RAIRO. RECHERCHE OPÉRATIONNELLE

E. PERRIN
A. MANDRILLE
M. OUMOUN
C. FONTEIX
I. MARC

Optimisation globale par stratégie d'évolution

RAIRO. Recherche opérationnelle, tome 31, n° 2 (1997), p. 161-201

<http://www.numdam.org/item?id=RO_1997__31_2_161_0>

Recherche opérationnelle/Operations Research
(vol. 31, n° 2, 1997, pp. 161-201)

OPTIMISATION GLOBALE PAR STRATÉGIE D'ÉVOLUTION (*)

Technique utilisant la génétique des individus diploïdes

par E. PERRIN [1], A. MANDRILLE [1],
M. OUMOUN [2], C. FONTEIX [1] et I. MARC [1]

Communiqué par Pierre TOLLA

Résumé. – *Ce papier présente un nouvel algorithme d'optimisation globale s'inspirant des algorithmes génétiques et des algorithmes à stratégie d'évolution. Cette classe d'algorithmes est caractérisée par une recherche stochastique multi-points utilisant les capacités d'adaptation d'une population au cours de générations successives.*

L'algorithme proposé est basé sur un premier algorithme développé au laboratoire utilisant une représentation génétique du problème d'optimisation. La principale différence réside en la modélisation individuelle qui, à l'origine, était de type haploïde. Dans le présent travail, l'algorithme utilise un modèle individuel plus évolué qui est de type diploïde.

La structure de l'algorithme est décrite et la convergence asymptotique est démontrée. L'influence des différents paramètres de l'algorithme est ensuite discutée. Il en ressort l'importance du taux d'homozygotie, mais aussi de la nature de la dominance entre deux allèles, dans la convergence. Finalement, les performances de l'algorithme, avec deux types de dominance, sont comparées avec un algorithme génétique classique, ou hybride, dans le cas de la maximisation d'une fonction multimodale.

Ce travail met en évidence l'efficacité et les potentialités de l'algorithme.

Mots clés : Optimisation globale, stratégie évolutive, opérateurs génétiques, individus diploïdes.

Abstract. – *In this paper, a new algorithm for global optimization, based on genetic algorithms and evolution strategies, is presented. This class of algorithms is characterized by a stochastic search on sets of points and uses natural adaptive population ability.*

The proposed algorithm follows one which was first developped in the laboratory and used a genetic model for the optimization problem. The main difference lies in the modelling of individuals which first used the haploïd model. In the present work, a more evolved model is used, consisting of a diploid one.

Following the description of the algorithm, a demonstration of asymptotic convergence is provided. Then, the influence of the parameters is evaluated showing the great importance of homozygocity rate and the nature of the dominance. A maximisation problem is finally carried out, and the

(*) Reçu en juin 1994.

([1]) Laboratoire de Sciences du Génie Chimique, UPR 6811/CNRS-INPL, 1, rue Grandville, BP 451, 54001 Nancy.

([2]) INRIA, Lorraine, Technopôle, Metz-2000, 4, rue Marconi, 57070 Metz.

Recherche opérationnelle/Operations Research, 0399-0559/97/02/$ 7.00

performances with the two types of dominance are compared with those obtained through the intermediary of a classical and a hybrid genetic algorithm.
In conclusion, this study shows the efficiency and potentialities of such an algorithm.

Keywords: Global optimization, evolution strategy, genetic operators, diploids individuals.

I. INTRODUCTION

Dans le cadre d'études liées aux procédés complexes, généralement non linéaires et soumis à contraintes, la résolution de problèmes d'optimisation au moyen de techniques classiques est discutable dans bien des cas. En effet, pour que ces techniques soient efficaces il faut les adapter au problème traité. Il faut donc disposer d'une certaine connaissance sur la fonction à optimiser. Cette connaissance sera utilisée soit pour le réglage des paramètres de recherche, soit lors de la recherche (gradients de la fonction), soit pour l'initialisation. Ce dernier point est le plus critique car la recherche risque d'aboutir à une solution locale et non globale. Afin de s'affranchir de toute expertise durant la recherche, qui est nécessairement coûteuse en temps et en moyens, l'approche du problème dans sa globalité s'avère indispensable. Il est donc nécessaire de mettre au point des outils généraux d'optimisation globale utilisables en génie des procédés.

Des approches, prometteuses, issues de l'informatique avancée sont en plein développement depuis une vingtaine d'années. Il s'agit de l'émergence des bioalgorithmes basés sur une modélisation des grands principes d'évolution naturelle. Deux approches principales ont été développées : Les Algorithmes Génétiques (AG) et les Algorithmes à Stratégie (ASE). La grande caractéristique de ces algorithmes repose sur une recherche stochastique multipoints qui est indépendante de toute connaissance sur la fonction à optimiser. Les stratégies développées s'appuient sur les principes d'évolution naturelle d'une population d'individus, où chaque individu représente une solution potentielle du problème.

Les concepts de base des AG ont été dévelopés par Holland *et al.* [1]. Ces algorithmes se basent sur une connaissance de la génétique des populations. Un AG utilise alors les capacités d'adaptation d'une population donnée aux contraintes extérieures, ceci au cours de générations successives. L'évolution est guidée par l'utilisation d'opérateurs génétiques. L'ensemble des variables est modélisé par un code binaire représentant le chromosome d'un individu haploïde.

Les Algorithmes à Stratégie d'Évolution ont été introduits par Rechenberg *et al.* [2]. Ils utilisent les principes d'évolution d'une population d'individus haploïdes dont les gènes sont des nombres réels.

En s'appuyant sur les principes de ces techniques, un premier algorithme original a été développé au laboratoire [3]. Les opérateurs utilisés relèvent des AG et la représentation individuelle s'inspire de celle utilisée dans les ASE, à savoir des modèles d'individus haploïdes. Les études effectuées sur cet algorithme, afin d'en améliorer les performances, ont mis en évidence la nécessité de créer des enfants en dehors du domaine de naissance défini par les parents, afin de mieux couvrir le domaine de recherche. Une codification intéressante consiste à utiliser non plus un modèle d'individus haploïdes, mais un modèle d'individus diploïdes. Ce type de représentation permet alors de se rapprocher de structures plus évoluées, et de conserver un maximum d'information. Ce dernier algorithme fait l'objet du présent travail. Tout d'abord, nous décrivons l'algorithme, puis, nous en étudions la convergence. L'influence des paramètres est ensuite discutée. Finalement, nous comparons l'algorithme diploïde de base, dit à dominance partagée (AGDP), à un AG classique, puis une forme différente, dite à dominance vraie (AGDV), à un AG hybride.

II. DESCRIPTION DE L'ALGORITHME

II.1. Position du problème

Dans le cas de la minimisation d'une fonction $f : \mathbb{R}^L \rightarrow \mathbb{R}$, le problème d'optimisation globale consiste à trouver un vecteur $x^* \in \mathbb{R}^L$ tel que $\forall x \in \mathbb{R}^L$, $f(x^*) \leq f(x)$. x^* s'appelle l'optimum global. Cette définition est tout aussi valable pour un problème de maximisation qui peut se traduire en terme de minimisation par : $\text{Max}\{f(x)\} = -\text{Min}\{-f(x)\}$. Une des difficultés rencontrées dans la recherche d'un optimum global est la présence éventuelle d'optimums locaux $\hat{x}$ définis par : $\exists \varepsilon \in \mathbb{R}^L$, $\forall x \in \mathbb{R}^L : \|\hat{x} - x\| < \varepsilon \Rightarrow f(\hat{x}) \leq f(x)$.

Nous nous proposons d'aborder les problèmes décrits par la forme générale suivante :

$$\text{Min}\{f(x) | x \in X\}; \qquad f : \mathbb{R}^v \times \mathbb{R}^w \rightarrow \mathbb{R}, \quad \text{avec } v + w = L.$$
$$X = \{x \in \mathbb{R}^v \times \mathbb{R}^w | 1_j \leq x_j \leq u_j,\ j = 1, 2, ..., L;$$
$$c_k(x) \leq b_k,\ k = 1, 2, ..., m\},$$

où chaque x_j représente une composante du vecteur x, l_j et u_j étant respectivement ses limites inférieures et supérieures. $c_k(x)$ représentent les contraintes. On suppose v, w, m, l_j et u_j connus. L représente le nombre de variables du problème, qui sont divisées en v variables réelles et w variables entières.

Avant de détailler plus avant l'algorithme, il est nécessaire de définir au préalable un certain nombre de concepts utilisés pour la représentation génétique du problème d'optimisation.

II.2. Modèle diploïde

Un individu diploïde est représenté par un génome formé de deux chromosomes possédant chacun L gènes. La valeur d'un gène est appelée allèle. L'individu est caractérisé par son phénotype qui résulte d'une combinaison entre allèles de même rang en considérant la dominance d'un gène sur son homologue.

Plus concrètement, chaque gène représente une variable du problème à traiter. Les dominances sont propres à chaque couple d'allèles. Elles peuvent être partagées (ou partielles) et sont représentées par des nombres réels dans l'intervalle [0; 1], ou elles peuvent être vraies (ou totales) en prenant les valeurs 0 ou 1. Le premier type de dominance permet d'obtenir des caractères partagés entre deux allèles, contrairement au second, qui permet de représenter l'un ou l'autre des allèles.

Un individu est défini par trois vecteurs (voir *fig.* 1) :

- $C = (g_1, g_2, ..., g_j, ..., g_L)$ qui repésente le premier chromosome.
- $C' = \left(g'_1, g'_2, ..., g'_j, ..., g'_L\right)$ qui repésente le second chromosome.
- $D = (d_1, d_2, ..., d_j, ..., d_L)$ qui représente le vecteur des dominances.

Ainsi le phénotype d'un individu s'écrit $x = (x_1, x_2, ..., x_j, ..., x_L)$, où chaque composante est définie par :

$x_j = d_j\, g_j + (1 - d_j)\, g'_j$, pour les variables réelles.

$x_j = \textit{partie-entière}\,(d_j\, g_j + (1 - d_j)\, g'_j)$, pour les variables entières.

II.3. Paramètres de l'algorithme

L'algorithme utilise cinq paramètres constants :

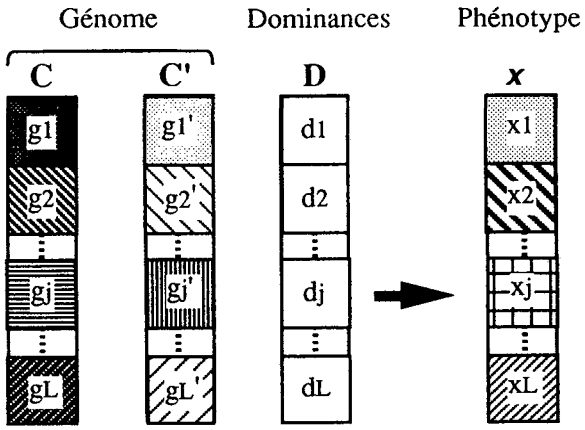


Figure 1. – Représentation d'un individu diploïde. Le phénotype d'un individu est caractérisé par son génome et son vecteur de dominances.

– la taille de la population ($N \in \mathbb{N}$), représente le nombre d'individus à chaque génération

– le nombre de gènes ($L \in \mathbb{N}$) correspond au nombre de variables du problème à traiter.

– le taux de mutation ($M \in [0; 1]$) indique dans quelle proportion appliquer l'opération de mutation lors des naissances.

– le taux de survie ($G \in [0; 1]$) représente le patrimoine génétique transmis d'une génératon à l'autre. Autrement dit, le taux de mortalité $1 - G$ définit la sélectivité entre deux générations.

– le taux d'homozygotie ($H \in [0; 1]$) indique dans quelle proportion appliquer l'opérateur d'homozygotie lors des naissances.

– la précision de la recherche ($\varepsilon \in \mathbb{R}$) est utilisée comme critère d'arrêt.

II.4. Opérateurs génétiques

L'algorithme utilise deux opérateurs primaires et trois opérateurs dits secondaires. *Opérateurs primaires* :

– la *sélection*, ou mortalité, consiste à éliminer les individus les moins adaptés, c'est-à-dire les $N(1 - G)$ points pour lesquels la fonction est la moins optimale. Il s'agit d'une stratégie élitiste.

– durant la phase de *reproduction*, ou naissance, on sélectionne aléatoirement deux individus (parents), et on reconstruit un individu

(enfant) par l'action des opérateurs secondaires : le croisement, la mutation, l'homozygotie.

