# ***Module : Optimisation pour Master 1 Automatique et systèmes***

***Objectifs :*** Maîtriser les techniques d'optimisations rencontrées dans l'industrie et la direction de grands systèmes de production. De plus aider à la prise de décision pour avoir des performances maximales.

| **Connaissances préalables :** Mathématiques. | ***Contenu de la matière :*** *http://eln.univ-batna2.dz/optimisation-0* | |
|---|---|---|
| ***Chapitre* 1**: Rappels mathématiques (Convexité, Minimum, Gradient) *(2S)* | ***Chapitre* 3**: Optimisation sans contraintes méthodes globales *(3S)* | ***Chapitre* 5**: Programmation non linéaire *(4S)* |
| ***Chapitre* 2**: Optimisation sans contraintes méthodes locales *(3S)* | ***Chapitre* 4**: Programmation linéaire *(3S)* | |

***Références:*** Stephen Boyd, «*Convex Optimization*», Cambridge University Press, 2004. Michel Bierlaire, «*Optimization : principles and algorithms* », EPFL, 2015. Jean-Christophe Culioli, *« Introduction à l'optimisation »,* Ellipses, 2012. Rémi Ruppli, *« Programmation linéaire : Idées et méthodes »*, Ellipses, 2005. Pierre Borne, *« Programmation linéaire et applications »* , Technip, 2004.

## Chapitre 1 : Rappels mathématiques et notions de base

### 1.1 Les fonctions scalaires à une seule variable

Définir une fonction $f$ sur un domaine de définition non vide $D_f$ de $\mathbb{R}$, c'est faire correspondre à tout $x \in D_f$ un réel $y = f(x)$ appelé image de $x$. La représentation graphique de $f$ dans le plan $(O, \vec{\imath}, \vec{\jmath})$ est l'ensemble $C_f$ des points de coordonnées $(x, f(x))$.

$$f: D_f \to \mathbb{R}$$
$$x \to f(x)$$

#### 1.1.1 Fonctions majorées, minorées, bornées

- $f$ est majorée ($f(D_f)$ *admet une borne supérieure dans* $\mathbb{R}$) $\Leftrightarrow (\exists M \in \mathbb{R}: M \geq f(x), \forall x \in D_f)$.
- $f$ est minorée ($f(D_f)$ *admet une borne inférieure dans* $\mathbb{R}$) $\Leftrightarrow (\exists m \in \mathbb{R}: m \leq f(x), \forall x \in D_f)$
- $f$ est bornée $\Leftrightarrow (\exists M > 0: |f(x)| \leq M, \ \forall x \in D_f)$

#### 1.1.2 Dérivée et sens de variation d'une fonction

Soit $f$ une fonction définie et dérivable sur un intervalle ouvert *I*

- $f$ est constante sur *I* si et seulement si $f'(x) = \frac{df(x)}{dx} = 0, \forall x \in I$.
- $f$ est croissante sur *I* si et seulement si $f'(x) > 0, \forall x \in I$.
- $f$ est décroissante sur *I* si et seulement si $f'(x) < 0, \forall x \in I$

#### 1.1.3 Condition suffisante d'extremum local

$f$, $f' = \frac{df}{dx}$ $et$ $f'' = \frac{d^2f}{dx^2}$ sont continues sur $]a, b[$, si en $x_0 \in ]a, b[$, on a $f'(x_0) = 0$ $et$ $f''(x_0) \neq 0$

La fonction $f$ présente un extremum local en $x_0$. Ce dernier est associer à un maximum $M = f(x_0)$ si $f''(x_0) < 0$, correspond à un minimum $m = f(x_0)$ si $f''(x_0) > 0$

### 1.1.4 Convexité et concavité d'une fonction

Une fonction $f$ définie sur un intervalle *I* est dite convexe sur *I* si, pour tout deux points $A(a, f(a))$ et $B(b, f(b))$, le graphe de $f$ entre $a$ et $b$ se trouve sous le segment de droite reliant [*AB*]. Réciproquement la fonction est concave si le graphe de $f$ entre $a$ et $b$ se trouve au-dessus du segment [*AB*]. A noter que si $f$ est convexe alors $-f$ est concave. Soit $f$ une fonction au moins est deux fois dérivable sur $I$, alors :

- $f$ est convexe (admet un minimum) sur *I* si, et seulement si, $f'$ est croissante ($f''$ positive).
- $f$ est concave (admet un maximum) sur *I* si, et seulement si, $f'$ est décroissante ($f''$ négative).

## 1.2 Fonction scalaire de plusieurs variables

### 1.2.1 Gradient

Soit $f: \mathbb{R}^n \to \mathbb{R}$ une fonction différentiable dont les dérivées partielles $\partial f / \partial x_i$ existent pour tout $i$. La fonction $\nabla f(x): \mathbb{R}^n \to \mathbb{R}^n$ est appelée le gradient de $f$ au point ou vecteur $x = (x_1, \cdots, x_n)$. Ce vecteur est l'extension de la dérivée. Il détermine la direction de la plus forte pente (sens croissant de la fonction $f$).

$$\nabla f(x) = \begin{pmatrix} \dfrac{\partial f(x)}{\partial x_1} \\ \vdots \\ \dfrac{\partial f(x)}{\partial x_n} \end{pmatrix}$$

### 1.2.2 Matrice Hessienne

Soit $f: \mathbb{R}^n \to \mathbb{R}$ une fonction deux fois différentiable. La fonction notée $\nabla^2 f(x): \mathbb{R}^n \to \mathbb{R}^{n \times n}$ est appelée matrice hessienne ou hessien de $f$. La matrice hessienne est toujours symétrique. Cette matrice est l'extension de la dérivée deuxième d'une fonction à une seule variable.

$$\nabla^2 f(x) = \begin{pmatrix} \dfrac{\partial^2 f(x)}{\partial x_1^2} & \dfrac{\partial^2 f(x)}{\partial x_1 \partial x_2} & \cdots & \dfrac{\partial^2 f(x)}{\partial x_1 \partial x_n} \\ \dfrac{\partial^2 f(x)}{\partial x_2 \partial x_1} & \dfrac{\partial^2 f(x)}{\partial x_2^2} & \cdots & \dfrac{\partial^2 f(x)}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial^2 f(x)}{\partial x_n \partial x_1} & \dfrac{\partial^2 f(x)}{\partial x_n \partial x_2} & \cdots & \dfrac{\partial^2 f(x)}{\partial x_n^2} \end{pmatrix}$$

