الجمهورية الجزائرية الديمقراطية الشعبية
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

13/89

وزارة التعليم العالي
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR

ÉCOLE NATIONALE POLYTECHNIQUE

DEPARTEMENT : electronique

PROJET DE FIN D'ETUDES

SUJET

CONTROLE OPTIMAL D'UN SYSTEME LINEAIRE AVEC DES CONTRAINTES LINEAIRES

Proposé par : mr: f.chigara

Etudié par : k.sid
m.taghi

Dirigé par : f.chigara

PROMOTION : jun 89

E.N.P. 10 Avenue Hacen Badi – EL-HARRACH – ALGER

DEDICACES

A MA MERE ET A MON PERE ,

A MES FRERES ET MES SOEURS ,

A MES GRANDS MERES ET MES GRANDS PERES

A TOUS CEUX QUI ME SONT CHERS ,

ET SURTOUT , A TOUS MES MEILLEURS AMIS

KAMEL

REMERCIEMENTS

Nos plus vifs remerciments à messieurs F.CHIGARA. S.AIT CHIKH, S.SALHI pour leurs precieux conseils.

Nous remercions egalement Monsieur SOFIANE ainsi que Monsieur MOULOUDE pour leurs aide materielle

SOMMAIRE

INTRODUCTION GENERALE

Comme les ressources non renouvelables de la planète commencent à diminuer,un interet grandissant se développe pour assurer une meilleure performance des processus chimiques,economiques ,industriels ou socials.

Cette performance dépend d'un large nombre de decisions (contraintes physiques,milieu environnant) et,pour parvenir a la maitrise optimale des activites de ces systemes,la theorie du controle optimal permet,grace au developpement de l'informatique,de choisir les meilleures decisions ou les meilleures structures de commande.

Notre travail consiste particulierement à étudier les differents algorithmes de controle et d'optimisation des systèmes linéaires.

Cette présente etude est separée en trois parties :

- La 1ere ou l'on expose les differentes definitions et objectifs de la theorie du controle optimal

- Dans la 2ème on présente la methode de programmation linéaire consistant en plusieures methodes (simplexe , simplexe revisée , Dantzing-Wolfe) permettant le controle des systemes linéaires statiques

- La 3ème est consacrée à la resolution des systemes dynamiques par les techniques du controle hierarchique.

Et nous terminons par une conclusion generale.

PREMIERE PARTIE

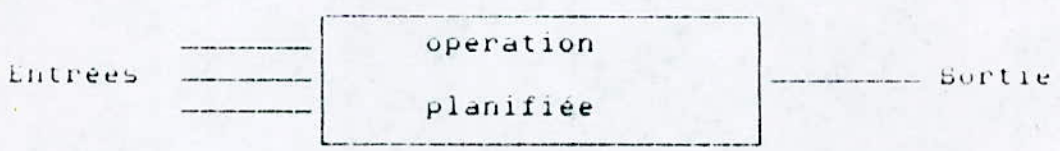
GENERALITES

I/ NOTION DE SYSTEME

Un systeme est un ensemble d'elements dont on a definie les inter-relations de type physique ou économique et surtout les finalités.Ces elements sont situés par rapport à une frontière qui constitue la frontiere du systeme.

Le systeme a des objectifs,un environnement,il peut etre concret ou abstrait,ouvert ou ferme,statique ou dynamique. Tout système possede en général plusieures entrees et plusieures sorties.Les entrées peuvent être des commandes ou des perturbations,donc pour pouvoir agir sur un systeme il est nécéssaire de bien connaitre son environnement et de bien le définir.

Le systeme le plus élementaire avec ses entrees,son operation planifiée et sa sortie peut être representé comme suit:



La notion de systeme est extremement générale,la relation par laquelle les entrées déterminent les sorties est l'expression des lois physiques (économiques ou autres)qui regissent le fonctionnement.

II/ NOTION DE SYSTEME LINEAIRE

Nous avons vu que les relations entrees-sorties du systeme sont l'expression des lois physiques régissant son fonctionnement,un cas particulièr extremement important en pratique est celui ou ces lois s'expriment par des equations (differentielles ou aux differences)lineaires,on parle alors de systèmes lineaires .
L'importance technique de l'hypothese de linéarite provient de l'existance,pour l'étude et la synthese des systemes lineaires,de methodes (analytiques,graphiques ou expérimentales)simples en leurs principe et extremement puissantes.

Par bonheur,un grand nombre de systemes de commande sont lineaires ou du moins peuvent etre consideres comme tels dans une large gamme de conditions d'emplois, cela explique l'importance dominante des methodes lineaires en matière d'automatisme.

Du point de vue physique

a/ de la linéarité des équations résulte le principe d'additivité ou de superposition:la réponse d'un système linéaire à plusieures entrées est la somme de ce que seraient ses réponses à chacune d'elles (les causes ajoutent leurs effets).
b/ du fait que les variables d'entrée et de sortie sont ordinairement évaluées à partir de leurs valeurs à l'équilibre résulte le principe d'homogeneité:
si les entrées sont multipliées,les sorties se trouvent multipliées par le même facteur (proportionnalité des effets aux causes).
c/ dans le cas d'un système invariant (statique),de la constance des coefficients des équations résulte le principe d'invariance temporelle:
si l'action de certaines entrées est simplement decalée dans le temps, il en va de même de la réponse correspondante.

III/ NOTION DE GRAND SYSTEME

Les problèmes des systèmes complexes (systèmes d'ordre elevé comprenant des sous systèmes interconnectés)posent des difficultés énormes du point de vue analyse et de contrôle; de tels systèmes sont rencontrés dans l'industrie et dans le domaine socio-économique .

Bien qu'il soit possible d'entamer directement la phase d'analyse des systèmes d'ordre réduit (définition des entrées-sorties,contrôle,construction du modèle,estimation des paramètres,définition du critère etc...)ainsi que la phase contrôle (synthèse et implementation des algorithmes),ceci n'est en général pas possible pour les systèmes d'ordre élevé; les difficultés peuvent être d'ordre théorique (mauvaise convergence ou divergence de l'algorithme)ou de consideration économique.

Le traitement complexe exige:

a/ de nouvelles méthodes analytiques étant donnée que la dimensionnabilité est élevée.
b/ la nécessité d'obtenir un état intermédiaire avant le contrôle;cet état doit consister en:
- soit la réduction de la dimension du problème par une procédure d'agrégation.
- soit la décomposition du système en définissant des sous systèmes adéquats,dans le but d'utiliser les méthodes de décomposition-coordination.
c/ la synthèse des algorithmes de contrôle utilisant les principes de décomposition-coordination.

IV/ NOTIONS GENERALES DE CONTROLE DES PROCESSUS

1- PROBLEME DU CONTROLE

1-1- DEFINITIONS :

Comme nous l'avons mentionné auparavant, il existe parmi les entrées d'un système des variables de commande et de contrôle qui nous permettent d'agir sur les sorties d'un processus dans le sens de modifier leurs comportements.
- L'évolution de ces commandes dans le temps est représentée par des trajectoires dites de commande.
- La période d'évolution est appelée horizon, elle peut être imposée (fixe) ou infinie.
- En général, les commandes et les sorties n'évoluent pas de manière arbitraire mais doivent satisfaire certaines conditions relatives à l'environnement et limitations physiques, ces conditions sont appelées contraintes.

Les commandes et les sorties satisfaisant ces contraintes sont qualifiées d'admissibles ou réalisables.
- Les dites contraintes se présentent souvent sous forme d'inégalités de type $(h(x,y)\leq 0)$ et quelques fois sous forme d'égalités $(h(x,y)=0)$.

1-2- NOTION D'OPTIMISATION

Ayant défini les différentes variables et contraintes imposées au système, on constate alors qu'il existe un grand nombre de commandes admissibles permettant d'atteindre l'objectif du processus.

Le choix d'une trajectoire de commande parmi l'espace des commandes admissibles conduit à la notion d'optimisation.
Il est alors nécessaire de définir un indice de performance (ou une fonction critère, économique, objectif ou coût) pour faire le choix évoqué ci-dessus.
La formulation de cet indice est un travail assez compliqué pour les raisons suivantes: [10] , [11]
- La liberté de choix qui existe souvent.
- Le choix qui influence un grand nombre de parmetres.

La trajectoire de commande optimisant le critère de performance est satisfaisant les contraintes est qualifiée d'optimale.
Notons enfin que la fonction critère est très souvent une fonction scalaire.
Nous donnons dans la table (1) les différentes formes mathématiques de fonction objectif.

Processus lineaire statique:

$J1 = \sum_i Ci.Xi$ (J1 est une fonction lineaire des variables Xi, Ci: constante)

$J2 = \sum_i Ci.Xi^2$ (forme quadratique)

Processus lineaire dynamique:

$J3 = \int_0^T (x-x)\, Q\, (x-x)\, dt$ (Pb à erreur minimale, x: etat de reference)

$J4 = G(x,u,t)\Big|_{t0}^{tf} + \int_0^T L(x,u,t)dt$ (forme generale des Pbs dynamiques, L peut etre lineaire ou quadratique)

Table (1)

2 - NOTION DE MODELE MATHEMATIQUE

Comme nous l'avons déjà mentionné, le comportement d'un système peut être représenté par un ensemble de relations mathématiques; constituant ainsi le modèle mathématique de ce système.

Ce modèle doit refléter fidelement les contraintes et l'objéctif du systeme réel.

La formulation mathématique du modèle s'effectue de différentes façons selon le type de système:

a/ par des relations aux equations algébriques (processus statiques).

b/ par des relations intégro-differentielles (processus dynamiques).

c/ par des equations aux dérivées partielles (système à paramétres distribués).

d/ par des equations aux differences (systèmes à temps discrets) .

La table (2) donne les différents modèles mathématiques des systèmes linéaires :

type de systèmes linéaires	forme	observations
système statique	$Y = A\,X$	X:vecteur d'entrée K^m Y:vecteur de sortie K^n A:matrice (mxn)
système dynamique autonome	$\dot{X} = A\,X(t)$	X K ,A:matrice (nxn)
système dynamique	$\dot{X}(t) = A\,X(t) + B\,U(t)$	U K ;B:matrice (nxm)
système dynamique variant	$\dot{X}(t) = A(t)X(t) + B(t)U(t)$	A,B:matrices fonction du temps
système dynamique asservi	$\dot{X}(t) = A(t)X(t) + B(t)U(X(t))$	

Table (2)

3 - DIFFERENTES METHODES D'OPTIMISATION DES SYSTEMES LINEAIRES

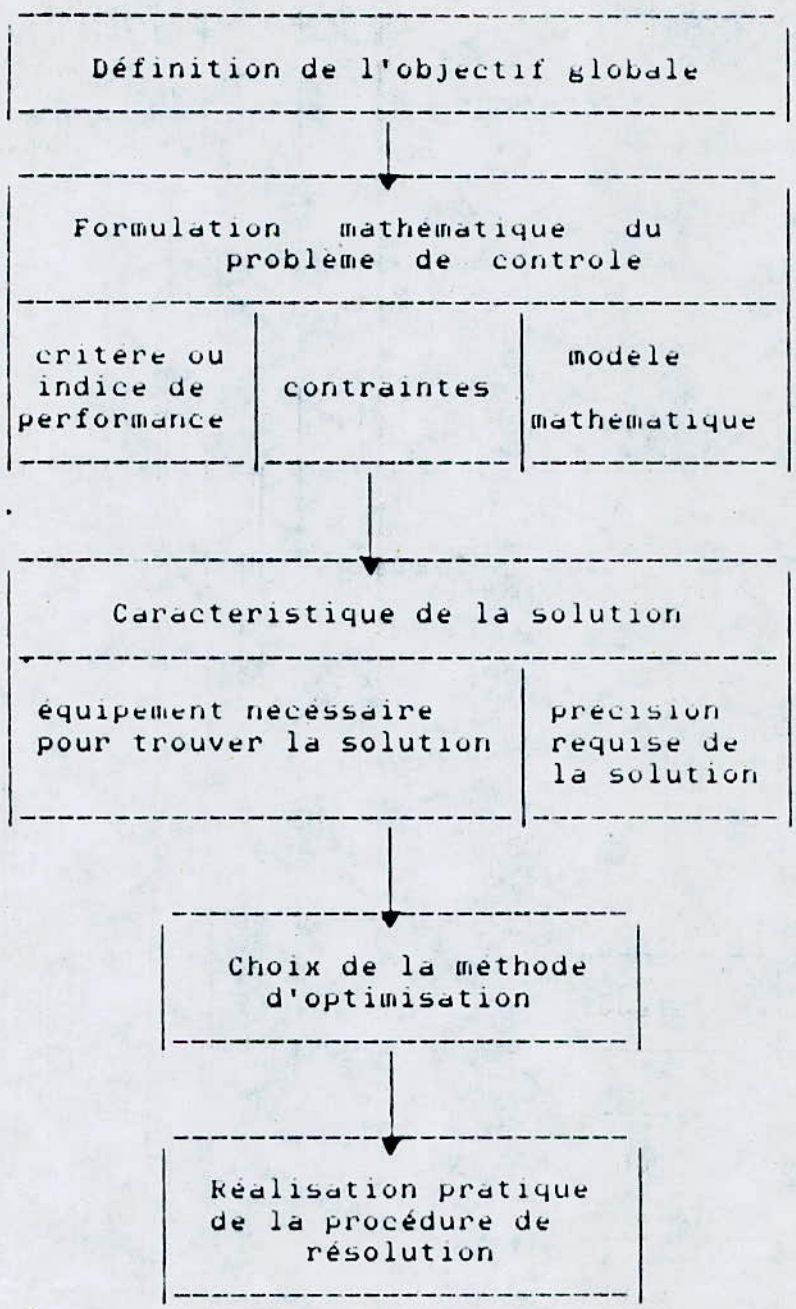
Ayant maintenant la formulation du probleme (modele mathematique,fonction objectif,contraintes)à notre disposition,on peut donc appliquer l'une des méthodes d'optimisation citée dans la table (3) suivant le type de probleme posé,afin de trouver les trajectoires ou controleurs optimaux.

Representation du probleme	Methode de résolution
1 - Processus linéaire statique $J=G_i \dot{x}_i \quad i=1..n$ $a(i,j)\ x_i \ (\ ,=,\ )b_i$ $j=1..m : x_i \ \ 0; i=1..n$	methode de la programmation lineaire (Simplexe)
2 - Processus linéaire dynamique $J=J(x,u,t)$ linéaire $\dot{x}=A\ x + B\ u$ $x \in X, u \in U$ x:vecteur d'état dans E u:vecteur de commande dans E A:matrice (mxn) B:colonne d'ordre n	méthode de la programmation dynamique (principe de maximum)
3 - Processus statique ou dynamique complexe	décomposition-coordination controle hierarchique et optimisation.

Table (3)

4 - CONCLUSION

L'organigramme ci-dessous décrit les differentes etapes de contrôle et optimisation d'un processus.


Définition de l'objectif globale
Formulation mathématique du problème de controle
critère ou indice de performance
contraintes
modèle mathématique
Caracteristique de la solution
équipement nécessaire pour trouver la solution
précision requise de la solution
Choix de la méthode d'optimisation
Réalisation pratique de la procédure de résolution

V/ METHODE D'ANALYSE DES GRANDS SYSTEMES

1 - PRINCIPE D'AGREGATION :

La description précise de nombreux systèmes physiques conduit à un nombre important d'equations différentielles.

De ce fait,l'analyse de ces systèmes et l'application de résultats classiques de la théorie du contrôle optimale sont difficiles,voire impossibles.

L'agrégation,qui est une des techniques d'analyse des grands systèmes dynamiques linéaires,permet d'éviter ces difficultés en reduisant le système réel de dimension n en un modèle réduit de dimension m telque $m < n$.

L'avantage de cette technique consiste en l'existance d'une relation linéaire entre les états du système reel et réduit de la forme $Z = L X$.

Cette relation explicite,a meme un certain nombre de propriétés remarquables tant en modélisation qu'en contrôle et commande.

L'objectif de cette méthode est de déterminer les matrices (F,G,L)du modèle réduit à partir du système réel(voir [7] et [8]).

Système réel $\begin{cases} \dot{x}(t)=A\,x(t)+B\,u(t) \\ y(t)=C\,x(t)+D\,u(t) \end{cases}$ ---- modele agrege $\begin{cases} \dot{z}(t)=F\,z(t)+G\,u(t) \\ y(t)=H\,z(t) \end{cases}$

2 - PRINCIPE DE LA DECOMPOSITION

La décomposition d'un grand système consiste à subdiviser ce dernier en N sous systèmes,invariants,coordonnés ensembles pour donner le système initial.

- Une première méthode consiste à partir de processus invariants dans le temps,le subdiviser en plusieurs sous processus.Dans ce cas on parle de décomposition statique du grand système(voire le chapitre de decomposition de Dantzing-Wolfe).
- Une autre méthode pourra se faire en partant des observations du changement d'état de grand système.Celle ci se caractérise par la décomposition du système en sous systèmes représentés par les vecteurs d'état et les matrices d'état aux quels leurs equations d'état convergent à la solution globale du système .Dans ce cas on parle de la décomposition dynamique du grand système(voir la méthode de coordination du but).

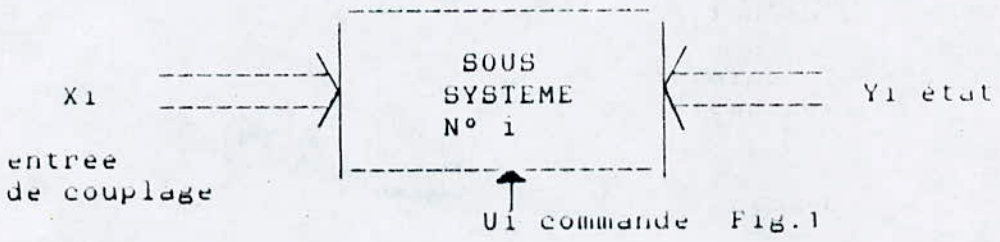


Fig.1

Pour reduire la complexité du probleme globale de grande dimension,on effectue une partition du critere.C'est sur ce principe que repose la theorie du controle hierarchique.

3 - PRINCIPES GENERAUX DU CONTROLE HIERARCHIQUE

3 - 1 - STRUCTURE A PLUSIEURS NIVEAUX-PLUSIEURS OBJECTIFS

Le probleme de controle ou de commande peut etre representé par la structure de la Fig.(2) dite à simple niveau-simple objectif.

