N° d'ordre : 23/2005-M/MT

MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITÉ DES SCIENCES ET DE LA TECHNOLOGIE HOUARI BOUMEDIENNE

FACULTÉ DES MATHÉMATIQUES

Mémoire présenté pour l'obtention du diplôme de Magister

en Mathématiques

Spécialité : Recherche Opérationnelle

Par

ARRACHE Saïda

THÈME

Accroissement de la vitesse de convergence de la méthode de Frank-Wolfe : Application aux réseaux de trafic urbain.

Soutenu publiquement, le 17/12/2005, devant le jury composé de :

M^r M. AIDER	Professeur	U.S.T.H.B.	Président.
M^r R. OUAFI	Maître de conférences	U.S.T.H.B.	Directeur de thèse.
M^lle I. BOUCHEMAKH	Maître de conférences	U.S.T.H.B.	Examinatrice.
M^r D. CHAABANE	Maître de conférences	U.S.T.H.B.	Examinateur.
M^r M. MOULAI	Maître de conférences	U.S.T.H.B.	Examinateur.

Remerciements

Qu'il me soit permis de remercier M^r^ OUAFI pour m'avoir dirigées tout le long de l'année, pour ses nombreux conseils, son appuie et surtout, d'avoir été à l'origine de ma découverte du monde de la recherche en me transmettant ses connaissances en mathématique. Un grand merci.

Je tiens à remercier M^r^ AIDER qui me fait l'honneur de bien vouloir présider le jury de cette soutenance.

Je remercie également M^elle^ BOUCHEMAKH, M^r^ MOULAI et M^r^ CHAABANE, pour les conseils pratiques et les remarques judicieuses qu'ils m'ont apportés et d'avoir accepté de faire partie du jury.

Je serai toujours reconnaissante à ma famille : « mon père, ma mère, mes frères, ma belle sœur, mes oncles, mes tantes, mes cousins et ma meilleure amie», pour tout le support dont j'ai eu la chance de profiter.

Mes plus vifs remerciements vont également à tous mes amis, pour leurs soutiens moral et leurs encouragements.

Et enfin, par la même occasion, je tiens à exprimer ma profonde gratitude à toutes les personnes qui m'ont aidé à mener à bien ce travail de loin ou de prêt.

Saïda.

TABLE DES MATIERES

Liste des Figures

Liste des courbes

Liste des Tableaux

INTRODUCTION GENERALE

Connue pour être l'une des branches de la recherche opérationnelle, la programmation mathématique est un outil incontournable pour la résolution de certains problèmes engendrés par l'activité économique ou industrielle, admettant plusieurs solutions réalisables. Sa puissance tient lieu du fait qu'elle prend en charge la modélisation et la résolution de ces problèmes. En effet, la résolution par la programmation mathématique peut être vue comme étant une série de tâches qu'il faut effectuer pour bâtir, utiliser et maintenir un système d'aide à la décision s'appuyant sur un programme mathématique. Cette discipline a atteint aujourd'hui un certain degré de maturité, son application est en expansion croissante et se diffuse de plus en plus, dans plusieurs domaines d'intérêt, en l'occurrence: l'affectation des emplois du temps, la collecte des déchets, le raffinage, l'exploration spatiale et les systèmes d'écoulement…etc.

Ce présent travail s'est focalisé sur l'étude de l'une des méthodes de résolution en programmation mathématique, qui est *« la méthode de Frank-Wolfe »*. Cette dernière a fait son apparition en programmation quadratique, aussitôt elle s'est avérée très efficace pour la résolution des problèmes de flot de grandes dimensions, avec des atouts particulièrement intéressants, reposant sur la merveilleuse fusion de la programmation mathématique et la théorie des graphes. La méthode de Frank-Wolfe est réputée pour ses nombreux avantages. En revanche, elle souffre d'une vitesse de convergence très lente ce qui constitue son principal inconvénient. Pour cela, on se propose dans ce mémoire de palier à cet inconvénient tout en préservant les différents avantages de cette méthode. Notre travail est donc, une contribution à l'amélioration de cet algorithme en terme de taux de convergence.

Afin d'accomplir cet objectif, le présent mémoire est organisé comme suit :

Le premier chapitre, est réservé essentiellement à l'étude de la méthode de Frank-Wolfe ; on retrouve également certains concepts liés à celle-ci.

Le deuxième chapitre est consacré à l'étude de la convergence de la méthode de Frank-Wolfe, pour les besoins de ce chapitre, la notion d'opérateurs algorithmiques a été également abordée.

Dans le troisième chapitre, nous passons en revue quelques versions modifiées de l'algorithme de Frank-Wolfe qui ont été élaborées en vue d'améliorer la vitesse de convergence de cette méthode.

Nous nous sommes proposés dans le quatrième chapitre d'atteindre cet objectif en développant de nouvelles variantes palliatives à cet inconvénient, trois variantes ont été présentées à cet effet: la technique du pas élargi " **FWλ**", la variante de Fukushima " **FWF**" et enfin la variante combinée "**FWFλ**".

Du large domaine d'application de la méthode de Frank-Wolfe, nous avons choisi le problème de l'affectation du trafic urbain. Ce dernier, fait l'objet de deux chapitres :
le chapitre cinq est consacré à la présentation et à la modélisation du problème ; sa résolution par les différentes variantes étudiées a été traitée au sixième chapitre.

Enfin, dans le septième chapitre nous avons illustré ce travail par des tests numériques établis sur des problèmes réels et fictifs de l'affectation du trafic urbain, où l'algorithme de Frank-Wolfe et les trois versions modifiées ont été implémentés en langage de programmation C++ sous environnement C++ Builder5 . Une étude comparative a été effectuée, révélant la nette amélioration de la vitesse de convergence de ces nouvelles variantes, par rapport à celle de la méthode de Frank-Wolfe classique, cette comparaison a mis également en évidence la supériorité de la nouvelle variante, la variante combinée "**FWFλ**".

Chapitre I

Méthode de Frank-Wolfe

I. Présentation de la méthode de Frank-Wolfe

I.1. Introduction

Dans ce qui suit, nous présenterons la méthode de Frank-Wolfe [**8**] qui a été développée principalement pour la résolution des problèmes quadratiques, toutefois elle s'est avérée très prometteuse pour la résolution des problèmes de programmation convexe avec contraintes linéaires. Cette méthode est très appréciée dans le domaine du transport. Avant d'entamer sa présentation, nous devons d'abord définir quelques concepts :

I.2. La programmation non linéaire

Un problème de programmation mathématique est dit *non linéaire* si l'objectif, ou au moins une contrainte est *non linéaire.* Il est représenté sous cette forme :

$$(\mathbf{P})\begin{cases} \max f(x) \\ s.c. \\ g_i(x) \le 0, \quad i = 1,...m \\ h_k(x) = 0, \quad k = 1,...p \end{cases}$$

Où, $f(x), g_i(x), i = 1,...m$ et $h_k(x), k = 1,...p$ sont des fonctions continûment différenciables définies sur IR^n à valeurs dans IR. Il s'agit donc de la généralisation au cas non linéaire du problème classique de la programmation linéaire :

$$(\mathbf{P.L})\begin{cases} \max \ cx \\ s.c. \ \ Ax \le b \\ x \ge 0 \end{cases}$$

Les modèles non linéaires sont généralement, beaucoup plus difficiles à résoudre que les modèles linéaires. Nous allons souligner quelques différences avec la programmation linéaire [**16**]:

- Contrairement à la programmation linéaire on peut avoir une solution optimale qui n'est pas située en un point extrême de la région réalisable.

Il existe toutefois des cas, où la solution n'est même pas située sur la frontière de la région, .i.e. une solution optimale intérieure. Ceci est illustré par l'exemple [**4**] ci-après :

$$\begin{cases} \min z = (x_1 - 3)^2 + (x_2 - 3)^2 \\ s.c. \\ x_1 \leq 4 \\ x_2 \leq 6 \\ 3x_1 + 2x_2 \leq 18 \\ x_1 \, , \; x_2 \geq 0 \end{cases}$$

dont la représentation graphique est donnée par la figure I.1, où l'on peut voir que la solution optimale est strictement intérieure. En effet, les courbes de niveau de la fonction objectif forment des cercles concentriques, et il faut se situer au centre de ces cercles, c'est-à-dire au point (3,3) qui est strictement intérieur à la région.

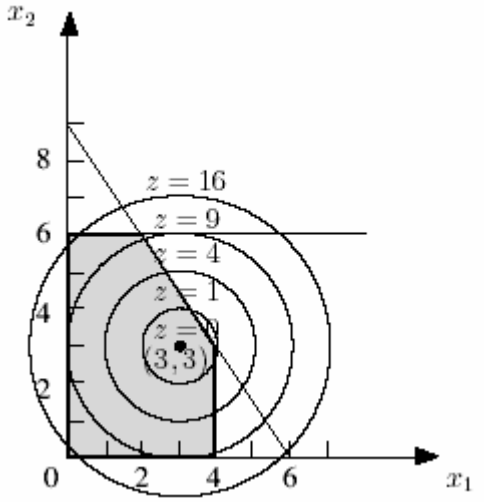


Figure I.1: ***Exemple illustratif N°1***

- La principale difficulté de la programmation non linéaire est que l'on peut avoir des optima locaux qui ne sont pas globaux. Illustrons ceci sur l'exemple [**4**] suivant :

$$\begin{cases} \max z = 3x_1 + 5x_2 \\ s.c. \\ x_1 \leq 4 \\ 8x_1 - x_1^2 + 14x_2 - x_2^2 \leq 49 \\ x_1 \; , \; x_2 \geq 0 \end{cases}$$

Pour pouvoir tracer sa représentation graphique, remarquons que :

$$(x_1 - 4)^2 + (x_2 - 7)^2 \geq 16$$

correspond au plan moins une clique de rayon 4, centré en (4,7). La représentation graphique est donnée par la figure I.2 où l'on peut voir que le point (4,3) est un minimum local non global :

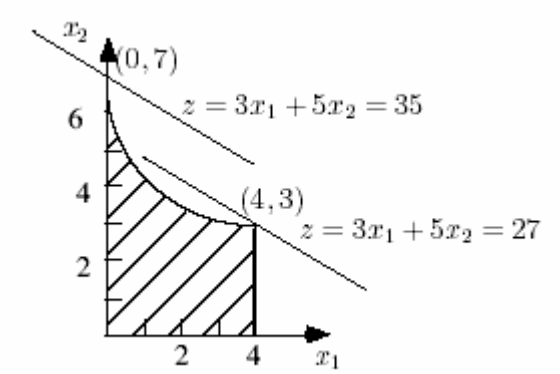


Figure I.2: ***Exemple illustratif N°2***

La détermination de tous les optima locaux est en général, un problème très délicat. En effet, lorsqu'un optimum local est atteint, dans quelle direction faut-il continuer la recherche ?
Cependant, il existe une classe de problèmes pour lesquels tous les optimums locaux sont des optimums globaux. Il s'agit des problèmes convexes.

I.3. Les problèmes convexes

Définition 1

Un ensemble $S \subset IR^n$ est dit *convexe*, si et seulement si :

$$\forall p \in S, \forall q \in S, \forall \lambda \in [0,1] \text{ alors } \lambda p + (1-\lambda)q \in S$$

Ce cas est illustré par la figure I.3, à la différence des ensembles non convexes illustrés par la figure I.4.

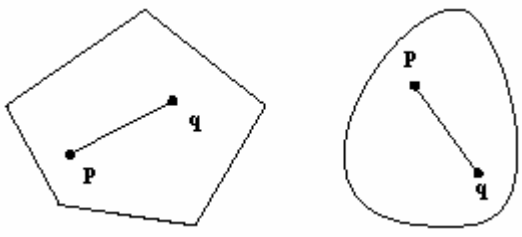


Figure I.3 : ***Ensembles Convexes***

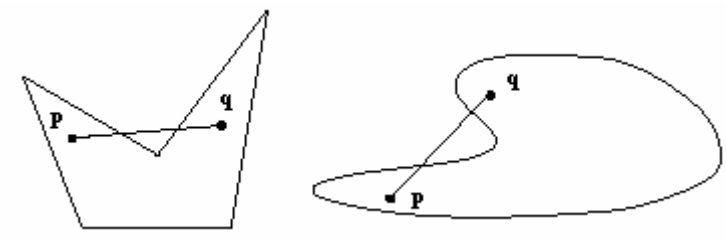


Figure I.4: ***Ensembles Non Convexes***

Définition 2

Une fonction $f : IR^n \to IR$, définie sur un ensemble S convexe, est *convexe*, si elle vérifie :

$$\forall p \in S, \forall q \in S, \forall \lambda \in [0,1] : f(\lambda p + (1-\lambda)q) \leq \lambda f(p) + (1-\lambda) f(q).$$

Définition 3

L'épigraphe d'une fonction f noté $épi(f)$ est l'ensemble des vecteurs à n+1 composantes (x, γ) :

$$\{(x,\gamma) \in IR^n \times IR / \gamma \geq f(x)\} \subset IR^{n+1}$$

Propriété 1

Une fonction f est convexe si et seulement si, $épi(f)$ est un ensemble convexe.

Ceci est illustré par la figure I.5, pour l'exemple de la fonction x^2 où l'on voit que l'épigraphe est bien un ensemble convexe. La définition de fonction convexe se ramène donc à celle d'ensemble convexe.

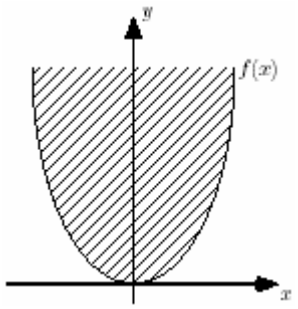


Figure I.5 : ***Exemple de fonction convexe***

Ainsi, nous pouvons définir la notion de problème convexe :

Définition 4

Un problème de programmation mathématique est dit convexe, s'il consiste à minimiser une fonction convexe (*resp: maximiser une fonction concave*) sur un domaine convexe.

Théorème 1 [**16**]

Pour un programme convexe, tout optimum local est un optimum global.

Avant de présenter l'algorithme de résolution, nous allons définir les très importantes conditions d'optimalité, vérifiées à l'optimum d'un problème non linéaire.

I.4. Conditions de Kuhn et Tucker [16]

Considérons le problème (**P**) défini précédemment :

Une notion importante en programmation non linéaire est celle de contraintes actives :

Définition 5

Une inégalité $g_i(x) \leq 0$ est active en x si $g_i(x) = 0$.

Il en découle immédiatement de cette définition, que les contraintes $h_k(x) = 0$ sont actives pour tout k. Par exemple, dans la figure I.6 [**4**], seule g_2 est active en x^*.

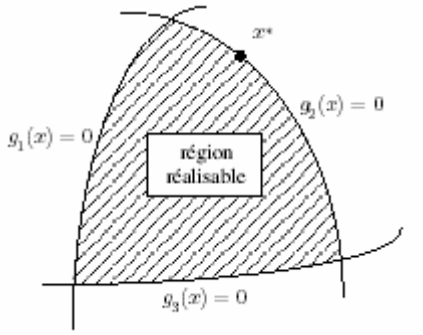


Figure I.6 : ***illustration du concept de la contrainte active***

Le concept de contrainte active est important, car tout comme en programmation linéaire, seules les contraintes actives à l'optimum comptent pour le problème. On pourrait résoudre le problème sans les autres contraintes et obtenir la même solution.

Pour écrire les conditions d'optimalité, on a besoin qu'une condition de régularité soit satisfaite.

Définition 6

Soit x^* satisfaisant les contraintes :

$$\begin{cases} g_i(x^*) \leq 0, & i = 1,...m \\ h_k(x^*) = 0, & k = 1,...p \end{cases}$$

Le point x^* est dit régulier pour ces contraintes si les gradients des contraintes actives sont linéairement indépendants.

On peut alors écrire les conditions nécessaires suivantes :

Théorème 2 *:* Conditions nécessaires de Kuhn –Tucker [**16**]

Soit x^* une solution réalisable pour le problème (**P**), supposons que x^* est un point régulier pour les contraintes. Alors, une condition nécessaire pour que x^* soit un optimum local du problème (**P**) est l'existence des multiplicateurs $\lambda \in IR^p$, $\mu \in IR^m$, $\mu \geq 0$ tels que:

$$\nabla f(x^*) + \sum_{k=1}^{p} \lambda_k \nabla h_k(x^*) + \sum_{i=1}^{m} \mu_i \nabla g_i(x^*) = 0 \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\text{(I)}$$

$$\mu_i g_i(x^*) = 0, \ \forall i = 1,...m. \quad \ldots\ldots\text{(II)}$$

$$\mu_i \geq 0, \ \forall i = 1,...m$$

Les conditions (II) sont les conditions de complémentarité indiquant que si une contrainte n'est pas active, son multiplicateur μ_i doit être nul.

Pour bien comprendre la signification pratique des conditions (I), nous allons les réécrire au moyen de la fonction de Lagrange. La fonction de Lagrange est obtenue en multipliant le membre de gauche de chaque contrainte d'égalité par un multiplicateur λ_k, le membre de gauche de chaque contrainte d'inégalité par son multiplicateur μ_i et en additionnant le tout à la fonction objectif :

$$L(x, \lambda, \mu) = f(x) + \sum_{k=1}^{p} \lambda_k h_k(x) + \sum_{i=1}^{m} \mu_i g_i(x)$$

Remarquant qu'à cet effet, on passe d'un problème d'optimisation sous contraintes à un problème d'optimisation sans contraintes. Les conditions d'optimalité pour une fonction définie sur IR^n sont simplement l'annulation de son gradient. On remarque que *les conditions de Kuhn et Tucker* (I) ne sont rien d'autre que l'annulation du gradient par rapport à x du gradient du Lagrangien : $\nabla_x L(x^*, \lambda, \mu) = 0$

Tandis que les contraintes (II) peuvent s'interpréter par $L(x^*) = f(x^*)$

Proposition 1

Si le problème est convexe, les conditions nécessaires « (I) et (II) » sont également suffisantes pour montrer que l'on est à l'optimum global.

I.5. Classe des problèmes non Linéaires sous contraintes linéaires

Soit le problème suivant :

$$(\mathbf{P'})\begin{cases} \min f(x) \\ s.c. \\ x \in X \end{cases}$$

Avec, $X = \{x \in IR^n / Ax \le b, x \ge 0\}$

Ce type de problème peut être résolu par une des méthodes des directions réalisables.

I.5.1. Les méthodes des directions réalisables

L'idée de ces méthodes est de se déplacer dans la région réalisable d'une solution à une autre, et à chaque itération dans ces méthodes les deux étapes suivantes sont requises :

- Le choix de la direction de descente.
- La détermination du pas de descente le long de cette direction.

Le principe de ces méthodes se résume comme suit :

Partant d'un point réalisable de départ x^k, on détermine la direction réalisable d^k en ce point, un pas de descente α_k suivant cette direction sera calculé dans le but de rester toujours dans la région réalisable. On obtient ainsi un autre point réalisable.

$$x^{k+1} = x^k + \alpha_k d^k.$$

Définition 7

Soit x^k une solution réalisable. Le vecteur $d^k \neq 0$ est une direction réalisable en x^k s'il existe $\delta > 0$ tel que : $(x^k + \alpha d^k) \in X \quad \forall\, 0 \le \alpha \le \delta.$

Si de plus $f(x^k + \alpha d^k) < f(x^k) \qquad \forall 0 \le \alpha \le \delta$

Le vecteur $d^k \neq 0$ est une direction de descente réalisable améliorante en x^k

La figure I.7 illustre le concept de directions réalisables :

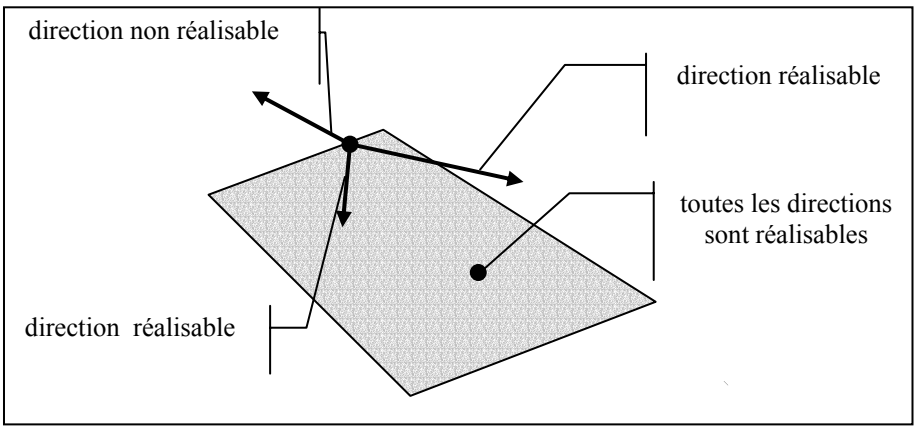


Figure I.7 : ***les directions réalisables***

I.5.1.1. *Caractéristiques des directions réalisables*

Proposition 2 [**25**]

Considérons le problème suivant :

$$\begin{cases} \min f(x) \\ s.c. \\ \quad Ax \le b \\ \quad Ex = e \end{cases}$$

Où A, E sont respectivement des m×n, r×n matrices, $b \in IR^m, e \in IR^L, x \in IR^n$.

Soit x une solution réalisable donnée. Supposons que $A_1 x = b_1$ et $A_2 x < b_2$ ou A^t se décompose en (A_1^t, A_2^t) et b^t en (b_1^t, b_2^t) ; alors :

- $d \neq 0$ est une direction réalisable au point x si et seulement si :
  - $A_1 d \le 0$
  - $Ed = 0$
- Si de plus $\nabla f(x)^t d < 0$, alors d est une direction de descente réalisable.

I.5.2. La méthode de Frank-Wolfe

La méthode de Frank-Wolfe [**8**] appartient à la catégorie des méthodes des directions réalisables, elle permet de résoudre les problèmes d'optimisation non linéaires sous contraintes linéaires, son principe est particulièrement simple. En effet, cette procédure combine des approximations linéaires de l'objectif pour trouver la direction de recherche, avec une recherche unidimensionnelle pour déterminer le pas suivant cette direction.

I.5.2.1 *Présentation de la méthode*

Considérons le problème (**P′**) défini précédemment :

$$(\mathbf{P'}) \begin{cases} \min f(x) \\ s.c. \\ \quad x \in X \end{cases}$$

Avec, $X = \{x \in IR^n / Ax \le b, x \ge 0\}$,

On suppose que f est convexe et différentiable.

A chaque itération k dans l'algorithme de Frank-Wolfe les étapes suivantes sont requises :

a) Calcul de la direction :

Soit x^k la solution réalisable de départ. Comme seul l'objectif est non linéaire, on peut se ramener à un problème de programmation linéaire en remplaçant la fonction objectif $f(x)$ par son développement de Taylor limité à l'ordre 1 :

$$f(y) \sim f(x^k) + \nabla f(x^k)^t (y - x^k) = f(x^k) + \sum_{j=1}^{n} \frac{\partial f(x^k)}{\partial x_j} (y_j - x_j^k)$$

En éliminant les constantes, cela revient à minimiser la fonction linéaire :

$$w(y) = \sum_{j=1}^{n} \frac{\partial f(x^k)}{\partial x_j} y_j = \sum_{j=1}^{n} c_j y_j$$

L'algorithme résout donc, le sous problème équivalent suivant :

$$\text{PL(k)} \begin{cases} \min w(y) = \sum_{j=1}^{n} c_j y_j \\ s.c. \\ \quad Ay \le b \\ \quad y \ge 0 \end{cases}$$

Soit y^k sa solution optimale.

La direction de Frank-Wolfe est alors définie par :

$$d^k = y^k - x^k.$$

b) Calcul du pas de descente:

Dans cette phase, la fonction objectif du problème (**P'**) est minimisée le long de la droit passant par le point x^k et de direction d^k. Le point obtenu de cette minimisation unidimensionnelle est x^{k+1}.

Analytiquement cela signifie que :

$$x^{k+1} = x^k + \alpha_k d^k \text{ avec } \alpha_k \in [0,1]$$

Tel que : $$f(x^{k+1}) = \min_{0 \le \alpha \le 1} f(x^k + \alpha d^k)$$

c) Critère d'arrêt [17]:

À chaque itération de l'algorithme de Frank-Wolfe, nous pouvons obtenir une borne inférieure de la valeur optimale de la fonction objectif $f(x^*)$:

- D'une part, pour une fonction convexe et différentiable, $\forall z \in IR^n$ et $\forall t \in IR^n$ on a :

$$f(z) + \nabla f(z)^t (t - z) \le f(t).$$

Et si on prend, $z = x^k$ et $t = x^*$, on aura :

$$f(x^k) + \nabla f(x^k)^t (x^* - x^k) \le f(x^*).....(1)$$

- D'autre part, y^k est solution optimale du sous problème PL(k) d'où :

$$f(x^k) + \nabla f(x^k)^t (x^* - x^k) \ge f(x^k) + \nabla f(x^k)^t (y^k - x^k).....(2)$$

De (1) & (2), on déduit que: $f(x^*) \ge f(x^k) + \nabla f(x^k)^t (y^k - x^k)$

Ainsi, $f(x^*) - f(x^k) \ge \nabla f(x^k)^t (y^k - x^k)$

Donc, la solution du sous problème PL(k) est une borne inférieure de la valeur optimale du problème (**P′**). De ce fait, l'algorithme s'arrêtera lorsque la valeur de la fonction objectif diffère de sa borne inférieure d'un seuil fixé ε>0.

$$\nabla f(x^k)^t (y^k - x^k) > -\varepsilon$$

L'algorithme de Frank-Wolfe se résume comme suit :

I.5.2.2. *Algorithme de Frank-Wolfe*

0. *Initialisation* : Soit x^1 une solution réalisable de départ, poser $k = 1$.

1. *Détermination de la direction de descente* :

Résoudre le problème linéaire PL(k) obtenu par approximation linéaire de l'objectif :

$$\text{PL(k)} \begin{cases} \min w(y) = \sum_{j=1}^{n} c_j y_j \\ s.c. \\ \quad Ay \le b \\ \quad y \ge 0 \end{cases}$$

Soit y^k sa solution optimale.