### 1.2.3 Convexité et concavité par le Hessien

Soit $f: X \subseteq \mathbb{R}^n \to \mathbb{R}$ une fonction continue et deux fois différentiable sur un l'ensemble *X*; alors, elle est *:*

| Convexe | Strictement convexe | Concave | Strictement concave |
|---|---|---|---|
| $\forall\, x \in X: \nabla^2 f(x)$ semi − définie positive | $\forall\, x \in X: \nabla^2 f(x)$ définie positive | $\forall\, x \in X: \nabla^2 f(x)$ semi − définie négative | $\forall\, x \in X: \nabla^2 f(x)$ Définie négative |

Une Matrice $A \in \mathbb{R}^{n \times n}$ est semi-définie positive (notée $A \succcurlyeq 0$) si ses valeurs propres $\lambda_i$ sont positives ou nulle. Les valeurs $\lambda_i$ sont les $n$ racines du polynôme caractéristique: $\det(\lambda I_n - A) = 0$

Soit $f(x)$ une fonction de $\mathbb{R}^n \to \mathbb{R}$ avec les points stationnaires (critiques) $x^*$ tels que $\nabla f(x^*) = 0$. Alors

Si $\nabla^2 f(x^*)$ définie positive $(\nabla^2 f(x^*) \succ 0)$: $x^*$ est un minimum local de $f$

Si $\nabla^2 f(x^*)$ définie négative $(\nabla^2 f(x^*) \prec 0)$: $x^*$ est un maximum local de $f$

Si $\nabla^2 f(x^*)$ non- définie $(\nabla^2 f(x^*) = 0)$: $x^*$ est un point de selle (ou point-col)

Si $\nabla^2 f(x)$ pour tout $x$ est Semi-définie positive $(\nabla^2 f(x^*) \succcurlyeq 0)$: $x^*$ est un minimum global de $f$

Si $\nabla^2 f(x)$ pour tout $x$ est Semi-définie négative $(\nabla^2 f(x^*) \preccurlyeq 0)$: $x^*$ est un maximum global de $f$

Si $\nabla^2 f(x)$ quelconque (cas général): $x^*$ On ne peut pas vérifier que $x^*$ est un minimum global

# Chapitre2. Optimisation sans contraintes

## 2.1 Définition du problème d'optimisation

L'optimisation est une branche des mathématiques et de l'informatique. Dans la quelle on cherche à modéliser, à analyser et à résoudre graphiquement, analytiquement ou numériquement les problèmes qui consistent à minimiser ou maximiser une fonction sur un ensemble. Les solutions trouvées satisfaisant un objectif quantitatif tout en respectant d'éventuelles contraintes. Si les inconnus sont des entiers on parle d'optimisation combinatoire, si c'est des fonctions on parle d'une commande optimale et si c'est des réels on parle d'optimisation continue. La formulation mathématique d'un Problème d'Optimisation (*PO*) est :

$$\begin{matrix}\min_{x\in\mathbb{R}^n} f(x) \\ ou \\ \max_{x\in\mathbb{R}^n} f(x)\end{matrix} \quad sous\ contraintes \begin{cases} C_E(x) = 0 \\ C_I(x) \leq 0 \\ x \in X \end{cases} \rightarrow Formulation\ standard\ du\ problème\ noté\ (PO)$$

**Notations**

$x : n$ Variables ou paramètres ou inconnues → vecteur de $\mathbb{R}^n$

$f$ **:** Critère ou fonction coût ou fonction objectif →fonction de $\mathbb{R}^n$ dans $\mathbb{R}$ tel que $x \in \mathbb{R}^n \rightarrow f(x) \in \mathbb{R}$

$C_E(x) : p$ Contraintes d'égalité→fonction de $\mathbb{R}^n$dans $\mathbb{R}^p$ tel que $x \in \mathbb{R}^n \rightarrow C_E(x) \in \mathbb{R}^p$

$C_I(x) : q$ Contraintes d'inégalité→fonction de $\mathbb{R}^n$ dans $\mathbb{R}^q$ tel que $x \in \mathbb{R}^n \rightarrow C_I(x) \in \mathbb{R}^q$

$X$ : Ensemble convexe (pour min) ou concave (pour max) $X \subseteq \mathbb{R}^n$→valeurs admissibles des variables $x$

Si $x^*$ est optimale pour avoir le $f(x^*) = \min f(x)$ alors il l'optimale pour avoir $-f(x^*) = \max(-f(x))$

## 2.2 Optimisation sans contraintes

Cette optimisation consiste à chercher en absence de contraintes $C_E(x)\ et\ C_I(x)$ la ou le (s) solution(s) $x^*$ qui minimise la fonction objective $f(x)$. Plusieurs méthodes existent, chacune a des avantages et des inconvénients. Dans ce qui suit on étudier le cas d'une minimisation, la maximisation se fait de manière analogue. Ci-après quelques algorithmes numériques qui sont de types itératifs appelés méthodes de descentes. Leur principe consiste à faire les itérations telles : $x_{k+1} = x_k - \lambda_k \cdot d_k$ tout en assurant $f(x_{k+1}) < f(x_k)$. Le vecteur $d_k$ est la direction de descente en $x_k$. Le scalaire $\lambda_k$ est le pas à l'itération $k$. Ces méthodes itératives nécessitent une condition initiale $x_0$ et un critère d'arrêt d'itération. Des tests de convergence vers la solution optimale sont appliqués à $f(x)$ dans l'intervalle des valeurs admissibles.

### 2.2.1 Méthodes du gradient

L'algorithme du gradient utilise la direction de descente $d_k = \nabla f(x_k)$ est procède à une amélioration successive afin de s'approcher de la solution optimale $x^*$ qui donne le minimum $f(x^*)$ de la fonction $f(x)$.

$$Algorithme\ gradient : \begin{cases} Initialisation\ k = 0 & Itération & Critère\ d'arrêt \\ Choix\ de\ x_0, \lambda > 0\ et\ \varepsilon & x_{k+1} = x_k - \lambda \cdot \nabla f(x_k) & \begin{matrix} Si\ \|x_{k+1} - x_k\| < \varepsilon \\ ou\ si\ \lambda \cdot \|\nabla f(x_k)\| < \varepsilon \end{matrix} \end{cases}$$

Avec $\varepsilon$ est un réel positif (petit) que nous devons fixer, il représente la précision désirée. Le pas de la descente $\lambda$ est un réel strictement positif, il détermine la vitesse et la façon de convergence. Si $\lambda = \lambda_0$ est fixe (la méthode du gradient à pas fixe). Si $\lambda = \lambda_k$ change à chaque itération $k$ (gradient est à pas optimal).

