
> di:=multiply(x[i], inverse(multiply(transpose(X),X)) ,x[i] ):
> dij:=multiply(x[i], inverse(multiply(transpose(X),X)) ,x[j] ):
> deltaij:=evalf(dj-((dj)*(di)-(dij)^2)-di):
> if (deltaij>delta) then bon:=j: mauv:=i: delta:=deltaij: end if: end do: end do:
> if ( delta>0) then SB := {op(SB minus {mauv}),bon};
    SH := {op(SH),mauv} minus {bon}; end if:
> X:= <seq(x[i],i=SB)>:
> with(linalg): det:= det(multiply(transpose(X),X)); end do:
```

L'algorithme de FDOP

```
> det:= det(multiply(transpose(X),X)); w:=det: q:=det:
> m:= 0: maxi:= 0:
> for t from 1 by 1 while (pas > 0) do
> r[1]:=SH[1],SH[2],SH[3],SH[4],SH[5],SH[6],SH[7],SH[8],SH[9],SH[10],SH[11],SH[12],SH[13],
    SH[14],SH[15],SH[16],SH[17],SH[18],SH[19]:
> r[2]:=SH[20],SH[21],SH[22],SH[23],SH[24],SH[25],SH[26],SH[27],SH[28],SH[29],SH[30],
    SH[31],SH[32],SH[33],SH[34],SH[35],SH[36],SH[37],SH[38]:
> r[3]:=SH[39],SH[40],SH[41],SH[42],SH[43],SH[44],SH[45],SH[46],SH[47],SH[48],SH[49],
    SH[50],SH[51],SH[52],SH[53],SH[54],SH[55],SH[56],SH[57]:
> for j from 1 by 1 to R do
> sk := SB union {r[j]}: sl := SH minus {r[j]}:
> X:= <seq(x[i],i=sk)>:
> v:=20:
> while (v>N)do
> v:=100000:
> for i in sk do
> di:=multiply(x[i], inverse(multiply(transpose(X),X)) ,x[i] ):
> if (di<v) then mauv:=i: v:=di: end if: end do:
> sk := sk minus {mauv}: sl := {op(sl),mauv}:
> X:= <seq(x[i],i=sk)>: v:=nops(sk): end do:
> with(linalg): det:= det(multiply(transpose(X),X)):
> pas:=det-w:
> if (pas>0) then SB:=sk: SH:=sl:
> X:= <seq(x[i],i=SB)>:
> w:=det: end if: end do:
> X:= <seq(x[i],i=SB)>:
> with(linalg): det:= det(multiply(transpose(X),X));
> pas:=det-q; if ( pas > 0) then m:=m+1: q:=det: end if: end do:
```

II. Les programmes de l'application du chapitre 4

```
> restart:
> with(LinearAlgebra):
> with(linalg):
> y[1]:= <1 |0 |0 |0 |-1 |0 |0 |0 >;
> y[2]:= <1 |1 |0 |0 |-1 |1 |0 |0 >;
> y[3]:= <1 |0.5 |0.866 |0 |-1 |(0.5)^2|(0.866)^2|(0)^2>;
> y[4]:= <1 |-0.5 |0.866 |0 |-1 |(-0.5)^2|(0.866)^2|(0)^2>;
> y[5]:= <1 |-1 |0 |0 |-1 |(-1)^2|(0)^2|(0)^2>;
> y[6]:= <1 |-0.5 |-0.866 |0 |-1 |(0.5)^2|(0.866)^2|(0)^2>;
> y[7]:= <1 |0.5 |-0.866 |0 |-1 |(0.5)^2|(0.866)^2|(0)^2>;
> y[8]:= <1 |0.5 |0.289 |0.816 |-1 |(0.5)^2|(0.289)^2|(0.816)^2>;
> y[9]:= <1 |-0.5 |0.289 |0.816 |-1 |(0.5)^2|(0.289)^2|(0.816)^2>;
> y[10]:=<1 |0 |-0.577 |0.816 |-1 |(0)^2|(0.5777 )^2|(0.816)^2>;
> y[11]:=<1 |0.5 |-0.289|-0.816|-1 |(0.5)^2|(0.289)^2|(0.816)^2>;
> y[12]:=<1 |-0.5 |-0.289|-0.816|-1 |(0.5)^2 |(0.289)^2|(0.816)^2>;
> y[13]:=<1 |0 |0.5777 |-0.816|-1 |(0)^2|(0.5777 )^2|(0.816)^2>;
> y[14]:=<1 |0 |0 |0 |1 |0 |0 |0 >;
> y[15]:=<1 |1 |0 |0 |1 |1 |0 |0 >;
> y[16]:=<1 |0.5 |0.866 |0 |1 |(0.5)^2|(0.866)^2|(0)^2>;
> y[17]:=<1 |-0.5 |0.866 |0 |1 |(-0.5)^2|(0.866)^2|(0)^2>;
> y[18]:=<1 |-1 |0 |0 |1 |(-1)^2|(0)^2|(0)^2>;
> y[19]:=<1 |-0.5 |-0.866 |0 |1 |(0.5)^2|(0.866)^2|(0)^2>;
> y[20]:=<1 |0.5 |-0.866 |0 |1 |(0.5)^2|(0.866)^2|(0)^2>;
> y[21]:=<1 |0.5 |0.289 |0.816 |1 |(0.5)^2|(0.289)^2|(0.816)^2>;
> y[22]:=<1 |-0.5 |0.289 |0.816 |1 |(0.5)^2|(0.289)^2|(0.816)^2>;
> y[23]:=<1 |0 |-0.577 |0.816 |1 |(0)^2|(0.5777 )^2|(0.816)^2>;
> y[24]:=<1 |0.5 |-0.289|-0.816|1 |(0.5)^2|(0.289)^2|(0.816)^2>;
```

