

UNE HEURISTIQUE D'OPTIMISATION GLOBALE BASÉE SUR LA Ψ -TRANSFORMATION

ALEXANDRE DOLGUI¹ ET VALERY SYSOEV²

Communiqué par Michel Minoux

Abstract. In this paper, we study a heuristic algorithm for global optimization, which is based on the Ψ -transformation. We illustrate its behavior first, on a set of continuous non-convex objective functions – we search the global optimum of each function. Then, we give an example from combinatorial optimization. It concerns the optimization of scheduling rules parameters of a manufacturing system. Computational results are presented, they look encouraging.

Résumé. Dans cet article nous étudions une heuristique d'optimisation globale basée sur la Ψ -transformation. Nous illustrons son comportement sur deux types d'exemples. D'abord, nous utilisons un ensemble de fonctions objectif continues non convexes. Nous recherchons l'optimum global de chaque fonction. Ensuite, nous donnons un exemple d'optimisation combinatoire. Cet exemple est lié à l'optimisation paramétrique des règles d'ordonnancement dans un atelier de production manufacturière. Les résultats des tests sont présentés, ils sont encourageants.

Mots Clés. Optimisation globale, métaheuristiques, Ψ -transformation, méthode Monte-Carlo.

Classification Mathématique. 90B35.

Reçu en septembre 1999.

¹ École des Mines de Saint Étienne, Centre SIMMO, 158 cours Fauriel, 42023 Saint-Étienne Cedex 2, France ; e-mail : dolgui@emse.fr

² Chaire de Modélisation Mathématique des Systèmes Technologiques, Académie Technologique de Voronezh, 19 avenue Révolution, 394017 Voronezh, Russie.

1. INTRODUCTION

Dans beaucoup d'applications des méthodes d'optimisation nous rencontrons le problème dit d'"optimisation globale". En effet, la fonction objectif peut avoir beaucoup d'optima locaux. Il faut donc trouver le meilleur, c'est-à-dire l'optimum global. Ce problème apparaît aussi bien dans le domaine de l'optimisation des fonctions continues non convexes qu'en optimisation combinatoire.

Dans la littérature, il existe de nombreux travaux présentant l'état de l'art et faisant différentes classifications des méthodes proposées.

En suivant Törn [13], nous classifions ces méthodes dans les catégories suivantes :

- 1) méthodes garantissant le résultat optimal (procédures par séparation et évaluation, par exemple) ;
- 2) méthodes de recherche globale (recherche aléatoire de type Monte-Carlo) ;
- 3) méthodes adaptatives (recuit simulé, algorithmes génétiques, recherche Tabou) ;
- 4) couplage de la recherche globale avec la recherche locale ("multistart", méthodes de regroupement, etc.) ;
- 5) méthodes Bayésiennes [9].

D'autres classifications existent, par exemple, celle proposée par Rinnoy Kan *et al.* [8].

Notons ici que ces classifications ne doivent pas être prises au pied de la lettre et que certaines méthodes peuvent, suivant les points de vue, être rattachées à des catégories différentes.

De manière générale, il est de tradition de distinguer les méthodes "classiques" des nouvelles méthodes heuristiques (méta-heuristiques) [10]. Les origines des premières se trouvent dans l'optimisation globale des fonctions non convexes sans contraintes. Les dernières ont été initialement proposées pour des problèmes combinatoires, parmi elles, les plus connues sont le recuit simulé, les algorithmes génétiques et la recherche Tabou.

Beaucoup de travaux en cours sont consacrés au mixage et à l'adaptation de ces deux catégories de méthodes à différents domaines d'application, comme le montre l'état de l'art récent fait par Berthieu et Siarry [1].

Dans cet article, nous étudions une nouvelle heuristique qui, à notre avis, peut ouvrir de nouvelles voies de recherche. Dans cette heuristique, la recherche globale est faite en utilisant l'idée de la Ψ -transformation proposée par Chichinadze [2], cette dernière étant réalisée à travers des simulations Monte-Carlo de la fonction objectif. Deux algorithmes complémentaires permettent d'améliorer les performances de l'algorithme global. Le premier a comme rôle de réduire l'espace de recherche au fur et à mesure de l'avancement de l'algorithme. Le deuxième réalise, à la fin de chaque itération, une descente simple pour trouver l'optimum local correspondant.

Nous présentons d'abord, l'idée générale de la méthode et l'algorithme d'optimisation utilisé, ensuite nous donnons les résultats obtenus à partir des deux types

de tests suivants : le premier est réalisé sur un ensemble des fonctions théoriques complexes avec un comportement et un optimum global connu. Le deuxième type de test concerne un problème d'optimisation combinatoire difficile avec un nombre important des variables de décision. Pour ce test nous comparons les résultats de la meilleur heuristique connue à ce jour avec ceux obtenus par l'heuristique proposée.

Étant donné que la méthode repose sur l'estimation de la mesure de Lebesgue des ensembles des solutions possibles, en annexe nous rappelons quelques définitions et propriétés théoriques connues afin éventuellement d'aider à cerner la classe des problèmes susceptibles d'être résolus avec l'approche proposée.

La méthode est limitée à des nombres de variables relativement petits. Nous pensons que son utilisation pour un nombre de variables plus important serait possible par exemple en la combinant avec d'autres techniques.

2. MÉTHODE DE LA Ψ -TRANSFORMATION

Nous étudions le problème d'optimisation suivant :

$$F(\mathbf{X}) = F(x_1, x_2, \dots, x_n) \xrightarrow{X \in D} \min, \quad (1)$$

$$D : \{\mathbf{X} = (x_1, x_2, \dots, x_n), g_l(\mathbf{X}) \leq 0, l = 1, 2, \dots, b_i \leq x_i \leq a_i, i = 1, \dots, n\}.$$

Aucune condition particulière n'est posée sur la forme de la fonction objectif (1) ; il faut simplement que, pour chaque vecteur admissible \mathbf{X} des variables de décision, il soit possible de donner la valeur correspondante du critère $F(\mathbf{X})$ et que $F(\mathbf{X})$ soit une fonction mesurable au sens de Lebesgue (*cf.* annexe 1).

L'idée de la Ψ -transformation [2] consiste à substituer à la fonction $F(\mathbf{X})$ ayant plusieurs optima locaux, une fonction $\mu = \Psi(\xi)$ qui est une fonction monotone avec un seul optimum. Pour cela, nous utilisons ξ qui est un scalaire défini dans l'intervalle des valeurs atteintes par la fonction $F(\mathbf{X})$. Une transformation faisable de ce type est présentée dans les figures 1 et 2 [2].

Dans [12], la fonction $\mu = \Psi(\xi)$ prend les valeurs des cardinaux des ensembles E_j obtenus en partitionnant la fonction $F(\mathbf{X})$ grâce à la partition de Lebesgue (L -partition), *i.e.* E_j est l'ensemble de solutions \mathbf{X} pour lesquelles :

$$\begin{aligned} F(\mathbf{X}) &\leq \xi_j, \\ \xi_j \in L &= [\inf |F(\mathbf{X})| = \xi_0 < \xi_1 < \dots < \xi_M = \sup |F(\mathbf{X})|], \\ \xi_{j+1} - \xi_j &= \text{const}, \quad j = 0, 1, \dots, M - 1. \end{aligned}$$

E_j est l'ensemble de solutions \mathbf{X} pour lesquelles la fonction objectif $F(\mathbf{X})$ est inférieure ou égale au seuil ξ_j . Les valeurs des seuils $\xi_j, j = 0, 1, \dots, M$ sont obtenues par la partition de l'intervalle de toutes les valeurs possibles de la fonction objectif en M sous-intervalles de même longueur.

En d'autres termes [3, 4], si nous définissons un ensemble E_j de la manière suivante :

$$E_j = \{\mathbf{X} | F(\mathbf{X}) \leq \xi_j, \mathbf{X} = (x_1, x_2, \dots, x_n), \mathbf{X} \in D\},$$

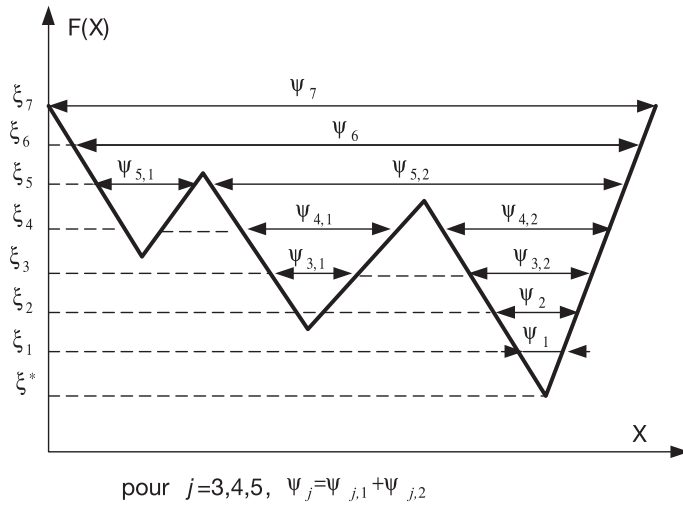


FIGURE 1. Un exemple de fonction avec plusieurs optima locaux.

