الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire

وزارة التعليم العالي و البحث العلمي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

جامعة ابن خلدون تيارت
Université Ibn-Khaldoun de Tiaret

****

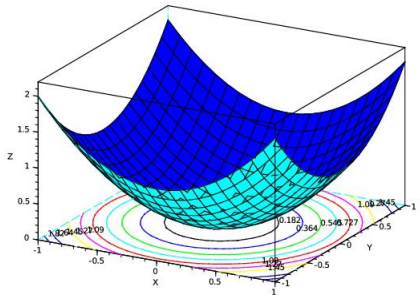
Faculté des Sciences Appliquées
Département de Génie Mécanique

****

Polycopié du cours

intitulé

Optimisation



Présenté aux étudiants au parcours:

Master académique en Génie Mécanique
Spécialité : Énergétique

Élaboré par
Dr BENARBIA Djamila

Évalué par **Dr MAKHFI Souad** et **Dr BOUAKSA Fethia**

****

2024-2025

Semestre : 3
Unité d'enseignement : UEM 2.1
Matière : Optimisation
VHS : 37h30 (cours : 1h30, TP : 1h00)
Crédits : 3
Coefficient : 2

Objectifs de l'enseignement

Se familiariser avec les modèles de recherche opérationnelle. Apprendre à formuler et à résoudre les problèmes d'optimisation et maitriser les techniques et les algorithmes appropriés.

Connaissances préalables recommandées

Notions de bases de mathématiques. Algèbre linéaire. Algèbre matricielle.

Contenu de la matière

Chapitre I : Optimisation linéaire

- Formulation générale d'un programme linéaire
- Exemples de programmes linéaires (Problème de production, Problème de Mélange, Problème de découpage, Problème de transport)
- Résolution du problème par la méthode Simplexe :
  Bases et solutions de base des programmeslinéaires
  L'algorithme du simplexe
  Initialisation de l'algorithme du simplexe (la méthode à deux phases).

Chapitre II : Optimisation non- linéaire sans contraintes

- Positivité, Convexité, Minimum
- Gradient et Hessien
- Conditions nécessaires pour un minimum
- Conditions suffisantes pour un minimum
- Méthodes locales
- Méthodes de recherche unidimensionnelle
- Méthodes du gradient
- Méthodes des directions conjuguées
- Méthode de Newton
- Méthodes quasi-Newton

Chapitre III : Optimisation non-linéaires avec contraintes

- Multiplicateurs de Lagrange
- Conditions de Karush-Kuhn-Tucker
- Méthode des pénalités
- Programmation quadratique séquentielle

Chapitre IV : Méthodes d'optimisation stochastiques

- L'algorithme génétique
- La méthode d'essaim particulaire

Table des matières

Avant-propos

L'**optimisation** est une discipline fondamentale des mathématiques appliquées et de l'ingénierie, qui vise à déterminer la meilleure solution possible à un problème donné, selon un ou plusieurs critères définis. Elle intervient dans une multitude de domaines scientifiques, techniques et économiques, dès lors qu'il s'agit de maximiser un gain ou de minimiser un coût, une erreur, un risque ou une consommation de ressources. L'optimisation se trouve ainsi au cœur de nombreuses prises de décision, aussi bien dans l'industrie, les transports, la finance, l'énergie, l'intelligence artificielle ou encore les sciences du vivant.

Historiquement, l'optimisation a émergé avec le calcul différentiel au XVIIe siècle, avec des travaux fondateurs tels que ceux de Fermat et Newton, qui cherchaient à identifier les maxima et minima de fonctions continues. Elle s'est ensuite développée en tant que discipline à part entière, notamment à partir du XXe siècle avec la formalisation de la programmation linéaire, l'essor de la recherche opérationnelle, et les progrès en informatique numérique. Aujourd'hui, l'optimisation est omniprésente dans les algorithmes et les technologies modernes, depuis les systèmes de recommandation jusqu'à la gestion des réseaux électriques intelligents.

Un problème d'optimisation consiste, de manière générale, à rechercher une ou plusieurs **solutions optimales** parmi un ensemble de solutions admissibles, en minimisant ou maximisant une **fonction objectif**, parfois sous **contraintes**. La forme mathématique de ce problème peut varier considérablement selon la nature des variables (réelles, entières, catégorielles), la linéarité des fonctions, la convexité de l'espace de recherche, ou encore la disponibilité d'informations sur le modèle.

Il existe une distinction essentielle entre les **méthodes déterministes**, qui supposent une connaissance complète et exacte du problème à résoudre, et les **méthodes stochastiques**, qui intègrent ou exploitent des aspects aléatoires, soit dans la formulation du problème

(par exemple, données incertaines), soit dans les mécanismes de recherche de solutions (par exemple, choix aléatoires dans les algorithmes).

Dans un monde de plus en plus complexe, dynamique et incertain, les méthodes traditionnelles d'optimisation atteignent leurs limites. C'est dans ce contexte que les **approches stochastiques** prennent toute leur importance, en offrant des solutions plus souples, adaptatives et robustes face à l'imprécision ou à la variabilité des données. Ces méthodes permettent d'explorer des espaces de recherche de grande dimension, non convexes ou discontinus, là où les méthodes classiques échouent.

CHAPITRE I

OPTIMISATION LINÉAIRE

1. Formulation générale d'un problème linéaire

Un programme d'optimisation est dit linéaire si toutes les équations et les inéquations appelées "**contraintes**" sont linéaires (c'est-à-dire que les variables ne sont pas élevées au carré, ne servent pas d'exposant, ne sont pas multipliées entre elles...).

La **programmation linéaire** est une méthode mathématique utilisée pour résoudre des problèmes d'optimisation, dans lesquels on cherche à **optimiser** (**maximiser ou minimiser**) une fonction linéaire (appelée fonction objectif), tout en respectant un ensemble de contraintes linéaires.

Elle s'applique à de nombreux domaines tels que :

- la gestion de la production,
- la logistique et le transport,
- la planification financière,
- l'ingénierie et la gestion des ressources.

L'intérêt de la programmation linéaire réside dans sa capacité à modéliser rigoureusement des situations complexes du monde réel, puis à en extraire une solution optimale à l'aide d'outils mathématiques comme le simplexe ou les méthodes graphiques (dans le cas de deux variables).

Mathématiquement le problème d'optimisation s'écrit sous la forme suivante:

$$\text{Min/max } (Z) = \sum_{i=1}^{n} C_i X_j \qquad (1)$$

$$\underline{\text{Sous contraintes}} \quad (a_{i1}X_1 + a_{i2}X_2 + \ldots\ldots\ldots\ldots + a_{in}X_j) \begin{pmatrix} \leq \\ = \\ \geq \end{pmatrix} (b_i) \qquad (2)$$

$$X_j \geq 0 \qquad (3)$$

$$\{i = 1, \ldots\ldots n\}; \{j = 1, \ldots\ldots\ldots m\}$$

En sachant que:

1) Z: est la fonction à optimiser appelée **fonction coût/économique** ou bien très fréquemment **fonction objectif.**

NB: D'une manière générale, les fonctions objectifs à minimiser correspondent à une énergie fournie, à des risques, des dépenses...; en revanche, les fonctions à maximiser correspondent à un profit, un rendement, des performances...etc.

2) L'ensemble des équations (2) désigne les contraintes, qui traduisent les limites imposées par les ressources disponibles ou les exigences du problème.

Avec:

- ✓ a_{ij}, b_i, C_i : sont des constantes connues.
- ✓ X_j :désigne les inconnus du problème à résoudre appelés variables de décision.
- ✓ La contrainte $X_j \geq 0$ est appelée contrainte de positivité (condition de non négativité).

Définition -1-: On appelle une solution d'optimisation linéaire tout vecteur X qui satisfait toutes les contraintes de l'équation (2).

Une solution est dite **solution réalisable** si elle vérifie les conditions de non négativité, indiquées dans l'équation (3).

Définition -2-: Une solution réalisable est dite **solution optimale** s'il n'existe pas d'autres solutions réalisables qui fournissent une valeur optimale (maximale ou minimale) de la fonction objectif.

Remarque: Dans un problème d'optimisation ayant des solutions réalisables, il se peut que la valeur optimale de la fonction objectif soit infinie. Dans ce cas, on dit que la solution optimale est infinie.

2. Exemples de programmes linéaires

2.1 Problème de production

Une entreprise fabrique deux produits, A et B. Chaque unité de A nécessite 3 heures de machine et 4Kg de matière première. Chaque unité de B nécessite 5 heures de machine et 2Kg de matière première. On dispose de 120 heures de machine et de 100Kg de matière première. Le bénéfice est de 40 UM par unité de A et 60 UM par unité de B.

Le problème consiste à chercher la quantité optimale des deux produits A et B afin de maximiser le bénéfice total de l'entreprise.

Description du problème imposé

Produit	Temps machine (Heures)	Quantité Matière première (kg)	Bénéfice (UM)
A	3	4	40
B	5	2	60
Total	120	100	/

Formulation mathématique du problème

- **Variables de décision**

X= nombre d'unités A à produire

Y= nombre d'unités B à produire

- **Fonction Objectif**

Il s'agit de chercher à optimiser le plan de production de l'entreprise de fabrication afin de maximiser ses bénéfices.

Donc, on écrit: **Maximiser (Z)= 50X+80Y**

- **Les contraintes**

- Contrainte liée au temps de machine $3\mathbf{X+6Y \leq 120}$
- Contrainte liée à la matière première $\mathbf{4X+2Y \leq 100}$
- Contrainte de positivité $\mathbf{X \geq 0}$; $\mathbf{Y \geq 0}$

Résolution graphique du problème

Comme le programme décrit l'exemple ci-dessus ne contient que deux variables de décision, il sera possible de résoudre ce problème via une représentation graphique.

Avant de procéder à la recherche de la solution du problème, on doit étudier les deux premières contraintes dont chacune serve à fournir une droite qui permet la délimitation du domaine des solutions réalisables. Comme décrit la figure montrée ci-après.

Les deux contraintes de non-négativité, $X \geq 0$; $Y \geq 0$ indiquent que cette région se trouve dans le premier quadrant.

On étudie l'équation de la première droite (Δ): **3X+6Y ≤120**

X	**0**	**40**
Y	**20**	**0**

On étudie l'équation de la première droite (Δ'): **4X+2Y ≤100**

X	**0**	**25**
Y	**50**	**0**

La première contrainte est une droite passant par les points (0 ;20) et (40 ;0), tandis que la deuxième passe par (0;50) et (25 ;0). L'intersection de ces deux droites est le point (20 ;10). La région des points satisfaisant les inégalités est la région hachurée sur la figure I.1.

À noter que la solution optimale est l'un des sommets du polyèdre ayant les points (0 ;0), (0 ;20), (20 ;10) et (25 ;0).

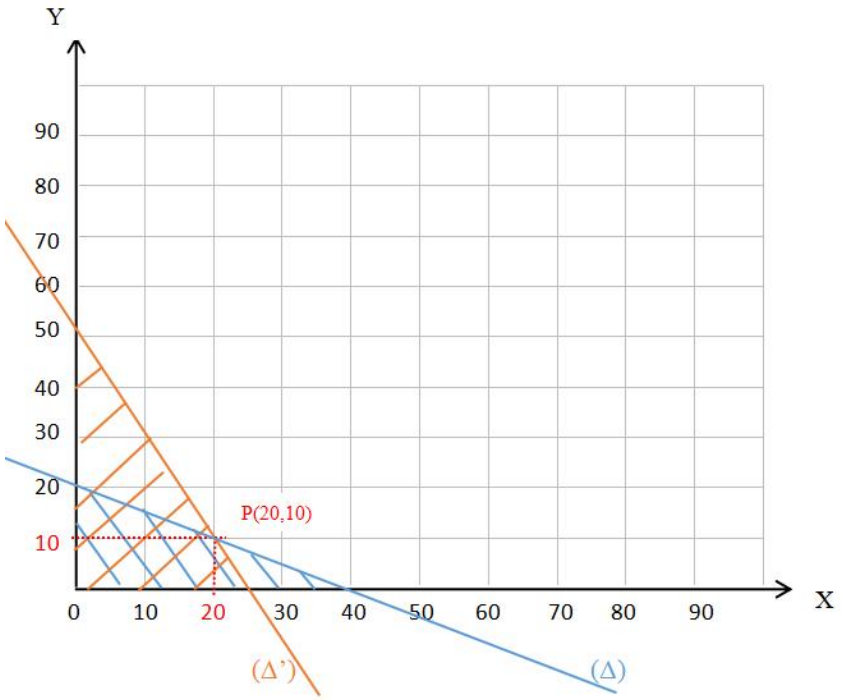


Figure I.1: Représentation graphique du problème

La maximisation de la fonction coût est bien vérifiée en remplaçant les coordonnées du point P(20;10); on aura le **Max(Z)=1800UM**

2.2 Problème de mélange

On veut préparer un mélange gazeux, contenant au moins 40% d'un ingrédient actif. Deux composant bruts sont disponible;

- ✓ Premier composant: Coûte 10UM/Kg et contient 15% d'ingrédient actif.
- ✓ Deuxième composant: Coûte 32UM/Kg et contient 45% d'ingrédient actif.

On cherche à minimiser le coût pour produire 100Kg de mélange.

Formulation mathématique du problème

- **Variables de décision**

X_1= Quantité du premier produit en Kg

X_2= Quantité du deuxième produit en Kg

- **Fonction Objectif**

Il s'agit de chercher à optimiser le plan de confection du mélange en minimisant les dépenses.

Donc, on écrit: **Minimiser (Z)= $10X_1+32X_2$**

- **Les contraintes**

- Contrainte liée au mélange total **$X_1+X_2=100$**
- Contrainte liée à l'ingrédient actif **$(0,15).X_1+(0,45).X_2 \leq (0,4).100$**
- Contrainte de positivité **$X_1 \geq 0$; $X_2 \geq 0$**

Exercice 1

Une entreprise fabrique deux types de parfum, P1 et P2, qui rapportent respectivement 300UM et 500UM par litre. Les parfums sont obtenus à partir de trois types d'essence A, B et C. L'état du stock et les quantités nécessaires à la fabrication d'un litre de chaque parfum sont données dans le tableau ci-dessous :

	Essence de type A (en litres)	Essence de type B (en litres)	Essence de type C (en litres)	Profit (en €/ litre)
Parfum P1	1	0	3	300
Parfum P2	0	2	2	500
Stocks	4	12	18	

1. Modéliser ce problème en programme linéaire.

2. Résoudre ce problème à l'aide du Solver.

Solution

1. Modélisation du problème en programme linéaire

- **Variables de décision**

X_1= Quantité du parfum type P1 à produire

X_2= Quantité du parfum type P2 à produire

Programme d'optimisation

$$\text{Max } (Z)= 300X_1+500X_2$$

$$\text{Sous contraintes } \quad 3.X_1+2.X_2 \leq 18$$

$$X_1 \leq 4$$

$$2.X_2 \leq 12$$

$$X_1 \geq 0 \ ; \ X_2 \geq 0$$

2. Solution numérique du problème

```
>> Z=[-300 -500];
>> A=[3 2;1 0;0 2];
>> b=[18 4 12];
>> Lb=[0 0];
>> [x,Z]=linprog(Z,A,b,[],[],Lb)
Optimization terminated.

x =

    2.0000
    6.0000


Z =

  -3.6000e+03
```

Après exécution du programme sur le code Matlab, on a obtenu les résultats suivants:

- Fonction Objectif à maximiser **Z=3 600UM**
- Variable de décision; **X_1=2L; X_2=6L**

2.3 Problème de découpage

Une usine a reçu des plaques en métal d'une largeur de 200cm et d'une longueur de 500cm. Il faut en fabriquer au moins 30 plaques de largeur de 110cm, 40 plaques de 75cm de largeur et 15 plaques de largeur de 60cm.