Opérateurs secondaires :

– le *croisement* est un processus d'échange inter chromosomique au cours duquel des recombinants (nouvelles combinaisons de gènes liés) sont formés. Concrètement, il consiste à créer un enfant pour lequel chacun des deux chromosomes est caractéristique de chacun des deux parents. Chaque allèle

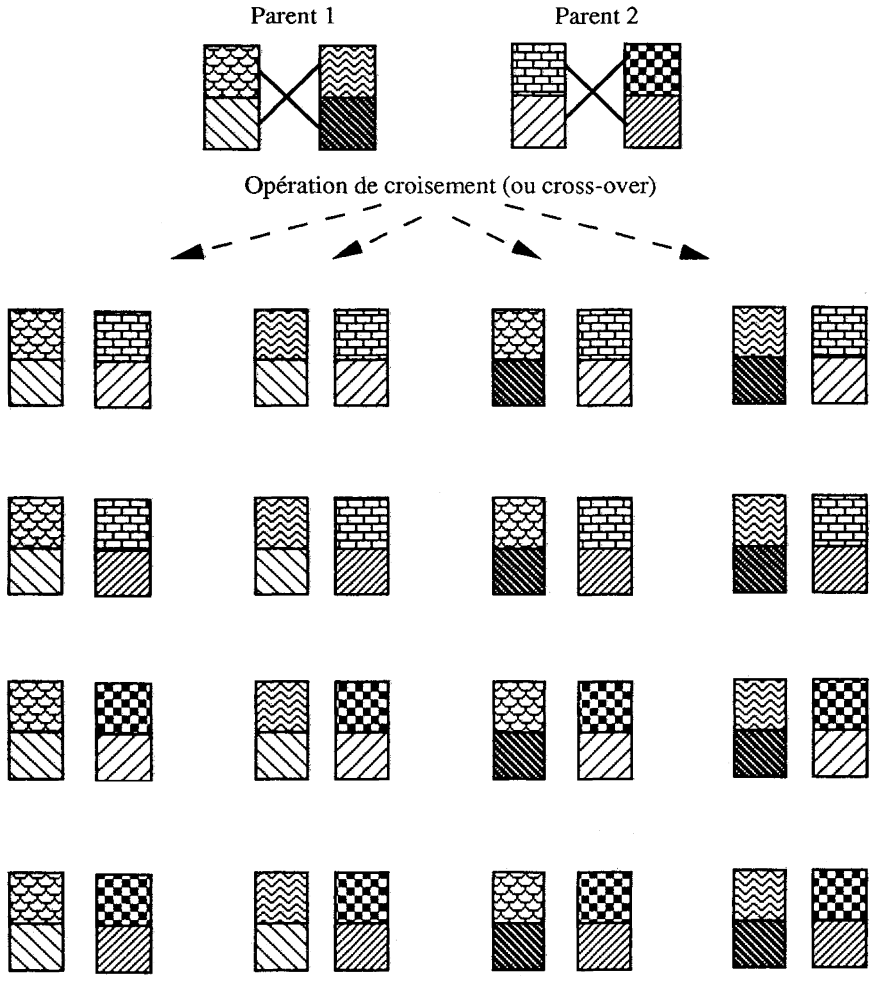


Figure 2. – Illustration des différentes combinaisons possibles lors d'une opération de croisement à partir des parents 1 et 2, pour $L = 2$. Parmi ces 16 possibilités, un seul enfant sera retenu.

d'un chromosome de l'enfant hérite aléatoirement de l'un des deux allèles du génome parental (processus multi-crossover). La figure 2 représente toutes les combinaisons possibles à partir de parents possédant deux gènes. Sous une forme différente, la figure 3 illustre le domaine dans lequel se trouvera le phénotype de l'enfant créé. Le vecteur des dominances est tiré aléatoirement dans $[0; 1]^L$.

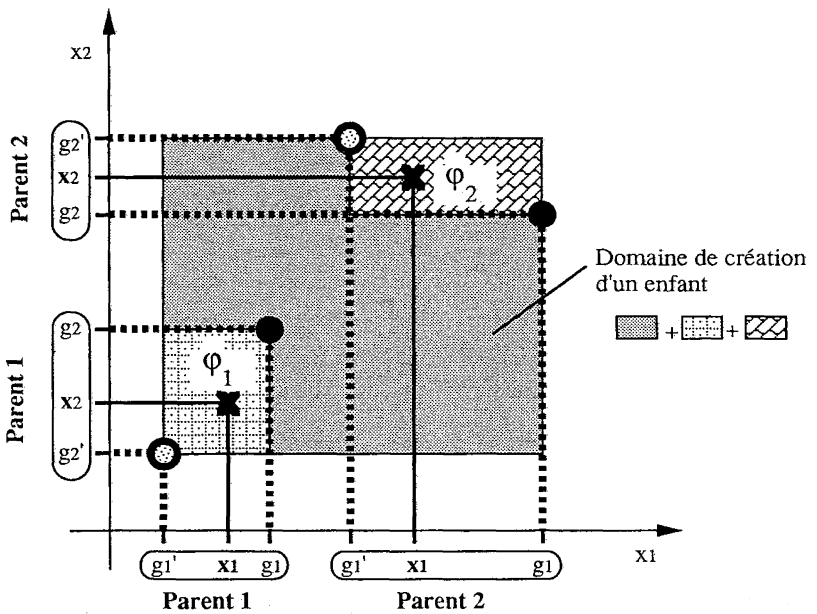


Figure 3. – Illustration de la reproduction. Domaine de création d'un enfant par recombinaison. φ_1 et φ_2 représentent les phénotypes respectifs des parents 1 et 2. ● et ◎ correspondent aux chromosomes C et C′ de chacun des parents.

– la *mutation* consiste, pour l'enfant sélectionné, à tirer aléatoirement la totalité des gènes des deux chromosomes, et la totalité des dominances.

– l'*homozygotie* consiste à modifier un enfant en recopiant son phénotype sur ses deux chromosomes. Ceux-ci sont alors identiques. Les dominances demeurent, quant à elles, inchangées.

Un individu dont les taux chromosomes sont différents est dit hétérozygote.

Les opérations de mutation et d'homozygotie sont appliquées selon leurs taux respectifs (M et H). Ceci signifie qu'un enfant peut muter avec une probabilité égale à M, et peut être homozygote avec une probabilité égale à H.

II.5. Stratégie de recherche

La stratégie de recherche est schématisée dans la figure 4. À partir d'une population initiale composée de N individus hétérozygotes créés aléatoirement dans l'espace de recherche, la population va évoluer au cours des générations (itérations), pour progressivement converger vers la solution.

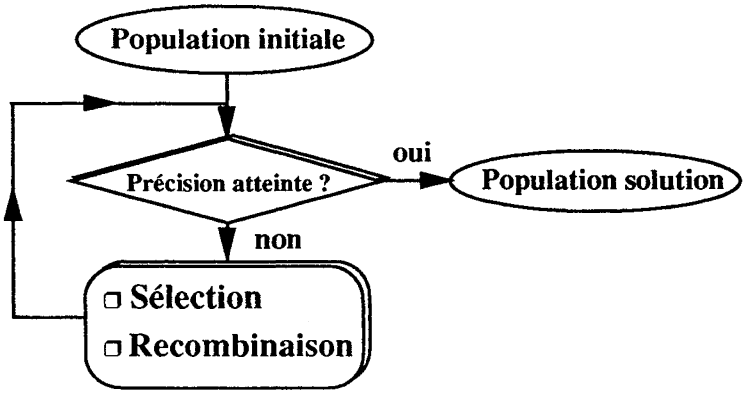


Figure 4. – Organigramme général de l'algorithme proposé.

Une génération est soumise à deux phrases consécutives. La première, appelée phase de décès, consiste à appliquer l'opérateur de sélection à la population considérée. En pratique, les individus sont triés dans un ordre croissant (minimisation), ou décroissant (maximisation), suivant leur capacité d'adaptation (valeur de f). Les $S = NG$ premiers individus sont conservés, et représentent des parents potentiels pour la phase suivante, dite phase de naissance. Cette dernière utilise l'opérateur de recombinaison autant de fois que nécessaire pour créer $N(1 - G)$ enfants viables. Un enfant est viable si il est au moins aussi bon que le parent le moins performant.

L'évolution se termine lorsqu'à la fin d'une génération la dispersion sur la population (écart entre les valeurs extrêmes de la fonction à optimiser) est inférieure à la précision (ε) requise.

III. CONVERGENCE DE L'ALGORITHME

III.1. Démonstration de la convergence

La convergence est démontrée ici dans le cas d'une minimisation. Cette démonstration est directement transposable au cas d'une maximisation.

Soient $x_1^p, ..., x_i^p, ..., x_N^p$ les N individus de la génération p, $(p \in \mathbb{N})$, et on pose :

$$F_i^p = f(x_i^p)$$

On note λ la mesure de Lebesgue sur $\mathbb{R}^v \times \mathbb{N}^w$ et I la borne inférieure essentielle de f; si f est une fonction continue, I coïncide avec la borne inférieure de f. Dans la suite, on montre que les nombres F_i^p tendent vers I lorsque p tend vers l'infini.

Par définition de l'infimum essentiel, on peut écrire :

$$\forall \varepsilon > 0, \quad \exists B \subset X, \quad (\lambda(B) > 0), \quad x \in B \Rightarrow 0 \leq f(x) - I \leq \varepsilon.$$

Pour $i \in \{1, ..., S\}$ fixé, où $S = NG$ représente le nombre de survivants d'une génération, la suite $(F_i^p)_{p \geq 0}$ est décroissante et minorée, elle est donc convergente de limite $l_i \geq I$. On montre que $l_i = I$ pour tout $i \in \{1, ..., S\}$, par récurrence sur i.

$i = 1$: supposons que $l_1 = \lim_{p \to +\infty} F_1^p > I$. Si, à une certaine génération p_0, un enfant x^{p_0} vérifiant $f(x^{p_0}) < l_1$ est engendré, cela contredira le fait que $F_I^p \geq l_1$; $\forall p \geq 0$ car $f(x_1^{p_0+1}) \leq f(x^{p_0}) < l_1$. On vérifie que la probabilité d'apparition d'un tel individu tend vers 1 lorsque p_0 tend vers l'infini.

Soit $\varepsilon = \frac{l_1 - I}{2}$ par définition de I, il existe $B \subset X$ tel que $\lambda(B) > 0$ et $0 \leq f(x) - I < \varepsilon$ pour tout $x \in B$, par conséquent :

$$\forall x \in B, \quad f(x) < I + \varepsilon = \frac{l_1 - I}{2} + I = \frac{l_1 + I}{2} < l_1.$$

La probabilité, à une génération donnée, pour qu'un enfant x soit engendré et se retrouve, suite à une mutation, dans B, est égale à $M \frac{\lambda(B)}{\lambda(X)}$. Ainsi, la probabilité pour qu'au cours de p générations, un individu au moins soit muté et se retrouve dans B, est égale à $1 - q^{p(N-S)}$ $(q = 1 - M \frac{\lambda(B)}{\lambda(X)}$, en se rappelant que M est la probabilité d'apparition d'un mutant parmi les enfants). Il est clair que cette probabilité tend vers 1 lorsque p tend vers l'infini, par conséquent la probabilité pour que la limite de la suite $(F_1^p)_{p \geq 0}$ soit différente de I est nulle.

HYPOTHÈSE DE RÉCURRENCE : On suppose le résultat vrai pour $i = 1, ..., i_0$ $(i_0 < S)$. On montre que $F_{i_0+1}^p$ converge vers I. On suppose donc

que : $\lim_{p\to+\infty} F_{i_0+1}^p = l_{i_0+1} > I$. Comme les suites $F_i^p\,(i = 1, ..., i_0)$ convergent vers I, il existe $\varepsilon > 0$ et $p_0 \in \mathbb{N}$ tels que

$$\forall p \geq p_0,\ F_i^p \in [I, I+\varepsilon[\qquad \text{et} \qquad \forall p \geq p_0,\ F_{i_0+1}^p \in [l_{i_0+1}, l_{i_0+1}+\varepsilon[$$

avec $I + \varepsilon < l_{i_0+1}$.

Comme précédemment, on peut montrer que la probabilité d'apparition, à une génération $p > p_0$, d'un enfant x vérifiant $f(x) \in [I, I+\varepsilon[$ tend vers 1 lorsque p tend vers l'infini. Si un tel individu apparaît effectivement à la génération p_1, alors on aura $f(x_{i_0+1}^{p_1+1}) \leq I + \varepsilon < l_{i_0+1}$ puisque $f(x_i^{p_1+1}) \leq I + \varepsilon$ pour $i = 1, ..., i_0$. Ceci contredit l'inégalité $l_{i_0+1} > I$ et implique que la probabilité pour que $l_{i_0+1} > I$ est nulle.

Ayant montré que $\lim_{p\to+\infty} F_i^p = I$ pour $i = 1, ..., S$, il s'ensuit que le même résultat est vrai pour i quelconque car $F_i^{p+1} \leq F_S^p$ pour tout $i > S$.

Il est à noter l'importance des mutations pour s'assurer de la convergence vers l'optimum global. De plus, le test d'arrêt utilisé en pratique, et qui n'est pas traité dans la démonstration, s'il est vérifié trop rapidement, risque d'arrêter la recherche sur un optimum local.

III.2. Illustration de la convergence

Pour illustrer le comportement de l'algorithme, nous proposons d'observer son évolution au cours des itérations lors de la minimisation de la fonction suivante :

$$f_1(x) = 0,5\,(x_1^2 + x_2^2) + \sin(x_1^2 + x_2^2) \qquad \text{avec} \quad x = (x_1, x_2) \in \mathbb{R}^2.$$

Cette fonction possède un minimum global $f_1(x^*) = 0$ pour $x^* = (0, 0)$, et une infinité de minimums locaux.

On définit par Ω le domaine de détection du minimum absolu (*fig.* 5). X est le domaine de recherche, et D_p le domaine de viabilité lors de la phase de naissance à la génération p. Ce dernier est défini par $D_p = \{x \in R^\nu \times N^w | f(x) \leq f(\tilde{x}^p)\}$ où $\tilde{x}^p$ représente le parent le moins performant. Pour l'exemple traité, D_p est inclus dans le disque de rayon $r = \|\tilde{x}^p\|$ (*fig.* 6). Lors de la génération p, on note $M_p(i)$ la probabilité pour que i individus survivants appartiennent à Ω suite à la phase de mortalité, q_p la probabilité pour qu'un enfant naisse dans Ω, $N_p(i)$ la probabilité pour que i enfants naissent dans Ω, $P_p(i)$ la probabilité pour que i individus

appartiennent à Ω suite à la phase de naissance, et E_p la proportion moyenne d'individus appartenant à Ω.

Étudions l'évolution de E_p en supposant une répartition uniforme de la population initiale.