L'utilisation de l'algorithme du gradient présente des points forts et des points faibles, on peut citer :

- L'algorithme très facile à mettre en œuvre et nécessite de nombreuses itérations pour converger.
- L'inconvénient majeur apparait lorsque la direction du gradient est presque orthogonale à la direction menant au minimum. Alors apparition du comportement « zig-zag » et progression lente.
- Un $\lambda$ plus petit atténue les zigzags des itérés mais augmente le nombre d'itérations. Un pas trop grand fait diverger la méthode. La recherche du $\lambda_k$ optimal, peut se révéler très longue. Inversement, utiliser un pas $\lambda$ fixe peut conduire à de mauvais résultats.
- Difficulté de convergence lorsque le minimum de la fonction se trouve au fond d'une vallée étroite.

### 2.2.2 Méthode du gradient conjugué

Cette méthode s'applique à la recherche du minimum des fonctions qui s'écrivent sous la forme quadratique : $f(x) = \frac{1}{2}x^T Ax + B^T x + C$. Avec la matrice $A$ est symétrique et définie positive.

***Définition des Directions conjuguées :*** Des vecteurs $d_1, d_2, \cdots, d_k$ de $\mathbb{R}^n$ sont dits $A$-conjugués (ou conjugués 2 à 2 par rapport à $A$,) si : $(d_i, d_j)_A = d_i^T A d_j = 0 \qquad \forall i \neq j$

La notion de conjugaison équivaut à la notion d'orthogonalité du produit scalaire associé à la matrice $A$.

L'idée de la méthode est de construire itérativement des directions $d_1, d_2, \cdots, d_k$ mutuellement conjuguées. A chaque étape $k$ la direction $d_k$ est obtenue comme combinaison linéaire du gradient en $x_k$ et de la direction précédente $d_{k-1}$. Les coefficients étant choisis de telle manière que $d_k$ soit conjuguée avec toutes les directions précédentes. L'algorithme prend la forme suivante

$$\begin{array}{lll} Initialisation\ k = 0 & Itération & Critère\ d'arrêt \\ x_0\ et\ \varepsilon & \begin{cases} x_{k+1} = x_k - \lambda_k \cdot d_k \\ \lambda_k = \dfrac{\nabla f(x_k)^T \cdot d_k}{d_k^T A d_k} \\ d_{k+1} = -\nabla f(x_{k+1}) + \beta_k d_k \\ \beta_k = \dfrac{\nabla f(x_{k+1})^T \cdot \nabla f(x_{k+1})}{\nabla f(x_k)^T \cdot \nabla f(x_k)} \end{cases} & \\ d_0 = -\nabla f(x_0) & & \|x_{k+1} - x_k\| < \varepsilon \end{array}$$

L'algorithme du gradient conjugué consiste à construire simultanément ces directions conjuguées par le procédé de Gram-Schmidt. L'algorithme du gradient conjugué converge en au plus $n$ itérations

### 2.2.3 Méthode de Newton

L'approche applique la méthode de Newton-Raphson pour la recherche des racines à la fonction $\nabla f(x)$.

$$\text{L'algorithme} \begin{cases} Initialisation\ k = 0 & Itération & Critère\ d'arrêt \\ x_0 \text{ au voisinage de } x^* et\ \varepsilon & x_{k+1} = x_k - \left(\nabla^2 f(x_0)\right)^{-1} \cdot \nabla f(x_k) & Si\ \|x_{k+1} - x_k\| < \varepsilon \end{cases}$$

- Fonctionne très bien pour les petits dimensions ($1 \leq n \leq 10$) lorsque le calcule $\left(\nabla^2 f(x_0)\right)^{-1}$ est facile.
- $x_{k+1}$ n'est pas toujours bien définie, possible que $\left(\nabla^2 f(x_0)\right)^{-1}$ n'existe pas (*f* linéaire $\nabla^2 f(x_0) = 0$
- La direction n'est pas toujours une direction de descente.
- Sensibilité au choix du point de départ $x_0$: Si ce point doit être proche de la solution optimale.

### 2.2.4 Méthodes quasi-Newton

La méthode Quasi-Newton est élaborée pour pallier aux inconvénients de la méthode de Newton en termes de rapidité et d'invisibilité de la matrice Hessienne. L'idée est d'approximer $\left(\nabla^2 f(x_0)\right)^{-1}$ par une matrice $S_{k+1}$ qui vérifie l'équation de type sécante $\left(x_{k+1} - x_k = S_{k+1}\left(\nabla f(x_{k+1}) - \nabla f(x_k)\right)\right)$.

La matrice d'approximation $S_{k+1}$ n'est pas forcément symétrique et n'est pas forcément positive. Plusieurs méthodes sont utilisées pour calculer cette approximation. L'algorithme suivant utilise (Broyden).

$$\begin{array}{ccc} Initialisation\ k = 0 & & Itérations \\ x_0, \varepsilon\ et & & \begin{cases} x_{k+1} = x_k - S_k \cdot \nabla f(x_k) \\ S_{k+1} = S_k + \dfrac{(d_k - S_k \cdot y_k) \cdot (d_k - S_k \cdot y_k)^T}{(d_k - S_k \cdot y_k)^T \cdot y_k} \end{cases} et \begin{cases} d_k = x_{k+1} - x_k \\ y_k = \nabla f(x_{k+1}) - \nabla f(x_k) \end{cases} \\ S_0 = \left(\nabla^2 f(x_0)\right)^{-1} ou\ I & & \end{array}$$

- Evite le calcul (coûteux) des matrices $\nabla^2 f(x_k)$ et $\left(\nabla^2 f(x_k)\right)^{-1}$ à chaque itération.
- Plus robustes (peu sensible) par rapport au du point de départ $x_0$.