2. *Critère d'arrêt* : Si $\nabla f(x^k)(y^k - x^k) > -\varepsilon$, **arrêter.**

Sinon aller à l'étape **3.**

3. *Recherche du pas de descente* : résoudre le problème unidimensionnel :

$$f(x^k + \alpha_k d^k) = \min_{0 \le \alpha \le 1} f(x^k + \alpha d^k)$$

Déterminer la nouvelle solution x^{k+1} avec : $x^{k+1} = x^k + \alpha_k d^k$

Poser $k = k + 1$ et aller à l'étape **1.**

I.5.2.3. *Exemple d'application* (*voir Annexe* I)

II. Convergence de la Méthode de Frank-Wolfe

II.1. Introduction

Dans cette partie, on se propose d'étudier la convergence de la méthode de Frank-Wolfe. Afin d'accomplir cette tâche, on introduit quelques notions importantes de la théorie de convergence des algorithmes.

II.2. Théorie de convergence des algorithmes

Soit le problème suivant :

$$(\mathbf{P'})\begin{cases} \min f(x) \\ s.c. \\ x \in X \end{cases}$$

Avec, $X = \{x \in IR^n / Ax \leq b, x \geq 0\}$

II.2.1. *Notion d'algorithme*

Définition 8

Un algorithme de résolution est un processus itératif qui permet, à partir de la donnée du point initial x^1, d'engendrer une suite potentiellement infinie de points $x^1, x^2, ..., x^k, ...$, selon un ensemble d'instruction prescris et un test d'arrêt.

II.2.2. *Opérateur algorithmique (application multivoque)*

Définition 9

Un opérateur algorithmique A est une application de IR^n dans $P(IR^n)$ qui fait correspondre à chaque point $x \in IR^n$ un sous ensemble $A(x)$ de IR^n. Ainsi, se donnant un point initiale x^1, l'opérateur algorithmique A génère la suite $x^1, x^2, ..., x^k, ...$ Avec $x^{k+1} \in A(x^k)$.

Cette notion permet de décrire un algorithme particulier, or l'objet d'une théorie générale de la convergence est d'étudier le comportement d'une classe d'algorithmes plutôt que d'étudier tous les algorithmes de cette classe, lesquels ne diffèrent souvent en pratique que par des détails de mise en œuvre. Ainsi, la définition d'un opérateur algorithmique dans le cas général, peut s'énoncer comme suit :

Soient $A_1, A_2, ..., A_p$ un certain nombre d'algorithmes se rattachant à une même famille.

On peut considérer globalement cette famille d'algorithmes comme une application à composantes multiples qui, à un point x^k, associerait les p points $A_1(x^k), A_2(x^k),..., A_p(x^k)$.

Autrement dit: $A : IR^n \to IR^n$

$$x^k \to x^{k+1} \in A(x^k) = \{A_1(x^k), A_2(x^k),..., A_p(x^k)\}.$$

Remarque

Dans le cas d'un algorithme particulier à composante simple, on écrit tout simplement :

$$x^{k+1} = A(x^k).$$

II.2.3. *Convergence des algorithmes*

Nous dirons qu'un algorithme, décrit par un opérateur algorithmique A, est globalement convergeant (*ou encore possède la propriété de convergence globale*) si, quelque soit le point de départ x^1 choisi, la suite $\{x^k\}_1^\infty$ engendrée par $x^{k+1} \in A(x^k)$ converge vers un point satisfaisant une condition nécessaire d'optimalité.

Nous allons voir maintenant que les conditions qui assurent la propriété de convergence globale peuvent se ramener essentiellement à imposer à l'opérateur A une propriété de fermeture généralisant directement la notion de continuité.

II.2.4. *Opérateurs fermés*

Définition 10

Un opérateur algorithmique $A : IR^n \to IR^n$ est fermé au point x^∞, si pour $k \in \{1,2,...\}$

$$\left.\begin{array}{ll} & x^k \to x^\infty \\ & y^k \in A(x^k) \\ \text{et} & y^k \to y^\infty \end{array}\right\} \text{alors} \quad y^\infty \in A(x^\infty)$$

II.2.4.1. Composition des opérateurs

Définition 11

Soit $A : X \to Y$ et $B : Y \to Z$ deux opérateurs algorithmiques.

L'opérateur C=BA est défini de la manière suivante :

$$C : X \to Z$$
$$x \to C(x) = \bigcup_{y \in A(x)} \{B(y)\}.$$

Proposition 3 [**16**]

Soit $A : X \to Y$ et $B : Y \to Z$ deux opérateurs algorithmiques. On suppose que :

i. A est fermé en $x^\infty \in X$ et que B est fermé sur $A(x)$;

ii. Y est un ensemble fermé et toute suite $\{y^k\}$ telle que : $y^k \in A(x^k)$ avec : $x^k \to x^\infty$ admet une sous suite convergente.

Alors, l'opérateur algorithmique composé C=BA est fermé au point x^∞.

Corollaire 1

Soit X, Y, Z des ensembles fermés non vides de IR^n, IR^p et IR^q respectivement. Soient $A : \mathrm{X} \to \mathrm{Y}$ et $B : \mathrm{Y} \to \mathrm{Z}$ deux opérateurs.

Supposons que A est fermé au point x^∞ et B est fermé au point $A(x^\infty)$.

Si Y est compact alors : C=BA : $\mathrm{X} \to \mathrm{Z}$ est fermé au point x^∞.

II.2.5. *Théorème de convergence*

Lemme 1 [**24**]

Soit $Z : X \subset IR^n \to IR$ une fonction continue. Supposons qu'il existe une suite $\{x^k\}_1^\infty$ dans X telle que :

i. Pour tous $k \in IN^*$ $\quad Z(x^{k+1}) \le Z(x^k)$

ii. $x^k \to x^\infty$ pour tout $k \in K$ et $K \subset \{1,2,...\}$.

$$\text{Alors} : \lim_{k\to\infty} Z(x^k) = \lim_{k\in K} Z(x^k) = Z(x^\infty)$$

Théorème 2 [**24**]

Considérons le problème (**P'**) définie précédemment et soit Ω l'ensemble des solutions de ce problème.

Soit $A : IR^n \to IR^n$ un opérateur algorithmique pour le problème (**P'**) qui à partir d'un point initial $x^1 \in IR^n$ génère la suite $\{x^k\}_1^\infty$ tel que :

Si $x^k \in \Omega$ terminer ; sinon $x^{k+1} \in A(x^k)$, poser $k = k+1$ continuer.

Si :

i. Tous les points x^k sont dans un ensemble compact $\mathrm{X} \subset IR^n$.

ii. Il existe une fonction continue : $Z : IR^n \to IR$ telle que :

a. Si $x \notin \Omega$ alors : $y \in A(x)$ on a : $Z(y) < Z(x)$.

b. Si $x \in \Omega$ alors : $\forall y \in A(x)$ on a : $Z(y) \le Z(x)$.

iii. L'opérateur A est fermé en tout point x, tel que : $x \notin \Omega$.

Alors, tout point d'accumulation de $\{x^k\}_1^\infty$ est une solution optimale du problème (**P'**).

Preuve

- Si l'algorithme s'arrête d'après (**ii.b.**) le point x considéré est un point solution.
- Si l'algorithme génère une suite $\{x^k\}_1^\infty$; d'après (**i.**) il existe une sous suite convergente

$$x^k \to x^\infty \quad \forall k \in K \text{.....(1).}$$

En utilisant (**ii.a.**) on a :

$$Z(x^{k+1}) < Z(x^k) \quad \forall k \in K \text{.....(2).}$$

De (1) et (2) et le Lemme 1 :

$$\lim_{k\to\infty} Z(x^k) = Z(x^\infty) \text{.....(3).}$$

Reste à montrer que $x^\infty \in \Omega$

Résonnons par l'absurde; supposons que $x^\infty \notin \Omega$ considérons la sous suite $\{x^{k+1}\}_{k\in K}$, cette sous suite est contenue dans le compact X, donc il existe $K' \subset K$ tel que : $\{x^{k+1}\}_{k\in K'}$ converge.

D'après le Lemme 1 :

$$\lim_{k\to\infty} Z(x^{k+1}) = Z(x^{\infty+1}) \text{.....(4).}$$

De (3) et (4) on obtient :

$$Z(x^{\infty+1}) = Z(x^\infty) \text{.....(5).}$$

$x^\infty \notin \Omega$, de (**iii.**) l'opérateur A est fermé au point $x^\infty \notin \Omega$, ceci se traduit par :

$$\left.\begin{array}{ll} & x^k \to x^\infty \\ & x^{k+1} \in A(x^k) \\ \text{et} & x^{k+1} \to x^{\infty+1} \end{array}\right\} \text{alors} \quad x^{\infty+1} \in A(x^{\infty+1}) \text{.....(6)}$$

De (6) et (**ii.a.**) on a $Z(x^{\infty+1}) < Z(x^\infty)$ contradiction avec (5). De ce fait, $x^\infty \in \Omega$.

II.2.6. *Algorithmes mixés*

On suppose qu'on dispose d'un algorithme A (opérateur algorithmique), on définie une procédure dans la quelle A est utilisé pendant un nombre d'itérations assez important, et dans le reste des itérations (nombre finie assez petit), on introduit un autre opérateur algorithmique. La procédure globale sera définie par un algorithme dit mixé.

Définition 12

Soit un algorithme "B"de programmation non linéaire définie par son opérateur algorithmique $B : IR^n \to IR^n$, et de fonction dévaluation Z.

Un algorithme mixé "A" dérivé de "B"est définie de la manière suivante :

$A \equiv A_k : IR^n \to IR^n$ opérateur de base avec $A_k = B$ pour $k \in K$, et $K \subset IN$

et pour $k \in IN / K$, avec $|IN / K|$ fini. Un autre opérateur algorithmique est utilisé tel que :

$$Z(x^{k+1}) \le Z(x^k) \text{.....(7)}$$

Remarque

- L'opérateur algorithmique introduit pendant les $|IN / K|$ itérations n'est autre qu'une modification appropriée de l'opérateur B.
- L'objectif recherché dans cette modification réside dans l'amélioration de la performance de l'algorithme B en terme de vitesse de convergence.

II.2.6.1. Théorème de convergence des algorithmes mixés [17]

Soit B un opérateur algorithmique de base $B : IR^n \to IR^n$, de fonction d'évaluation Z, d'ensemble de solution Ω, vérifiant les conditions **(i.)**, **(ii.)** et **(iii.)** du théorème de convergence des algorithmes (*Théorème*1) et soit "A" algorithme mixé dérivé de l'algorithme "B " .

Pour $x^1 \in IR^n$ donné, supposons que $\{x^k\}_1^\infty$ est la suite de point générée par l'algorithme mixé A ; de plus on suppose que les conditions suivantes sont vérifiées :

i. Tous les $x^k \in$ X, ou X est compact.

ii. Soit x^* un point solution ($x^* \in \Omega$) et soit y un point de l'algorithme "A" alors :
Si $Z(y) < Z(x^*)$. alors $y \in \Omega$.

Alors l'algorithme mixé "A" :

Soit se termine en un point solution, ou bien tout point d'accumulation de $\{x^k\}_1^\infty$ est un point de Ω.

Preuve

Considérons une sous suite convergente arbitraire $x^k \to x',\ k \in K'$.

On doit montrer que *x'* est un point solution (notons que l'ensemble d'indice *K'* n'est pas nécessairement lié à *K*).

D'après (7), et les propriétés de "*B*" :

$$Z(x^{k+1}) \le Z(x^k) \text{ pour } k = 1,2,....$$

Le Lemme1 entraîne $\lim_{k\to\infty} Z(x^k) = Z(x')$.....(8)

La condition (**i.**) implique l'existence de $K^1 \subset K$ telque : $x^k \to x^\infty, k \in K^1, (x^\infty \in \mathrm{X})$

En utilisant encore une fois le Lemme1 on obtient :

$$\lim_{k\to\infty} Z(x^k) = Z(x^\infty).....(9)$$

De (8) et (9) :

$$Z(x') = Z(x^\infty).....(10)$$

Si on effectue le même résonnement pour $\{x^k\}_{K^2}$ avec $K^2 \subset K^1$ on aura :

$$\lim_{k\to\infty} Z(x^{k+1}) = Z(x^{\infty+1}).....(11)$$

De (9) et (11):

$$Z(x^{\infty+1}) = Z(x^\infty).....(12)$$

Reste à montrer que $x^\infty \in \Omega$.

Résonnons par l'absurde, supposons que $x^\infty \notin \Omega$, comme $A_k = B$ pour $k \in K$, et $K^1 \subset K$ et sachant que "*B*" vérifie les conditions du théorème de convergence des algorithmes énoncé précédemment, notamment la fermeture de l'opérateur *B* ce qui entraîne :

$$\left.\begin{array}{l} x^k \to x^\infty, k \in K^1 \\ x^{k+1} \in B(x^k), k \in K^1 \\ \text{et} \quad x^{k+1} \to x^{\infty+1}, k \in K^1 \end{array}\right\} \text{alors} \quad x^{\infty+1} \in B(x^{\infty+1}).$$

Et d'après (**ii.a.**) du même théorème $Z(x^{\infty+1}) < Z(x^\infty)$.....(13)

Contradiction avec (12).

De ce fait, $x^\infty \in \Omega$.....(14)

De (10) et (14) et la condition (**ii.**) on obtient $x' \in \Omega$ d'où le résultat.

II.3. Convergence de La méthode de Frank-Wolfe [17]

Le procédé itératif de la méthode de Frank-Wolfe pour la résolution du problème (**P'**), peut être complètement décrit par l'opérateur algorithmique suivant :

$$\boldsymbol{A} : IR^n \to IR^n$$
$$x^k \to \boldsymbol{A}(x^k) = x^{k+1} \text{ pour } k \in IN^* \text{ avec } \boldsymbol{A} = \boldsymbol{MD}$$

L'opérateur algorithmique $\boldsymbol{A}$ est composé de deux opérateurs :

- Le premier étant l'opérateur de la direction de recherche :

$$\boldsymbol{D} : IR^n \to IR^{2n}$$
$$x^k \to \boldsymbol{D}(x^k) = \left\{(x^k, d^k) / d^k = y^k - x^k\right\}$$

  Avec y^k Solution du sous problème PL(k)

- Le second est l'opérateur du pas de descente :

$$\boldsymbol{M} : IR^{2n} \to IR^n$$
$$(x^k, d^k) \to \boldsymbol{M}(x^k, d^k) = \left\{x^{k+1} / f(x^{k+1}) = \min_{0 \le \alpha \le 1} f(x^k + \alpha d^k)\right\}$$

Pour établir la convergence de la méthode de Frank-Wolfe, nous appliquerons le théorème fondamental de convergence des algorithmes énoncé précédemment « *Théorème*1 ».de ce fait, nous devons vérifier les conditions suivantes :

i. Tous les points x^k sont dans un ensemble compact $X \subset IR^n$

ii. Il existe une fonction continue : $Z : IR^n \to IR$ telle que :

  a. Si $x \notin \Omega$ alors : $y \in A(x)$ on a : $Z(y) < Z(x)$.

  b. Si $x \in \Omega$ alors : $\forall y \in A(x)$ on a : $Z(y) \le Z(x)$.

iii. L'opérateur A est fermé en tout point x, tel que : $x \notin \Omega$.

Preuve

i) Montrons que tous les points x^k appartiennent à un compact $X \subset \mathbb{R}^n$

Il faut prouver que tous les points générés par l'algorithme appartiennent à $\overline{C_0(X)}$. $\overline{C_0(X)}$ est la fermeture de l'enveloppe convexe de X où $X = \left\{x^1, y^1, ..., y^p\right\}$ et $y^1, ..., y^p$ sont les points extrêmes de le région réalisable.

Un enveloppe convexe est par définition : l'ensemble des points de IR^n qui peuvent s'écrire comme combinaison convexe des points de X.

D'après la construction des points de l'algorithme de Frank-Wolfe, nous avons pour tout $k \ge 1, x^{k+1} \in [x^k, y^k]$, ce la signifie que x^{k+1} est choisi sur le segment $[x^k, y^k]$.

Ainsi $x^2 \in [x^1, y^1]$; donc x^2 est combinaison convexe des deux points x^1, y^1 (avec x^1 solution réalisable de départ de l'algorithme de Frank-Wolfe), en effet :

$$\begin{aligned} x^2 &= x^1 + \alpha_1 (y^1 - x^1) \\ &= x^1 (1 - \alpha_1) + \alpha_1 y^1. \end{aligned}$$

d'où $x^2 \in C$

De la même manière, par récurrence sur $k \geq 1$, il en résulte que $x^{k+1} \in C, \forall k = 1,2,...$

Reste à montrer que *C* est compact.

On sait que tous les points de la région réalisable sont de norme finie, et x^1 la solution réalisable est également finie, donc *C* est borné.

Et comme de plus *C* est fermé, nous concluons que *C* est compact.

ii)

a. Si $x \in \Omega$ alors pour tout *y* solution du sous problème PL(*x*), on a :

$$\nabla f(x)^t (y - x) < 0 \text{.....(I)}$$

Considérons $z \in A(x) = MD(x) = M(x,d)$, où $d = y - x$

et $f(z) = \underset{0 \leq \alpha \leq 1}{\text{Min}} f(x + \alpha d)$

soit $g(\alpha) = f(x + \alpha d)$

alors $\dfrac{dg(\alpha)}{d\alpha} = \dfrac{d}{d\alpha}\left[f(x + \alpha d)\right] = \nabla f(x + \alpha d)^t d$

D'aprés (I) $\left.\dfrac{dg(\alpha)}{d\alpha}\right|_{\alpha=0} = \nabla f(x)^t d < 0$

ceci implique pour tout $0 \leq \alpha \leq 1$

$g(\alpha) < g(0)$

autrement dit; $f(x + \alpha d) < f(x)$

ainsi, $f(z) < f(x)$

b. étant donné que $\Omega = \left\{x^* \in \text{IR}^n \,/\, f(x^*) = \min f(x) \text{ avec } x \in X\right\}$, il est clair que si $x \in \Omega$, alors : soit l'algorithme se termine à l'étape3, ou bien pour tout $z \in A(x)$ $f(z) \leq f(x)$.

iii) On doit monter que l'opérateur algorithmique de l'algorithme de Frank-Wolfe est fermé en tout point $y \notin \Omega$.

D'après le Corollaire1, et sachant que l'opérateur algorithmique ***A*** est composé de deux opérateurs ***A*=*MD***, donc il suffit de vérifier que les deux opérateurs ***M*** et ***D*** sont fermés :

- **Fermeture de l'opérateur *M*** :

On doit montrer si :

$$\text{Pour } k \in \{1,2,...\} \quad \begin{array}{l} (x^k, d^k) \to (x^\infty, d^\infty) \\ x^{k+1} \in M(x^k, d^k) \\ \text{et} \quad x^{k+1} \to x^{\infty+1} \\ \text{alors} \quad x^{\infty+1} \in M(x^\infty, d^\infty) \end{array}$$

Considérons la suite $\{\alpha_k\}_1^\infty$ associée à la suite $\{x^k\}_1^\infty$. Comme $J = [0,1]$ est fermé et borné, donc compact ; il existe $K_1 \subset IN^*$ et $\{\alpha_k\}_{k\in K_1}$ tels que :

Pour $k \in K_1; \alpha_k \to \alpha_\infty$, et $\alpha_\infty \in J$, soit maintenant :

$$\begin{array}{l} x^{k+1} \in M(x^k, d^k) \\ \text{donc} \quad x^{k+1} = x^k + \alpha_k d^k \\ \text{avec pour tout } \alpha \in J = [0,1]; f(x^k + \alpha_k d^k) \le f(x^k + \alpha d^k) \text{.....(II)} \end{array}$$

En vertu de la continuité de *f*, on peut passer à la limite dans (II) :

$$f(x^{\infty+1}) = \lim_{\substack{k\in K_1 \\ k\to\infty}} f(x^{k+1}) \le \lim_{\substack{k\in K_1 \\ k\to\infty}} f(x^k + \alpha d^k) = f(x^\infty + \alpha d^\infty) \text{.....(III)}$$

Or la formule (III) est vérifiée pour tout $\alpha \in J$; et en particulier $x^* \in M(x^\infty, d^\infty)$. c'est-à-dire :

$$f(x^{\infty+1}) \le f(x^*).$$

D'autre part $x^* \in M(x^\infty, d^\infty)$; ceci implique que :

$$f(x^*) = \min_{\alpha\in J} f(x^\infty + \alpha d^\infty) \le f(x^\infty + \alpha_\infty d^\infty) = f(x^{k+1}) \text{.....(IV)}$$

De (III) et (IV) on obtient :

$$\begin{array}{l} f(x^*) = f(x^{\infty+1}) \\ \text{et } x^{\infty+1} \in M(x^\infty, d^\infty) \end{array} \quad \text{d'où } \boldsymbol{M} \text{ est fermé.}$$

- **Fermeture de l'opérateur *D*** :

On doit montrer si :

$$\text{Pour } k \in \{1,2,...\} \quad \begin{array}{l} x^k \to x^\infty \\ y^k \in D(x^k) = (x^k, d^k) \\ \text{et} \quad y^k \to y^\infty \\ \text{alors} \quad y^\infty \in D(x^\infty) = (x^\infty, d^\infty) \end{array}$$

En d'autres termes, il faut montrer que y^{∞} est solution du sous problème $\mathrm{PL}(x^{\infty})$.

y^k est solution du sous problème $\mathrm{PL}(k)$; d'où pour tout y faisable :

$$\nabla f(x^k)^t y^k \leq \nabla f(x^k)^t y \text{.....(V)}$$

f est supposée continûment différentiable. Ainsi, en passant à la limite dans (V), on obtient pour tout y faisable ;

$$\nabla f(x^{\infty})^t y^{\infty} \leq \nabla f(x^{\infty})^t y \text{.....(VI)}$$

Par hypothèse, tous les points extrêmes y^k de la région faisable sont de norme finie.

D'après (**i.**) ils appartiennent tous à l'ensemble compact C, où :

$$C \subset X = \{x \in IR^n / Ax \leq b, x \geq 0\}.$$

Il en résulte que y^{∞} est aussi réalisable, et ceci entraîne avec (VI) que y^{∞} est solution du sous problème $\mathrm{PL}(x^{\infty})$; autrement dit $y^{\infty} \in D(x^{\infty})$, et ***D*** est fermé au point z.

Il en résulte ainsi, que tout point d'accumulation de la suite $\{x^k\}_1^{\infty}$ est une solution optimale du problème (**P'**).

II.4. Taux de convergence de l'algorithme de Frank-Wolfe

Considérons le problème (**P'**) définie précédemment avec : f une fonction convexe continûment différentiable.

Deux cas ont été envisagés, afin de dégager des informations pratiques sur le taux de convergence de cet algorithme :

***Premier cas** : la fonction « f » est faiblement convexe*

Soit x^* la solution optimale du problème, avec $f(x^*)=M$ et supposons sans perte de généralité que $M=0$. A l'itération k, soient : x^k la solution courante, y^k la solution du sous problème PL(k) et la direction de descente $d^k = y^k - x^k$.

- Comme f est une fonction convexe, donc :

$$f(x^k) + \nabla f(x^k)^t (x^* - x^k) \leq f(x^*) = 0 \text{.....(1)}$$

- Et comme y^k est solution du sous problème PL(k) d'où, $\forall y \in X$:

$$\nabla f(x^k)^t y \geq \nabla f(x^k)^t y^k \text{.....(2)}$$

De (1) et (2), on obtient : $f(x^k) + \nabla f(x^k)^t (y^k - x^k) \leq f(x^*) = 0$

Autrement dit, $\nabla f(x^k)^t d^k \leq -f(x^k)$.....(3)

D'après la définition d'une fonction faiblement convexe : $\forall u \in IR^n, \forall v \in IR^n$ et $\forall w \in IR^n$ de norme finie avec $u \neq v$, il existe une constante C tel que : $(u-v)^t H(w)(u-v) \leq C.$

H est la Hessienne de f.

En terme de produit scalaire :

$$\begin{aligned}(u-v)^t H(w)(u-v) &= (H(w)(u-v),(u-v)) \\ &\leq \|H(w)(u-v)\|.\|u-v\| \\ &\leq \left\|\frac{H(w)(u-v)}{(u-v)}\right\|.\|u-v\|^2 \\ &\leq \max_i|\lambda_i|.\|u-v\|^2 \\ &\leq A.c = C.....(4)\end{aligned}$$

(Comme f est convexe et u, v sont bornés, les λ_i sont les valeurs propres de H(w))

De (4) et pour $\alpha_k \leq \frac{f(x^k)}{C}$ on obtient :

$$\begin{aligned}\nabla f(x^k)^t d^k + \alpha_k d^{kt} H(w) d^k &\leq -f(x^k) + \alpha_k C \\ &\leq -f(x^k) + f(x^k) = 0\end{aligned}$$

Ainsi, pour $0 \leq \alpha_k \leq \frac{f(x^k)}{C}$ et pour w approprié, on a :

$$\begin{aligned}f(x^k + \alpha_k d^k) &= f(x^k) + \alpha_k \nabla f(x^k)^t d^k + \tfrac{1}{2}\alpha_\kappa^2 d^{kt} H(w) d^k \\ &= f(x^k) + \tfrac{1}{2}\alpha_k \nabla f(x^k)^t d^k + \tfrac{1}{2}\alpha_k [\nabla f(x^k)^t d^k + \alpha_\kappa d^{kt} H(w) d^k] \\ &\leq f(x^k) + \tfrac{1}{2}\alpha_k \nabla f(x^k)^t d^k \\ &\leq (1 - \tfrac{1}{2}\alpha_k) f(x^k).....(5)\end{aligned}$$

Posons $m = \min\left(1, \frac{f(x^k)}{C}\right)$, alors : $f(x^{k+1}) = \min_{0\leq\alpha\leq 1} f(x^k + \alpha d^k) \leq \min_{0\leq\alpha\leq m} f(x^k + \alpha d^k)$

En utilisant la relation (5), ceci entraîne que :

$$\begin{aligned}f(x^{k+1}) &\leq (1 - \tfrac{1}{2}m) f(x^k) \\ &\leq \max(\tfrac{1}{2}, (1 - \tfrac{1}{2}\frac{f(x^k)}{C})) \times f(x^k).....(6)\end{aligned}$$

En Posant $a_k = \frac{f(x^k)}{2C}$ dans la relation (6), on obtient : pour tout $k \geq 0$

$$a_{k+1} \leq a_k \times \max(\tfrac{1}{2}, 1 - a_k).....(7)$$

Distinguons deux cas :

1. $a_0 \leq ½$ ce qui entraîne $\forall k \geq 0 : a_k \leq ½$ et par conséquent :

$$a_{k+1} \leq a_k(1-a_k).....(8)$$

2. $a_0 > ½$ de la relation (8) on a : $a_1 \leq a_0/2$

- Si $a_1 \leq ½$ alors, $\forall k \geq 1 : a_k \leq ½$ et l'on a la relation (8).
- Si $a_1 > ½$ alors, $a_2 \leq \frac{a_0}{2^2}$ et ainsi de suite, on obtient par conséquent : $a_k \leq a_0(2)^{-k}$

D'où pour $k \to 0, a_k \leq 0$, ce qui signifie l'existence d'un entier $K' \geq 0$ tel que :

Pour $k \leq K', a_k \geq ½$

Et pour $k > K', a_k < ½$

Des deux cas cités plus haut, on peut dire qu'il existe un entier $K \geq 0$ tel que :

- Pour $k \leq K, a_k \geq ½$ et $a_{k+1} \leq a_k/2$ d'où $a_k \leq a_0(2)^{-k}$ ..... (9)
- Pour $k > K, a_k < ½$ et $a_{k+1} \leq a_k(1-a_k)$

$$\text{Donc } \frac{1}{a_{k+1}} \geq \frac{1}{a_k} \times \left(\frac{1}{1-a_k}\right) = \frac{1}{a_k} + \left(\frac{1}{1-a_k}\right)$$

$$= \frac{1}{a_k} + 1 + \left(\frac{a_k}{1-a_k}\right)$$

$$\text{Or } \frac{a_k}{1-a_k} < 1 \text{ d'où } \frac{1}{a_{k+1}} \geq \frac{1}{a_k} + 1$$

$$\text{Ainsi, pour tout } k > K, \frac{1}{a_k} \geq k + K + \frac{1}{a_k}.....(10)$$

De (9) et (10), on obtient comme résultat final:

$$f(x^k) - f(x^*) \leq \begin{cases} f(x^0)(2)^{-k} & \text{si } k \leq K \\ \dfrac{f(x^K)}{2C + (k-K)f(x^K)} & \text{si } k > K \end{cases}$$

On constate de la seconde relation que la progression de la fonction objectif vers son optimum est dans le pire des cas comparable à celle de $\frac{1}{k} \to 0$ quand $k \to \infty$.