Initialisation (génération 0) : Probabilité pour qu'un individu de la population initiale se trouve dans Ω : $q_0 = \text{Min}\left(1, \frac{\lambda(\Omega)}{\lambda(X)}\right)$

Probabilité pour i individus : $P_0(i) = C_N^i \, q_0^i \, (1 - q_0)^{N-i}$.

Proportion moyenne d'individus appartenant à Ω : $E_0 = q_0$.

Autre génération (génération p) : *Phase de mortalité* :

$$M_p(i) = \begin{cases} P_{p-1}(i) & \text{si} \quad i < S \\ \sum_{i=S}^{N} P_{p-1}(i) & \text{sinon,} \end{cases}$$

où $S = GN$ représente le nombre d'individus survivants.

Phase de naissance : Probabilité pour qu'un enfant naisse dans Ω, en supposant uniforme sa répartition possible dans D_p : $q_p = \text{Min}\left(1, \frac{\lambda(\Omega)}{\lambda(D_p)}\right)$.

Probabilité pour i enfants : $N_p(i) = C_{N-S}^i \, q_p^i \, (1 - q_p)^{N-S-i}$

Finalement, le nombre moyen d'individus dans Ω, à l'issue de la génération p, s'exprime sous la forme :

$$E_p = \frac{\sum_{i=0}^{N} (i \, P_p(i))}{N}, \text{ avec } P_p(i) = \sum_{k=\text{Max}(0, i+S-N)}^{\text{Min}(S,i)} (M_p(k) \, N_p(i-k)).$$

Pour calculer E_p, il faut connaître q_p et par conséquent D_p. Nous étudions, dans la suite, l'évolution du domaine de viabilité D_p.

Posons $\lambda_P = \lambda(D_p)$, et soit $P_p(\lambda_p)$ la densité de probabilité de répartition de D_p. On a $\int_0^{\lambda_0} P_p(\lambda_p) \, d\lambda_p = 1$ et $E(\lambda_p) = \int_0^{\lambda_0} \lambda_p \, P_p(\lambda_p) \, d\lambda_p$.

Soit $P(D_{p+1}/D_p) = P((D_{p+1}/S)/D_p)$ la densité de probablité conditionnelle d'obtenir D_{p+1} pour S (nombre de points conservés de p

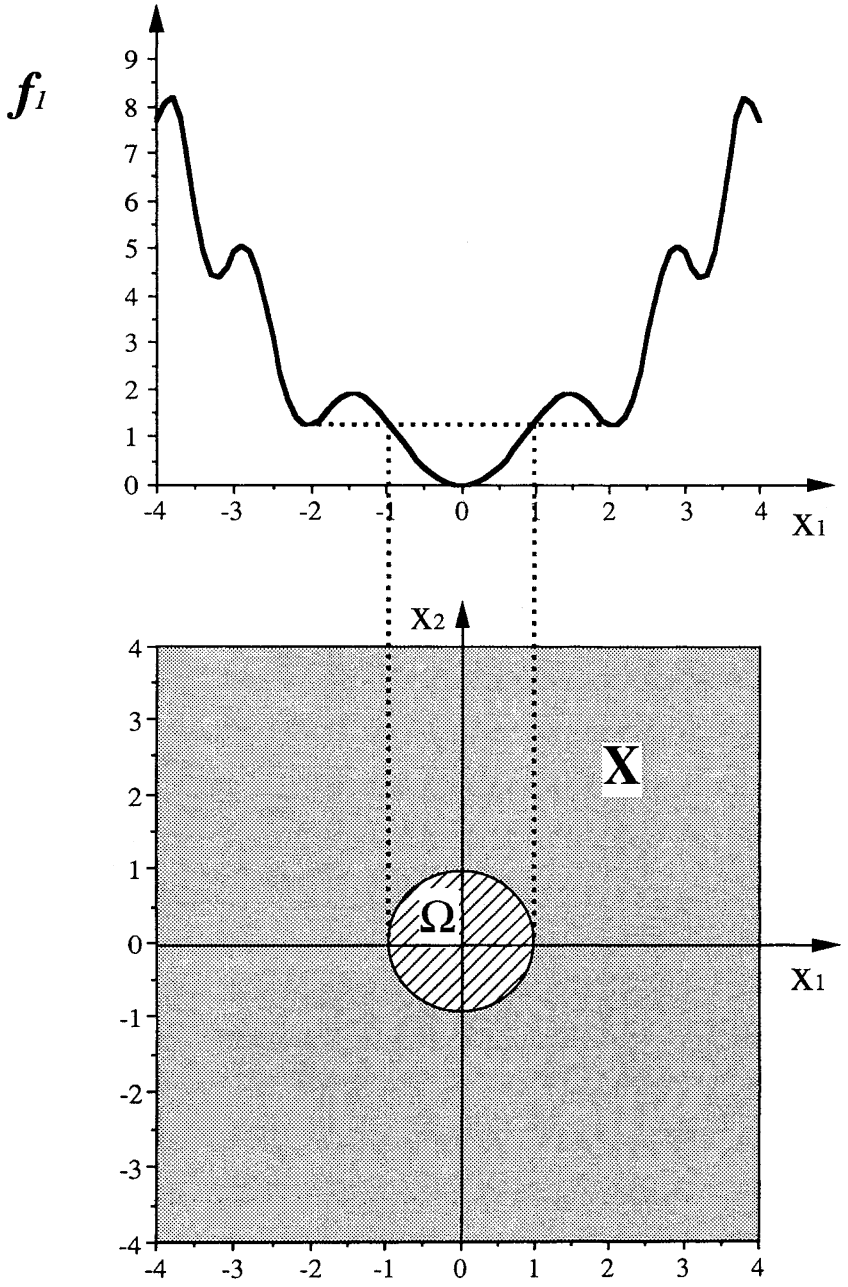


Figure 5. – Allure de $f_1(x)$. Définition du domaine de recherche (X) et domaine de détection du minimum absolu (Ω).

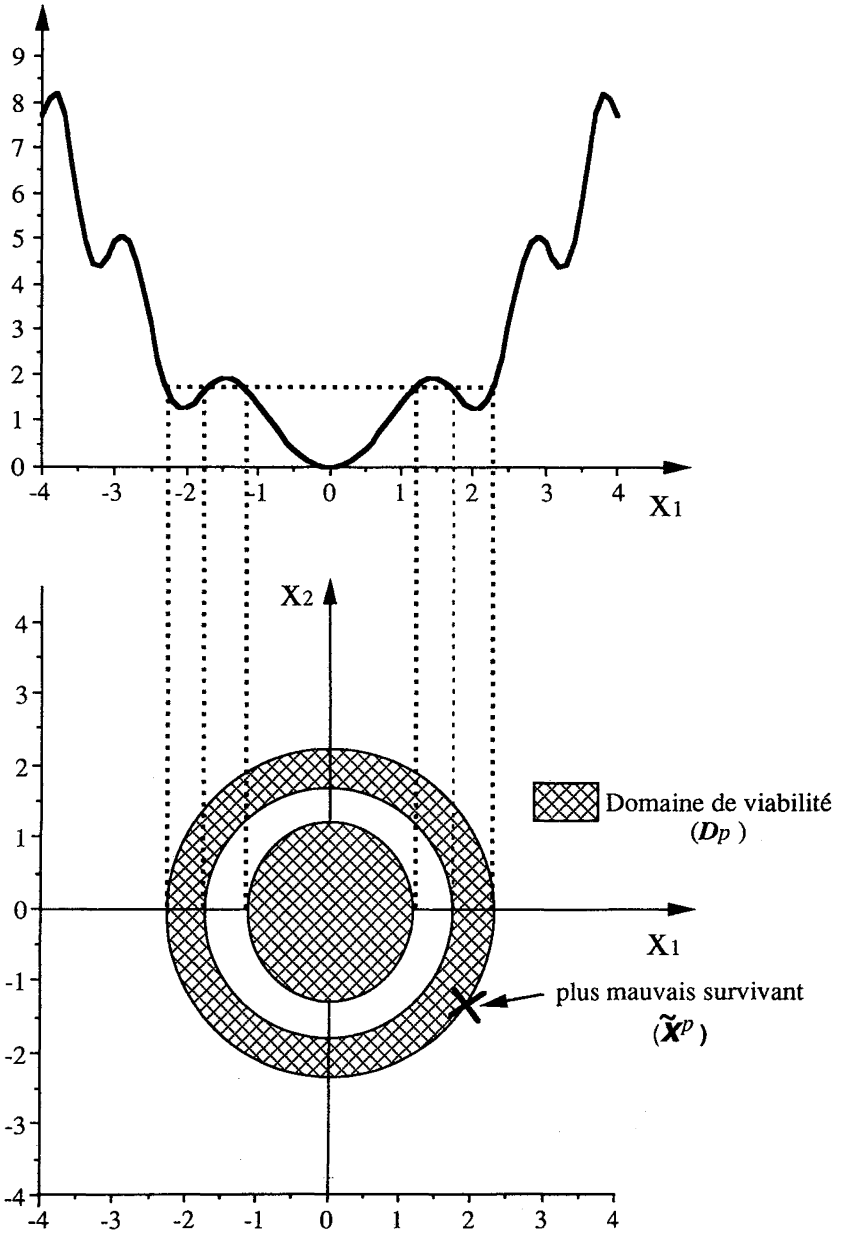


Figure 6. – Exemple de domaine de viabilité.

à $p+1$) et Dp *fixés*. En appliquant la théorie Bayesienne, il est possible d'exprimer la densité de probabilité de répartition de D_{p+1} par :

$$P_{p+1}(\lambda_{p+1}) = \int_0^{\lambda_0} P(D_{p+1}/D_p)\, P_p(\lambda_p)\, d\lambda_p, \text{ qui se réduit à}$$

$$P_{p+1}(\lambda_{p+1}) = \int_{\lambda_{p+1}}^{\lambda_0} P(D_{p+1}/D_p)\, P_p(\lambda_p)\, d\lambda_p.$$

En effet, d'après la stratégie utilisée on a $\forall p,\ \lambda_{p+1} \leq \lambda_p$. C'est pourquoi le cas $(\lambda_{p+1} > \lambda_p)$ implique $(\mathrm{P}(D_{p+1}/D_p) = 0)$.

A ce niveau de l'étude, on peut émettre trois hypothèses simplificatrices :

1) la probabilité de choix sur la valeur de S *a priori* est indépendante de D_p. De plus, elle peut être choisie *a priori* (indépendamment de toute information) entre 0 et N, chaque valeur étant équiprobable on peut se fixer : $P(S/D_p) = \frac{1}{N+1}$.

2) en supposant une répartition uniforme des points dans D_p, la probabilité pour que S points se trouvent dans D_{p+1} fixé (avec $D_{p+1} \subseteq D_p$) est égale à :

$$P[(S/D_{p+1})/D_p] = C_N^s \left(\frac{\lambda_{p+1}}{\lambda_p}\right)^S \left(1 - \frac{\lambda_{p+1}}{\lambda_p}\right)^{N-S},$$

où $\left(\frac{\lambda_{p+1}}{\lambda_p}\right)$ représente la probabilité qu'un point se trouve dans D_{p+1}.

3) en supposant une répartition *a priori* uniforme des D_{p+1} dans D_p, la probabilité d'obtenir un domaine D_{p+1} dans D_p fixé est égale à : $P(d_{p+1}/D_p) = \frac{1}{\lambda_p}$.

La résolution du problème nécessite de vérifier la présence simultanée des événements S et D_{p+1}, c'est-à-dire écrire l'égalité entre leurs densités de probabilité respectives :

$$\begin{aligned} P[S \text{ et } D_{p+1}/D_p] &= P[(S/D_{p+1})/D_p]\, P[D_{p+1}/D_p] \\ &= P[(D_{p+1}/S)/D_p]\, P[S/D_p]. \end{aligned}$$

En exprimant $E(\lambda_{p+1})$ à partir de $\mathrm{P}_{p+1}(\lambda_{p+1})$ et en appliquant les différentes hypothèses proposées, il résulte une évolution théorique moyenne de λ_p qui s'exprime par :

$$E(\lambda_{p+1}) = \frac{S+1}{N+2} E(\lambda_p) \text{ sous forme récurrente,}$$

ou

$$E(\lambda_p) = \left(\frac{S+1}{N+2}\right)^p \lambda(X) \text{ en fonction de la valeur initiale.}$$

Ces résultats théoriques ont été vérifiés par l'expérience. Pour ce faire, l'algorithme a été lancé 100 fois avec les valeurs de paramètres suivantes : $N = 200$; $G = 0,2$; $M = 10^{-2}$; $H = 0,4$. Le test d'arrêt original s'est vu remplacé par un arrêt systématique au bout de 20 générations pour chaque essais.

A chaque génération les données suivantes ont été mémorisées :

– le nombre d'individus n_Ω appartenant à Ω,

– la mesure $\lambda_p = \lambda(D_p)$ qui dans le cas étudié correspond à une surface,

– la répartition des individus dans D_p.