```
> y[25]:=<1 |-0.5 |-0.289|-0.816|1 |(0.5)^2 |(0.289)^2|(0.816)^2>;
> y[26]:=<1 |0 |0.5777 |-0.816|1 |(0)^2|(0.5777 )^2|(0.816)^2>;
> for i from 1 to 13 do
> x[i]:=<y[i][2] |y[i][3] |y[i][4]|-1 >;
> end do:
> for i from 14 to 26 do
> x[i]:=<y[i][2] |y[i][3] |y[i][4]|1 >;
> end do:
> k:=3;
> n:=26;
> N:=12;
> p:=8;
> S := {1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26};
> SP:= {1,2,3,4};
> SB := {1, 3, 5, 7, 9, 11, 14, 15, 17, 19, 23, 26};
> Y:= <seq(y[i],i=SB)>:
> det:= 6461.860318;
```

L'algorithme de Donev

```
> alphamin:=0.001; Ns:= 10; alpha:=0.005;
> for itr from 1 by 1 while ((alpha >= alphamin)and(itr<=Ns)) do
> delta_max:=0:
> for i in SB do
> t:=i:
> dt:=multiply(y[t], inverse(multiply(transpose(Y),Y)) ,y[t] ):
> for j from 1 by 1 to k do
> for p in {-1,1} do
> n:=n+1: a:=n: e[j]:=p :
> for l in SP minus {j} do e[l]:=0: end do:
```

```
> E:=Vector[row]([seq(e[j],j=SP)]):
> x[a]:=x[t]+alpha*E:
> y[a]:= <1 |x[a][1]| x[a][2]|x[a][3] | x[a][4]|(x[a][1])^2|(x[a][2])^2|(x[a][3])^2>:
> da:=multiply(y[a], inverse(multiply(transpose(Y),Y)) ,y[a] ):
> dta:=multiply(y[t], inverse(multiply(transpose(Y),Y)) ,y[a] ):
> delta:=evalf(1+da-((da)*(dt)-(dta)^2)-dt):
> s:=0:
> for i from 1 by 1 to k do
> if ((x[a][i]<=1)and(x[a][i]>=-1)) then s:= s+1: end if : end do :
> if (( delta > delta_max )and (s=k)) then bon := a: mauv := t:
    delta_max:=delta: end if :
> end do: end do: end do:
> if ( delta_max > 1) then SB := {op(SB minus {mauv}),bon}:
> else alpha:=alpha/2: end if:
> Y:= <seq(y[i],i=SB)>:
> with(linalg): det:= det(multiply(transpose(Y),Y)): end do:
```

L'algorithme de Fdop modifie