- ***Deuxième cas** : la fonction « f » est fortement convexe et la solution optimale x* appartient à la frontière de X, mais n'est pas un point extrême de la région réalisable:*

De ce fait, il existe $K \geq 0$ tel que pour tout $k > K$ aucun x^k n'est un point extrême de X. On suppose également que x^K n'appartient pas à la face de X contenant x^*.

Soit α_k solution du sous problème unidimensionnel: $\min_{0\leq\alpha\leq 1} f(x^k + \alpha d^k)$

Les hypothèses énoncées précédemment entraînent que pour tout $k > K$ $\alpha_k < 1$

Soit $\{\lambda_y^k\}$ une représentation barycentrique de x^k sur Y (avec Y : l'ensemble des solutions du sous problème PL(k)) $x^k = \sum_{y\in Y} \lambda_y^k y.$

Nous avons :

$$\begin{aligned} x^{k+1} &= x^k + \alpha_k d^k \\ &= x^k + \alpha_k (y^k - x^k) \\ &= \alpha_k y^k + (1-\alpha_k) x^k \\ &= \alpha_k y^k + \sum_{y\in Y} (1-\alpha_k)\lambda_y^k y \qquad \text{.....(11)} \\ &= \sum_{y\in Y} \lambda_y^{k+1} y^{k+1} \end{aligned}$$

Pour des λ_y^{k+1} appropriés, représentant barycentriquement x^{k+1} dans Y.

La caractéristique de ces représentations définies récursivement par la relation (11) est :

$$\lambda_y^{k+1} \geq (1-\alpha_k)\lambda_y^k$$

Et en particulier pour tout $k > K$: $\lambda_y^k \geq \lambda_y^k \times \prod_{K\leq j\leq k} (1-\alpha_k)$.....(12)

La convergence de $x^k \to x^*$ implique que : $\lambda_y^k \to 0$ si y n'est pas sur la face contenant x^*.

Pour les $y \in Y$, par hypothèse $\lambda_y^K \neq 0$ (car x^K n'appartient pas à la face de X contenant x^*.)

Par conséquent le produit (12) converge vers zéro quand $K \to \infty$. Ainsi, $\sum \alpha_k$ diverge.

Lemme 2

Soit b_n, pour n=1,2,… une suite de nombres réels telle que la série $\sum |b_n|$ diverge. Alors :

$$\forall \varepsilon > 0 : \sum_{n=k}^{\infty} b_n^2 \geq \frac{1}{k^{(1+\varepsilon)}} \forall k \in IN^*.$$

En utilisant le lemme2 on obtient : $\forall \varepsilon > 0 : \sum_{n=k}^{\infty} \alpha_n^2 \geq \frac{1}{k^{(1+\varepsilon)}} \forall k \in IN^*$.....(13).

Lemme 3

Soit g une fonction fortement convexe à une seule variable α, définie sur [0, A] tel que :

$$g(0) = 0 \text{ et } g'(0) < 0$$

Soit α^* le minimum de g sur l'intervalle [0, A] alors : $\alpha^* \geq \frac{-g'(0)}{c}$, c : une constante positive.

Lemme 4

Soit g une fonction faiblement convexe, suivant les mêmes hypothèses du lemme 3, on a :

$$g(\alpha^*) \leq \frac{-g'(0)^2}{c'}.$$

Par application des résultats des deux lemme 3 et 4 à la fonction g suivante :

$$g(\alpha) = f(x^k + \alpha_\kappa d^k) - f(x^k) \quad \forall k \geq K$$
$$g'(\alpha) = \nabla f(x^k + \alpha_\kappa d^k)^t d^k$$
$$g''(\alpha) = d^{k\,t} \nabla f(x^k + \alpha_\kappa d^k)^t d^k$$

On obtient :

$$0 \leq \alpha_\kappa \leq -\nabla f(x^k + \alpha_\kappa d^k)^t d^k \Big/ c\left\|d^k\right\|^2$$
$$c\left\|d^k\right\| \times \alpha_\kappa \leq -\nabla f(x^k + \alpha_\kappa d^k)^t d^k \Big/ c\left\|d^k\right\| \text{.....(14)}$$

Et $$f(x^k) - f(x^{k+1}) \geq (\nabla f(x^k)^t d^k)^2 \Big/ c'\left\|d^k\right\|^2 \text{.....(15)}$$

En itérant la relation (15) pour $j \geq k$ on aura :

$$f(x^k) - f(x^{k+1}) \geq (\nabla f(x^k)^t d^k)^2 \Big/ c'\left\|d^k\right\|^2$$
$$f(x^{k+1}) - f(x^{k+2}) \geq (\nabla f(x^{k+1})^t d^{k+1})^2 \Big/ c'\left\|d^{k+1}\right\|^2$$
$$\vdots$$
$$f(x^j) - M \geq (\nabla f(x^j)^t d^j)^2 \Big/ c'\left\|d^j\right\|^2$$

En sommant ces inégalités membre à membre on obtient : $f(x^k) - M \geq \frac{1}{c'} \sum_{j \geq k} \left(\frac{\nabla f(x^j)^t d^j)}{\left\|d^j\right\|} \right)^2$

De la relation (14) :

$$f(x^k) - M \geq \frac{c^2}{c'} \sum_{j \geq k} \alpha_j^2 \left\|d^j\right\|^2$$
$$\geq \frac{c^2}{c'} \min_{y \in Y} \left\|x^* - y\right\|^2 \sum_{j \geq k} \alpha_j^2$$

En utilisant (13), on aura : $f(x^k) - M \geq \frac{c^2}{c'} \min_{y \in Y} \left\|x^* - y\right\|^2 \times \frac{1}{k^{(1+\varepsilon)}}$

Enfin, $\forall k > K$ et $\forall \varepsilon > 0 : f(x^k) - M \geq \frac{L}{k^{(1+\varepsilon)}}$ avec L : une constante positive.

Dans ce cas également, on obtient une évaluation sensiblement comparable à celle déterminée dans le premier cas.

De ces résultats, on peut déduire que l'algorithme de Frank-Wolfe souffre d'une certaine lenteur de convergence et nécessite des modifications visant à améliorer sa vitesse de convergence.

III. Versions Modifiées de la Méthode de Frank-Wolfe

La méthode de Frank-Wolfe s'est avérée très prometteuse pour la résolution des problèmes de routage, de télécommunication et de trafic urbain. C'est toujours la méthode la plus populaire dans ces secteurs ; toutefois sa vitesse de convergence n'est pas entièrement satisfaisante. Cette lenteur est due principalement au chemin en zigzag décrit par les points de l'algorithme. Ce dernier est provoqué par le schéma de construction des directions de descente faisables, lesquelles pointent toujours vers un point extrême de la région réalisable. Pour remédier à cet inconvénient beaucoup de tentatives ont été élaborées réparties en trois catégories de modifications à savoir:

- *Modification du pas de descente*

  Telle la technique proposée par Weintraub & al. [**23**] qui consiste à modifier la longueur du pas de descente obtenu de l'optimisation unidimensionnelle, définie comme forme de compensation des insuffisances des directions de recherche.

- *Modification du sous problème* PL(k)

  Le sous problème linéaire obtenu par approximation de la fonction objectif noté PL(k) est modifié afin d'éviter de générer les points extrêmes comme solution.

- *Modification de la direction de descente*

  Dans ces techniques, la direction de descente est combinée avec les précédentes dans le but de trouver de meilleures directions de recherche que celle de la méthode de Frank-Wolfe :

  - ***Fukushima*** [**10**] détermine la direction de descente par combinaison d'un certain nombre de points y^k solutions du sous problème PL(k) déterminés précédemment par l'algorithme de Frank-Wolfe ; c'est-à-dire :

$$d^k = \sum_{i=1}^{k} \lambda_i y^i - x^k \text{ avec} \sum_{i=1}^{k} \lambda_i = 1, \lambda_i \geq 0$$

  - ***Lupi*** [**15**] détermine la direction de descente "d_L^k"par combinaison convexe de la direction de descente générée précédemment par l'algorithme "d_L^{k-1}" et la direction de descente de l'algorithme de Frank-Wolfe"d^k", tels que d_L^k et d_L^{k-1} sont orthogonale.

- ***L'approche du PARTAN*** [**1**] cette technique consiste à introduire une étape supplémentaire dans l'algorithme de Frank-Wolfe : à chaque itération *k*, le nouveau point généré par l'algorithme est combiné avec le précédent de l'itération *k*-1, de sorte à minimiser l'objectif le long de la droite reliant ces deux points.

Dans ce qui suit nous exposons avec plus de détails quelques variantes qui ont été développées dans ce contexte :

III.1. La technique du PARTAN

La méthode du parallel tangents « PARTAN » est introduite afin de réduire le phénomène de zigzag dans les méthodes de gradient, pour l'optimisation sans contraintes, et cela en effectuant à chaque itération une recherche de pas de descente supplémentaire. Ainsi, pour une solution courante x^k ; dans un premier temps on calcule le pas de descente suivant la direction du gradient, on obtient une solution intermédiaire v^k; on effectue ensuite une seconde recherche de pas de descente depuis le point v^k, dans la direction $d^k = v^k - x^{k-1}$ produisant ainsi, la nouvelle solution x^{k+1}.

La variante de Frank-Wolfe incluant la technique du PARTAN notée « *PFW* » se résume comme suit :

Etape 0 (*Initialisation*) poser k=0 et choisir $v^k \in X$ la solution réalisable de départ.

Poser $x^{k+1} = v^k$.

Etape 1 *Choix de la direction de descente* : poser *k=k+1* trouver y^k solution du sous problème PL (k), Poser $d^k = y^k - x^k$

Etape 2 *Recherche du pas de descente* : trouver α_k, solution du problème unidimensionnel :

$$\min_{0 \le \alpha_k \le 1} f(x^k + \alpha_k d^k)$$

Etape 3 Poser $v^k = x^k + \alpha_k d^k$.

Etape 4 Si le *test de convergence* est vérifié, **Stop**.

Sinon, Si $k < 2$ poser $x^{k+1} = v^k$. aller à l' **Etape1**.

Sinon aller à l' **Etape 5**.

Etape 5 (*L'étape de PARTAN*) Calculer x^{k+1} par :

$$x^{k+1} = (1 - \tau_k) v^k + \tau_k x^{k-1}.$$

Avec : τ_k : est solution du problème suivant $\min_{\tau_k^{\min} \le \tau_k \le 1} f(x^k)$

Où $\tau_k^{\min} \leq 0$ représente la plus négative longueur de pas qui gardera x^k dans la région faisable.

Une formule pour le calcul récursif exact du paramètre $\tau_k^{\min}$ est déterminée par Arezki & Van Vliet [**1**]. Elle exprime $\tau_k^{\min}$ en termes de $\alpha \,\&\, \tau$ à partir des deux dernières itérations seulement.

Remarque : En ce qui concerne la preuve de convergence de cette technique voir [**20**].

III.2. Direction de Frank-Wolfe conjuguée (CFW)

Le principe des méthodes des directions conjuguées dans le cas des problèmes quadratiques convexes sans contraintes est le suivant :

A partir d'un point initiale x^0 minimiser la fonction objectif successivement suivant n directions linéairement indépendantes $d^1, d^2, ..., d^n$ possédant la propriété d'être mutuellement conjuguées par rapport au hessien « H » de la fonction objectif, cette propriété s'exprime par :

$$\left.\begin{array}{l} \forall i \;\; i = \overline{1,n} \\ \forall j \;\; j = \overline{1,n} \end{array}\right\} \Rightarrow d_i^T H d_j = 0$$

Ces méthodes convergent en au plus n étapes, dans IR^n.

Dans le cas de la méthode des Gradients conjugués, on obtient la direction conjuguée par combinaison linéaire de la direction du gradient et des directions précédentes $d^1, d^2, ..., d^n$ c'est-à-dire :

$$d^k = \nabla f(x^k) + \beta_k d^{k-1}$$

Les coefficients de la combinaison linéaire β_k étant choisis de telle sorte que d^k soit conjugué par rapport à toutes les directions précédentes.

Suivant la même idée une variante de la méthode de Frank-Wolfe a été proposée par Daneva & Lindberg [**3**] appelé méthode de Frank-Wolfe conjuguée « *CFW* ». Son principe est le suivant: Soit x^k la solution courante obtenue par minimisation de l'objectif suivant la direction

$$d_{CFW}^{k-1} = S_{CFW}^{k-1} - d^{k-1}$$

du point x^{k-1} vers le point $S_{CFW}^{k-1} \in X$.

Soit $y_{FW}^k \in X$ le point obtenu après résolution du problème linéaire PL(k) au point x^k ; la direction de recherche de la méthode de Frank-Wolfe est déterminée par :

$$d_{FW}^k = y_{FW}^k - x^k$$

et la direction de descente de la méthode de CFW est définie par :

$$d^k = d_{FW}^k + \beta_k d_{CFW}^{k-1},$$

où β_k : est choisi de tel sorte que d^k soit conjuguée par rapport à d_{CFW}^{k-1}.

Le problème dans le choix de cette direction de recherche repose sur la difficulté de détermination de la longueur maximum du pas. Cependant, de manière équivalente la direction de CFW peut être déterminée par

$$d_{CFW}^k = S_{CFW}^k - x^k$$

$$\text{où } S_{CFW}^k = \alpha_k S_{CFW}^{k-1} + (1-\alpha_k) y_{FW}^k \quad 0 \le \alpha_k \le 1$$

est combinaison convexe des deux points y_{FW}^k et S_{CFW}^k qui appartiennent à *X*, dans ce cas la valeur du pas maximum est toujours égale à 1.

L'algorithme de *CFW* peut être résumé en les étapes suivantes:

Etape0 (*Initialisation*) poser $\text{LBD}_0 = -\infty$ (la borne inférieure de f^* "la valeur de la fonction objectif à l'optimum"), *k*=0 et choisir $x_0 \in X$ la solution réalisable de départ.

Etape 1 *Choix de la direction de descente de FW* : au point x^k trouver y_{FW}^k solution du sous problème PL (k), Poser $d_{FW}^k = y_{FW}^k - x^k$

Etape 2 *Choix de la direction de descente de CFW* : Si *k*=0 poser $d_{CFW}^k = d_{FW}^k$

Sinon calculer :

$$\begin{cases} N_k = (S_{CFW}^{k-1} - x^k)^T H_k(x^k) d_{FW}^k, \\ D_k = (S_{CFW}^{k-1} - x^k)^T H_k(x^k)(y_{FW}^k - S_{CFW}^{k-1}), \\ \alpha_k = \begin{cases} \dfrac{N_k}{D_k}, & \text{Si } D_k \neq 0 \text{ et } \dfrac{N_k}{D_k} \in [0, 1-\delta] \\ 0, & \text{sinon} \end{cases} \\ S_{CFW}^k = \alpha_k S_{CFW}^{k-1} + (1-\alpha_k) y_{FW}^k \\ d_{CFW}^k = S_{CFW}^k - x^k \end{cases}$$

Etape 3 *Recherche du pas de descente* : trouver γ_k, solution du problème unidimensionnel :

$$\min_{0 \le \gamma_k \le 1} f(x^k + \gamma_k d_{CFW}^k)$$

Etape 4 Poser $x^{k+1} = x^k + \gamma_k d_{CFW}^k$ et *k*=*k*+*1*.

Etape 5 *test de convergence* : calculer $\text{LBD}_k = \max\{\text{LBD}_{k-1}, f_k(y_{FW}^k)\}$

Si $\dfrac{f(x^k) - \text{LBD}_k}{\text{LBD}_k} < \varepsilon$ **terminer**.

Sinon aller l' **Etape 1**.

Pour plus de performance Daneva & Lindberg [**3**] ont également proposé une autre variante nommée :

III.3. Direction de Frank-Wolfe Bi-conjuguée (BFW)

L'idée de cette méthode est similaire à celle de la variante *CFW* à l'exception d'une seule différence qui réside dans la construction du point S_k, en effet à l'itération k, le point S_{CFW}^k de la variante *CFW* est calculé à partir d'une combinaison convexe du point y_{FW}^k et le point précédent S_{CFW}^{k-1}, tandis que dans la technique *BFW*, le point S_{BFW}^k est combinaison convexe du point y_{FW}^k et les deux points générés précédemment par l'algorithme S_{BFW}^{k-1} et S_{BFW}^{k-2}. Ainsi, la direction de *BFW* peut être déterminée par $d_{BFW}^k = S_{BFW}^k - x^k$ où :

$$S_{BFW}^k = \beta_k^0 y_{FW}^k + \beta_k^1 S_{BFW}^{k-1} + \beta_k^2 S_{BFW}^{k-2} \quad \beta_k^i \geq 0, \text{ avec } \sum_{i=0}^{2} \beta_k^i = 1$$

Les coefficients de la combinaison linéaire β_k^i étant choisis de telle sorte que d^k soit conjuguée, par rapport au deux dernières directions S_{BFW}^{k-1} et S_{BFW}^{k-2}, c'est à dire :

$$(d_{BFW}^k)^T H_k d_{BFW}^{k-1} = 0$$
$$(d_{BFW}^k)^T H_k d_{BFW}^{k-2} = 0.$$

L'algorithme BFW :

Etape0 (*Initialisation*) poser $\text{LBD}_0 = -\infty$, k=0 et choisir $x_0 \in X$ la solution réalisable de départ.

Etape 1 *Choix de la direction de descente de FW* : au point x^k trouver y_{FW}^k solution du sous problème PL (k), Poser

$$d_{FW}^k = y_{FW}^k - x^k .$$

Etape 2 *Choix de la direction de descente de BFW* :

Si k=0 poser $d_{BFW}^k = d_{FW}^k$

Si k=1 poser $d_{BFW}^k = d_{CFW}^k$

Sinon calculer: $\begin{cases} S_{BFW}^k = \beta_k^0 y_{FW}^k + \beta_k^1 S_{BFW}^{k-1} + \beta_k^2 S_{BFW}^{k-2} \\ d_{BFW}^k = S_{BFW}^k - x^k \\ \beta_k^i \geq 0 \ \ \forall i = \overline{0,2}, \text{ avec } \sum_{i=0}^{2} \beta_k^i = 1 \end{cases}$

Etape 3 *Recherche du pas de descente* : trouver γ_k, solution du problème unidimensionnel :

$$\min_{0 \leq \gamma_k \leq 1} f(x^k + \gamma_k d_{BFW}^k)$$

Etape 4 Poser $x^{k+1} = x^k + \gamma_k d^k_{BFW}$ et $k=k+1$.

Etape 5 *test de convergence* : calculer $\text{LBD}_k = \max\{\text{LBD}_{k-1}, f_k(y^k_{FW})\}$.

Si $\dfrac{f(x^k) - \text{LBD}_k}{\text{LBD}_k} < \varepsilon$ **terminer**.

Sinon aller l'**Etape 1.**

Remarque : En ce qui concerne les preuves des convergences de ces deux méthodes voir [**3**].

III.4. Technique du pas de descente non monotone

Afin d'assurer la convergence globale, les algorithmes de combinaison convexes existants (tel l'algorithme de Frank-Wolfe) exigent l'utilisation d'une technique de recherche de pas de descente, qui garantit une diminution monotone de la fonction objectif. Toutefois, cette technique peut ralentir la vitesse de convergence aux étapes intermédiaires du processus de minimisation. Par conséquent, il s'avère qu'une stratégie idéale de recherche de pas de descente pour l'algorithme de Frank-Wolfe devrait permettre une augmentation de la valeur de la fonction à chaque étape, en conservant cependant la convergence globale.

En 1986, Grippo & al [**11**] ont utilisé la technique non monotone de recherche de pas de descente dans la méthode de Newton pour les problèmes d'optimisation sans contraintes. Cette technique a été rapidement appliquée dans d'autres domaines de l'optimisation.

Une nouvelle variante da la méthode de Frank-Wolfe utilisant cette technique de recherche de pas de descente non monotone a été récemment proposée par Ziyou Gao & al. [**26**] pour la résolution du problème d'équilibre utilisateur (voir chapitre IV). Cette technique peut être considérée comme une généralisation de la technique de Grippo et al. Dans la quelle la valeur de la fonction objectif peut être augmentée au cours des itérations mais sans affectation des propriétés de convergence.

Ainsi, le nouvel algorithme noté *IFW* peut être résumé en les étapes suivantes :

Etape1 *Choix de la direction de descente* : trouver y^k solution du sous problème PL (k) :

$$\begin{cases} \min f_k(x^k) \\ x \in X \end{cases}$$

Poser $d^k = y^k - x^k$

Si $\nabla f(x^k)^T d^k = 0$, **stop**.

Etape 2 *Recherche du pas de descente* : trouver $\beta_k = \sigma_{h_k} a,$ avec :

h_k : est le premier entier *h* non négatif pour le quel :

$$f(x^k + \sigma_k a d^k) \leq \max_{j=0,1,\ldots M} \left\{ f(x^{k-j}) \right\} + \gamma \sigma_h a \nabla f(x^k)^T d^k$$

avec $a > 0,\ 0 < \sigma < 1, M$ un entier non négatif et $0 < \gamma < \frac{1}{2}$,

Etape3 Poser $x^{k+1} = x^k + \beta_k d^k$.

Etape 4 Si le *test de convergence* est vérifié. **Stop**

Sinon, poser *k=k+1*, aller à l' **Etape1**.

Remarque: En ce qui concerne la preuve de convergence de cette technique voir [**26**].

Chapitre II

Nouvelles Variantes de la Méthode de Frank-Wolfe

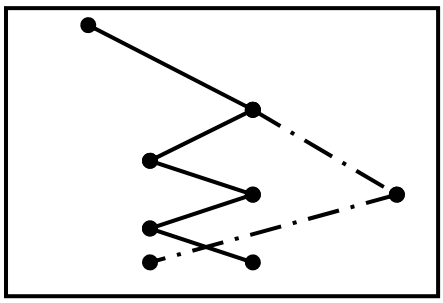
II.1. Introduction

Nous avons vu au chapitre précédent quelques variantes qui ont été développées dans le contexte d'amélioration de la vitesse de convergence de la méthode de Frank-Wolfe [**8**], de notre part, nous nous sommes proposés dans notre étude d'atteindre cet objectif en développant des variantes visant à remédier à cet inconvénient. Nous proposerons à cet effet trois techniques à savoir :

- ✓ La technique du pas élargi notée **FWλ** [**17**];
- ✓ La variante de Fukushima désignée par **FWF** [**10**];
- ✓ Et enfin la nouvelle variante, la variante combinée **FWFλ** déterminée par combinaison des deux premières techniques [**18**].

II.2. Technique du pas élargi

Cette technique consiste à utiliser durant un certain nombre d'itérations un pas de descente plus grand que celui obtenu de l'optimisation unidimensionnelle, tout en gardant les directions de mouvement de base. Il en résulte ainsi un chemin en zigzag plus étendu, où les points de l'algorithme avanceraient plus rapidement vers la solution optimale :



—— Chemin emprunté par les points de l'algorithme de Frank Wolfe.

—· L'effet du pas élargi.

II.2.1. *Présentation de la variante* (FWλ)

Nous allons introduire cette variante sous forme d'un algorithme mixé :

Soit $A_k : \mathbb{R}^n \to \mathbb{R}^n$ L'opérateur algorithmique de la méthode de Frank-Wolfe.

Et $B_k : \mathbb{R}^n \to \mathbb{R}^n$ L'opérateur de la version modifiée.

Avec pour $k \in K \backslash K^0$, $|K^0|$ fini $B = A_k$.

Et pour $k \in K^0$, on introduit la technique du pas élargi dans l'algorithme de Frank-Wolfe.

La modification consiste à modifier la valeur du pas de descente α_κ solution du problème unidimensionnel suivant :

$$\min_{0 \le \alpha \le 1} f(x^k + \alpha d^k)$$

Pour tout $k \in K^0$, les étapes supplémentaires suivantes sont requises :

(i) poser $\gamma_k = \lambda_k \alpha_k$; λ_k est le multiplicateur du pas de descente à l'itération k, il est choisi strictement supérieur à 1.

(ii) Poser $\beta_k = \min(\gamma_k, 1)$. (cette condition nous évite de sortir de la région faisable).

(iii) Soit $x^{k+1} = x^k + \beta_k d.^k$

Si $f(x^{k+1}) < f(x^k)$ poser $x^{k+1} = x^{k+1}$.

Sinon poser $x^{k+1} = x^k + \alpha_k d^k$.

(Si en modifiant le pas de descente, on n'améliore pas la fonction objectif on prend λ_k=1).