- Donner le modèle mathématique permettant de minimiser la quantités des déchets dégagés de cette opération?

Formulation mathématique du problème

- **Variables de décision**

Une plaque de 200cm de largeur peut être découpée en plusieurs façons:

Variante de découpage	Quantité de déchets dégagés (cm)	Variable de décision
01 plaque de 75cm + 02 plaques de 60cm	5cm	X_1
01 plaque de 110cm + 01 plaques de 75cm	15cm	X_2
01 plaque de 110cm + 01 plaques de 60cm	30cm	X_3
03 plaques de 60cm	20cm	X_4
02 plaques de 75cm	50cm	X_5

Variables : X_1, X_2 ,X_3,X_4 et X_5 sont les nombres de plaques a découper par la 1ère , 2ème ,3ème, 4ème et la 5ème façons respectivement.

- **Fonction objectif à minimiser** (correspond déchet total)**:**

$$\textbf{Min } Z = 5X_1 + 15X_2 + 30X_3 + 20X_4 + 50X_5$$

- **Avec les contraintes**

➢ Plaques de largeur 110cm $\quad X_2 + X_3 \geq 30$

➢ Plaques de largeur 75cm $\quad X_1 + X_2 + 2X_5 \geq 40$

➢ Plaques de largeur 60cm $\quad 2X_1 + X_3 + 3X_4 \geq 15$

➢ Positivité des variables $\quad X_1 ; X_2 ; X_3 ; X_4 ; X_5 \geq 0$

2.4 Problème de transport

Il s'agit de type de problème que l'on peut résoudre par la programmation linéaire, Il se définit comme suit.

Connaissant les quantités disponibles de chacune des unités de production, les quantités requises aux points de distribution et le coût de transport d'un bien d'une usine vers un point de vente, il s'agit de déterminer le plan de transport optimal, c'est-à-dire de déterminer les quantités de biens que chaque usine va envoyer vers chacun des points de vente afin que le coût de transport total soit minimum. On suppose qu'il est possible d'expédier des produits de n'importe quelle origine vers n'importe quelle destination.

Application 1

Une organisation possède quatre centres de distribution : (Tlemcen, Alger, Constantine et Béchar) de stocks de produits respectivement de : 120 kg, 100 kg, 100kg et 100 kg, pour lesquels elle a reçu des commandes de ses antennes d'Oran (80 kg), Bejaia (190 kg) et Tamanrasset (150 kg).

Les coûts de transport d'un kilo de produits, suivant les liaisons routières considérées, sont donnés par le tableau suivant :

	Oran	**Bejaia**	**Tamanrasset**
Tlemcen	80DA	200DA	-
Alger	100DA	100DA	-
Constantine	-	400DA	900DA
Bechar	-	600DA	800DA

Ce problème peut se formuler sous forme d'un programme linéaire. On note par C_{ij} le coût de transport de produit de l'origine **i** vers la destination **j**.

C_{11}= 80, C_{12}=200, C_{13}=0 ;

C_{21}=100, C_{22}= 100, C_{23}=0 ;

C_{31}= 0 , C_{32}=400, C_{33}=900 ;

C_{41}=0, C_{42}= 600, C_{43}= 800.

On note par a_i la quantité du produit disponible au niveau des origines **i** et b_j la quantité à recevoir au niveau des destination **j**.

A noter que la quantité $a_1+a_2+a_3+a_4 \geq b_1+b_2+b_3$. Dans le cas contraire, le problème n'a pas de solutions réalisables.

On note par X_{ij} la quantité du produit à expédier de l'origine **i** vers une destination **j**.

Donc la fonction coût à minimiser

$$Z=80X_{11}+100X_{12}+100X_{21}+100X_{22}+400X_{32}+900X_{33}+600X_{42}+800X_{43}$$

On peut également décrire ce problème à travers le schéma de transport suivant:

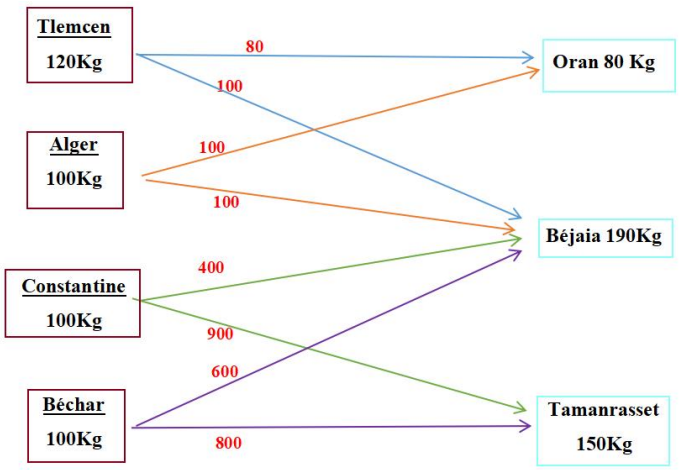


Figure I.2: Schéma du plan de transport

Formulation mathématique du problème

- **Fonction Objectif**

Il s'agit d'un problème de transport, la fonction économique doit être minimisée:

$$\textbf{Min } (Z)=80X_{11}+100X_{12}+100X_{21}+100X_{22}+400X_{32}+900X_{33}+600X_{42}+800X_{43}$$

- **Sous Contraintes:**

$$X_{11}+X_{12} \leq 120$$
$$X_{21}+X_{22} \leq 100$$
$$X_{32}+X_{33} \leq 100$$
$$X_{42}+X_{43} \leq 100$$
$$X_{11}+X_{21}=80$$
$$X_{12}+X_{22}+X_{32}+X_{42}=190$$
$$X_{33}+X_{43}=150$$

Avec $X_{ij} \geq 0$; i=1,2,3,4 ; j=1,2,3

3. Résolution du problème par la méthode Simplexe :

La méthode de simplexe est un outil principal pour la résolution des problèmes de programmation linéaire.C'est une méthode algébrique itérative qui permet de trouver la solution exacte d'un problème de programmation linéaire en un nombre fini d'étapes.

La forme matricielle permet de représenter un problème de programmation linéaire sous une forme plus concise.

$$\text{Max } (Z) = C.X \qquad (4)$$

$$A.X \begin{pmatrix} \leq \\ = \\ \geq \end{pmatrix} b \qquad (5)$$

$$\text{avec } X \geq 0 \qquad (6)$$

Où **C** est un vecteur-ligne de dimension (1 × n), x un vecteur-colonne de dimension (n×1), **A** une matrice de dimension (m×n), **b** un vecteur-colonne de dimension (m× 1) et 0 le vecteur nul à n composantes.

Un problème de programmation linéaire peut se présenter sous différentes formes:

3.1 Forme canonique

Si la fonction objectif doit être maximisée et si toutes les contraintes sont sous formes des inéquations de type inférieures ou égales (≤), on peut dire que le programme linéaire se présente sous une **forme canonique.**

$$\text{Max } (Z) = C.X \qquad (7)$$

$$A.X \leq b \qquad (8)$$

$$\text{avec } X \geq 0 \qquad (9)$$

NB: toute contrainte peut être transformé sous forme canonique.

Application 2

Écrire le programme linéaire suivant sous la forme canonique;

Min (Z) = $5X_1+10X_2-X_3$

Sous contraintes $X_1-8X_2-2X_3 \leq 25$

$-3X_1+X_2+7X_3=65$

$5X_1-10X_2-X_3 \geq 45$

$X_1;X_2;X_3 \geq 0$

Solution

1ère transformation

Max (-Z)=$-5X_1-10X_2+X_3$

Sous contraintes $X_1-8X_2-2X_3\leq 25$

$-3X_1+X_2+7X_3\leq 65$

$-3X_1+X_2+7X_3\geq 65$ →

$5X_1-10X_2-X_3\geq 45$

$X_1;X_2;X_3\geq 0$

2ème transformation

Max (-Z)=$-5X_1-10X_2+X_3$

Sous contraintes $X_1-8X_2-2X_3\leq 25$

$-3X_1+X_2+7X_3\leq 65$

$3X_1-X_2-7X_3\leq -65$

$-5X_1+10X_2+X_3\leq -45$

$X_1;X_2;X_3\geq 0$

Forme canonique

3.2 Forme standard

Si la fonction objectif doit être maximisée et si toutes les contraintes sont sous formes des équations (=), on peut dire que le programme linéaire se présente sous une **forme standard.**

$$\text{Max } (Z) = C.X \qquad (10)$$

$$A.X = b \qquad (11)$$

$$\text{avec } X \geq 0 \qquad (12)$$

NB: toute contrainte peut être transformée sous forme standard.

La méthode de simplexe. exige que le programme linéaire soit sous forme standard. Pour cette raison, il faut transformer les inégalités rencontrées en égalités. Cette transformation se fait simplement en introduisant des variables non-négatives (qui vérifient les contraintes de non-négativité) appelées **variables d'écart**.

- **Si les contraintes sont du type : $a_{i1}X_1+a_{i2}X_2+.....+a_{in}X_n \leq b_i$**

On introduit une nouvelle variable positive $X_{n+1}\geq 0$ et on écrit :

$a_{i1}X_1+a_{i2}X_2+.....+a_{in}X_n+ X_{n+1} = b_i$

- **Si les contraintes sont du type : $a_{i1}X_1+a_{i2}X_2+.....+a_{in}X_n \geq b_i$**

On introduit une nouvelle variable positive $X_{n+1}\geq 0$ et on écrit :

$a_{i1}X_1+a_{i2}X_2+.....+a_{in}X_n -X_{n+1} = b_i$

Application 3

Écrire le programme linéaire suivant sous la forme standard;

Min (Z) = $5X_1+10X_2-X_3$

Sous contraintes $X_1-8X_2-2X_3 \leq 25$

$-3X_1+X_2+7X_3=65$

$5X_1-10X_2-X_3 \geq 45$

$X_1;X_2;X_3 \geq 0$

Solution

Max (-Z) = $-5X_1-10X_2+X_3+0.X_4+0.X_5$

Sous contraintes $X_1-8X_2-2X_3+X4 =25$

$-3X_1+X_2+7X_3=65$

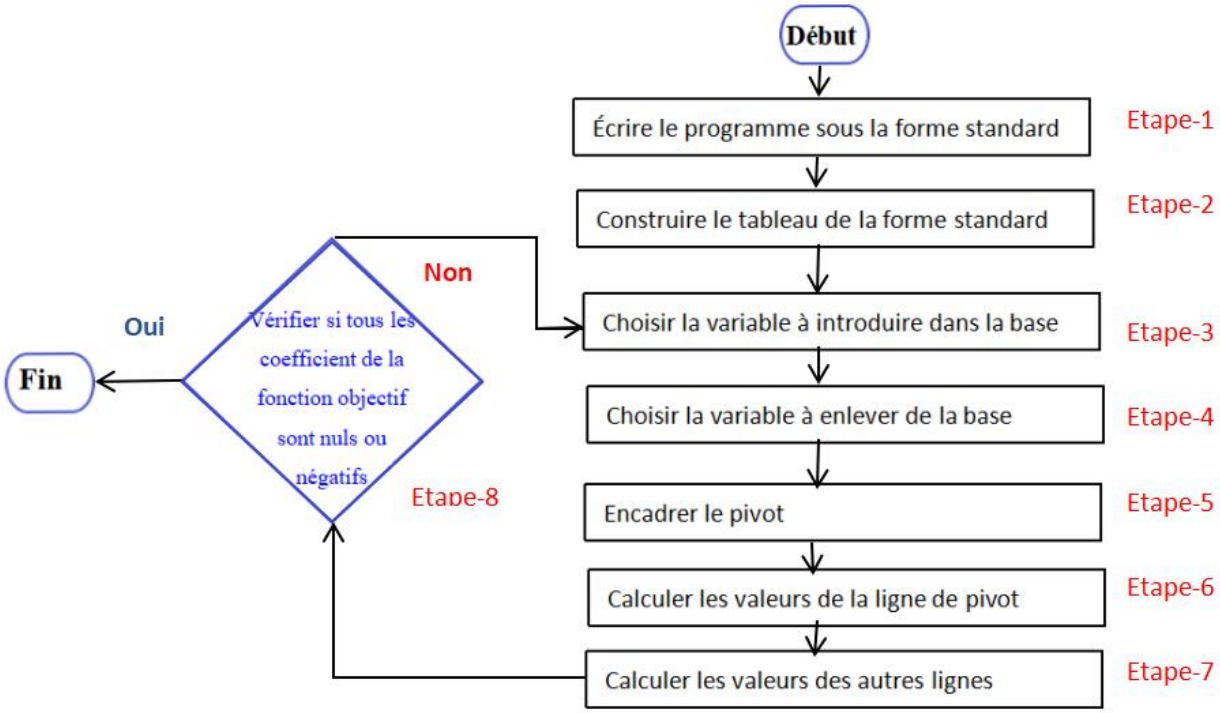
$5X_1-10X_2-X_3-X_5=45$

$X_1;X_2;X_3 \geq 0$

Forma Standard

3.3 L'algorithme du simplexe

Les étapes de résolution des programmes d'optimisation linéaire sont résumées dans l'organigramme suivant:



Remarques

- Il se peut que la solution optimale d'un problème de programmation linéaire possédant une solution réalisable de base soit infinie.

- L'algorithme de Simplexe est implémenté sur la minimisation, c'est pour cette raison qu'on doit inverser le signe de la solution obtenue à la fin lorsqu'il s'agit d'une maximisation.

Application 4

On considère le problème d'optimisation suivant:

$Max(Z) = 30X_1+50X_2$
Sous contraintes $3X_1+2X_2 \leq 1800$
$X_1 \leq 400$
$X_2 \leq 600$
$X_1;X_2 \geq 0$

Solution

Étape 1: Écrire le programme d'optimisation sous forme standard

$Max(Z) = 30X_1+50X_2+0.X_3+0.X_4+0.X_5$
Sous contraintes $3X_1+2X_2+X_3=1800$
$X_1+X_4=400$
$X_2+X_5=600$
$X_1;X_2;X_3;X_4;X_5 \geq 0$

Étape 2: Construire le tableau de la forme standard

Variables hors base

La base		X1	X2	X3	X4	X5	bi	
	X3	3	2	1	0	0	1800	900
	X4	1	0	0	1	0	400	+∞
	X5	0	1	0	0	1	600	600
	Z	30	50	0	0	0	0	0

Le tableau est de dimension (LixCj) tel que i désigne le nombre de lignes de 1 à 4 et j désigne le nombre de colonnes de 1 à 6.