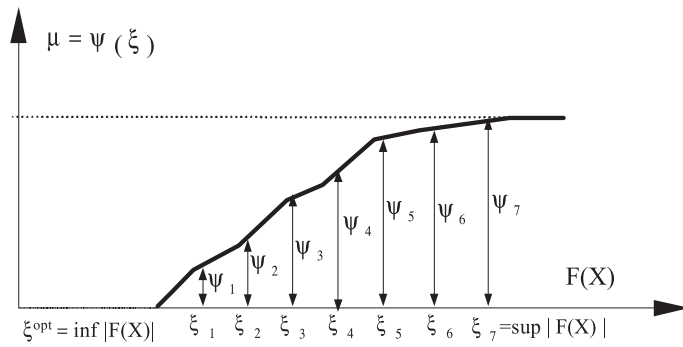


FIGURE 2. Résultat d'une Psi-transformation.

avec la fonction caractéristique

$$\Theta(\mathbf{X}, \xi_j) = \begin{cases} 1, & \text{si } \mathbf{X} \text{ appartient à } E_j \\ 0, & \text{sinon,} \end{cases}$$

alors $\mu_j = \Psi(\xi_j)$ est définie comme étant la mesure de l'ensemble E_j dans l'espace D :

$$\Psi(\xi_j) = \int_D \Theta(\mathbf{X}, \xi_j) d\mathbf{X}$$

est une fonction croissante; sa valeur est égale à zéro aux points suivants :

$$\xi \leq \xi_{\text{opt}}, \quad \xi_{\text{opt}} = \min_{\mathbf{X} \in D} F(\mathbf{X}).$$

Le problème de la minimisation de $F(\mathbf{X})$ peut être donc transformé en recherche de

$$\xi_{\text{opt}} = \max\{\xi | \Psi(\xi) = 0\}.$$

Ces considérations théoriques permettent de proposer une approche pratique qui consiste à trouver, pour chaque valeur ξ_j , par la méthode Monte-Carlo, la valeur $\tilde{\mu}_j = \tilde{\Psi}(\xi_j)$ qui est une estimation de $\Psi(\xi_j)$. Une approximation statistique des points obtenus est ensuite faite pour avoir des courbes analytiques de $\tilde{\Psi}(\xi)$. Une prévision de la valeur de la fonction objectif à l'optimum global est alors :

$$\tilde{F}_{\text{min}}(\mathbf{X}) = \xi_{\text{opt}} = \max\{\xi | \tilde{\Psi}(\xi) = 0\}$$

et les coordonnées de l'optimum global sont estimées de la manière suivante

$$\tilde{\mathbf{X}}_{\text{opt}} = \tilde{\Psi}_X(\xi_{\text{opt}}) = (\tilde{x}_{1\text{opt}}, \tilde{x}_{2\text{opt}}, \dots, \tilde{x}_{n\text{opt}}) = F^{-1}(\xi_{\text{opt}}).$$

Pour $F^{-1}(\xi)$, nous pouvons procéder de la même manière que pour $\Psi(\xi)$, cette fois en utilisant des centres de gravité $\mathbf{X}_j^* = (x_{1j}^*, x_{2j}^*, \dots, x_{nj}^*)$ où :

$$x_{ij}^* = \Psi_{x_i}(\xi_j) = \frac{\int x_i \cdot \Theta(\mathbf{X}, \xi_j) d\mathbf{X}}{\int \Theta(\mathbf{X}, \xi_j) d\mathbf{X}}, \quad i = 1, 2, \dots, n, \quad j = 1, 2, \dots, M.$$

Nous obtenons ainsi par la méthode Monte-Carlo $\tilde{x}_{ij}^* = \tilde{\Psi}_{x_i}(\xi_j)$; une approximation statistique de ces points nous donne des courbes analytiques $\tilde{\Psi}_{x_i}(\xi)$ pour chaque $x_i, i = 1, 2, \dots, n$. Ces dernières sont utilisées pour obtenir $\tilde{\mathbf{X}}_{\text{opt}} = \tilde{\Psi}_X(\xi_{\text{opt}}) = F^{-1}(\xi_{\text{opt}})$.

Une illustration des calculs de $\tilde{\Psi}_{x_i}(\xi_j)$ est donnée dans la figure 3.

3. ALGORITHME D'OPTIMISATION

L'algorithme d'optimisation est expliqué dans la figure 4 (voir également [4,12]). Nous détaillons ci-dessous ses étapes.

3.1. DONNÉES ET PARAMÈTRES

Nous supposons que pour chaque vecteur \mathbf{X} nous savons calculer la valeur de la fonction objectif $F(\mathbf{X})$ et celles des fonctions $g_l(\mathbf{X})$ qui définissent les contraintes (par exemple, par des calculs analytiques ou avec un modèle de simulation).

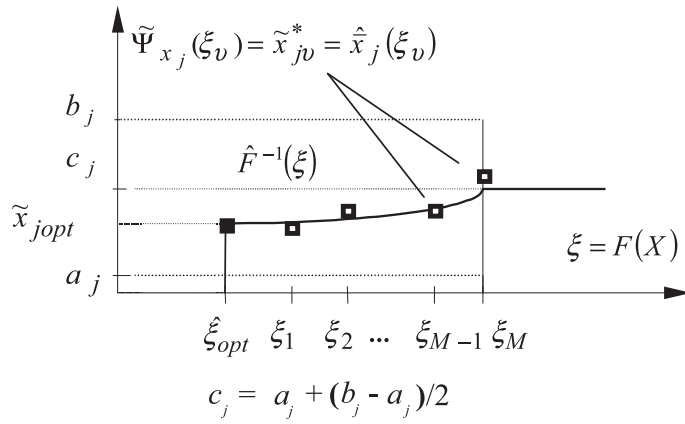


FIGURE 3. Estimation des centres de gravité.

- 1) données et paramètres ;
- 2) formation du domaine de recherche D^p ;
- 3) calcul des $\inf \tilde{F}(X)$ et $\sup \tilde{F}(X)$;
- 4) $R = (\inf \tilde{F}(X) - \sup \tilde{F}(X))$; $R > 0 ?$;
- 5) $\sup \tilde{F}(X) = 0$;
- 6) L -partition ;
- 7) estimation des $\tilde{\mu}_j$ et \tilde{X}_j^* ;
- 8) approximation par des fonctions analytiques ;
- 9) calcul des $\tilde{F}_{\min}(X)$, \tilde{X}_{opt} et $F_{\min}(X)$;
- 10) vérification : $\tilde{X}_{\text{opt}} \in D^p ?$;
- 11) retour dans D^p ;
- 12) résultats d'optimisation ;
- 13) vérification $|F_{\text{opt}}^{\rho-1}(X) - F_{\text{opt}}^{\rho}(X)| < \varepsilon ?$;
- 14) fin.

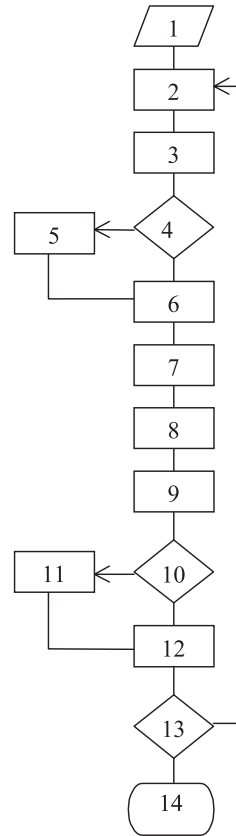


FIGURE 4. Algorithme d'optimisation.

Les paramètres de l'algorithme sont les suivants : P est le nombre d'essais aléatoires initiaux, M représente le nombre d'intervalles de L -partition et le $N \gg P$ indique le nombre total de simulations Monte-Carlo. Les P premières simulations permettent d'obtenir une L -partition. Un nombre plus important de simulations $N-P$ est ensuite réalisé pour trouver des approximations appropriées des fonctions de la Ψ -transformation.

Notre expérience montre qu'avec $20 \leq P \leq 50$, $N = \eta P$ ($\eta = 10, \dots, 100$) et $M \geq 3\chi$, où χ est le nombre de paramètres de la fonction d'approximation choisie (par exemple, pour une fonction quadratique $M \geq 9$), les résultats sont satisfaisants sur les exemples que nous avons traités.