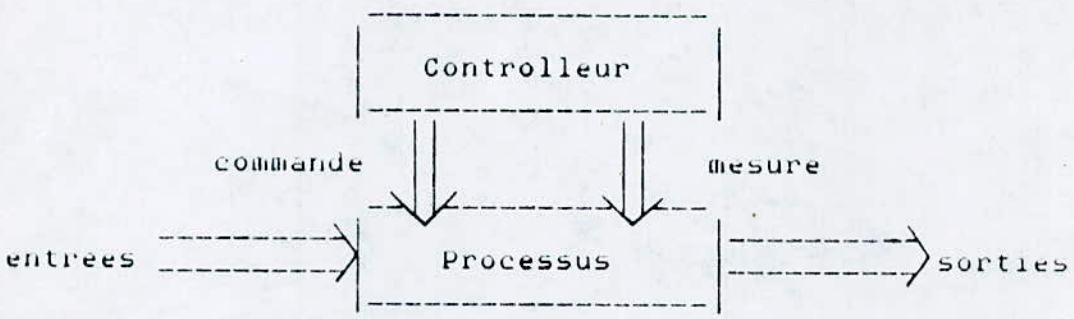


Fig.2

La mise sous cette forme d'un grand systeme complexe est tres difficile en effet si:

Nous considerons le système globale comme forme de sous systèmes indépendants on peut alors le mettre sous la forme représentée dans la Fig.(3) dite à simple niveau-plusieurs objectifs.

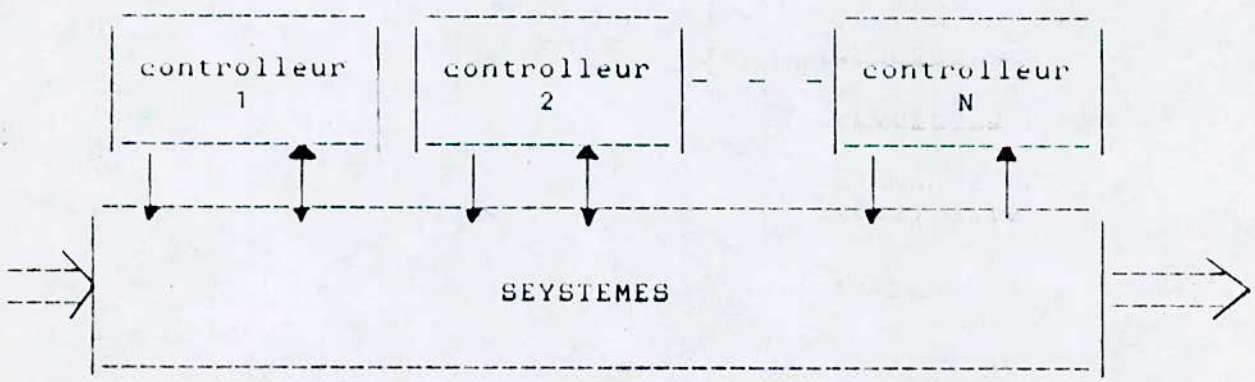


Fig.3

En réalité l'hypothèse d'indépendance n'est pas vraie surtout dans le cas de systèmes multidimensionnels ou il y'a de fortes interactions entre les différents sous-systemes

et,en cas d'absence d'un ordre de priorite,il peut y avoir des conflits entre les differents controlleurs (auxquels sont associés chacun une fonction objectif).

Pour pallier à cette situation et resoudre les dits conflits,on introduit un second niveau de contrôle qui prendera en compte les interactions et modifiera eventuellement l'objectif de quelques controlleurs;nous obtenons ainsi une structure de contrôle appelée structure à plusieurs niveaux-plusieurs objectifs.

En somme,nous considérons que l'operation de contrôle consiste en plusieurs controlleurs hierarchiquement distribués et disposés en forme pyramidale voir Fig.(4) de telle façons qu'au premier niveau à chaque sous-systeme on associe un controlleur cette structure se caractérise par les points suivants:

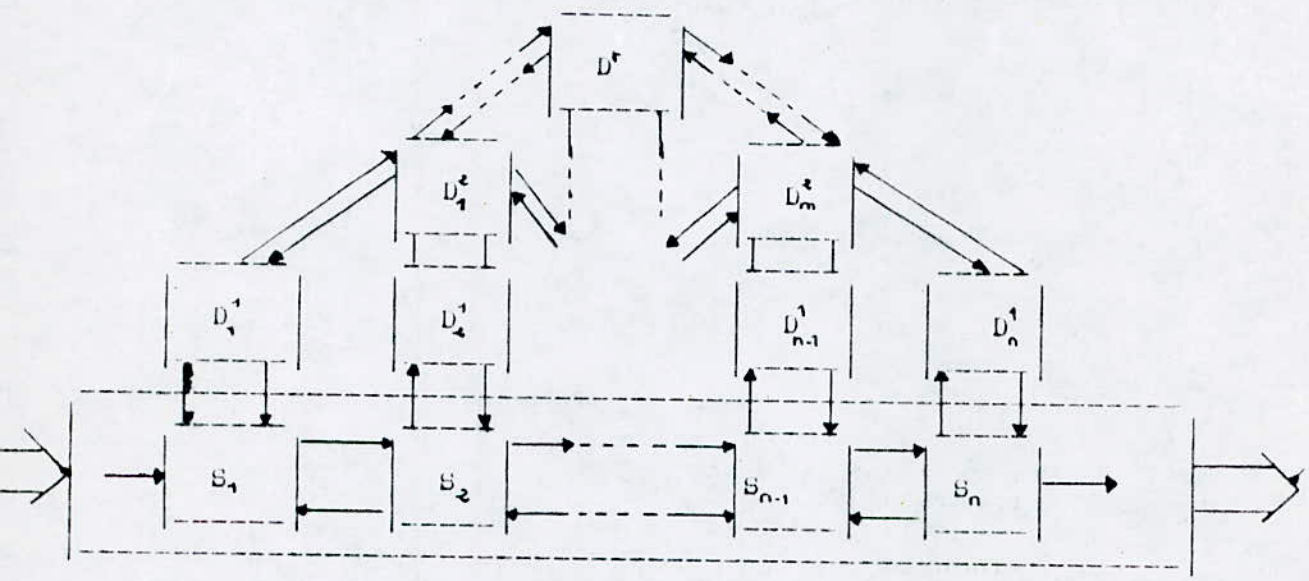


Fig.4

1/ Les controlleurs des niveaux bas sont independants les uns des autres et sont en général inconnues des autres unités.

2/ Ils reçoivent l'information de ceux du niveau hierarchiquement superieur et les utilisent pour contrôler ou commander les sous-systèmes ou controlleurs d'un niveau inferieur

3/ Les controlleurs sont distincts en terme fonctionnel ce qui signifie qu'ils peuvent être implementés sur un même ordinateur ou bien distribué sur un ensemble d'ordinateurs connectés.

4/ Les niveaux superieurs de la hierarchie dont les objectifs doivent être définis,sont des niveaux coordinateurs,ils assurent l'obtention d'une solution du problème globale.

Deux notions sont fondamentales dans ces structures multiniveaux;la décomposition et la coordination.

3 - 2 - DECOMPOSITION DES PROBLEMES

Il existe deux modes de décomposition:

3 - 2 - 1 - DECOMPOSITION VERTICALE

Cette méthode n'introduit pas de modification dans le critère, mais elle impose une hierarchie dans l'ordre de résolution des equations du système.

Illustrons ce mode par un exemple simple:

Soit la fonction J des variables (x,y,u,w) séparable sous la forme :

$$J = J1(x,y,u) + J2(x,y,w)$$

On cherche à determiner $J^* = \text{optimum } J$

J1 et J2 étant liées et $J1 = J1(x^*, y^*, u^*)$ il est necessaire que J2 soit optimale pour les même valeurs de x et de y; c'est à dire $J2 = J2(x^*, y^*, w^*)$ pourque soit verifiée l'equation

$$J^* = J1^* + J2^*$$

Cette méthode consiste à traiter les variables communes du premier niveau qui fournira les valeurs x^* et y^* par une méthode itérative.

3 - 2 - 2 - DECOMPOSITION HORIZONTALE

Cette méthode modifie le critère par l'adjonction d'une variable de coordination, afin d'obtenir des sous fonctions indépendantes. Cette décomposition est utilisée si on introduit des variables de coordination liées à la nature fortement non linéaire du critère [5].

Soit par exemple, la fonction J des variables (x,u,w) séparable

$$J = J1(x,u) + J2(x,w)$$

En posant $x = z$

on écrit $$J = J1(z,u) + J2(z,w)$$

Le paramètre de Lagrange permet d'ecrire le Lagrangien :

$$L = J1(x,u) + J2(x,w) - \lambda(x,z)$$

qui devient : $L = L1 + L2$

$$L1 = J1(x,u) + \lambda x$$

$$L2 = J2(x,w) + \lambda z$$

L1 et L2 sont indépendants pour un λ donné.

La methode consiste donc à déterminer | au niveau superieur par un algorithme coordinateur,puis resoudre independamment L1 et L2 qui se situent au même niveau inferieur de hierarchie.On definit ainsi une structure hirarchisée à deux niveaux de calcul avec décomposition horizontale du niveau inferieur [5] voir Fig.(6)

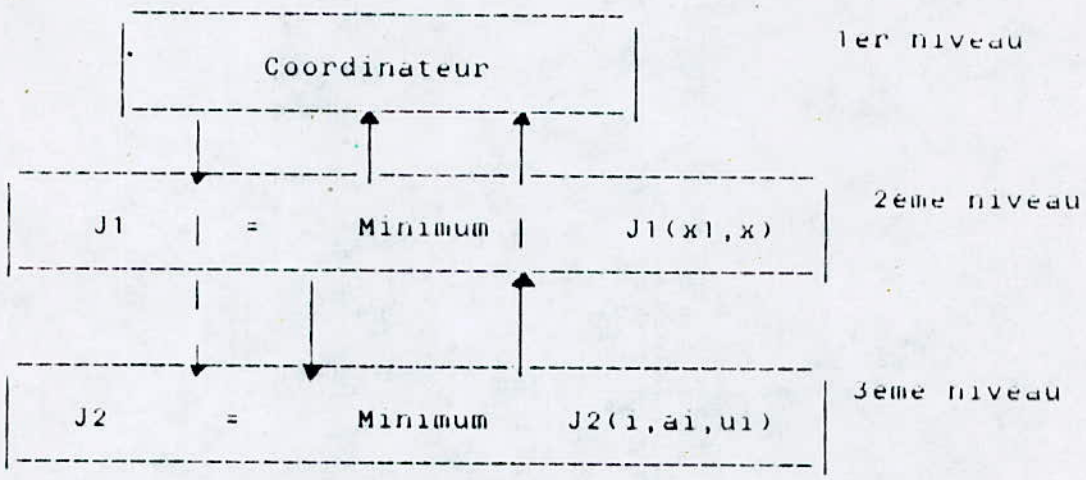


Fig.5 : Principe de décomposition verticale

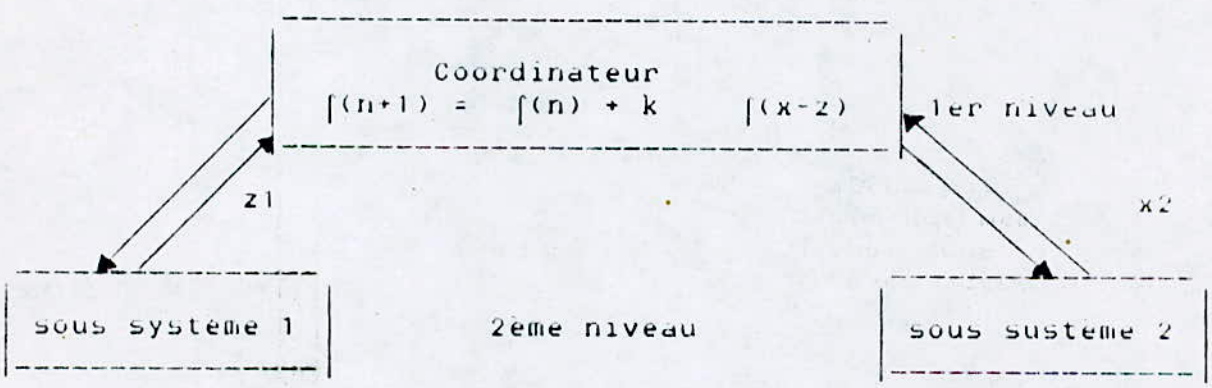


Fig.6 : Princie de la décomposition horizontal

3 - 2 - 3 - PRINCIPE DE LA COORDINATION

Le principe de la décomposition hierarchique est de décomposer le probleme globale P,dont l'objectif est d'optimiser un certain critère globale associé à un systeme complexe en un certain nombre de sous problemes Pi telque :

La solution partielle (P1,P2,...,Pn)n'implique pas la solution du probleme globale

Pour éliminer le conflit,il est nécessaire d'introduir "un vecteur d'intervention au parametre de coordination" et remplacer Pi par Pi(α) on obtient alors:

(P1(α),P2(α),...,Pi(α),...) α = α implique la solution du problème P.

L'optimisation exige donc un choix de α .Parmi les nombreuses méthodes de coordination existantes pour simplifier la présentation nous presentons deux approches tres utilisées;

a/ Methode de coordination du critere:voir le chapitre sur la méthode de coordination de but de la 3eme partie.

b/ Méthode de coordination du modèle :voir [15]

2eme PARTIE

PROGRAMMATION LINEAIRE

CHAPITRE I --- GENERALITES ---

I - 1 - INTRODUCTION

Bien qu'ils ne représentent souvent qu'une approximation brutale de la réalité, les modèles linéaires sont les plus répandus et utilisés en recherche opérationnelle.

Un modèle linéaire est décrit par un vecteur $x=(x1,x2,\ldots,xp)$ de variables dites "principales" (ou structurelles). Ces variables sont soumises à un nombre fini de contraintes du type:

$$\sum_{j=1}^{p} a_{ij}\, x_j \geq, =, \leq b_j \qquad i=1,2,\ldots m \qquad (i)$$

Les coefficients a_{ij}, ainsi que les seconds membres b_i sont réels.

Souvent les variables principales représentent des "niveaux d'activités" et doivent être positives ou nulles, si bien que certaines contraintes de (i) s'écrivent

$$x_j \geq 0 . \qquad (ii)$$

Un vecteur x^0 satisfait (i) est appelé solution réalisable, ou seulement solution de (i). Nous noterons E l'ensemble des solutions de (i).

Pour comparer les éléments de E, le modèle linéaire propose d'évaluer x par une forme linéaire de type:

$$Z(x) = \sum_{j=1}^{p} c_j\, x_j$$

ce qui peut encore s'écrire sous forme vectorielle

$$Z(x) = C\, X$$

avec $C = (c1,c2,\ldots cp,0,0,\ldots 0)$

et

$$X = (\underbrace{x1,\ x2,\ \ldots\ xp}_{\text{variables principales}} \ , \ \underbrace{s1\ ,\ s2\ ,\ \ldots\ sk}_{\text{varibles d'ecart}})$$

Selon le cas, il conviendra de maximiser ou de minimiser. Dans le cas de maximisation, l'objet du programme linéaire est la recherche d'une solution x^* de E vérifiant:

$$\text{quelque soit } x \in E \quad Z(x) \leq Z(x^*)$$

Nous dirons que x^* est solution optimale du programme linéaire.

Nous donnerons ici une présentation algébrique de la programmation linéaire car d'une part, nous le croyons plus simple et d'autre part, nous pensons qu'elle ne fait appel qu'aux concepts fondamentaux de l'algèbre linéaire.

Pour pouvoir appliquer les résultats de l'algebre lineaire au programme lineaire défini par (i) nous transformerons (i) en un système d'equations linéaires en ne laissons subsister que des inéquations de type (ii).

$$\sum_{j=1}^{p} a_{ij} x_j \geq b_j \quad (1)$$

nous pouvons la remplacer par le systeme suivant:

$$\sum_{j=1}^{p} a_{ij} x_j + s_j = b_j \quad (2)$$

De la même façons l'inequation : $\sum_{j=1}^{p} a_{ij} x_j \leq b_j$

peut être remplacée par le système: $\sum_{j=1}^{p} a_{ij} x_j - s_j = b_j ;\ s_j \geq 0$

Les variables ainsi définies sont appelées variables d'écart(ou de surplus).

Nous avons donc maintenant obtenu un système qui ne comporte que deux types de contraintes: equations et contraintes de positivités de certaines variables(type(ii)).

Pour pouvoir traiter de la même maniere toutes les variables (principales et d'écart)il serait souhaitable que toute variable soit soumise à une contrainte de positivité.

Pour cela nous utilisons l'artifice suivant:

Soit xj une variable de signe quelconque,nous la remplaçerons alors par l'expression suivante:

$$x_j = x'_j - x''_j \ ;\ x', x''_j \geq 0$$

grace à cet artifice et avec une extension des notations precedentes tout programme lineaire peut être formulé de la maniere suivante :

$$\begin{cases} A\,X = b & (i) \\ X \geq 0 & (ii) \\ \text{opt } Z(X) = C\,X & (iii) \end{cases}$$

Remarques:

- (i) etant un système d'equation lineaire,nous pourons en toute généralité supposer que le vecteur b a toutes ses composantes positives(b≥0)
- A étant une matrice (nxm),nous supposerons que n>m sinon E ne possède qu'un seul ou pas d'element,et la solution,si elle existe,la notre programme lineaire est triviale.
- (i) et (ii) définissent E,l'ensemble des solutions réalisables.

(iii) est la fonction objectif(economique,critere...)à optimiser.

- Nous supposons le rang de A égale à n s'il n'en etait pas ainsi certaines contraintes seraient redondantes et donc inutiles car elles ne modifient pas E.

La forme générale de la matrice A est la suivante:

	1	2	. . . .	p	. . . .			n
1	a11	a12	. . . .	a1p	1 0 0 0	0	0	0
2	a21	a22	. . . .	a2p	0 1 0 0	0	0	0
m1	.	.	. . . .	.	0 0 0 0 ...1.	0	0	0
m1+1	.	.	. . . .	.	0 0 0 0 ..-1.	0	0	0
m2	.	.	. . . .	.	0 0 0 0 ..-1.	0	0	0
m2+1	.	.	. . . .	.	0 0 0 0	0	0	0
m	am1	.	. . . .	amp	0 0 0 0	0	0	0

On remarque que les colonnes d'ecart(associees aux variables d'écart)sont des vecteurs canoniques vrais ou changes de signe de R^n.
Nous noterons aj le jème vecteur colonne de A.

I-2- PROPRIETES FONDAMENTALES DES PROGRAMMES LINEAIRES

Les propriétés qu'on énoncera résultent des definitions et corrolaires donné par l'annexe A

Theorème 2-1: l'ensemble des programmes d'un probleme de programmation linéaire et l'ensemble de ses programmes optimaux sont,dans l'espace des activités R^n,des polyedres convexes bornés inferieurement.De plus,le polyédre des programmes optomaux est l'intersection des programmes avec un hyperplan d'appui à ce polyèdre.

Corrolaire 1: S'il existe plusieurs programmes optimaux distincts,toute combinaison linéaire convexe de ces programmes est aussi un programme optimal.
- Si on appelle programme extremal un programme constitué par les valeurs des coordonnées d'un point extrémal du polyedre des programmes,on a également le:

Corrolaire 2:S'il existe au moins un programme,il existe au moins un programme extremal.S'il existe au moins un programme optimal fini,il existe au moins un programme optimal extrémal

Theorème 2-2: Un programme non nul est extrémal si et seulement s'il est la solution d'un système régulier de contraintes saturées (voir annexe A).