```
> alphamin:=0.001; alpha:=0.005; comp[0]:= 6461.860318; w[0]:= comp[0]:;
> A:={}: Ns:= 10: m:= 0: g:= 10; pas:= 10; maxi:= 0;
> for itr from 1 by 1 while ((maxi < 4)and (itr<=Ns) ) do
> r[1]:=SB[1],SB[2],SB[3],SB[4]: r[2]:=SB[5],SB[6],SB[7],SB[8]:
    r[3]:=SB[9],SB[10],SB[11],SB[12]:
> for i in SB do x[i]:=<y[i][2] |y[i][3] |y[i][4]|y[i][5] >; end do:
> for hj from 1 by 1 to R do a:=hj; sv:= {};
> for hi from 1 by 1 to M do b:=hi; t:=r[a][b]; sg:={};
> for o from 1 by 1 to k do e[o]:=-1 :
> for l in SP minus {o} do e[l]:=0: end do:
> E:=Vector[row]([seq(e[j],j=SP)]):
```

```
> z[n+1]:=x[t]+alpha*E: u[o]:=1 :
> for l in SP minus {o} do u[l]:=0: end do:
> F:=Vector[row]([seq(u[j],j=SP)]): z[n+2]:=x[t]+alpha*F: G:={n+1,n+2}:
> for l in G do
> for u from 1 by 1 to k do
> if ((z[l][u]<-1)or(z[l][u]>1))then e:=l; G:= G minus {e}; end if ; end do; end do;
> sg:= sg union G ; n:=n+2: end do;
> Y:= <seq(y[i],i=SB)>;
> for i in sg do
> y[i]:= <1 |z[i][1]| z[i][2]|z[i][3] | z[i][4]|(z[i][1])^2|(z[i][2])^2|(z[i][3])^2>:
> d[i]:=multiply(y[i], inverse(multiply(transpose(Y),Y)) ,y[i] ): end do;
> L:=sort([seq(d[c],c=sg)]):
> for l in sg do if (d[l]=L[nops(sg)]) then q:=l: end if: end do:
> sv:= {op(sv),q}; end do;
> sk:= SB union sv;
> Y:= <seq(y[i],i=sk)>;
> v:=20;
> while (v>N)do
> for i in sk do
> d[i]:=multiply(y[i], inverse(multiply(transpose(Y),Y)) ,y[i] ): end do:
> K:=sort([seq(d[i],i=sk)]):
> for i in sk do if (d[i]=K[1]) then p:=i end if; end do:
> mauv:=p; sk := sk minus {mauv}; Y:= <seq(y[i],i=sk)>; v:=nops(sk); end do;
> Y:= <seq(y[i],i=sk)>;
> with(linalg): det[a]:= det(multiply(transpose(Y),Y));
> comp[a]:=det[a]; g:=comp[a]-comp[a-1];
> if ( g > 0 ) then SB:=sk; else comp[a]:=comp[a-1]: end if;
> Y:= <seq(y[i],i=SB)>: end do;
> det[itr]:=comp[a]; w[itr]:= det[itr]; pas:=w[itr]-w[itr-1];
```

```
> if ( pas > 0) then m:=m+1; comp[0]:=w[itr];
> else m:=m; comp[0]:=w[itr-1]; end if; maxi:=m; end do;
```

III. Les programmes du chapitre 5

```
> x[1]:= <-0.2057 |-0.9786 >;
> x[2]:= <-0.2394 |0.9709 >;
> x[3]:= <0.6100 |0.7924 >;
> x[4]:= <0.6369 |-0.7709 >;
> x[5]:= <-0.9085 |0.4179 >;
> x[6]:= <0.9999 |0.0172 >;
> x[7]:= <-0.8932 |-0.4496 >;
> x[8]:= < 0.0755|0.4943 >;
> x[9]:= <0.3908 | -0.3127>;
> x[10]:=< -0.4657|-0.1822 >;
> for i from 1 to 10 do
> y[i]:=<1 |x[i][1]| x[i][2]|(x[i][1])*(x[i][2])|(x[i][1])^2|(x[i][2])^2>;
> end do:
> k:=2; n:=10; N:=6; p:=6;
> S := {1,2,3,4,5,6,7,8,9,10};
> SP:= {1,2}; SB :={1, 2, 3, 4, 5, 8}; Y:= <seq(y[i],i=SB)>:
```

L'algorithme de Donev