3.2. CONSTRUCTION D'UNE L -PARTITION

En utilisant P fois la simulation Monte-Carlo du vecteur \mathbf{X} , nous obtenons un ensemble de points $\{\mathbf{X}_\alpha\} \in D^\rho$. Les valeurs correspondantes de la fonction objectif sont $\{F(\mathbf{X}_\alpha)\}$, où $\alpha = 1, 2, \dots, P$, D^ρ est le domaine de recherche à l'itération ρ (voir Sect. 3.7). L'intervalle des valeurs possibles de $F(\mathbf{X})$ est estimé donc avec :

$$\begin{aligned} \inf \tilde{F}(\mathbf{X}) &= \min_{\alpha} \{F(\mathbf{X}_\alpha)\}, \\ \sup \tilde{F}(\mathbf{X}) &= \max_{\alpha} \{F(\mathbf{X}_\alpha)\} \end{aligned}$$

où $\tilde{F}(\mathbf{X})$ est une fonction empirique $\tilde{F}(\mathbf{X}) = \{F(\mathbf{X}_\alpha)\}$ définie initialement pour $\{\mathbf{X}_\alpha\} \in D^\rho$, elle sera utilisée pour la construction de la L -partition de la fonction $F(\mathbf{X})$ et pour la Ψ -transformation (elle se substitue à la fonction $F(\mathbf{X})$ qui est inconnue). (Notons qu'en faisant la Ψ -transformation, nous compléterons le domaine de définition de cette fonction par des points obtenus par $N-P$ simulations Monte-Carlo supplémentaires et par le point qui est une prévision de l'optimum global.)

Nous vérifions la condition $R = (\inf \tilde{F}(\mathbf{X}), \sup \tilde{F}(\mathbf{X})) < 0$. Si dans D^ρ la fonction $\tilde{F}(\mathbf{X})$ prend des valeurs positives et négatives, alors nous choisissons l'intervalle $[\inf \tilde{F}(\mathbf{X}), 0]$, sinon l'intervalle $[\inf \tilde{F}(\mathbf{X}), \sup \tilde{F}(\mathbf{X})]$.

Pour obtenir la L -partition, l'intervalle choisi est divisé en M sous-intervalles de même longueur.

3.3. CALCUL DES ESTIMATIONS $\tilde{\mu}_j$ ET \tilde{x}_j^*

Par $N-P$ simulations supplémentaires, nous complétons l'ensemble $\{\mathbf{X}_\alpha\} \in D^\rho$ et nous calculons $\{F(\mathbf{X}_\alpha)\}$, $\alpha = 1, 2, \dots, N \gg P$ (les P simulations initiales sont donc également utilisées), alors nous avons :

$$\tilde{\mu}_j = S_j/N,$$

où S_j est le nombre des valeurs $\tilde{F}(\mathbf{X}) \in \{F(\mathbf{X}_\alpha)\}$ telles que $\tilde{F}(\mathbf{X}) \leq \xi_j, j = 1, 2, \dots, M$.

$$\tilde{x}_{ij}^* = \left(\sum_{\omega=1}^{S_j} x_{i\omega} \right) / S_j,$$

$$\tilde{\mathbf{X}}_j^* = (\tilde{x}_{1j}^*, \tilde{x}_{2j}^*, \dots, \tilde{x}_{nj}^*)$$

où $\tilde{\mathbf{X}}_j^*$ est une estimation du centre de gravité de l'ensemble E_j , \tilde{x}_{ij}^* est la moyenne de la variable de décision i pour les solutions appartenant à l'ensemble E_j .

3.4. APPROXIMATION DES RÉSULTATS

À partir de deux séquences de valeurs $\{\tilde{\mu}_j\} \Leftrightarrow \{\xi_j\}$, $\{\tilde{x}_{ij}^*\} \Leftrightarrow \{\xi_j\}$, $j = 1, 2, \dots, M$, $i = 1, 2, \dots, n$, en utilisant, par exemple, des fonctions quadratiques suivantes :

$$\tilde{\Psi}(\xi) = B_1(\mu)\xi^2 + B_2(\mu)\xi + B_3(\mu),$$

$$\tilde{\Psi}_{x_i}(\xi) = B_1(x_i)\xi^2 + B_2(x_i)\xi + B_3(x_i),$$

nous obtenons $\tilde{\mu} = \tilde{\Psi}(\xi)$ et $\tilde{x}_i^* = \tilde{\Psi}_{x_i}(\xi)$, $i = 1, 2, \dots, n$.

La méthode peut bien sûr s'étendre à des approximations de degré 3, 4 ou plus. Il nous semble pourtant qu'étant donné que nous ne cherchons pas une prévision exacte de l'optimum global, qui est souvent impossible, mais simplement à détecter sa zone d'attraction, des approximations plus simples et plus rapides nous suffisent.

3.5. PRÉVISIONS DE L'OPTIMUM GLOBAL

Le calcul des prévisions de la valeur de l'optimum global $\tilde{F}_{\min}(\mathbf{X})$, d'une part, et de ses coordonnées \tilde{X}_{opt} , de l'autre, ainsi que le calcul de la valeur réelle de la fonction objectif obtenue $F_{\min}(X)$ peut se faire donc de la manière suivante :

$$\tilde{F}_{\min}(\mathbf{X}) = \xi_{\text{opt}} = \min_{\xi} (B_1(\mu)\xi^2 + B_2(\mu)\xi + B_3(\mu)),$$

$$\tilde{x}_{i \text{ opt}} = B_1(x_i)\xi_{\text{opt}}^2 + B_2(x_i)\xi_{\text{opt}} + B_3(x_i), i = 1, 2, \dots, n,$$

$$F_{\min}(\mathbf{X}) = F(\tilde{X}_{\text{opt}}).$$

Étant donné que, dans la plupart de cas, ξ_{opt} se trouve en dehors de l'intervalle initial de L -partition, pour obtenir $\tilde{x}_{i \text{ opt}}$, nous procédons par extrapolation.

Après l'extrapolation, il faut vérifier la condition $\tilde{X}_{\text{opt}} \in D^\rho$. Si la solution \tilde{X}_{opt} obtenue n'appartient plus à D^ρ , alors il faut revenir au domaine D^ρ (en remontant des fonctions quadratiques utilisées).

La recherche se termine quand $|F_{\min}^{\rho-1} - F_{\min}^\rho| \leq \varepsilon$, où la valeur de ε est donnée par l'utilisateur.

3.6. OPTIMISATION LOCALE

À la fin de chaque itération ρ , nous utilisons l'algorithme d'optimisation locale ci-dessous [4, 12]. Dans cet algorithme, nous fixons tous les variables sauf une, et pour celle-ci, nous balayons son domaine de définition, en essayant d'améliorer la valeur de la fonction objectif. Cette procédure est répétée pour chaque variable. Et cela dans une boucle qui se termine quand nous ne pouvons plus améliorer le critère, c'est-à-dire, quand nous trouvons un optimum local (la précision est définie par le pas Δx_i qui est propre à chaque itération ρ et à chaque variable $i = 1, 2, \dots, n$).

Étape 1. $F_{\min} = F(\widetilde{\mathbf{X}}_{\text{opt}})$

$$h_i = 1, \quad i = 1, 2, \dots, n, \quad h_i = \Delta x_i, \quad i = n_1 + 1, n_1 + 2, \dots, n,$$

où Δx_i dépend du numéro de l'itération ρ .

Étape 2. Pour chaque i ,

$$\begin{aligned} Y_i^+ &= \{\widetilde{x}_{i \text{ opt}} + h_i, \widetilde{x}_{i \text{ opt}} + 2 \cdot h_i, \dots, b_i; x_k = \widetilde{x}_{k \text{ opt}}, \forall k \neq i\} \\ Y_i^- &= \{\widetilde{x}_{i \text{ opt}} - h_i, \widetilde{x}_{i \text{ opt}} - 2 \cdot h_i, \dots, a_i; x_k = \widetilde{x}_{k \text{ opt}}, \forall k \neq i\} \\ Y_i &= Y_i^- \cup Y_i^+. \end{aligned}$$

Étape 3. Calculer $F(\mathbf{X})$ pour chaque $\mathbf{X} \in \bigcup_{i=1}^n Y_i$.

Étape 4. Choisir la meilleure solution :

$$\mathbf{X}' = \operatorname{argmin} \left\{ F(\mathbf{X}) \mid \mathbf{X} \in \bigcup_{i=1}^n Y_i \right\}.$$

Étape 5. $\widetilde{\mathbf{X}}_{\text{opt}} = \mathbf{X}'$, $\Delta F = |F(\mathbf{X}_{\text{opt}}) - F_{\min}|$.