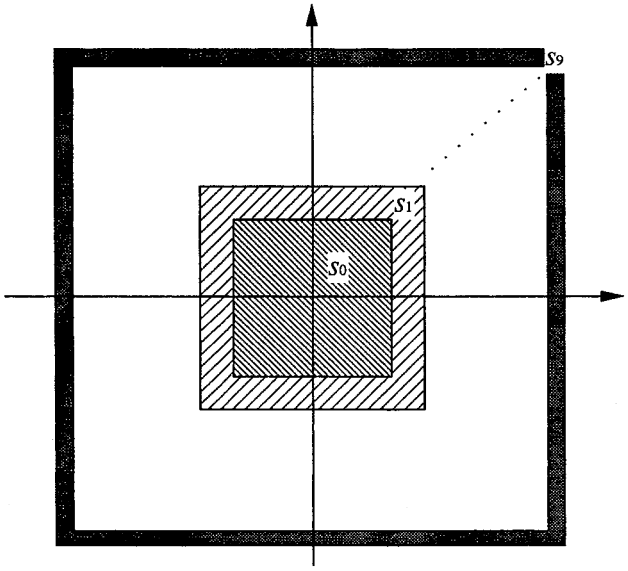
Pratiquement, D_p a été découpé en 10 secteurs s_i (*fig.* 7 *a* et 7 *b*) tels que :

$$\lambda(s_i) = \frac{\lambda(D_p)}{10} \qquad \text{pour} \quad i = 0, 1, ..., 9$$

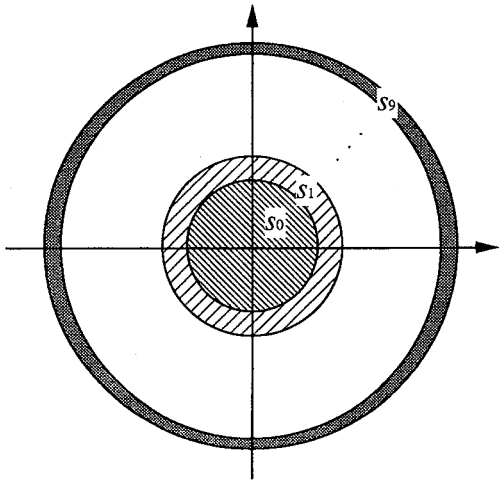
$$\bigcup_{i=0}^{9} s_i = D_p \qquad \text{et} \qquad s_i \cap s_j = \varnothing \qquad \text{avec} \quad i, j = 0, 1, ..., 9 \quad \text{et} \quad i \neq j$$

Les figures 8 *a* à 8 *c* représentent les évolutions théoriques et expérimentales de E_p pour différents domaines de recherche X ($[-4; 4]^2$, $[-256; 256]^2$, et $[0 : 4]^2$). Dans chaque cas, les points expérimentaux sont calculés par $E_p = E\left(\frac{n_\Omega}{N}\right)$, où $E(\ )$ représente l'espérance mathématique sur 100 valeurs. Les réponses sont relativement similaires avec une erreur quasiment systématique entre évolution théorique et évolution expérimentale, qui semble due aux valeurs initiales. Il est à noter que dans le cas où Ω est au centre du domaine de recherche (*fig.* 8 *a* et 8 *b*), l'évolution expérimentale est supérieure à l'évolution théorique. Par contre, dans le cas où Ω se situe sur le bord du domaine (*fig.* 8 *c*), on observe un effet contraire.

L'étude théorique de la répartition de la population initiale met en évidence sa non-uniformité (*fig.* 9 *a*). En effet, il est aisé de montrer que la densité de probabilité de chaque élément du phénotype s'exprime, sous une forme réduite ($0 \leq z \leq 1$), par : $P(z) = -2z\ln(z) - 2(1-z)\ln(1-z)$. Pour le problème traité cette distribution favorisant le centre du domaine de recherche explique que les valeurs expérimentales soient supérieures aux valeurs théoriques lorsque Ω se situe au centre du domaine (*fig.* 8 *a* et 8 *b*). Et



(a)



(b)

Figure 7. – (a) Découpage du domaine de recherche (X) en dix secteurs de même surface (S_0 à S_9); (b) découpage d'un domaine de viabilité (D_p) en dix secteurs de même surface (S_0 à S_9).

lorsque Ω se situe sur le bord du domaine, l'effet inverse se produit (*fig.* 8 *c*). Si une répartition uniforme de la population initiale s'avère nécessaire, il

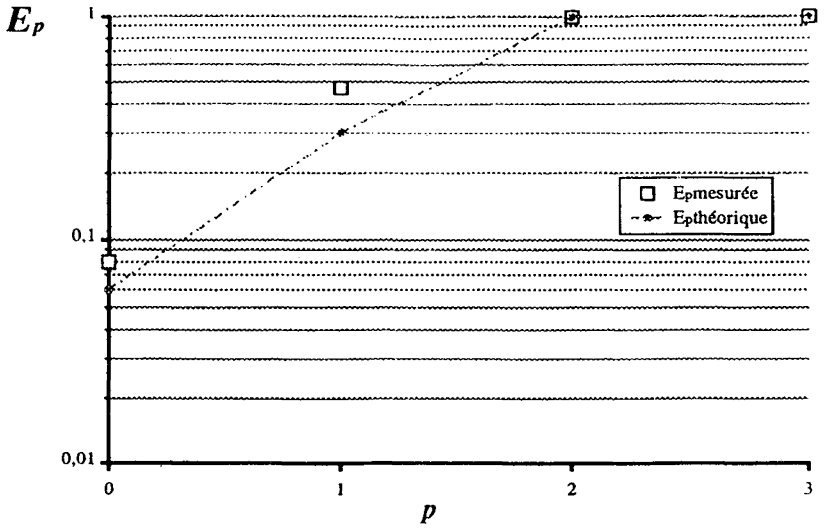


Figure 8 a. – $X = [-4; 4]^2$.

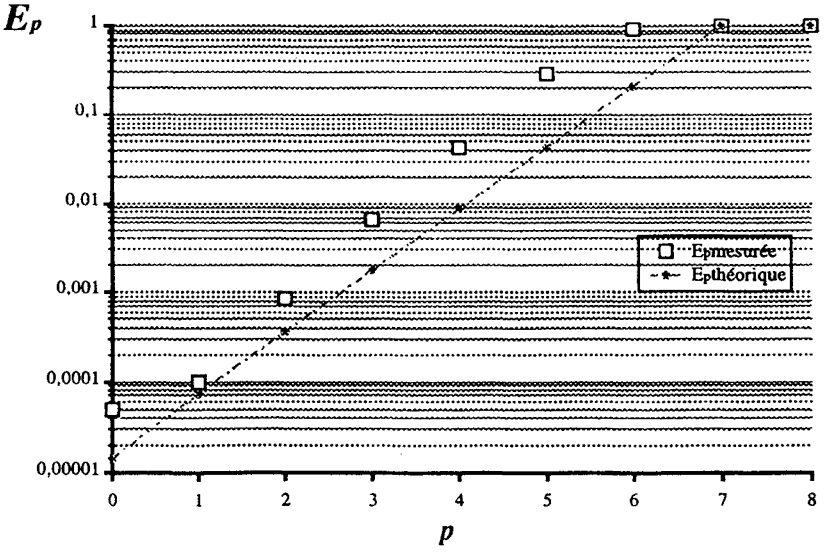


Figure 8 b. – $X = [-256; 256]^2$.

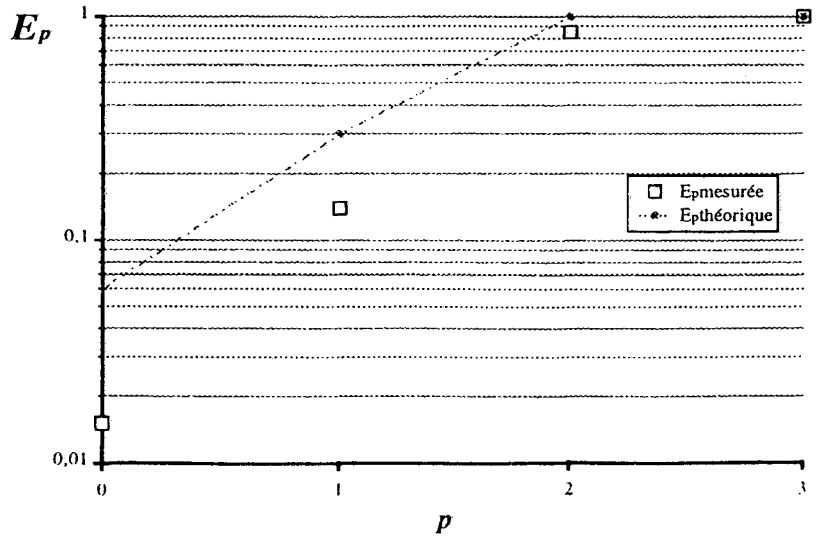


Figure 8 c. – $X = [0; 4]^2$.

suffit, par exemple, de ne pas tirer au sort les dominances mais de les fixer à 1 (dominance vraie).

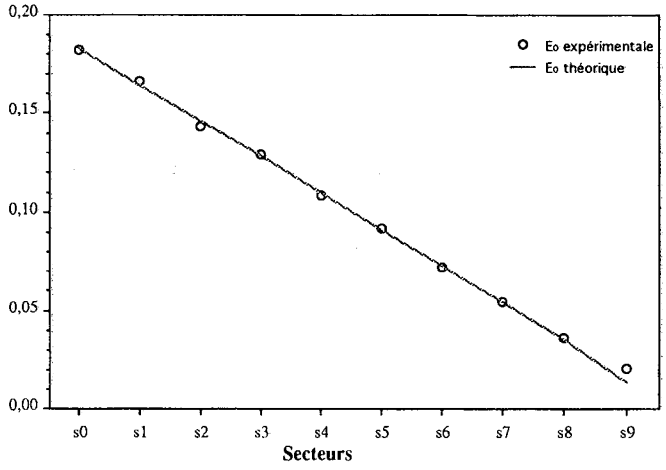
La distribution de la population au cours des générations suivantes (*fig.* 9 *b*), c'est-à-dire pour $p > 0$, n'est pas non plus uniforme. Cette répartition, qui est due aux mécanismes de reconstruction de la population (croisement et homozygotie), permet à l'algorithme de converger plus rapidement que prévu.

IV. INFLUENCE DES PARAMÈTRES DE L'ALGORITHME

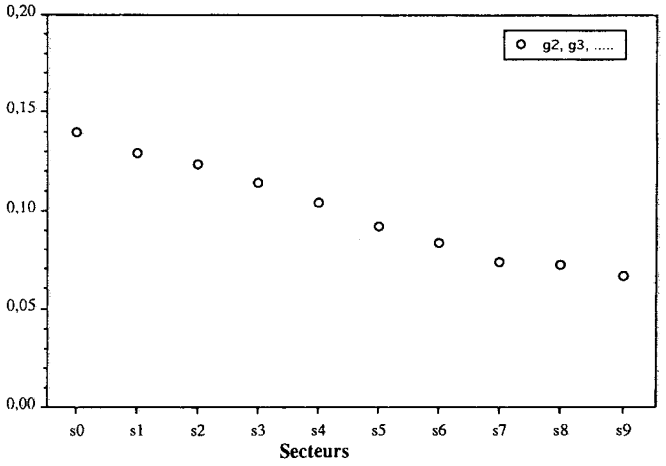
Dans la suite, l'influence des paramètres de l'algorithme est illustrée sur deux fonctions test : la fonction f_1 présentée précédemment et une fonction proposée par De Jong [4] :

$$f_2(x) = \frac{1}{\frac{1}{500} + \sum_{j=1}^{25} \left(\frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6} \right)} \qquad \text{avec} \quad x = (x_1, x_2)$$

$$\text{et} \quad -65,536 \leq x_i \leq 65,536$$



(a)



(b)

Figure 9. – (a) Répartition de la population initiale; (b) répartition de la population dans le domaine de viabilité D_p à la fin d'une génération $(p > 0)$.

où

$$(a_{ij}) = \begin{pmatrix} -32 & -16 & 0 & 16 & 32 & -32 & -16 & 0 & \dots & -32 & -16 & 0 & 16 & 32 \\ -32 & -32 & -32 & -32 & -32 & -16 & -16 & -16 & \dots & 32 & 32 & 32 & 32 & 32 \end{pmatrix}$$

Cette fonction possède un optimum global $f_2(x) = 1$ en $x = (-32; 32)$.

Dans le cadre des résultats présentés ultérieurement, l'influence de chaque paramètre est étudiée en le faisant varier dans son domaine possible et en fixant les autres paramètres aux valeurs suivantes : $N = 200$; $M = 0,01$; $G = 0,2$; $H = 0,4$, $\varepsilon = 10^{-8}$. Dans tous les cas, pour chaque jeu de paramètres, un point représente une moyenne de 50 valeurs.

IV.1. Taille de la population

La taille de la population N doit être suffisamment élevée pour constituer un échantillon représentatif de l'espace de recherche, mais pas trop pour ne pas « alourdir » l'algorithme. Les figures 10 *a*, 10 *b* et 10 *c* représentent respectivement la valeur de l'erreur commise sur l'obtention de $F(x^*)$, le nombre de générations, et le nombre de naissances obtenus en moyenne au cours d'une recherche, et ce, en fonction de N. On vérifie que, plus N est élevée plus la solution est précise. Bien que le nombre de générations nécessaires ait tendance à se stabiliser, le nombre de naissances (nombre de calculs effectifs) a tendance à croître avec N. Ces résultats indiquent que le choix de N nécessite un compromis entre précision des résultats et durée de calcul. Ce choix de N devra donc tenir compte du nombre de variables (L) et de la taille de l'espace de recherche (X).