```
> alphamin:=0.001; Ns:=10 ; alpha:=0.05;
> for itr from 1 by 1 while ((alpha >= alphamin)and(itr<=Ns)) do
> delta_max:=0:
> for i in SB do
> t:=i:
> dt:=multiply(y[t], inverse(multiply(transpose(Y),Y)) ,y[t] ):
> for j from 1 by 1 to k do
```

```
> for p in {-1,1} do
> n:=n+1: a:=n:
> e[j]:=p : for l in SP minus {j} do e[l]:=0: end do:
> E:=Vector[row]([seq(e[j],j=SP)]):
> x[a]:=x[t]+alpha*E:
> y[a]:=<1 |x[a][1]| x[a][2]|x[a][1]*x[a][2]|(x[a][1])^2|(x[a][2])^2>:
> da:=multiply(y[a], inverse(multiply(transpose(Y),Y)) ,y[a] ):
> dta:=multiply(y[t], inverse(multiply(transpose(Y),Y)) ,y[a] ):
> delta:=evalf(1+da-((da)*(dt)-(dta)^2)-dt):
> s:=0:
> for i from 1 by 1 to k do
> if ((x[a][i]<=1)and(x[a][i]>=-1)) then s:= s+1: end if : end do :
> if (( delta > delta_max )and (s=k)) then
> bon := a: mauv := t: delta_max:=delta:
> end if : end do: end do: end do:
> if ( delta_max > 1) then SB := {op(SB minus {mauv}),bon}:
> else alpha:=alpha/2: end if:
> Y:= <seq(y[i],i=SB)>:
> with(linalg): det:= det(multiply(transpose(Y),Y)): end do:
```

L'algorithme de Donev modifie

```
> alphamin:=0.001; Ns:=10;
> for i in SB do
> di:=multiply(y[i], inverse(multiply(transpose(Y),Y)) ,y[i] ):
> alpha:=0.05; delta_max:=0: t:=i: mauv:=i:
> for s from 1 by 1 while ((alpha >= alphamin ) and ( s<=Ns )) do
> for j from 1 by 1 to k do
> for p in {-1,1} do n:=n+1: a:=n:
> e[j]:=p :
```