Étape 6. Si $\Delta F \neq 0$ aller à l'étape 1.

3.7. RÉDUCTION DE L'ESPACE DE RECHERCHE

Au fur et à mesure des itérations, la recherche locale devient de plus en plus gourmande en temps de calcul. Pour pallier cela et pour augmenter la précision des prévisions, nous essayons de réduire en même temps l'espace de recherche. L'idée de base est la suivante : plus nous avançons dans l'algorithme, plus il faut concentrer la recherche autour des meilleures solutions.

Supposons qu'à l'itération $\rho = 0$, après la recherche locale, nous ayons trouvé dans l'espace D un minimum \widetilde{F}_{\min}^1 et ses coordonnées $\widetilde{x}_{i \text{ opt}}^1$. Nous allons construire un nouvel espace de recherche $D^1 \subset D$ pour l'itération suivante en utilisant la procédure ci-dessous [4, 12] :

- pour chaque paramètre x_i , en fixant les autres paramètres à leur valeur $\tilde{x}_{m \text{ opt}}^1$, $m \neq i$, nous balayons l'espace D avec un nouveau pas h_i plus petit ;
- les valeurs de la fonction $F(\mathbf{X})$ obtenues pour $\tilde{x}_{i \text{ opt}}^1 + kh_i$ et pour $\tilde{x}_{i \text{ opt}}^1 - kh_i$ ($k = 1, 2, \dots$) sont comparées ; le nouveau espace de recherche est alors construit :

$$D^1 = (A^1 \leq \mathbf{X} \leq B^1 : A^1 = \{a_i^1 = \tilde{x}_{i \text{ opt}}^1 - kh_i\}, B^1 = \{b_i^1 = \tilde{x}_{i \text{ opt}}^1 + kh_i\}),$$

où k est telle que

$$\begin{aligned} F(a_i^1) &< \tilde{F}_{\min}^1, \text{ et } F(\tilde{x}_{i \text{ opt}}^1 + (k+1)h_i) \geq \tilde{F}_{\min}^1, \\ F(b_i^1) &< \tilde{F}_{\min}^1, \text{ et } F(\tilde{x}_{i \text{ opt}}^1 + (k+1)h_i) \geq \tilde{F}_{\min}^1. \end{aligned}$$

Si nous ne trouvons pas une telle valeur de k , nous gardons l'espace courant. Le pas h_i dépend du numéro de l'itération ρ ; plus ρ est grand plus le pas est petit. Après $r+1$ itérations, nous avons $D^r \subseteq D^{r-1} \subseteq \dots \subseteq D^\rho \subseteq \dots \subseteq D^1 \subseteq D^0 = D$.

Cette procédure permet de réduire l'espace de recherche et de diminuer ainsi le nombre de calculs nécessaires (il faut bien sûr garder les meilleures solutions, même si elles ne sont plus dans l'espace de recherche courant).

4. EXEMPLES NUMÉRIQUES

4.1. OPTIMISATION GLOBALE DE FONCTIONS NON CONVEXES

4.1.1. Fonctions avec des formes analytiques particulières

Pour tester notre algorithme, nous avons d'abord utilisé sept fonctions non convexes avec des formes analytiques particulières [14]. Elles ont été construites comme superposition des fonctions ci-dessous.

- *Fonction de forme exponentielle (T1) :*

$$T_1(\mathbf{X}) = A \pm \sum_{i=1}^m \Phi_i(\mathbf{X}) \exp(-b_i [f_i(a_i \mathbf{X}) - y_i^*])^2 \quad (2)$$

où $\mathbf{X} = (x_1, x_2, \dots, x_n) \in D \subset R^n$; $A, a_i, b_i > 0$ sont des constantes ; $\Phi_i(\mathbf{X}) \geq 0$, $f_i(\mathbf{X})$ sont des fonctions connues ; y_i^* est une valeur de la fonction $f_i(a_i \mathbf{X})$.

- *Fonction avec domaine d'attraction des optima sous forme d'anneau (T2) :*

$$T_2(\mathbf{X}) = \pm \Phi(\mathbf{X}) \text{Sin}f(|\mathbf{X} - \mathbf{X}_0|) + A, \quad (3)$$

où $\mathbf{X} \in D \subset R^n$; $\Phi(\mathbf{X}) \geq 0$ et $f(*)$ sont des fonctions avec des propriétés données ; $\mathbf{X}_0 \in R^n$ est un vecteur donné ; A est une constante.

- *Fonction avec optima sur les frontières de leur domaine de définition (T3) :*

$$T_3(\mathbf{X}) = A + R(\mathbf{X} - \mathbf{X}_0)f(\mathbf{X}), \quad \mathbf{X} \in D, \quad (4)$$

où D est un domaine de R^n connexe et limité avec comme frontière Γ_{n-1} ; \mathbf{X}_0 est un point interne de D ; A est une constante; $R(\mathbf{X} - \mathbf{X}_0) = 1$ si $\mathbf{X} \in \Gamma_{n-1}$ et $R(\mathbf{X} - \mathbf{X}_0) > (<)1$ si $\mathbf{X} \in D$ et $\mathbf{X} \notin \Gamma_{n-1}$; $f(\mathbf{X})$ est une fonction connue.

- *Fonction continue par morceaux (T4) :*

$$T_4(\mathbf{X}) = A \pm \sum_i J_i(\mathbf{X}) f_i(\mathbf{X}); \quad \mathbf{X} \in D, \quad (5)$$

où A est une constante; $J_i(\mathbf{X}) = \begin{cases} 1, & \mathbf{X} \in D_i \\ 0, & \mathbf{X} \notin D_i \end{cases}$ sont des fonctions indicateurs des domaines D_i ; $D_i \cap D_j = \emptyset, i \neq j$; $D = \bigcup_i D_i$ est le domaine de définition de $T_4(\mathbf{X})$; $f_i(\mathbf{X})$ sont des fonctions continues avec les domaines d'attraction des optima connus.

4.1.2. Fonctions utilisées

Par combinaison des fonctions type **T1-T4**, sept fonctions concrètes avec une topologie difficile ont été obtenues. Elles ont été utilisées pour étudier le comportement de l'algorithme d'optimisation.

Fonction F1. Pour la construction de cette fonction nous avons utilisé des fonctions $T_1(\mathbf{X})$ et $T_4(\mathbf{X})$. Le domaine de définition est choisi comme $D = \{(x_1, x_2) : 0 \leq x_1 \leq 1, 0 \leq x_2 \leq 1\}$, des domaines D_i sont donnés par leurs fonctions indicateurs :

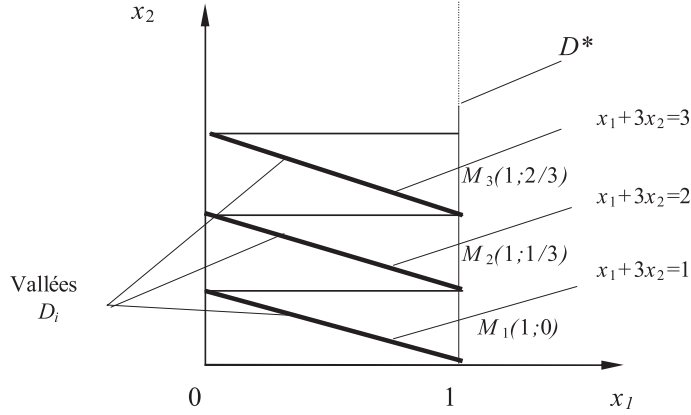
$$J_1(x_1, x_2) = \begin{cases} 1, & 0 \leq x_2 < 1/3, \\ 0, & 1/3 \leq x_2 \leq 1. \end{cases}$$

$$J_2(x_1, x_2) = \begin{cases} 1, & 1/3 \leq x_2 < 2/3, \\ 0, & 0 \leq x_2 < 1/3, 2/3 \leq x_2 \leq 1. \end{cases}$$

$$J_3(x_1, x_2) = \begin{cases} 1, & 2/3 \leq x_2 \leq 1, \\ 0, & 0 \leq x_2 < 2/3. \end{cases}$$

La forme générale de la fonction $F_1(\mathbf{X})$ est la suivante :

$$F_1(\mathbf{X}) = A \pm \sum_{i=1}^3 J_i(\mathbf{X}) f_i(\mathbf{X}).$$

FIGURE 5. Représentation de la fonction $F_1(\mathbf{X})$.