II.2.2. *Convergence de l'algorithme* (FWλ)

La convergence de cette technique, découle du théorème de convergence des algorithmes mixés (énoncé au chapitre I), Il suffit de vérifier les conditions de convergence suivantes [**17**] :

1. Pour tout $k \in K^0$; $f(x^{k+1}) \le f(x^k)$.

2. Pour tout $k \in K$; $x^k \in X$, où X est un compact.

3. Soit $x^* \in \Omega$ et soit y un point de l'algorithme ; si $f(y) \le f(x^*)$ alors $y \in \Omega$

Ω est l'ensemble des solutions du problème proposé.

II.2.3 *Démonstration de la convergence*

La condition 1 : est vérifiée d'après (iii) de le variante **(FWλ)**.

La condition 2 : Tous les points construits par l'algorithme sont dans un compact, y compris les points x^k pour k$\in$ K^0, puisque ces points sont toujours choisis sur les intervalles $[x^{k-1}, y^{k-1}]$ en vertu du (ii) de l'algorithme modifié.

La condition 3 : est triviale, étant donnée que l'ensemble des solutions du problème proposé est :

$$\Omega = \{ x^* \,|\, f(x^*) = \min_{x \in S} f(x) \}$$

$$\text{où } X = \{ x \in \mathrm{IR}^n \,|\, Ax \le b \quad x \ge 0 \}$$

Il en résulte ainsi, que la variante de la méthode de Frank Wolfe proposée est globalement convergente.

II.3. Modification de la direction de descente

Dans cette partie nous présenterons une autre version modifiée de l'algorithme de Frank-Wolfe, il s'agit de la variante de Fukushima [**10**], cette dernière est dans le cas le plus défavorable équivalente à celle de la méthode de Frank-Wolfe classique.

On désignera par **(FWF)** cette variante.

II.3.1. *Présentation de la variante* (FWF)

On suppose que la solution optimale appartienne à une face « S » de la région réalisable (on montre que la convergence est moins lente dans le cas où l'optimum est un point intérieur de la région réalisable [**12**]).

S est un polyèdre convexe, donc S peut être représenté par l'enveloppe convexe d'un nombre fini de points extrêmes de la région faisable $y^1, ..., y^n$.

Soit x^* la solution optimale du problème ;

$$x^* \in S, \text{donc il existe } \delta_i ; i = 1,...,n \text{ tels que : } x^* = \sum_{i=1}^{n} \delta_i y^i$$

$$\text{avec} \quad \sum_{i=1}^{n} \delta_i = 1 \qquad \delta_i \ge 0 \quad \text{pour } i=1,\ldots,n.$$

Ceci nous permet d'écrire x^* sous la forme :

$$x^* = x^k + \alpha_k d^k \text{ avec } \alpha_k = 1$$

$$\text{et } d^k = \sum_{i=1}^{n} \delta_i y^i - x^k \ldots\ldots\ldots\ldots (*)$$

Comme il est pratiquement impossible de déterminer avec exactitude les y^i et les δ_i On notera cependant que l'algorithme de Frank-Wolfe génère y^j comme solutions des sous problèmes PL(i), ce qui laisse à penser qu'une combinaison convexe d'un certain nombre des y^i générés précédemment par l'algorithme, pourrait servir à approximer (*) et à fournir ainsi, une direction plus raffinée que celle de Frank-Wolfe, qui utilise seulement le y^i le plus récemment généré. De plus, afin de garantir une efficacité maximale à l'algorithme modifié, les deux directions sont comparées en terme de dérivées directionnelles et celle dont la valeur est la plus petite est sélectionnée pour être la direction de descente actuelle.

II.3.2. *Algorithme de* (FWF)

La variante **FWF** peut être résumé en les étapes suivantes:

Etape0. Soit x^1 une solution réalisable initiale et soit ℓ un entier positif. Poser k=1.

Etape1. Résoudre le sous problème PL (k), soit y^k la solution optimale.

Etape2. Si $\nabla f(x^k)(y^k - x^k) = 0$ terminer, sinon aller à **l'Etape3**.

Etape3. Soit μ_i; i = k-q…, k choisi tels que: $\sum_{i=k-q}^{k} \mu_i = 1, \ \mu_i \geq 0; \ i = k\text{-}q,\ldots,k$

Poser: $v^k = (\sum_{i=k-q}^{k} \mu_i y^k) - x^k$, où q = min {k, ℓ }-1,

Et $w^k = y^k - x^k$ .....(I)

Etape4. Calculer les dérivées directionnelles:

$$\gamma_1^k = \nabla f(x^k) v^k / \|v^k\|$$
$$\gamma_2^k = \nabla f(x^k) w^k / \|w^k\| \quad \text{.....(II)}$$

Poser: $d^k = \begin{cases} v^k & \text{si } \gamma_1^k < \gamma_2^k \\ w^k & \text{sinon} \end{cases}$ .....(III)

Etape5. Résoudre le problème unidimensionnel:

$$\min_{0 \leq \alpha \leq 1} f(x^k + \alpha . d^k) \quad \text{.....(IV)}$$

Soit α_k sa solution optimale.

Poser : $x^{k+1} = x^k + \alpha_k d^k$; .....(V)

k=k+1, aller à **l' Etape1.**

II.3.3. *Convergence de la variante* (FWF)

Supposons que les μ_i i = k-q…, k sont des fonctions continues de $y^{k-q},...,y^k$ et de x^k

I.e. $\mu_i = \mu_i(y^{k-q},...,y^k,x^k)$ i = k-q…, k.

pour tout k et $v^k \neq 0$.

Alors, l'algorithme **FWF** soit se termine en une solution optimale ou bien génère une suite infinie, dont tout point d'accumulation est une solution optimale [**10**].

II.3.4. *Démonstration de la convergence*

L'opérateur algorithmique de la modification **FWF** est : **A =MDC**

C, **D**, **M** sont des opérateurs définis comme suit :

$$C(y^{k-q-1},...,y^{k-1},x^k) = \left\{(y^{k-q},...,y^k,x^k)/y^k \text{ est solution du sous problème PL(k)}\right\}$$

$$D(y^{k-q},...,y^k,x^k) = \left\{(y^{k-q},...,y^k,x^k,d^k)/d^k \text{ est défini par (I)(II)(III)}\right\}$$

$$M(y^{k-q},...,y^k,x^k,d^k) = \left\{(y^{k-q},...,y^k,x^{k+1})/x^{k+1} \text{ est défini par (IV)(V) avec } x^k \text{ et } d^k \text{ donnés}\right\}$$

Ainsi : **C**: résume l'Etape1.

D: résume les Etape 3 et 4

M: résume l'étape5.

Soit Z la fonction de descente donnée par :

$$Z(y^{k-q-1},...,y^{k-1},x^k) = f(x^k)$$

on pose $z^k = (y^{k-q-1},...,y^k,x^k)$.

L'algorithme **FWF** est donc parfaitement défini par l'opérateur algorithmique **A**, qui à z^k associe $z^{k+1} \in A(z^k)$.

L'ensemble des solutions Ω est défini par l'ensemble des points z dont la composante x est solution optimale du problème proposé.

Rappelons les conditions de convergence du théorème de convergence des algorithmes :

i. Tous les points z^k appartiennent à un ensemble compact.

ii. Il existe une fonction continue : $Z : IR^n \rightarrow IR$ telle que :

a. Si $z \notin \Omega$ alors : $y \in A(z)$ on a : $Z(y) < Z(z)$.

b. Si $z \in \Omega$, alors soit l'algorithme s'arrête

Soit $\forall y \in A(z)$ on a : $Z(y) \leq Z(z)$.

iii. L'opérateur A est fermé en tout point $z \notin \Omega$.

Nous allons monter que les points z^k appartiennent à l'ensemble compact C = $\overline{Co(X)}$ où $X = \{x^1, y^1, ..., y^p\}$.

Nous avons : $z^k = (y^{k-q-1}..., y^{k-1}, x^k)$.

Donc, il nous suffit de vérifier que tous les x^k appartiennent à C.

Distinguons deux cas :

1er cas : $x^k = x^{k-1} + \alpha_{k-1} d^{k-1};\ 0 \le \alpha_{k-1} \le 1$

avec $d^{k-1} = v^{k-1} = \sum_{i=k-q-1}^{k-1} \mu_i y^i - x^{k-1}$

et $\sum_{i=k-q-1}^{k-1} \mu_i = 1;\ \mu_i \ge 0,\ i = k-q-1,...,k-1$

d'où : $x^k = (1-\alpha_{k-1})x^{k-1} + \alpha_{k-1} \sum_{i=k-q-1}^{k-1} \mu_i y^i$ .....(VI)

2ème cas : $x^k = x^{k-1} + \alpha_{k-1} d^{k-1};\ 0 \le \alpha_{k-1} \le 1$

Avec $d^{k-1} = w^{k-1} = y^{k-1} - x^{k-1}$

ce qui nous donne: $x^k = (1-\alpha_{k-1})x^{k-1} + \alpha_{k-1} y^{k-1}$ .....(VII)

On remarque dans les deux cas que x^k appartient respectivement aux segments :

$$[x^{k-1}, \sum_{i=k-q-1}^{k-1} \mu_i y^i] \text{ et } [x^{k-1}, y^{k-1}].$$

Or y^{k-1}, tout comme $\sum_{i=k-q-1}^{k-1} \mu_i y^i$ appartiennent à C ; d'où par récurrence sur k = 2,3,…. étant donné que $x \in$ C, on déduit que tous les x^k définis soit par (VI) ou par (VII) appartiennent à C.

La condition (**ii-a**) est vérifiée puisque l'algorithme **FWF** engendre une suite $\{f(x^k)\}$ strictement décroissante la condition (**ii-b**) est aussi vérifiée par l'algorithme qui s'arrête à l'étape 3 si pour un certain k, x^k est une solution optimale .

De ce fait, Il nous reste à montrer, que l'opérateur algorithmique **A=MDC** est fermé. En vertu du corollaire 1 (chapitre I), il suffit de montrer que **M**, **D** et **C** sont fermés. Remarquons que **M** et **C** sont les opérateurs utilisés par l'algorithme de **FW**, puisque ils ne sont pas concernés par la modification de Fukushima, donc la fermeture de **M** et **C** découle du théorème de convergence de l'algorithme de Frank-Wolfe). Par conséquent, il nous reste uniquement à montrer la fermeture de l'opérateur **D**.

Soit $z \notin \Omega$; on suppose que :

$$z^j \in z$$
$$(z^j, d^j) \in D(z^j)$$
$$\text{Et} \quad d^j \in d$$

Soit $v^j, w^j, \gamma_1^j, \gamma_2^j$ et v, w, γ_1, γ_2 défini par (I) et (II) correspondant respectivement à z^j et z.

Par hypothèse, z est non optimal est μ_i i = k-q…, k sont continues par rapport à z^j, il en résulte que $v \neq 0$ et $v^j \neq 0$ pour tous les j suffisamment grand, ainsi que $v^j \to v, w^j \to w, \gamma_1^j \to \gamma_1, \gamma_2^j \to \gamma_2$ lorseque $j \to +\infty$

$$\text{d'où : } d = v \quad \text{si} \gamma_1 < \gamma_2$$
$$\text{et} \qquad d = w \quad \text{si} \gamma_2 \leq \gamma_1$$

Autrement dit, $(z, d) \in D(z)$ et **D** est fermé au point z. ainsi, il en résulte que l'opérateur algorithmique **A** de la variante **FWF** est fermé.

En conclusion, la variante **FWF** est globalement convergente.

II.4. Variante combinée (FWFλ)

Dans ce qui suit, nous allons présenter une variante qui est le résultat de la combinaison des deux techniques exposées plus haut, pour des questions de convergence, la conception de cette variante doit avoir la forme d'un algorithme mixé, ce qui nous laisse par conséquent deux possibilités à distinguer :

1^{ére} possibilité

Considérer l'algorithme (**FWF**) comme algorithme de base et la variante de pas descente la modification à incorporer.

2^{éme} possibilité

Prendre l'algorithme (**FWλ**) comme algorithme de base et la modification de Fukushima comme mesure de compensation.

Etant donné que la variante (**FWF**) agit sur les directions de descente qui sont responsables du phénomène de zigzag et d'après son schéma de construction (voir la section II.3.2.), cette variantes est dans le pire des cas équivalente à la méthode de Frank-Wolfe classique, ce qui représente une garantie sûre de l'amélioration de la vitesse de convergence, contrairement à la variante (**FWλ**) qui repose sur une technique heuristique.

Ainsi, la première possibilité est la mieux appropriée pour concevoir un algorithme mixé.

II.4.1. *Principe de la variante* (FWFλ)

Dans cette variante la technique du pas élargi sera appliquée durant les "ℓ" premières itérations, où la direction de Frank-Wolfe est conservée, vu que le nombre "ℓ" des y^k utilisés dans la modification de Fukushima n'est pas encore atteint ; pour n'appeler le programme (**FWF**) qu'ensuite, de la manière suivante :

II.4.2. *Algorithme de* (FWFλ)

Etape0. Soit x^1 une solution réalisable initiale et soit ℓ un entier positif. Poser k=1.

Etape1. Résoudre le sous problème PL (k), soit y^k la solution optimale.

Etape 2. Recherche de la direction de descente:

Soit μ_i, i = k-q…, k choisi tels que: $\sum_{i=k-q}^{k} \mu_i = 1, \ \mu_i \geq 0; \ i = k-q,\ldots,k$

$$\text{Poser: } v^k = \left(\sum_{i=k-q}^{k} \mu_i y^k\right) - x^k, \text{ où q = min } \{k, \ell\}\text{-1,}$$

$$\text{et} \quad w^k = y^k - x^k$$

Calculer les dérivées directionnelles:

$$\gamma_1^k = \nabla f(x^k) v^k / \left\| v^k \right\|$$

$$\gamma_2^k = \nabla f(x^k) w^k / \left\| w^k \right\|$$

$$d^k = \begin{cases} v^k & \text{si } \gamma_1^k < \gamma_2^k \\ w^k & \text{sinon} \end{cases}$$

Etape 3. Recherche unidimensionnelle : Déterminer α_k qui minimise :

$$\min_{0 \leq \alpha \leq 1} f(x^k + \alpha d^k)$$

Si $k \in K^0$ aller à **l'Etape 4.**

Si non aller à **l'Etape 5.**

Etape 4. Calcule du pas élargi :

Poser $\gamma_k = \lambda_k \alpha_k$; λ_k est le multiplicateur du pas de descente à l'itération k, avec $\lambda_k > 1$.

Poser $\beta_k = \min(\gamma_k, 1)$.

Soit $x^{k+1} = x^k + \beta_k d.^k$

Si $f(x^{k+1}) < f(x^k)$ poser $x^{k+1} = x^{k+1}$ aller à **l'Etape 6**.

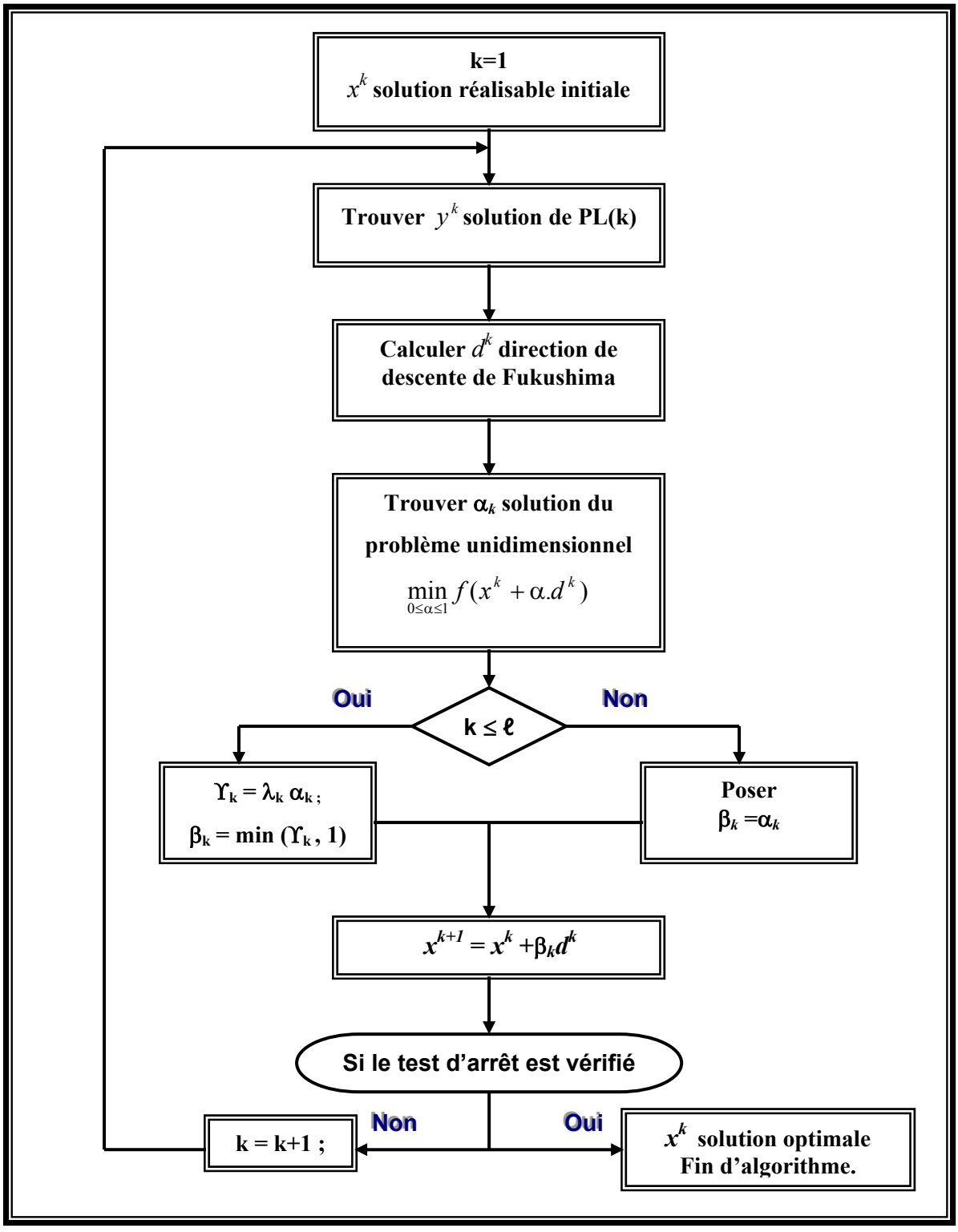
Sinon aller à **l'Etape 5.**

Etape 5. Déterminer la nouvelle solution x^{k+1} avec : $x^{k+1} = x^k + \alpha_k d^k$

Etape 6. Test de convergence : Si le critère d'arrêt est vérifier, **terminer**.

Sinon, mettre $k = k + 1$, aller à **l'Etape1.**

II.4.3. *Organigramme de l'algorithme* (FWFλ)



II.4.4. *Convergence de la variante* (FWFλ)

Nous avons vu précédemment que la conception de la variante combinée a la forme d'un algorithme mixé, en considérant l'algorithme de Fukushima comme algorithme de base et la variante **FWλ** la modification à incorporer, pendant un certain nombre d'itération K^0 avec "$|K^0|=\ell$ " :

Pour tout k$\in K^0$, les étapes supplémentaires suivantes sont requises :

1. Poser $\gamma_k = \lambda_k \alpha_k$; λ_k est le multiplicateur du pas de descente à l'itération k, il est choisi strictement supérieur à 1.
2. Poser $\beta_k = \min(\gamma_k, 1)$. (cette condition nous évite de sortir de la région faisable).
3. Soit $x^{k+1} = x^k + \beta_k d.^k$

$$\text{Si } f(x^{k+1}) < f(x^k) \text{ poser } x^{k+1} = x^{k+1}.$$
$$\text{Sinon poser } x^{k+1} = x^k + \alpha_k d^k.$$

La convergence de la variante **FWFλ**, découle du théorème de convergence des algorithmes mixés, Il suffit de vérifier les conditions de convergence suivantes:

- **i.** Pour tout k$\in K^0$; $f(x^{k+1}) \le f(x^k)$.
- **ii.** Pour tout k$\in$ K; $x^k \in C$, où C est un compact.
- **iii.** Soit $x^* \in \Omega$ et soit y un point de l'algorithme ; si $f(y) \le f(x^*)$ alors $y \in \Omega$

  Ω est l'ensemble des solutions du problème proposé.

II.4.5. *Démonstration de la convergence*

La condition 1 : est vérifiée d'après (**3**) de le variante **FWλ**.

La condition 2 : Tous les points construits par l'algorithme sont dans un compact (voir la section **II.3.4.i.**), y compris les points x^k pour $k\in K^0$, puisque ces points sont toujours choisis sur les intervalles $[x^{k-1}, y^{k-1}]$ d'après (**2**) de la technique **FWλ**.

La condition 3 : est triviale. Étant donnée que l'ensemble des solutions du problème proposé est :

$$\Omega = \{ x^* \,|\, f(x^*) = \min_{x \in S} f(x) \}$$

$$\text{où } X = \{ x \in \mathrm{IR}^n \,|\, Ax \le b \quad x \ge 0 \}$$

Il en résulte ainsi, que cette nouvelle variante de la méthode de Frank Wolfe est globalement convergente.

Chapitre III

Réseau de trafic urbain

III.1. Introduction

Les décisions individuelles sont la cause principale de l'écoulement du trafic présent dans les zones urbaines. D'une part, ces décisions dépendent, en partie tout au moins, du niveau de congestion du réseau et de la localisation des points de congestion. D'autre part, la congestion en tout point du réseau est fonction du nombre de véhicules présents en ce point, et donc des décisions individuelles des usagers du réseau.

Dans ce qui suit, nous allons voir comment modéliser ces interactions entre congestion et décision de déplacement pour obtenir les flux de véhicules résultant sur chacun des axes du réseau de transport urbain.

Ainsi, afin de déterminer les flux pour chacun des axes du réseau, on partira de la remarque suivante :

La congestion s'accroît avec le flux de véhicules sur l'arc, mais, d'autre part, la congestion fait dévier les véhicules des axes encombrés.

Les flux résulteront donc d'un équilibre entre ces deux mouvements en sens opposé [**21**].

III.2. Représentation du réseau de trafic urbain

Le réseau réel, constitué de routes et carrefours, que l'on recherche à modéliser, est représenté par un réseau, au sens de la théorie des graphes, tel que celui de la figure ci-dessous.

C'est sur ce réseau que l'on va représenter l'écoulement des flux de véhicules.

Un carrefour est représenté par un sommet. Certains de ces sommets sont des entrées du réseau (origines), où les véhicules apparaissent. De façon symétrique, d'autres sont des sorties de réseau (destinations).

Ces sommets sont reliés par des arcs qui représentent les portions de routes entre les carrefours. En matière de circulation routière, on travaille uniquement avec des arcs orientés.

Chaque arc a un seul sommet de départ et un seul sommet d'arrivé. En revanche, un sommet peut avoir plusieurs arcs incidents intérieurs ou extérieurs.

La figure suivante [**5**] représente un réseau de 5 sommets connectés par 10 arcs.

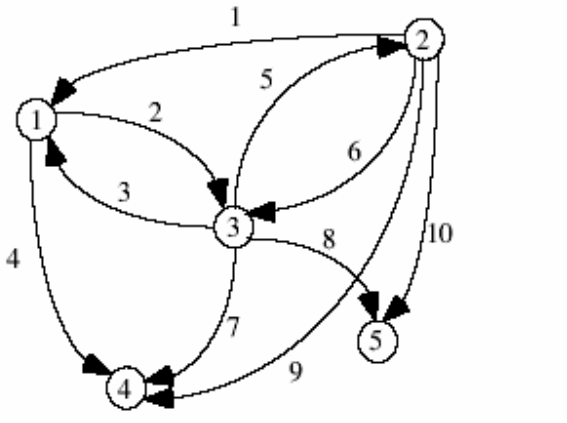


Figure* III.1** ***: Représentation d'un réseau

Ainsi l'arc 2 représente le flux de 1 à 3 et l'arc 3 le flux de 3 vers 1.

Pour terminer la représentation du réseau, il faut définir *les fonctions de performance des arcs.*

III.3. Fonctions de performance des arcs

En trafic urbain, ce qui est déterminant pour le choix d'un itinéraire est le *temps de parcours*, plutôt que la distance. Ainsi, généralement, c'est ce critère de mesure de la performance d'un arc qui est retenu.

Le temps de traversée d'un arc dépend du flux traversant cet arc, plus l'arc est chargé, plus le temps de traversée augmente.

Ainsi, *la fonction de performance* donne la relation entre le temps de traversée de l'arc et le flux qui le traverse. Elle a typiquement la forme illustrée à la figure suivante **[5]**:

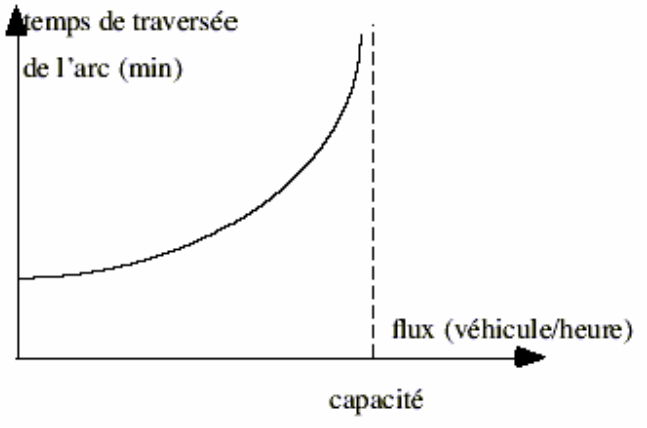


Figure* III.2 *: Fonction de performance

Elle est croissante, et non linéaire à cause des effets de congestion du réseau.

Comme le montre la figure précédente, la fonction de performance augmente très lentement, au début, c'est tout à fait normal car la circulation est fluide. Cependant, quand le flot commence à atteindre un certain volume (correspondant à la capacité de l'arc), le temps de parcours commence à croître très rapidement.

La fonction de performance de l'arc (i, j) donnant le temps de traversée de l'arc en fonction du flux est notée : $t_{ij} = t_{ij}(x_{ij})$

La forme générale de la fonction de performance est :

$$t_{ij} = t_{ij}^{0}(1 + (\frac{x_{ij}}{C_{ij}})^{m_{ij}})$$

Avec :

t_{ij}^{0} est le temps de traversée à vide.

x_{ij} le flot total sur l'arc (i,j).

C_{ij} la capacité de l'arc (i,j).

m_{ij} est une constante.

La plus populaire est : $t_{ij} = t_{ij}^{0}(1 + (\frac{x_{ij}}{C_{ij}})^{4})$

En général, les paramètres de cette fonction dépendent des caractéristiques physiques de la voirie (longueur, largeur, présence de feu de croisement, conception géométrique,…etc.).