Theorème 2-3: theorème fondamentalde la programmation linéaire
Etant donné un problème de programmation linéaire sous forme standard:
a/ S'il admet au moins un programme fini,il admet au moins

un programme de base (voir chapitre II-1).

b/ S'il admet au moins un programme optimal fini, il admet au moins un programme optimal de base.

Ces theoremes nous permettent de conclure que l'ensemble des solutions(programmes)de base réalisable d'un problème est identique à l'ensemble des sommets d'un polyedre convexe.

Mais lorsqu'on aborde la résolution pratique des problemes de programmation linéaire la méthode qui .consiste à résoudre tous les systèmes donnant des solutions de base est à exclure.elle conduit en effet à un volume considérable de calculs quand on sait que le nombre de solutions d'un système,à m equations et n inconnues,(si toutes les sous matrices d'ordre m étaient régulières)serait égale à :

$$C_n^m = \frac{n!}{m!\,(n-m)!}$$

C'est à dire pour un problème comportant 10 equations à 20 inconnues,le calcul de toutes les solutions de base exigerait la résolution de 250000 systèmes;ceci d'une part,et sans compter d'autre part les calculs inutiles et le cas ou le programme optimal est infini.

Tout ceci nous amène à considerer une méthode de résolution des programmes linéaires beaucoup plus efficace dite méthode du simplexe qui nous permettre d'eviter la majorité des inconvénients cités ci-dessus .

CHAPITRE II

LA METHODE DU SIMPLEXE

C'est une méthode qui consiste à construire et examiner une suite de sommets adjacents $x^0, x, x^2, \ldots, x^k, x^{k+1}, \ldots, x$ telle que

1/ Le segment (x^k, x^{k+1}) est une arête de polyèdre des solutions de E

2/ $z(x^{k+1}) > z(x^k)$

La convergence de cette méthode est assurée par la convexité de E et le caractère linéaire de la fonction objectif z .

II-1- BASE REALISABLE D'UN PROGRAMME LINEAIRE (voir annexe B)

Rappellons la définition de l'espace des solutions E de notre programme linéaire :

$$\begin{cases} A\,X = b & (i) \\ X \geq 0 & (ii) \end{cases}$$

Sous la forme vectorielle E peut etre definie par:

$\sum_{j=1}^{n} a_j x_j = b \; ; \; X \geq 0$ (a_j:le jème vecteur colonne de A)

Appelons $N=(1,2,3,\ldots\ldots n)$ l'ensemble des indices des colonnes de A.Une base réalisable de E est un ensemble de m vecteurs colonnes a_j;soit $(a_j)_{j\in\beta}$;telque:

1/ Les $a_j; j\in\beta$,sont indépendants

2/ L'unique solution du système :

$$\sum_{j\in\beta} a_j x_j = b \quad ;\text{verifie pour tout } j\in\beta \;\; x_j \geq 0$$

Nous appelerons variables de base les x_j , $j\in\beta$ et nous noterons X_β le vecteur colonne associé.

Nous appelerons variables hors base les x_j , $j\in(N-\beta)=\bar{\beta}$ et nous noterons $x_{\bar{\beta}}$ le vecteur colonne associé.

Si B est la matrice carrée des vecteurs colones $a_j, j\in\beta$, la solution du système (i) est donnée par:

$$X_\beta = B^{-1} b$$

(X_β est un vecteur colonne de variables;par contre X_B est un vecteur colonne numérique)

B est la matrice de passage de la base canonique C à la base β,il en résulte que le vecteur colonne des coordonnée de a_j dans la base β,soit y_j s'exprime par :

$$y_j = B^{-1} a_j$$

Il est important de remarquer que si ß est une base réalisable de E ,alors le vecteur de R^n défini par :

$$X_B = B^{-1} b \quad j\epsilon ß \qquad ; \qquad X_j = 0 \quad j\epsilon \bar{ß}$$

est une solution du programme linéaire;c'est un élément de E,nous appellerons une telle solution "solution de base".

Notons qu'une solution de base possède au moins (n-m) coordonnées nulles.

La valeur de la fonction associée à une base ß est:

$$z = z(X_B) = \Sigma\, c_j x_j + \Sigma_{\bar{ß}} c_j \cdot 0$$
$$j\epsilon ß \qquad j\epsilon \bar{ß}$$

en notons $C_ß$ le vecteur lignes des $c_j, j\epsilon ß$ on obtient:

$$z = C_ß X_B = C_ß B^{-1} b$$

Définissons également,en liaison avec la notion de base réalisable,les quantités:

$$z_j = C_ß y_j = C_ß B^{-1} a_j \quad j=1,2,\ldots n$$

Ces quantités,ou plus précisement,les scalaires $c_j - z_j$ $j=1\ldots n$ vont jouer un rôle important dans la suite.

Appelons en effet,$\bar{B}$ la matrice des colonnes hors base (B possède (n-m)colonnes et m lignes)associée à une base ß nous pouvons exprimer les variables de base en fonction des variables hors-base de la façons suivante:

(1) peut s'écrire $B_ß X_ß + \bar{B} X_{\bar{ß}} = b$

en prémultipliant par : $B_ß^{-1}$,nous obtenons

$$X_ß = B_ß^{-1} b - B_ß^{-1} \bar{B} X_{\bar{ß}} \qquad (1')$$

d'autre part,(11) peut s'écrire: $X_ß \geq 0$, $X_{\bar{ß}} \geq 0$ (11')

Pour la fonction objectif nous avons: $z = C_ß X_ß + C_{\bar{ß}} X_{\bar{ß}}$

Donc en substituant,d'après (1'),nous obtenons:

$$z = C_ß B_ß^{-1} b - C_ß B_ß^{-1} \bar{B} X_{\bar{ß}} + C_{\bar{ß}} X_{\bar{ß}}$$

soit $z = z + (C_{\bar{ß}} - C_ß B_ß^{-1} \bar{B}) X_{\bar{ß}}$ (111')

Examinons le vecteur : $C_{\bar{ß}} - C_ß B_ß^{-1} \bar{B}$

C'est un vecteur dont la composante associée à l'indice j de $\bar{ß}$ est: $c_j - C_ß B_ß^{-1} a_j$; c'est à dire ,$c_j - z_j$,nous voyons donc que $c_j - z_j$ est le coefficient de la variable $x_j (j\epsilon ß)$ lorsque l'on exprime la fonction objectif z par rapport aux variables hors-base.

Remarque:

Les scalaires $c_j - z_j$ $j=1,\ldots,n$ sont appelés les coûts marginaux associés à la base β. Ils représentent en effet l'accroissement de la fonction critère correspondant à un accroissement d'une unité de la variable x_j.

Notons enfin que ces coûts associées aux variables de base sont nulles.

Comme nous l'avons déja signaler nous ne nous intéressons désormais dans Σ qu'aux solutions de base; pour cela nous allons examiner dans le paragraphe suivant comment, à partir d'une base réalisable donnée, en construire une nouvelle:

- qui n'en diffère que par un seul vecteur

- telle que la solution de base correspondante soit la meilleur au sens de la fonction objectif

I-2- AMELIORATION D'UNE SOLUTION DE BASE

Considérons une base réalisable β supposée connue (nous assimilons la base à l'ensemble des indices des vecteurs colonnes de A qui le composent).

Soit alors i_0 un indice hors-base et j_0 un indice de base; déterminons une condition nécessaire et suffisante pour que $\delta = \beta \cup \{i_0\} - \{j_0\}$ soit une base réalisable: la matrice des coordonnées de δ dans la base β est la suivante en posant, $\beta = (a_{i0}, a_{i1}, \ldots, a_{i(n-1)})$ et $\delta = (a_{j0}, a_{i1}, \ldots, a_{i(n-1)})$

	a_{j0}	a_{i1}	a_{i2}	. . .	$a_{i(n-1)}$
a_{i0}	$Y_{i0,j0}$	0	0	. . .	0
a_{i1}	$Y_{i1,j0}$	1	0	. . .	0
a_{i2}	$Y_{i2,i2}$	0	1	. . .	0
.	.	.	.		.
.	.	.	.		.
.	.	.	.		.
$a_{i(n-1)}$	$Y_{i(n-1),j0}$	0	0	. . .	1

La matrice ci-dessus doit être régulière d'où une condition nécessaire et suffisante pour que δ soit une base est que $y_{i0,j0}$ soit différent de 0 (déterminant).

Examinons maintenant qu'elles sont les conditions pour que δ soit réalisable.

β étant une base réalisable, il en résulte que la solution unique de l'équation vectorielle:

$$\sum_{k=0}^{n-1} a_{ik} x_i = b \quad (1)$$

est telle que quelque soit k $x_i \geq 0$

l'équation vectorielle associée a δ est:

$$x'_{j_0}\, aj_0 + \sum_{k=1}^{n-1} x'_{1_k}\, a1_k = b$$

$$\text{mais} \quad aj0 = \sum_{k=0}^{n-1} y1_k\, j_0\, a1_k \qquad (2)$$

(1) et (2) nous permettent d'écrire:

$$x1_o \left[\frac{1}{y1_o\, j_o} aj_0 - \sum_{k=1}^{n-1} \frac{y1_k\, j_o}{y1_o\, j_o} a1_k \right] + \sum_{k=1}^{n-1} x1_k\, a1_k = b$$

et en regroupant les termes

$$\frac{x1_o}{y1_o\, j_o} aj_0 + \sum_{k=1}^{n-1} \left[x1_k - x1_o \frac{y1_k\, j_o}{y1_o\, j_o} \right] a1_k = b$$

D'ou nous pouvons tirer: $x'j_o = \dfrac{x1_o}{y1_o\, j_o}$

$$x'1_k = x1_k - x1_o \frac{y1_k\, j_o}{y1_o\, j_o} \qquad k = 1,2,.....n$$

δ ne sera realisable que si les x' sont positifs,ou nuls,pour cela envisageons deux cas:

1er cas / $x1_o = 0$ alors $x'j0 = 0$

et $x'j_k = x1_k \qquad k=1,2,.....n-1$

Dans ce cas on dit que la base ß est degeneree(voir chapitre II-6)

2ème cas / $x1_o \geq 0$

La condition $x'j_o \geq 0$ implique $y1_o\, j_o$ superieur a 0

Examinons maintenant les n-1 conditions $x'1 \geq 0$

Pour les indices $k/y1_o\, j_o \leq 0$, on a $x'1_k \geq 0$

Soit K♦ l'ensemble des indices/$y1_k > 0$;pour verifier $x'1_k \geq 0$ on doit avoir quelque soit $k \in K♦$ $\dfrac{xi_o}{yi_o\, j_o} \leq \dfrac{xi_k}{yi_k\, j_o}$ cette derniere condition est dite " 2eme CRITERE DE DANTZING "

Remarques:

- Si l'on se fixe aj comme vecteur"entrant"dans la base ß,alors nous voyons en générale;le vecteur"sortant"de cette même base est détermine de façons unique par le 2ème critère de Dantzing .
- Notons que la discusion du cas 2 a été fondée sur l'existance d'un indice i_o tel que $i_o \in K♦ (y1_o\, j_o) > 0)$,si K♦ était vide la solution du programme linéaire serait alors infinie [3]

K♦ etant non vide,examinons dans quelle condition le changement de base réalisable precédent peut être profitable sur le plan de la fonction objectif.

Posons: $r = \min_{k \in K^+} \left[\frac{x1_k}{y1_{k\,j_0}} \right]$

exprimons la valeur z_δ de la de base associee a la base δ:

$$z = cj_0\, x'j_0 + \sum_{k=1}^{n-1} c1_k\, x'1_k$$

en remplaçant les x' par leurs expressions en fonction de x,et en regroupant les termes,il vient :

$$z_\delta = \sum_{k=1}^{n-1} c1_k\, x1_k + \frac{x1_0}{y1_0\, j_0} \left[cj_0 - \sum_{k=0}^{n-1} c1_k\, y1_{k\,j_0} \right]$$

De maniere plus condensee , nous pouvons ecrire

$$z_\delta = z_\beta + r(cj_0 - zj_0)$$

Si donc nous nous voulons que z_δ soit meilleur que z_β ,nous devons choisir le vecteur aj de maniere à agir sur le terme $r(cj_0 - zj_0)$ et comme $r \geq 0$ (résultant des discussions precedentes)il en résulte que pour une maximisation il faut choisir le aj_0 entrant telque $cj_0 - zj_0$ soit le plus grand coût marginal positif

Parcontre dans le cas d'une minimisation il faut choisir aj_0 /$cj_0 - zj_0$ soit le plus petit coût marginal negatif,il s'agit la du 1er critère de DANTZING

II-3 CONDITIONS D'OPTIMALITE

Si tous les couts marginaux associés à la base β sont negatifs ou nuls (dans le cas d'une maximisation)il n'existe pas de base réalisable adjacente dont la solution de base serait strictement meilleur que celle associée à β

Montrons que la solution de base associee à β est une solution optimale de nôtre programme lineaire.

Soit $x' = (x'1, x'2, \ldots\ldots, x'n)$ un élement quelconque de E.La valeur de la solution est:

$$z' = z'(x') = \sum_{j=1}^{n} cj\, x'j$$

x étant la solution de base associee à β

D'autre part,d'après l'hypothèse,nous avons :

$$cj - zj \leq 0 \qquad j = 1, 2, \ldots\ldots n$$

nous allons montrer qu'alors $z = z(x) \geq z(x') = z'$

x' étant une solution du programme lineaire,il en resulte que:

$$\sum_{j=1}^{n} x'j\, aj = b \qquad (6)$$

or $aj = \sum_{k=0}^{n-1} y1_{k\,j}\, a1_k$

et (6) devient : $\sum_{j=1}^{n} \left(\sum_{k=0}^{n-1} x'j\, y1_{k\,j} \right) a1_k = b$

Comme la decomposition de b sur la base β est unique,on a les relations suivantes:

$$x1_k = \sum_{j=1}^{n-1} x'j\, y1_{k\,j} \qquad k = 0, 1, \ldots\ldots n-1 \qquad (7)$$

x est une solution de base il en resulte que:

$$z(x) = \sum_{k=0}^{n} ci_k \; xi_k$$

en remplaçant xi par son expression dans (7),il vient:

$$z(x) = \sum_{k=0}^{n-1} ci_k \left(\sum_{j=1}^{n} x'j \; yi_k j \right)$$

en inversant à nouveau les sigmas somme;nous aurons:

$$z(x) = \sum_{j=1}^{n} x'j \sum_{k=0}^{n-1} ci_k \; yi_k j = \sum_{j=1}^{n} x'j \; zj$$

or $zj \geq cj$ quelque soit $j=1,2,.....n$

et $x'j \geq 0$ quelque soit $j=1,2,.....n$

Donc $z(x) \geq \sum_{j=1}^{n} x'j \; cj = z(x')$

Nous venons donc de démontrer que la solution de base x associée à la base ß est une solution optimale du programme linéaire.

Les conditions d'optimalité d'une solution de base sont donc :

- Dans le cas d'une maximisation quelque soit j=1,2,...n
  cj - zj ≤0 .
- Dans le cas d'une minimisation quelque soit j=1,2,...n
  cj - zj ≥0 .

ayant décrit les formules de changement de base (critere de Dantzing)ainsi que les conditions d'optimalité nous sommes donc en mesure d'énoncer l'algorithme du Simplexe.

II-4- ALGORITHME DU SIMPLEXE

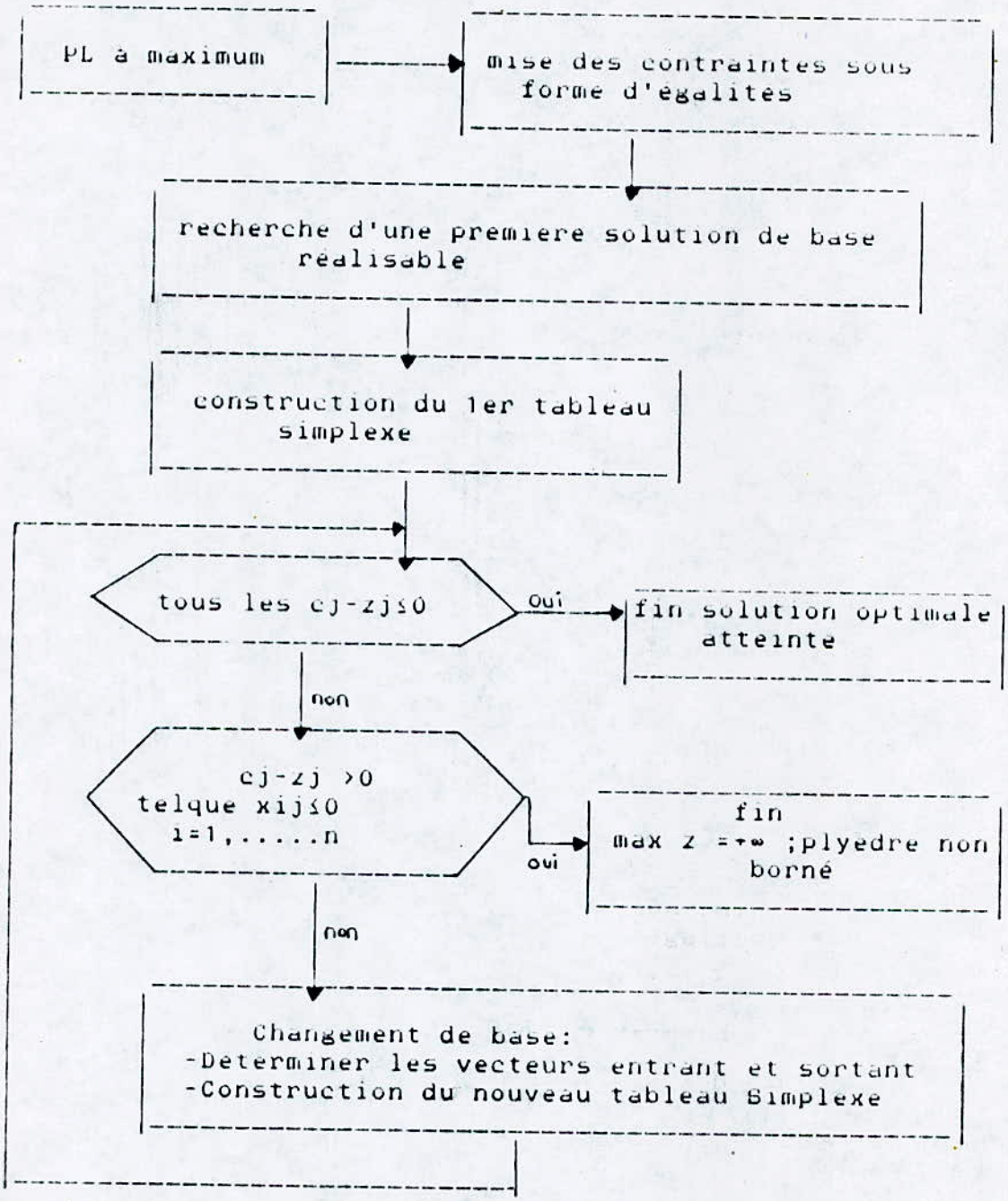
II-4-1- PRINCIPE:

Connaissant une base realisable de depart inutile) ß0 ,appliquons les règles operatoires suivantes (pour un programme linéaire à maximiser)en partant de i=0.