La constante $A = 7$ et des fonctions $f_i(\mathbf{X})$ sont obtenues à base de $T_1(\mathbf{X})$:

$$f_i(\mathbf{X}) = A_i - \Phi_i(\mathbf{X}) \exp(-b_i[g_i(a_i\mathbf{X}) - y_i^*])^2, \quad 1 \leq i \leq 3,$$

où

$$A_i = 0;$$

$$\Phi_1(\mathbf{X}) = 2/(2 - x_1), \quad \Phi_2(\mathbf{X}) = 2.2/(2 - x_1), \quad \Phi_3(\mathbf{X}) = 2.25/(2 - x_1);$$

$$b_1 = b_2 = b_3 = a_1 = a_2 = a_3 = 1;$$

$$g_1(x_1, x_2) = g_2(x_1, x_2) = g_3(x_1, x_2) = x_1 + 3x_2;$$

$$y_1^* = 1, \quad y_2^* = 2, \quad y_3^* = 3.$$

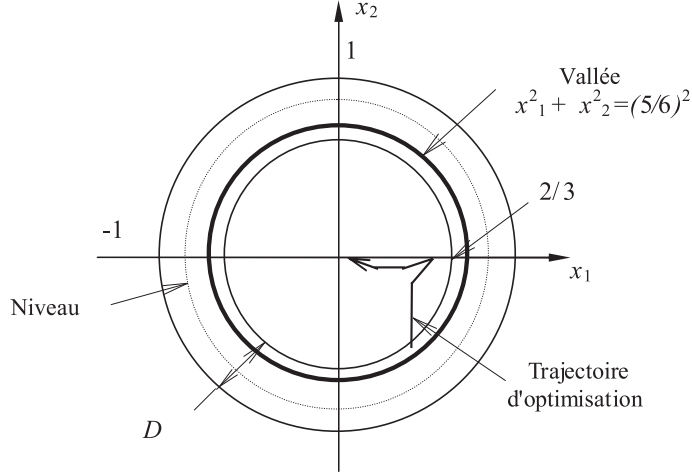
La fonction obtenue $F_1(\mathbf{X})$ est donc la suivante :

$$\begin{aligned} F_1(\mathbf{X}) = & 7 - (2/(2 - x_1)) \exp(-(x_1 + 3x_2 - 1)^2) J_1(x_1, x_2) \\ & - (2.2/(2 - x_1)) \exp(-(x_1 + 3x_2 - 2)^2) J_2(x_1, x_2) \\ & - (2.25/(2 - x_1)) \exp(-(x_1 + 3x_2 - 3)^2) J_3(x_1, x_2). \end{aligned} \quad (6)$$

Cette fonction a trois vallées (voir Fig. 5) : $x_1 + 3x_2 = 1$; $x_1 + 3x_2 = 2$; $x_1 + 3x_2 = 3$. Les optima locaux se trouvent aux points $M_1 = (1; 0)$, $M_2 = (1; 1/3)$, $M_3 = (1; 2/3)$. L'optimum global (minimum) est $M_3 = (1; 2/3)$.

Fonction F_2 . Cette fonction utilise le modèle $T_2(\mathbf{X})$, $D = \{(x_1, x_2) : (2/3)^2 \leq x_1^2 + x_2^2 \leq 1\}$, avec : $\Phi(\mathbf{X}) \equiv 5$; $A = 3$; $\mathbf{X}_0 = 0$, $f(|\mathbf{X} - \mathbf{X}_0|) = f(|\mathbf{X}|) = 3\pi(\sqrt{x_1^2 + x_2^2} - 2/3)$. Nous avons donc :

$$F_2(\mathbf{X}) = -5 \sin \left(3\pi \left(\sqrt{x_1^2 + x_2^2} - 2/3 \right) \right) + 3. \quad (7)$$

FIGURE 6. Représentation de la fonction $F_2(\mathbf{X})$.

Sa valeur optimale est $F_2(\mathbf{X}) = -2$. En effet, avec $D_k^* = \{(x_1, x_2) : 3\pi(\sqrt{x_1^2 + x_2^2} - 2/3) = \pi/2 + 2\pi k\} = \{(x_1, x_2) : x_1^2 + x_2^2 = (1/6 + 2/3 + 2k)^2 = (5/6 + 2k)^2\}$, $D \cap (\bigcup_k D_k^*) = \{(x_1, x_2) : x_1^2 + x_2^2 = (5/6)^2\}$ représente le fond de la vallée (voir Fig. 6).

Fonction F_3 . Pour cette fonction, nous avons utilisé $T_3(\mathbf{X})$:

$$F_3(\mathbf{X}) = A + R(\mathbf{X} - \mathbf{X}_0)f(\mathbf{X}),$$

avec $\mathbf{X} = (x_1, x_2)$, $D = \{(x_1, x_2) : |x_1| + |x_2| \leq 1\}$, $A = 5$, $\mathbf{X}_0 = 0$, et étant donné que

$$\begin{aligned} \sin 2\varphi &= 2 \sin \varphi \cos \varphi = 2 \left(x_1 / \sqrt{x_1^2 + x_2^2} \right) \left(x_2 / \sqrt{x_1^2 + x_2^2} \right) \\ &= 2(x_1 x_2) / (x_1^2 + x_2^2), (\varphi \neq 0), \end{aligned}$$

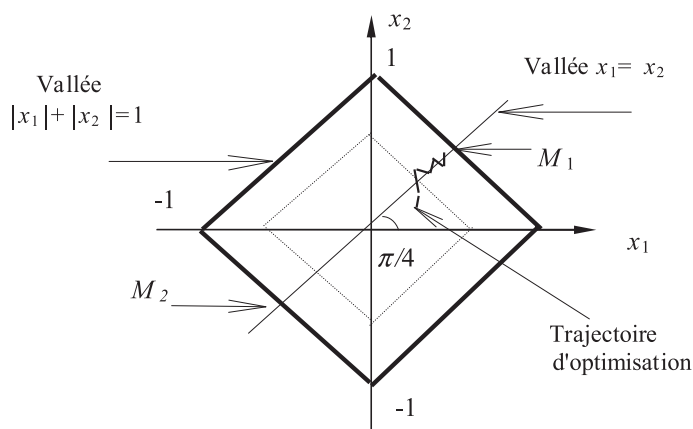
nous avons

$$F_3(\mathbf{X}) = 5 - 4 \sin 2\varphi (|x_1| + |x_2|) = 5 - 8 \left((x_1 + x_2) / (x_1^2 + x_2^2) \right) / (|x_1| + |x_2|). \quad (8)$$

La valeur de la fonction $F_3(\mathbf{X})$ est maximale quand $\varphi = \pi/4$ (point M_1) ou $\varphi = \pi/4 + \pi$ (point M_2). Cela correspond à la droite passant par l'origine avec l'angle $\varphi = \pi/4$ (voir Fig. 7).

Les autres fonctions sont les suivantes [14].

Fonction F_4 . Nous avons $F_4(\mathbf{X}) = 5 / (1 + |x_1 - 5/6|) \sin(3\pi(\sqrt{x_1^2 + x_2^2} - 2/3)) + 3$, avec comme domaine de définition $D = \{(x_1, x_2) : (2/3)^2 \leq x_1^2 + x_2^2 \leq 1\}$, l'optimum global est au point $\mathbf{X} = (5/6; 0)$.

FIGURE 7. Représentation de la fonction $F_3(\mathbf{X})$.

Fonction F_5 . Nous avons $F_5(\mathbf{X}) = 5 - 4x_2^2/(x_1^2 + x_2^2) \max\{|x_1|, |x_2|\}$, avec $D = \{(x_1, x_2) : -1 \leq x_1, x_2 \leq 1, x_i \neq 0\}$. Elle a deux optima globaux : $\mathbf{X} = (0; -1)$ et $\mathbf{X} = (0; 1)$.

Fonction F_6 . Nous avons $F_6(\mathbf{X}) = x_1^2 + x_2^2 - \cos 18x_1 - \cos 18x_2$, si $D = \{(x_1, x_2) : -1 \leq x_1, x_2 \leq 1; x_1 + 1.1x_2 - 0.1 \leq 0; -x_1 + 0.5x_2 - 0.5 \leq 0\}$, l'optimum global se trouve en $\mathbf{X} = (0; 0)$.

Fonction F_7 . Cette fonction a été proposée par Chichinadze dans [2] :

$$F_7(\mathbf{X}) = x_1^2 - 12x_2 + 11 + 10 \cos(\pi x_1/2) + 8 \sin(\pi x_1) - \exp(-(x_2 - 0.5)^2/2)/\sqrt{2\pi},$$

$$D = \{(x_1, x_2) : 0 \leq x_1 \leq 10, 0 \leq x_2 \leq 5, 0.5x_1 + 2x_2 \leq 8\}.$$

4.1.3. Résultats de l'optimisation

Les fonctions F_1 - F_7 représentent assez bien différentes topologies difficiles pour l'optimisation globale : continue par morceaux, symétrique, avec plusieurs vallées et avec plusieurs optima globaux.