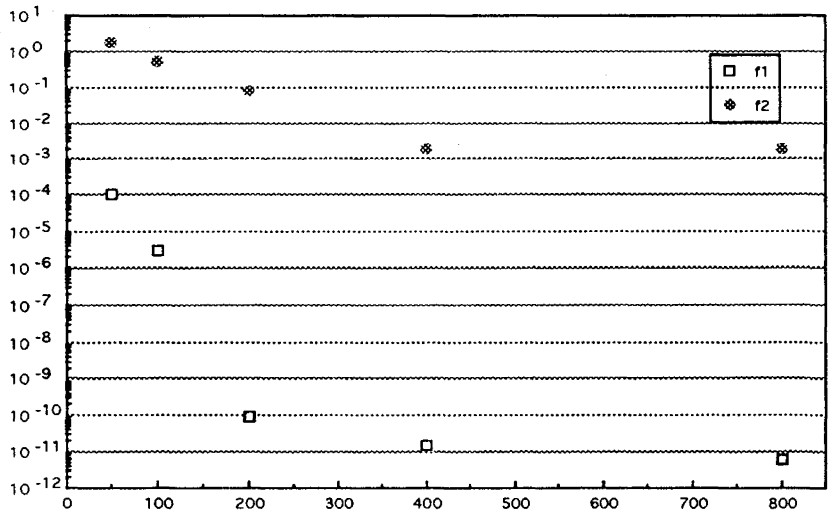


Figure 10 a.

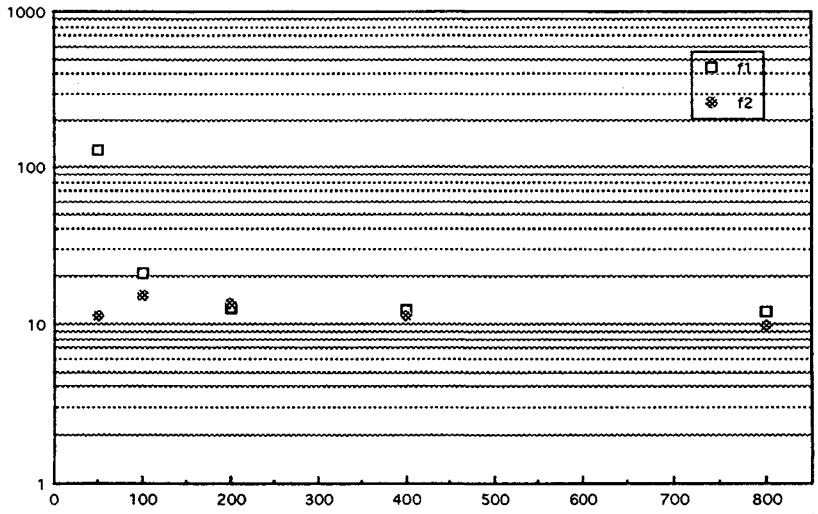


Figure 10 b.

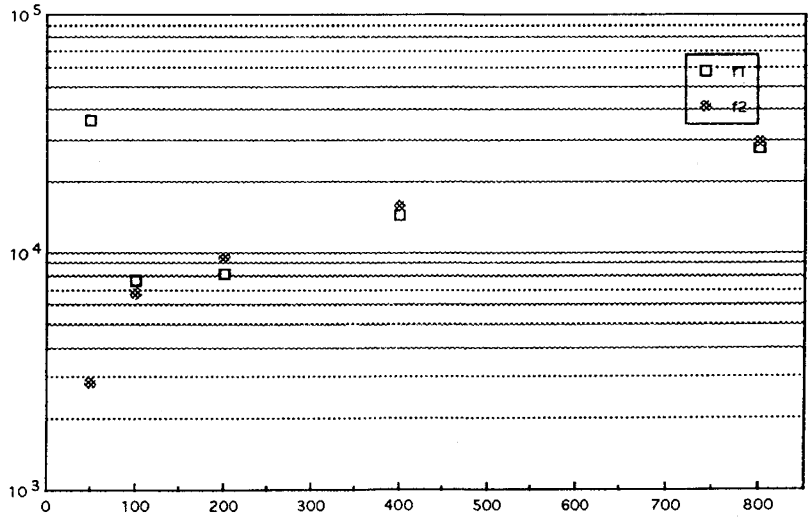


Figure 10 c.

Figure 10. – (a) Erreur commise sur la valeur optimale obtenue en fonction de la taille de la population *N*; (b) nombre de générations nécessaires à la convergence en fonction de la taille de la population *N*; (c) nombre de calculs nécessaires à la convergence en fonction de la taille de la population *N*. (Le nombre de calculs est exprimé en terme de nombre total de naissances).

IV.2. Taux de mutation

Le taux de mutation M permet d'éviter la convergence vers des solutions locales. Il permet une exploration annexe dans tout le domaine de recherche. En pratique, étant donnée la recherche en temps fini nous avons pu observer l'effet de cet opérateur dans les premières générations si le nombre d'extremums est grand. Par contre, son effet est prolongé pour des fonctions relativement planes.

IV.3. Taux de survie

Plus le taux de survie G est grand, plus le nombre de générations nécessaires pour converger l'est aussi (*fig.* 11 *b*), mais contrairement aux apparences le nombre de calculs est quasiment constant (*fig.* 11 *c*). Cette compensation s'explique du fait qu'en augmentant G, on diminue le nombre de naissances. Ainsi, l'apport de chaque génération est moindre et nécessite plus de générations pour finalement aboutir à un nombre total de calculs équivalent. Bien que pour les cas étudiés G n'ait pas une grande influence sur les résultats, il faut qu'il soit suffisamment élevé pour ne pas converger vers un optimum local par perte trop rapide du matériel génétique entraînant des phénomènes de consanguinité.

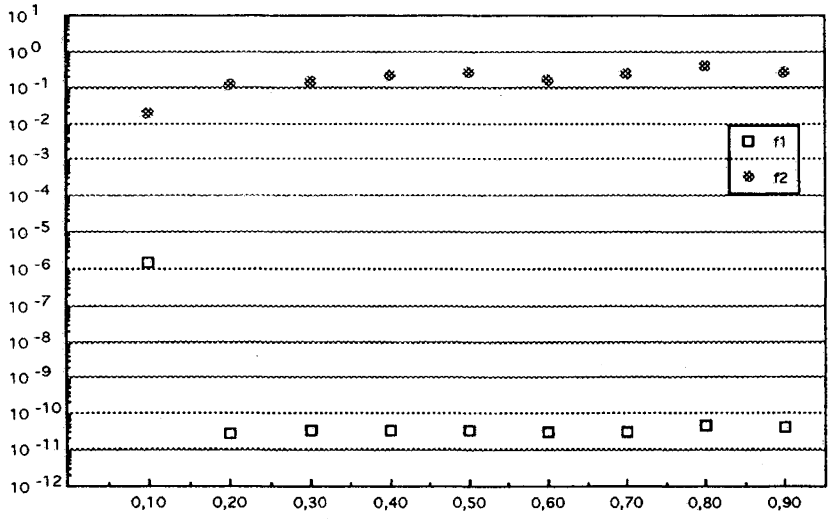


Figure 11 a.

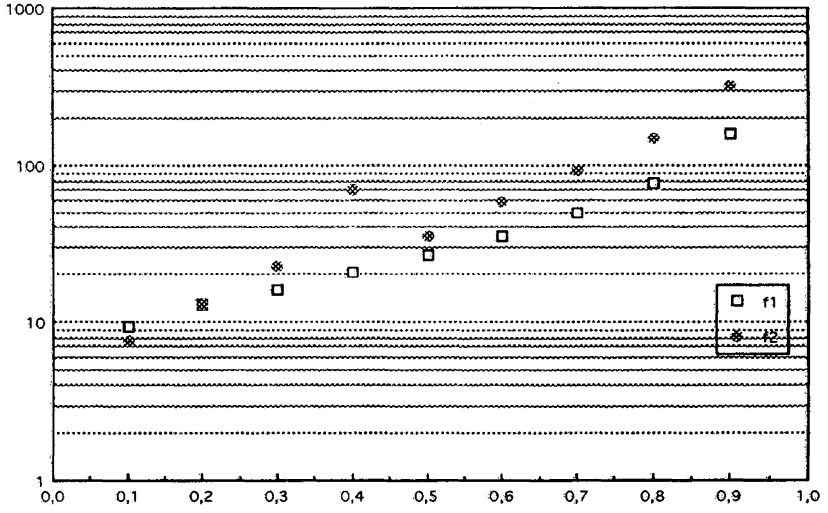


Figure 11 b.

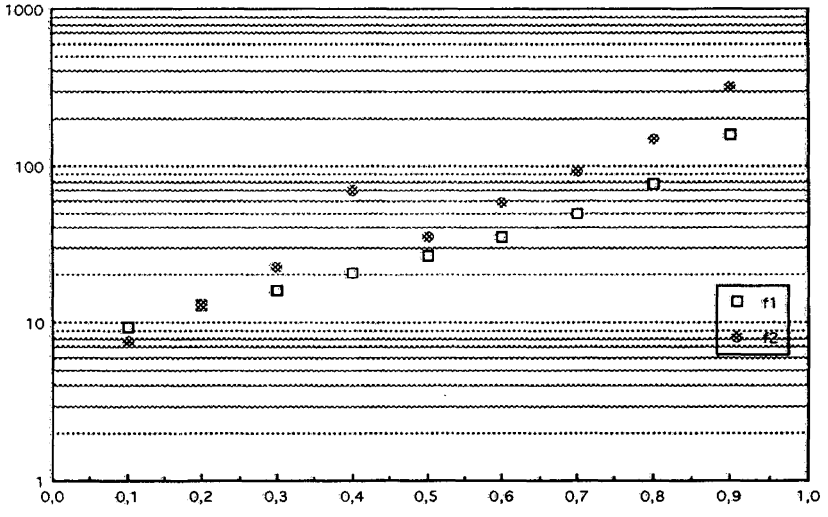


Figure 11 c.

Figure 11. – (a) Erreur commise sur la valeur optimale obtenue en fonction du taux de survie *G*; (b) nombre de générations nécessaires à la convergence en fonction du taux de survie *G*; (c) nombre de calculs nécessaires à la convergence en fonction du taux de survie *G*.

IV.4. Taux d'homozygotie

En absence d'homozygotie ($H = 0$), on s'aperçoit que l'algorithme tel qu'il est conçu ne converge pas. En réalité, il se comporte comme suit :

– le nombre de calculs par génération augmente progressivement,

– au bout de quelques générations, l'algorithme reste bloqué dans la phase de naissance.

Ce blocage traduit une difficulté croissante à générer des enfants viables (la mortalité infantile s'accroît). La probabilité pour qu'un enfant soit viable s'amenuise au cours des générations (rapport du domaine de viabilité sur le domaine de naissance). Ce phénomène s'explique par le fait que le domaine de naissance n'évolue plus alors que le domaine de viabilité, lui, se rétrécit, du fait de la stratégie élitiste. En effet, le domaine de naissance d'une génération est limité par la diversité des combinaisons chromosomiques des parents de cette même génération. Si cette diversité s'amenuise à l'extrême, alors le domaine de naissance devient figé. C'est ce qui se passe quand $H = 0$ et en considérant comme négligeable l'influence des mutants.

A partir d'une population initiale, constituée de N individus hétérozygotes à L gènes et possédant chacun sa spécificité chromosomique, on peut envisager, dans l'absolu, $[N\,(2\,N + 1)]^L$ combinaisons de gènes liés. Dès la première génération, il ne reste plus que $S = NG$ individus ayant une propre spécificité chromosomique. Ainsi, à partir de cette phase, on ne peut

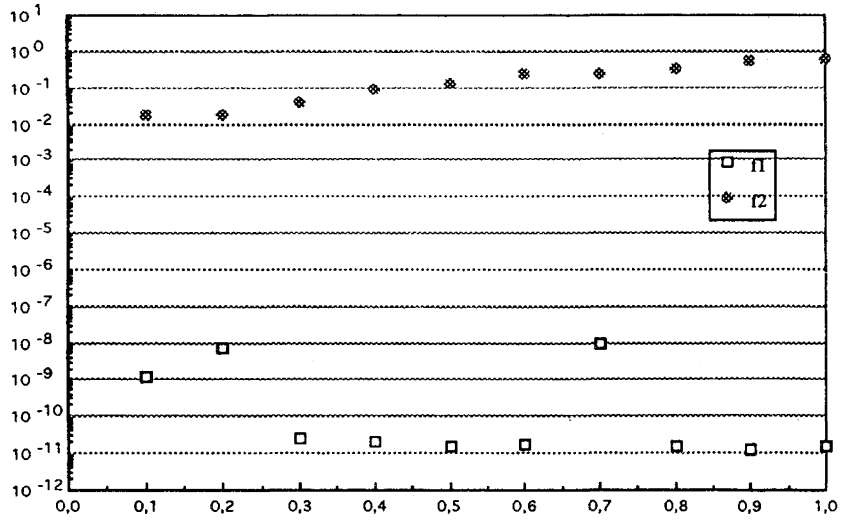


Figure 12 a.

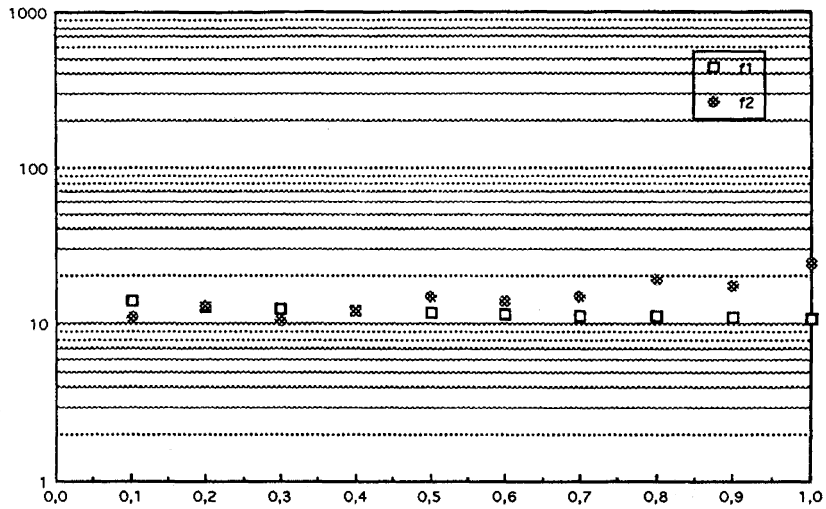


Figure 12 b.