Règles operatoires:

- Déterminer les coûts marginaux associées à la base ßi en cours ;
- Choisir le plus grand;s'il est negatif ou nul,la base ßi est optimale ;sinon faire rentrer le vecteur colonne correspondant dans la base (premier critère de Dantzing) et faire sortir le vecteur imposé par le second critere de Dantzing.Si un tel changement de base n'est pas possible de base,la solution du programme linéaire est infinie,sinon incrementer d'une unité la valeur de i et nommer ßi la nouvelle base réalisable.
- retourner à règles opératoires.

II-4-2- ORGANIGRAMME



Note : Pour les programmes linéaires à minimum même procédure que ci-dessus mais on remplace : -z0 et cj-zj par z0 et zj-cj

Changement de base:

1/ détermination des vecteurs entrant et sortant

ak entre si $c_k - z_k = \max_j (c_j - z_j / c_j - z_j > 0)$

ar sort si $a_r / a_{rk} = \min_i (a_i / a_{ik}$; telque $a_{ik} > 0)$

2/ Construction du nouveau tableau simplexe

ligne du pivot : $a'_k = a_r / a_{rk}$; $a'_{kj} = a_{rj} / a_{rk}$

ailleurs (i k): $a'_i = a_i - (a_{ik} / a_{rk})\, a_r$

$$a'_{ij} = a_{ij} - (a_{ik} / a_{rk})(c_k - z_k)$$

et $-z'_0 = -z_0 - (a_r / a_{rk})(c_k - z_k)$; $c_j - z'_j = c_j - z_j - (a_{rj} / a_{rk})(c_{jk} - z_k)$

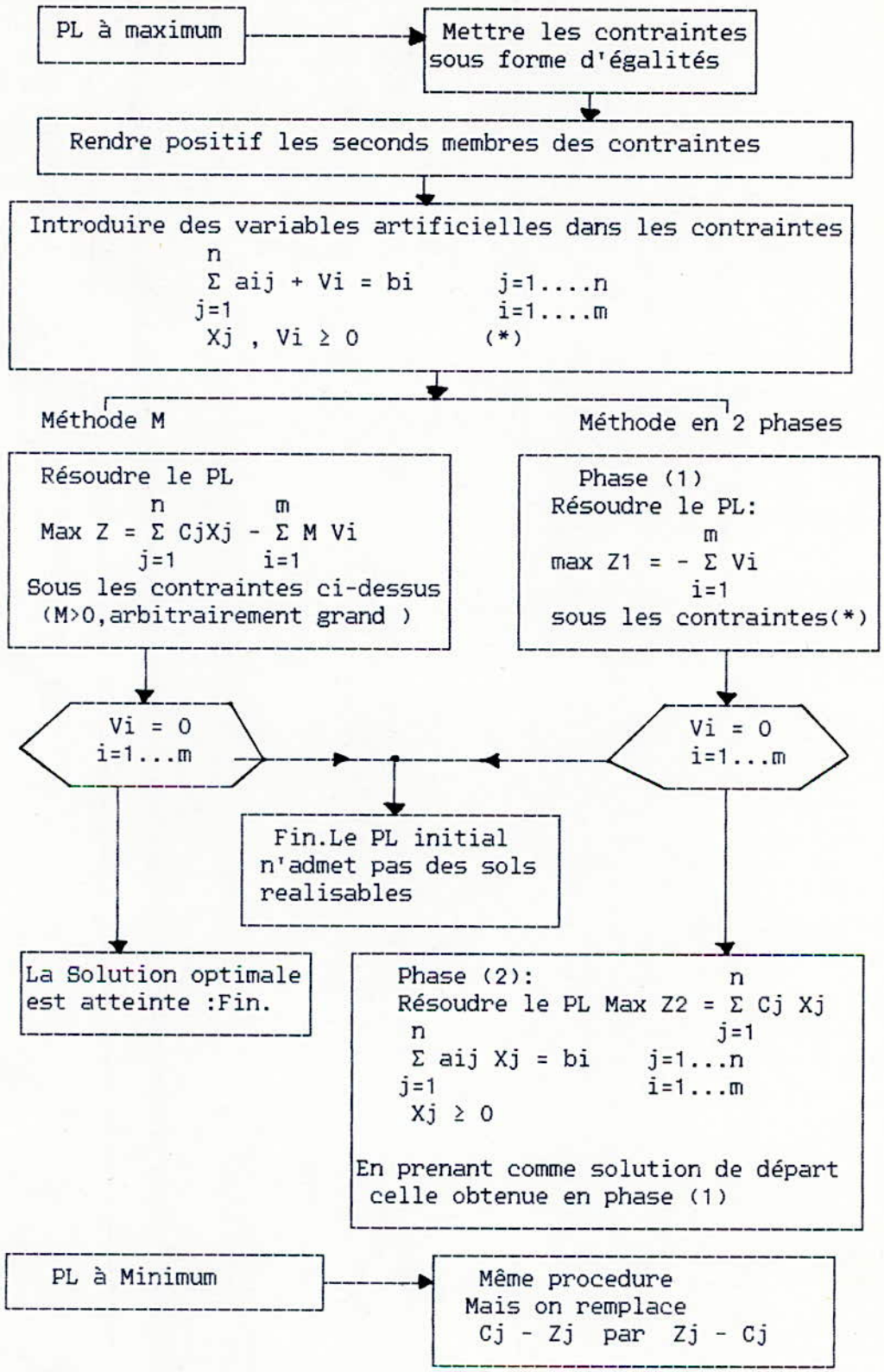
Notons que pour faciliter les calculs numériques on utilise les tableaux du simplexe où figurent à chaque iteration toutes les informations relatives au programme linéaire telque la matrice A le vecteur b les coûts marginaux, les coefficients cj de la fonction critere et les vecteurs qui sont en base [2].

II-5- BASE REALISABLE INTIALE

L'application de la méthode du simplexe à la résolution des programmes linéaires suppose la connaissance préalable d'un solution de base réalisable de départ. Lorsque celle-ci ne peut s'obtenir d'une manière immédiate. On peut avoir recours à deux méthodes qui permettent de construire de maniere systematique cette solution de base réalisable de départ. Ces deux méthodes sont:
La methode M et la méthode en deux phases, reposent sur l'introduction des varibles artificielles .

- Organigrammes de la méthode M et la methode en deux phases:
  Une solution de base réalisable de depart peut etre obtenue par l'application de l'une des deux methodes (voir la demonstration de ces méthodes dans [2] et [3])
  décrites par l'organigramme suivant:

ORGANIGRAMME DE LA METHODE M et CELLE EN 2 PHASES



Remarques:

1/ il n'est pas necessaire d'introduire m variables artificielles pour les contraintes (i)il suffit de les introduire en nombre suffisant pour avoir une premiere solution de base réalisable .

2/ lorsqu'une variable artificielle sort de la base,elle ne doit plus être prise en considération dans la suite de la procédure.

3/ dans la méthode en deux phases,le 1er tableau simplexe de la phase (2) est identique au 1er tableau simplexe de la phase (1) à l'exeception de la ligne cj-zj qui est modifiée par réintroduction des cj.

II-6- DEGENERESCENCE ET CYCLAGE

1 - a/ Solution de base dégénérée :

Une solution de base est dite dégénerée si l'une au moins des variables de base est egale à zéro,c'est à dire,si les vecteurs aj(j=1.....m)formant une base à dimensions des contraintes,les combinaisons linéaires à coefficient non négatifs

$$a1\ x1 + a2\ x2 + \ldots\ldots + am\ xm$$

Comporte au moins un coefficient x1 egale à zéro
la dégénérescence est plus fréquente qu'il n'y parait:

a/ la solution de base réalisable de depart peut evidement être dégenérée .

b/ en appliquant l'algorithme du simplexe,on passe d'une base réguliere(non dégénérée) à une base degenerée lorsque plusieurs variables sont succeptibles d'etre mises hors-base et ne peuvent être départagées par les régles du simplexe.

c/ on passe d'une base degenerée à une autre base dégénérée si la variable mise hors-base est justement celle qui était nulle en base.

2 - Dégénérescence et cyclage

En l'absence de dégénérescence,l'algorithme du simplexe converge vers la solution optimale en un nombre fini de pas, en effet,il n'y a qu'un nombre fini de solution de base réalisables et puisque à chaque itération,la fonction critere est superieure(inferieure)à celle de l'itération precedente pour un problème à maximum(minimum),aucune base ne se présente deux fois

En cas de dégénérescence,ces arguments ne sont plus valables,il se pourrait que l'ont effectue plusieurs chargements sans que la valeur de la fonction objectif change et qu l'on retrouve après un certain nombre de pas une solution de base réalisable déja rencontrée,l'algorithme recommencera alors indéfiniment le cycle de changement de base et ne convergera plus vers la solution optimale.

Le cyclage est donc une consequence accidentelle de la dégenerescence.
Cet inconvinient est toutefois d'ordre théorique,dans la pratique les erreurs d'arrondis rendent un tel cyclage impossible.
Notons enfin qu'il existe un methode dite de perturbation qui permet de passer d'une solution de base realisable à une solution optimale sans retourner à une solution de base anterieure(évitant ainsi le cyclage);son principe est décrit dans le livre de programmation lineaire(voir [1] et [2]).

II-7-INTERPRETATION ECONOMIQUE DE LA METHODE DU SIMPLEXE

Considérons une entreprise qui a n activités J(J=1,2,...n) faisant intervenir m biens i(i=1,2,....m).Ces biens pourraient être soit des facteurs de production,auquels cas ils sont consommés par l'activité considérée,soit des produits,auquel cas ils résultent de cette activité.
Un problème fréquent qui se pose à l'entreprise est le suivant:
Etant donné une disponibilité bi pour chacun des m biens i et un profit unitaire cj pour chacune de n activités j,quel doit être le niveau de chaque activité j pour que la quantité totale des biens i soit inferieure ou égale à la disponibilité bi et que le profit total soit maximal?
(les disponibilités bi,profit,consommation sont des quantités algébriques devant être interpretées au sens large).
Mathématiquement,le problème est décrit par la programmation linéaire [1]
où les:

* xj représentent l'intensité des activités j (output)

* bi representent les disponiblités algebriques des biens i (input)

ou par conventions

si bi≥0 ,bi est effectivement une disponibilite
si bi≤0 ,bi représente une demande égale à -bi

* Les cj sont les profits unitaires attachés à chacune des activités j (profit/input),ou,par convention :

si cj≥0 ,cj est effectivement un profit
si cj≤0 ,cj represente un cout égale à -cj

* Les aij représentent les proportions dans lesquelles l'activité j met en oeuvre le bien i(input/output),ou,par convention :

aij≥0 signifie que j consomme i
aij≤0 signifie que j produit i

* Les coûts marginaux |cj-zj|, j n'appartenant pas à β°(base optimale)sont les quantités dont varie la fonction économique l'orsqu'on modifie la solution optimale en ajoutant une unité à la variable hors-base xj c'est la raison pour laquelle ces quantités (cj-zj) sont appelées les coûts profit ou shadow costs [2].

II-9-PROGRAMMATION DE LA METHODE DU SIMPLEXE

Etant donné le nombre important des problèmes,de différents types et dans différents domaines;traités par la méthode de programation linéaire nous donnons en annexe H1 un programme informatique de résolution des programmes linéaires à l'aide de la méthode du simplexe.

CHAPITRE III

DUALITE EN PROGRAMMATION LINEAIRE

I- PROGRAMME PRIMAL ET DUAL

Forme canonique:

Considerons le programme lineaire a maximum suivant,donné sous forme canonique:

$$\text{Max } Z1 = \sum_{j=1}^{n} cj\ xj$$

$$\begin{cases} \sum_{j=1}^{n} aij\ xj \leq bi \ ; i=1,2,\ldots m \quad \text{(probleme primal)} \\ \qquad\qquad\qquad\quad j=1,2,\ldots n \\ xj \geq 0 \end{cases}$$

Par definition,le dual de ce programme lineaire est le programme linéaire à minimum dont la forme canonique :

$$\text{Min } Z2 = \sum_{i=1}^{m} bi\ yi$$

$$\begin{cases} \sum_{i=1}^{m} aij\ yi \geq cj \ ; j=1,2,\ldots n \quad \text{(probleme dual)} \\ \qquad\qquad\qquad\quad i=1,2,\ldots m \\ yi \geq 0 \end{cases}$$

La forme canonique de ces deux problemes est remarquable par sa symétrie,celle-ci correspond à une symetrie par rapport à la diagonale dans le tableau suivant:

	x1	x2 . . .	xn	≥ 0
y1	a11	a12 . . .	a1n	≤ b1
y2	a21	a22 . . .	a2n	≤ b2
.	.	.	.	. .
.	.	.	.	. .
ym	am1	am2 . . .	amn	≤ bm
				min
0	c1	c2 . . .	cn	max

Les deux programmes linéaires precedents s'écrivent sous forme matricielle :

$$\text{Max } \{\ Z1 = C\ X\ /\ A\ X \leq b\ ,\ X \in \mathbb{R}^n\ \}$$

$$\text{Min } \{\ Z2 = b\ Y\ /\ A\ Y \geq C\ ,\ Y \in \mathbb{R}^n\ \}$$

Forme générale :

A partir de la forme canonique ,il est possible de définir les deux programmes linéaires sous forme generale:

$$\text{Primal :}\quad \text{Max } Z1 = \sum_{j=1}^{n} c_j x_j$$

$$\begin{cases} \sum_{j=1}^{n} a_{ij} x_j \leq b_i \ ; i=1,2,\ldots h \leq m \\ \sum_{j=1}^{n} a_{ij} x_j = b_i \ ; i=h+1,h+2,\ldots m \\ x_j \geq 0 \ ; j=1\ldots\ldots\ldots k \leq n \\ x_j \text{ quelconque } ; j=k+1 \ldots\ldots n \end{cases}$$

$$\text{Dual :}\quad \text{Min } Z2 = \sum_{i=1}^{m} b_i y_i$$

$$\begin{cases} \sum_{i=1}^{m} a_{ij} y_i \geq c_j \ ; j=1,2,\ldots k \leq n \\ \sum_{i=1}^{n} a_{ij} y_i = c_j \ ; j=k+1;k+2,\ldots\ldots n \\ y_i \geq 0 \ ; i=1,2,\ldots\ldots h \leq m \\ y_i \text{ quelconque } ; i=h+1 \ldots\ldots\ldots m \end{cases}$$

autrement dit :

- à toute variable d'écart identiquement nulle(contrainte prenant la forme d'une égalité: $\sum_{j} a_{ij} x_j = b_i$)correspond une variable structurelle duale sous ristriction de signe.
- à toute variable duale sous ristriction de signe correspond une variable d'écart duale nulle (contrainte du type $\sum_{i} a_{ij} y_i = c_j$).

II- THOREME I I PROPRIETES FONDAMENTALES

Théoreme 1- Le dual du dual est le primal (involution)

Theoreme 2- Si le primal admet une solution optimale finie x^*,le dual admet une solution optimale finie y^* et,en ces solutions les fonctions objectifs sont égales $Z1(x^*) = Z2(y^*)$.

Theoreme 3- Une condition nécessaire et suffisante pour que les vecteurs réalisables $X=(x1,x2,\ldots,xn)$et $Y=(y1,y2,\ldots,ym)$soient optimaux (dans le problème primal et dual respectivement)et que leurs composantes vérifient les relations

$$\begin{cases} x_j \left(\sum_{j=1}^{m} a_{ij} - c_j \right) = 0 & ;j=1,2,.....n \\ y_i \left(\sum_{i=1}^{n} a_{ij} - b_i \right) = 0 & ;i=1,2,.....m \end{cases}$$

ie: en la solution optimale

- à toute variable d'écart primale (duale)positive correspond une variable structurelle duale(primale)nulle.
- à toute variable structurelle primale(duale)positive correspond une variable d'écart duale(primale)nulle.

$$\text{Primal (variables structurelles)} \begin{bmatrix} x_j = 0 & u_j > 0 \\ x_j > 0 & u_j = 0 \end{bmatrix} \text{Dual (variables d'ecart)}$$

$$\text{(variables d'ecart)} \begin{bmatrix} S_i = 0 & y_i > 0 \\ S_i > 0 & y_i = 0 \end{bmatrix} \text{(variables structurelles)}$$

III- RESOLUTION DU DUAL

Le dual d'un programme linéaire est lui meme un programme lineaire,on peut donc en trouver la solution optimale en appliquant la méthode du simplexe décrite précédemment.

IV- PASSAGE DU DERNIER TABLEAU SIMPLEXE DU PRIMAL AU DERNIER TABLEAU SIMPLEXE DU DUAL

Lorsque le primal admet une solution optimale finie,il est facile de construire le dernier tableau simplexe du dual à partir du dernier tableau simplexe du primal(et vice versa)il est existe en effet une relation très étroite entre les tableaux simplexes finals-au signe près.

* ligne $\begin{bmatrix} x_j \\ \\ S_i \end{bmatrix}$ en base = colonne $\begin{bmatrix} u_j \\ \\ y_j \end{bmatrix}$ hors-base
* colonne $\begin{bmatrix} x_j \\ \\ t_i \end{bmatrix}$ hors-base = ligne $\begin{bmatrix} u_j \\ \\ y_j \end{bmatrix}$ en base

* ligne cj-z(1)j = colonne P0(dual)
* colonne P0(primal) = ligne z(2)i-bi

avec $z(1)j = \sum_{i} cj\ x(1)ij$, $z(2)i = \sum_{j} bi\ x(2)ij$

où x(1)ij , x(2)ij representent respectivement les elements des derniers tableaux simplexe primal et dual (problème à maximum)
et P0 est le vecteur de base.

V- INTERPRETATION ECONOMIQUE DU LA METHODE DUALE

Etant donné un profit unitaire cj pour chacune des n activies et une disponibilité bi pour chacun des m biens i,quel doit être le prix unitaire de chaque bien i pour que la valeur totale des biens consommés par j soit superieur ou égale au profit cj et que la valeur totale des biens disponibles soit minimale?

Donc le prix fictif traduit la relation existante entre la fonction économique à l'optimum et la disponibilité des biens :

- Si yi=0 ,une petite variation des disponibilités bi n'a pas d'influence sur la valeur optimale de Z

- Si yi>0,une petite variation des disponibilités bi conduit à une variation yi de la valeur optimale.

VI- PRINCIPAUX ALGORITHMES ISSUS DE LA DUALITE

Outre la résolution des problèmes duals à l'aide de la méthode de simplexe la notion de dualité nous donne la possibilité de résoudre les programmes lineaires à l'aide de plusieurs manieres:

IV-1 ALGORITHMES DUAL-SIMPLEXE

Nombreux sont les problèmes de programmation linéaire dont la résolution à l'aide du simplexe necessite l'introduction de variables artificielles ce qui augmenterait malheureusement le nombre de variables,on pourrait alors recourir au principe de dualité pour diminuer l'encombrement des systèmes à résoudre mais souvent la dualité ne permet pas de surmonter cette difficulté pour cause soit de non existance de solution duale réalisable de départ soit encore de l'augmentation du nombre de variables dûe à l'introduction des variables artificielles.