### 2.2.5 Méthode de Levenberg-Marquardt

L'algorithme de Levenberg-Marquardt *(LM)* est utilisé pour la résolution des problèmes de régression non-linéaire des moindres carrés. La fonction à minimiser a en général la forme suivante

$$f(x) = \frac{1}{2} r(x) \cdot r(x)' = \frac{1}{2} \sum_{i=1}^{m} (g(t_i, x) - y_i)^2 = \frac{1}{2} \sum_{i=1}^{m} (g_i(x) - y_i)^2$$

$$\nabla f(x) = \nabla r(x) \cdot r(x) = \sum_{i=1}^{m} \nabla g_i(x) \cdot (g_i(x) - y_i)$$

Avec $y_i$ est un vecteur de valeur $y_1 \cdots y_m$ et $t$ est un vecteur de valeurs $t_1 \cdots t_m$. Avec une association d'un modèle $y_i = g(t_i, x)$. L'algorithme de Levenberg-Marquardt permet pour un ensemble de paires empiriques $(t_i, y_i)$ de trouver les paramètres de $x$ pour la courbe du modèle $g(t_i, x)$ afin que $f(x)$ la somme des carrés des écarts soit minimisée. Les itérations de l'algorithme Levenberg-Marquardt sont données par :

$$\begin{array}{ccc} Initialisation\ k = 0 & Itération & Critère\ d'arrêt \\ x_0 \text{ au voisinage de } x^*, \lambda\ et\ \varepsilon & H_k = \sum_{i=1}^{m} \nabla g_i(x_k) \cdot \nabla g_i(x_k)^T & Si\ \|\nabla f(x_k)\| < \varepsilon \\ & x_{k+1} = x_k - (H_k + \lambda I)^{-1} \cdot \nabla f(x_k) & \end{array}$$

Si $\lambda$ est très grand, on retombe alors sur la méthode du gradient.

Plus stable que celui de Gauss-Newton, il trouve une solution même s'il est démarré très loin d'un minimum. Cependant, pour certaines fonctions très régulières, il peut converger légèrement moins vite.

## Chapitre3. Optimisation avec contraintes

Dans cette section, on cherche à trouver la solution $x^*$ qui minimise la fonction $f(x)$ sous les contraintes $C$

$$\min_{x\in C\subseteq\mathbb{R}^n} f(x)\ avec \begin{cases} contraintes\ d'égalité\ h=(h_1,h_2,\cdots,h_p): & h_i(x)=0,\quad i=\{1,2,\cdots,p\} \\ \text{ontraintes d'inégalité}\quad g=(g_1,g_2,\cdots,g_q) & : \quad g_j(x)\le 0,\quad j=\{1,2,\cdots,q\} \\ C:\text{ensemble des } x\in\mathbb{R}^n \text{ admissibles } tel\ que\ C=\{x\in\mathbb{R}^n: h_i(x)=0, g_j(x)\le 0\} \end{cases}$$

La difficulté posée par le problème d'optimisation avec contraintes est qu'on doit en même temps minimiser la valeur de la fonction objectif et identifier des points admissibles (qui satisfont les contraintes).

### 3.1 Méthode du gradient projeté

#### 3.1.1 Notion de la projection

Soit $C$ un convexe fermé non vide de $\mathbb{R}^n$. Pour tout $x\in\mathbb{R}^n$, la projection de $x$ sur $C$ et est noté :

$\hat{x}=\Pi_C(x)$ tel que $(\Pi_C(x)=\hat{x}\in C \Leftrightarrow \|x-\hat{x}\|\le\|x-y\| \qquad \forall y\in C)$

**Cas particulier 1:** Si $C$ contraintes de bornes : $C=\{x=(x_1,x_2,\cdots,x_n);\ x_i\ge a_i,\ i\in I,\ x_j\le b_j,\ j\in J\ \}$.

Alors la projection est : $\left(\Pi_C(x)\right)_i=\max\{x_i,a_i\}=\hat{x}^+ \quad ou \quad \left(\Pi_C(x)\right)_i=\min\{x_i,b_j\}=\hat{x}^-$

Si on a les deux contraintes à la fois $C=\{x=(x_1,x_2,\cdots,x_n);\ a_i\le x_i\le b_i,\ i\in I\}$, alors i-ème composante de $\Pi_C(x)=\Pi_C\left((x_1,x_2,\cdots,x_n)\right)$ est noté

$$\left(\Pi_C(x)\right)_i=\hat{x}=\min\{\max\{x_i,a_i\},b_i\}=\begin{cases} a_i & si\ x_i\le a_i \\ x_i & a_i\le x_i\le b_i \\ b_i & si\ x_i\ge b_i \end{cases}$$

**Cas particulier 2:** si $C$ est de type contraintes linéaires tel que: $C=\{x\in\mathbb{R}^n: Ax-B=0\}$ avec $A$ de rang maximal, alors la projection $\hat{x}$ d'un point $x_0$ sur $C$ satisfait $\min_{x\in C}\frac{1}{2}\|x_0-x\|^2$. Elle est égale à :

$$\Pi_C(x_0)=\hat{x}=(I-A^T(AA^T)^{-1}A)\cdot x_0+A^T(A\cdot A^T)^{-1}B$$

#### 3.1.2 Le Principe de la méthode du gradient projeté

L'utilisation de l'algorithme du gradient sans contraint dans notre cas (avec contraintes) ne garantit pas que les valeurs calculées de $x_k$ reste sur $C$ ($x_k$ inadmissible). Il est donc nécessaire de se ramener sur $C$ quand on obtient un point non admissible. Alors on projette à chaque itération celui-ci sur $C$ à l'aide de l'opérateur noté $\Pi_C(\cdot)$: $\mathbb{R}^n\to C$ afin de génère un point admissible $\hat{x}_k$ à partir du point $x_k$ tel que :

| $Initialisation\ k=0$ | $Itération$ | $Critère\ d'arrêt$ |
|---|---|---|
| $x_0\ et\ \varepsilon$ | $d_k=\nabla f(x_k)$<br>$x_{k+1}=\Pi_C(x_k-\lambda\cdot d_k)$ | $Si\ \|x_{k+1}-x_k\|<\varepsilon$ |

Dans le cas des contraintes linéaires d'égalités : $C=\{x\in\mathbb{R}^n: Ax-B=0\}$ l'algorithme devient :

| $Initialisation\ k=0$ | $Itération$ | $Critère\ d'arrêt$ |
|---|---|---|
| $x_0\ et\ \varepsilon$ | $d_k=x_k-\lambda\cdot\nabla f(x_k)$<br>$x_{k+1}=(I-A^T(AA^T)^{-1}A)\cdot d_k+A^T(A\cdot A^T)^{-1}B$ | $Si\ \|x_{k+1}-x_k\|<\varepsilon$ |

L'algorithme du gradient projeté est difficile à mettre en œuvre, car le calcul de la projection est souvent aussi difficile que le problème initial. Cependant l'algorithme peut être utilisé pour des contraintes simples.