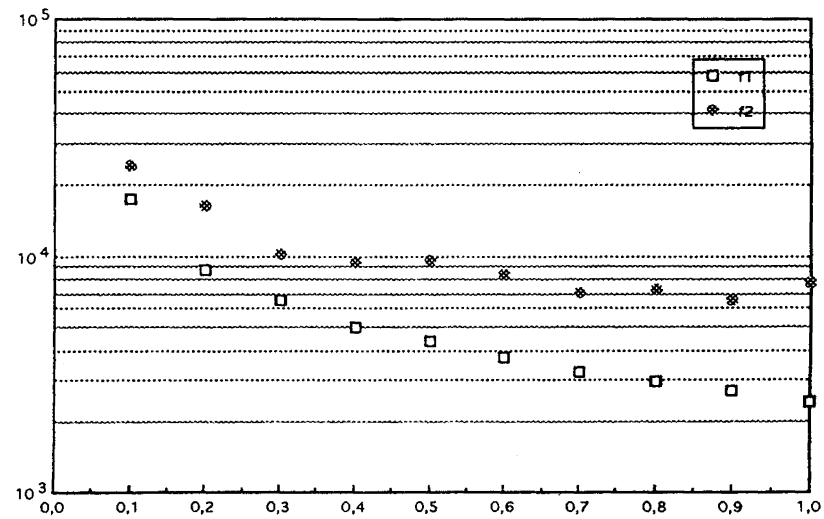


Figure 12 c.

Figure 12. – (a) Erreur commise sur la valeur optimale obtenue en fonction du taux d'homozygotie *H*; (b) nombre de générations nécessaires à la convergence en fonction du taux d'homozygotie *H*; (c) nombre de calculs nécessaires à la convergence en fonction du taux d'homozygotie *H*.

plus envisager que $G^L\,[N\,(2\,GN+1)]^L$ combinaisons de gènes liés, soit environ $1/G^{2\,L}$ fois plus faible que pour la population initiale. Au cours des générations, cette diversité génétique va s'amenuiser encore. En dernière limite, il ne restera plus qu'un seul type d'individu (consanguinité), le test

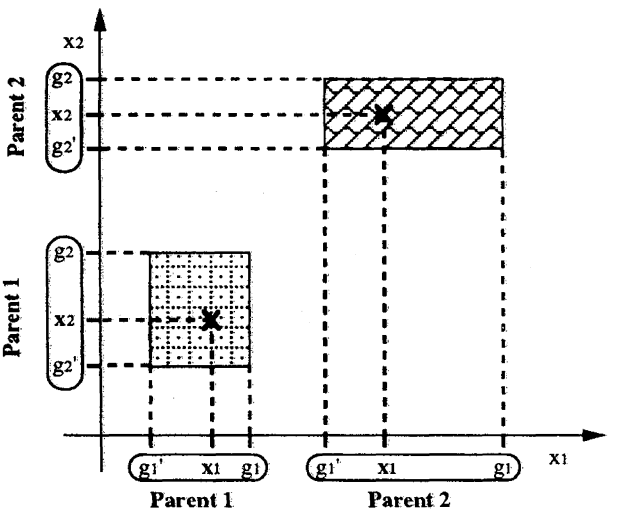


Figure 13 a.

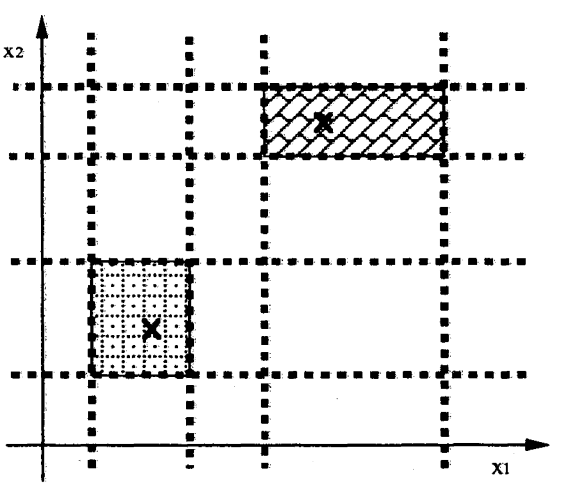


Figure 13 b.

d'arrêt ne sera réalisé à coup sûr uniquement si $\varepsilon > |F(C) - F(C')|$ (F, C et C' représentant respectivement la fonction à optimiser, le premier chromosome et le second chromosome de l'individu).

En négligeant les mutants, pour $H = 0$, l'algorithme est donc restreint au maillage chromosomique (différentes combinaisons de gènes liés) de

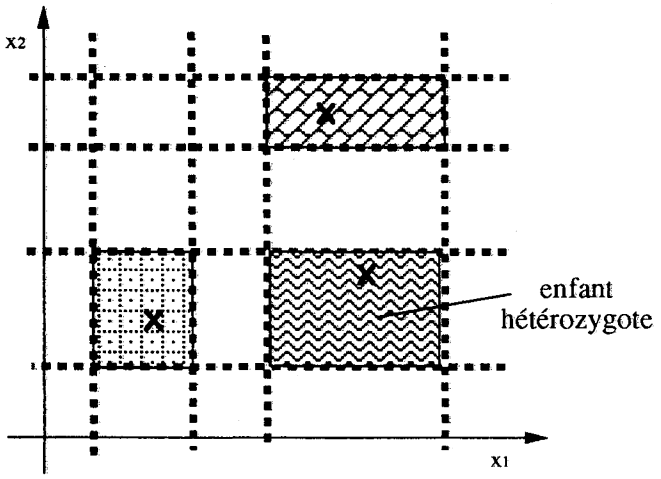


Figure 13 c.

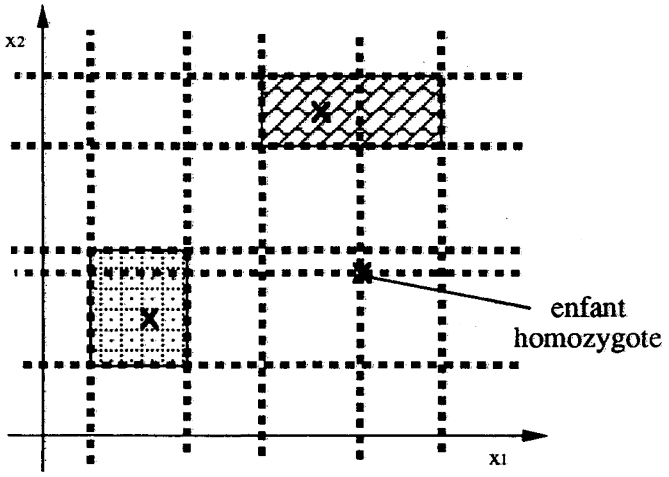


Figure 13 d.

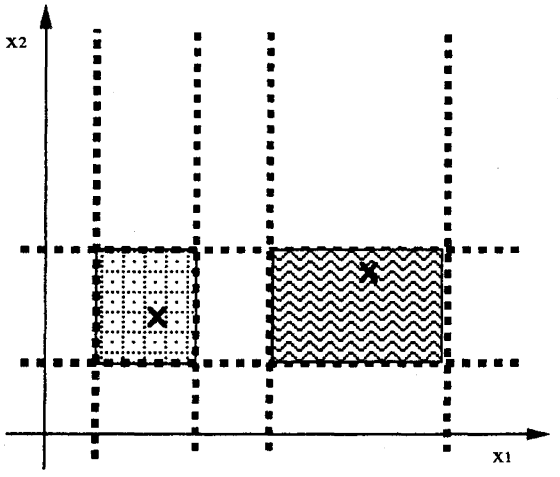


Figure 13 e.

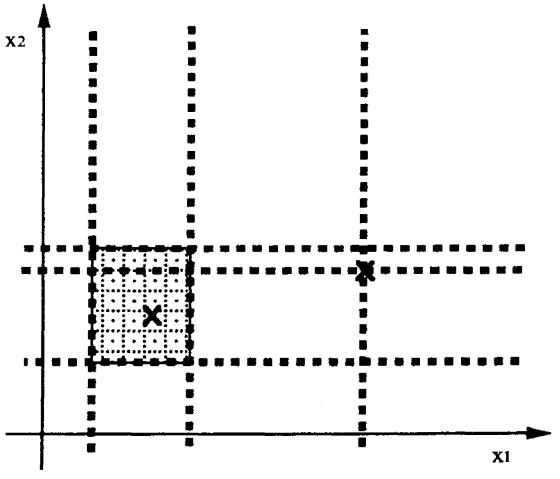


Figure 13 f.

Figure 13. – (a) Exemple de deux parents possédant deux gènes; (b) maillage défini par les deux parents; (c) création d'un enfant hétérozygote à partir des deux parents. Le maillage n'est pas modifié; (d) création d'un enfant homozygote. Nouveau maillage fourni par l'enfant; (e) évolution du maillage après élimination du parent 2, dans le cas où l'enfant est hétérozygote; (f) évolution du maillage après élimination du parent 2, dans le cas où l'enfant est homozygote.

la population initiale. Quel est alors effectivement l'apport de la présence d'homozygotes?

Les individus homozygotes par eux-mêmes n'apportent rien de particulier. Ce qui importe, c'est l'opération d'homozygotie qui en copiant un phénotype dans deux chromosomes crée du matériel génétique nouveau à chaque génération. Ce nouveau matériel, d'une part, compense la perte occasionnée par la phase de mortalité, et, d'autre part, crée un maillage qui se rétrécit au cours des générations (*fig.* 13).

L'influence du taux d'homozygotie est illustrée dans les figures 14. Quand H est faible (*fig.* 14 *a*), dans un premier temps, le domaine se rétrécit suivant une loi exponentielle décroissante, puis l'évolution ralentit pour finalement atteindre une asymptote (perte de la diversité de la population). Pour une valeur de H suffisamment grande, l'évolution est exponentiellement décroissante, mais avec une décroissance plus prononcée quand H se rapproche de 1 (voir *fig.*14 *b* et 14 *c*).

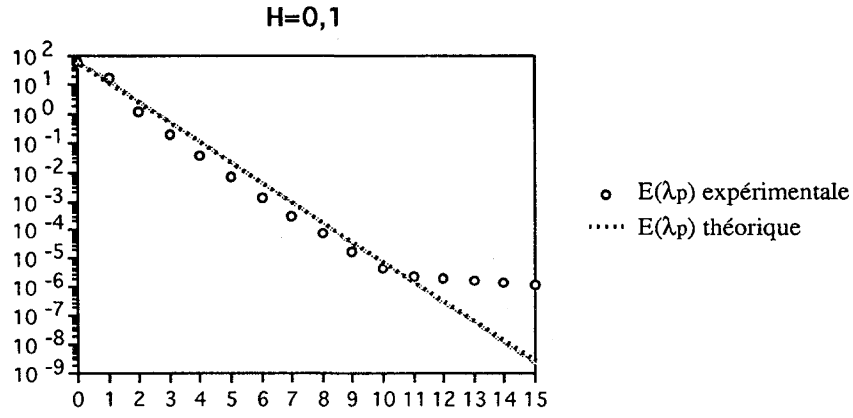


Figure 14 a.

En conclusion, il faudra un taux d'homozygotie suffisamment élevé pour compenser, à chaque génération, l'appauvrissement génétique, mais pas trop élevé, pour ne pas perdre la spécificité des individus diploïdes, concernant leur capacité à effectuer la recherche dans un domaine large. Il est à nouveau indispensable de trouver un compromis entre rapidité de convergence et convergence vers un optimum global et non local.

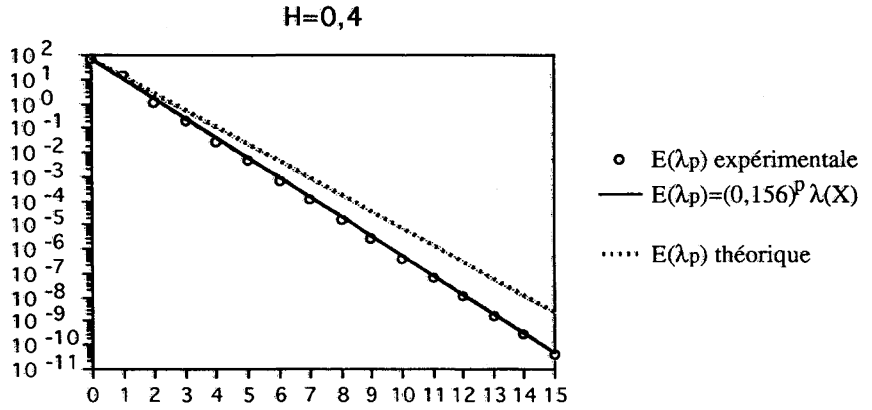


Figure 14 b.

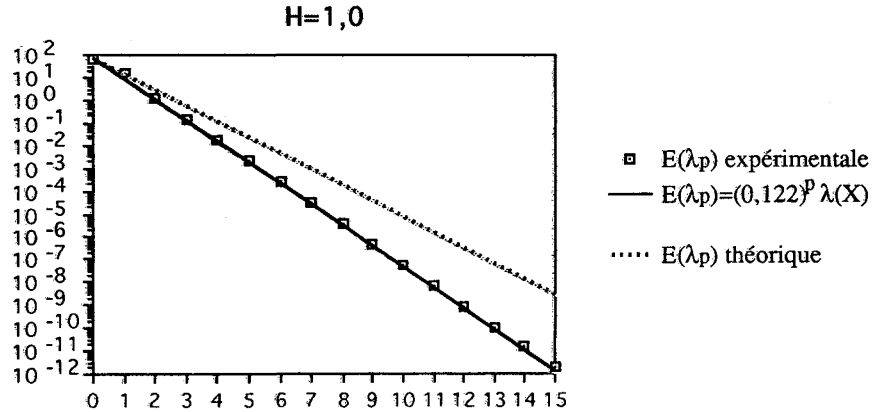


Figure 14 c.