La méthode dual-simplexe permet de résoudre ce genre de programme linéaire sans introduction de variables artificielles et sans recours explicite à la dualité.

a/ Conditions d'applications

Comme nous l'avons indiqué ci-dessus cet algorithme s'applique à tout programme linéaire(appelé ci-desous primal)

- Possédant une base de départ réalisable

- dont le dual(appelé ci-desous dual)possede une base de départ réalisable grâce à la correspondance déjà évoquée; à toute variable en base du dual correspond une variable hors-base du primal et la base de départ du dual est réalisable (P0≥0)si et seulement si tous les elements de la ligne cj-zj (ou zj-cj selon qu'il s'agit ou non d'un programme lineaire à maximum)sont ≤0 ainsi dans un problème à maximum ou ≥0 dans un problème à minimum.

Règles de changement de base:

Ces règles se déduisent immédiatement de celles de l'algorithme du simplexe appliqués au dual.

Algorithme dual-simplexe appliqué au primal
(primal à minimum)

* ar (Sr) sort de base si
  $ar = \min_i(ai/aij<0)$
* Zk (ak) entre en base si
  $\frac{(Zk-ck)}{ark} = \min_j\left(\frac{(Zj-cj)}{arj} / arj<0\right)$

Algorithme simplexe appliqué au dual
(dual à maximum)

* ur (yr) entre en base si
  $cr-Zr = \max_i(ci-Zi/ci-Zi>0)$
* yk (uk) sort de base si
  $\frac{yk}{akr'} = \min_j\left(\frac{yj}{ajr} / ajr>0\right.$

Remarques :

Dans l'algorithme du simplexe on détermine d'abord le vecteur entrant puis le vecteur sortant,dans l'algorithme dual-simplexe,on procède inversement; détermination du vecteur sortant puis celui entrant:

- les pivots sont dans le dual,contrairement au simplexe,tous strictement négatifs

b/ Evolution et règle d'arrêt

Le primal auquel on applique l'algorithme dual-simplexe évolue parallèlement au dual auquel on appliquerait l'algorithme simplexe

Algorithme dual simplexe appliqué au primal (primal à minimum)	Algorithme simplexe appliqué au dual (dual à maximum)
On evolue d'une solution de base non réalisable en solution de base non réalisable tant qu'il existe xi négatif tous les cj-Zj sont non positifs.	On evolue d'une solution de base non réalisable en solution de base non réalisable tant qu'il existe des ci-Zi positifs tous les éléments de P0 sont non négatifs.
On s'arrete des que la base est realisable(tous les xi≥0	On s'arrete des que tous les cj-Zj ≤ 0

VI-2 METHODE DE LA CONTRAINTE ARTIFICIELLE

De même qu'une solution de base réalisable de départ est necessaire à l'application de l'algorithme du simplexe,on ne peut entamer une procédure dual simplexe si on ne dispose pas d'une solution de base non réalisable telle que tous les cj-zj(ou zj-cj,selon qu'on a affaire à un probleme à maximum ou à minimum)soient négatif ou nuls.
On a vu que cette condition est imposée par l'existance,dans le problème dual,d'une solution de base realisable cette fois si cette condition n'est pas remplie, il y'a toujours moyen d'introduire,parallelement aux variables artificielles du dual,des contraintes artificielles,en nombre suffisant pour que les conditions initiales soient remplies et que la méthode de duale-simplexe puisse s'appliquée ,cependant ceci compromettra l'un des objectifs de la méthode duale-simplexe,qui consiste entre autre à simplifier l'etude des programmes linéaires,dans la mesure ou cela gonflera le probleme.
La methode de la contrainte permet d'arriver au même resultat (cj-zj≤0 quelque soit j) sans accroissement notable de la taille du probleme(une seule contrainte supplémentaire)

Principe et organigramme de la methode

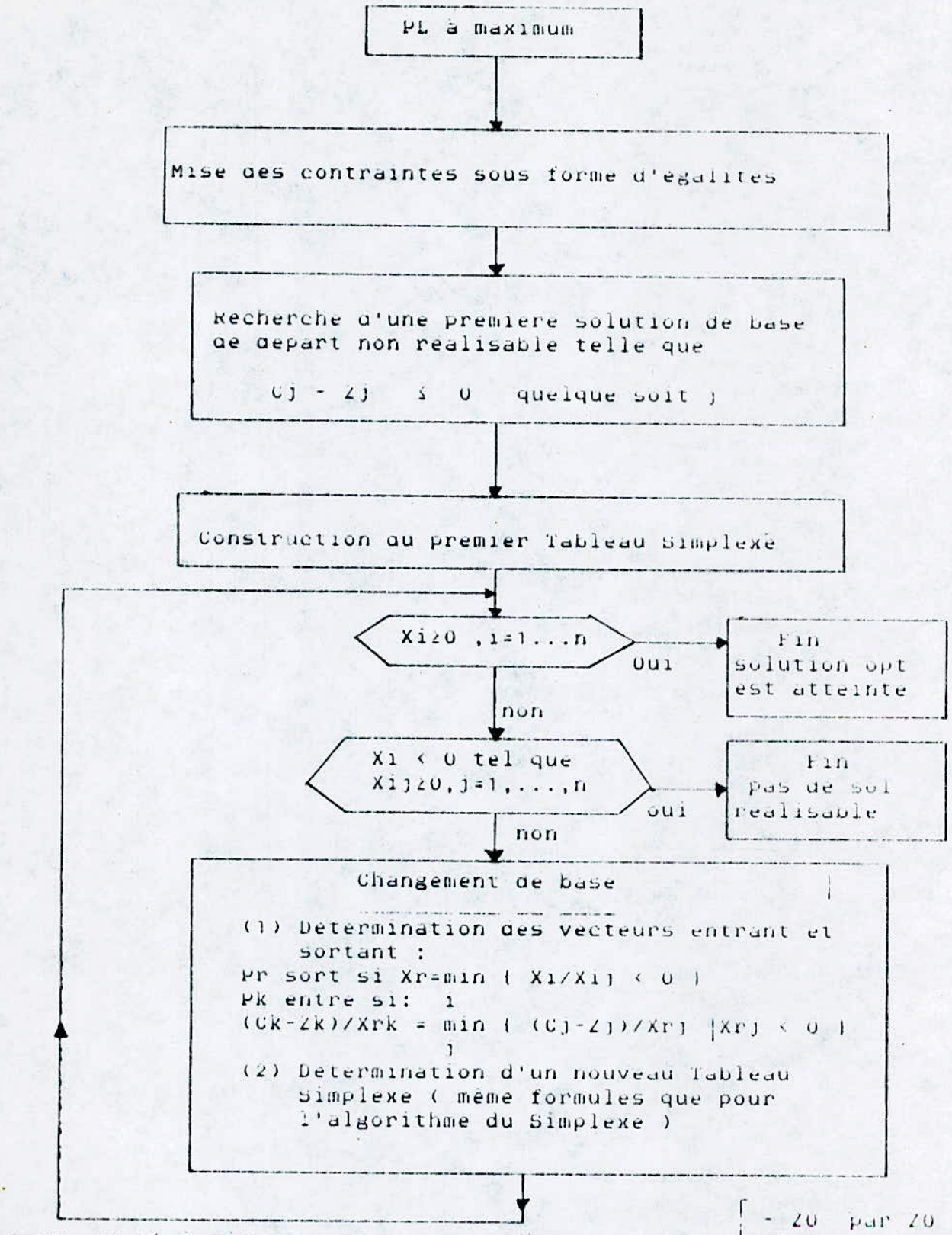
Supposons que le tableau simplexe initial du programme linéaire à maximum fournisse une solution de base realisable :

- Si tous les cj-zj sont negatifs ou nuls on applique

directement la méthode duale-simplexe

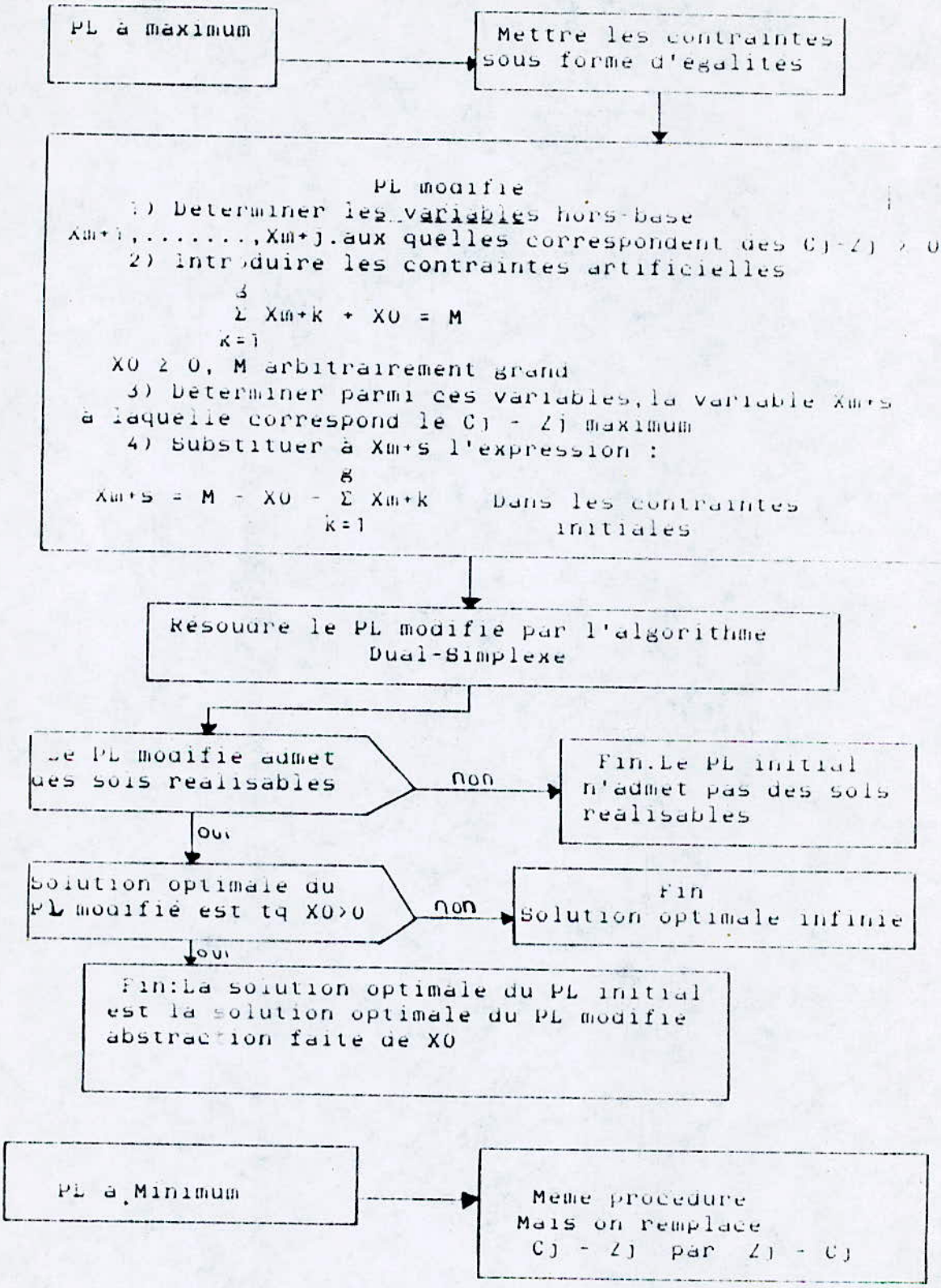
-sinon les cj-zj correspondant à certaines variables hors-base sont strictement positifs:
soient $x_{m+1}, x_{m+2}, \ldots, x_{m+g}$ on peut appliquer la méthode illustrée par l'organigramme dans la page suivante

ORGANIGRAMME DE L'ALGORITHME DUAL-SIMPLEXE



pour un PL à min meme procedure mais on remplace $\begin{bmatrix} -Z0 & \text{par} & Z0 \\ Cj-Zj & & Zj-Cj \end{bmatrix}$

ORGANIGRAMME DE LA METHODE DE CONTRAINTE ARTIFICIELLE



Remarque : Des que x0 entre en base, la contrainte artificielle devient lache pour la solution de base realisable correspondante, les variables de base autres que x0 ne sont plus explicitement fonction de M et elles constituent une solution de base du programme lineaire "initial" satisfaisant au critere d'optimalité(les cj-zj ne sont plus fonction de M) on peut donc à partir de ce moment supprimer dans le tableau simplexe la ligne et la colonne relatives à x0 et continuer l'application de l'algorithme dual-simplexe dans le tableau réduit(qui'est un tableau relatif au programme principal).

CHAPITRE IV

FORME REVISEE DE LA METHODE DU SIMPLEXE

I- INTRODUCTION

L'examen de déroulement des calculs dans l'algorithme ordinaire du simplexe conduit à constater que plusieurs invariants se trouvent présent dans chaque tableau de cette méthode. En effet, à chaque itération on retrouve:

- La sous matrice identité $I_{m,m}$;
- Les mêmes coûts unitaires ;
- Des coûts marginaux nuls pour les variables de la base

Tandis que seules les informations suivantes sont nécessaires:

1/ les coûts marginaux des variables hors-base $\bar{c}_j$
2/ les valeurs des variables en base $\bar{b}$
3/ et le vecteur qui entre dans la base $\bar{a}_s$

Ainsi donc un seul vecteur hors-base est vraiment utilisé par la méthode du simplexe, les autres colonnes restantes seront mémorisées pour rien entrainant ainsi un espace mémoire occupé sans interet et un temps de calcul important (gaspillage).
La méthode du simplexe revisée consiste justement à eviter toutes les demarches inutiles en générant à chaque itération à partir du problème original (en utilisant l'inverse de la matrice de base) les coûts marginaux et le vecteur qui entre dans la base donc: $\bar{c}_j$ et $\bar{a}_s$.

II- PRINCIPES

Considérons le programme linéaire suivant :

$$\text{Max } Z = c_1 x_1 + c_2 x_2 + \ldots + c_n x_n$$

$$\begin{cases} \sum_{j=1}^{n} a_j x_j = b \\ x_j \geq 0 \quad ; \; j = 1 \ldots\ldots n \end{cases}$$

ou $a_j = (a_{1j}, a_{2j}, \ldots\ldots, a_{mj})^T$ jème colonne de la matrice A qui est de rang m

Soit $B = [a_{j1} \ldots\ldots, a_{jm}]$ la base de ce programme linéaire

et soit $x_B = (x_{j1} \ldots\ldots x_{jm})^T$ et $c_B = (c_{j1} \ldots\ldots c_{jm})$;

(vecteur ligne)

Les vecteurs correspondants respectivement aux variables de base et aux facteurs de coût relatif à ces variables

x_B ,supposé realisable, est donné par : $x_B = B^{-1} b = \bar{b}$

Dans cette methode on considerera le systeme complete ou au systeme de m contraintes viendera s'ajouter la (m+1)eme equation décrivant la forme linéaire à optimiser $(\sum_{j=1}^{n} cj - Z = 0)$ avec -Z comme variable de base permanente.
ainsi les vecteurs du systeme initial deviendront :

$$\hat{a}j = (a1j,\ldots,amj,cj)^T \;;j:1\ldots n$$

$$\hat{a}n+1 = (0,\ldots,0,1)^T \quad , \quad \hat{b} = (b1,\ldots,bm,0)^T$$

et le systeme complete est décrit par l'equation :

$$\sum_{j=1}^{n} \hat{a}j\, xj + \hat{p}n+1(-Z) = \hat{b}$$

et comme B est la matrice de base, la (m+1)x(m+1)matrice :

$$\hat{B} = [\hat{a}j1,\ldots,\hat{a}jm,\hat{a}n+1] = \begin{bmatrix} B & 0 \\ c_B & 1 \end{bmatrix}$$

est également la base du systeme complete

On montre également (annexe C)que :

$$\hat{B}^{-1} = \begin{bmatrix} B^{-1} & 0 \\ -c_B B^{-1} & 1 \end{bmatrix}$$

existe si B^{-1} existe.

Definition : le vecteur ligne $\pi = (\pi 1,\ldots,\pi m) = c_B B^{-1}$

est appelé le vecteur des multiplicateurs du simplexe associés à la base B.

Interpretation: (voir [4]) $\pi = c_B B^{-1} \longrightarrow \pi B - c_B = 0$

Donc π est un vecteur qui nous permet d'éliminer les coefficients des variables de base
On peut donc écrire :

$$\hat{B}^{-1} = \begin{bmatrix} B^{-1} & 0 \\ -\pi & 1 \end{bmatrix} \qquad 4.2.1$$

En faisant des réarrangements des coefficients aij de telle maniere à faire apparaitre la matrice B dans les m premieres colonnes, le système peut être écrit sous la forme suivante:

$$\begin{bmatrix} B & | & (aj) & | & 0 \\ - & - & - & - & - \\ c_B & | & (cj) & | & 1 \end{bmatrix} \begin{bmatrix} x \\ - \\ -Z \end{bmatrix} = \begin{bmatrix} b \\ - \\ 0 \end{bmatrix} \qquad 4.2.2$$

En multipliant 4.2.1 et 4.2.2 nous obtenons:

$$\begin{bmatrix} B^{-1} & | & 0 \\ - & - & - \\ -\pi & | & 1 \end{bmatrix} \begin{bmatrix} B & | & (aj) & | & 0 \\ - & - & - & - & - \\ c_B & | & (cj) & | & 1 \end{bmatrix} \begin{bmatrix} x \\ - \\ -Z \end{bmatrix} = \begin{bmatrix} B^{-1} & | & 0 \\ - & - & - \\ -\pi & | & 1 \end{bmatrix} \begin{bmatrix} b \\ - \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} B^{-1} + 0 \times c_B & | & B^{-1} \times (aj) + 0 \times (cj) & | & B^{-1} \times 0 + 0 \times 1 \\ - & - & - & - & - \\ -\pi \times B + c & | & -\pi \times (aj) + 1 \times (cj) & | & -\pi \times 0 + 1 \end{bmatrix} \begin{bmatrix} x \\ - \\ -Z \end{bmatrix} = \begin{bmatrix} B^{-1} & & 0 \\ - & - & - \\ -\pi \times b + 1 \times & & 0 \end{bmatrix}$$

$$\begin{bmatrix} I & | & B^{-1} \times (aj) & | & 0 \\ - & - & - & - & - \\ -\pi \times B + c_B & | & -\pi \times (aj) + (cj) & | & 1 \end{bmatrix} \begin{bmatrix} x \\ - \\ -Z \end{bmatrix} = \begin{bmatrix} B^{-1} \times b \\ - \\ -\pi \times b \end{bmatrix}$$

Or $-\pi \times B - c_B = 0$ pour toutes les variables en base, on peut donc écrire le système sous la forme canonique :

$$x_{j1} \cdot \cdot \cdot x_{jn} \begin{bmatrix} \\ \sum_{j \in \bar{\beta}} \bar{a}_j x_j \\ \\ \end{bmatrix} = \begin{bmatrix} b_1 \\ . \\ . \\ . \\ b_n \end{bmatrix}$$

$-Z + \sum_{j \in \bar{\beta}} \bar{c}_j x_j = -Z_0$: $\bar{\beta}$ étant l'ensemble des indices des variables hors-base

ou (par identification)

$$\bar{b} = B^{-1} b \qquad 4.2.3$$

$$Z_0 = \pi b \qquad 4.2.4$$

$$\bar{a}_j = B^{-1} a_j \qquad 4.2.5$$

$$\bar{c}_j = c_j - \pi a_j \qquad 4.2.6$$

Les deux dernieres formules sont fondamentales, elles montrent comment peuvent être calculees, pour $\hat{B}^{-1}$ donnee (ou B^{-1} et π), les quantités $\bar{c}_j$ et $\bar{a}_j$ necessaires à chaque iteration du simplexe en utilisant les informations initiales c_j et a_j .
ainsi les vecteurs entrant et sortant sont facilement determinés à l'aide des deux criteres de Dantzing.