III.4. La notion d'équilibre

Pour illustrer la manière dont un point d'équilibre est atteint sur un réseau de transport, utilisons l'exemple [**21**] d'un grand axe de pénétration urbain deux voies dans chaque sens qui est systématiquement congestionné à l'heure de pointe.

Les autorités responsables de l'infrastructure estiment que le flux actuel pourrait être fluidifié moyennant une troisième voie dans chaque sens.

On construit cette troisième voie et que constate-t-on au bout de quelques jours ? Le flux total sur l'axe a augmenté car des usagers qui se déroutaient sur les voies latérales voyant l'amélioration sur l'axe principal, vont rejoindre cet axe principal. Ceci va décongestionner à son tour, les voies latéralement qui vont voir arriver un nouveau trafic.

L'équilibre sera atteint lorsque aucun usager n'aura plus intérêt à changer d'itinéraire pour diminuer son temps de trajet.

La définition d'équilibre [5]

La distribution des utilisateurs sur le réseau urbain peut être modélisée grâce à la notion d'équilibre. On veut déterminer comment les voitures se répartissent sur le réseau; On déterminera ainsi le flux (et, en conséquence, le temps de traversée) sur chaque arc du réseau.

Il faut encore spécifier la règle en fonction de laquelle les conducteurs vont choisir leur itinéraire. Il semble raisonnable de penser que, en ville, chaque utilisateur du réseau choisit sa route de façon **à minimiser son temps de trajet total.**

« Le temps de trajet d'un automobiliste est la somme des temps de traversée des arcs de son itinéraire. Le temps de traversée de chaque arc dépendant du nombre d'usagers qui le traversent au même moment ».

Définition : *Une condition d'équilibre est qu'aucun usager ne puisse améliorer son temps de trajet en changeant unilatéralement de route. Cette caractérisation est connue sous le nom de* ***condition d'équilibre de l'utilisateur*** (*Premier principe de Wardrop* [**22**]).

Trois remarques sur cette définition :

1. Cette définition suppose que les usagers sont parfaitement informes de l'état de congestion du réseau et connaissent les déviations.

2. Ils agissent de façon rationnelle (en cherchant leurs plus court chemin) et autonome (sous la contrainte d'aucune autorité).

3. Cette définition est statique, c'est-à-dire pour une demande donnée. Hors la demande évolue dans la journée. Aussi, généralement, le genre de modèle est calculé pour l'heure de pointe du matin ou du soir.

III.5. Modélisation du problème d'équilibre de l'utilisateur

Un problème important dans l'étude du trafic routier est l'évaluation des flots de voitures sur les routes d'un réseau donné. Les demandes de transport pour tous les couples origines destinations étant supposées connues, le calcul des flots sur les arcs est souvent ramené au calcul d'un *équilibre de Wardrop* [**22**] qui consiste à supposer que les usagers ont une information parfaite sur le réseau qui les conduit à n'utiliser que les routes de temps de parcours minimal. On se donne alors sur chaque route une relation liant le temps de parcours au flot sur chaque arc. L'équilibre sera atteint si tous les chemins utilisés pour aller d'un point à un autre ont le même temps de parcours.

Une formulation mathématique plus précise de l'équilibre de l'utilisateur peut s'énoncer de la manière suivante :

Définition : *Au point d'équilibre, pour chaque paire origine-destination,*

- *les temps de trajet de tous les chemins utilisés sont égaux,*
- *ces temps de trajet sont inférieurs au temps de trajet qui serait expérimenté par un véhicule unique sur un chemin non utilisé pour cette paire.*

Nous allons maintenant s'intéresser à la formulation du problème de l'équilibre de l'utilisateur sous forme d'un problème d'optimisation.

III.5.1. *Notations Générales*

- Le réseau sera représenté par un graphe orienté noté $\boldsymbol{G=(N, A)}$ où $\boldsymbol{N}$ est un ensemble de sommets numérotés consécutivement, et $\boldsymbol{A}$ est un ensemble d'arcs également numérotés consécutivement.
- Notons $\boldsymbol{O}$ l'ensemble des sommets d'origine et $\boldsymbol{D}$ l'ensemble des sommets de destination.
- Les sommets sont numérotés de 1 à n, les r premiers sommets sont des destinations « $|\boldsymbol{D}| = r$ » et s représente le nombre de sommets origines « $|\boldsymbol{O}| = s$ ».
- La matrice ***origine destination*** est notée $\boldsymbol{Q}$, de format $s \times r$, autrement dit $\boldsymbol{q_{od}}$ est le taux de voyage entre l'origine $\boldsymbol{o}$ et la destination $\boldsymbol{d}$ exprimé en véhicules par unité de temps (souvent l'heure). « C'est la demande totale entre la paire ***od »***.

- Soit t_{ij} le temps de traversée de l'arc **(*i, j*)**.

  i et **j** représentent respectivement le sommet extrémité initiale et le sommet extrémité finale de l'arc.
- c_p représente le temps de parcours d'un chemin ***p***.
- $\delta_{(i,j),p}$ est une matrice ***d'incidence arc/chemin***, indiquant à quels chemins ***p*** l'arc **(*i, j*)** appartient, tel que :

$$\delta_{(i,j),p} = \begin{cases} 1 & \text{Si l'arc } (i,j) \text{ appartient au chemin p,} \\ 0 & \text{Sinon} \end{cases}$$

- $\delta'_{p,od}$ est une matrice ***d'incidence chemins/paires*** indiquant à quelle paire ***od*** un chemin ***p*** appartient. Elle est définie comme suit :

$$\delta'_{p,od} = \begin{cases} 1 \text{ si } p \in od \\ 0 \quad \text{sinon} \end{cases}$$

- On note par :

  $x = (...,x_{ij},...)$ le vecteur des flux sur les arcs.

  $t = (...,t_{ij},...)$ le vecteur des temps de traversées des arcs.

III.5.2. *Définition des Variables*

On définit trois variables :

x_{ij} Représente la valeur du flot totale sur l'arc (*i, j*).

x_{ij}^{d} Représente la valeur du flot sur l'ars (*i, j*) de destination ***d***.

C_{od} ensemble des chemins reliant la paire ***od***.

f_p Note le flux le long du chemin ***p***, $\boldsymbol{p} \in C_{od}$.

III.5.3. *Fonction objectif*

Le problème d'affectation du trafic, a pour but la minimisation du temps de parcours sur tous les chemins utilisés entre une même paire origine destination. Ainsi L'expression de la fonction objectif est la suivante :

$$\min z(x) = \sum_{(i,j)\in A} \int_0^{x_{ij}} t_{ij}(\omega) d\omega$$

Le problème de l'équilibre de l'utilisateur peut être modélisé sous forme d'un programme non linéaire avec des contraintes linéaires [**20**] ; toutefois il existe deux formulations équivalentes possibles pour ce problème :

- la version arc- sommet si on considère les variables comme étant les flots sur les arcs.
- la version arc-chemin si on considère les variables comme étant les flots sur les chemins.

III.5.4. *La formulation arc- sommet*

Pour chaque sommet **i** $\in \boldsymbol{N}$, on définit l'ensemble des sommets successeurs qui recevront du trafic découlant de **i**, et l'ensemble des sommets prédécesseurs émettant du trafic vers **i**, désignés par $\Gamma^{+}(i)$ et $\Gamma^{-}(i)$ respectivement, tel que :

$$\Gamma^{+}(i) = \{ \text{j avec j} \in \boldsymbol{N} / (i, j) \in \boldsymbol{A} \}.$$

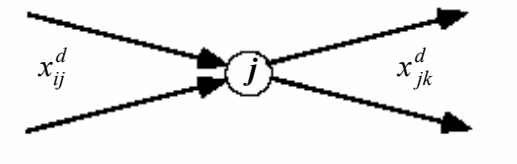
$$\Gamma^{-}(i) = \{ \text{j avec j} \in \boldsymbol{N} / (j, i) \in \boldsymbol{A} \}.$$

Les contraintes

Elles caractérisent la loi de conservation de flot pour le réseau de trafic urbain :

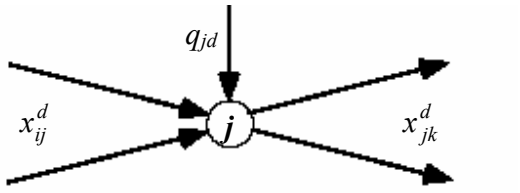
- L'ensemble de ces variables de flux doit satisfaire une équation de conservation du flux en chaque sommet. Cette loi de conservation exprime simplement que la somme des flux sortant d'un sommet est égale à la somme des flux entrant. Elle s'interprète dans notre cas par (voir figure suivante) :

$$\sum_{\text{i} \in \Gamma^{-}(j)} x_{ij}^{d} = \sum_{k \in \Gamma^{+}(j)} x_{jk}^{d} \qquad \forall d \in \boldsymbol{D}; \forall j \notin \boldsymbol{O};\ j \neq d\ .$$



- En un sommet d'offre, c'est à dire en un sommet d'injection dans le réseau (dans notre cas un sommet origine), notons q_{od}, l'injection dans le réseau. L'équation précédente devient (voir figure suivante):

$$\sum_{\text{i} \in \Gamma^{-}(j)} x_{ij}^{d} - \sum_{k \in \Gamma^{+}(j)} x_{jk}^{d} = -q_{jd} \qquad \forall j \in \boldsymbol{O}\ ; \forall d \in \boldsymbol{D};\ j \neq d\ .$$



Ainsi le problème sera formulé comme suit :

$$\min z(\mathrm{x}) = \sum_{(i,j)\in \boldsymbol{A}} \int_0^{x_{ij}} t_{ij}(\omega) d\omega$$

$$\text{s.c.} \begin{cases} \sum_{i\in\Gamma^-(j)} x_{ij}^d - \sum_{k\in\Gamma^+(j)} x_{jk}^d = \begin{cases} -q_{jd} & \text{si } j \text{ est une origine} \\ 0 & \text{sinon} \end{cases} & \forall j = 1,...,n\,; \forall d = 1,...,r\,;\, j \neq d \\ x_{ij}^d \geq 0 & \forall (i,j) \in \boldsymbol{A}\ \ \forall d = 1,...,r \end{cases}$$

III.5.5. *La formulation arc- chemin*

a) ***Les contraintes***

- Le temps de parcours d'un chemin quelconque ***p*** entre l'origine *o* et la destination, *d* est la somme des temps de parcours des arcs successifs du chemin entre *o* et *d* :

$$c_p = \sum_{(i,j)\in A} t_{ij}\delta_{(i,j),p} \qquad (1)$$

- Un flux sur un arc (i, j), « x_{ij} », sera décomposé en la somme des flux le long des différents chemins ***p*** qui passent par cet arc :

$$x_{ij} = \sum_{o=1}^{s}\sum_{d=1}^{r}\sum_{p\in C_{od}} f_p \delta_{(i,j),p} \quad \forall (i,j) \in \boldsymbol{A} \qquad (2)$$

- Enfin, la demande totale entre *o* et *d*, « $\boldsymbol{q_{od}}$ », est répartie entre les chemins ***p*** liant *o* à *d* :

$$\sum_{p\in C_{od}} f_p \delta'_{p,od} = q_{od} \quad \forall o = 1,...,s; \forall d = 1,...,r. \qquad (3)$$

Remarque:

On regroupe en un seul vecteur tous les flux le long des chemins **f** = (. . ., $\boldsymbol{f_p}$, . . .) où l'on range l'un à la suite de l'autre tous les flux le long de tous les chemins de la première paire origine-destination, Puis tous les chemins le long de la seconde paire origine-destination, etc. De la même manière, on regroupe en un seul vecteur tous les coûts le long de tous les chemins **c** = (. . ., $\boldsymbol{c_p}$, . . .).

Ainsi, le flot correspondant à un équilibre de l'utilisateur peut être obtenu en résolvant le problème de programmation mathématique suivant :

$$\min z(x) = \sum_{(i,j)\in A} \int_0^{x_{ij}} t_{ij}(\omega) d\omega$$

$$\text{s.c.} \begin{cases} \sum_{p\in C_{od}} f_p \delta'_{p,od} = q_{od} & \forall o = 1,...,s; \forall d = 1,...,r. \\ f_p \geq 0 & \forall p \in C_{od};\ o = 1,...,s; \\ & d = 1,...,r. \\ x_{ij} = \sum_{o=1}^{s} \sum_{d=1}^{r} \sum_{p\in C_{od}} f_p \delta_{(i,j),p} & \forall (i,j) \in A \quad .....(5) \end{cases} \qquad (4)$$

b) ***Interprétation des formules***

- La fonction objectif est la somme des intégrales des fonctions de performance des arcs, elle correspond au temps de parcours moyen sur tous les arcs du graphe.
- La première contrainte représente les équations de conservation du flux :
  Elles établissent que la somme des flux sur tous les chemins ***p*** reliant la paire ***od*** est égale au taux de voyage entre ***o*** et ***d***.
- La deuxième contrainte assure que la solution soit physiquement réaliste : elles assurent que les flux soient positifs ou nuls.

Beckmann [**2**] a démontré que le vecteur ***x*** de flux sur tous les arcs du réseau qui est solution du programme mathématique formulé précédemment vérifie l'équilibre de Wardrop.
Toute fois, il y a deux hypothèses pour que cette formulation soit équivalente au problème d'équilibre de l'utilisateur :

Hypothèse 1 : On suppose que le temps de traversée d'un arc n'est fonction que du flux de cet arc. Mathématiquement :

$$\frac{\partial t_{ij}(x_{ij})}{\partial x_{kl}} = 0, \quad \forall (i,j) \neq (k,l) \qquad (6)$$

Hypothèse 2 : Les fonctions de performance des arcs sont strictement croissantes :

$$\frac{\partial t_{ij}(x_{ij})}{\partial x_{ij}} > 0, \quad \forall (i,j) \in A \qquad (7)$$

Théorème **1** [20]

Si x est une solution du problème (4), alors x fournit bien une solution du problème d'équilibre de l'utilisateur, (à savoir que les conditions d'équilibres seront vérifiées)

Démonstration

Nous allons montrer que le flot qui résout le problème (4) satisfait aussi les conditions d'équilibre de l'utilisateur. Rappelons qu'en l'optimum d'un problème d'optimisation, les conditions de Kuhn et Tucker doivent être satisfaites.

***Conditions de Kuhn-Tucker* :**

Soit x* un minimum local pour le problème :

$$\min f(x)$$

$$\text{s.c. } h(x) = 0,\ g(x) \le 0$$

et supposons x* régulier pour les contraintes. Alors il existe des multiplicateurs

$$\lambda \in R^m \text{ et } \mu \in R^p \text{ avec } \mu \ge 0 \text{ tels que: } \nabla f(x^*) + \sum_{i=1}^{m} \lambda_i \nabla h_i(x^*) + \sum_{k=1}^{p} \mu_k \nabla g_k(x^*) = 0$$

$$\mu_k g_k(x^*) = 0,\ \forall k = 1,...p$$

Nous allons voir que ces conditions correspondent pour le problème (4) précisément aux conditions d'équilibre de l'utilisateur.

Pour écrire ces conditions, écrivons le Lagrangien correspondant au problème (4) :

$$L(\mathbf{f}, \lambda, \mu) = z(\mathbf{x}(\mathbf{f})) + \sum_{(o,d)} \lambda_{od} (q_{od} - \sum_{p \in C_{od}} f_p \delta'_{p,od}) - \sum_{p \in C_{od}} \mu_p f_p$$

Où λ_{od} note le multiplicateur associé à l'équation de conservation de flux de la paire origine- destination od et μ_p note le multiplicateur associé à l'inéquation : $-f_p \le 0$

Remarquons aussi que l'on a substitué à **x** son expression en fonction de **f** donnée par la relation (5) pour avoir un problème en terme de flux de chemins uniquement.

Les conditions du premier ordre s'écrivent :

$$\frac{\partial L(\mathbf{f}, \lambda, \mu)}{\partial f_p} = 0$$

$$\mu_p f_p = 0$$
$$\mu_p \ge 0$$

Calculons chacun des termes de la dérivée partielle du Lagrangien par rapport à f_p :

$$\frac{\partial z(\mathbf{x}(\mathbf{f}))}{\partial f_p} = \sum_{(i,j)\in A} (\frac{\partial z}{\partial x_{ij}})(\frac{\partial x_{ij}}{\partial f_p}) = \sum_{(i,j)\in A} (t_{ij})(\delta_{(i,j),p}) = c_p$$

D'autre part : $$\frac{\partial\left[\sum_{(o,d)} \lambda_{od}(q_{od} - \sum_{p\in C_{od}} f_p \delta'_{p,od}\right]}{\partial f_p} = -\lambda_{od}$$

où *od* est la paire à laquelle appartient le chemin ***p***.

Donc les conditions du premier ordre s'écrivent :

$$\begin{aligned} c_p - \lambda_{od} - \mu_p &= 0 \\ \mu_p f_p &= 0 \\ \mu_p &\geq 0 \end{aligned}$$

On peut éliminer μ_p du système pour obtenir :

$$(c_p - \lambda_{od}) \geq 0 \qquad \text{(I)}$$

$$f_p(c_p - \lambda_{od}) = 0 \qquad \text{(II)}$$

Remarquons que les conditions de complémentarité (II) impliquent que : si $f_p > 0$, alors $c_p = \lambda_{od}$

D'autre part, les inéquations (I) impliquent que : si $f_p = 0$, alors $c_p \geq \lambda_{od}$

Ce sont précisément les conditions d'équilibre. Le multiplicateur de Lagrange λ_{od} représentant le temps minimum entre l'origine *o* et la destination *d*.

Ainsi, les deux problèmes sont équivalents.

Théorème 2 [20]

Le problème de l'équilibre de l'utilisateur formulé précédemment, admet une solution unique.

Démonstration

Comme les deux problème sont équivalent (d'après le théorème1), cela revient à prouver l'unicité de la solution du problème (4).

Pour prouver que la solution de (4) est unique, il suffit de montrer que $z(\mathbf{x})$ est une fonction strictement convexe.

Comme la région est linéaire, donc elle est convexe.

Ainsi, il suffit de montrer que la matrice Hessienne des dérivées partielles seconde $\nabla^2 z(\mathbf{x})$ est définie positive.

Une matrice H est définie positive si pour tout vecteur x, $x^T Hx \geq 0$.

Les dérivées du premier ordre sont : $\dfrac{\partial z(\mathbf{x})}{\partial x_{ij}} = t_{ij}(x_{ij})$

Et les dérivées du second ordre sont, par **l'hypothèse 1** :

$$\frac{\partial z(\mathbf{x})}{\partial x_{ij}\partial x_{kl}} = \begin{cases} t'_{ij}(x_{ij}) & \text{Si } (i, j) = (k, l) \\ 0 & \text{Sinon.} \end{cases}$$

Ce qui donne une matrice des dérivées partielles secondes diagonales :

$$\nabla^2 z(x) = \begin{bmatrix} t'_{i_1 j_1}(x_{i_1 j_1}) & 0 & \cdots & 0 \\ 0 & t'_{i_2 j_2}(x_{i_2 j_2}) & \cdots & 0 \\ & & \ddots & \vdots \\ 0 & 0 & \cdots & t'_{i_n j_n}(x_{i_n j_n}) \end{bmatrix}$$

Cette matrice est définie positive car elle est diagonale et tous ses éléments diagonaux sont strictement positifs par **l'hypothèse 2.**

On en conclut qu'il existe une solution unique au problème (4) en terme de flux d'arc x_{ij} .

III.6. Remarque

Lorsque l'équilibre utilisateur est atteint, les chemins utilisés sont effectivement les plus courts en terme de temps de parcours. Dans le cas où un agent extérieur au réseau peut agir sur les flux, en les orientant à sa demande (cela suppose une coopération des utilisateurs qui doivent se soumettre aux ordres du gestionnaire) pour minimiser le temps de parcours globale, on atteint ***un équilibre système*** (appelé également ***équilibre social***) résumé par le second principe de Wardrop [**22**]:

« *Le temps moyen de parcours pour l'ensemble des utilisateurs est minimum* ».

Dans cette hypothèse tous les utilisateurs n'utilisent pas le plus court chemin. Le flot résultant est tel que les coûts marginaux (temps de parcours) sur les chemins inutilisés pour relier une paire origine destination sont égaux et plus petits que ceux des chemins inutilisés reliant la même paire origine destination.

Le problème d'équilibre social peut être modélisé sous forme d'un programme non linéaire, dont les contraintes sont les mêmes contraintes du problème d'équilibre utilisateur formulé précédemment , et la fonction objectif est définie par :

$$\min z(x) = \sum_{(i,j)\in A} x_{ij} t_{ij}(\omega) d\omega$$

et qui correspond à la minimisation du temps de parcours pour l'ensemble des utilisateurs.

Ainsi, d'une manière générale, le problème de l'équilibre de l'affectation du trafic urbain peut être modélisé comme suit :

$$\min z(\mathrm{x}) = \sum_{(i,j)\in A} f_{ij}\left(\sum_{d=1}^{r} x_{ij}^{d}\right)$$

$$\text{s.c.} \begin{cases} \sum_{i\in\Gamma^{-}(j)} x_{ij}^{d} - \sum_{k\in\Gamma^{+}(j)} x_{jk}^{d} = \begin{cases} -q_{jd} & \text{si } j \text{ est une origine} \\ 0 & \text{sinon} \end{cases} & \forall j = 1,...,n\,; \forall d = 1,...,r\,; j \neq d \\ x_{ij}^{d} \geq 0 & \forall (i,j) \in A \;\; \forall d = 1,...,r \end{cases}$$

Dans le cas d'une présentation arc-sommet, et :

$$\min z(\mathrm{x}) = \sum_{(i,j)\in A} f_{ij}\left(\sum_{d=1}^{r} x_{ij}^{d}\right)$$

$$: \qquad \text{s.c.} \begin{cases} \sum_{p\in C_{od}} f_p \delta'_{p,od} = q_{od} & \forall o = 1,...,s; \forall d = 1,...,r. \\ f_p \geq 0 & \forall p \in C_{od}\,;\; o = 1,...,s; \\ & \qquad d = 1,...,r. \\ x_{ij} = \sum_{o=1}^{s}\sum_{d=1}^{r}\sum_{p\in C_{od}} f_p \delta_{(i,j),p} & \forall (i,j) \in A \end{cases}$$

Pour la représentation arc-chemin.

Avec $$f_{ij}\left(\sum_{d=1}^{r} x_{ij}^{d}\right) = \begin{cases} \int_0^{x_{ij}} t_{ij}(\omega) d\omega & \text{pour le problème d'équilibre utilisateur} \\ x_{ij} t_{ij}(x_{ij}) & \text{pour le problème d'équilibre système.} \end{cases}$$

Chapitre IV

Résolution du problème de l'affectation du trafic urbain « *par les nouvelles variantes de la méthode de Frank-Wolfe* »

IV.1. Introduction

Pour rappel, le problème d'affectation du trafic urbain, formulé au chapitre III, a pour but la détermination des flux sur le réseau (et en conséquence, le temps de traversée) sur chaque arc du réseau. Nous avons également vu au chapitre III comment ce problème pouvait être formulé comme un problème de minimisation d'une fonction convexe sous des contraintes purement linéaires. Une méthode particulièrement bien adaptée est la méthode de Frank-Wolfe (vue au chapitre I). Nous verrons dans ce qui suit que cette méthode utilise largement des sous problèmes pour la détermination du plus court chemin sur un graphe.

IV.2. Résolution du problème de l'affectation du trafic urbain

D'après le chapitre précédent, le problème de l'équilibre de l'affectation du trafic urbain pouvait être formulé comme un problème d'optimisation convexe, sous deux formes équivalentes:

La première étant la représentation « arc sommet » :

$$\min z(\mathrm{x}) = \sum_{(i,j)\in A} f_{ij}\left(\sum_{d=1}^{r} x_{ij}^{d}\right)$$

$$\text{s.c.}\begin{cases} \sum_{i\in\Gamma^{-}(j)} x_{ij}^{d} - \sum_{k\in\Gamma^{+}(j)} x_{jk}^{d} = \begin{cases} -q_{jd} & \text{si } j \text{ est une origine} \\ 0 & \text{sinon} \end{cases} & \forall j = 1,...,n\,;\forall d = 1,...,r\,;\, j \neq d \\ x_{ij}^{d} \geq 0 & \forall (i,j) \in A \;\; \forall d = 1,...,r \end{cases} \quad \text{(I)}$$

La seconde est la représentation « arc chemin » :

$$\min z(\mathrm{x}) = \sum_{(i,j)\in A} f_{ij}\left(\sum_{d=1}^{r} x_{ij}^{d}\right)$$

$$\text{s.c.} \begin{cases} \sum_{p\in C_{od}} f_p \delta'_{p,od} = q_{od} & \forall o = 1,...,s; \forall d = 1,...,r. \\ f_p \geq 0 & \forall p \in C_{od};\ o = 1,...,s; \\ & d = 1,...,r. \\ x_{ij} = \sum_{o=1}^{s}\sum_{d=1}^{r}\sum_{p\in C_{od}} f_p \delta_{(i,j),p} & \forall (i,j) \in A \end{cases} \qquad \text{(I')}$$

IV.2.1. *Adaptation de la méthode de Frank-Wolfe*

Rappelons ici son principe (voir chapitre I). Comme toute méthode d'optimisation non linéaire, cette méthode comporte les deux étapes essentielles qui sont :

1. Le ***choix d'une direction de descente*** pour la fonction objectif;

2. Le ***choix du pas*** le long de cette direction de descente.

Pour le **choix de la direction**, on va déterminer celle-ci par:

$$d^k = y^k - x^k.$$

Où y^k minimise l'approximation linéaire de $z(x)$ en x^k sous les contraintes linéaires. Ensuite une recherche unidimensionnelle sera effectuée le long de la direction d^k.