```
> for l in SP minus {j} do e[l]:=0: end do:
> E:=Vector[row]([seq(e[j],j=SP)]):
> x[a]:=x[t]+alpha*E:
> y[a]:=<1 |x[a][1]| x[a][2]|x[a][1]*x[a][2]|(x[a][1])^2|(x[a][2])^2>:
> da:=multiply(y[a], inverse(multiply(transpose(Y),Y)) ,y[a] ):
> dia:=multiply(y[i], inverse(multiply(transpose(Y),Y)) ,y[a] ):
> delta:=evalf(1+da-((da)*(di)-(dia)^2)-di):
> if (((x[a][1]<=1)and(x[a][1]>=-1)and(x[a][2]<=1)and(x[a][2]>=-1))and
( delta > delta_max )) then
> q := a: delta_max:=delta: end if : end do; end do;
> if ( delta_max <= 1) then alpha:=alpha/2: end if:
> t:=q; end do; bon:=t;
> if ( delta_max > 1) then SB := {op(SB minus {mauv}),bon}: end if:
> Y:= <seq(y[i],i=SB)>:
> with(linalg): det:= det(multiply(transpose(Y),Y)):   > end do:
```

Les tableaux de comparaison

- **Cas d'un plan avec 2 facteurs : k=2, N=6, n=10, p=6.**

Nombre d'itération	1		2		3	
Algorithmes	Déterminant	Temps	Déterminant	Temps	Déterminant	Temps
DONEV	3.482820189	0.8	4.187175132	1.3	4.988455199	1.7
DONEV modifié	6.066008344	0.9	10.33192424	1.2	16.63730826	1.8
Nombre d'itération	4		5		6	
Algorithmes	Déterminant	Temps	Déterminant	Temps	Déterminant	Temps
DONEV	5.894747316	2.4	6.932376497	2.8	8.116944499	2.9
DONEV modifié	25.54164858	2.2	37.62574501	2.4	53.45445622	2.8
Nombre d'itération	7		8		9	
Algorithmes	Déterminant	Temps	Déterminant	Temps	Déterminant	Temps
DONEV	9.441220525	3.8	10.91533139	4	12.45144428	4.2
DONEV modifié	73.53161455	3.1	92.17800624	3.8	112.4332584	4.1
Nombre d'itération	10		15		20	
Algorithmes	Déterminant	Temps	Déterminant	Temps	Déterminant	Temps
DONEV	14.11121245	5	24.68002206	5.9	38.27936646	7.8
DONEV modifié	133.5911373	4.2	163.4263057	5.5	163.5130460	7.9

- **Cas d'un plan avec 4 facteurs : $k = 4, N = 15, n = 35, p = 15$.**

Nombre d'itération	1		2		3	
Algorithmes	Déterminant	Temps	Déterminant	Temps	Déterminant	Temps
DONEV	0.8533421344	14.2	1.013034062	25.5	1.194151874	42.8
DONEV modifié	4.978510711	14.4	30.13499069	35.1	155.8798669	40.7
Nombre d'itération	4		5		6	
Algorithmes	Déterminant	Temps	Déterminant	Temps	Déterminant	Temps
DONEV	1.405946817	59.8	1.642281313	78.8	1.911341309	79.6
DONEV modifié	635.9433814	57.6	2273.176254	80.9	7474.439176	86.7
Nombre d'itération	7		8		9	
Algorithmes	Déterminant	Temps	Déterminant	Temps	Déterminant	Temps
DONEV	2.215955842	91.8	2.563652406	110.1	2.959335058	131.1
DONEV modifié	22258.10362	111.5	67023.73136	110.4	185324.9452	110.6
Nombre d'itération	10		15		20	
Algorithmes	Déterminant	Temps	Déterminant	Temps	Déterminant	Temps
DONEV	3.418182352	147.7	6.901753488	239	13.58352351	323.2
DONEV modifié	511366.1482	131	46461764.80	233.2	614840073.2	315.6

- **Cas d'un plan avec 3 facteurs : $k = 3, N = 10, n = 20, p = 10$.**

Nombre d'itération	1		2		3	
Algorithmes	Déterminant	Temps	Déterminant	Temps	Déterminant	Temps
DONEV	4.554568187	2.5	5.513151445	4.4	6.600293659	5.9
DONEV modifié	14.56918007	2.8	43.77904009	4.8	125.0240720	6.2
Nombre d'itération	4		5		6	
Algorithmes	Déterminant	Temps	Déterminant	Temps	Déterminant	Temps
DONEV	7.872398429	7.4	9.323979477	9.5	11.02510919	9.4
DONEV modifié	307.7387079	6.9	656.1389880	7.7	1342.344121	10.8
Nombre d'itération	7		8		9	
Algorithmes	Déterminant	Temps	Déterminant	Temps	Déterminant	Temps
DONEV	12.94245083	10.2	15.18514623	11.1	17.71698558	12.5
DONEV modifié	2631.647005	12.2	5065.016455	12.6	9736.758288	13.7
Nombre d'itération	10		15		20	
Algorithmes	Déterminant	Temps	Déterminant	Temps	Déterminant	Temps
DONEV	20.65960344	17.9	43.08613666	25.1	83.15370781	32
DONEV modifié	16804.37484	14.6	100201.0669	18.9	108473.6374	26.6

- **Cas d'un plan avec 5 facteurs : $k = 5, n = 23, p = 21$.**

Nombre d'itération	1		2		3	
Algorithmes	Déterminant	Temps	Déterminant	Temps	Déterminant	Temps
DONEV	0.0020954747	33.9	0.0028028579	71.1	0.0036753303	88.6
DONEV modifié	0.08644990857	37.5	2.871575591	60.3	59.02125237	86.2
Nombre d'itération	4		5		6	
Algorithmes	Déterminant	Temps	Déterminant	Temps	Déterminant	Temps
DONEV	0.0048280404	137.5	0.0063554299	149.4	0.0083359784	192.1
DONEV modifié	881.3352419	135.1	11758.96546	185.2	79909.88259	217
Nombre d'itération	7		8		9	
Algorithmes	Déterminant	Temps	Déterminant	Temps	Déterminant	Temps