Les résultats d'optimisation de ces fonctions par notre algorithme de Ψ -transformation sont donnés dans le tableau 1. Les trajectoires de déplacement des centres de gravité des ensemble $E_j(\xi_j)$, pour les fonctions F_2 et F_3 , sont données dans les figures 6 et 7 respectivement. Pour la fonction F_2 l'algorithme n'a pas donné des résultats satisfaisants à cause sa topologie particulière. Cette fonction a un nombre infini d'optima globaux qui se trouvent sur l'anneau. L'utilisation des centres de gravité n'est pas appropriée pour ce cas, car elle nous amène inévitablement au point $(0,0)$.

Nous allons présenter plus en détails les résultats de l'optimisation (minimisation) de la fonction F_6 [12]. Cette fonction a 10 optima locaux. L'optimum global se trouve au point $x_1 = x_2 = 0$, sa valeur optimale est $F_6(0, 0) = -2$.

TABLEAU 1. Résultats d'optimisation pour les fonctions F_1 - F_7 .

Fonction	Coordonnées de l'optimum global				Valeur de l'optimum global	
	Réelles		Obtenues par l'algorithme		Réelle	Obtenue par l'algorithme
	x_1	x_2	x_1	x_2	$F(x_1, x_2)$	$F(x_1, x_2)$
F_1	1,00000	0,66666	1,00000	0,67072	4,75000	4,75068
F_2	l'algorithme ne marche pas					
F_3	$\pm 0,5$	$\pm 0,5$	$\pm 0,50047$	0,49951	1,00000	1,00009
F_4	0,83333	0,00000	0,83333	0,01905	-2,00000	-1,99998
F_5	0,00000	$\pm 1,00000$	0,00075	$\pm 1,00000$	1,00000	1,00000
F_6	0,00000	0,00000	-0,00012	-0,00017	-2,00000	-1,9999927
F_7	5,62680*	0,49710*	5,623802	0,499857	-40,9623*	-40,964149

* sont les valeurs données par Chichinadze dans [2].

Deux itérations de l'algorithme ont été faites, la première avec $M = 50$ (nombre d'intervalles de la partition de Lebesgue) et $P = 30$ (nombre de simulations initiales pour faire la partition de Lebesgue). L'algorithme d'optimisation locale n'a pas été utilisé. Les résultats suivants ont été obtenus : $x_{1\text{opt}}^1 = -0,027395$; $x_{2\text{opt}}^1 = -0,140744$; $\tilde{F}_{\min}^1 = -1,61393$, où \tilde{F}_{\min}^1 est la prévision de la valeur de l'optimum global à la première itération. La valeur réelle de la fonction $F_{\min}^1(x_{1\text{opt}}^1, x_{2\text{opt}}^1) = -0,03962$.

Après l'application de la procédure de réduction du domaine de recherche, nous avons obtenu le domaine D^1 avec les restrictions suivantes sur les variables x_1 et x_2 : $-0,027395 \leq x_1 \leq 0,002604$; $-0,140744 \leq x_2 \leq 0,00425$.

La deuxième itération a été faite avec $M = 10$ et $P = 30$, elle a donné les résultats suivants : $x_{1\text{opt}}^2 = -0,00011844$; $x_{2\text{opt}}^2 = -0,00017394$; $\tilde{F}_{\min}^2 = -2,34797$; $F_{\min}^2 = -1,9999927$.

Les résultats sont satisfaisants. Les valeurs intermédiaires sont données dans la figure 8, les fonctions quadratiques sont présentées avec des échelles différentes. L'axe vertical droit correspond à l'évolution des centres de gravité pour la première itération, l'axe vertical gauche correspond à la deuxième itération. Les courbes avec des carrés (1 et 4) donnent les résultats de la première itération, les courbes continues (2 et 3) représentent la deuxième itération.

Pour ξ et pour chaque itération respectivement, les intervalles $(0; -1,3)$ et $(0; -1,99)$ sont des intervalles où l'approximation a été effectuée, et les intervalles $(-1,3; -1,61)$ et $(-1,99; -2,3)$ sont ceux qui correspondent à l'extrapolation.

Dans le tableau 2 nous donnons des résultats d'optimisation des fonctions F_6 et F_7 avec un algorithme modifié [12]. La modification consistait à introduire les poids des valeurs utilisées dans les calculs des fonctions $\tilde{\mu} = \tilde{\Psi}(\xi)$ et $\tilde{x}_i^* = \tilde{\Psi}_{x_i}$.

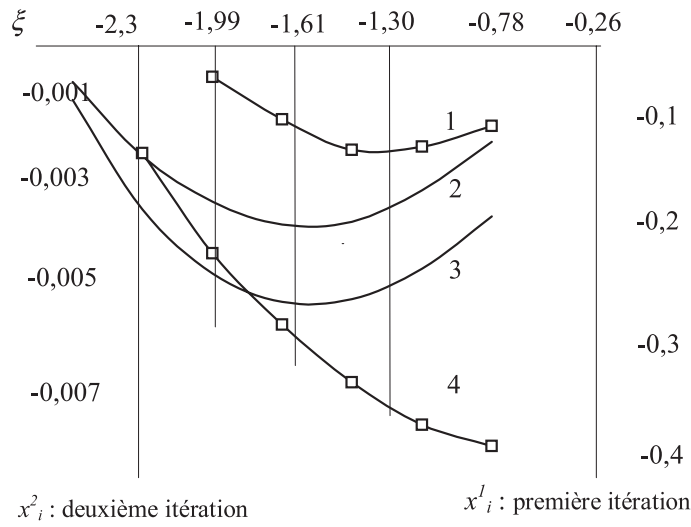


FIGURE 8. Recherche des coordonnées de l'optimum global pour F_6 :

1. $\tilde{x}_1^{*1} = 0,07\xi^2 + 0,07\xi - 0,09$;
2. $\tilde{x}_2^{*2} = 0,0014\xi^2 + 0,002\xi - 0,0028$;
3. $\tilde{x}_1^{*2} = 0,019\xi^2 + 0,003\xi - 0,0039$;
4. $\tilde{x}_2^{*1} = 0,07\xi^2 + 0,03\xi - 0,038$.

Pour l'optimisation de F_6 , la recherche locale n'a pas été utilisée, par contre, elle a été utilisée pour F_7 .

Les poids $\alpha_j, j = 1, 2, \dots, M$, ont été calculés de la manière suivante : $\alpha_j = 1$ (le même poids pour toutes les valeurs) ; $\alpha_j = 1 + (1 - j)/M$ (augmentation linéaire des poids) ; $\alpha_j = \exp(-(j - 1)/10)$ (augmentation exponentielle des poids) où j est le numéro de l'ensemble E_j .

Le tableau 2 montre que l'introduction des poids permet d'améliorer la qualité des résultats, surtout à la première itération.

4.2. APPLICATION EN OPTIMISATION COMBINATOIRE

4.2.1. Un problème d'ordonnancement job-shop

Nous présentons ici rapidement un exemple d'ordonnancement dynamique des ateliers de type job-shop [3, 4, 6, 11]. Dans la figure 9, nous expliquons le fonctionnement de ce type d'atelier.

Les règles d'ordonnancement sont les suivantes. À l'intérieur de l'atelier, l'ordonnancement suit une règle de priorité simple : la pièce la plus prioritaire est celle dont la date de livraison demandée est la plus proche. Par contre, au lancement des commandes en production, deux paramètres sont utilisés :

TABLEAU 2. Résultats obtenus pour les fonctions F_6, F_7 avec l'utilisation des poids.

Poids	Valeurs	Fonction F_6		Fonction F_7		
		Première itération	Deuxième itération	Première itération	Deuxième itération	Recherche locale
$\alpha_j = 1$	$\tilde{F}(\mathbf{X})$	-1,6139313	-2,3479703	-34,880400	-41,003429	
	$F(\mathbf{X})$	-0,0396200	-1,9999927	-40,702970	-40,958447	-40,963338
	\tilde{x}_1	-0,0273952	-0,0001184	5,686335	5,514509	5,6195087
	\tilde{x}_2	-01407446	-0,0001739	1,150759	0,595467	0,5004673
$\alpha_j = 1 + \frac{1-j}{M}$	$\tilde{F}(\mathbf{X})$	-1,5846919	-2,1826236	-36,17339	-40,968857	
	$F(\mathbf{X})$	-0,2049693	-1,9999402	-40,870156	-40,643071	-40,964078
	\tilde{x}_1	-0,0245259	-0,0004811	5,644394	5,542277	5,6222774
	\tilde{x}_2	-0,1289550	0,0003678	1,137199	0,443958	0,4989577
$\alpha_j = \exp\left(-\frac{(j-1)}{10}\right)$	$\tilde{F}(\mathbf{X})$	-1,6242987	-2,2053418	-37,887526	-40,984035	
	$F(\mathbf{X})$	-0,3820997	-1,9999973	-40,823948	-40,805066	-40,964149
	\tilde{x}_1	-0,0238304	-0,0001107	5,5832	5,681356	5,6238029
	\tilde{x}_2	-0,1171669	-0,0000721	1,080616	0,401026	0,4998571

- le nombre maximum nm_i de pièces de type i admis dans le système de production (en-cours + produits finis) à tout instant ;
- le délai r_i qui définit la date au plus tôt de lancement en production des commandes de type i .