Étape 3: Choisir la variable à introduire dans la base

On choisit la variable hors base ayant le plus grand coefficient positif de la ligne Z.

Dans ce cas 50, donc on va choisir la variable X_2 et par conséquence la colonne pivot Cj=2

Étape 4: Choisir la variable à enlever de la base

On doit choisir la variable de base qui correspond au rapport bi/Ec le plus faible positif .

Il faut diviser la colonne des solutions (bi) par les éléments de la colonne pivot (Cj=2).

Dans cet exemple; le rapport le plus faible positif égal à 600, donc on va sélectionner la variable de base X_5, et par conséquence la ligne pivot Li=3

Étape 5: Encadrer le pivot

L'intersection de la ligne pivot et la colonne pivot nous donne la valeur de pivot =1 et sa localisation $\mathbf{E_{ij}=E_{LC}=E_{32}}$

Étape 6: Calcul des valeurs de la ligne de pivot

On divise la ligne de pivot sur le pivot c'est à dire qu'on utilise tout simplement la formule $\mathbf{E_{ij}=E_{ij}/E_{LC}}$

0	1	0	0	1	600

Étape 7: Calcul des valeurs des éléments des autres lignes

On applique la formule $E_{ij}=E_{ij}-(E_{iC}/E_{LC}).E_{Lj}$

Après calculs de tous les élément on doit dresser le nouveau tableau.

		X1	X2	X3	X4	X5	bi	
	X3	3	0	1	0	-2	600	200
	X4	1	0	0	1	0	400	400
X2	X5	0	1	0	0	1	600	+∞
	Z	30	0	0	0	0	-30000	-1000

Étape 8: On vérifie si les coefficients de la ligne Z sont tous nul ou négatifs? **Non**

Il existe un coefficient =30>0; On doit retourner depuis l'étape 3.

Étape 3': Choisir la variable à introduire dans la base

On choisit la variable hors base ayant le plus grand coefficient positif de la ligne Z.

Dans ce cas 30, donc on va choisir la variable X_1 et par conséquence la colonne pivot Cj=1

Étape 4: Choisir la variable à enlever de la base

On doit choisir la variable de base qui correspond au rapport bi/Ec le plus faible positif .

Dans ce cas, le rapport le plus faible positif égal à 200, donc on va sélectionner la variable de base X_3, et par conséquence la ligne pivot Li=1

Étape 5: Encadrer le pivot

L'intersection de la ligne pivot et la colonne pivot nous donne la valeur de pivot =3 et sa localisation $\mathbf{E_{ij}=E_{LC}=E_{11}}$

Étape 6: Calcul des valeurs de la ligne de pivot

On divise la ligne de pivot sur le pivot

0	1	1/3	0	-2/3	200

Étape 7: Calcul des valeurs des éléments des autres lignes

On applique la formule $E_{ij}=E_{ij}-(E_{iC}/E_{LC}).E_{Lj}$

Après calculs de tous les élément on doit dresser le nouveau tableau.

		X1	X2	X3	X4	X5	bi
X1	X3	0	1	1/3	0	-2/3	200
	X4	0	0	-1/3	1	2/3	200
X2	X5	0	1	0	0	1	600
	Z	0	0	-10	0	-30	-36000

Étape 8: On vérifie si les coefficients de la ligne Z sont tous nul ou négatifs? **OUI**

Le critère d'arrêt est bien vérifié, d'où la solution optimale **Z=-(-36000)** avec $\mathbf{X_1=200}$ et $\mathbf{X_2=600}$.

CHAPITRE II

OPTIMISATION NON LINÉAIRE SANS CONTRAINTES

1. Notions de base

L'optimisation non linéaire sans contraintes est une branche fondamentale de l'optimisation mathématique qui vise à trouver les extrema (minima ou maxima) d'une fonction réelle de plusieurs variables, sans qu'aucune contrainte explicite ne limite l'espace des solutions. Contrairement à l'optimisation linéaire, où les fonctions objectif et les contraintes sont linéaires, l'optimisation non linéaire considère des fonctions plus générales, souvent différentiables, dont la complexité peut refléter des phénomènes réels dans des domaines variés comme l'économie, l'ingénierie, ou encore l'intelligence artificielle.

Un problème d'optimisation non linéaire sans contraintes peut se présenter sous la forme; **Min f(x)**

$$x \in R^n$$

1.1 Gradient

Soit f une fonction de $\mathbf{R^n}$ dans **R** admettant en point **x** des dérivées partielles de premier ordre.

On appelle **∇f(x) gradient** de la fonction f au point x, le vecteur-colonne:

$$\nabla f(x)=\left(\frac{\partial f}{\partial x_1}(x);.....;\frac{\partial f}{\partial x_n}(x)\right)^t \qquad (1)$$

Si la fonction f admet en x^0 des dérivées partielles continues, on applique la formule de Taylor à l'ordre 1.

$$f(x)=f(x^0)+(x-x^0)^t .\nabla f(x^0)+\|x-x^0\| .\varepsilon(x) \qquad (2)$$

$\varepsilon \to 0$ lorsque $x \to x^0$

Le gradient joue un rôle très important dans le développement et l'analyse des algorithmes d'optimisation. Dans l'espace (milieu à trois dimensions), on écrit:

$$\nabla f(x,y,z)=\left(\frac{\partial f}{\partial x}(x,y,z)\vec{i}+\frac{\partial f}{\partial y}(x,y,z)\vec{j}+\frac{\partial f}{\partial z}(x,y,z)\vec{k}\right) \qquad (3)$$

Application 01

Soit la fonction **f** de $\mathbf{R^2}$ dans **R** définie par :

$$f(x,y) = x^2 + y^2 - 3xy + 4y$$

1. Déterminer l'expression du gradient de la fonction

Solution

$$\nabla f(x,y)=\begin{bmatrix} 2x-3y \\ 2y-3x+4 \end{bmatrix}$$

1.2 Hessien

Si f une fonction de $\mathbf{R^n}$ dans **R** admettant en point **x** des dérivées partielles de second ordre, on pose: $\nabla^2 f(x)=\nabla(\nabla f(x))^t$ (4)

On écrit:

$$\nabla^2 f(x) = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2}(x) & \dots & \frac{\partial^2 f}{\partial x_1 \partial x_n}(x) \\ \frac{\partial^2 f}{\partial x_2 \partial x_1}(x) & \dots & \frac{\partial^2 f}{\partial x_2 \partial x_n}(x) \\ \dots & \dots & \dots \\ \frac{\partial^2 f}{\partial x_n \partial x_1}(x) & \dots & \frac{\partial^2 f}{\partial x_n^2}(x) \end{pmatrix} \quad (5)$$

$\nabla^2 f(x)=\nabla(\nabla f(x))^t = H(x)$ appelé Hessien de la fonction f.

- Si f est une fonction de classe C^2 (admet des dérivées partielles d'ordre 2 continues), le Hessien de la fonction est une matrice **symétrique**

$$\frac{\partial^2 f}{\partial x_i x_j}(x) = \frac{\partial^2 f}{\partial x_j x_i}(x) \quad (6)$$

- Si f est une fonction de classe C^2 en x^0 on écrit la formule de Taylor à l'ordre 2.

$$f(x)=f(x^0)+ (x-x^0)^t .\nabla f(x^0)+\frac{1}{2}(x-x^0)^t .\nabla^2 f(x^0)(x-x^0)+\|x - x^0\|^2.\varepsilon(x) \quad (7)$$

$\varepsilon \to 0$ lorsque $x \to x^0$

Application 02

Soit la fonction **f** de $\mathbf{R^2}$ dans **R** définie par :

$$f(x,y) = x^2 + y^2 - 3xy + 4y$$

- Déterminer le Hessien de la fonction

Solution

$$H(x)=\begin{bmatrix} 2 & -3 \\ -3 & 2 \end{bmatrix}$$

- Positivité

➢ Une matrice **A** se définie **positive**: $\mathbf{A=A^T}$ **>0** si et seulement si $\mathbf{X^TA.X>0}$ $\forall X\neq 0$

➢ Une matrice **A** se définie **semi positive**: $\mathbf{A=A^T \geq 0}$ si et seulement si $\mathbf{X^TA.X\geq 0}$ $\forall X\neq 0$

1.3 Fonction quadratiques

Soit A une matrice symétrique d'ordre n, b un vecteur colonne d'ordre n et c un nombre réel.

L'application f de $\mathbf{R^n}$ dans **R** se définie par

$$\mathbf{f(x) = \frac{1}{2}X^TAX + b^TX + C} \qquad (8)$$

Cette application est appelée **fonction quadratique**.

Application 03

Soit la fonction **f** de $\mathbf{R^2}$ dans **R** définie par : $f(x_1, x_2) = 2x_1^2 + x_2^2 + 2x_1 + 3x_2$

Écrire f sous la forme $f(x) = \frac{1}{2}X^TAX + b^TX + C$

Solution

$$A = \begin{bmatrix} 4 & 0 \\ 0 & 2 \end{bmatrix},\ b=\begin{pmatrix} 2 \\ 3 \end{pmatrix},\ C=0$$

1.4 Convexité

La convexité est à la base une propriété géométrique, assez intuitive d'ailleurs, qui permet de caractériser certains objets. On voit assez bien ce qu'est un objet convexe dans un espace à deux ou trois dimensions.

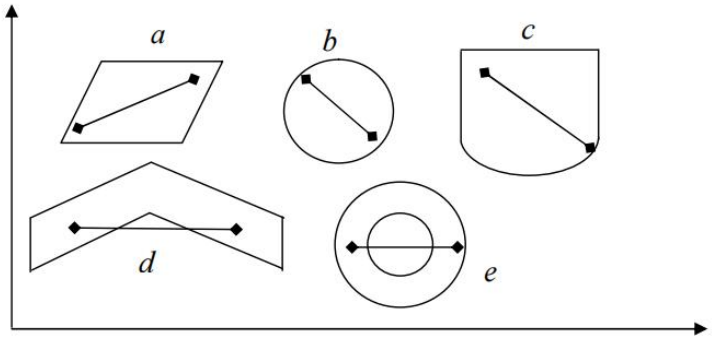


Figure II.1: Exemples des objets convexes et non convexes

Les ensembles a , b , c sont des ensembles convexes ; d , e sont des ensembles non convexes.

- Un ensemble **D** est dit **convexe** si, pour tout point a et b dans **D**, les segment [ab] est inclus dans **D**.

$$\forall t \in [0,1], \text{ le point } t.a+(1-t)b \in \mathbf{D} \qquad (9)$$

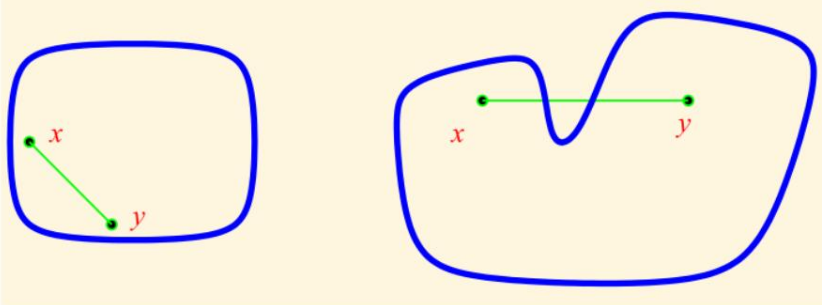


Figure II.2: Exemples des ensembles convexes et non convexes

- Une fonction f de $\mathbf{R}^n$ dans **R** définie sur un ensemble convexe **D** est dite **convexe** si, pour tout point a et b $\in R^n$ et tout t $\in [0,1]$.

$$f(t.a+(1-t)b) \leq t.f(a)+(1-t)f(b) \qquad (10)$$

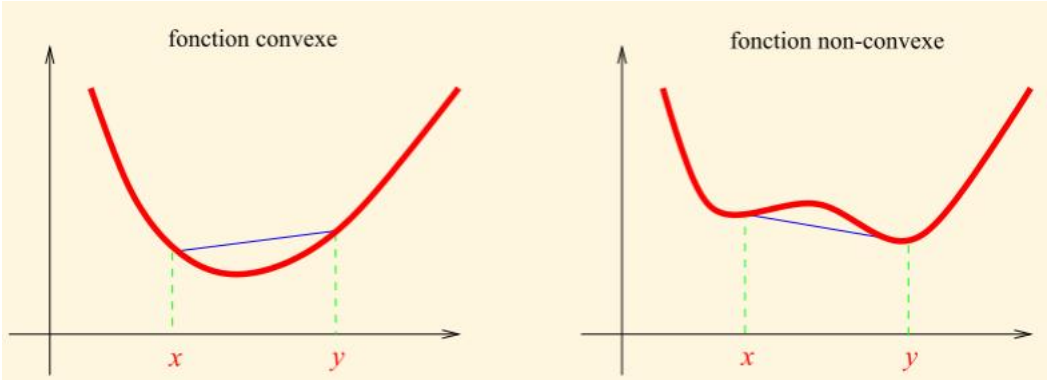


Figure II.3: Exemples de fonctions convexes et non convexes

Remarque : **F** est concave si $-\mathbf{F}$ est convexe.

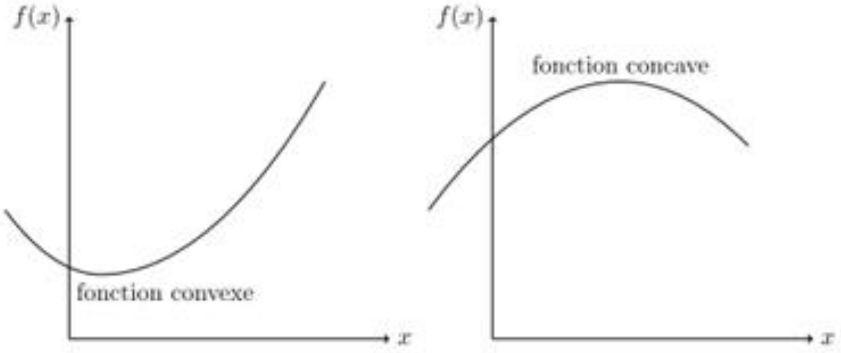


Figure II.4: Fonctions convexes et concave

- **Tests de convexité et de concavité**

➢ Soit la fonction f est deux fois dérivable d'une seule variable. Alors :

1. f est convexe si et seulement si : $\forall x, \frac{d^2 f(x)}{dx^2} \geq 0$ (11)
2. f est strictement convexe si : $\forall x, \frac{d^2 f(x)}{dx^2} > 0$ (12)
3. f est concave si et seulement si : $\forall x, \frac{d^2 f(x)}{dx^2} \leq 0$ (13)
4. f est strictement concave si : $\forall x, \frac{d^2 f(x)}{dx^2} < 0$ (14)

➢ Soit la fonction f est deux fois dérivable à deux variables. Dans ce cas, on utilise la Table suivante comme conditions nécessaires.