Si on considère la première représentation à savoir :

1. *La formulation « arc sommet »*

On a donc y^k est solution du problème linéaire **PL(k)** suivant :

$$\min_{y} z(x^k) + \nabla z(x^k)(y - x^k)$$

$$\text{s.c.} \begin{cases} \sum_{i\in\Gamma^-(j)} y_{ij}^d - \sum_{k\in\Gamma^+(j)} y_{jk}^d = \begin{cases} -q_{jd} & \text{si } j \text{ est une origine} \\ 0 & \text{sinon} \end{cases} & \forall j = 1,...,n;\ \forall d = 1,...,r;\ j \neq d \\ y_{ij}^d \geq 0 & \forall (i,j) \in A\ \forall d = 1,...,r; \end{cases} \qquad \text{(II)}$$

Examinons plus en détails la fonction objectif du programme linéaire (II). Les constantes n'interviennent pas dans la minimisation, il est équivalent de minimiser:

$$\min \sum_{(i,j)\in \boldsymbol{A}} \sum_{d=1}^{r} \frac{\partial z(x^k)}{\partial x_{ij}^d} y_{ij}^d = \sum_{(i,j)\in \boldsymbol{A}} \sum_{d=1}^{r} t_{ij}^k y_{ij}^d$$

$$\text{s.c.} \begin{cases} \sum_{i\in\Gamma^-(j)} y_{ij}^d - \sum_{k\in\Gamma^+(j)} y_{jk}^d = \begin{cases} -q_{jd} & \text{si } j \text{ est une origine} \\ 0 & \text{sinon} \end{cases} & \forall j = 1,...,n;\ \forall d = 1,...,r;\ j \neq d \\ y_{ij}^d \geq 0 & \forall (i,j) \in \boldsymbol{A}\ \forall d = 1,...,r; \end{cases} \quad \text{(III)}$$

Avec $t_{ij}^k = t_{ij}(x_{ij}^k)$ Autrement dit, un problème linéaire en y dont les contraintes sont les contraintes du problème original (I), et les coefficients objectifs sont simplement les temps de traversées des arcs fixés en x_{ij}^k.

On peut également écrire ce programme sous la forme suivante:

$$\min_{y\geq 0} \sum_{(i,j)\in \boldsymbol{A}} \sum_{d=1}^{r} t_{ij}^k y_{ij}^d$$

$$\text{s.c.} \begin{cases} \sum_{i\in\Gamma^-(j)} y_{ij}^1 - \sum_{k\in\Gamma^+(j)} y_{jk}^1 = \begin{cases} -q_{j1} & \text{si } j \text{ est une origine} \\ 0 & \text{sinon} \end{cases} & j = 2,...,n. \\ \vdots & \\ \sum_{i\in\Gamma^-(j)} y_{ij}^d - \sum_{k\in\Gamma^+(j)} y_{jk}^d = \begin{cases} -q_{jd} & \text{si } j \text{ est une origine} \\ 0 & \text{sinon} \end{cases} & j = 1,...,n;\ j \neq d \\ \vdots & \\ \sum_{i\in\Gamma^-(j)} y_{ij}^r - \sum_{k\in\Gamma^+(j)} y_{jk}^r = \begin{cases} -q_{jr} & \text{si } j \text{ est une origine} \\ 0 & \text{sinon} \end{cases} & j = 1,...,n;\ j \neq r \\ y_{ij}^d \geq 0 & \forall (i,j) \in \boldsymbol{A}\ \ \forall d = 1,...,r. \end{cases} \quad \text{(IV)}$$

Remarquons que ce problème est décomposé en r blocs, dont la résolution de chaque bloc est indépendante des autres, et cela vient du fait que les variables diffèrent d'un bloc à un autre. Ainsi, résoudre le problème (IV) revient à résoudre r blocs de contraintes séparément, où chaque bloc correspond à un problème d'équilibre relatif à une destination d fixée :

$$\min_{y\geq 0} \sum_{(i,j)\in \boldsymbol{A}} t_{ij}^k y_{ij}^d$$

$$\text{s.c.} \begin{cases} \sum_{i\in\Gamma^-(j)} y_{ij}^d - \sum_{k\in\Gamma^+(j)} y_{jk}^d = \begin{cases} -q_{jd} & \text{si } j \text{ est une origine} \\ 0 & \text{sinon} \end{cases} & \forall j = 1,...,n;\ \ j \neq d \\ y_{ij}^d \geq 0 & \forall (i,j) \in \boldsymbol{A}; \end{cases} \quad \text{(V)}$$

Étudions maintenant plus en détail le programme (V) :

Ce dernier peut s'interpréter comme étant un problème de minimisation du coût total de transport dans un réseau à « *s* » origine et une seule destination « *d* » avec les t_{ij} comme coût de transport des arcs (i, j) et les q_{jd} étant les quantités à transporter de l'origine *j* à la destination *d*. Or, une telle affectation sera atteinte en affectant toute la quantité à transporter d'une paire *origine destination* au plus court chemin de la paire.

Ainsi, le noyau de ce programme est la recherche du plus court chemin dans un graphe, en considérant les t_{ij} comme longueurs des arcs.

Un arc peut appartenir à plusieurs plus courts chemins reliant les différentes origines à la destination *d*. Pour cela, et afin de calculer la solution optimale y^k du sous problème **PL(k)**, nous allons introduire une variable:

$$w_{ij}^{od} = \begin{cases} q_{od} & \text{Si l'arc } (i,j) \text{ appartient au plus court chemin reliant la paire } \textbf{\textit{od}} \\ 0 & \text{Si non} \end{cases}$$

$$\text{Avec } y_{ij}^{d} = \sum_{o=1}^{s} w_{ij}^{od} \ .$$

Ainsi, la direction de descente peut être calculée comme suit:

$$d^k = y^k - x^k$$

Remarque :

Dans le cas où plusieurs chemins sont minimaux, n'importe lequel peut être choisi.

Pour initialiser l'algorithme, on effectuera un chargement *tout ou rien* sur base des temps de traversées à vide: $t_{ij}^{0} = t_{ij}(0), \quad \forall (i,j) \in A$

Reste maintenant la seconde partie de la méthode de Frank-Wolfe, à savoir :

***La recherche unidimensionnelle*:** On va déterminer le pas α_k dans la direction d^k qui minimise: $z(x^k + \alpha(y^k - x^k))$

En limitant α à l'intervalle $[0,1]$ de manière à ne pas sortir de la région réalisable.

Le critère d'arrêt est définie par : $\nabla f(x^k)(y^k - x^k) > -\varepsilon$.....(*) (voir chapitre I).

En conséquence, l'algorithme peut être résumé comme suit:

Pas 0. *Initialisations*: Chargement «***tout ou rien***» du réseau sur base des plus courts chemins calculés avec les temps: $t_{ij}^0 = t_{ij}(0),\ \forall (i,j) \in \boldsymbol{A}$.

Ceci fournit x_{ij}^1. Soit $k = 1$, $d = 1$.

Pas 1. *Détermination de la direction de descente.*

Pour chaque sommet destination *d* fixé, faire :

- Déterminer le plus court chemin entre chaque sommet origine *o* du réseau et le sommet *d*; calculé avec les t_{ij}^k comme longueurs des arcs.
- Pour chaque sommet origine *o* du réseau, calculer y_{ij}^k tel que: $y_{ij}^k = y_{ij}^k + q_{od}$

$$\text{Poser } d_{ij}^k = y_{ij}^k - x_{ij}^k \quad \forall (i,j)$$

Pas 2. *Recherche unidimensionnelle*: Déterminer α_k qui minimise:

$$\min_{0 \le \alpha \le 1} f(x^k + \alpha d^k)$$

Pas 3. *Déterminer la nouvelle solution* x^{k+1} avec: $x^{k+1} = x^k + \alpha_k d^k$

Pas 4. *Test de convergence* : Si le critère (*) est satisfait, terminer.

Sinon, mettre $k = k + 1,\ y_{ij} = 0\ \forall (i,j), d = 1,$

Calculer $t_{ij} = t_{ij}(x_{ij}^k)$, retour au **pas 1.**

Fin d'algorithme.

2. *La formulation «arc chemin»*

Pour la détermination de ***la direction de recherche*** :

Après approximation linéaire de la fonction objectif, le problème (I′) est équivalent au programme linéaire suivant :

$$\min_{y \ge 0} z(x^k) + \nabla z(x^k)(y - x^k)$$

$$\text{s.c.} \begin{cases} \sum_{p \in C_{od}} g_p \delta'_{p,od} = q_{od} & \forall o = 1,...,s; \forall d = 1,...,r. \\ g_p \ge 0 & \forall p \in C_{od};\ o = 1,...,s;\ d = 1,...,r. \end{cases} \qquad \text{(VI)}$$

La relation entre les variables y_{ij} et les variables g_p étant donnée par:

$$y_{ij} = \sum_{o=1}^{s} \sum_{d=1}^{r} \sum_{p \in C_{od}} g_p \delta_{(i,j),p} \quad \forall (i,j) \in \boldsymbol{A}$$

Comme, les constantes n'interviennent pas dans la minimisation de la fonction objectif, le programme linéaire précèdent est équivalent à:

$$\min \sum_{(i,j)} \frac{\partial z(x^k)}{\partial x_{ij}} y_{ij} = \sum_{(i,j)} t_{ij}^k y_{ij}$$

$$\text{s.c.} \begin{cases} \sum_{p \in C_{od}} g_p \delta'_{p,od} = q_{od} & \forall o = 1,...,s; \forall d = 1,...,r. \\ g_p \geq 0 & \forall p \in C_{od};\ o = 1,...,s; \\ & d = 1,...,r. \end{cases} \qquad \text{(VII)}$$

$$\text{Avec } t_{ij}^k = t_{ij}(x_{ij}^k).$$

Ce programme cherche donc un flot *y* réalisable qui minimise le temps total passé sur le réseau, en supposant que les temps de traversées des arcs sont fixes (indépendants des flux). Or une telle affectation sera atteinte en affectant tous les usagers d'une paire origine destination au plus court chemin de la paire. Cette affectation pourra être réalisée par *un chargement tout ou rien du réseau* sur base du plus court chemin.

En effet, remplaçons dans l'expression de l'objectif les variables de flux totaux y_{ij} par celles de flux sur les chemins g_p :

On a $c_p = \sum_{(i,j)} t_{(i,j)} \delta_{(i,j),p}$ et $y_{ij} = \sum_p g_p \delta_{(i,j),p}$.

Donc $\sum_{(i,j)} \sum_p t_{ij}^k g_p \delta_{(i,j),p} = \sum_p \sum_{(i,j)} t_{ij}^k \delta_{(i,j),p} g_p = \sum_p c_p g_p$.

On obtient ainsi, le programme suivant:

$$\min \sum_{p \in C_{od}} c_p g_p$$

$$\text{s.c.} \begin{cases} \sum_{p \in C_{od}} g_p \delta'_{p,od} = q_{od} & \forall o = 1,...,s; \forall d = 1,...,r. \\ g_p \geq 0 & \forall p \in C_{od};\ o = 1,...,s; \\ & d = 1,...,r. \end{cases} \qquad \text{(VIII)}$$

Ce problème peut être décomposé en un programme par paire origine destination ***od***:

$$\min \sum_{p \in od} c_p g_p$$

$$\text{s.c.} \begin{cases} \sum_{p \in od} g_p = q_{od} \\ g_p \geq 0 \quad \forall p \in od \end{cases} \qquad \text{(IX)}$$

Dont la solution optimale est obtenue en affectant tout q_{od} au chemin *p* dont le temps de parcours c_p est minimum: il s'agit donc bien d'une affectation *tout ou rien*.

Ainsi, à chaque itération *k*, on détermine la direction de descente, par un chargement tout ou rien sur base de plus court chemin. Une fois les g_p obtenus, la direction peut être calculée comme suit :

$$y_{ij}^k = \sum_p g_p \delta_{(i,j),p}$$

$$d^k = y^k - x^k.$$

Pour ***la recherche unidimensionnelle***. On va déterminer le pas α_k dans la direction d^k qui minimise :

$$z(x^k + \alpha(y^k - x^k)) \text{ avec } \alpha \in [0,1]$$

Ainsi, l'algorithme de Frank-Wolfe adapté pour la résolution du problème d'affectation de trafic urbain, en considérant la formulation arc- chemin, peut être résumé comme suit :

Pas 0. *Initialisations*: Chargement tout ou rien du réseau sur base des plus courts chemins calculés avec les temps: $t_{ij}^0 = t_{ij}(0), \forall(i,j)$. Ceci fournit x_{ij}^1. Soit $k = 1$.

Pas 1. *Détermination de la direction*: pour chaque paire origine destination, déterminer le plus court chemin calculé avec les t_{ij}^k comme longueurs des arcs. Ceci fournira les flots y_{ij}^k.

Pas 2. *Recherche unidimensionnelle*: Déterminer α_k qui minimise:

$$\min_{0 \le \alpha \le 1} f(x^k + \alpha d^k)$$

Pas 3. *Effectuer le pas*: Mettre $x_{ij}^{k+1} = x_{ij}^k + \alpha_k (y_{ij}^k - x_{ij}^k)$

Pas 4. *Test de convergence*: Si le critère (*) est satisfait, stop.

Sinon, mettre $k = k + 1$ et retour au pas 1.

Fin d'algorithme.

IV.2.2 *Adaptation des versions modifiées de la méthode de Frank-Wolfe*

Dans les deux chapitres I et II, nous avons vue quelques versions modifiées de la méthode de Frank-Wolfe qui ont été développées dans le contexte d'amélioration de la vitesse de convergence de cette méthode. Dans ce qui suit nous allons nous intéressés à l'adaptation de ces versions modifiées au cas du problème d'affectation du trafic urbain. Les premières techniques que nous allons adaptées à savoir:"**PFW**, **CFW**, **BFW** et **IFW**" (voir chapitre I), seront présentées dans le cas de la formulation arc-chemin du problème de l'affectation du trafic urbain et les trois autres versions: "**FWλ**, **FWF** et la nouvelle variante **FWFλ**" (exposées au chapitre II), dans le cas de la modélisation arc-sommet du problème.

Tout d'abord on note par:

$x = (..., x_{ij}, ...)$ le vecteur des flux sur les arcs.
$t = (..., t_{ij}, ...)$ le vecteur des temps de traversées des arcs.
$y^k = (..., y_{ij}^k, ...)$ le vecteur solution du sous problème PL(k).
$d = (..., d_{ij}, ...)$ le vecteur direction de descente.

Ainsi, les versions modifiées de la méthode de Frank-Wolfe pour la résolution du problème de l'affectation du trafic urbain, peuvent être résumées comme suit:

1. La technique du PARTAN Frank-Wolfe (PFW)

Etape 0: *Initialisations* : Poser *k*=0, chargement « ***tout ou rien*** » du réseau sur base des plus courts chemins calculés avec les temps : $t_{ij}^0 = t_{ij}(0), \ \forall (i,j)$
Ceci fournit v^k avec $v^k = (..., v_{ij}^k, ...)$. Poser $x^{k+1} = v^k$.

Etape 1: *Choix de la direction de descente*: poser *k=k+1*, pour chaque paire origine destination, déterminer le plus court chemin calculé avec les t_{ij}^k comme longueurs des arcs. Ceci fournira les flots y_{ij}^k, poser $d_{ij}^k = y_{ij}^k - x_{ij}^k, \quad \forall (i,j)$

Etape 2: *Recherche du pas de descente* : trouver α_k, solution du problème unidimensionnel :

$$\min_{0 \le \alpha_k \le 1} f(x^k + \alpha_k d^k)$$

Etape 3: Poser $v^k = x^k + \alpha_k d^k$.

Etape 4: Si le *test de convergence* est vérifié, Stop

Sinon, Si $k < 2$ poser $x^{k+1} = v^k$. Aller à l'**Etape1**.

Sinon aller à l'**Etape 5**.

Etape 5 (*L'étape de PARTAN*) Calculer x^{k+1} par:

$$x^{k+1} = (1-\tau_k)v^k + \tau_k x^{k-1}.$$

Avec : τ_k : est solution du problème suivant $\min_{\tau_k^{\min} \le \tau_k \le 1} f(x^k)$

Où $\tau_k^{\min} \le 0$ représente la plus négative longueur de pas qui gardera x^k dans la région faisable.

2. La méthode de Frank-Wolfe conjuguée (CFW)

Etape0 (*Initialisation*) poser *k*=0, chargement «***tout ou rien***» du réseau sur base des plus courts chemins calculés avec les temps: $t_{ij}^0 = t_{ij}(0),\ \forall (i,j)$.

Ceci fournit la solution réalisable de départ x^0.

Etape 1 *Choix de la direction de descente de* **FW** *au point* x^k :

Pour chaque paire origine destination, déterminer le plus court chemin calculé avec les t_{ij}^k comme longueurs des arcs. Ceci fournira les flots y_{FW}^k, Poser $d_{FW}^k = y_{FW}^k - x^k$.

Etape 2 *Choix de la direction de descente de* **CFW**: Si *k*=0 poser $d_{CFW}^k = d_{FW}^k$

$$N_k = (S_{CFW}^{k-1} - x^k)^T H_k(x^k) d_{FW}^k,$$
$$D_k = (S_{CFW}^{k-1} - x^k)^T H_k(x^k)(y_{FW}^k - S_{CFW}^{k-1}),$$
$$\alpha_k = \begin{cases} \dfrac{N_k}{D_k}, & \text{Si } D_k \neq 0 \text{ et } \dfrac{N_k}{D_k} \in [0, 1-\delta] \\ 0, & \text{sinon} \end{cases}$$

Sinon calculer: $S_{CFW}^k = \alpha_k S_{CFW}^{k-1} + (1-\alpha_k) y_{FW}^k$

$$d_{CFW}^k = S_{CFW}^k - x^k$$

avec $S^k = (..., S_{ij}^k, ...)$

H : hessien de la fonction objectif.

Etape 3 *Recherche du pas de descente*: trouver γ_k, solution du problème unidimensionnel:

$$\min_{0 \le \gamma_k \le 1} f(x^k + \gamma_k d_{CFW}^k)$$

Etape 4 Poser $x^{k+1} = x^k + \gamma_k d_{CFW}^k$ et *k*=*k*+*1*.

Etape 5 Si le *test d'arrêt* est vérifié, Stop

Sinon, aller à l' **Etape1**.

3. La méthode de Frank-Wolfe Bi-conjuguée (BFW)

Etape0 (*Initialisation*) poser *k*=0, chargement «***tout ou rien***» du réseau sur base des plus courts chemins calculés avec les temps: $t_{ij}^0 = t_{ij}(0),\ \forall (i,j)$.

Ceci fournit la solution réalisable de départ x^0.

Etape 1 *Choix de la direction de descente de* **FW** *au point* x^k :

pour chaque paire origine destination, déterminer le plus court chemin calculé avec les t_{ij}^k comme longueurs des arcs. Ceci fournira les flots y_{FW}^k, Poser $d_{FW}^k = y_{FW}^k - x^k$.

Etape 2 *Choix de la direction de descente de* **BFW**:Si *k*=0 poser $d_{BFW}^k = d_{FW}^k$

Si *k*=1 poser $d_{BFW}^k = d_{CFW}^k$

$$S_{BFW}^k = \beta_k^0 y_{FW}^k + \beta_k^1 S_{BFW}^{k-1} + \beta_k^2 S_{BFW}^{k-2}$$

Sinon calculer: $d_{BFW}^k = S_{BFW}^k - x^k$

$$\beta_k^i \geq 0 \ \ \forall \mathrm{i} = \overline{0,2},\ \text{avec} \sum_{\mathrm{i}=0}^{2} \beta_k^i = 1$$

Etape 3 *Recherche du pas de descente* : trouver γ_k, solution du problème unidimensionnel :

$$\min_{0 \leq \gamma_k \leq 1} f(x^k + \gamma_k d_{BFW}^k)$$

Etape 4 Poser $x^{k+1} = x^k + \gamma_k d_{BFW}^k$ et *k*=*k*+*1*.

Etape 5 Si le *test d'arrêt* est vérifié, Stop

Sinon, aller à l' **Etape1**.

4. La technique du pas de descente non monotone (IFW)

Etape 0: Initialisations: Chargement «***tout ou rien***» du réseau sur base des plus courts chemins calculés avec les temps: $t_{ij}^0 = t_{ij}(0),\ \forall (i,j)$. Ceci fournit la solution réalisable de départ x^1. Soit $k = 1$.

Etape 1: *Détermination de la direction de recherche*: pour chaque paire origine destination, déterminer le plus court chemin calculé avec les t_{ij}^k comme longueurs des arcs. Ceci fournira les flots y_{ij}^k, poser $d_{ij}^k = y_{ij}^k - x_{ij}^k, \quad \forall (i,j)$.

Etape 2 : *La recherche du pas de descente* : trouver $\beta_k = \sigma_{h_k} a,$ avec:

h_k : est le premier entier *h* non négatif pour le quel :

$$f(x^k + \sigma_k a d^k) \le \max_{j=0,1,...M} \left\{ f(x^{k-j}) \right\} + \gamma \sigma_h a \nabla f(x^k)^T d^k$$

avec $a > 0,\ 0 < \sigma < 1, M$ un entier non négatif et $0 < \gamma < \frac{1}{2}$,

Etape3 : Poser $x^{k+1} = x^k + \beta_k d^k$.

Etape 4 : Si le *test d'arrêt* est vérifié, Stop

Sinon, poser *k=k+1*, aller à l' **Etape1**.

5. La technique du pas élargi (FWλ)

Pas 0. *Initialisations*: Chargement « ***tout ou rien*** » du réseau sur base des plus courts chemins calculés avec les temps : $t_{ij}^0 = t_{ij}(0), \forall (i, j)$. Ceci fournit la solution réalisable de départ x^1 .Soit $k = 1, d = 1$.

Pas 1. *Détermination de la direction de descente* :

Pour chaque sommet destination *d* fixé, faire :

- Déterminer le plus court chemin entre chaque sommet origine *o* du réseau et le sommet *d* ; calculé avec les t_{ij}^k comme longueurs des arcs.
- Pour chaque sommet origine *o* du réseau, calculer y_{ij}^k tel que: $y_{ij}^k = y_{ij}^k + q_{od}$

Poser $d_{ij}^k = y_{ij}^k - x_{ij}^k \ \ \forall (i, j).$

Pas 2. *Recherche unidimensionnelle*: Déterminer α_k qui minimise :

$$\min_{0 \le \alpha \le 1} f(x^k + \alpha d^k)$$

Si $k \in K^0$ aller au **pas 3.**

Si non aller au **Pas 4.**

Pas 3. *Détermination du pas élargi*:

Poser $\gamma_k = \lambda_k \alpha_k$; λ_k est le multiplicateur du pas de descente à l'itération *k*, avec $\lambda_k > 1$.

Poser $\beta_k = \min(\gamma_k, 1)$.

Soit $x^{k+1} = x^k + \beta_k d.^k$

Si $f(x^{k+1}) < f(x^k)$ poser $x^{k+1} = x^{k+1}$ aller au **Pas 5**.

Sinon aller au **pas 4.**

Pas 4. *Déterminer la nouvelle solution* x^{k+1} avec: $x^{k+1} = x^k + \alpha_k d^k$

Pas 5. *Test de convergence* : Si le critère (*) est satisfait, terminer.

Sinon, mettre $k = k + 1,\ y_{ij} = 0\ \ \forall (i,j),\ d = 1,$

Calculer $t_{ij} = t_{ij}(x_{ij}^k)$, retour au **pas 1.**

Fin d'algorithme.

6. La variante de Fukushima (FWF)

Pas 0. *Initialisations*: Chargement «***tout ou rien***» du réseau sur base des plus courts chemins calculés avec les temps : $t_{ij}^0 = t_{ij}(0), \quad \forall (i,j)$

Ceci fournit x^1. Soit $k = 1, d = 1$.

Pas 1. *Résolution du sous problème* **PL**(k) :

Pour chaque sommet destination *d* fixé, faire :

- Déterminer le plus court chemin entre chaque sommet origine *o* du réseau et le sommet *d* ; calculé avec les t_{ij}^k comme longueurs des arcs.
- Pour chaque sommet origine *o* du réseau, calculer y_{ij}^k tel que : $y_{ij}^k = y_{ij}^k + q_{od}$

Pas 2. Si $\nabla f(x^k)(y^k - x^k) = 0$ terminer, sinon aller au **Pas 3**.

Pas 3. *Détermination de la direction de descente* :

Soit μ_i; i = *k*-q…, *k* choisi tels que: $\sum_{i=k-q}^{k} \mu_i = 1, \quad \mu_i \geq 0; \ i = k - q, \ldots, k$

Poser: $v^k = (\sum_{i=k-q}^{k} \mu_i y^k) - x^k$, où q = min $\{k, \ell\}$-1,

Et $w^k = y^k - x^k$

$$\text{Avec } v^k = (\ldots, v_{ij}^k, \ldots) \ \& \ w^k = (\ldots, w_{ij}^k, \ldots).$$

Calculer les dérivées directionnelles:

$$\gamma_1^k = \nabla f(x^k) v^k / \left\| v^k \right\|$$
$$\gamma_2^k = \nabla f(x^k) w^k / \left\| w^k \right\|$$
$$d^k = \begin{cases} v^k & \text{si } \gamma_1^k < \gamma_2^k \\ w^k & \text{sinon} \end{cases}$$

Pas 4. *Recherche unidimensionnelle* : Déterminer α_k qui minimise :

$$\min_{0 \leq \alpha \leq 1} f(x^k + \alpha d^k)$$

Pas 5. *Déterminer la nouvelle solution* x^{k+1} avec : $x^{k+1} = x^k + \alpha_k d^k$

Pas 6. *Test de convergence* : Si le critère (*) est satisfait, terminer.

Sinon, mettre $k = k + 1, \ y_{ij} = 0 \ \forall (i,j), \ d = 1,$

Calculer $t_{ij} = t_{ij}(x_{ij}^k)$, retour au **pas 1.**

Fin d'algorithme.

7. La Variante combinée (FWFλ)

Pas 0. *Initialisations*: Chargement «***tout ou rien***» du réseau sur base des plus courts chemins calculés avec les temps : $t_{ij}^0 = t_{ij}(0),\ \forall(i,j)$

Ceci fournit x^1. Soit $k = 1$, $d = 1$.

Pas 1. *Résolution du sous problème* **PL**(k) :

Pour chaque sommet destination *d* fixé, faire :

- Déterminer le plus court chemin entre chaque sommet origine *o* du réseau et le sommet *d* ; calculé avec les t_{ij}^k comme longueurs des arcs.
- Pour chaque sommet origine *o* du réseau, calculer y_{ij}^k tel que : $y_{ij}^k = y_{ij}^k + q_{od}$

Pas 2. *Détermination de la direction de descente*:

Soit μ_i, $i = k\text{-}q\ldots, k$ choisi tels que: $\sum_{i=k-q}^{k} \mu_i = 1,\ \ \mu_i \geq 0;\ i = k-q,\ldots,k$

Poser: $v^k = (\sum_{i=k-q}^{k} \mu_i y^k) - x^k$, où $q = \min\{k, \ell\}$-1,

Et $w^k = y^k - x^k$

Calculer les dérivées directionnelles:

$$\gamma_1^k = \nabla f(x^k) v^k / \|v^k\|$$
$$\gamma_2^k = \nabla f(x^k) w^k / \|w^k\|$$
$$d^k = \begin{cases} v^k & \text{si } \gamma_1^k < \gamma_2^k \\ w^k & \text{sinon} \end{cases}$$

Pas 3. *Recherche unidimensionnelle* : Déterminer α_k qui minimise :

$$\min_{0 \leq \alpha \leq 1} f(x^k + \alpha d^k)$$

Si $k \in K^0$ aller au **pas 4.**

Si non aller au **Pas 5.**

Pas 4. *Détermination du pas élargi* :

Poser $\gamma_k = \lambda_k \alpha_k$; λ_k est le multiplicateur du pas de descente à l'itération *k*, avec $\lambda_k > 1$.