Figure 14. – Évolution du domaine de viabilité au cours des générations. Influence du taux d'homozygotie *H*. Comparaison avec l'évolution théorique : $E(\lambda p) = (0,203)^p \lambda(X)$ **pour** $G = 0,2$.

IV.5. Test d'arrêt

Dans la pratique, le test d'arrêt est le paramètre limitant de l'algorithme. Il doit, d'une part, permettre une recherche suffisamment longue pour s'assurer de la globalité de la solution, et, d'autre part, savoir terminer la recherche pour qu'elle ne dure pas indéfiniment. Sans connaissances *a priori* sur la

fonction, le critère proposé est apparu relativement efficace dans bien des cas étudiés au laboratoire. La difficulté pour proposer un critère d'arrêt robuste, vient du fait que la valeur de l'optimum global est *a priori* inconnue. Bien sûr, d'autres critères sont envisageables.

V. COMPARAISON DES PERFORMANCES DE L'AGDP AVEC UN ALGORITHME GÉNÉTIQUE CLASSIQUE

L'objectif de cette partie consiste à comparer les performances de l'algorithme initial (AGDP) à un algorithme génétique classique.

Dans cette étude, nous avons repris le problème utilisé par Yen *et al.* [5]. Il s'agit de maximiser la fonction :

$$f(x) = \sum_{i=1}^{L} \sin(x_i) \cdot \left(\sin\left(\frac{i \cdot x_i^2}{\pi} \right) \right)^{2m} \qquad \text{pour} \quad x_i \in [0; \pi].$$

$L = 10$ représente le nombre de variables. Plus m est grand, plus la recherche des maximums est difficile; les bassins d'attraction devenant plus étroits. Cette fonction possède un grand nombre de maximums locaux (L !).

La maximisation est effectuée pour $m = 10$ et $m = 100$. L'optimum théorique de f vaut 9,660 pour $m = 10$, et 9,655 pour $m = 100$. Afin d'évaluer les performances d'un algorithme d'optimisation, deux critères sont proposés :

1) *la moyenne des valeurs maximales de* f, sur 10 essais, obtenues au bout de 10^6 évaluations de la fonction.

2) *le nombre moyen d'évaluations de la fonction* f, sur 10 essais, pour obtenir une valeur de f supérieure à 8,5.

Dans la suite, nous noterons $\overline{\Sigma}_1$, le premier critère, et $\overline{\Sigma}_2$, le second critère.

Dans notre cas, le nombre d'évaluations de la fonction correspond au nombre d'individus de la population auquel est ajouté le nombre total de naissances pour chaque génération. Ce nombre d'évaluations est mémorisé dès qu'un des individus possède une valeur de f supérieure à 8,5. L'ensemble des nombres mémorisés sert alors à évaluer $\overline{\Sigma}_2$.

Nous avons utilisé un taux de survie **G** de 90 %, un taux de mutation **M** de 20 %, un taux d'homozygotie **H** de 50 %, et une taille de population **N** = 500.

Les résultats obtenus sont présentés dans les tableaux I et II, par comparaison à ceux obtenus avec un algorithme génétique classique [5]. Bien

TABLEAU I

Valeur du critère $\overline{\Sigma}_1$, correspondant à la moyenne, sur 10 essais, des valeurs maximales obtenues après 10^6 évaluations de la fonction.

Algorithme	$m = 10$	$m = 100$
A.G. conventionnel	9,592	9,069
AGDP	9,523	9,496

TABLEAU II

Valeur du critère $\overline{\Sigma}_2$, correspondant à la moyenne, sur 10 essais, du nombre d'évaluations de la fonction pour obtenir une valeur de f supérieure ou égale à 8,5.

Algorithme	$m = 10$	$m = 100$
A.G. conventionnel	$5{,}2 \cdot 10^3$	$8{,}9 \cdot 10^4$
AGDP	$1{,}2 \cdot 10^4$	$2{,}0 \cdot 10^4$

que pour $m = 10$ l'algorithme diploïde fournisse une valeur $\overline{\Sigma}_1$ légèrement inférieure à celle obtenue par l'algorithme génétique, pour $m = 100$, ce dernier est beaucoup moins performant (tab. I).

En terme de rapidité de convergence (tab. II), pour $m = 10$, notre algorithme nécessite 2 fois plus d'évaluations de la fonction, mais pour $m = 100$ il en nécessite 4 fois moins. De plus, l'écart entre les performances obtenues pour $m = 10$ et $m = 100$ est beaucoup moins important. Une autre série de 10 essais nous a fourni $\overline{\Sigma}_1 = 9,567$ et $\overline{\Sigma}_2 = 1,9 \cdot 10^4$ pour $m = 100$.

Pour vérifier la répétabilité des résultats obtenus avec l'algorithme diploïde, nous avons effectué 100 essais, les moyennes $\overline{\Sigma}_1$ et $\overline{\Sigma}_2$ étant alors calculées sur ces 100 essais. Les valeurs obtenues sont du même ordre de grandeur que celles acquises sur 10 essais, avec $\overline{\Sigma}_1 = 9,481$ et $\overline{\Sigma}_2 = 1,1 \cdot 10^4$ pour $m = 10$, et $\overline{\Sigma}_1 = 9,432$ et $\overline{\Sigma}_2 = 2,0 \cdot 10^4$ pour $m = 100$.

La figure 15 *a* représente la distribution des 100 valeurs finales obtenues pour $m = 10$ et $m = 100$. Dans le cas de la plus faible valeur de m, 40 % des résultats se situent entre 9,5 et 9,6 et 17 % sont proches de l'optimum global (valeurs supérieures à 9,6). Pour $m = 100$, 31 % des résultats se situent entre 9,5 et 9,6 et 15 % sont proches de l'optimum global (valeurs supérieures à 9,6).

La figure 15 *b* illustre la distribution du nombre d'évaluations de la fonction pour atteindre une valeur supérieure à 8,5. Ici, seuls 99 résultats sont utilisés, puisque, dans un seul cas, l'algorithme a fourni une valeur finale inférieure au seuil de 8,5 (*fig.* 15 *a*). Pour $m = 10$, le nombre d'évaluations

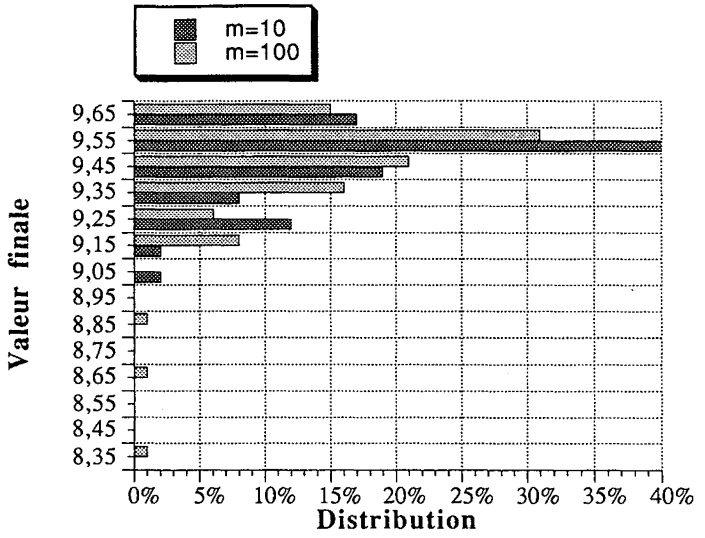


Figure 15 a.

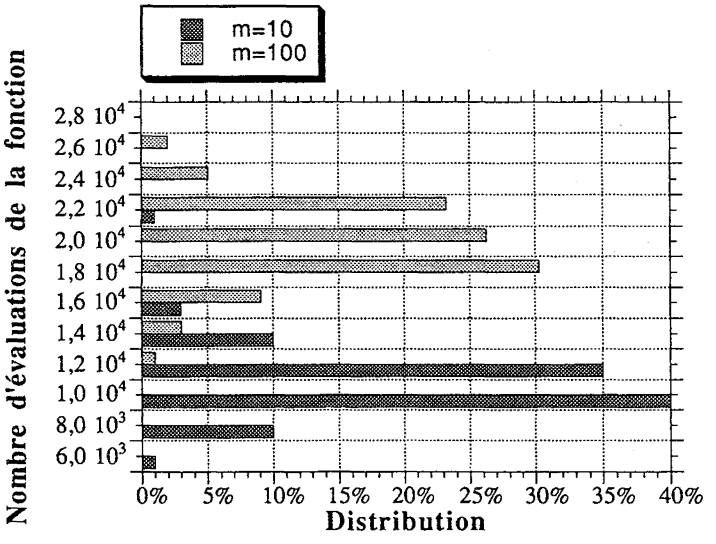


Figure 15 b.

Figure 15. – (a) Histogramme, sur 100 essais, des valeurs maximales obtenues avec l'algorithme diploïde au bout de 10^6 évaluations de la fonction; (b) histogramme, sur 100 essais, du nombre d'évaluations de la fonction pour obtenir une valeur de f supérieure à 8,5.

de la fonction est compris entre 5.10^3 et $2{,}3.10^4$. Le nombre d'évaluations pour $m = 100$ est, lui, compris entre $1{,}1.10^4$ et $2{,}7.10^4$. Dans tous les cas, que ce soit pour $m = 10$ ou $m = 100$, il faut moins de 10^5 évaluations de la fonction pour approcher, à 10^{-3} près, la valeur finale obtenue par l'algorithme. Il faut en moyenne 3.10^4 et 4.10^4 évaluations, respectivement pour $m = 10$ et $m = 100$, pour approcher la valeur finale avec cette précision.

La figure 16 illustre l'évolution moyenne du nombre d'évaluations de la fonction au cours des 400 premières générations. L'algorithme diploïde nécessite plus d'évaluations de la fonction durant les 100 premières générations que pour les générations suivantes. Au bout de 100 générations pour $m = 10$ (150 pour $m = 100$) le nombre d'évaluations de la fonction nécessaire pour créer des enfants viables à chaque génération est quasiment constant, et vaut environ 60. Cette valeur correspond au nombre d'individus à créer à chaque génération $((1 - G)^* N = 50)$, augmenté du nombre de mutants générés $((1 - G)^* N^* M = 10)$ qui, eux, ne sont plus viables à ce niveau.

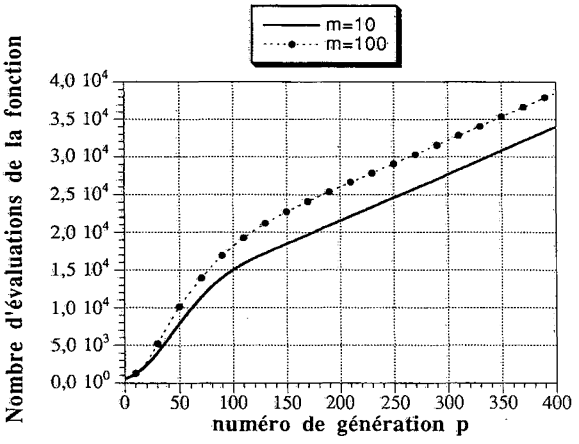


Figure 16. – Évolution du nombre moyen d'évaluations de la fonction au cours des générations.

En résumé, l'AGDP fournit une meilleure approche de la solution optimale qu'un algorithme génétique classique lorsque le problème est complexe ($m = 100$). En terme de vitesse de convergence, même s'il est plus lent à s'approcher de la zone solution (critère $\overline{\Sigma}_2$), il fournit plus rapidement une solution dans cette zone. Cependant, l'algorithme ne nous paraît pas

suffisamment performant en terme de précision sur les solutions obtenues.

Une analyse plus approfondie du comportement de notre algorithme nous a conduit à mettre en évidence l'effet principal de la dominance partagée. En effet, la présence d'homozygotes dans le cas d'une recombinaison trop importante entraîne un risque de perte d'information, et, par conséquent, une focalisation trop rapide sur des solutions locales. Ainsi, étant donné la contrainte de viabilité sur les enfants créés, à partir de génotypes différents, on aboutit à des phénotypes peu variés. En conclusion, dominance partagée et homozygotie ne créent pas suffisamment de mélanges (brassage de l'information génétique). Pour éviter ces problèmes, il est indispensable d'associer plusieurs propriétés :

– favoriser la diversité des combinaisons de gènes (génotypes différents mais avec des phénotypes tout aussi différents),

– permettre un bon brassage de l'information,

– conserver une bonne proportion de la population sous forme d'individus homozygotes. Ces derniers permettent d'améliorer l'information génétique, et, donc, de mieux converger vers la solution.

Un moyen simple permettant de favoriser le brassage génétique consiste à utiliser une proportion d'individus à dominance vraie (*i.e.* choisie dans $\{0; 1\}^L$), et une proportion d'individus à dominance partagée (*i.e.* choisie dans $[0; 1]^L$) qui deviennent tous homozygotes.