	Convexe	Strict. convexe	Concave	Strict. concave
$\frac{\partial^2 f(x_1,x_2)}{\partial x_1^2}$	≥ 0	> 0	≤ 0	< 0
$\frac{\partial^2 f(x_1,x_2)}{\partial x_2^2}$	≥ 0	> 0	≤ 0	< 0
$\frac{\partial^2 f(x_1,x_2)}{\partial x_1^2}\frac{\partial^2 f(x_1,x_2)}{\partial x_2^2} - \left(\frac{\partial^2 f(x_1,x_2)}{\partial x_1 x_2}\right)^2$	≥ 0	> 0	≥ 0	> 0

Tableau II.1: Table de test de convexité des fonctions à deux variables

Ces tests se généralisent à des fonctions de plus de deux variables en utilisant le Hessien (la matrice des dérivées secondes). Deux résultats supplémentaires sont utiles pour tester la convexité :

– Une fonction qui s'exprime comme la somme de fonctions convexes (concaves) est convexe (concave) ;

– L'opposé d'une fonction concave est convexe et vice-versa.

Application 04

Tester la convexité des fonctions suivantes

- $f(x) = 7x - x^2$
- $f(x) = \frac{1}{6}x^4 + x - 3$
- $f(x) = x^5 - 3x^4$
- $f(x_1, x_2) = x_1^2 - x_1 x_2 + x_2^2$
- $f(x_1, x_2) = x_1^2 - 2x_1 x_2 + x_2^4$

Solution

- $f(x) = 7x - x^2$ est concave, $f''(x) = -2 < 0$
- $f(x) = \frac{1}{6}x^4 + x - 3$ est convexe, $f''(x) = 2x^2 > 0$
- $f(x) = x^5 - 3x^4$ n'est ni convexe, ni concave. $f''(x) = 20x^3 - 36x^2$ vaut $-16 < 0$ en $x = 1$ et $+16 > 0$ en $x = 2$
- $f(x_1, x_2) = x_1^2 - x_1x_2 + x_2^2$ est convexe : $\nabla^2 f(x_1, x_2) = \begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix}$ donc $(a, b)\nabla^2 f(x_1, x_2) \begin{pmatrix} a \\ b \end{pmatrix} = 2a^2 - 2ab + 2b^2 = (a - b)^2 + a^2 + b^2$
- $f(x_1, x_2) = x_1^2 - 2x_1x_2 + x_2^4$ n'est ni convexe ni concave : $(a, b)\nabla^2 f(x_1, x_2) \begin{pmatrix} a \\ b \end{pmatrix} = 2a^2 - 4ab + 12b^2x_2^2$ est positif quand $(x_2, a, b) = (1, 1, 1)$ et négatif quand $(x_2, a, b) = (0, 1, 1)$.

1.5 Minimum

a) Minimum global

Soit f une fonction de R^n dans R, f(x*) est le minimum global de f si et seulement si:

$$\forall x \in R^n \cdot \mathbf{f(x) \geq f(x^*)} \qquad (15)$$

x* est minimiseur global de f.

b) Minimum local

Soit f une fonction de R^n dans R, f(x*) est le minimum local de f si et seulement si:

$$\exists\, \varepsilon, \forall x \in]x^* - \varepsilon, x^* + \varepsilon[, \mathbf{f(x) \geq f(x^*)} \qquad (16)$$

x* est minimiseur local de f.

c) Point stationnaire

X_0 est un point stationnaire si et seulement si $\nabla f(x_0) = 0$

Dans les points stationnaires, il y a les minima et les maxima (locaux et globaux) et il existe aussi des d'autres points.

- Un point stationnaire qui ne représente ni un minimum ni maximum de la fonction est un point singulier.

d) Minimum de fonctions convexes

Théorème : Soit : $\Re^n \rightarrow \Re$ une fonction convexe possédant des dérivées partielles. f admet un minimum globale en a si et seulement si $\nabla f(a) = 0_{\Re}$.

Corollaire : Si f est convexe, tout minimum local est un minimum global.

e) Condition nécessaire pour l'existence d'un minimum local

a est un minimum local de la fonction f si et seulement si, il existe au voisinage Va de a tel que pour tout $x \in Va$, $f(a) \leq f(x)$.

Soit :f de $\Re^n \rightarrow \Re$ de classe C^2 si f possède un minimum local en a, alors:

$\nabla f(a) = 0_{\Re}$.

$\nabla^2 f(a)$ est une matrice définie positive.

f) Condition suffisante pour l'existence d'un minimum local

Soit :f de $\Re^n \rightarrow \Re$ de classe C^2 si

$\nabla f(a) = 0_{\Re}$.

$\nabla^2 f(a)$ est une matrice définie positive, alorspossède un minimum local en a

2. Méthodes d'optimisation non linéaire sans contraintes

Pour déterminer un point où une fonction f atteint un minimum local, les méthodes consistent à construire une suite $x_0, x_1, \ldots, x_k, \ldots$ qui doit converger vers un point x* vérifiant une condition nécessaire d'optimalité. Cette condition (par exemple la condition du premier ordre, $\nabla f(x^*) = 0$) n'est en général pas suffisante et le comportement de f au voisinage de x* doit donc faire l'objet d'une étude supplémentaire (pouvant porter entre autres sur le Hessien de f en x*).

2.1 Méthodes de recherche unidimensionnelle

Soit f une application de $\Re$ dans $\Re$. Les méthodes d'optimisation de telles fonctions ont d'autant plus d'importance qu'elles servent d'outils pour l'optimisation de fonctions de plusieurs variables.

2.1.1 Méthode de dichotomie «1er ordre»

Une fonction est dite unimodale s'il existe un réel x* pour lequel la fonction est strictement décroissante sur $]-\infty, x^*]$ et strictement croissante sur $[x^*; +\infty[$.

Le point x* est le minimum global.

- Algorithme de recherche dichothomique

f est une fonction unimodale de classe C^1 sur [a,b] telle que: $f'(a)<0<f'(b)$

1. Initialisation

Choix de [a,b] tel que $f'(a)<0<f'(b)$

2. Critère d'arrêt

Si $\|b - a\| < \varepsilon$, STOP

3. Si non, on pose $m=\frac{a+b}{2}$

4. f'(m)≥0 alors b=m, sinon a=m

On retourne à l'étape **2**

2.1.2 Méthode de Newton «Second ordre»

A partir d'un point x_k, on approche f par le développement de Taylor:

$$g(x)=f(x^k)+ (x-x^k) .\nabla f(x^k)+\frac{1}{2}(x-x^0) .\nabla^2 f(x^k)(x-x^k) \qquad (17)$$

$$g'(x)=\nabla f(x^k)+(x-x^0) .\nabla^2 f(x^k)(x-x^k) \qquad (18)$$

Si $f''(x_k)>0$ (cas où f est convexe autour de x_k), on pose: $x_{k+1}=x_k - \frac{f'(X_k)}{f''(x_k)}$, où $g'(x_{k+1})=0$ (g atteint son minimum autour de x_k)

Si $f''(x_k)\leq 0$ (la méthode échoue)

➢ Algorithme de Newton unidimensionnel

f est une fonction de classe C^2 et x_0 est un point dans le voisinage d'un minimiseur de f.

1. Initialisation k=0

Choix de $x_0 \in R$ dans un voisinage de x*

2. Itération k

$$\mathbf{x_{k+1}=x_k - \frac{f'(X_k)}{f''(x_k)}},$$

3. Critère d'arrêt

Si $\|x_{k+1} - x_k\| < \varepsilon$, STOP

4. Si non, on pose k=k+1, On retourne à l'étape **2**

D'un point de vue pratique, cette méthode souffre de nombreux inconvénients :

- ✓ la méthode peut diverger si le point de départ est trop éloigné de la solution.
- ✓ la méthode n'est pas définie si $\boldsymbol{f''(x_k)= 0.}$
- ✓ la méthode peut converger indifféremment vers un minimum, un maximum ou un point de selle.

Cette méthode est donc peu utilisée. Son intérêt majeur est d'illustrer dans $\Re$ le fonctionnement des algorithmes de minimisation multidimensionnelle du second ordre.

2.2 Méthodes du gradient

Il s'agit d'une famille de méthodes itératives qui s'appliquent à des fonctions dérivables. Ces méthodes sont appelées aussi les méthodes directionnelles ou méthodes de descentes.

On veut minimiser une fonction f. Pour cela on se donne un point de départ arbitraire x_0. Pour construire l'itéré suivant x_1 il faut penser qu'on veut se rapprocher du minimum de f; on veut donc que $f(x_1) < f(x_0)$. On cherche alors x_1 sous la forme

$$x_1 = x_0 + \rho_1 d_1$$

où d_1 est un vecteur non nul de $\Re$

et ρ_1 est un réel strictement positif. En pratique donc, on cherche d_1 et pour que $f(x_0+\rho_1 d_1) < f(x_0)$. On ne peut pas toujours trouver d_1. Quand d_1 existe on dit que c'est **une direction de descente** et ρ_1 est **le pas de descente**. La direction et le pas de descente peuvent être fixes ou variables à chaque itération. Le schéma général d'une méthode de descente est le suivant :

$$x_0 \in \Re^n \quad donnée$$
$$x_{k+1} = x_k + \rho_k d_k \quad , d_k \in \Re^n - \quad 0 \ , \rho_k \in \Re^{+*}$$

Ou ρ_k et d_k sont choisis de telle sorte que $f(x_k + \rho_k d_k) < f(x_k)$

De tels algorithmes sont souvent appelés méthodes de descente. Essentiellement, la différence entre ces algorithmes réside dans le choix de la direction de descente d_k.

Comme on veut $f(x_1+ \rho_1 d_1)< f(x_1)$, on peut choisir en première approximation

$d_k = -\nabla f(x_k)$. La méthode ainsi obtenue s'appelle l'algorithme du Gradient. Le pas est choisi constant ou variable.

- Algorithme du gradient

Initialisation k=0

Choix de $x_0 \in R$ et $\rho_0>0$

Itération k

$$\mathbf{x_{k+1}=x_k + \rho_k d_k} \ ,$$

Critère d'arrêt

Si $\|x_{k+1} - x_k\| <\varepsilon$, STOP

Si non, on pose k=k+1, On retourne à l'étape **2**

2.2.1 Méthode du gradient à pas constant

On utilise le plus souvent la méthode du gradient à pas constant (ρ=constant).Toutefois, on peut faire varier le pas à chaque itération : on obtient alors la méthode du gradient à pas variable.

Application 05

Soit la fonction $f\ x\ = x_1^2 + \frac{1}{2}x_2^2 - 3(x_1 + x_2)$ trouver le minimum de cette fonction en partant du point de cordonnée $x_0 = (-2,1.5)$; critère d'arrêt est $\|x_{k+1} - x_k\| < 10^{-3}$

Solution

1- $x_0 = (-2,1.5) \rho = 0.45$

2- Calcul de la dérivée par rapport à x_1

$$\frac{\partial f}{x_1} = 2x_1 - 3$$

- Calcul de la dérivée par rapport à x_2

$$\frac{\partial f}{x_1} = x_2 - 3$$

- 1ère itération

$$x_{1\,1} = x_{1\,0} + \rho\ \nabla f(x_1)$$
$$x_{2\,1} = x_{2\,0} + \rho\ \nabla f(x_1)$$

3- Test d'arrêt *Si* $x_{1\,1} - x_{1\,0} < 10^{-3}, STOP$
$x_{2\,1} - x_{2\,0} < 10^{-3}, STOP$

Si non on pose k=2 et on retourne à 2

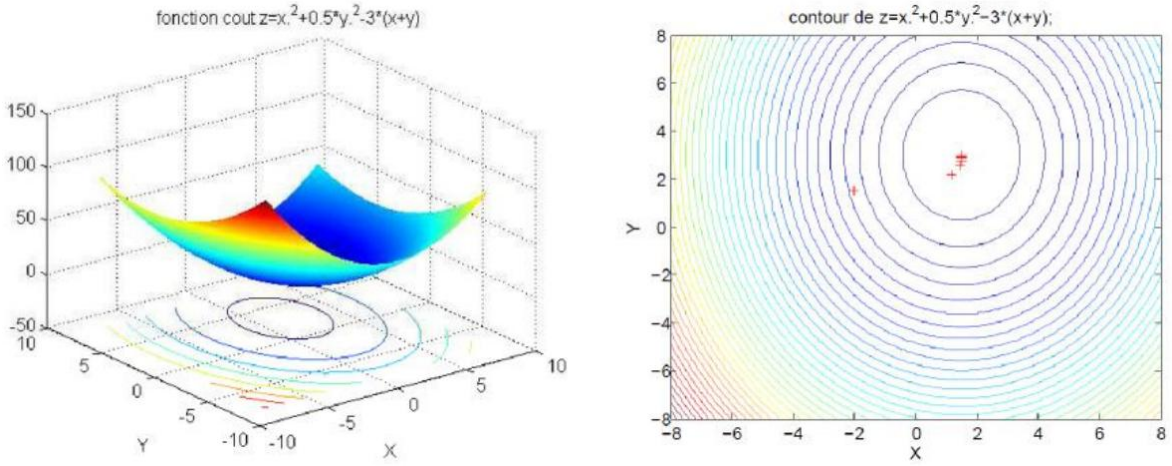


Figure II.5: Représentation graphique de la fonction

Itérations

Itérations	x_1	x_2	f(x)
1	-2	1,5000	6,6250
2	1,1500	2,1750	6,6250
3	1,4650	2,5462	-6,2872
4	1,4965	2,7504	-6,6458
5	1,4996	2,8627	6,7188
6	1,4999	2,9245	-6,7406
7	1,4999	2,9585	-6,7471

2.2.2 Méthode du gradient à pas variable

La méthode de la plus forte pente à pas optimal est la méthode de gradient la plus utilisée. Cette méthode propose un choix du pas qui rend la fonction coût minimale le long de la direction de descente choisie.

On choisit ici $-\nabla f(x\,k)$ pour $d\,k$: il s'agit de la descente de plus forte pente. On pose ensuite $g(s) = f(x\,k - s.\nabla f(x\,k))$ et on calcule ρk de façon à minimiser g pour $\rho \geq 0$. On est alors ramené à un problème d'optimisation unidimensionnelle.

L'algorithme de la plus forte pente peut s'écrire de la suivante:

- Choisir un point de départ x^0;
- $k \leftarrow 0$
- répéter
  - $d^k \leftarrow -\nabla f(x^k)$
  - déterminer s_k tel que $f(x^k + s_k d^k) = \underset{s\geq 0}{\text{Min}}\ f(x^k + s d^k)$
  - $x^{k+1} \leftarrow x^k + s_k d^k$
  - $k \leftarrow k + 1$

tant que le test d'arrêt n'est pas vérifié.

Le test d'arrêt peut être par exemple :

- le gradient est très petit : $\sum_{i=1}^{n}\left(\frac{\partial f}{\partial x_i}(x^k)\right)^2 \leq \varepsilon$, où ε est un paramètre donné ;
- la suite x^k est « presque » stationnaire : $|f(x^{k+1}) - f(x^k)| \leq \varepsilon$ (ε donné).