### 3.2 Méthode de Lagrange-Newton pour des contraintes d'égalité

L'algorithme Lagrange-Newton est basé sur la linéarisation d'équations. Considérons le cas particulier

$$\min_{x\in\mathbb{R}^n} \frac{1}{2}x^TQx + C^Tx + D \;\; Avec\; contrainte\;\; \{Ax - b = 0\}\;\; et\; Q\text{: définie positive}$$

$$\begin{cases} \nabla\left(\frac{1}{2}x^TQx + C^Tx + D + \lambda^T(Ax-b)\right) = 0 \\ Ax - b = 0 \end{cases}\Bigg|_{x=x^*\; et\;\; \lambda=\lambda^*} \Rightarrow \begin{pmatrix} Qx^* + C + \mathrm{A}^T\lambda^* \\ Ax - b \end{pmatrix} = 0$$

$$\begin{pmatrix} Q & \mathrm{A}^T \\ A & 0 \end{pmatrix}\begin{pmatrix} x^* \\ \lambda^* \end{pmatrix} = \begin{pmatrix} -C \\ b \end{pmatrix}$$

Si les contraintes sont indépendantes. Alors le système linéaire admet une solution optimale unique $x^*\; et\; \lambda^*$.

### 3.3 Méthode de Newton projetée (pour des contraintes de borne)

Cette méthode est analogue à celle développée dans la méthode du gradient projeté : puisque elle fait la projection des itérés successifs sur l'ensemble des contraintes de bornes tel que :

$$C = \{x \in \mathbb{R}^n : a \le x \le b\}\; ou\; C = \{x = (x_1, x_2, \cdots, x_n);\; x_i \ge a_i, \quad i \in I, \quad x_j \le b_j, \quad j \in J\;\}$$

En exploitant la projection sur des contraintes de borne l'algorithme de la méthode de Newton projeté est :

| $Initialisation\;\; k = 0$ | $Itération$ | $Critère\; d'arrêt$ |
|---|---|---|
| | $d_k = x_k - \left(\nabla^2 f(x_0)\right)^{-1} \cdot \nabla f(x_k)$ | |
| $x_0\; et\; \varepsilon$ | $x_{k+1} = \Pi_C(d_k)\; avec$ | $Si\; \|x_{k+1} - x_k\| < \varepsilon$ |
| | $\left(\Pi_C(d_k)\right)_i = \min\{\max\{(d_k)_i, a_i\}, b_i\}$ | |

### 3.4 Méthode de pénalisation

Le principe de cette méthode est de transformer le problème sous contraintes en un problème sans contraintes. La méthode associée une pénalité $P(x)$ très grande à tout $x$ qui n'est pas une solution admissible. Le principe de la méthode est formulé comme suit : On veut

$$\min_{x\in C\subseteq\mathbb{R}^n} f(x)\; sous\; contraintes\; C = \{x \in \mathbb{R}^n : h_i(x) = 0, g_j(x) \le 0\}$$

Alors, on construit le problème sans contraintes :

$$\min_{x\in\mathbb{R}^n}\left(f(x) + \alpha\cdot P(x)\right)\; où\quad \alpha \in \mathbb{R}^+\; et\; P(x)\text{: } \mathbb{R}^n \to \mathbb{R}\; vérifié \begin{cases} P(x) : Continue \\ P(x) \ge 0 \quad \forall x \in \mathbb{R}^n \\ P(x) = 0 \Leftrightarrow x \in C \end{cases}$$

Plusieurs pénalités peuvent être utilisées on peut citer celle de Courant-Beltrami définie comme suit

$$P(x) = \sum_{i=1}^{p}\left(g_i^+(x)\right)^2 + \sum_{i=1}^{q}\left(h_i(x)\right)^2 \quad où \quad g_i^+(x) = max\{0, g_i(x)\}$$

Intuitivement pour $\alpha$ grand, toute solution (optimale) de $\min_{x\in\mathbb{R}^n}\left(f(x) + \alpha\cdot P(x)\right)$ à une pénalité nulle c'est-à-dire est réalisable pour $\min_{x\in C\subseteq\mathbb{R}^n} f(x)$. Pour chercher la valeur de $\alpha$, on se donne une suite $\alpha_k$ (positif) strictement croissante ($\alpha_{k+1} > \alpha_k > 0$) avec ($\lim_{k\to+\infty} \alpha_k = +\infty$).L'algorithme de la méthode de pénalité est

| $Initialisation\ k = 0$ | $Itération$ | $Critère\ d'arrêt$ |
|---|---|---|
| $x_0,\quad \alpha_0\ (ex = 1),$<br>$\varepsilon,\ \beta(grand\ ex = 10\ ou\ 100)$ | $x_{k+1}^{*} solution\ du \min_{x\in\mathbb{R}^n}\left(f(x_k) + \alpha_k \cdot P(x_k)\right)$<br>$avec\ x_k\ état\ initial$<br>$\alpha_{k+1} = \beta \cdot \alpha_k$ | $Contraintes\ satisfaitent$<br>$ou\ \|x_{k+1} - x_k\| < \varepsilon$ |

On peut utiliser les méthodes d'optimisation sans contraintes pour résoudre le problème de minimisation $\min_{x\in\mathbb{R}^n}\left(f(x_k) + \alpha_k \cdot P(x_k)\right)$. (BFGS, gradient conjugué,...). La suite $x_k$ converge vers une solution optimale

### 3.5 Dualité, fonction duale, problème dual et Méthode d'Uzawa

#### 3.5.1 Dualité, fonction duale et problème dual

$\min_{x\in C} f(x)$ sous $C = \left\{x \in \mathbb{R}^n : h_i(x) = 0, g_j(x) \leq 0\right\}$, est appelé problème primal, Le Lagrangien $L$ est :

$$L(x,\lambda,\mu) = f(x) + \sum_{i=1}^{p} \lambda_i \cdot h_i(x) + \sum_{j=1}^{q} \mu_j \cdot g_j(x) = f(x) + \lambda \cdot h(x) + \mu \cdot g(x)$$

Si $x^*$ est la solution optimal alors, on définit la fonction duale $q(\lambda,\mu)$: $\mathbb{R}^p \times \mathbb{R}^{q+} \to \mathbb{R}$ de variables duales $\lambda$ et $\mu$ associé au problème primal par : $q(\lambda,\mu) = L(x^*,\lambda,\mu) = \min_{x\in\mathbb{R}^n} L(x,\lambda,\mu)$.