Poser $\beta_k = \min(\gamma_k, 1)$.

Soit $x^{k+1} = x^k + \beta_k d.^k$

Si $f(x^{k+1}) < f(x^k)$ poser $x^{k+1} = x^{k+1}$ aller au **Pas 6**.
Sinon aller au **pas 5.**

Pas 5. *Déterminer la nouvelle solution* x^{k+1} avec : $x^{k+1} = x^k + \alpha_k d^k$

Pas 6. *Test de convergence* : Si le critère (*) est satisfait, terminer.

Sinon, mettre $k = k + 1$, $y_{ij} = 0 \ \forall(i,j)$, $d = 1$,

Calculer $t_{ij} = t_{ij}(x_{ij}^k)$, retour au **pas 1.**

Fin d'algorithme.

IV.3. Conclusion

Ainsi, l'application de la méthode de Frank-Wolfe et de ses versions modifiées pour la résolution du problème de l'affectation du trafic urbain, nécessite une suite d'affectations tout ou rien du réseau. Ces affectations consistent à mettre tous les voyageurs d'une même paire origine destination sur le plus court chemin de cette paire. Le problème central consiste donc à déterminer le plus court chemin dans un réseau, dont il existe plusieurs algorithmes de résolution tel que l'algorithme de Dijkstra que nous avons choisi d'adapter (voir Annexe II).
Pour la recherche unidimensionnelle, c'est la méthode de la section dorée qui va être utilisée (voir Annexe II).

Chapitre V

Implémentation & Résultats

V.1. Introduction

Ce chapitre est consacré à l'étude des performances des versions modifiées de l'algorithme de Frank-Wolfe et en particulier de la nouvelle modification **FWFλ**, la présentation de divers résultats obtenus, et la comparaison entre les différentes variantes.

Nous avons implémenté l'algorithme de **FW** ainsi que les variantes **FWλ**, **FWF**, et **FWFλ** en langage de programmation C++ sous environnement C++ Builder5 ; l'algorithme de Dijkstra et la méthode de la section dorée, ont été utilisés respectivement pour la résolution des problèmes de plus court chemin et des problèmes unidimensionnels.

Etant donné, que le temps d'exécution requis par les tâches exécutables des opérations supplémentaires introduites par les différentes variantes présentées est pratiquement négligeable, la comparaison est essentiellement basée sur le nombre d'itérations. Pour les problèmes de grandes dimensions dont le temps d'exécution sur ordinateur est très crucial, la comparaison est aussi faite en terme d'unité de temps d'exécution afin de mesurer les gains en temps de calcul.

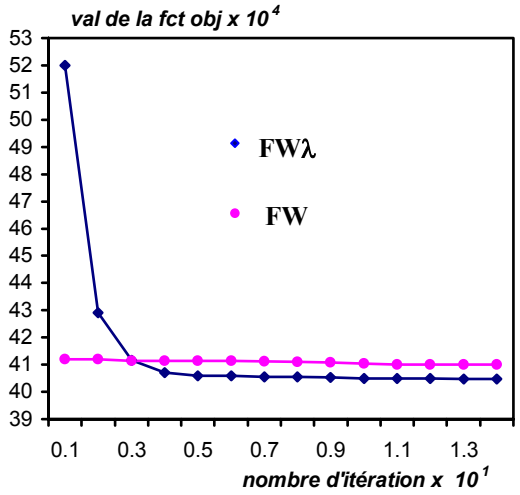
- La solution réalisable de départ notée x^1, est obtenue par la méthode de " ***tout ou rien*** " (mentionnée au chapitre IV).
- Les stratégies de modification du pas de descente ont été réalisées par des essais répétés sur un ensemble de problème de l'affectation du trafic urbain, les plus représentatives, sont les suivantes :
  - Pour $N \le 25 \quad \lambda_k = 1.6 \quad K^o = \{k \in K / k \le 5\}.$
  - Pour $N > 25 \begin{cases} \lambda_k = 1.4 \quad K^o = \{k \in K / k \le 5\}. \\ \lambda_k = 1.5 \quad K^o = \{k \in K / k \le 10\}. \end{cases}$
- Les coefficients μ_i (voir chapitre II) sont tous égales à 1/q+1. ($q = \min\{k, \ell\} - 1$)
- Le nombre ℓ (qui désigne le nombre maximum des y^i solutions des sous problèmes **PL**(i) utilisées pour la détermination de v^i "direction de descente de Fukushima") est égale à $|K^0|$.

- Le critère d'arrêt (voir chapitre I), est défini à partir de la variation de la fonction objectif entre deux itérations successives, le seuil inférieur de la variation est limité à $\varepsilon = 10^{-4}$:

$$\nabla f(x^k)(y^k - x^k) > -10^{-4}$$

V.2. Performance de la variante FWλ

Les expériences numériques effectuées ont montré une nette amélioration de la vitesse de convergence de la méthode; les meilleurs résultats de l'ordre de 40% de gain en moyenne sur le temps d'exécution comparés à la méthode de Frank-Wolfe classique, peuvent être obtenus particulièrement pour des solutions faisables initiales assez proches de l'optimum (voir la courbe V.1 qui représente un réseau de 37 arcs et 15 sommets).



Courbe* V.1 *: Réseau de 37 arcs et 15 sommets

V.3. Performance de la variante FWFλ

V.3.1. *Premier test*

Un premier test a été effectué sur un problème réel proposé par L. J. Leblanc [**14**], le problème considère le réseau du trafic urbain de la ville de Sioux Falls (Sud de Dakota ; USA), composé de 24 sommets et 76 arcs, chaque sommet du réseau est à la fois une origine et une destination. Dans ce test :

- Nous avons appliqué la stratégie de modification du pas de descente suivante :

$$\mathrm{K}^{o} = \{k \in \mathrm{K} / k \le 10\} \text{ et } \lambda_k = 1.5 \quad \forall\, k \in \mathrm{K}^{0}.$$

- La fonction objectif utilisée est : $f(x) = \sum_{(i,j)} A_{ij} \times x_{ij} + \frac{B_{ij}}{5} \times x_{ij}^{5}$.

  Avec: x_{ij} représente la valeur du flot total sur l'arc (i, j).

A_{ij} & B_{ij} les paramètres relatifs à l'arc (i, j) (voir tableau V.3).

- La formule des fonctions de performances des arcs est la suivante :

$$t_{ij} = t_{ij}(x_{ij}) = A_{ij} + (B_{ij} \times x_{ij}^{4}) \qquad \forall (i, j).$$

- Le réseau est représenté à la figure V.1:

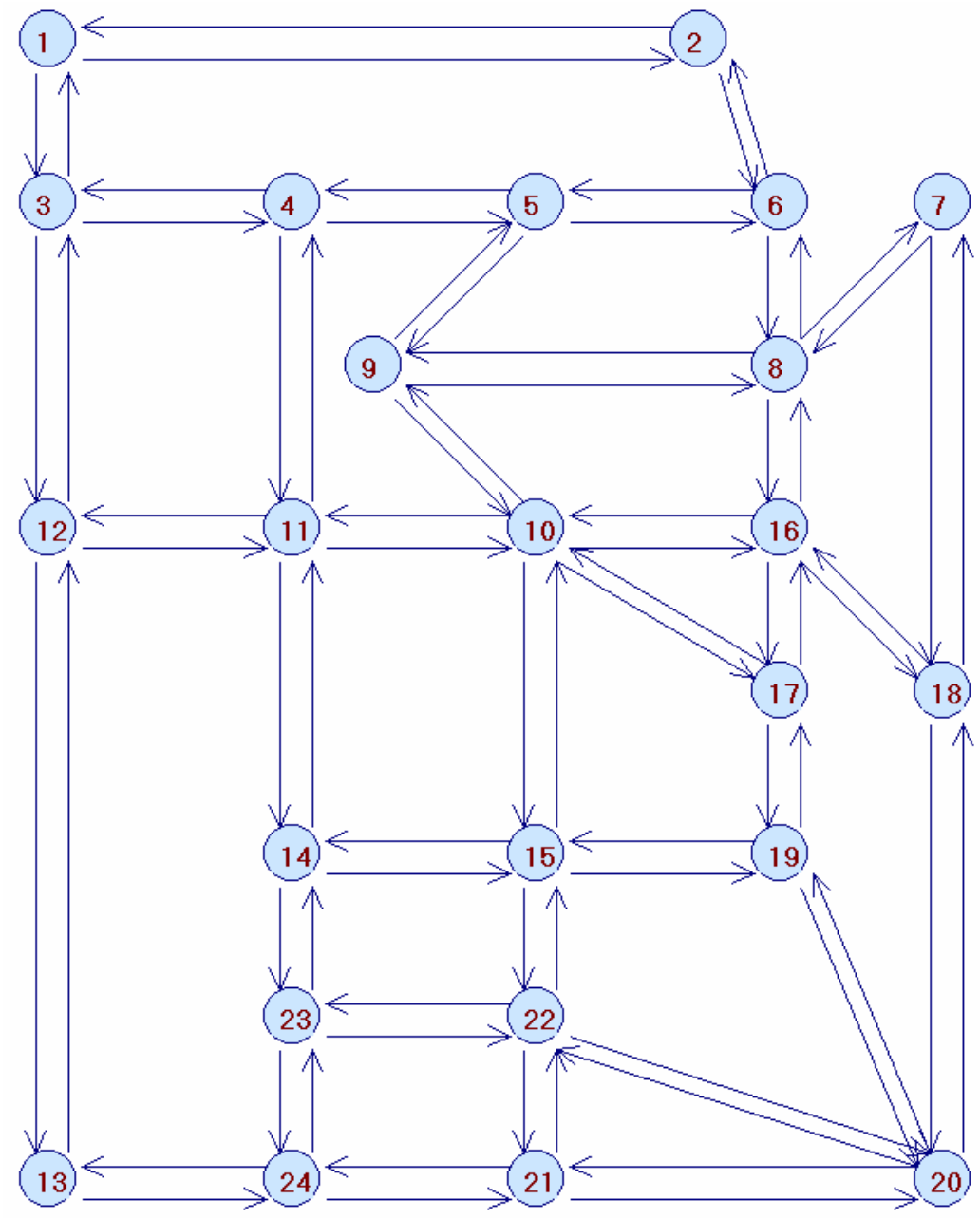


Figure V.1 : ***Le Réseau de Sioux Falls***

V.3.1.1. *Les données relatives au réseau de Sioux Falls*

a. La configuration du Réseau est donnée par la matrice d'adjacence **M** (illustrée par le tableau V.1), avec :

$$\mathbf{M}[i][j] = \begin{cases} 1 \text{ si l'arc } (i, j) \text{ existe} \\ 0 \text{ sinon} \end{cases}$$

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	S20	S21	S22	S23	S24
S1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S2	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S3	1	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
S4	0	0	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
S5	0	0	0	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S6	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S7	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
S8	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
S9	0	0	0	0	1	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S10	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	1	1	0	0	0	0	0	0	0
S11	0	0	0	1	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0
S12	0	0	1	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0
S13	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1
S14	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	0
S15	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	1	0	0
S16	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0
S17	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0
S18	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0
S19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	0	0	0
S20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	0	0
S21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1
S22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	1	0
S23	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	1
S24	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	1	0

Tableau V.1 : ***la matrice d'adjacence***

b. La demande totale entre toute paire origine destination du réseau, est représentée par le tableau V.2 suivant:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	0	1	1	5	2	3	5	8	5	13	5	2	5	3	5	5	4	1	3	3	1	4	3	1
2	1	0	1	2	1	4	2	4	2	6	2	1	3	1	1	4	2	0	1	1	0	1	0	0
3	1	1	0	2	1	3	1	2	1	3	3	2	1	1	1	2	1	0	0	0	0	1	1	0
4	5	2	2	0	5	4	4	7	7	12	14	6	6	5	5	8	5	1	2	3	2	4	5	2
5	2	1	1	5	0	2	2	5	8	10	5	2	2	1	2	5	2	0	1	1	1	2	1	0
6	3	4	3	4	2	0	4	8	4	8	4	2	2	1	2	9	5	1	2	3	1	2	1	1
7	5	2	1	4	2	4	0	10	6	19	5	7	4	2	5	14	10	2	4	5	2	5	2	1
8	8	4	2	7	5	8	10	0	8	16	8	6	6	4	6	22	14	3	7	9	4	5	3	2
9	5	2	1	7	8	4	6	8	0	28	14	6	6	6	9	14	9	2	4	6	3	7	5	2
10	13	6	3	12	10	8	19	16	28	0	40	20	19	21	40	44	39	7	18	25	12	26	18	8
11	5	2	3	15	5	4	5	8	14	39	0	14	10	16	14	14	10	1	4	6	4	11	13	6
12	2	1	2	6	2	2	7	6	6	20	14	0	13	7	7	7	6	2	3	4	3	7	7	5
13	5	3	1	6	2	2	4	6	6	19	10	13	0	6	7	6	5	1	3	6	6	13	8	8
14	3	1	1	5	1	1	2	4	6	21	16	7	6	0	13	7	7	1	3	5	4	12	11	4
15	5	1	1	5	2	2	5	6	10	40	14	7	7	13	0	12	15	2	8	11	8	26	10	4
16	5	4	2	8	5	9	14	22	14	44	14	7	6	7	12	0	28	5	13	16	6	12	5	3
17	4	2	1	5	2	5	10	14	9	39	10	6	5	7	15	28	0	6	17	17	6	17	6	3
18	1	0	0	1	0	1	2	3	2	7	2	2	1	1	2	5	6	0	3	4	1	3	1	0
19	3	1	0	2	1	2	4	7	4	18	4	3	3	3	8	13	7	3	0	12	4	12	3	1
20	3	1	0	3	1	3	5	9	6	25	6	5	6	5	11	16	17	4	12	0	12	24	7	4
21	1	0	0	2	1	1	2	4	3	12	4	3	6	4	8	6	6	1	4	12	0	18	7	5
22	4	1	1	4	2	2	5	5	7	26	11	7	13	12	26	12	17	3	12	24	18	0	21	11
23	3	0	1	5	1	1	2	3	5	18	13	7	8	11	10	5	6	1	3	7	7	21	0	7
24	1	0	0	2	0	1	1	2	2	8	6	5	7	4	4	3	3	0	1	4	5	11	7	0

Tableau V.2 : ***la matrice de la demande***

c. Les paramètres A_{ij} & B_{ij} relatifs aux différents arcs du réseau sont représentés dans le tableau V.3:

Arcs	Arcs	A(I)	B(I)	Arcs	Arcs	A(I)	B(I)
(1,2)	(2,1)	0.06	2E-8	(11,12)	(12,11)	0.06	1.55E-5
(1,3)	(3,1)	0.04	2E-8	(11,14)	(14,11)	0.04	1.061E-5
(2,6)	(6,2)	0.05	1.241E-5	(12,13)	(13,12)	0.03	1E-8
(3,4)	(4,3)	0.04	7E-8	(13,24)	(24,13)	0.04	8.93E-6
(3,12)	(12,3)	0.04	2E-8	(14,15)	(15,14)	0.05	1.085E-5
(4,5)	(5,4)	0.02	3E-8	(14,23)	(23,14)	0.04	1.02E-5
(4,11)	(11,4)	0.06	1.55E-5	(15,19)	(19,15)	0.03	1E-7
(5,6)	(6,5)	0.04	1.001E-5	(15,22)	(22,15)	0.03	5.3E-7
(5,9)	(9,5)	0.05	7.5E-7	(16,17)	(17,16)	0.02	4.01E-6
(6,8)	(8,6)	0.02	5.21E-6	(16,18)	(18,16)	0.03	3E-8
(7,8)	(8,7)	0.03	1.19E-6	(17,19)	(19,17)	0.02	5.54E-6
(7,18)	(18,7)	0.02	1E-8	(18,20)	(20,18)	0.04	2E-8
(8,9)	(9,8)	0.1	2.306E-5	(19,20)	(20,19)	0.04	9.58E-6
(8,16)	(16,8)	0.05	1.157E-5	(20,21)	(21,20)	0.06	1.373E-5
(9,10)	(10,9)	0.03	1.2E-7	(20,22)	(22,20)	0.05	1.13E-5
(10,11)	(11,10)	0.05	7.5E-7	(21,22)	(22,21)	0.02	4.01E-6
(10,15)	(15,10)	0.06	2.7E-7	(21,24)	(24,21)	0.03	7.9E-6
(10,16)	(16,10)	0.04	1.08E-5	(22,23)	(23,22)	0.04	9.6E-6
(10,17)	(17,10)	0.08	1.93E-5	(23,24)	(24,23)	0.02	4.51E-6

Tableau V.3 : ***Les paramètres relatifs aux différents arcs***

V.3.1.2. *Résultats obtenus*

Sur cet exemple (voir le tableau V.4), on peut constater que la variante **FWFλ** est nettement supérieure à la méthode de **FW** :

Du tableau V.4, on remarque que la variante du pas de descente n'a pas apporté une assez bonne amélioration puisque l'algorithme démarre avec une solution réalisable très loin de l'optimum.

En effet, durant les dix premières itérations (voir le tableau V.4), le résultat obtenu de la méthode de **FW** est meilleur que celui de la variante **FWFλ**. Cependant, l'efficacité de la variante **FWFλ** apparaît nettement lors de l'introduction de la direction de Fukushima.

itération	FW	FWFλ
0	158.670	158.670
1	63.438	77.249
2	56.059	66.472
3	53.106	58.391
4	50.457	54.654
5	48.999	52.453
6	47.372	50.598
7	46.469	47.809
8	45.106	46.909
9	44.531	46.318
10	44.210	45.094
11	43.891	44.744
12	43.748	44.239
13	43.420	43.967
14	43.286	43.587
15	43.196	43.373
16	43.097	43.196
17	43.009	42.983
18	42.953	42.570
19	42.843	42.482
20	42.793	42.448
21	42.743	42.425
22	42.708	42.408
23	42.667	42.393
24	42.642	42.361
25	42.610	42.338
26	42.590	42.322
27	42.568	42.316
28	42.555	
29	42.534	
30	42.518	
31	42.503	
32	42.491	
33	42.470	
34	42.461	
36	42.454	
37	42.449	
38	42.444	
39	42.440	
40	42.438	
41	42.433	
42	42.432	
43	42.430	
44	42.429	
45	42.425	
46	42.420	
47	42.420	
48	42.404	
49	42.398	
50	42.381	
51	42.380	
52	42.357	
53	42.334	
54	42.320	
55	42.317	

Tableau V.4 : ***Résultat obtenu***

À la lecture de ces résultats, on remarque que la variante **FWFλ** a atteint l'optimum au bout de vingt-sept itérations. Tandis que l'algorithme de **FW**, est allé jusqu'à cinquante-cinq itérations pour la même valeur de l'optimum.

V.3.2. *Second test*

Afin d'analyser le comportement de la variante **FWFλ**, sur des réseaux de grande taille, nous avons effectué une série d'expériences numériques sur des problèmes fictifs de l'affectation du trafic urbain.

La fonction objectif des problèmes tests est la fonction définie précédemment:

$$f(x) = \sum_{(i,j)} A_{ij} \times x_{ij} + \frac{B_{ij}}{5} \times x_{ij}^{5}$$

Les données sont générées aléatoirement à savoir : la configuration des réseaux, les demandes entre les origines destinations et les paramètres relatifs aux arcs: A_{ij} & B_{ij}.

D'après les résultats obtenus, nous avons constaté que l'avantage de la variante **FWFλ** devient en général plus significatif lorsque la taille du réseau augmente, et la précision demandée est plus fine, et pour des solutions réalisables de départ assez loin de l'optimum.

V.4. Comparaison entres les différentes méthodes

Dans ce cadre, nous avons résolu une série de problèmes fictifs de l'affectation du trafic urbain, par les différentes méthodes : **FW**, **FWλ**, **FWF**, et **FWFλ**. Les résultats les plus significatifs sont illustrés dans le tableau V.5 suivant:

(arcs, sommets)	Nom de la méthode	L'optimum	itération	Temps CPU (sec)	Choix de λ
(37,15)	FW	40335.25	114	0.485	K<5, λ=1.6
	FWλ	40341.15	50	0.221	
	FWF	40271.31	30	0.133	K<5, λ=1.6
	FWFλ	40276.78	14	0.065	
(66,23)	FW	1155411.00	104	0.960	K<5, λ=1.4
	FWλ	1149440.00	71	0.675	
	FWF	1148184.00	39	0.349	K<5, λ=1.4
	FWFλ	1143727.00	15	0.149	
(75,27)	FW	253959.18	135	1.558	K<5, λ=1.4
	FWλ	253684.16	26	0.315	
	FWF	252393.93	30	0.344	K<5, λ=1.4
	FWFλ	251750.00	14	0.176	
(80,30)	FW	1255272.00	149	2.335	K<5, λ=1.4
	FWλ	1245144.00	121	1.865	
	FWF	1241521.00	37	0.615	K<5, λ=1.4
	FWFλ	1232720.00	20	0.337	
(90,35)	FW	87389.00	149	3.024	K<10, λ=1.4
	FWλ	86174.37	20	0.388	
	FWF	85784.06	31	0.645	K<10, λ=1.4
	FWFλ	85488.75	15	0.346	

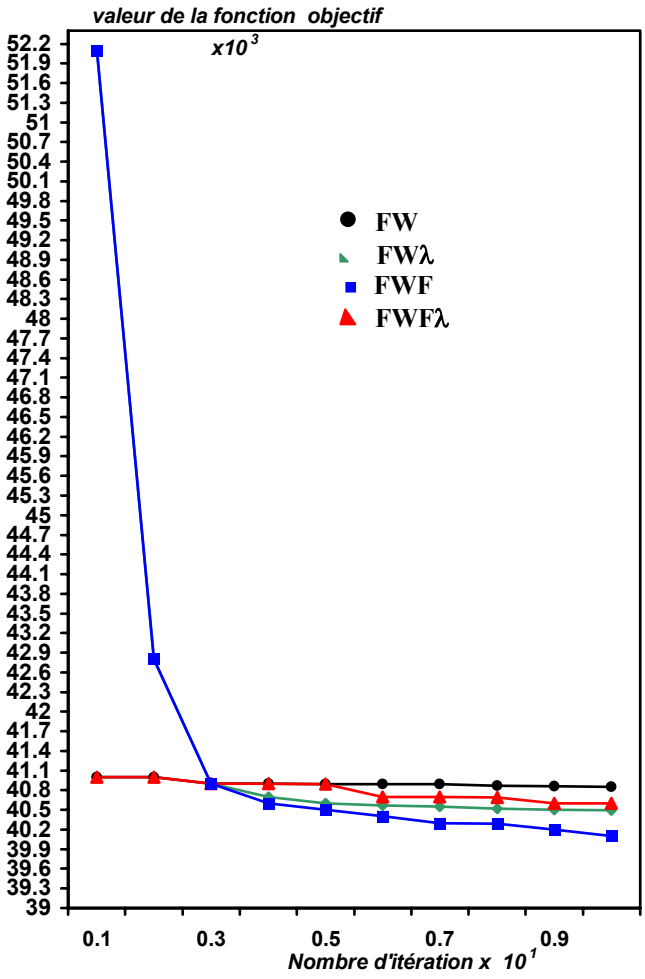
(110,85)	**FW**	**998809.94**	**91**	**2.458**	**K<5, λ=1.4**
	FWλ	**998201.00**	**60**	**1.459**	
	FWF	**992837.37**	**20**	**0.513**	**K<5, λ=1.4**
	FWFλ	**992837.37**	**18**	**0.458**	
(120,40)	**FW**	**204905.69**	**199**	**5.460**	**K<5, λ=1.4**
	FWλ	**203138.06**	**175**	**3.946**	
	FWF	**202362.12**	**75**	**2.011**	**K<5, λ=1.4**
	FWFλ	**200969.37**	**34**	**0.927**	
(330,128)	**FW**	**593610.39**	**251**	**8.032**	**k<10, λ=1.6**
	FWλ	**590869.97**	**234**	**7.997**	
	FWF	**587772.46**	**118**	**5.328**	**k<10, λ=1.6**
	FWFλ	**586931.82**	**67**	**3.151**	

***Tableau* V.5**

À la lecture des résultats obtenus, on constate que :

- La variante **FWFλ** donne une très bonne approximation de l'optimum.
- La supériorité de **FWFλ** est très remarquable. Elle réduit en moyenne à plus de 85% le nombre d'itérations exigé par la méthode de **FW**, et à plus de 55% de celui requis par la méthode **FWF**.
- Nous enregistrons également, une bonne performance des variantes **FWF** et **FWλ**.
- Dans les deux problèmes trois et cinq, on remarque que la technique **FWλ** est meilleure que la variante **FWF**.

La courbe V.2 représente l'effet d'amélioration porté par chaque variante, à travers l'évaluation de la fonction objectif, durant les premières itérations du premier problème (Réseau constitué de *37 arcs et 15 sommets*):



Courbe* V.2 : *Evolution de la fonction objectif

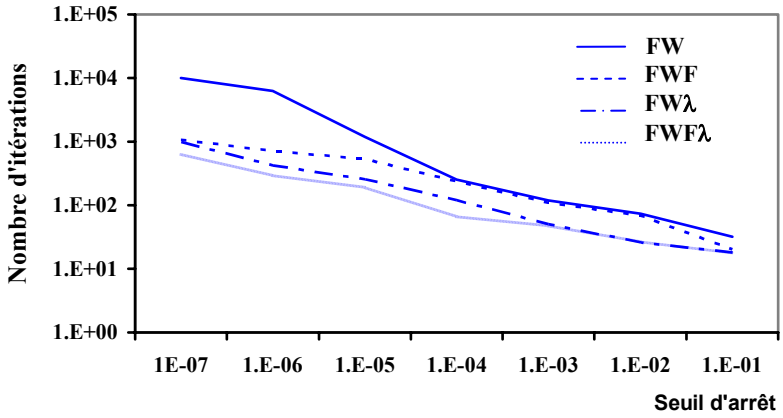
Variation du seuil d'arrêt

Dans ce cas, nous allons étudier l'évolution du nombres d'itérations requis par application des différentes variantes, par rapport à la variation du seuil d'arrêt " ε ".