On peut aussi exiger que l'un de ces tests soit vérifié sur plusieurs itérations ou que plusieurs tests soient satisfaits simultanément.

L'inconvénient de cette méthode est que la vitesse de convergence peut être très faible (linéaire avec un coefficient proche de 1). Cette lenteur peut s'expliquer de la façon suivante :

l'égalité $\frac{d}{ds}[f(x^k - s.\nabla f(x^k))](s_k) = 0$ s'écrit : $[\nabla f(x^k)]^t.\nabla f(x^{k+1}) = 0$; les directions de déplacement successives sont orthogonales.

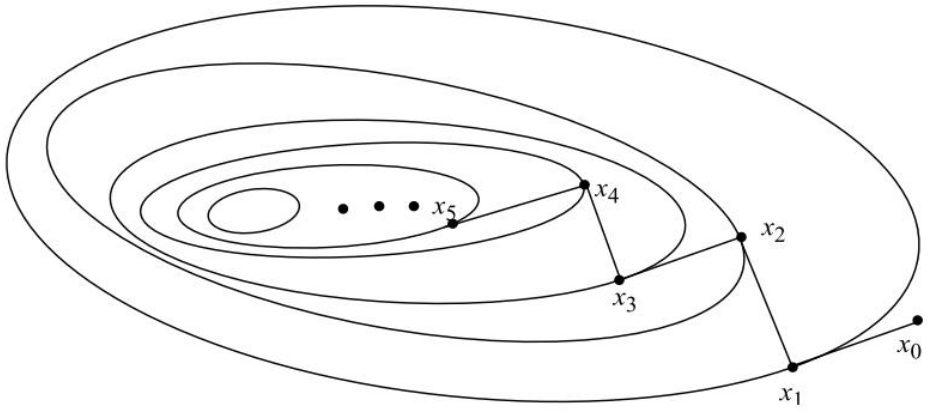


Figure II.6: Courbes de niveau de déplacements

Application 06

Calculer les trois itérations avec la méthode de plus forte pente à pas optimal.

$$f(x,y) = 4x^2 - 4xy + 2y^2 \text{ avec } x^0 = (2\ ,\ 3)^t$$

Solution

$\nabla f(x_0\ ,\ y_0) = (8x_0 - 4y_0\ ,\ -4x_0 + 4y_0)^t$

On part de $x\ (0) = (2\ ,\ 3)^t$

Première itération

on a $$\nabla f(x^{(0)}) = {}^t(4,4)$$

d'où $$x^{(0)} - \rho\nabla f(x^{(0)}) = {}^t(2,3) - \rho\,{}^t(4,4) = {}^t(2-4\rho, 3-4\rho)$$

Ainsi

$$f(x^{(0)} - \rho\nabla f(x^{(0)})) = 4(2-4\rho)^2 - 4(2-4\rho)(3-4\rho) + 2(3-4\rho)^2 = 10 - 32\rho + 32\rho^2$$

Cette dernière fonction est un trinôme du second degré, qui est minimale en $\rho = 1/2$. On trouve ainsi

$$\boxed{x^{(1)} = {}^t(0,1)}$$

Deuxième itération

on a $$\nabla f(x^{(0)}) = {}^t(-4,4)$$

d'où $$\begin{aligned} x^{(1)} - \rho\nabla f(x^{(1)}) &= {}^t(0,1) - \rho\,{}^t(-4,4) \\ &= {}^t(4\rho, 1-4\rho) \end{aligned}$$

Ainsi

$$\begin{aligned} f(x^{(1)} - \rho\nabla f(x^{(1)})) &= 4(4\rho)^2 - 4(4\rho)(1-4\rho) + 2(1-4\rho)^2 \\ &= 2 - 32\rho + 160\rho^2 \end{aligned}$$

Cette dernière fonction est un trinôme du second degré, qui est minimale en $\rho = 1/10$. On trouve ainsi

$$\boxed{x^{(2)} = {}^t\left(\frac{2}{5}, \frac{3}{5}\right)}$$

Troisième itération

on a $$\nabla f(x^{(2)}) = {}^t\left(\frac{4}{5}, \frac{4}{5}\right)$$

d'où $$\begin{aligned} x^{(2)} - \rho\nabla f(x^{(2)}) &= {}^t\left(\frac{2}{5}, \frac{3}{5}\right) - \rho\,{}^t\left(\frac{4}{5}, \frac{4}{5}\right) \\ &= {}^t\left(\frac{2}{5} - \frac{4\rho}{5}, \frac{3}{5} - \frac{4\rho}{5}\right) \end{aligned}$$

Ainsi

$$\begin{aligned} f(x^{(2)} - \rho\nabla f(x^{(2)})) &= 4(\frac{2}{5} - \frac{4\rho}{5})^2 - 4(\frac{2}{5} - \frac{4\rho}{5})(\frac{3}{5} - \frac{4\rho}{5}) + 2(\frac{3}{5} - \frac{4\rho}{5})^2 \\ &= \frac{2}{5} - \frac{32}{25}\rho + \frac{32}{25}\rho^2 \end{aligned}$$

Cette dernière fonction est un trinôme du second degré, qui est minimale en $\rho = 1/2$. On trouve ainsi

$$\boxed{x^{(3)} = {}^t\left(0, \frac{1}{5}\right)}$$

Valeurs de $f(x^{(k)})$

L'algorithme minimise bien la fonctionnelle f ; on a en effet

$$\begin{aligned} f(x^{(0)}) &= 10 & f(x^{(1)}) &= 2 \\ f(x^{(2)}) &= \frac{2}{5} & f(x^{(3)}) &= \frac{2}{25} \end{aligned}$$

2.3 Méthodes du gradient conjugué

Les méthodes du gradient conjugué sont utilisées pour résoudre les problèmes d'optimisation non linéaires sans contraintes spécialement les problèmes de grandes tailles. On l'utilise aussi pour résoudre les grands systèmes linéaires.

Elles reposent sur le concept des directions conjuguées parce que les gradients successifs sont orthogonaux entre eux et aux directions précédentes.

La méthode du gradient conjugué est la base des méthodes rapides de minimisation des fonctions régulières et c'est la meilleure méthode de résolution des grands systèmes issus de l'approximation des équations aux dérivées partielles. Le principe de cette méthode est d'accélérer la méthode du gradient en cherchant à l'étape k directement le minimum de la fonction f(x) dans le plan (P) formé au point x_k par les directions d_{k-1} et g_k .

2.3.1 Cas des fonctions quadratiques

Soit $f(x) = \frac{1}{2}X^T AX + b^T X + C$ une fonction quadratique, où A est définie positive.

La méthode consiste, à partir d'un point x_0 , à minimiser f suivant n directions d^0, d^1 , …, d^{n-1} mutuellement conjuguées par rapport à A, c'est-à-dire vérifiant :

Pour $1 \le i < j \le n$, $(d^i)^t . A . d^j = 0$

Soient n telles directions : d^0 , d^1 , …, d^{n-1}

Ayant déterminé x^k, le point x^{k+1} est le point : $x^{k+1} = x^k + s_k d^k$

où s_k est choisi de façon à minimiser $q(x^k + s_k d^k)$.

On a donc : $(d^k)^t . \nabla q(x^k + s_k d^k) = 0$ ou encore : $(d^k)^t . [A(x^k + s_k d^k) + b] = 0$

d'où l'on déduit : $s_k = -\dfrac{(d^k)^t . (Ax^k + b)}{(d^k)^t . A . d^k}$

Lemme : Si d^0, d^1, …, d^{k-1} sont mutuellement conjuguées, alors on a pour tout $i < k$ la relation : $(d^i)^t . \nabla q(x^k) = 0$,.

Preuve : on a en effet $(d^i)^t . \nabla q(x^k) = (d^i)^t . (Ax^k + b)$

$$= (d^i)^t [A\,(x^i + \sum_{j=i}^{k-1} s_j d^j) + b]$$

$$= (d^i)^t . (Ax^i + b) + s_i . (d^i)^t . A . d^i$$

$= 0$ d'après la valeur de s_i calculée ci-dessus.

Théorème : Le point x^n est l'optimum de $q(x)$ sur $\mathbb{R}^n$.

Preuve : Les directions d^0, d^1, … , d^{n-1} étant mutuellement conjuguées, elles forment une base de $\mathbb{R}^n$. D'après le lemme, $\nabla q(x^n) = 0$, ce qui démontre le théorème.

La méthode de Fletcher et Reeves engendre au fur et à mesure les directions d^i ; on l'explicite ci-dessous en posant : $g^k = \nabla q(x^k) = Ax^k + b$.

- Choisir un point de départ x^0 ; poser $d^0 := -g^0$.
- Pour k variant de 0 à n faire :
  - $s_k \leftarrow -\dfrac{(d^k)^t.g^k}{(d^k)^t.A.d^k}$
  - $x^{k+1} \leftarrow x^k + s_k.d^k$
  - $b_k \leftarrow \dfrac{(g^{k+1})^t.A.d^k}{(d^k)^t.A.d^k}$
  - $d^{k+1} \leftarrow -g^{k+1} + b_k.d^k$

(on pourra remarquer l'égalité suivante : $(d^k)^t.g^k = -\|g^k\|^2$)

Pour justifier la méthode, il suffit de vérifier que d^0, d^1 … d^{n-1} sont mutuellement conjuguées. Nous montrons qu'elles le sont à l'aide d'une récurrence. Pour cela, soit k compris entre 0 et $n - 2$; on suppose que les directions d^0, d^1 … d^k sont mutuellement conjuguées. On a alors pour $k + 1$:

$$(d^k)^t.A.d^{k+1} = (d^k)^t.A.(-g_{k+1} + b_k.d^k)$$

$$= -(d^k)^t.A.g^{k+1} + b_k.(d^k)^t.A.d^k = 0 \text{ d'après le choix de } b_k.$$

Pour $i < k$, $(d^{k+1})^t.A.d^i = -(g^{k+1})^t.A.d^i + b_k.(d^k)^t.A.d^i = -(g^{k+1})^t.A.d^i$

$$\text{Or : } A.d^i = A\left(\frac{x^{i+1} - x^i}{s_i}\right) = \frac{Ax^{i+1} - Ax^i}{s_i} = \frac{g^{i+1} - g^i}{s_i}$$

D'autre part : $g^{i+1} = -d^{i+1} + b_i.d^i$ et $g^i = -d^i + b_{i-1}.d^{i-1}$.

D'après le lemme, g^{k+1} est orthogonal à d^{i+1}, d^i et d^{i-1} ; $A.d^i$ étant combinaison linéaire de ces trois vecteurs, $(g^{k+1})^t.A.d^i = 0$, ce qui montre l'égalité $(d^{k+1})^t.A.d^i = 0$.

Pour terminer, nous démontrons une formule qui nous sera utile dans le paragraphe suivant. On a : $g^{k+1} - g^k = A(x^{k+1} - x^k) = s_k A.d^k$.

$$\text{D'où : } (g^{k+1})^t.A.d^k = \frac{(g^{k+1})^t.(g^{k+1} - g^k)}{s_k}$$

Comme $g^k = -d^k + b_{k-1}.d^{k-1}$, le lemme montre l'égalité $(g^{k+1})^t.g^k = 0$.

$$\text{D'où : } b_k = \frac{1}{s_k}\frac{(g^{k+1})^t.g^{k+1}}{(d^k)^t.A.d^k} = -\frac{(g^{k+1})^t.g^{k+1}}{(g^k)^t.d^k}$$

$(g^k)^t.d^k = (g^k)^t(-g^k + b_{k-1}.d^{k-1}) = -(g^k)^t.g^k$ d'après le lemme.

$$\text{On en déduit le résultat : } b_k = \frac{\left\|g^{k+1}\right\|^2}{\left\|g^k\right\|^2}.$$

Application 07

Soit la fonction $f\ x\ = x_1^2 + \frac{1}{2}x_2^2 - 3(x_1 + x_2)$ trouver le minimum de cette fonction en partant du point de cordonnée $x_0 = (-2{,}1.5)$; critère d'arrêt est $\|x_{k+1} - x_k\| < 10^{-3}$

Solution

Etape 0 : (initialisation)

Soit x_0(-2.1.5) le point de départ, $g_0 = \nabla f\ x_0\ = \begin{matrix} \frac{\partial f}{x_1} = 2x_1 - 3 = -7 \\ \frac{\partial f}{x_2} = x_2 - 3 = -1.5 \end{matrix}$

Sous la forme suivante $\nabla q\ x_0\ = Ax_0 + b$, Matrice $A = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}$ et $b = \begin{pmatrix} 3 \\ -3 \end{pmatrix}$ on pose $d_0 = -\nabla f\ x_0\ = \begin{pmatrix} 7 \\ 1.5 \end{pmatrix}$

Poser $k = 0$ et aller à l'étape 1:

Etape 1 :

si $g_0 = 0$: STOP ($x^* = x_k$)."Test d'arrêt" Test non vérifier donc aller a l'étape suivante.

Etape 2

$$x_{1_{0+1}} = x_{1_0} + \alpha_0 d_0$$
$$x_{2_{0+1}} = x_{2_0} + \alpha_0 d_0$$

$$\alpha_0 = \frac{-d_0^T g_0}{d_0^T A d_0} = \frac{-\ 7\quad 1.5 \times \begin{matrix} -7 \\ -1.5 \end{matrix}}{7\quad 1.5 \times \begin{matrix} 2 & 0 \\ 0 & 1 \end{matrix} \begin{matrix} 7 \\ 1.5 \end{matrix}} = 0.5112$$

$$x_{1_1} = -2 + \ 0.5112\ * 7 = 1.5786$$
$$x_{2_1} = 1.5 + \ 0.5112\ * 1.5 = 2.2668$$

Test d'arrêt

$$g_1 = \nabla f\ x_1\ = \begin{matrix} \frac{\partial f}{x_1} = 2x_{1_1} - 3 = 0.1571 \\ \frac{\partial f}{x_2} = x_{2_1} - 3 = -0.7332 \end{matrix} \neq 0$$

on pose k=k+1

$$x_{1_2} = x_{1_1} + \alpha_1 d_{1_1}$$
$$x_{2_2} = x_{2_1} + \alpha_1 d_{1_2}$$
$$d_1 = -\nabla q\ x_1\ + \beta_1 d_0$$
$$\beta_1 = \frac{g_1^T A d_0}{d_0^T A d_0}$$

$$g_1 = \nabla f\ x_1\ = \begin{matrix} 0.1571 \\ -0.7332 \end{matrix}$$

$$\beta_1 = \frac{0.1571\quad -0.7332 \times \begin{matrix} 2 & 0 \\ 0 & 1 \end{matrix} \times \begin{matrix} 7 \\ 1.5 \end{matrix}}{7\quad 1.5 \times \begin{matrix} 2 & 0 \\ 0 & 1 \end{matrix} \begin{matrix} 7 \\ 1.5 \end{matrix}} = 0.0110$$

$$d_{1_1} = -\nabla q\ x_1\ + \beta_1 d_{0_1} = 0.1571 + \ 0.0110\ * 7 = -0.0803$$
$$d_{1_2} = -\nabla q\ x_1\ + \beta_1 d_{0_2} = -0.7332 + \ 0.0110\ * 1.5 = 0.7496$$

$$\alpha_1 = \frac{-d_1^T g_1}{d_1^T A d_1} = \frac{-\ -0.0803\quad -0.7496 \times \begin{matrix} 0.1571 \\ -0.7332 \end{matrix}}{-0.0803\quad 0.7496 \times \begin{matrix} 2 & 0 \\ 0 & 1 \end{matrix} \begin{matrix} -0.0803 \\ 0.7496 \end{matrix}} = 0.9780$$

$$x_{1_2} = 1.5786 + \ 0.9780\ * (-0.0803) = 1.5000$$
$$x_{1_2} = 2.2668 + \ 0.9780\ * (0.7496) = 3.0000$$

Test d'arrêt $\quad g_2 = \nabla f\ x_1\ = \begin{matrix} 0.888 \times 10^{-15} \\ 0 \end{matrix} \cong 0$ arrêt.