L'equation 4.2.6_ permet de calculer $\bar{c}_j$ d'où on obtient le vecteur entrant as en appliquant le 1er critere de Dantzing

$$\bar{c}_s = \min_j \bar{c}_j$$

L'equation 4.2.5 nous permettera alors de calculer notre $\bar{a}_s$
Les équations 4.2.3 et 4.2.4 nous donnent le vecteur : $\bar{b}$
Ce qui déterminera le vecteur sortant (2ème critere de Dantzing)

$$\bar{a}_{jr} \text{ tel que } \frac{\bar{b}_r}{\bar{a}_{rs}} = \min_{a_{is}>0} \frac{\bar{b}_i}{\bar{a}_{is}}$$

Ayant determiné les vecteurs entrant et sortant il nous reste maintenant la génération de la nouvelle base
à cet effet considérons la matrice partitionnée suivante :

$$\left[\begin{array}{cccc|c|c} a_{j1} & . & . & a_{jm} \; a_{n+1} & I & \begin{array}{c} a_{1s} \\ a_{2s} \\ . \\ . \\ . \\ a_{ms} \\ c_s \end{array} \end{array}\right] \qquad 4.2.7$$

Multiplions cette matrice par $\hat{B}^{-1}$ on obtient alors

$$\left[\begin{array}{c|c|c} I & \hat{B}^{-1} & \begin{array}{c} \bar{a}_{1s} \\ \bar{a}_{2s} \\ . \\ . \\ \bar{a}_{ms} \\ \bar{c}_s \end{array} \end{array}\right] \qquad 4.2.8$$

en faisant une operation de pivotage autour de $\bar{a}_{rs}$ sur la matrice 4.2.8 on obtient :

$$\left[\; u1 \;.\;.\;.\; ur-1 \;\; a \;\; ur+1 \;.\;.\;.\; \mid \; Q \; \mid \; ur \;\right]$$

T

ou ui = (0,0.....,ei,0.....0) ;ei=1 la ième coordonnée du vecteur égale à 1

L'operation de pivotage de la matrice 4.2.8 est équivalente à la multiplication de 4.2.7 pour la matrice $\hat{B}^{-1}_{new}$ ce qui donne:

$$\left[\; \hat{B}^{-1}_{new} \;.\; \hat{B} \;\mid\; \hat{B}^{-1}_{new} \;\mid\; \begin{matrix} ur \\ - \end{matrix} \;\right] \qquad \text{d'ou } Q = \hat{B}^{-1}_{new}$$

ainsi l'inverse de la nouvelle base $\hat{B}^{-1}_{new}$ est obtenue par une operation de pivotage autour de $\bar{a}rs$ sur la matrice

$$\left[\begin{array}{c|c} \hat{B}^{-1} & \begin{matrix} \bar{a}1s \\ . \\ \bar{a}rs \\ . \\ \bar{a}ms \\ \bar{c}s \end{matrix} \end{array}\right]$$

On retiendra alors les (m+1) premieres colonnes de la matrice résultante pour former la nouvelle base $\hat{B}^{-1}_{new}$.

III- ALGORITHME DE LA FORME REVISEE DU SIMPLEXE

* A , b , C , $\hat{B}^{-1}$ sont les informations initiales du problème (données)

1/ La (m+1)ème ligne de $\hat{B}^{-1}$ = (-π1,-π2,.....,-πm,1) donnera le vecteur π des multiplicateurs du simplexe,pour chaque variable hors-base on peut ainsi calculer les facteurs de coûts relatifs

$$\bar{c}j = cj - \pi\, aj = cj - \sum_{i=1}^{m} \pi i\, aij$$

2/ Selection de la colonne entrante : $\bar{c}s = \min_{\bar{c}j<0} \bar{c}j$

3/ Si $\bar{c}s \geq 0$,Stop :la solution est optimale,elle est donnée par les formules (4.2.3 et 4.2.4)

4/ Si $\bar{c}_s < 0$: calcul de la colonne transformée

$$\begin{bmatrix} \bar{a}_s \\ \cdot \\ \cdot \\ \cdot \end{bmatrix} = \hat{B}^{-1} \begin{bmatrix} a_s \\ \cdot \\ \cdot \\ \cdot \end{bmatrix} = \begin{bmatrix} \bar{a}_{1s} \\ \cdot \\ \cdot \\ \bar{a}_{ms} \\ \bar{c}_s \end{bmatrix}$$

5/ Si tous les $a_{ij} \geq 0$:Stop la solution optimale est infinie

6/ S'il existe $\bar{a}_{ij} > 0$, calculer $\bar{b}$ par 4.2.3 et 4.2.4
et utiliser la règle :

$$\frac{\bar{b}_r}{\bar{a}_{rs}} = \min_{a_{rs}>0} \frac{\bar{b}_i}{\bar{a}_{is}} = \theta$$

7/ Construire la matrice $\left[\begin{array}{c|c} \hat{B}^{-1} & \bar{a}_s \\ & \bar{c}_s \end{array}\right]$ et la transformer
à l'aide du pivotage autour de $\bar{a}_{rs}$
les (m+1) premières colonnes donneront alors la nouvelle base
8/ Ecrire la solution de base

$$(x_B)_i = (x_B)_i - \theta\, \bar{a}_{is} \quad i \text{ différent de } r$$
$$(x_B)_r = \theta$$

9/ Retour à 1 .

IV- RESOLUTION INFORMATIQUE DES PROGRAMMES LINEAIRES A L'AIDE DE LA METHODE REVISEE DU SIMPLEXE

Voir annexe H2
Remarque :
On constate que cette méthode suppose la connaissance préalable d'une base réalisable de départ.
Si l'obtention d'une telle base n'est pas immédiate ont peut alors avoir recours à la méthode des deux phases décrites précédement mais au lieu d'utiliser la méthode ordinaire dans les deux phases on utilisera la méthode revisée dans chacune des deux phases.

CHAPITRE VI

PRINCIPE DE DECOMPOSITION DE DANTZING-WOLFE

I- INTRODUCTION

L'algorithme du simplexe est très efficace dans les systemes relativement bas mais des que l'ordre de systeme devient trés élevé on se heurte à des difficultés de calculs, pour éviter ces difficultés dans certains cas on peut recourir à l'algorithme de décomposition de Dantzing et Wolfe qui permet de rendre tout problème de programmation linéaire en scindant l'ensemble des contraintes,c'est à dire en partitionnant horizontalement la matrice (A,b).

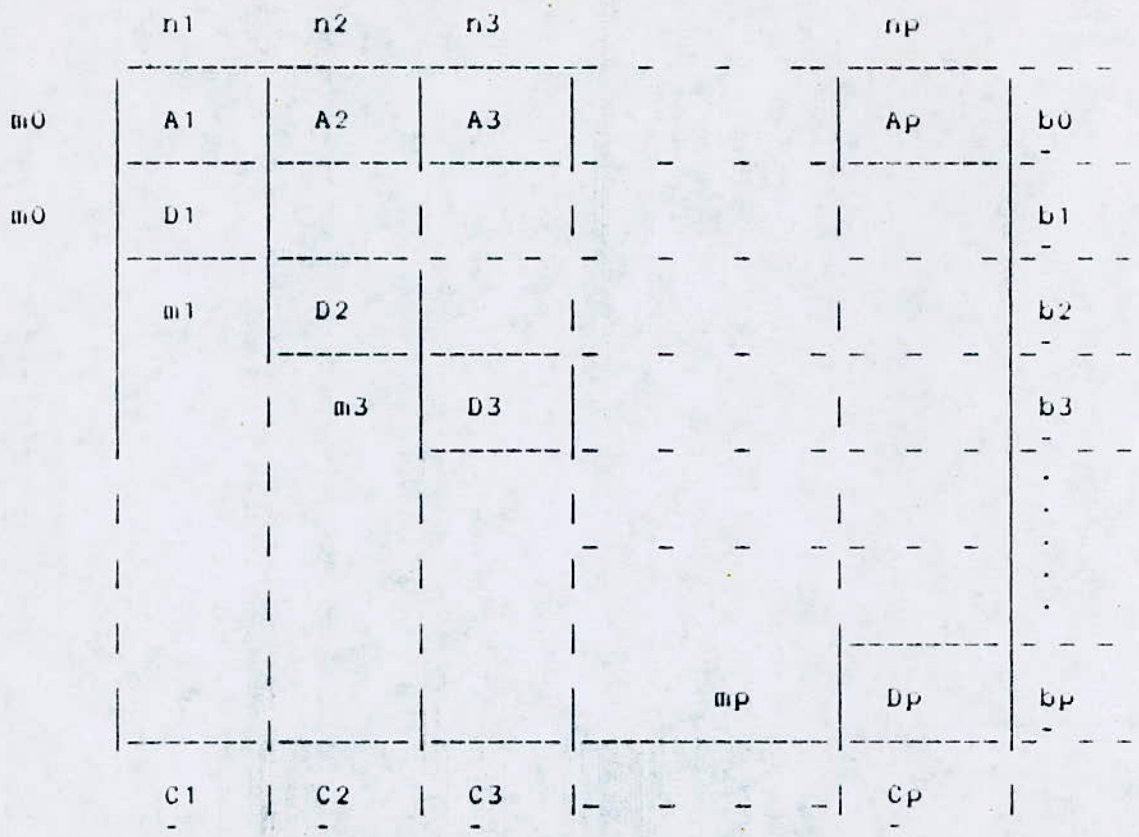
Ainsi cette méthode s'applique donc directement a tous les problèmes de programmation linéaire dans lesquels l'ensemble des contraintes peut être partitionné en (p+1) sous ensembles de façons que p sous ensembles constituent des systèmes mutuellement independants,c'est à dire fassent;chacun;intervenir des inconues differentes,les contraintes du (p+1)ème sous-ensemble étant des contraintes de liaisons affectant l'ensemble de toutes les variables.

La résolution du problèmes initial est alors remplacee par celles des (p+1) problèmes:les p problèmes indépendants et un problème de liaison (coordination) de taille en general tres inferieure à celle du probleme primitif.

La procedure est plus efficace lorsque les p problemes independants ont une structures simples.

II- FORMULATION DU PROBLEME ORIGINALE ET D'UN PROBLEME EQUIVALENT

La structure caractéristique des problemes que nous venons de decrire(et qui sont appelés problèmes angulaires lorsque p>1)est figurée par la matrice suivante dans laquelle seuls les blocs encadres peuvent contenir des élements non nuls.



La matrice de la liaison A composée des m0 premières lignes est partitionnée verticalement en p sous-matrices Aj (j=1.....p) ayant respectivement les mêmes colonnes que les matrices Dj des p problèmes indépendants.
Certaines des matrices Dj peuvent ne pas exister et la matrice A peut être partiellement creuse(c'est à dire à beaucoup d'éléments nuls) le vecteur demande $\vec{b}$ et le vecteur coût $\vec{C}$ sont partitionnés respectivement horizontalement et verticalement, comme la matrice D

Le probleme original est donc un probleme à $\sum_{j=0}^{p} m_j$ équations et $\sum_{j=1}^{p} n_j$ inconues.

Le problème linéaire s'écrit alors :

$$\text{Min } Z = \vec{C}_1 \vec{X}_1 + \vec{C}_2 \vec{X}_2 + \ldots + \vec{C}_p \vec{X}_p = \sum_{j=1}^{p} \vec{C}_j \vec{X}_j$$

$$A_1 \vec{X}_1 + A_2 \vec{X}_2 + \ldots + A_p \vec{X}_p = \sum_{j=1}^{p} A_j \vec{X}_j$$

($\vec{V}$: indique que V represente un vecteur)

$$
\begin{aligned}
&D1\ X1 && && = b1\\
&\quad - && && \quad -\\
&\quad D2\ X2 && && = b2\\
&\quad\ - && && \quad -\\
&\qquad\ \ddots && && \quad \vdots\\
&\qquad\qquad\ \ddots && && \quad \vdots\\
&\qquad\qquad\ \ddots && && \quad \vdots\\
&\qquad\qquad\ \ddots && && \quad \vdots\\
&\qquad\qquad\ \ddots && && \quad \vdots\\
&\qquad\qquad\ \ddots && && \quad \vdots\\
&\qquad\qquad\qquad Dp\ Xp && && = bp\\
&\qquad\qquad\qquad\ \ - && && \quad -
\end{aligned}
$$

D'où sous forme vectorielle :

$$
(I)\quad \left[\begin{array}{l}
\text{Min } Z = \sum_{j=1}^{p} \underline{Cj}\ \underline{Xj}^{T} \\
\\
\sum_{j=1}^{p} \underline{Aj}\ \underline{Xj} = b0 \qquad (1)\\
\\
\underline{Dj}\ \underline{Xj} = \underline{bj}\\
\\
\underline{Xj} \geq 0
\end{array}\right.
$$

où chacun des systèmes de contraintes :

$$
\left[\begin{array}{l}
\underline{Dj}\ \underline{Xj} = \underline{bj} \qquad (2)\\
\\
\underline{Xj} \geq 0
\end{array}\right.
$$

Definit un tronçon (annex A) Sj dans l'espace R^{nj} des solutions (ce tronçon est d'ailleur entierement situé dans la variété linéaire définie par les équations (2)).Nous supposon que tous les tronçons Sj sont bornes,c'est à dire sont,par définition des polyèdres convexes.

Soient $\underline{xj}^{k}$ $(k=1.....,Sj)$ les point extrémaux du polyedre convexe Sj.

Tout point xj de Sj peut être représenté comme une combinaison linéaire convexe des points extrémaux

ce qui implique que pour $\lambda j^{k} \geq 0$ et $\sum_{k=1}^{Sj} \lambda j^{k} = 1$

on peut ecrire $\underline{Xj} = \sum_{k=1}^{Sj} \lambda j^{k}\ \underline{xj}^{k}$, $j=1.....p$ (3)

Les relations (1) deviennent lorsqu'on y remplace X_j par l'expression (3) :

$$(II)\quad \begin{cases} \text{Min } Z = \sum_{j=1}^{p} \sum_{k=1}^{S_j} C_j \lambda_j^k \bar{X}_j^k \\ \sum_{j=1}^{p} \sum_{k=1}^{S_j} A_j \lambda_j^k \bar{X}_j^k = b_0 \\ \sum_{k=1}^{S_j} \lambda_j^k = 1 \; ; \; \lambda_j^k \geq 0 \end{cases} \qquad (4)$$

$j = 1 \ldots\ldots p$; $k = 1 \ldots\ldots S_j$

Les relations (4) définissant un nouveau problème linéaire (II) par rapport aux variables λ_j^k (appelé programme maître); celui-ci est équivalent au problème (I) lorsqu'on lui adjoint les relations (3) de définition des X_j à partir des λ_j^k $(j=1.....p)$. En effet à toute solution optimale $\bar{X}_j$ de (I) correspond une solution optimale de (II) définie par les valeurs λ_j^{*k} des coefficients des combinaisons linéaires convexes (3) et réciproquement.

En passant de (I) à (II), on fait décroître le nombre d'equations de $\sum_{j=}^{p} m_j$ à $m_0 + p$ au détriment d'une augmentation du nombre des variables:

de $\sum_{j=1}^{p} n_j$ à $\sum_{j=1}^{p} S_j$

Le bilan peut être considéré comme positif, puisque le nombre des variables n'influe qu'assez peu sur le volume des calculs.

Les quantités $\bar{X}_j^k$, points extremaux, doivent être connues pour pouvoir résoudre le programme maître, nous montrerons ultérieurement comment les obtenir, maintenant, considérons ces quantités comme données, le programme maître pourra alors être résolu à l'aide de la méthode révisée du simplexe Soit B la base correspondante au programme d'une iteration précédente et C_{-B} le vecteur coût relatif aux varaibles de base; la solution en cours serait optimale pour tout vecteur P_j^k hors-base telle que: $Z_j^k - C_j^k = C_{-B} B^{-1} P_j^k - C_j^k$ soit positif ou nul.

Or $C_j^k = C_j - C_j X_j$ (d'après la définition de Π)

et $P_j^k =$

$$\begin{bmatrix} r_0 \text{ lignes} & \hat{A}_j^k \hat{X}_j \\ & 0 \\ & \vdots \\ & 1 \leftarrow (r_0 + j)\text{ème position} \\ & \vdots \\ & 0 \end{bmatrix}$$

Posons $B^{-1} = [\, R_0 \mid V_1\ V_2 \ldots V_n \,]$

où R_0 est $(r_0 + n) \times r_0$ -matrice composée des r_0 premières colonnes de B^{-1}, et V_j est la $(r_0 + j)$ème colonne de B^{-1}

Donc $Z_j^k - C_j^k = C_B [\, R_0 \mid V_1 \ldots V_n \,] \hat{P}_j^k - C_j \hat{X}_J^k$

$$= C_B [\, R_0 \hat{A}_j \hat{X}_j^k + V_j \,] - C_j \hat{X}_j^k$$

$$= [\, C_B R_0 A_j - C_j \,] \hat{X}_j^k + C_B V_j \qquad (5)$$

Si la solution en cours n'est pas optimale (il existe $Z_j^k - C_j^k < 0$) alors on applique le critère d'entrée de Dantzing afin de sélectionner le vecteur P_j^k entrant en base ce qui équivaut à chercher $\min (Z_j^k - C_j^k) = \delta_j$

Le calcul de $Z_j^k - C_j^k$ suppose la connaissance du vecteur P_j^k donc des points extremaux $\hat{X}_j^k$ d'où pour faciliter la tâche et ne pas être amené à déterminer tous les points extremaux pour toutes les itérations on calculera $\hat{X}_j^k$ à chaque itération pour k donné ainsi le calcul de $\delta_j = Z_j^k - C_j^k$ pour $j = 1 \ldots p$ sera immédiat, on déterminera ensuite $\delta = \min_j (\delta_j)$

- Si $\delta < 0$, la variable λ_j^k correspondant à δ est sélectionnée pour rentrer en base

- Si $\Gamma > 0$, la solution optimale est atteinte.