- Un premier test à été effectué sur un réseau fictif, constitué de *330 arcs et 128 sommets*, le résultat obtenu est représenté dans le tableau V.6, la courbe V.3 illustre ces résultats:

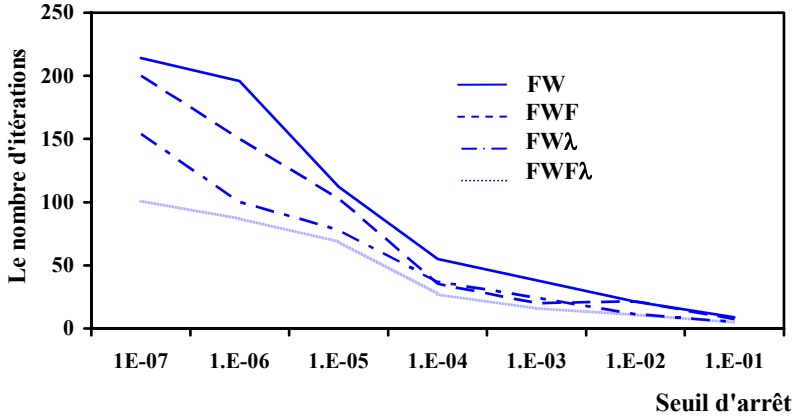
Le seuil d'arrêt	Le nombre d'itérations			
	FW	FWλ	FWF	FWFλ
10^{-1}	32	20	18	18
10^{-2}	74	68	27	26
10^{-3}	119	108	50	48
10^{-4}	251	234	118	67
10^{-5}	1198	525	253	194
10^{-6}	6233	702	419	297
10^{-7}	9976	1054	987	640

***Tableau* V.6**



***Courbe* V.3 *:* Comparaison entre FW, FWF, FWλ et FWFλ**

- La courbe V.4, représente l'effet de la variation du seuil d'arrêt, dans le cas du réseau réel de Sioux Falls (présenté plus haut) :



Courbe* V.4: Comparaison entre FW, FWF, FWλ et FWFλ dans le cas du Réseau de *Sioux Falls

Des courbes V.3 et V.4 :

On constate que la comparaison globale est nettement favorable à la variante **FWFλ**.

On remarque également que : Pour $\varepsilon > 10^{-4}$, le nombre d'itération requis par les deux variantes **FWFλ** et **FWF** est pratiquement le même. La performance de la nouvelle variante s'exprime nettement pour des valeurs de $\varepsilon \leq 10^{-4}$.

Il est à signaler que pour quelques valeurs du seuil d'arrêt, ($\varepsilon = 10^{-5}$ dans le premier test et $\varepsilon = 10^{-4}$ dans le second) on a constaté que la technique **FWλ** est meilleure que la variante **FWF**.

V.5. Conclusion

Il est montré que d'après les résultats obtenues, l'avantage de la variante **FWFλ** devient en général plus significatif, lorsque la taille du réseau augmente, et la précision demandée est plus fine.

Nous avons ici, du moins dans le cas de la méthode de Frank-Wolfe une confirmation de la thèse sur laquelle a été conçue la variante **FWFλ** et qui est : la combinaison de certaines techniques d'accélération de la vitesse de convergence, au sein d'un algorithme, engendre une amélioration additive, quitte à définir au préalable des bonnes conditions de compatibilités.

CONCLUSION GENERALE

Notre thème de recherche intitulé : « *Accroissement de la vitesse de Convergence de la Méthode de Frank-Wolfe, application aux réseaux de trafic urbain* » fût élaboré en deux étapes :

La première étant l'étude de la méthode de Frank-Wolfe. C'est le travail présenté dans les deux premiers chapitres de ce mémoire. Nous avons constaté, que la méthode de Frank-Wolfe jouit d'une grande importance, vue ses nombreux avantages et la diversité de ses applications. Toutefois, elle a pour inconvénient une vitesse de convergence très lente. De ce fait, des améliorations pratiques palliatives à cet inconvénient furent nécessaires. La synthèse des différents travaux effectués a montré que la méthode de Frank-Wolfe peut toujours être sujet à des remaniements, se transformant ainsi, en une méthode beaucoup plus efficace. Suivant ce principe, de nouvelles variantes ont été présentées dans ce mémoire, en l'occurrence: la technique du pas élargi « **FWλ** », la variante de Fukushima « **FWF** » et la variante combinée « **FWFλ** ».

Dans la seconde étape, la méthode de Frank-Wolfe et les trois versions modifiées ont été appliquées pour la résolution du problème de l'affectation du trafic urbain. Ces applications ont mis en évidence, d'un côté les différents avantages de la méthode de Frank-Wolfe. En particulier, l'utilisation des divers techniques de la recherche opérationnelle dans son procédé de résolution: la recherche du plus court chemin dans un graphe, la recherche unidimensionnelle, recherche arborescente…etc. Et d'un autre côté, l'efficacité des nouvelles variantes qui réside dans l'amélioration de la vitesse de convergence, comparées à la méthode de Frank-Wolfe classique. Favorisant beaucoup plus la variante **FWFλ** par rapport aux autres.

Bien que notre étude soit restreinte au problème de l'affectation du trafic urbain, l'application de la méthode de Frank-Wolfe et de ces versions modifiées, peut être étendue également pour la résolution d'une large gamme de problèmes de grandes dimensions, à titre d'exemple : le problème de l'affectation du trafic à demande élastique, le problème d'extension des réseaux du trafic urbain, ainsi que les problèmes de routage dans les réseaux de communication d'informations.

Bibliographie

[**1**] Y. Arezki,D.Van.Valiet.A Full Analytical implementation of the PARTAN/Frank-Wolfe algorithm for equilibrium assignement, *Transportation Sci*, 24(1):58-62, 1990.

[**2**] M. Beckmann, C, McGuire, C. Winsten, *Studies in the Economics of Transportation* , Yale University Press, New Haven, Conn., 1956.

[**3**] M. Daneva, P.O. Lindberg. The Stiff is Moving - Conjugate Direction Frank-Wolfe Methods with Applications to Traffic Assignment. Submitted. (2004).

[**4**] D. De Wolf, Recherche opérationnelle, Université du Littoral. Dunkerque, Sep, 2003.

[**5**] D. De Wolf, Transport et Environnement, Université du Littoral. Dunkerque, Sep, 2003.

[**6**] C. Duhamel, Ph.Mahey. Multicommodity flow problems with a bounded number of paths: a flow deviation approach. March 22, 2005.

[**7**] M. Florian. An improved linear approximation algorithm or the network equilibrium (packet switching) problem, in: Proccedings of the 1977 IEEE Conference onDecision Control, 1997, pp. 812-818.

[**8**] M. Frank and PH. Wolfe. An algorithm for quadratic programming. *Naval Res. Logist. Quart.* 3:95-110, 1956.

[**9**] L. Fratta, M. Gerla, and L. Kleinrock. The flow deviation method: An approach to store-and-forward communication network design. *Networks*, 3:97-133, 1973

[**10**] M. Fukushima. A modified Frank-Wolfe algorithm for solving the traffic assignment problem. *Transportation Res*. PartB, 18(2):169-177, 1984.

[**11**] L. Grippo, F. Lampariello, S. Lucidi, A non-monotone line search technique for Newton's method, SIAM J. Number. Anal. 23 (4) (1986) 707–716.

[**12**] C.C. Holloway. An Extension to the Frank-Wolfe method of feasible directions. Mathematical Programming. 6, 14-27, (1974).

[**13**] Holmgren, J., Hägglöf, K., Lindberg, P.O., (2004). A Comment on Some Test Networks for the Traffic Assignment Problem. Forthcoming.

[**14**] L.J. Leblanc. Mathematical *programming algorithms for large scale network equilibrium and network design problems*. PhD thesis, IE/MS Dept, Northwestern University, Evanston IL, 1973.

[**15**] M. Lupi. Convergence of the Frank-Wolfe algorithm in Transportation network. Civil Engeenring System, 3:7-15,1986.

[**16**] M. Minou, Programmation Mathématique, Théorie et algorithmes, Tome1; BORDAS et C.N.E.T.-E.N.S.T., Paris 1983.

[**17**] R. Ouafi. Analyse et contrôle des réseaux de trafic urbain par la méthode de Frank-Wolfe. Thèse de Doctorat 3ème cycle, Université Paris 6, 1988.

[**18**] R. Ouafi, S. Arrache. Improved Frank-Wolfe method: Application to the traffic assignement problem. Sur l'acte du colloque International sur l'optimisation et les systèmes d'information, *COSI'05*. Université de Béjaia, juin 2005.

[**19**] A.Ouorou, P.Mahey, and J.-Ph.Vial. Asurvey of algorithms for convex multicommodity flow problems. Management Sci., 46(1):126-147, 2000.

[**20**] M. Patriksson. *The traffic Assignment Problem – Models and Methods*. VSP, Utrecht, 1994.

[**21**] SHEFFI Yosef, Urban Transportation Networks: Equilibrium Analysis with Mathematical Programming Methods, Prentice-Hall, New Jersey, 1985.

[**22**] Wardrop, J.G., (1952) *Some* theoretical aspects of road traffic research.*Proceedings of the Institute of Civil Engineers 1*, Part II:325-378

[**23**] A. Weintraub, J. Gonzalez, An algorithm for the traffic assignment problem, Networks 10 (1980) 197–209.

[**24**] Willard I. Zangwill. *Nonlinear programming*: a unified approach. Prentice-Hall Inc., Englewood Cliffs, N.J., 1969.

[**25**] G. Zoutendijk. *Methods of Feasible Directions*. Elsevier, Amsterdam and D. Van Nostrand, Princeton, N.J.,1969.

[**26**] Ziyou Gao, W.H.K. Lam, S.C. Wrong, H. Yang. The convergence of equilibrium algorithms with non-monotone line search technique. Applied Mathematics and Computation, 148 pp1-13,2004.

ANNEXE I

Pour mieux comprendre le fonctionnement de l'algorithme de Frank-Wolfe, voyons deux itérations de son application sur l'exemple ci-après:

Exemple [4]

Soit le problème de maximisation suivant :

$$\begin{cases} \max f(x) = 5x_1 - x_1^2 + 8x_2 - 2x_2^2 \\ 3x_1 + 2x_2 \le 6 \\ x_1, x_2 \ge 0 \end{cases}$$

Nous partons du point initial $x^0 = (0, 0)$.

1ère itération

1. Evaluation du gradient en $x^0 = (0, 0)$:

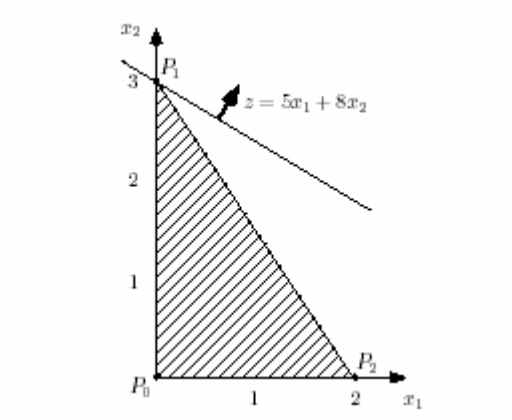
$$\frac{\partial f}{\partial x_1} = 5 - 2(0) = 5$$

$$\frac{\partial f}{\partial x_2} = 8 - 4(0) = 8$$

2. Résoudre le problème linéaire :

$$\text{PL(1)} \begin{cases} \max z = 5y_1 + 8y_2 \\ 3y_1 + 2y_2 \le 6 \\ y_1, y_2 \ge 0 \end{cases}$$

Par application de l'algorithme de simplexe, la solution optimale est $y^0 = (0, 3)$ (voir la figure suivante)



3. Recherche unidimensionnelle :

$$x^1 = x^0 + \alpha(y^0 - x^0)$$
$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} + \alpha \begin{pmatrix} 0-0 \\ 3-0 \end{pmatrix} = \begin{pmatrix} 0 \\ 3\alpha \end{pmatrix}$$

On obtient donc le problème de minimisation unidimensionnelle :

$$\max_{0 \le \alpha \le 1} h(\alpha) = 24\alpha - 18\alpha^2 .$$

Dont le maximum est obtenu en annulant la dérivée : $\alpha = 2/3$.

D'où $x^1 = (0, 0) + 2/3\ [(0, 3) - (0, 0)] = (0, 2)$

2ème itération

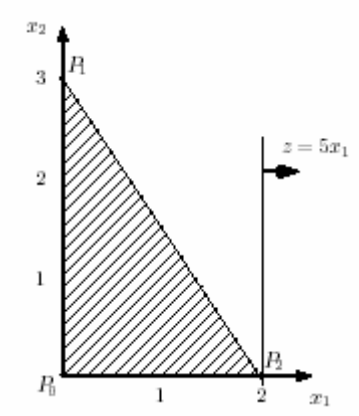
1. Evaluation du gradient en $x^0 = (0, 2)$:

$$\frac{\partial f}{\partial x_1} = 5 - 2(0) = 5$$
$$\frac{\partial f}{\partial x_2} = 8 - 4(2) = 0$$

2. Résoudre le problème linéaire :

$$\text{PL}(2)\begin{cases} \max z = 5y_1 \\ 3y_1 + 2y_2 \le 6 \\ y_1, y_2 \ge 0 \end{cases}$$

Par application de l'algorithme de simplexe, la solution optimale est $y^1 = (2, 0)$ (voir la figure suivante) :



3. Recherche unidimensionnelle :

$$x^2 = x^1 + \alpha(y^1 - x^1)$$

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 2 \end{pmatrix} + \alpha \begin{pmatrix} 2-0 \\ 0-2 \end{pmatrix} = \begin{pmatrix} 2\alpha \\ 2-2\alpha \end{pmatrix}$$

On obtient donc le problème de minimisation unidimensionnelle :

$$\max_{0 \le \alpha \le 1} h(\alpha) = 8 + 10\alpha - 12\alpha^2$$

qui est maximum pour : $\alpha = 5/12$.

D'où x^2 = (0, 2) + 5/12 [(2, 0) - (0, 2)] = (5/6, 7/6).

La solution optimale du problème est $x^* = (0.98, 1.48)$.

La valeur de la fonction objectif à l'optimum est $f(x^*) = 11.427$.

Cette solution est obtenue après 70 itérations, pour une valeur du seuil d'arrêt ε fixée à 10^{-3}.

ANNEXE II

I. Le problème du plus cours chemin

Le problème du plus court chemin peut se formuler de la manière suivante :

Etant donné un graphe G= (N, A) où à chaque arc est associée une longueur $t_{ij} \geq 0$,on veut déterminer un chemin allant du sommet origine ***o*** au sommet destination ***d*** tel que la longueur du chemin parcouru soit minimum. Les arcs du graphe représentent, les routes liant les différentes destinations et les longueurs des arcs représentent les temps de parcours.

Il existe cependant un algorithme spécialisé pour résoudre ce problème classique: il s'agit de l'algorithme de ***Dijkstra*** qui procède par marquages successifs des sommets.

Il consiste à attribuer un label à chaque sommet et à corriger progressivement ces labels. On va, à partir du sommet origine, progressivement déterminer le plus court chemin vers tous les autres sommets du réseau.

Les données du problème sont les suivantes :

- le réseau donné sous forme d'un graphe $G = (N, A)$;
- le temps de traversée t_{ij} de chaque arc $(i, j) \in A$.

Pour les besoins de l'algorithme, deux informations vont être stockées en chaque sommet :

1. Le label courant du sommet, noté d_i, qui indique la distance entre le sommet origine et le sommet i par le plus court chemin trouvé jusqu'à présent;

2. Le prédécesseur courant du sommet, noté p_i, qui note le sommet juste avant i dans le plus court chemin trouvé jusqu'à présent entre ***o*** et i. Ainsi le plus court chemin pourra être retracé en allant à rebours.

De plus, l'algorithme utilise et met à jour deux ensembles :

a. L'ensemble T des sommets dont on a déterminé exactement le plus court chemin entre le sommet origine et ce sommet. C'est l'ensemble des sommets déjà traités et qui ne nécessitent donc plus de traitement ultérieur.

b. L'ensemble $S = N \setminus T$ des sommets qui sont encore à traiter.

Dans l'algorithme, on dit qu'un sommet du premier ensemble a un label définitif, puisqu'il ne sera plus modifié, tandis qu'un sommet du second ensemble a un label provisoire qui pourra encore éventuellement être amélioré.

I.1 Algorithme de Dijkstra

L'initialisation, consiste à mettre tous les labels à ∞, sauf celui du sommet origine ***o*** qui sera mis à zéro :

$$d_i = \begin{cases} +\infty \text{ si } i \neq o \\ 0 \quad \text{sinon} \end{cases}$$

On initialise l'ensemble des sommets qui sont encore à traiter $S = N$.

A chaque itération, on sélectionne l'élément $i \in S$ de label provisoire d_i minimum. Soit :

$$i \in \mathrm{S} / d_i = \min_{i' \in \mathrm{S}} d_i$$

Pour tous les sommets $j \in S$ qui peuvent être atteints directement à partir de i par un seul arc (i, j), on examine si le chemin menant à j en passant par i ne serait pas plus court que l'ancien chemin menant à j. Dans ce cas, d_j et P_j sont remis jour comme suit :

$$\text{Si } d_i + t_{ij} < d_j \text{ alors } d_j = d_i + t_{ij}$$
$$\text{et } P_j = i$$

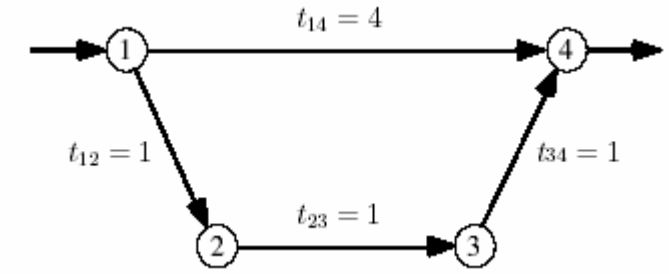
Enfin, i le sommet traité est déplacé de l'ensemble S vers l'ensemble T.

L'algorithme s'arrête, dès que l'on a déterminé le label définitif de ***d***, le sommet de destination, autrement dit dès que $\boldsymbol{d} \in T$.

Complexité : La complexité de l'algorithme de Dijkstra est $O(|N|^2 + |A|) = O(|N|^2)$

I.2 Exemple d'application

Appliquons l'algorithme de ***Dijkstra*** sur l'exemple illustré par la figure suivante où l'on cherche le plus court chemin entre le sommet 1 et le sommet 4.



La suite des itérations est donnée au tableau ci-dessous :

k	i	d_1	d_2	d_3	d_4	p_1	p_2	p_3	p_4
0		0	∞	∞	∞	–	–	–	–
1	1	0	1	∞	4	–	1	–	1
2	2	0	1	2	4	–	1	2	1
3	3	0	1	2	3	–	1	2	3
4	4	0	1	2	3	–	1	2	3

A la première itération, le sommet 1 est traité. Cela permet de réduire les labels des sommets 2 et 4 qui étaient à plus l'infini. A la deuxième itération, le sommet 2 est traité. Cela permet de réviser le label du sommet 3. A la troisième itération, le sommet 3 est traité, cela permet de réduire le label du sommet 4. Enfin, à la quatrième itération, le sommet 4 est traité et son label devient définitif. On a trouvé le plus court chemin entre 1 et 4. Il est de longueur trois.

On peut alors reconstituer le chemin le plus court en partant de la fin et en reprenant les prédécesseurs successifs on obtient le chemin 4, 3, 2 1.

II. Recherche linéaire en dimension 1 (Section dorée)

Étant donné une solution $x^k \in \mathrm{IR}^n$ et une direction $d^k \in \mathrm{IR}^n$ dans laquelle la fonction f décroît, nous étudierons dans cette section la question de trouver un scalaire $\alpha_k > 0$ qui minimise la valeur de $f(x^k + \alpha_k d^k)$. Pour ce faire, nous définissons une fonction d'une seule variable $h : \mathrm{IR} \to \mathrm{IR}$ comme suit :

$$h(\alpha) = f(x^k + \alpha_k d^k)$$

On observe que puisque d^k est une direction de descente pour la fonction f alors $h'(0) < 0$.

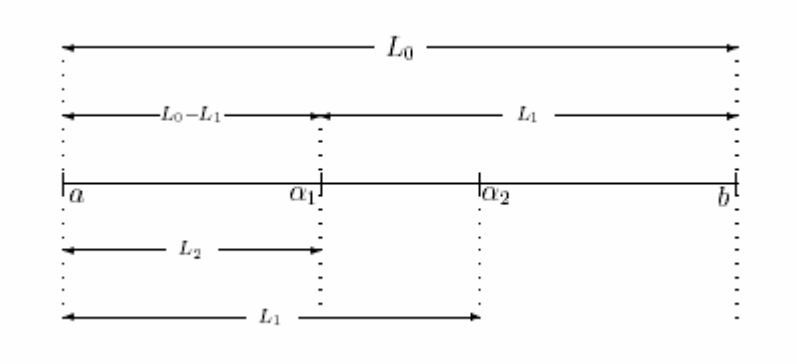
La recherche linéaire dans la direction d^k consiste à déterminer une petite valeur de $\alpha_k > 0$ telle que $h'(\alpha) = 0$.

S'il est possible d'évaluer la dérivée $h'(\alpha) = \nabla f(x^k + \alpha d^k) d^k$ nous pouvons utiliser une méthode permettant de trouver une racine d'une fonction, telle la méthode de Newton, la bissection, les points fixes par exemple, ou même résoudre de façon analytique. Sinon, on doit utiliser une méthode d'élimination de région comme la méthode de la section dorée, que nous avons choisi d'utiliser dans notre étude pour la détermination du pas de descente (voir chapitre VI).

II.1 Principe de la méthode de la Section Dorée

Considérons le problème :

$$\min_{\alpha} \quad h(\alpha)$$
$$s.c \quad \alpha \in [\mathrm{a}, \mathrm{b}]$$



-Figure1 : Méthode de la Section Dorée-

Où $h : \mathrm{IR} \rightarrow \mathrm{IR}$ est une fonction unimodale, i.e., si $\alpha_1 \leq \alpha_2$ et

$$\text{si } h(\alpha_1) \leq \alpha_2 \text{ alors } h(\alpha) \geq \alpha_2 \quad \forall \alpha \leq \alpha_2$$
$$\text{si } h(\alpha_1) \geq \alpha_2 \text{ alors } h(\alpha) \geq \alpha_1 \quad \forall \alpha \leq \alpha_1$$

Étant donné l'intervalle [a,b], on cherche à déterminer des valeurs α_1 et α_2 tels que $\mathrm{a} < \alpha_1 < \alpha_1 < \mathrm{b}$. On évaluera ensuite la fonction en ces points, ce qui nous permettra de réduire notre intervalle de recherche.

$$\text{si } h(\alpha_1) \leq \alpha_2 \text{ alors il existe un minimum dans } [\mathrm{a}, \alpha_2]$$
$$\text{si } h(\alpha_1) \geq \alpha_2 \text{ alors il existe un minimum dans } [\alpha_1, \mathrm{b}].$$

Pour se prémunir contre le pire cas, on veut que les intervalles $[\mathrm{a}, \alpha_2]$ et $[\alpha_1, \mathrm{b}]$ soient de même longueur. De plus, on aimerait réutiliser les points où la fonction a déjà été évaluée. On doit alors choisir α_1 et α_2 de façon à ce que les proportions soient conservées. La figure 1 nous permet de déterminer ces valeurs.

Soit $\tau = \dfrac{L_0}{L_1} = \dfrac{L_1}{L_2} = \ldots.$ la constante du rapport d'un intervalle à un autre. Il s'ensuit que :

$$L_2 = L_0 - L_1$$
$$\tau L_1 = L_0 - L_1$$
$$(\tau + 1) L_1 = L_0$$
$$(\tau + 1)\tau L_0 = L_0$$
$$\tau^2 + \tau - 1 = 0 \quad \Leftrightarrow \tau = \frac{-1 \pm \sqrt{5}}{2}$$

II.2 Exemple d'application

Le deux premières itérations de la méthode de la section dorée sur le problème $\begin{pmatrix} \min\limits_{\alpha} & \alpha^2 + 2\alpha \\ s.c & \alpha \in [-3,5] \end{pmatrix}$ donnent :

$L_0 = 8, \quad L_1 = \tau L_0 \approx 4.944, \quad \alpha_1 = 5 - L_1 = 0.056, \quad \alpha_2 = -3 + L_1 \approx 1.944.$

$h(-3) = 3, \quad h(0.056) \approx 0.115, \quad h(1.944) \approx 7.667, \quad h(5) = 35.$

Le nouvel intervalle est alors : [-3, 1.944].

$L_1 \approx 4.944, \quad L_2 = \tau L_1 \approx 3.0557, \quad \alpha_1 = 1.944 - L_2 = -1.112, \quad \alpha_2 = -3 + L_2 \approx 0.056.$

$h(-3) = 3, \quad h(-1.112) \approx -0.987, \quad h(0.056) \approx 0.115, \quad h(1.944) \approx 7.667.$

Le nouvel intervalle est alors : [-3, 0.056].

Notre approximation de la solution optimale est $\hat{\alpha} = \frac{-3+0.056}{2} = -1.472$ avec une erreur relative inférieur à $\Delta\alpha = \frac{L_2}{2} \approx 1.528.$

Résumé

Le but de ce travail est l'étude de la méthode de Frank-Wolfe, cette dernière à fait son apparition en programmation quadratique, aussitôt elle s'est révélée d'une façon remarquable, efficace pour la résolution des problèmes de flot de grandes dimensions et en particulier des problèmes de l'affectation du trafic urbain ; elle est très appréciée pour ses nombreux avantages; Néanmoins elle souffre d'une vitesse de convergence très lente, ce qui constitue le point faible et le problème majeur de cette méthode.

De ce fait, et compte tenu de l'importance du domaine d'application couvert par la méthode de Frank-Wolfe, beaucoup de tentatives ont été élaborées dans le but de remédier à cet inconvénient, depuis les premier travaux de "L. J. Leblanc" jusqu'au récemment les travaux de "Ziyou Gao & Al".

De notre part, pour atteindre cet objectif, nous avons proposé quelques variantes développées dans ce contexte ; des tests numériques ont été réalisés à cet effet, sur des problèmes de l'affectation du trafic urbain, illustrant l'efficacité de ces nouvelles modifications comparées à la méthode de Frank-Wolfe classique.