2.3.2 Cas des fonctions quelconques

L'algorithme de Fletcher et Reeves pour une fonction quelconque est le suivant :

- partir d'un point x^0 ;
- faire $d^0 \leftarrow -\nabla f(x^0)$ et $k \leftarrow 0$;
- répéter
  - choisir s_k minimisant $f(x^k + s\, d^k)$, par rapport à s
  - $x^{k+1} \leftarrow x^k + s_k.d^k$
  - $b_k \leftarrow \dfrac{\left\|\nabla f(x^{k+1})\right\|^2}{\left\|\nabla f(x^k)\right\|^2}$
  - $d^{k+1} \leftarrow -\nabla f(x^{k+1}) + b_k.d^k$
  - $k \leftarrow k + 1$

jusqu'à ce qu'un test d'arrêt soit vérifié.

Cette méthode a deux avantages : elle nécessite le stockage de très peu d'informations et sa vitesse de convergence est très supérieure à celle des algorithmes de gradient classiques.

2.4 Méthode de Newton

On suppose ici que f est deux fois continûment dérivable et que l'on sait calculer ses dérivées secondes. Au voisinage d'un point x^k , on approche f par la fonction quadratique donnée par la formule de Taylor d'ordre 2 :

$$q(x) = f(x^k) + (x - x^k)^t.\nabla f(x^k) + \frac{1}{2}(x - x^k)^t.\nabla^2 f(x^k).(x - x^k). \quad (19)$$

L'algorithme de Newton s'écrit :

$$\begin{cases} x_0 \in \mathbb{R}^n, \\ x_{k+1} = x_k - [H_f(x_k)]^{-1}\ \nabla f(x_k) \end{cases} \quad (20)$$

L'intérêt de cette suite est sa convergence quadratique vers un minimiseur local à la condition que x^0

soit assez proche d'un minimiseur. Néanmoins d'un point de vue pratique, cette Méthode comporte les

mêmes inconvénients que dans le cas monovariable :

- la méthode peut diverger si le point de départ est trop éloigné de la solution,
- la méthode n'est pas définie si la matrice Hessienne n'est pas inversible,

- la méthode peut converger indifféremment vers un minimum, un maximum ou un point de selle.

Algorithme

Données : f une fonction différentiable et x_0 un point initial

1- Initialisation

$k=0$: choix de $x_0 \in \Re$ dans un voisinage de x^*.

2- Itération k

$$x_{k+1} = x_k + D$$

3- Critère d'arrêt

Si $\|x_{k+1} - x_k\| < \epsilon, STOP$

Si non, On pose $k=k+1$ et on retourne à 2.

L'étape 2. de la méthode revient à résoudre le système linéaire suivant:

$$\nabla^2 f\ x_k\ D = -\nabla f(x_k)$$

Puis à poser $x_{k+1} = x_k + D$

Application 08

Soit la fonction $f\ x = x_1^2 + \frac{1}{2}x_2^2 - 3(x_1 + x_2)$ trouver le minimum de cette fonction en partant du point de cordonnée $x_0 = (-2,1.5)$; critère d'arrêt est $\|x_{k+1} - x_k\| < 10^{-3}$

Solution

1- $x_0 = (-2,1.5)$

2- Calcul de la dérivée par rapport à x_1

$$\frac{\partial f}{x_1} = 2x_1 - 3 = -7\ au\ point\ x_{1\,0}$$

- Calcul de la dérivée par rapport à x_2

$$\frac{\partial f}{x_1} = x_2 - 3 = -1.5\ au\ point\ x_{2\,0}$$

- Calcul de la dérivée deuxième par rapport à x_1

$$\frac{\partial^2 f}{{x_1}^2} = 2$$

- Calcul de la dérivée deuxième par rapport à x_2

$$\frac{\partial^2 f}{{x_1}^2} = 1$$

- Calcule de D au point x_0

$$D_{x1} = -\nabla^2 f\ x_k\ ^{-1}\nabla f\ x_k = \frac{2x_1 - 3}{2} = 3.5\ au\ point\ de\ cordonné\ x_{1\,0}, x_{2\,0}$$

$$D_{x2} = -\nabla^2 f\ x_k\ ^{-1}\nabla f\ x_k = \frac{x_2 - 3}{1} = 5\ au\ point\ de\ cordonné\ x_{1\,0}, x_{2\,0}$$

- 1ère itération

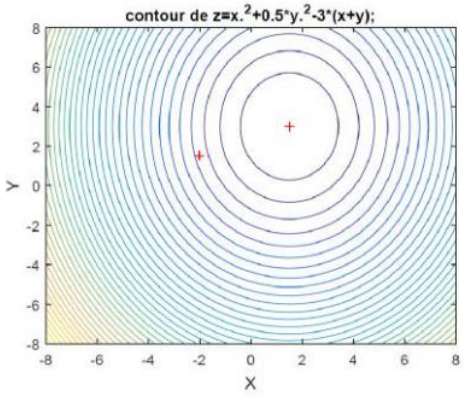
$$x_{1\,1} = x_{1\,0} + D = 1.5000$$

$$x_{2\,1} = x_{2\,0} + D = 3$$

3- Test d'arrêt *Si* $x_{1\,1} - x_{1\,0} < 10^{-3}, STOP$

$x_{2\,1} - x_{2\,0} < 10^{-3}, STOP$

Si non on pose k=2 et on retourne à 2



Les itérations

Itérations	x_1	x_2	$f(x)$
1	-2	1,50000000000000	6,62500000000000
2	1,5000000000000	3	-6.7500

2.5 Méthode quasi-Newton

Pour des problèmes de grandes dimensions, le calcul du Hessien est trop couteux. On peut alors utiliser des algorithmes, dits quasi-Newton, qui calculent donc une approximationB_k de $\nabla^2 f(x_k)^{-1}$ en fonction de $B_{k-1}, \nabla f\ x_k\ , \nabla f\ x_{k-1}\ , x_k\ et\ x_{k-1}$.

On trouve notamment deux méthodes de calcul :

- ✓ la formule de *Davidon-Fletcher-Powell* (DFP) :

$$B_k = B_{k-1} - \frac{B_{k-1} Y_k Y_k^T B_{k-1}}{Y_k^T B_{k-1} Y_k} + \frac{\overline{D}_{k-1} \overline{D}_{k-1}^T}{\overline{D}_{k-1}^T Y_k}$$

- ✓ la formule de Broyden-Fletcher-Goldfarb-Shanno (BFGS):

$$B_k = B_{k-1} + \left(1 + \frac{Y_k^T B_{k-1} Y_k}{Y_k^T \overline{D}_{k-1}}\right) \frac{\overline{D}_{k-1} \overline{D}_{k-1}^T}{Y_k^T \overline{D}_{k-1}} - \frac{B_{k-1} Y_k \overline{D}_{k-1}^T + \overline{D}_{k-1} Y_k B_{k-1}}{Y_k^T \overline{D}_{k-1}}$$

Avec

$$Y_k = \nabla f\ x_k\ - \nabla f(x_{k-1})$$

Ces formules donnent toujours des matrices définies positives. A l'aide de ces estimations, on établit un algorithme à convergence très efficace et particulièrement robuste. Son unique inconvénient est la nécessité du stockage en mémoire d'une matrice de taille $n \times n$. Dans la pratique, on utilise en général la méthode BFGS qui est plus efficace.

Algorithme

Données : f une fonction différentiable et x_0 un point initial

Etape 1 :soit $\varepsilon > 0$ critère d'arrêt, choisir β_1 défini positive quelconque (par exemple $\beta_1 = I$)

Poser k=1 et aller à l'étape suivante

Etape 2 : Si $\|\nabla f\ x_k\| < \varepsilon$ STOP ; Si non, poser $d_k = -\beta_k \times g_k$ et déterminer le pas optimale α_k solution du problème linéaire $\alpha_k = minf\ x_k + \alpha d_k\ ,\ \alpha \geq 0$(par une méthode de recherche linéaire par exemple dichotomie ou Wolfe) et poser

$$x_{k+1} = x_k + \alpha_k d_k$$

Etape 3 : construire β_{k+1} comme suit :

$$B_{k+1} = B_k - \frac{B_k Y_{k+1} Y_{k+1}^T B_k}{Y_{k+1}^T B_k Y_{k+1}} + \frac{\bar{D}_k \bar{D}_k^T}{\bar{D}_k^T Y_{k+1}} \quad \text{Selon } \mathbf{DFP} \quad \text{Ou}$$

$$B_{k+1} = B_k + \left(1 + \frac{Y_{k+1}^T B_k Y_{k+1}}{Y_{k+1}^T \bar{D}_k}\right) \frac{\bar{D}_k \bar{D}_k^T}{Y_{k+1}^T \bar{D}_k} - \frac{B_k Y_k \bar{D}_k^T + \bar{D}_k Y_{k+1} B_k}{Y_{k+1}^T \bar{D}_k} \quad \text{Selon } \mathbf{BFGS}$$

Avec

$$Y_{k+1} = \nabla f\ x_{k+1} - \nabla f(x_k)$$
$$D_k = \alpha_k d_k$$

Remplacer k par $k+1$ et aller à l'étape 1

CHAPITRE III

OPTIMISATION NON LINÉAIRE AVEC CONTRAINTES

1. Introduction

L'optimisation non linéaire avec contraintes consiste à minimiser ou maximiser une fonction non linéaire sous certaines contraintes, qui peuvent être elles-mêmes non linéaires.

Un problème d'optimisation non linéaire avec contraintes peut se présenter sous la forme; **Min f(x)** (1)

$\mathbf{x \in R^n}$

Sous contraintes

$h_i(x) \leq 0$ (2) $\{i = 1, \ldots . n\}$;(contraintes d'inégalité)

$g_j(x) = 0$ (3) $\{j = 1, \ldots \ldots m\}$(contraintes d'égalité)

2. Multiplicateurs de Lagrange et condition de Karush-Kuhn-Tucker

Lorsqu'il s'agit des contraintes d'égalité, le problème à résoudre peut s'écrire sous la forme:

$$\begin{cases} \text{Min } f(x) \\ hi(x) = 0 \end{cases} \quad \{i = 1, \ldots . n\};$$

La recherche d'un point minimum au sens de Karush-Kuhn-Tucker revient à résoudre le système de (n+m) équations pour trouver les inconnus (x; λi).

$$\nabla f(x) + \sum_i^n \lambda_i \nabla h_i(x) = 0 \qquad (4)$$

$$\text{Avec } \lambda_i h_i(x) = 0 \quad , \{i = 1, \ldots . n\} \quad (5)$$

Ces équations sont appelées équations de **Karush-Kuhn-Tucker** dont λi représentent **les multiplicateurs de Lagrange**.

Application 01

On considère le problème d'optimisation non linéaire avec contrainte, suivant :

$$\begin{cases} \text{Min } f(x) = 4.x_1^2 + x_2^2 - 2x_2 \\ \text{Sous contraintes } h_1(x) = -2.x_1 - x_2 = -5 \end{cases}$$

1. On cherche à minimiser la fonction f(x) dans le sens de Karush-Kuhn-Tucker, Déterminer la valeur du multiplicateur de Lagrange λ ?
2. En déduire la solution optimale x^* ?

Solution

Détermination la valeur du multiplicateur de Lagrange λ dans le sens de K-K-T

$$\nabla f(x)+ \lambda_1 h(x)=0 \quad .....(*)$$

$$\lambda_1 h(x)=0 \quad ..... \ (**)$$

$$(*) \rightarrow \begin{pmatrix} 8\,x_1 \\ 2x_2-2 \end{pmatrix}+\lambda_1 \begin{pmatrix} -2 \\ -1 \end{pmatrix}=0 \rightarrow \begin{cases} 8\,x_1-2\lambda_1 \\ 2x_2-2-\lambda_1 \end{cases} \rightarrow \begin{cases} x_1=\lambda_1/4 \\ x_2=1+\lambda_1/2 \end{cases}$$

$$(**) \rightarrow \lambda_1(-2x_1-x_2+5)=0 \rightarrow \text{D'où} \quad \lambda=4$$

La solution optimale x*

$x_1^*=x_1=\lambda_1/4=1$

$x_2^*=x_2=1+\lambda_1/2=3$

Lorsqu'il s'agit des contraintes d'égalités et des contraintes d'inégalités, le problème à résoudre peut s'écrire sous la forme:

$$\begin{cases} \text{Min } f(x) \\ h_i(x) = 0 \\ g_j(x) \leq 0 \end{cases} \qquad \{i = 1,\ldots..n\}; \{j = 1,\ldots..m\};$$

Une condition est nécessaire pour x* soit un minimum de f(x) et qu'il existe des nombres positifs $\boldsymbol{\lambda_i}$ et des nombres réels $\boldsymbol{\mu_j}$ tel que;

$$\nabla f(x^*)+\sum_i^n \lambda_i \nabla h_i(x*) + \sum_j^m \mu_j \nabla g_j(x*)=0 \qquad (6)$$

$$\text{Avec } \lambda_i h_i=0 \ , \{i = 1,\ldots..n\} \ (7)$$

Le Lagrangien est exprimé par $\boldsymbol{\mathcal{L}(x,\lambda_i, \mu_j)}=\nabla f(x)+\sum_i^n \lambda_i \nabla h_i(x) + \sum_j^m \mu_j \nabla g_j(x) \qquad (8)$

Le problème consiste à écrire $\nabla \boldsymbol{\mathcal{L}(x^*,\lambda_i, \mu_j)=0} \qquad (9)$

3. Méthodes de pénalité

Parmi les méthodes de résolution des problèmes d'optimisation non linéaire avec contraintes les plus connues la méthode de pénalisation (pénalité).

Le concept de base de cette méthode est de ramener le problème initial à la résolution d'une suite de problème d'optimisation sans contraintes.