Le problème dual par opposition au problème primal est définit comme suit :

$$\max_{(\lambda,\mu)\in X_q} q(\lambda,\mu) = \max_{(\lambda,\mu)\in X_q}\left(\min_{x\in\mathbb{R}^n} L(x,\lambda,\mu)\right) \quad avec\ \ avec\ \mu > 0$$

Dans un ensemble convexe de fonction est de contraintes, si un problème primal de minimisation admet une solution minimale, alors le problème dual admet une solution maximale. La valeur maximale du problème dual est la même que la valeur minimale du problème de minimisation primal. Dans ce cas on a :

$$\max_{(\lambda,\mu)\in X_q} q(\lambda,\mu) = \max_{(\lambda,\mu)\in X_q}\left(\min_{x\in\mathbb{R}^n} L(x,\lambda,\mu)\right) = L(x^*,\lambda^*,\mu^*) = \min_{x\in C} f(x) = \min_{x\in\mathbb{R}^n}\left(\max_{(\lambda,\mu)\in X_q} L(x,\lambda,\mu)\right)$$

Lorsque la solution optimale $(x^*,\lambda^*,\mu^*)$ existe, on peut donc résoudre le problème dual de maximisation (resp : minimisation) à la place du problème primal de minimisation (resp : maximisation).

#### 3.5.2 Algorithme d'Uzawa

L'algorithme d'Uzawa, consiste à utiliser l'algorithme du gradient projeté (à pas fixe ou optimal) pour maximiser de manière itérative la fonction duale, tout en tenant compte de la contrainte $\mu > 0$. Alors L'algorithme est formulé selon les équations suivantes :

| $Initialisation\ k = 0$ | $Itération$ | $Critère\ d'arrêt$ |
|---|---|---|
| $x_0, \lambda_0, \mu_0,$<br>$\rho > 0\ et\ \varepsilon > 0$ | $x_k\ solution\ de \min_{x\in\mathbb{R}^n} L(x,\lambda_k,\mu_k)$<br>$calcul\ de\ \lambda_{k+1}\ et\ \mu_{k+1}\ avec$<br>$\lambda_{k+1}^{i} = \lambda_k^{i} + \rho \cdot h_i(x_k)$<br>$\mu_{k+1}^{j} = \Pi_{C\equiv\mu_k^j>0}\left(\mu_k^j + \rho \cdot g_j(x_k)\right) = max\{0, \mu_k^j + \rho \cdot g_j(x_k)\}$ | $\|x_{k+1} - x_k\| < \varepsilon$ |

## Chapitre 4 : Programmation Linéaire (PL)

La programmation linéaire c'est un problème d'optimisation pour laquelle la fonction objectif et les contraintes sont toutes linéaires. De plus, les variables sont supposées être positives (contraintes de non-négativité). Pour une maximisation ou minimisation, le problème peut s'écrire sous la Forme Générale :

$$min \underset{x\in U}{ou} max\, f(x) = c^T \cdot x = \sum_{i=1}^{n} c_i\, x_i \;\; contraintes \;\; U = \left\{ x \in \mathbb{R}^n : \left( g_j(x) = a \cdot x - b \right) \begin{Bmatrix} \leq \\ \geq \\ = \end{Bmatrix} 0 \quad j = 1, \cdots, m \right\}$$

Où $n, m \in \mathbb{N}^*$ $f: \mathbb{R}^n \to \mathbb{R}$ est linéaire sur $\mathbb{R}^n$ et $g_1, \cdots, g_m$sont linéaires définies sur $\mathbb{R}^n$, avec les matrices $c(c_i) \in \mathbb{R}^{n\times 1}$ , $a(a_{ji}) \in \mathbb{R}^{m\times n}$ $et$ $b(b_j) \in \mathbb{R}^{m\times 1}$ sont des constantes connues. $x(x_i) \in \mathbb{R}^{n\times 1}$ et $x \in \mathbb{R}^n$.

### 4.1 La solution géométrique (ou graphique)

La solution graphique, repose sur le fait que $f(x) = c^T \cdot x$ est linéaire donc ses courbes de niveau sont des droites avec un gradient $\nabla f(x) = c$ qui détermine la direction de la plus forte pente (croissante). De plus, les contraintes $(a \cdot x - b = 0)$ forment un polytope. Alors, la solution optimale se trouve sur le ou les sommets de ce polytope qui donnent chacune la valeur maximale ou minimale de la fonction objectif. Les étapes à suivre pour la résolution d'une programmation linéaire avec la méthode graphique sont :

- A partir des contraintes représenter le polytope et la région admissible ou réalisable $U$.
- Déterminer les sommets du polytope $S_1(x_{11}, x_{21}, \cdots, x_{n1}), S_2(x_{12}, x_{22}, \cdots, x_{n2}), S_3(x_{13}, x_{23}, \cdots, x_{n3}), \ldots$
- Évaluer les valeurs $f = f(S_j)$ afin de déterminer le sommet $S_j^*$donnant le minimum ou le maximum.

**Remarque** : Lorsque la fonction objectif est minimale sur un côté du polygone; alors chaque point du côté est une solution. Lorsque la région admissible *U* est non bornée (infinie), la fonction objectif peut posséder : Un minimum et pas de maximum ; Un maximum et pas de minimum ; Ni maximum ni minimum

### 4.2 Méthodes algébrique par base d'un problème sous forme standard

Cette approche permet de parcourir les sommets du polytope de manière algébrique en utilisant certaines solutions réalisables (dites de base). Elle nécessite d'écrire le problème sous sa forme standard ou canonique

***La forme standard*** : Elle a des contraintes d'égalités et sur des variables positives

$$min \underset{x\in\mathbb{R}^n}{ou} max\, f(x) = c^T\, x = \sum_{i=1}^{n} c_i\, x_i \;\; contraintes \begin{cases} a\, x - b = 0 \\ x \geq 0 \end{cases} \Leftrightarrow \begin{cases} g_j(x) = \sum_{i=1}^{m} a_{ji} x_i - b_j = 0 & j = 1, \cdots, m \\ x_i \geq 0, & i = 1, \cdots, n \end{cases}$$

***La forme canonique :*** Un problème de programmation linéaire sous sa forme canonique s'écrit :

$$\max_{x\in\mathbb{R}^n} f(x) = c^T \cdot x = \sum_{i=1}^{n} c_i\, x_i \;\; contraintes \begin{cases} a \cdot x - b \leq 0 \\ x \geq 0 \end{cases} \Leftrightarrow \begin{cases} g_j(x) = \sum_{i=1}^{m} a_{ji} x_i - b_j \leq 0 & j = 1, \cdots, m \\ x_i \geq 0, & i = 1, \cdots, n \end{cases}$$

$$\min_{x\in\mathbb{R}^n} f(x) = c^T \cdot x = \sum_{i=1}^{n} c_i\, x_i \;\; contraintes \begin{cases} a \cdot x - b \geq 0 \\ x \geq 0 \end{cases} \Leftrightarrow \begin{cases} g_j(x) = \sum_{i=1}^{m} a_{ji} x_i - b_j \geq 0 & j = 1, \cdots, m \\ x_i \geq 0, & i = 1, \cdots, n \end{cases}$$