DONEV	0.0108567471	196.8	0.0140659246	244.5	0.0182030911	295.8
DONEV modifié	505515.8772	221.1	2919907.525	242.2	8802819.937	298.8
Nombre d'itération	10		15		20	
Algorithmes	Déterminant	Temps	Déterminant	Temps	Déterminant	Temps
DONEV	0.0235115713	361.6	0.0781222316	476.9	0.2402342136	698.7
DONEV modifié	34774205.17	304.9	75113157740	524.8	13634804620000	703

REFERENCES

[1] Philippe Triboulet (lycée Niepce – chalon sur Saône). Notions de bases sur les plans d'expériences. 09/09/2008.

[2] Jacques Goupy. Plans d'expériences pour surfaces de réponse : Technique et ingénierie, Série Génie industriel. Dunod.2009.

[3] Stéphane Vivier. Stratégies d'optimisation par la méthode des plans d'expériences et Application aux dispositifs électrotechniques modélisés par éléments finis. Doctorat délivré conjointement par l'Ecole ventrale de Lille et l'Université des sciences et technologies de Lille. 2002.

[4] Hichem El Mossaoui. proposition de plans d'experiences dans la methodologie de la recherche experimentale: étude et programmation. Mémoire de magister de mathematiques. Université de Blida. 2004.

[5] Pierre Franquart. Optimisations multi-critères et Méthodologie de la recherche expérimentale. Thèse pour obtenir le grade de Docteur en sciences de l'université de droit, d'économie et des sciences d'AIX-MARSEILLE 3. 1992.

[6] Anne Peissik. Méthodologie de la recherche expérimentale propriétés et caractéristique des matrices d'expériences pour les modèles polynomiaux du second degré. Thèse pour obtenir le grade de Docteur en sciences de l'université de droit, d'économie et des sciences d'AIX-MARSEILLE 3. 1995.

[7] Jean Jaque Droesbeke. Plan d'expérience, Application à l'Entreprise. Jeanne Fine. Gilbert Saporta. Édition technip Paris. 1997.

[8] Maurice Pillet. Les plans d'expériences par la méthode TAGUCHI. les éditions d'organisation.

[9] Tinsson, W. Plans d'expériences : constructions et analyses statistiques. 2010.

[10] Fabian Triefenbach, Design of Experiments : The D-Optimal Approach and Its Implementation As a Computer Algorithm, 2008.

[11] Stéphane GAZUT, Conception et mise en œuvre de nouvelles méthodes d'élaboration de plans d'expériences pour l'apprentissage de modèles non linéaires, Thèse de doctorat, Université PARIS-SUD XI, Faculté des sciences d'Orsay, 2007.

[12] Jean-Pierre Gauchi, Plans d'expériences optimaux : un exposé didactique, INRA, Unité de Mathématique Appliquées (MIA), Jouy-en-Josas, France, 2005.

[13] Richard LINDER, Les plans d'expériences, Un outil indispensable à l'expérimentateur, Collection du Laboratoire Central des Ponts et Chaussées.

[14] A.C. Atkinson, A. N. Donev, and R. D. Tobias, Optimum experimental design, with SAS, Oxford statistical science series.

[15] Toby J.Mitchell, An algorithm for the construction of "D-optimal" experimental designs, Technometrics.Vol. 16. No. 2. May 1974.

[16] Johan Dreo-Alain , Petrowski, patrick Siarry-Erie, Taillard, Méta heuristiques pour l'optimisation difficile, Ouvrage coordonné par Patrick Siarry .

[17] Approche méta heuristique basées sur la méthode de recherche harmonique pour la résolution des problèmes d'ordonnancement industriels, 2013.

[18] Messaoudi Ouchen, Résolution du problème d'ordonnancement $P/prec/C_{max}$ par un algorithme de colonie de Fourmis, Mémoire de magister.

[19] Joël Quinqueton, CERIC et LIRMM, Montpellier, Aspects socio-organisationnels dans les systèmes multi-agents : l'intelligence artificielle en essaim.

[20] Laurane Margot, Les algorithmes fourmis, Mars 2006.

[21] Fabian TEHEUX, Algorithme d'optimisation par colonie de fourmis : développement et application à la prédiction ab initio de la structure native des protéines, Mémoire

présenté sous la direction du Prof. Marianne ROOMAN et du Prof. Esteban ZIMANYI en vue de l'obtention du grade de Licencié en Informatique, Année académique 2005–2006.

[22] Walid TFAILI, Conception d'un algorithme de colonie de fourmis pour l'optimisation continue dynamique, THÈSE DE DOCTORAT DE L'UNIVERSITÉ PARIS 12-VAL DE MARNE UFR de Sciences et Technologie, 13 décembre 2007 .