Revenons à la détermination des points extrémaux $\hat{X}_j^k$: ces points sont obtenus en résolvant le programme linéaire suivant:

$$\begin{cases} \min Z' = -Z_j + C_j \\ D_j X_j = b_j \\ X_j \geq 0 \end{cases}$$

or (5) entraine que la minimisation de Z' equivaut à la minimisation de:

$$w_j = [\, C_{-B} R^0 A_j - C_j \,] X_j$$ (car $C_{-B} V_j$ est constant et independant de k)

associée à

$$\begin{cases} D_j X_j = b_j \\ X_j \geq 0 \end{cases} \quad \text{(III)}$$

D'ou $\Gamma_j = w_j + C_{-B} V_j$ avec $w_j^* = [C_{-B} R^0 A_j - C_j] \hat{x}_j^k$ valeur optimale de w_j

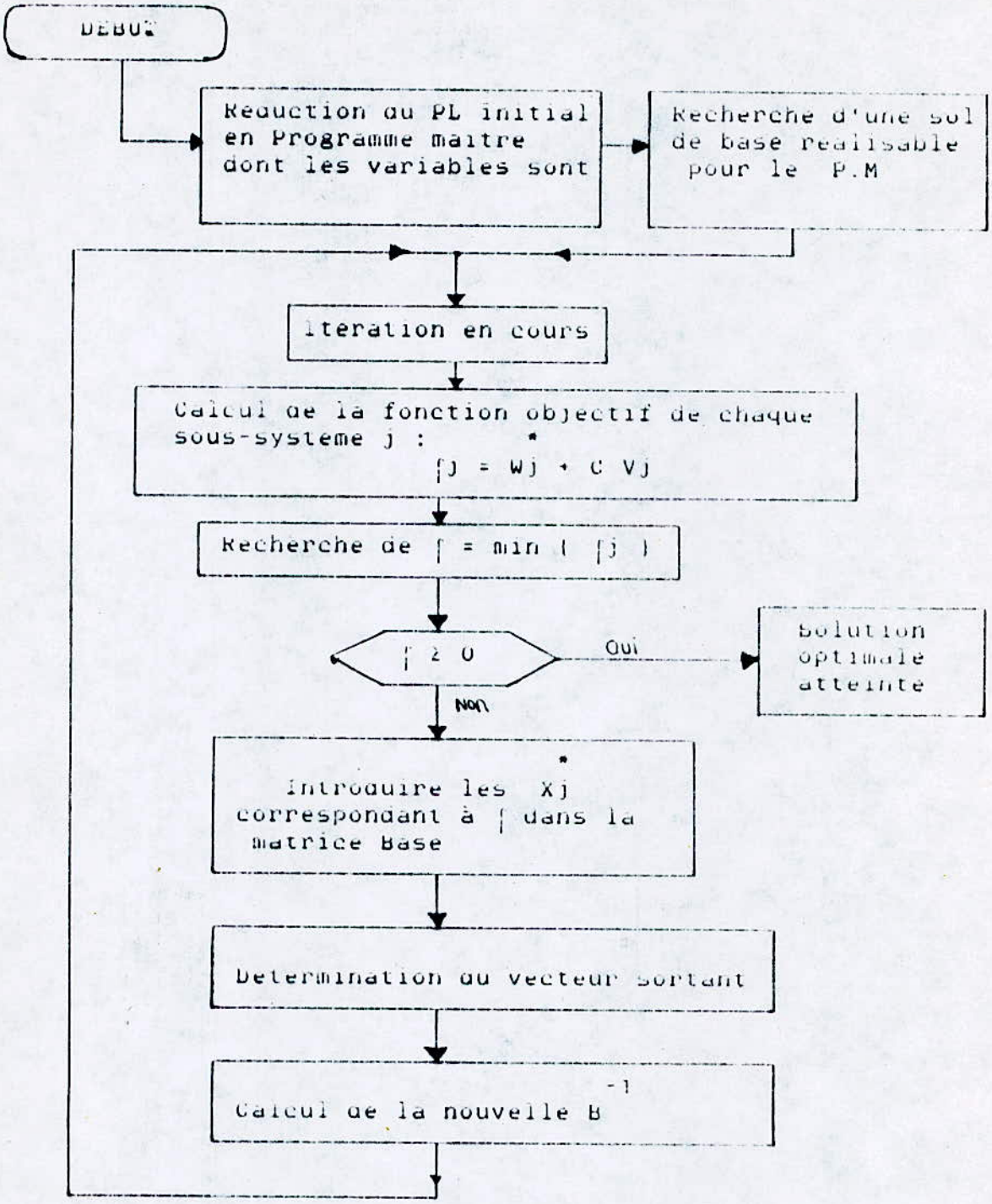
La détermination du vecteur sortant se fera à l'aide du critère de sortie de Dantzing.

Une fois que $\Gamma = \min_j (\Gamma_j) > 0$ la solution optimale du programme optimale globale est donnée par :

$$x_j^* = \sum_{j=1}^{Sj} \cdot \lambda_j^{*k} \, x_j^{*k} \quad j=1,2,.....p$$

ou λ_j^{*k} est la solution optimale du programme maitre et x_j^{*k} est le point extreme correspondant déduit de la résolution du programme locale (III).

ORGANIGRAMME DE LA METHODE DE DANTZING-WOLFE



2eme PARTIE

OPTIMISATION HIERARCHIQUE ET CONTROLE DES SYSTEMES LINEAIRES AVEC FONCTION OBJECTIF QUADRATIQUE

CHAPITRE I :

PROBLEME DE L'OPTIMISATION DYNAMIQUE

Considérons un système dynamique linéaire décrit par les équations d'état suivantes:

$$\dot{X}(t) = F(X(t),U(t),t)$$
$$X(t0) = X0 \qquad (I-1)$$

où X est un n-vecteur d'état, U un m-vecteur de commande et F un vecteur de fonctions non linéaires continues et dérivables;l'état initial X(t0) est supposé connu.
On veut choisir un vecteur U(t) ($t0 \leq t \leq tf$),où tf est le temps final,il peut être libre ou fixe,afin d'avoir un comportement désirable de nôtre systeme.

$$\text{Soit } J = h(X(tf),tf) + \int_{t0}^{tf} g(X(t),U(t),t)\,dt \qquad (I-2)$$

La fonction objectif du système dynamique (par convention on ne considérera que les problèmes de minimisation)ou h et g sont généralement des fonctions scalaires non linéaires.

Interprétation de h et g

* h(X(tf),tf):assure q'au temps final,notre systeme tendra vers un certain état désirable.

* la forme intégrale: assure que, sur l'intervalle d'optimisation, nous n'utiliserons pas un effort excessif de cônTrole et qu'on ne dévie pas beaucoup de la trajectoire désirable.

Le problème d'optimisation dynamique consiste à trouver un vecteur de commande U* admissible faisant suivre au système une trajectoire admissible X* qui minimise le critère J donné par la relation (II-2).

CHAPITRE II:

TECHNIQUES VARIATIONNELLES [5]

Dans ce chapitre nous presentons la methode classique de résolution des problèmes dynamiques, le principe du maximum dans les cas continus et discret.

II-1- PRINCIPE DU MAXIMUM (cas continu)

Etant donné le problème dynamique decrit par les equations (I-1)et(I-2), nous définissons la fonction HAMILTONIEN H:

$$H(X(t),U(t),\lambda(t),t) = g(X(t),U(t),t) + \lambda^{T}[F(X(t),U(t),t)] \quad ; (II-1)$$

ou $\lambda(t)$ est un n-vecteur de multiplicateurs de Lagrange[5]

II-1-1-CONDITONS NECESSAIRES D'OPTIMALITE[5]

$X^*(t)$, $U^*(t)$et $\lambda^*(t)$ sont les solutions optimales du problème dynamique si et seulement si:

$$\left.\begin{aligned} \dot{X}^*(t) &= \frac{\partial H}{\partial \lambda}(X^*(t),U^*(t),\lambda^*(t),t) \\ \dot{\lambda}(t) &= -\frac{\partial H}{\partial X}(X^*(t),U^*(t),\lambda^*(t),t) \\ 0 &= \frac{\partial H}{\partial U}(X^*(t),U^*(t),\lambda^*(t),t) \end{aligned}\right\} \quad t\in[t0,tf]\,;(II-2)$$

et

$$\left[\frac{\partial h}{\partial X}(X^*(tf),tf) - \lambda^*(tf)\right]\delta Xf + \left[H(X^*(tf),U^*(tf),\lambda^*(tf),tf) + \frac{\partial h}{\partial t}(X^*(tf),tf)\right]\delta tf = 0 \quad ; (II-3)$$

δXf, δtf etant respectivement des petites variations de l'etat et du temps finaux.

II-1-2-CONDITIONS AUX LIMITES

Dans les problèmes particuliers nous substituerons les etats et temps limites dans l'equation (II-3), ce qui donnera les conditions aux limites.

Nous donnons ici deux types de problemes aux limites:

a) problèmes à temps final fixe(cas tres frequent)

-tf etant spécifié,l'état X(tf) peut être fixe,libre ou variant dans une certaine région de l'espace d'état.

a-1) X(tf) spécifié entraine $\delta tf=0$ et $\delta X(tf)=0$ dans (II-3) ce.qui donne:

$$X(tf) = Xf$$

a-2) si X(tf) est libre,alors seul δtf qui est nul dans (II-3) on aura donc:

$$\frac{\partial h}{\partial X}(X^*(tf)) - \lambda^*(tf) = 0$$

a-3) si l'état final varie dans une région définie par $Q(X(t)=0$, Q etant un K-vecteur où chaque composante de Q représente une hypersurface dans l'espace d'état à n dimensions, alors l'équation relatives aux conditions aux limites s'écrira [5]:

$$\frac{\partial h}{\partial X}(X^*(tf)) - \lambda^*(tf) = \sum_{i=1}^{k} di \left[\frac{\partial qi}{\partial X}(X^*(tf))\right]$$

Pour déterminer les 2n constantes d'intégration dans la solution des équations d'état ainsi que les di on utilisera les n équations $X^*(t0)= X0$ et les équations $Q(X^*(tf))=0$.

b) problèmes à temps final libre (δtf non nul)

b-1) l'état final X(tf) est fixe:$\delta Xf=0$
l'équation (II-3) donne dans ce cas:

$$H(X^*(tf),U^*(tf),\lambda^*(tf),tf) + \frac{\partial h}{\partial t}(X^*(tf),tf) = 0$$

b-2) l'état final est libre,alors δtf et δXf sont arbitraires et indépendants d'où leurs coéfficients sont nuls dans (II-3):

$$\lambda^*(tf) = \frac{\partial h}{\partial X}(X^*(tf),tf) \quad ; \text{(n équations)}$$

et $$H(X^*(tf),U^*(tf),\lambda^*(tf),tf) + \frac{\partial h}{\partial t}(X^*(tf),tf) = 0$$

de manière similaire,on peut développer,pour d'autres types de problèmes des conditions aux limites par substitution dans l'équation (II-3).

II-2-PROBLEMES QUADRATIQUES LINEAIRES

Malgré qu'une description réelle des systèmes dynamiques nécessite généralement l'utilisation d'équations différentielles non linéaires, il est possible d'obtenir avec une bonne approximation le comportement dynamique de tels systèmes en linéarisant les équations non linéaires autour d'une trajectoire convenable, ainsi beaucoup de systèmes peuvent être décrit par les équations d'état:

$$\dot{X}(t) = A(t)\,X(t) + B(t)\,U(t) \quad ; \qquad (II\text{-}2\text{-}1)$$

$$X(t0) = X0 \quad ;$$

où A et B sont respectivement des (nxn) et (nxm) matrices.

La fonction critère à minimiser est une fonction quadratique:

$$J=(1/2)\,\|X(tf)\|^2 s(tf)+(1/2)\int_{t0}^{tf} [\,\|X(t)\|^2 q(t)+\|U(t)\|^2 r(t)\,]dt$$

avec $\|b\|^2 c = b\,c\,b$; (II-2-2)

où q,s:matrices réelles symétriques semi définies positives
r:matrice réelle symétrique définie positive.

Nous supposons que les états et les commandes ne sont pas bornés et que X(tf) est libre.

-Interprétation physique:

- Cette forme du critère signifie qu'on désire maintenir le vecteur d'état près de l'origine des espaces sans utiliser un effort de contrôle excessif.
- Les matrices de pondération q,r et s permettent de définir les efforts de contrôle fournis ainsi que de mettre les différents états au voisinage de l'origine.

La résolution de ce problème par la méthode du principe du maximum conduit [5] à résoudre des équations différentielles de Ricatti:

$$\dot{P}(t) = -P(t)\,A(t) - A(t)\,P(t) + P(t)\,B(t)\,r(t)^{-1}\,B()\,P(t) - q(t) \qquad (II\text{-}2\text{-}3)$$

avec P(tf) = s ;

où P : matrice (nxn);

et ce pour trouver la trajectoire de commande :[5]

$$U(t) = r(t)\,B(t)\,P(t) \qquad (II\text{-}2\text{-}4)$$

L'équation (II-2-3) est dite équation matricielle de RICATTI.

REMARQUE:Le problème décrit ci-dessus s'appelle problème de régulation, il consiste à trouver les commandes qui forçent les états à l'origine; mais dans beaucoup d'applications on voudrait que les sorties de nôtre système suivent certaines trajectoires désirables données, en plus le dit système est soumis à des perturbations .

On considère donc le problème de servomécanisme linéaire suivant:

Soit à minimiser :

$$J=(1/2)||\mu(tf)-Y(tf)||^2s+(1/2)\int_{t0}^{tf}[||\mu(t)-Y(t)||^2q+||U(t)||^2r(t)]dt$$

où Y sont les sorties du système tandis que μ sont les trajectoires de sortie désirables ;
le système dynamique s'écrira alors :

$$\dot{X}(t) = A\,X(t) + B\,U(t) + W \;; \quad X(t0)=X0$$
$$Y(t) = C\,X(t)$$

où W est une perturbation à l'entrée(supposé connue);
comme précédemment la résolution de ce système par le principe du maximum conduit à n équations de Ricatti [5].

II-3- PRINCIPE DU MAXIMUM DANS LE CAS DISCRET

Beaucoup de systèmes particulièrement ceux économiques sont naturellement représentés comme des processus discrets; pour de tels systèmes nous pouvons utiliser la version discrète du principe du maximum.

Soit un système dynamique discret non linéaire,avec vecteur d'état X(k) et U(k) celui de commande à l'instant k.

L'état du système à l'instant k+1 est relié au précédent k par la relation:

$$X(k+1) = F(Xk,Uk,k) \quad ; (II-3-1)$$

où F est une fonction continue doublement différentiable.

Nous souhaitons minimiser le critère :

$$J = [g(Xk,k)]_{k0}^{kf} + \sum_{k=k0}^{kf-1} h(Xk,Uk,k) \;;(II-3-2)$$

sous les contraintes (II-3-1).

II-3-1-CONDITIONS NECESSAIRES D'OPTIMALITE

Comme dans le cas continu nous définissons le Hamiltonien discret:

$$H(Xk,Uk,\ k+1,k) = Hk = h(Xk,Uk,k) + \lambda k+1\ F(Xk,Uk,k);(II-3-3)$$

où λk sont les multiplicateurs de Lagrange.

La trajectoire optimale doit satisfaire les conditions suivantes:

$$\begin{cases} \dfrac{\partial Hk}{\partial \lambda k+1} = X(k+1) \\ \dfrac{\partial Hk}{\partial Uk} = 0 \text{; ou } \dfrac{\partial hk}{\partial Uk} + \left[\dfrac{\partial F^T}{\partial Uk}\right] \lambda k+1 = 0 \\ \lambda k = \dfrac{Hk}{Xk} \text{ ; ou } \lambda k = \left[\dfrac{\partial F}{\partial Xk}\right]^T \lambda k+1 + \dfrac{\partial hk}{\partial Xk} \end{cases}$$

$$\longrightarrow \lambda k+1 = \left[\frac{\partial F^T}{\partial Xk}\right]^{-1} \left[\lambda k - \frac{\partial Hk}{\partial Xk}\right];$$

(si $\left[\dfrac{\partial F}{\partial Xk}\right]$ est une matrice régulière)

$$\begin{cases} \mu k0^T \left[\lambda k0 - \left[\dfrac{\partial gk0}{\partial Xk0}\right]\right] = 0 \\ \mu kf^T \left[\lambda kf - \left[\dfrac{\partial gkf}{\partial Xkf}\right]\right] = 0 \end{cases}$$

où μk sont des perturbations à différents instants k.

Remarque: ces équations ne sont applicables que s'il n'y a pas de contraintes, sur les états et les commandes, de type inégalités sinon on aura des difficultés dûes au fait que, contrairement au cas continu où une variation importante de la commande U(t) n'entraine pas une forte perturbation, dans le cas discret une variation importante de Uk provoquera une perturbation impotante.

II-4-PROBLEME QUADRATIQUE-LINEAIRE DISCRET

Considérons le problème du régulateur suivant:

$$MIN(J=(1/2)||X(kf)||^2 s(kf) + (1/2)\sum_{k0}^{kf-1} [||X(k)||^2 q(k) + ||U(k)||^2 r(k)])$$

ou q et s sont des matrices définies non négatives[annexeE]
r est une matrice définie positive ;
avec les contraintes dynamiques linéaires :

$$X(k+1) = A(k)\, X(k) + B(k)\, U(k)$$

Pour résoudre ce probleme, ecrivons son Hamiltonien :

$$H = (1/2)\,||X(k)||^2 q(k) + (1/2)\,||U(k)||^2 r(k) + \lambda(k+1)\,[A(k)\,X(k) + B(k)\,U(k)]$$

On applique alors le principe du maximum discret ce qui conduira finalement à la résolution d'une equation matricielle de RICATTI [5]:

$$P(k) = q(k) + A^{T}(k)[I + B(k)\,r^{-1}(k)\,B^{T}(k)\,P(k+1)]^{-1}\,A(k)$$

avec P(kf)= s (conditions aux limites)
qui donnera finalement le vecteur de commande:

$$U(k) = -\,r^{-1}(k)\,B^{t}(k)\,A^{-1}(k)\,[P(k) - q(k)]\,X(k)$$

CHAPITRE III:

OPTIMISATION HIERARCHIQUE ET CONTROLE AVEC FONCTION OBJECTIF QUADRATIQUE

III-1-INTRODUCTION

Comme nous l'avons montré, la résolution des problèmes quadratiques linéaires continus ou discrets conduit à l'intégration d'équations différentielles appropriées de type Ricatti.

L'equation Ricatti d'un système d'ordre n est une équation matricielle comprenant [nx(n+1)/2] equations differentielles non linéaires ou équations aux differences (selon le cas discret ou continu);donc la résolution d'un tel système demande un nombre de calculs important augmentant avec l'ordre du système.

De plus,l'intégration d'équations d'ordre élevé entraine beaucoup d'erreues d'arrondis(sur ordinateur),ce qui risque de distabiliser nôtre calcul numérique.