Les règles de transformation permettant le passage d'une forme à une autre sont sur le tableau suivant :

| | | |
|---|---|---|
| $\min_{x\in\mathbb{R}^n} f(y) \Leftrightarrow -\max_{x\in\mathbb{R}^n}(-f(y))$ | $g(y) \le 0 \Leftrightarrow -g(y) \ge 0$ | $g(y) \le 0 \Leftrightarrow \begin{cases} g(y) + e_1 = 0 \\ e_1 \ge 0 \end{cases}$ |
| $g(y) = 0 \Leftrightarrow \begin{cases} g(y) \le 0 \\ g(y) \ge 0 \end{cases}$ | $y \in \mathbb{R} \Leftrightarrow \begin{cases} y = e_1 - e_2 \\ e_1 \ge 0 \\ e_2 \ge 0 \end{cases}$ | $y \ge \ell \Leftrightarrow \begin{cases} y = e_1 + \ell \\ e_1 \ge 0 \end{cases}$ |

Les variables $y$ sont appelées variables de décision. Les variables $e_1, e_2$ sont appelées variables d'écart.

### 4.2.1 Méthodes algébrique par base

Cette approche utilise l'écriture standard. Le vecteur $x$ regroupe les variables de décision et d'écart.

$$\max_{x\in\mathbb{R}^{n+m}} f(x) = c^T \cdot x \;\; contraintes \begin{cases} a \cdot x - b = 0 \\ x \ge 0 \end{cases} \Leftrightarrow \begin{cases} g_j(x) = \sum_{i=1}^{m} a_{ji}x_i - b_j = 0 & j = 1, \cdots, m \\ x_i \ge 0, & i = 1, \cdots, n \end{cases}$$

Avec $rang(a) = m \le n$. On note $a_i$ les colonnes de la matrice $a$.

On choisit $m$ colonnes indépendants parmi $a_1, \cdots, a_n$ afin de construire une base $B$ de $m$ vecteur. Alors

$$\begin{cases} a \equiv (B \quad N), c \equiv (C_B \quad C_N) \\ x \equiv \begin{pmatrix} x_B \\ x_N \end{pmatrix} \end{cases} \Rightarrow \begin{cases} f(x) = c^T \cdot x \Leftrightarrow (C_B^T \quad C_N^T)\begin{pmatrix} x_B \\ x_N \end{pmatrix} = C_B^T \cdot x_B + C_N^T \cdot x_N \\ a \cdot x = b \Leftrightarrow (B \quad N) \cdot \begin{pmatrix} x_B \\ x_N \end{pmatrix} = B \cdot x_B + N \cdot x_N = b \end{cases}$$

Les variables $x_i$ qui sont dans le vecteur $x_B$ sont appelées variables de Base

Les variables $x_i$ qui sont dans le vecteur $x_N$ sont appelées variables hors Base

Avec $x_N = x_N^* = 0$ et puisque $B$ est inversible on peut extraire le vecteur $x_B^*$ tels que

$$x_B^* = B^{-1}b - B^{-1}N \cdot x_N = B^{-1}b - 0 = B^{-1}b = x^B = \begin{pmatrix} x_B^* \\ x_N^* \end{pmatrix} = \begin{pmatrix} x_B^* \\ 0 \end{pmatrix} \; avec \; (x^B\text{:solution de la base } B)$$

Chaque solution de base $x^B$ constitue l'un des sommets du polytope formé par les contraintes du problème de Programmation Linéaire ($PL$). La solution de base $x_B^*$ est réalisable non dégénérée, si elle vérifie les contraintes de non négativité (toutes les composantes de $x_B^*$ sont positives $x_B^* = B^{-1}b \ge 0$).

$$\text{Le nombre des sommets possible est égale à } \Gamma \le C_n^m = \frac{n!}{m!\,(n-m)!}$$

Deux bases (ou sommets) $B_\gamma \in \mathbb{R}^{m\times m}$ et $B_\alpha \in \mathbb{R}^{m\times m}$ sont adjacentes s'ils ont $(m-1)$ colonnes identiques (c-à-dire, ils diffèrent par une des variable). Le passage d'une base à une autre adjacente s'appelle pivotage ; dans lequel une variable base sorte de la base et une autre hors base entre en base.

### 4.2.2 Les étapes de la méthode algébrique par base :

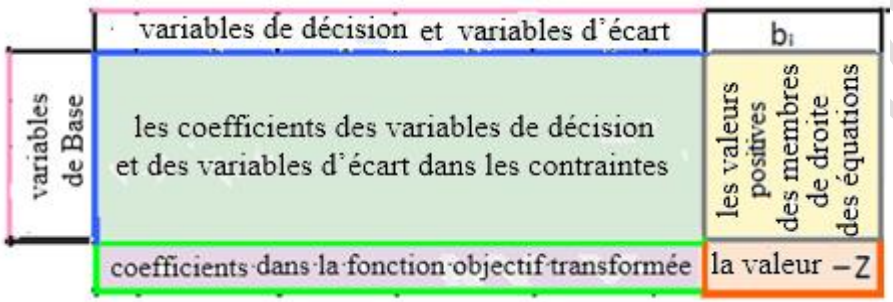
Si le problème d'optimisation linéaire admet une solution $x^*$, elle sera un sommet (≡bases) du polytope $P$ associé aux contraintes. La recherche systématique de la solution optimale suit les étapes suivantes :

- Choisir $m$ colonnes indépendantes parmi les colonnes de $a \rightarrow$ construire une base $B$ possible
- Vérifier que la base est réalisable : $B$ inversible et $x_B^* = B^{-1}b \ge 0 \rightarrow$ Solution de base admissible
- Calculer la valeur du coût associé à la base $B$ : $f = c^T x^B = C_B^T \cdot x_B^* + C_N^T \cdot x_N^* = C_B^T \cdot x_B^*$
- Sélection de la meilleure solution ($f$ maximale-si maximisation et $f$ minimale-si minimisation).