- **Principe général de méthode de pénalité**

On considère le problème d'optimisation suivant:

$$\begin{cases} \text{Min } f(x) \\ h_i(x) \leq 0 \end{cases} \qquad \{i = 1, \ldots . n\};$$

Alors, on procède à la résolution d'un autre problème d'optimisation sans contraintes exprimé par:

$$\mathbf{Min}\ \psi(\mathbf{x}) = \mathbf{f(x)} + \frac{\mathbf{1}}{\boldsymbol{\varepsilon}}\boldsymbol{\alpha}(\mathbf{x}) \quad (10)$$

Avec: $\varepsilon>0$ et $\alpha(x)$ est la fonction de pénalisation des contraintes définie par:

$\alpha(x) =0$ si $x \in \Omega$ (11)

$\alpha(x) =+\infty$ si $x \notin \Omega$ (12)

Pour qu'on puisse résoudre le problème, on doit supposer la fonction de pénalisation pour différentes contraintes:

- Contraintes $h(x) =0$; la fonction $\alpha(x) =\|h(x)\|^2$ (13)
- Contraintes $g(x) \leq 0$; la fonction $\alpha(x) =\|g(x)\|^2$ (14)

Algorithme de la méthode de pénalité extérieure

Initialisation

k = 1

Choisir x^0 et $\varepsilon^1 > 0$

Itération k tant que le critère d'arrêt n'est pas satisfait :

a) Résoudre le problème :

$$\text{Minimiser } \psi(x_k) = f(x_k) + \frac{1}{\varepsilon_k} \times \alpha(x_k)$$

b) k = k + 1 ; prendre $\varepsilon_{k+1} < \varepsilon_k$.

Il existe essentiellement deux manières de définir la fonction de pénalisation qui donne naissance à deux comportements différents du processus, soit des solutions successives de problèmes sans contraintes qui tendent vers un optimum du problème contraint par l'extérieur de Ω dont il s'agit de la méthode de pénalisation extérieure, ou bien, qu'ils tendent un optimum par l'intérieur de Ω qu'on appelle la méthode de pénalisation intérieure.

Algorithme de la méthode de pénalité intérieure

Pour cette méthode, la fonction α(x) est définie dans l'intérieur du domaine admissible Ω de façon qu'elle soit continue et positive, et qu'elle tendes vers $+\infty$ lorsque x approche de la frontière.

$$\alpha(x)=-\sum_j^m \frac{1}{h_j(x)} \qquad (15)$$

L'algorithme de la méthode de pénalité intérieure est donné par:

a)- Choisir x_0 admissible intérieur et $\varepsilon^1 > 0$,

b)- A l'itération k, on est au point x_k

Chercher : min $\psi(x_k, t_k)$ à partir de x_k par une méthode itérative d'optimisation non contrainte

c)- Si $\left|\frac{1}{\varepsilon_k} \times \alpha(x_{k+1})\right| < \bar{\varepsilon}$ alors x_{k+1} est une bonne approximation de l'optimum de *f* et on arrête les calculs ; si non choisir $\varepsilon_{k+1} < \varepsilon_k$ et retour en (b).

Application 02

On considère le problème d'optimisation non linéaire avec contrainte, suivant :

$$\begin{cases} \text{Maximiser} \quad Q = a_p \times f \times V_c \qquad \text{Avec} : a_p = 3 \\ \textit{avec} \\ \quad f \le 0.4 \\ \quad V_c \le 220 \\ \quad f^{0.71} \times V_c \le 166.995 \\ \quad f^{0.5} \times V_c = 80.276 \end{cases}$$

Solution

Le problème d'optimisation avec contraintes est transformé en un problème de minimisation de la fonction :

$$\psi(f, V_c, t) = -3 \times f \times V_c - \frac{1}{\varepsilon} \times \left[\frac{1}{f - 0.4} + \frac{1}{V_c - 220} + \frac{1}{f^{0.71} \times V_c - 167} + \frac{1}{f^{0.5} \times V_c - 80.276} \right]$$

4. Méthodes de programmation quadratique séquentielle

Pour cette méthode, l'idée essentielle consiste à résoudre une succession de problèmes quadratiques avec contraintes.

L'algorithme SQP est donné comme suit

Initialisation

$k = 1$

Choisir x^0, λ^0 et μ^0

Itération k tant que le critère d'arrêt n'est pas satisfait :

a) Résoudre le problème quadratique :

$$\text{Minimiser } \frac{1}{2} d^T H_k d + \nabla f(x_k)^T d$$

$$\nabla h_i(x_k)^T d + h_i(x_k) = 0$$

$$\nabla k_i(x_k)^T d + k_i(x_k) \leq 0$$

Trouver : d_k, λ^{k+1} et μ^{k+1}

$x_{k+1} = x_k + d_k$

b) $k = k + 1$.

CHAPITRE IV

METHODES D'OPTIMISATION STOCHASTIQUES

1. Introduction

L'optimisation stochastique regroupe un ensemble de techniques utilisées pour résoudre des problèmes d'optimisation dans des contextes où l'incertitude, le bruit ou l'incomplétude de l'information empêchent l'utilisation de méthodes déterministes classiques. Contrairement à l'optimisation déterministe, où les fonctions à optimiser sont parfaitement connues et évaluables de manière exacte, les méthodes stochastiques acceptent des fonctions bruitées, partiellement connues ou très coûteuses à évaluer.

Ces méthodes sont particulièrement utiles dans des domaines tels que l'apprentissage automatique, la recherche opérationnelle, la finance, ou encore les systèmes complexes où les fonctions objectifs peuvent être issues de simulations ou d'expérimentations.

Les algorithmes d'optimisation stochastique utilisent souvent des mécanismes probabilistes pour explorer l'espace des solutions. Ils cherchent à équilibrer exploration (chercher de nouvelles solutions prometteuses) et exploitation (améliorer les solutions déjà trouvées).

Parmi les approches les plus courantes, on trouve :

- ✓ **Descente de gradient stochastique (SGD) :** utilisée en apprentissage profond, elle évalue le gradient sur un échantillon aléatoire plutôt que sur l'ensemble des données.
- ✓ **Méthodes évolutionnaires :** comme les algorithmes génétiques ou les stratégies d'évolution, qui imitent les processus biologiques.
- ✓ **Recuit simulé :** inspiré de la physique statistique, il permet de sortir des optima locaux en acceptant temporairement des solutions moins bonnes.
- ✓ **Optimisation bayésienne :** particulièrement adaptée aux fonctions coûteuses à évaluer, elle utilise des modèles probabilistes (comme les processus gaussiens) pour guider la recherche.

L'un des grands avantages des méthodes stochastiques est leur capacité à échapper aux optima locaux, ce qui les rend efficaces dans des paysages d'optimisation

complexes et non convexes. Cependant, elles peuvent nécessiter un grand nombre d'évaluations et leur convergence peut être lente ou difficile à garantir.

2. L'algorithme génétique

Les algorithmes génétiques sont des algorithmes d'optimisation s'appuyant sur des techniques dérivées de la génétique et des mécanismes d'évolution de la nature : croisements, mutations, sélections, etc.... Ils appartiennent à la classe des algorithmes évolutionnaires.

2.1 Le codage

Chaque paramètre d'une solution est assimilé à un gène, toutes les valeurs qu'il peut prendre sont les allèles de ce gène, on doit trouver une manière de coder chaque allèle différent de façon unique (établir une bijection entre l'allèle "réel" et sa représentation codée).

Un chromosome est une suite de gène, on peut par exemple choisir de regrouper les paramètres similaires dans un même chromosome (chromosome à un seul brin) et chaque gène sera repérable par sa position : son locus sur le chromosome en question.

Chaque individu est représenté par un ensemble de chromosomes, et une population est un ensemble d'individus.

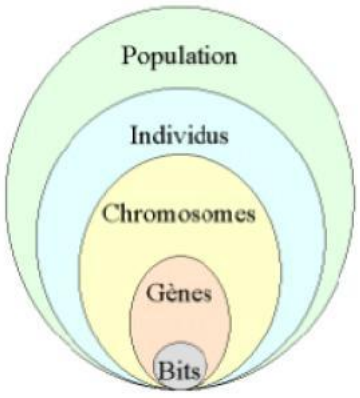


Figure IV.1: Les cinq niveaux d'organisation d'un algorithme génétique

Il existe trois principaux types de codage utilisables, et on peut passer de l'un à l'autre relativement facilement :

• **le codage binaire :** c'est le plus utilisé.

Chaque gène dispose du même alphabet binaire {0, 1}. Un gène est alors représenté par un entier long (32 bits), les chromosomes qui sont des suites de gènes sont représentés par des tableaux de gènes et les individus de notre espace de recherche sont représentés par des tableaux de chromosomes.

Ce cas peut être généralisé à tout alphabet allélique n-aire permettant un codage plus intuitif, par exemple pour le problème du voyageur de commerce on peut préférer utiliser l'alphabet allélique {c1, c2, c3, ..., cn} où ci représente la ville de numéro i.

• **le codage réel :** cela peut-être utile notamment dans le cas où l'on recherche le maximum d'une fonction réelle.

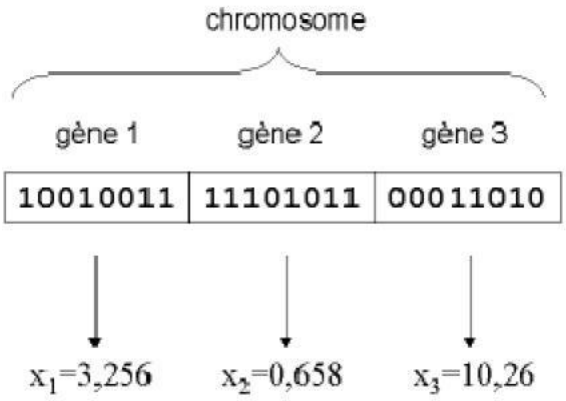


Figure IV.2 : Illustration schématique du codage des variables réelles

• **le codage Gray** : dans le cas d'un codage binaire on utilise souvent la "distance de Hamming" comme mesure de la dissimilarité entre deux éléments de population, cette mesure compte les différences de bits de même rang de ces deux séquences. Et c'est la que le codage binaire commence à montrer ses limites. En effet, deux éléments voisins en terme de distance de Hamming ne codent pas nécessairement deux éléments proches dans l'espace de recherche. Cet inconvénient peut être évité en utilisant un "codage de Gray" : le codage de Gray est un codage qui a comme propriété que : entre un élément n et un élément n + 1, donc voisin dans l'espace de recherche, un seul bit diffère.

2.2 L'opérateur de sélection

Cet opérateur est chargé de définir quels seront les individus de P qui vont être dupliqués dans la nouvelle population P' et vont servir de parents (application de l'opérateur de croisement).

Soit n le nombre d'individus de P, on doit en sélectionner n/2 (l'opérateur de croisement nous permet de repasser à n individus).

Cet opérateur est peut-être le plus important puisqu'il permet aux individus d'une population de survivre, de se reproduire ou de mourir. En règle générale, la probabilité de survie d'un individu sera directement reliée à son efficacité relative au sein de la population.

On trouve essentiellement quatre types de méthodes de sélection différentes :

• La méthode de la "loterie biaisée" (roulette wheel) de GoldBerg,

• La méthode "élitiste",

• La sélection par tournois,

• La sélection universelle stochastique.

a) La loterie biaisée ou roulette wheel :

Cette méthode est la plus connue et la plus utilisée.

Avec cette méthode chaque individu a une chance d'être sélectionné proportionnelle à sa performance, donc plus les individus sont adaptés au problème, plus ils ont de chances d'être sélectionnés.

Pour utiliser l'image de la "roue du forain", chaque individu se voit attribué un secteur dont l'angle est proportionnel à son adaptation, sa "fitness".

On fait tourner la roue et quand elle cesse de tourner on sélectionne l'individu correspondant au secteur désigné par une sorte de "curseur", curseur qui pointe sur un secteur particulier de celle-ci après qu'elle se soit arrêté de tourner.

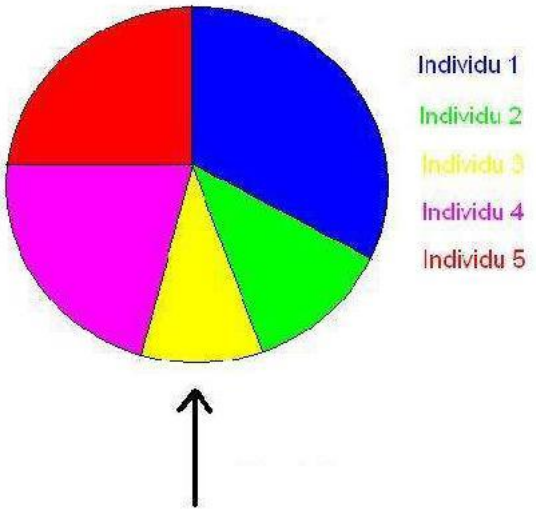


Figure IV. 3: la méthode de sélection de la loterie biaisée

Cette méthode, bien que largement répandue, a pas mal d'inconvénients :

• En effet, elle a une forte variance. Il n'est pas impossible que sur n sélections successives destinées à désigner les parents de la nouvelle génération P', la quasi-totalité, voire pire la totalité des n individus sélectionnés soient des individus ayant une fitness vraiment mauvaise et donc que pratiquement aucun
individu voire aucun individu a forte fitness ne fasse partie des parents de la nouvelle génération. Ce phénomène est bien sûr très dommageable car cela va complètement à l'encontre du principe des algorithmes génétiques qui veut que les meilleurs individus soient sélectionnés de manière à converger vers une solution la plus optimale possible.

• A l'inverse, on peut arriver à une domination écrasante d'un individu "localement supérieur". Ceci entraînant une grave perte de diversité. Imaginons par exemple qu'on ait un individu ayant une fitness très élavée par rapport au reste de la population, disons dix fois supérieure, il n'est pas impossible
qu'après quelques générations successives on se retrouve avec une population ne contenant que des copies de cet individu. Le problème est que cet individu avait une fitness très élevée, mais que cette fitness était toute relative, elle était très élevée mais seulement en comparaison des autres individus. On se retrouve donc face à problème connu sous le nom de "convergence prématurée; l'évolution se met donc à stagner et on atteindra alors jamais l'optimum, on restera bloqué sur un optimum local.

Il existe certaines techniques pour essayer de limiter ce phénomène, comme par exemple le "scaling", qui consiste à effectuer un changement d'échelle de manière à augmenter ou diminuer de manière forcée la fitness d'un individu par rapport à un autre selon leur écart de fitness.

Malgré tout, il est conseillé d'opter plutôt pour une autre méthode de sélection.

b) La méthode élitiste.

Cette méthode consiste à sélectionner les n individus dont on a besoin pour la nouvelle génération P' en prenant les n meilleurs individus de la population P après l'avoir triée de manière décroissante selon la fitness de ses individus.