Le lancement est effectué seulement si les deux conditions suivantes sont réunies :

- $d_i^j - r_i \leq 0$ où d_i^j est la date de livraison de la commande j de type i ;
- en introduisant dans le système le produit j de type i , le nombre total des produits de type i reste $\leq nm_i$.

Le critère d'optimisation est le coût moyen par période qui est la somme du coût moyen de stockage et du coût moyen des commandes retardées (rupture de production). Les variables décisionnelles sont les paramètres nm_i et r_i . Les coûts unitaires par produit et pour une période de temps sont connus.

L'heuristique la plus performante pour ce type de problème est celle de Proth et Sauer [11], elle utilise un modèle de simulation et un algorithme basé sur l'analyse de sensibilité et l'analyse de perturbations. Nous l'avons comparée avec notre algorithme. Pour cela nous avons créé et testé 5 exemples [4]. Ces données sont regroupées dans le tableau 3. Pour plus de détails, voir l'annexe 2.

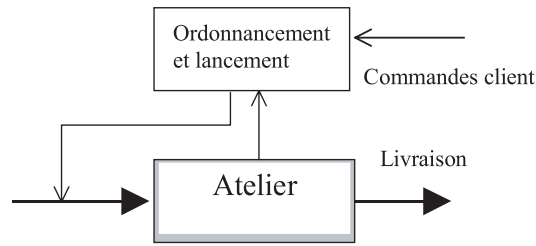


FIGURE 9. Atelier job-shop.

TABLEAU 3. Données pour les problèmes job-shop étudiés.

Exemple	Probabilité P_a^e	Temps moyen entre deux commandes	Délai minimal de livraison	Délai maximal de livraison
1	0,178	5,83	12	60
2	0,172	5,61	11	55
3	0,150	6,64	14	70
4	0,163	6,17	17	85
5	0,176	5,69	15	75

4.2.2. Résultats de l'optimisation

Comme dans [11], pour évaluer le critère nous avons utilisé un modèle de simulation. Chaque réplication était de 1000 périodes. Le nombre initial de répliques P a été pris égal à 300. Le nombre $N-P$ de répliques complémentaires a été choisi de sorte que le nombre total de répliques, pour notre algorithme et pour celui de [11], soit le même. Nous pouvons donc dire que les deux algorithmes demandent le même temps de calcul (les temps propres de deux algorithmes sont négligeables par rapport au temps d'une réplication).

Les résultats sont présentés dans le tableau 4.

TABLEAU 4. Résultats d'optimisation pour le problème d'ordonnancement job-shop .

Exemple	Méthode [11]	Algorithme proposé	Gain %
1	48.94	48.19	1.53
2	57,65	54,10	6,16
3	56,46	50,25	10,9
4	47,65	42,88	10,0
5	56,43	50,33	10,8

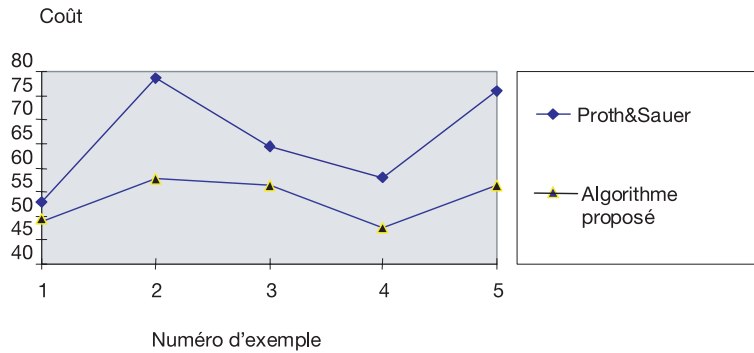


FIGURE 10. Résultats d'optimisation pour le problème d'ordonnement job-shop.

Ces résultats sont également présentés dans la figure 10.

5. CONCLUSIONS

Nous avons étudié un algorithme d'optimisation à base de Ψ -transformation. Les tests effectués montrent que cet algorithme donne des résultats encourageants pour des fonctions continues par morceaux, avec plusieurs vallées et plusieurs optima locaux, quand il n'y a qu'un seul optimum global. Quand il y en a plusieurs ou un nombre infini d'optima globaux, l'algorithme donne de résultats satisfaisants s'il est complété par un algorithme d'optimisation locale ou si l'on arrive à partager le domaine de définition de la fonction étudiée en sous-domaines ayant chacun un seul optimum global.

Les résultats obtenus pour le problème d'ordonnement job-shop que nous avons traité montre que la capacité de notre algorithme à détecter les zones d'attraction de l'optimum global est non négligeable et qu'il peut être utilisé, non seulement pour des fonctions continues non convexes, mais aussi dans l'optimisation combinatoire. Un autre exemple d'optimisation combinatoire concernant l'ordonnement de flow-shop de permutation est traité par cette méthode dans notre article [5].

L'algorithme ne donne pas de résultats satisfaisants quand la fonction a une topologie symétrique avec un centre de symétrie n'appartenant pas au domaine de définition. On peut expliquer cela par le fait que, dans ce cas, les centres de gravité des ensembles E_j n'appartiennent pas au domaine de définition de la fonction optimisée.

Et bien sûr, comme tous les algorithmes d'optimisation stochastique, notre algorithme a une probabilité d'erreur non nulle. Cette probabilité peut être diminuée en augmentant le nombre de simulations effectuées [7].

ANNEXE 1. ENSEMBLES MESURABLES AU SENS DE LEBESGUE

La méthode de la Ψ -transformation repose sur l'estimation de la mesure de Lebesgue des ensembles E_j dans l'espace euclidien, nous rappelons donc ici quelques définitions et propriétés théoriques connues [2].

Soit \mathbf{R}^n espace euclidien de dimension n . La notion d'un ensemble de \mathbf{R}^n mesurable au sens de Lebesgue et la notion de la mesure de Lebesgue correspondante se basent sur une généralisation de la notion de volume à n dimensions. Soit Δ parallélépipède dans \mathbf{R}^n défini comme $a_i \leq x_i \leq b_i, i = 1, 2, \dots, n$. Le volume en n dimensions est donc $v(\Delta) = (b_1 - a_1)(b_2 - a_2) \dots (b_n - a_n)$. Soit E ensemble de \mathbf{R}^n et $\Delta_k (k = 1, 2, \dots)$ un ensemble de parallélépipèdes tels que $\bigcup_{k=1}^{\infty} \Delta_k \supset E$. Prenons la suite $\sum_{k=1}^{\infty} v(\Delta_k)$ alors $\overline{m}(E) = \inf \sum_{k=1}^{\infty} v(\Delta_k)$ est la mesure externe de Lebesgue de l'ensemble E . Si l'ensemble E est fini, alors $0 \leq \overline{m}(E) < +\infty$. Si E est un ensemble fini et si Δ est un parallélépipède contenant E , alors $\Delta \setminus E$ est fini et, par conséquent, $\overline{m}(\Delta \setminus E) < +\infty$. Dans ce cas, $\underline{m}(E) = v(\Delta) - \overline{m}(\Delta \setminus E)$ est la mesure interne de l'ensemble E . Pour un ensemble fini E , $0 \leq \underline{m}(E) \leq \overline{m}(E) < +\infty$.

Définition 1. Un ensemble fini E de \mathbf{R}^n est mesurable au sens de Lebesgue si $\underline{m}(E) = \overline{m}(E)$. La valeur commune correspondante est la mesure de Lebesgue $m(E)$ de l'ensemble E .