### 4.3 Méthode simplexe (méthode des tableaux)

La méthode simplexe évite le balayage de toutes les bases existantes afin de trouver celle qui maximise ou minimise la fonction objectif. Elle parcoure à partir d'une base initiale admissible $x_0$ les autres sommets adjacent qui maximise ou minimise la fonction $f(x)$. Les étapes à suivre pour cette méthode sont :

1. Identifier la fonction objectif et les variables de décision ensuite écrire le système sous forme standard en indiquant les contraintes de non-négativité et transformer la fonction objectif tels que
   $f(x) = f = Z = c^T \cdot x = c_1x_1 + c_2x_2 + \cdots + c_nx_n$ sous la forme $c_1x_1 + c_2x_2 + \cdots + c_nx_n - Z = 0$
2. A partir des données précédentes construire le tableau initial suivant.




3. Résolution du système par la méthode de Gauss en commençant par trouver le pivot tel que :
   En déterminant la colonne pivot, pour une maximisation (resp : minimisation) celle où la valeur est la plus positive (resp : négative) sur la ligne de Z ;
   En déterminant la ligne pivot, celle où l'on retrouve la valeur minimale du quotient de chaque constante $b_j$ divisée par l'élément positif de la colonne pivot situé sur la même ligne que $b_j$.
   Ensuite, effectuer le pivotage, c'est-à-dire transformer le tableau de façon à obtenir 1 à l'endroit du pivot et obtenir 0 ailleurs dans la colonne pivot.
4. Répéter l'étape 3 jusqu'à ce que tous les éléments de la dernière ligne (Z) soient négatifs ou nuls pour une maximisation (resp : positifs ou nuls pour une minimisation). Dans ce cas la solution est le vecteur se trouvant dans la colonne des $b_j$.

### 4.4 Résolution de problèmes de minimisation par la méthode duale

Si on a un problème de minimisation primal on peut lui associer son problème dual tels que :

Problème linéaire primal

$$\min_{x\in\mathbb{R}^n} f(x) = c^T \cdot x \ \ avec \begin{cases} a \cdot x - b \geq 0 \\ x \geq 0 \end{cases} \Leftrightarrow$$

Problème linéaire Dual

$$\max_{y\in\mathbb{R}^m} q(y) = b^T \cdot y \ \ avec \begin{cases} a^t \cdot y - c \leq 0 \\ y \geq 0 \end{cases}$$

La résolution du problème dual peut être effectuée à l'aide de la méthode du simplexe.

Par le théorème de la dualité, le maximum de la fonction $q$ égale au minimum de la fonction $f$. De plus, nous acceptons sans démonstration que les valeurs de $x$ qui minimisent la fonction $f$ se trouvent avec un signe (-) sur la dernière ligne du tableau simplexe final, sous les variables d'écart.

Dans cette section, on cherche à trouver la solution $x^*$ qui minimise la fonction $f(x)$ sous les contraintes $C$

$$\min_{x\in C\subseteq\mathbb{R}^n} f(x)\ avec \begin{cases} contraintes\ d'égalité\ h=(h_1,h_2,\cdots,h_p): & h_i(x)=0,\quad i=\{1,2,\cdots,p\} \\ \text{ontraintes d'inégalité}\quad g=(g_1,g_2,\cdots,g_q) & : \quad g_j(x)\le 0,\quad j=\{1,2,\cdots,q\} \\ C:\text{ensemble des } x\in\mathbb{R}^n \text{ admissibles } tel\ que\ C=\{x\in\mathbb{R}^n: h_i(x)=0, g_j(x)\le 0\} \end{cases}$$

On dit programmation non linéaire la résolution du problème d'optimisation ci-dessus quand au moins l'une des fonctions $f(x), h_i(x)\ ou\ g_j(x)$ est non linéaire.

### 5.1 Méthode de résolution par substitution

Cette technique de résolution est utilisée lorsqu' il y a des contraintes d'égalité. En effet, si on a $n$ variables et $p$ contraintes d'égalité ; alors à partir des contraintes on peut faire des substitutions successives des inconnus ensuite les remplacer dans la fonction objectif. Dans ce cas cette dernière aura $(n-p)$ variables inconnus au lieu de $n$. On peut ainsi utiliser les techniques d'optimisation non linéaire sans contraintes pour trouver la solution optimale qui minimise ou maximise la fonction objectif.

### 5.2 Méthode du Lagrangien et Conditions d'optimalité Théorème KKT (Karush-Kuhn-Tucker)

Soit $x^*$ un minimum local de $f(x)$ sous les contraintes $C=\{x\in\mathbb{R}^n: h_i(x)=0, g_j(x)\le 0\}$, Alors, La fonction de Lagrange $L$ est constituée de l'objectif $f(x)$ et les contraintes $h_i(x)$ et $g_j(x)$ tels que :

$$L(x,\lambda,\mu)=f(x)+\sum_{i=1}^{q}\lambda_i\cdot h_i(x)+\sum_{j=1}^{p}\mu_j\cdot g_j(x)+=f(x)+\lambda^T\cdot h(x)+\mu^T\cdot g(x)$$

On suppose que les contraintes sont qualifiées (les $\nabla h_i(x)$ et les $\nabla g_i(x)$ sont indépendantes) en $x$. Alors, Si $x^*$ optimal, il existe $\lambda^{*T}=(\lambda_1,\lambda_2,\cdots\lambda_p)\in\mathbb{R}^p\ et\ \mu^T=(\mu_1,\mu_2,\cdots\mu_q)\in\mathbb{R}^{q+}\ \mu_i\ge 0\ \forall i$ tel que :

$$\begin{cases} \nabla L(x^*,\lambda,\mu)=\nabla f(x^*)+\sum_{i=1}^{p}\lambda_i\cdot\nabla h_i(x^*)+\sum_{j=1}^{q}\mu_i\cdot\nabla g_i(x^*)=\nabla f(x^*)+\lambda^T\cdot\nabla h(x^*)+\mu^T\cdot\nabla g(x^*)=0 \\ \mu_j\ge 0 \\ \mu_j g_j(x^*)=0\ \ \forall j=\{1,2,\cdots,q\} \\ h_i(x^*)=0\ \ et\ \ g_j(x^*)\le 0 \end{cases}$$

Ces conditions (KKT) sont appelées conditions de Karush-Kuhn-Tucker. Les $\lambda_i$ et $\mu_i$ sont appelés multiplicateurs de Lagrange. Afin que $f(x)$ admet un minimum il faut que le gradient $\nabla L(x^*,\lambda,\mu)=0$ que

Le hessien : $\nabla^2 L(x^*,\lambda,\mu)=\nabla^2 f(x^*)+\sum_{i=1}^{q}\lambda_i\cdot\nabla^2 h_i(x^*)+\sum_{j=1}^{p}\mu_j\cdot\nabla^2 g_j(x^*)$ est semi − définie positif

Pour un Problème de maximisation en change la somme (+) par soustraction (-)