Il est inutile de préciser que cette méthode est encore pire que celle de la loterie biaisée dans le sens où elle amènera à une convergence prématurée encore plus rapidement et surtout de manière encore plus sûre que la méthode de sélection de la loterie biaisée ; en effet, la pression de la sélection est trop forte, la variance nulle et la diversité inexistante, du moins le peu de diversité qu'il pourrait y avoir ne

résultera pas de la sélection mais plutôt du croisement et des mutations. Là aussi il faut opter pour une autre méthode de sélection.

c) La sélection par tournois :

Cette méthode est celle avec laquelle on obtient les résultats les plus satisfaisants. Le principe de cette méthode est le suivant : on effectue un tirage avec remise de deux individus de P, et on les fait "combattre". Celui qui a la fitness la plus élevée l'emporte avec une probabilité p comprise entre 0.5 et on répète ce processus n fois de manière à obtenir les n individus de P' qui serviront de parents.

La variance de cette méthode est élevée et le fait d'augmenter ou de diminuer la valeur de *p* permet respectivement de diminuer ou d'augmenter la pression de la sélection.

d) La sélection universelle stochastique :

Cette méthode semble être très peu utilisée et qui plus est possède une variance faible, donc introduit peu de diversité, nous n'entrerons donc pas dans les détails, on se contentera d'exposer sa mise en œuvre : On prend l'image d'un segment découpé en autant de sous-segments qu'il y a d'individus. Les individus sélectionnés sont désignés par un ensemble de points équidistants.

2.3 L'opérateur de croisement ou crossover

Le crossover utilisé par les algorithmes génétiques est la transposition informatique du mécanisme qui permet, dans la nature, la production de chromosomes qui héritent partiellement des caractéristiques des parents. Son rôle fondamental est de permettre la recombinaison des informations présentes dans le patrimoine génétique de la population.

Cet opérateur est appliqué après avoir appliqué l'opérateur de sélection sur la population P; on se retrouve donc avec une population P' de n/2 individus et on doit doubler ce nombre pour que notre nouvelle génération soit complète.

On va donc créer de manière aléatoire n/4 couples et on les fait se "reproduire". Les chromosomes (ensembles de paramètres) des parents sont alors copiés et recombinés de façon à former deux descendants possédant des caractéristiques issues des deux parents. Détaillons ce qui se passe pour chaque couple au niveau de chacun de leurs chromosomes :

Un, deux, voire jusqu'à lg - 1 (où lg est la longueur du chromosome) points de croisements (loci) sont tirés au hasard, chaque chromosome se retrouve donc séparé en "segments". Puis chaque segment du parent 1 est échangé avec son "homologue"

du parent 2 selon une probabilité de croisement pc . De ce processus résulte 2 fils pour chaque couple et notre population P' contient donc bien maintenant n individus.

On peut noter que le nombre de points de croisements ainsi que la probabilité de croisement permettent d'introduire plus ou moins de diversité. En effet, plus le nombre de points de croisements sera grand et plus la probabilité de croisement sera élevée plus il y aura d'échange de segments, donc d'échange de paramètres, d'information, et plus le nombre de points de croisements sera petit et plus la probabilité de croisement sera faible, moins le croisement apportera de diversité.

Ci-dessous, un schéma illustrant un croisement en un point, un autre pour un croisement en deux points, et enfin un schéma représentant un croisement avec lg - 1 points de croisements (on notera d'ailleurs sur ce schéma que l'échange d'un segment avec son homologue ne se fait pas toujours) :

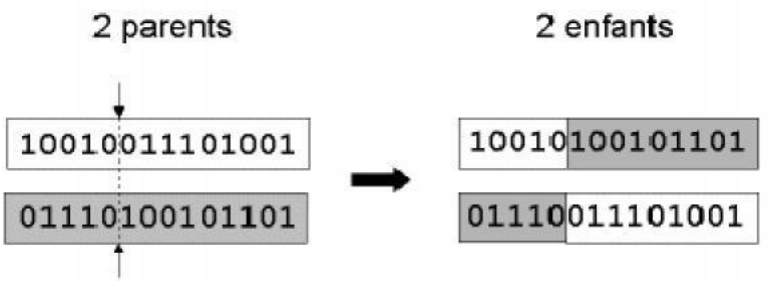


Figure IV.4: croisement avec un point de crossover

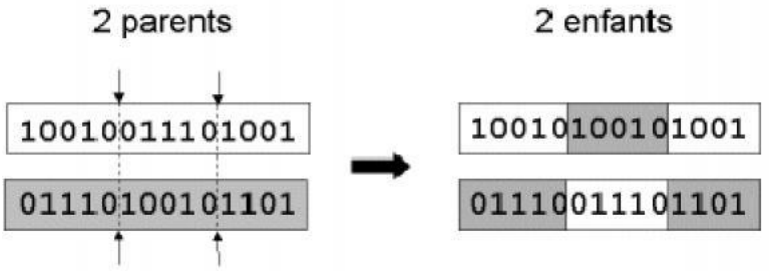


Figure IV.5: croisement avec 2 points de crossover

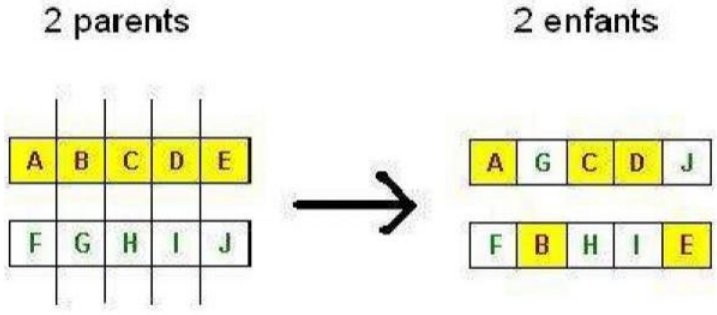


Figure IV.6: croisement uniforme

On peut citer aussi une autre méthode très utilisée dans le cas des problèmes modélisés par un codage binaire, il s'agit du ***croisement uniforme***. La mise en œuvre de ce procédé est fort simple, elle consiste à définir de manière aléatoire un "masque", c'est-à-dire une chaîne de bits de même longueur que les chromosomes des parents sur lesquels il sera appliqué. Ce masque est destiné à savoir, pour chaque locus, de quel parent le premier fils devra hériter du gène s'y trouvant; si face à un locus le masque présente un 0, le fils héritera le gène s'y trouvant du parent n° 1, si il présente un 1 il en héritera du parent n° 2. La création du fils n° 2 se fait de manière symétrique : si pour un gène donné le masque indique que le fils n° 1 devra recevoir celui-ci du parent n° 1 alors le fils n° 2 le recevra du parent n°2, et si le fils n° 1 le reçoit du parent n° 2 alors le fils 2 le recevra du parent n° 1.

L'opérateur de croisement favorise l'exploration de l'espace de recherche. En effet, considérons deux gènes A et B pouvant être améliorés par mutation. Il est peu probable que les deux gènes améliorés A' et B' apparaissent par mutation dans un même individu. Mais si un parent porte le gène mutant A' et l'autre le gène mutant B', l'opérateur de croisement permettra de combiner rapidement A' et B' et donc de créer un nouvel individu possédant cette combinaison, combinaison grâce à laquelle il est possible qu'il soit encore plus adapté que ses parents. L'opérateur de croisement assure donc le brassage du matériel génétique et l'accumulation des mutations favorables. En termes plus concrets, cet opérateur permet de créer de nouvelles combinaisons des paramètres des composants. Malgré tout, il est possible que l'action conjointe de la sélection et du croisement ne permette pas de converger vers la solution optimale du problème. En effet, imaginons que nous avons une population d'individus possédant un seul chromosome. Considérons un gène particulier de ce chromosome, on l'appellera G, gène ayant 2 allèles possibles : 0 et 1; si aucun individu de la population initiale ne possède l'allèle 1 pour ce gène, aucun croisement possible ne permettra d'introduire cet allèle pour notre gène G. Si la solution optimale au problème est telle que notre gène G possède l'allèle 1, il nous sera impossible d'atteindre cette solution optimale simplement par sélection et croisement. C'est pour remédier entre autre à ce problème que l'opérateur de mutation est utilisé.

2.4 L'opérateur de mutation

Cet opérateur consiste à changer la valeur allélique d'un gène avec une probabilité *pm* très faible, généralement comprise entre 0.01 et 0.001. On peut aussi prendre *pm = 1 /*

lg où lg est la longueur de la chaîne de bits codant notre chromosome. Une mutation consiste simplement en l'inversion d'un bit (ou de plusieurs bits, mais vu la probabilité de mutation c'est extrêmement rare) se trouvant en un locus bien particulier et lui aussi déterminé de manière aléatoire; on peut donc résumer la mutation de la façon suivante :

On utilise une fonction censée nous retourner *true* avec une probabilité *pm* .

Pour chaque locus **faire**

Faire appel à la fonction

Si cette fonction nous renvoie *true* **alors**

on inverse le bit se trouvant à ce locus

Fin Si

Fin Pour

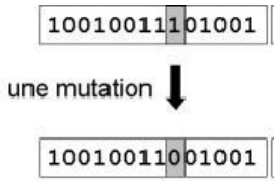


Figure IV.7: Mutation

L'opérateur de mutation modifie donc de manière complètement aléatoire les caractéristiques d'une solution, ce qui permet d'introduire et de maintenir la diversité au sein de notre population de solutions.

Cet opérateur joue le rôle d'un "élément perturbateur", il introduit du "bruit" au sein de la population. Cet opérateur dispose de 4 grands avantages :

• Il garantit la diversité de la population, ce qui est primordial pour les algorithmes génétiques.

• Il permet d'éviter un phénomène connu sous le nom de ***dérive génétique***. On parle de dérive génétique quand certains gènes favorisés par le hasard se répandent au détriment des autres et sont ainsi présents au même endroit sur tous les chromosomes. Le fait que l'opérateur de mutation puisse entraîner de manière aléatoire des changements au niveau de n'importe quel locus permet d'éviter l'installation de cette situation défavorable.

• Il permet de limiter les risques d'une convergence prématurée causée par exemple par une méthode de sélection élitiste imposant à la population une pression sélective trop forte. En effet, dans le cas d'une convergence prématurée on se retrouve avec une

population dont tous les individus sont identiques mais ne sont que des optimums locaux. Tous les individus étant identiques, le croisement ne changera rien à la situation. En effet, l'échange d'informations par crossover entre des individus strictement identiques est bien sûr totalement sans conséquences; on aura beau choisir la méthode de croisement qu'on veut on se retrouvera toujours à échanger des portions de chromosomes identiques et la population n'évoluera pas. L'évolution se retrouvant bloquée on n'attendra jamais l'optimum global. La mutation entrainant des inversions de bits de manière aléatoire permet de réintroduire des différences entre les individus et donc de nous extirper de cette situation.

Il est quand même utile de garder à l'esprit que ceci n'est pas une solution "miracle" et qu'il est bien entendu plus intelligent de ne pas utiliser de méthodes de sélection connues pour entrainer ce type de problème.

• La mutation permet d'atteindre la propriété d' ***ergodicité***.

L'ergodicité est une propriété garantissant que chaque point de l'espace de recherche puisse être atteint. En effet, une mutation pouvant intervenir de manière aléatoire au niveau de n'importe quel locus, on a la certitude mathématique que n'importe quel permutation de notre chaîne de bits peut apparaître au sein de la population et donc que tout point de l'espace de recherche peut être atteint. Grâce à cette propriété on est donc sûr de pouvoir atteindre l'optimum global. On notera que la mutation règle donc le problème exposé à la fin du Section sur le croisement.

2.5 L'opérateur de remplacement

Cet opérateur est le plus simple, son travail consiste à réintroduire les descendants obtenus par application successive des opérateurs de sélection, de croisement et de mutation (la population P') dans la population de leurs parents (la population P). Ce faisant il vont remplacer une certaine proportion de

ceux-ci, proportion pouvant bien sûr être choisie. Le rapport entre le nombre d'individus nouveaux allant être introduits dans la population P et le nombre d'individus de cette population est connu sous le nom de ***génération gap***.

On trouve essentiellement deux méthodes de remplacement différentes :

• Le ***remplacement stationnaire :*** dans ce cas, les enfants remplacent automatiquement les parents sans tenir compte de leurs performances respectives, et le nombre d'individus de la population ne varie pas

tout au long du cycle d'évolution simulé, ce qui implique donc d'initialiser la population initiale avec un nombre suffisant d'individus. Cette méthode peut être mise en œuvre de deux façons différentes :

• La première se contente de remplacer la totalité de la population P par la population P', cette méthode est connue sous le nom de ***remplacement générationnel*** et on a donc une génération gap qui vaut 1.

• La deuxième méthode consiste à choisir une certaine proportion d'individus de P' qui remplaceront leurs parents dans P (proportion égale à 100 % dans le cas du remplacement générationnel. Ce type de remplacement engendre une population ayant une grande variation et de se fait favorise la dérive génétique qui se manifeste d'autant plus que la population est de petite taille.

De plus dans bien des cas, étant donné que même un enfant ayant une faible performance remplace forcement un parent, on n'atteint pas la meilleure solution mais on s'en approche seulement.

• Le ***remplacement élitiste*** : dans ce cas, on garde au moins l'individu possédant les meilleures performances d'une génération à la suivante. En général, on peut partir du principe qu'un nouvel individu (enfant) prend place au sein de la population que s'il remplit le critère d'être plus performant que le moins performant des individus de la population précédente. Donc les enfants d'une génération ne remplaceront pas nécessairement leurs parents comme dans le remplacement stationnaire et par la même la taille de la population n'est pas figée au cours du temps.

Ce type de stratégie améliore les performances des algorithmes évolutionnaire dans certains cas. Mais présente aussi un désavantage en augmentant le taux de convergence prématuré.

Néanmoins, des implémentations plus fines procèdent de manière différente. Dans ce cas là, le taux de remplacement n'est pas de 100 %, la taille de la population augmente donc au cours des générations successives, on dit qu'il y a ***overcrowding.*** Il faut donc trouver un moyen pour sélectionner les parents qui seront supprimés, qui vont mourir. De Jong a proposé la solution suivante : imaginons qu'on veuille remplacer 30 % des parents, soit *np* le nombre de parents correspondants à ce pourcentage, on remplacera les *np* parents les plus proches de leurs descendants de P'. Cette méthode permet donc premièrement de maintenir la diversité et deuxièmement d'améliorer la fitness globale de la population.

Cours du MIT OpenCourseWare – Optimization Methods

https://ocw.mit.edu/courses/15-053-optimization-methods-in-management-science-fall-2016/

Boyd, S. & Vandenberghe, L.*Convex Optimization*Cambridge University Press, 2004.

Nocedal, J., & Wright, S. J. *Numerical Optimization* Springer, 2e édition, 2006.

Bertsekas, D. P.*Nonlinear Programming*Athena Scientific, 2e édition, 1999.

Chong, E. K. P. & Zak, S. H. *An Introduction to Optimization*Wiley, 4e édition, 2013.

Spall, J. C. *Introduction to Stochastic Search and Optimization*
Wiley, 2003.

Gendreau, M., & Potvin, J.-Y. (Eds.)*Handbook of Metaheuristics*Springer, 3e édition, 2019.