Définition 2. Soit M une famille de sous-ensembles de \mathbf{R}^n mesurables au sens de Lebesgue. Cette famille a des propriétés suivantes :

- 1) si $E_i, i = 1, 2, \dots$ est une suite d'ensembles de M , alors $\bigcup_{i=1}^{\infty} E_i \in M$ et $\bigcap_{i=1}^{\infty} E_i \in M$. Si, en plus, les différents ensembles E_i de cette suite n'ont pas d'intersections (*i.e.* $E_i \cap E_j = \emptyset$, pour $i \neq j$), alors

$$\overline{m} \left(\bigcup_{i=1}^{\infty} E_i \right) = \sum_{i=1}^{\infty} m(E_i) ;$$

- 2) si E_a et E_b sont deux ensembles de M , alors $E_a \setminus E_b \in M$. Si, en particulier, $E_b \subset E_a$, alors $m(E_a \setminus E_b) = m(E_a) - m(E_b)$;
- 3) si $E_a, E_b \in M$ et si $E_b \subset E_a$, alors $m(E_a) \leq m(E_b)$;
- 4) soit $E_1 \supset E_2 \supset E_3 \supset \dots \supset E_k \supset \dots$ une suite des ensembles mesurables, si $m(E_1) < \infty$, alors

$$\lim_{k \rightarrow \infty} m(E_k) = m \left(\bigcap_{k=1}^{\infty} E_k \right) ;$$

- 5) si $E_1 \subset E_2 \subset E_3 \subset \dots \subset E_k \subset \dots$ une suite des ensembles mesurables, alors

$$\lim_{k \rightarrow \infty} m(E_k) = m \left(\bigcup_{k=1}^{\infty} E_k \right).$$

Définition 3. Une fonction $f(x)$ est mesurable au sens de Lebesgue, si tout l'ensemble $\{f(x) < a\}$ est mesurable quel que soit la valeur de la constante a .

ANNEXE 2. PARAMÈTRES DE L'ATELIER JOB-SHOP ÉTUDIÉ

Pour tous les exemples étudiés dans la Section 4.2 nous utilisons quatre machines (centre de charge). La matrice \mathbf{C} ci-dessous donne les gammes de fabrication et la matrice \mathbf{T} , les temps opératoires. Les données sont générées de manière aléatoire en utilisant la loi uniforme dans l'intervalle de 0 à 4, pour la matrice \mathbf{C} et dans l'intervalle de 0 à 10 pour la matrice \mathbf{T} [4, 6].

Le premier exemple n'utilise que les deux premières lignes de ces matrices (il n'y a que deux types de produits). Les exemples 2, 3, 4, et 5 ont respectivement 4, 6, 8 et 10 types de produits (utilisent 4, 5, 6 et 10 lignes de matrices \mathbf{C} et \mathbf{T}). Dans l'exemple de la plus grande taille (10 types de produits), nous avons donc 20 variables de décision.

$$\mathbf{C} = \begin{bmatrix} 4, & 1, & 3, & 0, \\ 2, & 1, & 4, & 0, \\ 1, & 2, & 3, & 4, \\ 3, & 2, & 1, & 0, \\ 1, & 2, & 3, & 4, \\ 3, & 2, & 4, & 1, \\ 1, & 3, & 4, & 0, \\ 3, & 4, & 1, & 0, \\ 1, & 2, & 3, & 0, \\ 1, & 3, & 2, & 4, \end{bmatrix} \quad \mathbf{T} = \begin{bmatrix} 6, & 0, & 3, & 8, \\ 3, & 7, & 0, & 1, \\ 9, & 7, & 5, & 1, \\ 5, & 2, & 5, & 0, \\ 9, & 7, & 4, & 8, \\ 8, & 1, & 3, & 2, \\ 5, & 0, & 10, & 5, \\ 2, & 0, & 6, & 9, \\ 5, & 9, & 1, & 0, \\ 2, & 8, & 3, & 2, \end{bmatrix}.$$

Les coûts de stockage et les coûts de rupture (coûts de demandes retardées) sont connus. Les coûts unitaires de rupture de production (non livraison d'une commande pour la date demandée) sont donnés par le vecteur \mathbf{B} :

$$\mathbf{B} = (8, 4, 2, 2, 7, 3, 2, 3, 3, 2).$$

Les coûts unitaires de stockage sont donnés par le vecteur \mathbf{ST} :

$$\mathbf{ST} = (6, 10, 1, 8, 5, 5, 7, 6, 7, 8).$$

Les vecteurs \mathbf{B} et \mathbf{ST} sont générés de manière aléatoire en utilisant la loi uniforme dans l'intervalle de 1 à 10.

Le ratio de taux de commande des produits de différents types est donné par le vecteur \mathbf{Q} :

$$\mathbf{Q} = (0.2, 0.1, 0.1, 0.05, 0.05, 0.2, 0.1, 0.05, 0.05, 0.1).$$

Pour tous les exemples, le pourcentage relatif $(Q_i^e/Q_j^e) \times 100$ reste le même, \forall_i, \forall_j

$$Q_i^e = \frac{Q_i}{\sum_{j=1}^e Q_j},$$

où $e = 2, 4, 6, 8$ et 10 (le nombre de types de produits dans l'exemple correspondant). Les commandes arrivent suivant la loi de Bernoulli avec :

$$P_a^e = \frac{1}{\max_i \sum_{j=1}^e Q_j^e \cdot T_{ji}}.$$

Cette probabilité tient compte de la capacité de la machine goulet. Le délai entre la date d'arrivée de la commande et la date demandée de livraison du produit correspondant suit la loi uniforme dans l'intervalle entre le délai minimal de livraison et le délai maximal de livraison :

$$\left[\sum_{i=1}^4 T_{ji}, y \times \sum_{i=1}^4 T_{ji} \right],$$

où y est une constante (dans ces exemples, $y = 5$).

Le même intervalle que celui donné par les colonnes 4 et 5 du tableau 3 est utilisé pour les valeurs a_j et b_j des paramètres r_j, \forall_j . Les bornes a_j et b_j des paramètres n_j, \forall_j , sont calculés de la manière suivante :

$$a_j = 1;$$

$$b_j = y \times (P_a^e \cdot Q_j^e) \cdot \sum_{i=1}^4 T_{ji}$$

où $\sum_{i=1}^4 T_{ji}$ est le cycle de production ; $P_a^e \times Q_j^e$ est la demande moyenne par période pour le produit j .

RÉFÉRENCES

- [1] G. Berthiau et P. Siarry, État de l'art des méthodes d'optimisation globale. *RAIRO : Oper. Res.* (2001).
- [2] V.K. Chichinadze, *Non linear convex optimization*. Nauka, Moscow (1983) (en russe).
- [3] A. Dolgui, Optimisation des paramètres de modèles de simulation par Ψ -transformation, dans les *Actes de la première conférence francophone de modélisation et de simulation (MOSIM'97)*. Edn. Hermès, Paris (1997) 451-459.
- [4] A. Dolgui et D. Ofitserov, A stochastic method for discrete and continuous optimization in manufacturing systems. *J. Intelligent Manufacturing* **8** (1997) 405-413.
- [5] A. Dolgui et D. Ofitserov, Méthode de Ψ -transformation discrète et son application aux problèmes combinatoires en gestion de production, dans les *Actes de Deuxièmes Journées*

- Francophone de Recherche Opérationnelle (FRANCORO II)*, 6-8 avril 1998, Sousse, Tunisie, édité par F. Ben Abdelaziz, M. Haouari et K. Mellouli. CPU Press (2000) 25-32.
- [6] A. Dolgui et Ch. Thirion, Utilisation des plans d'expériences pour le paramétrage des heuristiques d'ordonnancement dans un atelier job-shop, dans les *Actes de la Deuxième Conférence Francophone de Modélisation et Simulation (MOSIM'99)*, édité par G. Habchi et A. Haurat. SCS Publication (1999) 351-356.
- [7] S.G. Fishman, *Monte-Carlo : Concepts, algorithms, and applications*. Springer-Verlag, New York (1996).
- [8] A.H.G. Kan Rinnooy et G.T. Timmer, Stochastic methods for global optimization. *Amer. J. Math. Management Sci.* **4** (1984) 7-40.
- [9] J. Mockus, Application of Bayesian Approach to Numerical Methods of Global and Stochastic Optimization. *J. Global Optim.* **4** (1994) 347-356.
- [10] I.H. Osman et G. Laporte, Metaheuristics : A bibliography. *Ann. Oper. Res.* **63** (1996) 513-623.
- [11] J.M. Proth et N. Sauer, Sensibility Analysis for Job-Shop Scheduling, dans *Proc. of the IMACS/IEEE Multiconference on Computational Engineering in Systems Applications. CESA'96* (1996) 208-211.
- [12] V.V. Sysoev, *Computer-aided design of production lines in electronics industry*. Radio i Sviazj, Moscow (1982) (en russe).
- [13] A. Törn et A. Zilinskas, *Global Optimization*, édité par G. Goos et J. Hartmanis. Springer Verlag, *Lecture Notes in Comp. Sci.* **350** (1989).
- [14] D.A. Velitchko, V.V. Sysoev et N.V. Beliaev, *Test functions for validation of optimization tools. Technical methods in machines reliability*. Technological Institute of Voronezh (1975) 110-118 (en russe).