VI. COMPARAISON DES PERFORMANCES DE L'AGDV AVEC UN ALGORITHME GÉNÉTIQUE HYBRIDE

L'objectif de cette partie est de montrer les performances obtenues en utilisant des individus diploïdes à dominance vraie. En prenant pour base le problème précédent, et les critères $\overline{\Sigma}_1$ et $\overline{\Sigma}_2$, nous comparons les performances de l'AGDV avec celles obtenues avec un AG hybride, puis nous analysons et détaillons les performances de l'AGDV.

VI.1. AG hybride

Yen *et al.* [5] proposent un algorithme couplant un AG avec une méthode de simplex concurrent. Comme pour un AG, cet algorithme est basé sur l'évolution d'une population au cours de générations successives. L'évolution d'une génération à une autre consiste à modifier la population P comportant

N individus, en la population P' de même taille. Trois mécanismes sont utilisés :

1) recopie des L meilleurs individus de P dans P', où L représente le nombre de variables du problème,

2) création de S individus par simplex concurrent, en utilisant les $L+S$ meilleurs individus de P, les L meilleurs de P étant utilisés pour calculer leur centroïde, et les autres étant réfléchis à travers ce centroïde pour créer les S nouveaux individus de P'.

3) création de N-L-S individus, permettant de compléter la nouvelle population P', en choisissant des couples de parents parmi les N individus de P et en appliquant une procédure classique de croisement.

VI.2. AGDV

La différence avec l'AGDP réside dans la définition des opérateurs secondaires utilisés lors de la phase de naissance.

Que ce soit pour le croisement ou la mutation, le vecteur des dominances est tiré aléatoirement, non plus dans $[0;\ 1]^L$, mais dans $\{0;\ 1\}^L$.

En ce qui concerne l'opération d'homozygotie, pour un enfant choisi comme homozygote, on génère un nouveau vecteur de dominance dans $[0;\ 1]^L$, et le nouveau phénotype est calculé. Finalement, on recopie le phénotype dans les deux chromosomes de l'enfant.

Dans tous les résultats présentés dans la suite, nous avons fixé $G = 90\,\%$, $M = 20\,\%$, et $H = 50\,\%$. Nous avons expérimenté l'algorithme avec trois tailles de population différentes : $N = 250$, $N = 500$, et $N = 1000$. Dans chaque cas, l'algorithme a été testé sur 100 essais, et chacun d'entre eux a été arrêté au bout de 10^6 évaluations de la fonction à optimiser.

VI.3. Comparaison des performances

Pour $N = 250$, les performances de l'AGDV sont relativement comparables à celles obtenues avec l'AG hybride proposé par Yen *et al.* [5]. Ces auteurs ont calculé les critères à partir de 10 valeurs, sans, toutefois, préciser le nombre de calculs nécessaires à l'obtention de la solution finale, hormis le nombre maximum d'évaluations de la fonction fixé à 10^6. Quant à nous, pour être plus représentatifs, nos calculs sont basés sur 100 valeurs.

Pour $m = 10$, l'AGDV, avec une taille de population $N = 250$, fournit une solution plus précise que l'AG hybride, $\overline{\Sigma}_1 = 9,638$ par rapport à 9,584 (tab. III), mais se montre plus lent dans l'approche de la zone optimale, pour

TABLEAU III
Valeur du critère $\overline{\Sigma}_1$, correspondant à la moyenne des valeurs maximales obtenues.

Algorithme	$m = 10$	$m = 100$
A.G. hybride de Yen *et al.*	9,584	9,590
AGDV		
$N = 250$	9,638	9,571
$N = 500$	9,654	9,630
$N = 1000$	9,659	9,650

TABLEAU IV
Valeur du critère $\overline{\Sigma}_2$, correspondant à la moyenne du nombre d'évaluations de la fonction pour obtenir une valeur de f supérieure ou égale à 8,5.

Algorithme	$m = 10$	$m = 100$
A.G. hybride de Yen *et al.*	$2{,}9 \cdot 10^3$	$8{,}9 \cdot 10^3$
AGDV		
$N = 250$	$5{,}0 \cdot 10^3$	$7{,}9 \cdot 10^3$
$N = 500$	$9{,}3 \cdot 10^3$	$1{,}4 \cdot 10^4$
$N = 1000$	$1{,}7 \cdot 10^4$	$2{,}7 \cdot 10^4$

ce qui concerne le critère de convergence défini par Yen *et al.* : $\overline{\Sigma}_2 = 5000$ par rapport à 2900 (tab. IV).

Par contre, pour $m = 100$, les résultats sont inversés : l'AGDV est plus rapide ($\overline{\Sigma}_2 = 7900$ devant 8900), mais fournit une solution à peine moins précise ($\overline{\Sigma}_1 = 9,571$ devant 9,590).

Lorsque la taille de la population est plus grande ($N > 250$), notre algorithme devient plus lent mais fournit des solutions nettement plus précises.

VI.4. Analyse des performances de l'algorithme à dominance vraie

Les tableaux V et VI indiquent, respectivement pour $m = 10$ et $m = 100$, les fréquences cumulées des solutions (sur 100 essais) pour différentes précisions vis-à-vis des solutions globales ($f = 9,660$ pour $m = 10$, et $f = 9,655$ pour $m = 100$). D'une manière générale, la précision obtenue croît avec la taille de la population N.

Les vitesses de convergence sont illustrées dans les tableaux VII ($m = 10$) et VIII ($m = 100$). Pour chaque taille de la population N, sont indiqués :

TABLEAU V

Fréquences cumulées, sur 100 essais, en fonction des erreurs relatives obtenues, par rapport à la solution globale ($f = 9,660$), et de la taille de la population. Cas où $m = 10$.

Erreur relative (%) Valeur de f	$< 0,01$ $> 9,659$	$< 0,06$ $> 9,654$	$< 0,2$ $> 9,641$	$< 0,5$ $> 9,612$	$< 1,2$ $> 9,544$
Taille de la population					
$N = 250$	16 %	42 %	58 %	89 %	100 %
$N = 500$	48 %	82 %	90 %	100 %	–
$N = 1000$	70 %	98 %	100 %	–	–

TABLEAU VI

Fréquences cumulées, sur 100 essais, en fonction des erreurs relatives obtenues, par rapport à la solution globale ($f = 9,655$), et de la taille de la population. Cas où $m = 100$.

Erreur relative (%) Valeur de f	$< 0,01$ $> 9,654$	$< 0,2$ $> 9,635$	$< 0,5$ $> 9,606$	$< 1,0$ $> 9,558$	$< 1,3$ $> 9,529$	$< 2,4$ $> 9,423$
Taille de la population						
$N = 250$	1 %	9 %	25 %	63 %	81 %	100 %
$N = 500$	8 %	56 %	85 %	98 %	100 %	–
$N = 1000$	45 %	97 %	99 %	100 %	–	–

le minimum, la moyenne, et le maximum du nombre d'évaluations de la fonction pour obtenir $\varepsilon = 10^{-4}$. ε correspond à l'écart entre le meilleur et le plus mauvais individu à l'issue d'une génération, que nous noterons p_ε. Dans tous les cas, pour $\varepsilon = 10^{-4}$, l'écart des valeurs, entre la solution obtenue et le meilleur individu de la génération p_ε, est inférieur à 10^{-3} (soit 10^{-4} %).

Pour $m = 10$ et $N = 250$, l'erreur relative est inférieure à 1,2 % (tab. V) pour un maximum de $3,4.10^4$ évaluations de la fonction (tab. VII). En augmentant la taille de la population ($N = 1000$), on obtient une erreur inférieure à 0,2 % (tab. V) pour $9,4.10^4$ calculs (tab. VII).

Dans le cas le plus complexe ($m = 100$), pour $N = 250$, l'erreur est inférieure à 2,4 % (tab. VI) pour $3,4.10^4$ calculs (tab. VIII), et pour $N = 1000$, elle est inférieure à 1 % (tab. VI) pour 10^5 calculs (tab. VIII).

Par rapport à l'AGDP, la précision finale est grandement accrue. En effet, pour une taille de population équivalente ($N = 500$) et $m = 100$, dans 85 % des cas testés avec une dominance vraie, l'erreur relative est inférieure à 0,5 % (*i.e.* $f > 9,6$), comparativement à l'AGDP qui fournit uniquement 15 % des solutions (*fig.* 15 *a*) pour une même précision.

TABLEAU VII

Nombre d'évaluations de la fonction à optimiser, en fonction de la taille de la population, pour obtenir un écart inférieur à 10^{-4} entre les valeurs extrêmes de la population. Cas où $m = 10$.

Nombre d'évalution de la fonction pour $\varepsilon = 10^{-4}$	Minimum	Moyenne	Maximum
Taille de la population			
$N = 250$	$1,7 . 10^4$	$2,5 . 10^4$	$3,4 . 10^4$
$N = 500$	$3,6 . 10^4$	$4,6 . 10^4$	$7,4 . 10^4$
$N = 1000$	$7,3 . 10^4$	$8,5 . 10^4$	$9,4 . 10^4$

TABLEAU VIII

Nombre d'évaluations de la fonction à optimiser, en fonction de la taille de la population, pour obtenir un écart inférieur à 10^{-4} entre les valeurs extrêmes de la population. Cas où $m = 100$.

Nombre d'évaluations de la fonction pour $\varepsilon = 10^{-4}$	Minimum	Moyenne	Maximum
Taille de la population			
$N = 250$	$2,2 . 10^4$	$2,8 . 10^4$	$3,4 . 10^4$
$N = 500$	$3,8 . 10^4$	$5,6 . 10^4$	$7,4 . 10^4$
$N = 1000$	$8,5 . 10^4$	$9,5 . 10^4$	$1,0 . 10^5$

En terme de vitesse de convergence, les performances sont relativement équivalentes. Nous remarquons, toutefois, que l'AGDV converge plus rapidement vers la zone optimale ($\overline{\Sigma}_2 = 1,4 . 10^4$ par raport à $\overline{\Sigma}_2 = 2.10^4$ avec une dominance partagée), mais se montre plus lent pour affiner la solution. Il lui faut, en moyenne, $5,6 . 10^4$ évaluations de la fonction contrairement à l'autre version (AGDP) qui nécessite, en moyenne, uniquement $4 . 10^4$ évaluations.

VII. CONCLUSION

Un nouvel outil d'optimisation globale a été proposé dans cet article. La convergence théorique vers un optimum global est démontrée. La mise en œuvre pratique engendre cependant un certain nombre de limitations.

Un tel algorithme nécessite un réglage de ses paramètres, qui sont, comparé à ses potentialités, en nombre restreint (4 paramètres et un test d'arrêt). Dans ce sens, nous avons décrit l'influence des différents paramètres.

Le test d'arrêt proposé, tout à fait arbitraire, pourrait dans certains cas provoquer la convergence vers un optimum local. Des études sont à réaliser dans ce sens, mais il faut conserver à l'esprit le but pratique d'un tel outil.

Actuellement, l'ensemble des dominances est tiré aléatoirement. Une gestion des dominances devrait pouvoir accélérer la convergence. Une étude ultérieure serait à mener.

D'autres améliorations sont à apporter concernant la répartition de la population initiale. Suivant le problème traité on pourrait choisir de favoriser telles ou telles zones de l'espace de recherche.

Les tests comparatifs mis en œuvre ont montré que l'algorithme dans sa version initiale (AGDP) était au moins aussi performant qu'un AG classique. L'analyse des résultats obtenus nous a conduit à proposer une orientation pour l'amélioration des performances : utiliser une dominance vraie. Les tests effectués pour valider ce choix ont montré que l'AGDV était beaucoup plus performant qu'un AG classique, et qu'il offrait des performances comparables à celles obtenues avec un AG hybride (en terme d'approche de la zone optimale). Toutefois, si l'on considère la précision finale obtenue pour une population suffisamment grande ($N > 250$), l'algorithme diploïde à dominance vraie est le plus performant.

Une autre solution pour améliorer les performances de l'algorithme consisterait à le coupler avec un algorithme rapide de recherche d'optimums locaux. Dans ce cas, le rôle de l'algorithme diploïde consisterait à explorer l'ensemble du domaine pour détecter les zones optimales, l'autre algorithme étant chargé d'extraire les solutions précises dans les zones détectées.

Il est indéniable que l'algorithme proposé est un puissant outil d'optimisation de par sa stratégie de recherche globale. Toutefois, à cause d'une certaine lourdeur (nombre de calculs pour obtenir une précision suffisante), il faut objectivement le réserver à des problèmes que d'autres algorithmes (par résolution analytique, algorithmes de type gradient, ...) moins coûteux en temps de calcul ne sont pas capables de résoudre, par exemple des problèmes pour lesquels l'accès à des connaissances spécifiques de la fonction objectif n'est pas aisé, ou présentant un grand nombre d'optimums locaux, ou pour traiter des fonctions peu régulières, discontinues, stochastiques, ou résultant de systèmes d'équations différentielles. Cet ensemble de caractéristiques est le lot courant de bien des problèmes rencontrés en génie des procédés.

BIBLIOGRAPHIE

1. J. H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, 1975.
2. F. Hoffmeister et T. Bäck, *Genetic Algorithms and Evolution Strategies: Similarities and differences,* Dortmund University, Bericht Nr. 365, Nov. 1990.

3. F. Bicking, C. Fonteix, J. P. Coriou, I. Marc, *Global Optimization by Artificial Life: A new technique using genetic population evolution,* RAIRO Recherche Opérationnelle, 1994, *28*, n° 1, p. 23-26.
4. D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning,* Addison-Wesley, Reading, Massachusetts, 1987.
5. J. Yen, D. Randolph, B. Lee, J. C. Liao, *A Hybrid Approach to Modeling Metabolic System Using Genetic Algorithm and Simplex Method,* IEEE International Conference on Systems, Man and Cybernetics, Vancouver, Canada, Oct. 1995, *2*, p. 1205-1210.