Ainsi les méthodes variationnelles décrites au chapitre II s'averent limitées d'où la nécessité d'utiliser dans les problèmes multidimensionnels et complexe les techniques du contrôle hiérarchique.

II-1-1-PROBLEME

Soit à contrôler de maniere optimale un systeme dynamique complexe,subdivisons ce système en N sous systemes interconnétes de la façon montrée sur la Fig.1;
pour chaque sous système i, Xi est un vecteur d'etat de dimension ni, Ui est un mi-vecteur de commande et Zi est un ri-vecteur d'entrée,lesquelles entrées sont générées par les états des autres sous systèmes.

On suppose que les sous systèmes eux mêmes sont decrits par des équations différentielles linéaires :

$$\dot{X}_i(t) = A_i X_i(t) + B_i U_i(t) + C_i Z_i(t) \qquad \text{(III-1)}$$

avec $X_i(0) = X_{i0}$

on suppose également que le vecteur des entrées Zi est une combinaison linéaire des états des N sous systemes:

$$Z_i = \sum_{j=1}^{N} L_{ij} X_j \qquad \text{(III-2)}$$

Il s'agit alors de choisir les commandes U1,U2,..,Un minimisant(par convention) les fonctions objectifs de type

$$J=\sum_{i=1}^{N}\left[(1/2)\,||Xi(t)||^2 fi+\int_0^T (1/2)\left[||XI(t)||^2 qi+||Ui(t)||^2 ri+\right.\right.$$
$$\left.\left.+||Zi(t)||^2 si\right]\right]$$

soumises aux contraintes (III-1) et (III-2)

Ou

qi,fi:des matrices semi definies positives
ri,si:des matrices définies positives

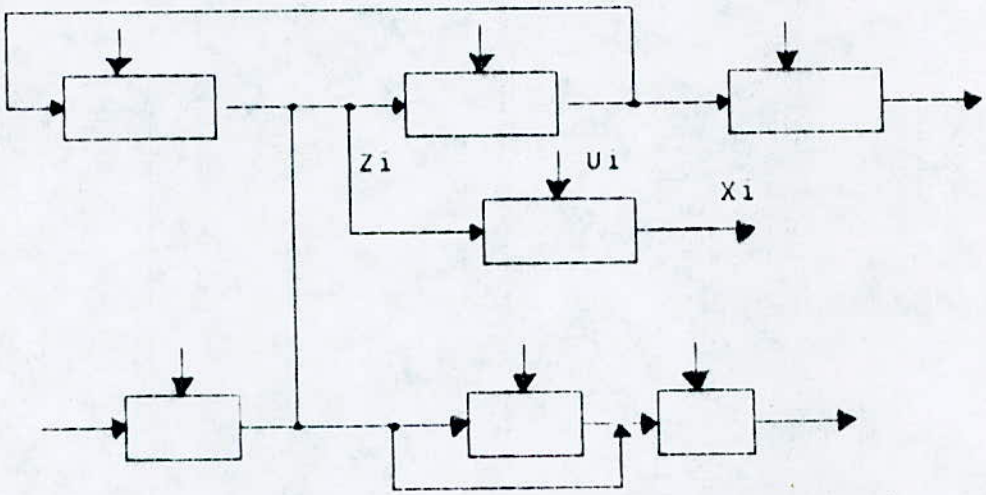


Fig.1 exemple de système interconnecté

Si on substitue Zi donné par (III-2) dans (III-1),on obtient la forme standard de l'équation d'état:

$$\dot{X} = A X + B U \quad ; \quad \text{où} \quad X=[X1\ldots Xn]^t \ , \ U=[U1\ldots Un]^t$$

III-2-APPROCHE DE COORDINATION DU BUT

III-2-1-FORMULATION

Cette approche est basée sur la possibilité de conversion du probleme de minimisation original en un probleme de maximisation plus simple ,et sa résolution par une sructure iterative à deux niveaux.

Pour cela on définit une fonction duale $\varphi(\lambda)$:

$$\varphi(\lambda)=\underset{X,U,Z}{MIN}\ [\ L(X,U,Z,\lambda)\ \text{avec les contraintes III-1}\ ]\ ;(III-4)$$

ou

$$L(X,U,Z,\lambda)=\sum_{i=1}^{N}\{(1/2)||Xi(T)||^2 fi+\int_0^T[(1/2)||Xi||^2 qi+(1/2)||Ui||^2 ri+(1/2)||Zi||^2 si+\lambda^T i(Zi-\sum_{j=1} Lij\ Xj)]dt\} \quad (III-5)$$

avec λ : vecteur des multiplicateurs de Lagrange

L :Lagrangien formé par l'introduction des multiplicateurs λ .

Et comme les contraintes de notre probleme sont lineaires et sa fonction critère est quadratique(ie: le probleme est convexe),le théoreme de dualité de dualité de Lagrange[5] nous permet d'écrire:

$$MAX\ \varphi(\lambda) = MIN_U\ J \quad (III-6)$$

la resolution de ce problème de maximisation est relativement aisée par les méthodes hiérarchiques.

Pour $\lambda = \lambda^*$ donnée
on a:

$$L(X,U,Z,\lambda^*)=\sum_{i=1}^{N}[(1/2)(||Xi||^2 qi+||Ui||^2 ri+||Zi||^2 si)+\lambda^{*T} i\ Zi-\sum_{j=1}^{N}\lambda^{*T} j\ Lij\ Xi]$$

ie: le Lagrangien L est additivement séparable(pour cette valeur λ^*),et peut donc être décomposé en N sous Lagrangiens indépendants ;ainsi il est possible de minimiser,pour un λ^* donné , chaque sous Lagrangien (sous les contraintes (III-1):

$$Li=(1/2)||Xi(T)||^2 fi+\int_0^T[(1/2)(||Xi||^2 qi+||Ui||^2 ri+||Zi||^2 si)+\lambda^{*T} i\ Zi-\sum_{j=1}^{N}\lambda^{*T} j\ Lji\ Xi]dt \quad (III-7)$$

independemment des autres sous systèmes,ce qui permettra d'obtenir $\varphi(\lambda^*)$ de l'équation (III-4).

La minimisation des sous Lagrangiens constitue le 1er niveau de la sructure hiérarchique,le second niveau alors à améliorer successivement $\varphi(\lambda^*)$ en échangeant les informations ,de manière itérative ,avec le 1er niveau.

Cette échange sera basé sur le calcul,à chaque itération, de l'erreur résidant dans les relations d'interconnection, donnée par le gradient:

$$\nabla\varphi(\lambda)\Big|_{\lambda=\lambda^*}=\begin{bmatrix}\vdots\\ Zi-\sum_{j=1}^{N} Lij\ Xj\\ \vdots\end{bmatrix}=\begin{bmatrix}\vdots\\ Ei\\ \vdots\end{bmatrix}=E \quad ;(III-8)$$

Donc il est possible d'envisager un algorithme à 2 niveaux comme le montre la Fig.2 où au 1er niveau s'effectue la résolution des équations (III-7),les résultats seront alors envoyés au niveau 2 où s'effectuera le calcul de l'erreur.

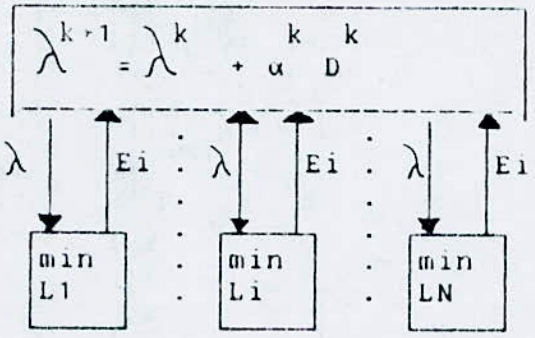


Fig.2 structure de coordination du but à 2 niveaux

Cette erreur sera utilisée dans cette procedure de gradient[Annexe D] pour produire un nouveau λ ;
par exemple : pour passer de l'itération k à la (k+1)eme:

$$\lambda^{k+1}(t) = \lambda^{k}(t) + \alpha^{k} D^{k}(t) \quad ; (0 \le t \le T) \qquad (III-9)$$

où α est le pas et D la direction de recherche.
Si l'on utilise la methode du gradient,on prend alors

$$D^{k}(t) = E^{k}(t) \quad ; \quad (0 \le t \le T)$$

si la methode du gradient qui est utilisee on aura:

$$D^{k+1}(t) = E^{k+1}(t) + \beta^{k+1} D^{k} \quad ; \quad (0 \le t \le T) \qquad (III-10)$$

$$\text{ou} \quad \beta^{k+1} = \int_0^T [(E^{k+1t} . E^{k+1}) dt] / [(E^{kt} . E^{k}) dt];$$

avec $D^{0} = E^{0}$

L'optimum sera atteint(arrêt des iterations) quand:
E^{k} ;0≤t≤T; est suffisemment proche de zero.

III-2-2-COMMENTAIRES

Quoique cette méthode facilite l'etude des systemes d'ordre elevé, deux inconvénients principaux font qu'elle n'est pas tres utilisée :

a) chaque iteration de la méthode du gradient nécessite la determination du pas optimal α car un pas constant nuirait à la convergence de la méthode (du point de vue rapidité).

Puisqu'au début de la procedure du gradient, on est loin de l'optimum, il faut alors choisir un pas important tandis qu'aux environs de l'optimum un pas faible serait convenable; cette recherche du pas augmentrait le temps de calcul et la capacité mémoire requise.

b) l'introduction du terme quadratique $||Zi||^2 si$ qui n'a d'ailleurs aucun sens physique et dont l'aventage serait d'eviter les solutions singulières, augmente encore plus le nombre de calculs et conduit aux mêmes inconvénients que ci-dessus.

Remarque: pour voir comment ce terme quadratique releve les singularités; considerons un sous système i , et minimisons son Lagrangien Li(III-7) sous les contraintes (III-1)
le Hamiltonien s'écrit:

$$Hi = (1/2)(||Xi||^2 qi + ||Ui||^2 ri + ||Zi||^2 si) + \lambda_i^T Zi - \sum_{j=1}^{N} \lambda_j Lij Xi +$$
$$+ pi (Ai Xi + Bi Ui + Ci Zi)$$

$$\frac{\partial Hi}{\partial Zi} = 0 \longrightarrow Zi = - si^{-1} [Ci^t pi + \lambda_i]$$

or si $||Zi||^2 si$ n'existait pas on aurait :

$$\frac{\partial Hi}{\partial Zi} = \lambda_i^T + pi\, Ci^t = 0 \longrightarrow \text{singularités (car } \lambda_i, pi, Ci \text{ sont données)}$$

Et comme l'introduction d'un terme lineaire ne leverait pas les singularités, on a alors opté à l'ajout d'un terme quadratique à la fonction objectif.

Tout ce qui precede montre que l'approche de coordination du but n'est pas tres pratique pour les systemes d'ordre tres élevé , vue le nombre de calculations et la capacité mémoire · importants qu'elle demande.
Pour alléger les éxigences de la résolution de tels problemes [temps de calcul et encombrement mémoire] TAMUKA a proposé une modification assez interessante de la méthode de coordination du but.

CHAPITRE IV:

LA METHODE A TROIS NIVEAUX DE TAMURA

Avant d'aborder ce chapitre nous developpons d'abord la methode de coordination du but en temps discrets.

VI-CAS DISCRET DE LA METHODE DE COORDINATION DU BUT

Soit à minimiser le critère :

$$J=\sum_{i=1}^{N}\Big[(1/2)||Xi(K)||^2 fi+\sum_{k=0}^{K-1}(1/2)(||Xi(k)||^2 qi+||Ui||^2 ri+||Zi||^2 si)\Big] \quad ; (IV-1)$$

sous les contraintes :

$$Xi(k+1)=Ai\,Xi(k)+Bi\,Ui(k)+Ci\,Zi(k) \quad ;(IV-2)$$

$$i=1,2...,N \quad ; \quad k=0,1...K-1$$

$$Xi(0)=Xi0 \quad ; \quad (IV-3)$$

comme dans le cas continu Zi est le vecteur des entrées d'interactions venant des autres sous systemes:

$$Zi(k)=\sum_{j=1}^{N} Lij\,Xj(k);\ k=0,1,...k-1\ ,i=1,2,...N \quad (IV-4)$$

pour résoudre ce problème il est necessaire, comme dans le cas continu , de maximiser la fonction duale :

$$\Phi(\lambda)=\text{MIN}\ L(X,U,Z,\lambda) \quad (IV-5)$$

sous les contraintes (V-2) et (V-3)

avec

$$L(X,U,Z,\lambda)=\sum_{i=1}^{N}\Big[(1/2)||Xi(K)||^2 fi+\sum_{k=0}^{K-1}(1/2)||Xi(k)||^2 qi+(1/2)||Ui(k)||^2 ri+(1/2)||Zi||^2 si+\lambda_i^T Zi-\sum_{j=1}^{N}\lambda_j^T Lji\,Xi(k)\Big]=\Sigma Li$$

(IV-6)

où

$$Li=(1/2)||Xi(K)||^2 fi+\sum_{k=0}^{K-1}(1/2)(||Xi(k)||^2 qi+||Ui(k)||^2 ri+||Zi(k)||^2 si)+\lambda_i^T Zi-\sum_{j=1}^{N}\lambda_j^T Lji\,Xi(k) \quad ;(IV-7)$$

Ainsi que dans le cas continu, il est possible de séparer, pour un donné le problème initial en N problèmes de minimisation des sous Lagrangiens indépendants Li; chacun des Li étant soumis aux contraintes (IV-2) et (IV-3).

Le vecteur des multiplicateurs de Lagrange peut etre amélioré au niveau 2 par la méthode du gradient comme dans le cas continu:

$$\nabla \varphi(\lambda)\Big|_{\lambda=\lambda^*} = \left[\begin{array}{c} \vdots \\ Zi(k) - \sum_{j=1}^{N} Lij\ Xj(k) \\ \vdots \end{array}\right] \quad \begin{array}{l} (IV\text{-}8) \\ ;i=1..N \\ k=0,1..K-1 \end{array}$$

IV-2-MODIFICATION DE TAMURA

On sait que pour $\lambda^*(k)$ donné ($k=0,\ldots K-1$), le problème du 1er niveau (minimisation de Li sous les contraintes (IV-2) et (IV-3)) peut lui même être traité par dualité et décomposition mais au lieu de diviser le lagrangien Li de chaque sous système, on le décompose suivant les instants k créant ainsi un autre niveau supplémentaire qui rendra les calculs plus simples.

Pour cela on définit le problème dual de minimisation de Li de (IV-7) sous les contraintes (IV-2) tel que:

$$\underset{p}{MAX}\ M(p)$$

ou

$$M(p)=\underset{X,U}{MIN}\Big[(1/2)||Xi(K)||^2 fi+\sum_{k=0}^{K-1}\Big[(1/2)(||Xi(k)||^2 qi+||Ui(k)||^2 ri+$$

$$+||Zi(k)||^2 si+pi(k)[Ai\ Xi(k)+Bi\ Ui(k)+Ci\ Zi(k)-Xi(k+1)]+$$

$$+\lambda i^{*T} Zi - \sum_{j=1}^{N} \lambda j^{*T} Lji\ Xi\Big] \qquad (IV\text{-}9)$$

avec les contraintes initiales : $Xi(0) = Xi0$

la résolution numérique du problème dual repose sur le calcul de la valeur de la fonction duale M(p) pour $p = p^*$ donné et par la suite la maximisation de M(p) par une technique du gradient.

Le gradient de M(p) s'écrit:

$$\nabla M(p)\Big|_{p=p^*} = -Xi(k+1)+Ai\ Xi(k)+Bi\ Ui(k)+Ci\ Zi(k); \quad (IV\text{-}10)$$

ou Xi, Ui sont les solutions des problèmes de minimisation des Li sous les contraintes (IV-2).

Pour calculer M(p),on minimisera cette fonction indépendemment pour chaque instant k(pour p^*et λ^* données) comme suit:

ecrivons le Hamiltonien du sous système i:

$$Hi(Xi(k),Ui(k),Zi(k),k)= (1/2)(||Xi(k)||^2 qi + ||Ui(k)||^2 ri +$$

$$+\frac{1}{2}||Zi(k)||^2 si) + \lambda i^{*T} Zi - \sum_{j=1}^{N} \lambda j^{*T} Lji\, Xj + pi^{*t}(k)\,[Ai\, Xi(k) +$$

$$+ Bi\, Ui(k) + Ci\, Zi(k)]; \quad k=0,1..K-1; i=1..N \qquad (IV-11)$$

en substituant Hi dans (IV-9) ,on aura :

$$M(p)=(1/2)||Xi(K)||^2 fi - pi^{*t}(K-1)\, Xi(k) +$$

$$+ \sum_{k=0}^{K-1} [\, Hi(Xi(k),Ui(k),Zi(k),k) - pi^{*t}(k-1)\, Xi(k)\,]$$

où pi(-1) est supposé nul;
d'où nôtre problème de minimisation deviendra fractionné comme suit:

* Pour K=0: trouver Ui(0),Zi(0) tels qu'on ait:

MIN [Hi(Xi(0),Ui(0),Zi(0))] ;avec Xi(0)=Xi0
la résolution de ce problème sera obtenue en faisant

$$\frac{\partial Hi}{\partial Ui} = 0 \quad \text{et} \quad \frac{\partial Hi}{\partial Zi} = 0;$$

$$\text{donc :} \quad Ui(0)=-ri^{-1}\, Bi^t\, pi^*(0)$$

$$Zi(0)=-si^{-1}\,(Ci^t\, pi^*(0)+\lambda i^*(k)) \qquad (IV-12)$$

* Pour k=0,1...K-1

problème : MIN [Hi(Xi(k),Ui(k),Zi(k),k)- pi*(k-1) Xi(k)]

la solution sera :

$$Xi(k)=-qi^{-1}(k)\,[Ai^t\, pi^*(k) - pi^*(k-1) - \sum_{j=1}^{N} [\lambda j^{*t}\, Lji]\,]$$

$$Ui(k)= -ri^{-1}(k)\, Bi^t\, pi^*(k) \qquad (IV-13)$$

$$Zi(k)= -si^{-1}(k)\,[\, Ci^t\, pi^*(k) + \lambda i^*(k)]$$

* Pour k=K

$$\text{problème :} \quad \text{MIN}\ [\ (1/2)\,||X_i(K)||^2 f_i - p_i^{T}(K-1)\ X_i(K)\ ]$$

$$\text{solution:} \quad X_i(K) = f_i^{-1}\ p_i^{*}(K-1) \qquad \text{(IV-14)}$$

Ainsi le problème global de minimisation de J (IV-1) sous les contraintes(IV-2) et (IV-3) peut être traité à l'aide d'un algorithme à 3 niveaux, où l'on substitue,au 1er niveau,les $\lambda^*(k)$ et $p^*(k)$ dans l'expression des solutions données par les relations (IV-12)-(IV-14) en vue d'optimiser les valeurs des vecteurs X , U et Z .

Au niveau 2, en partant des solutions obtenues au 1er niveau,on calculera le gradient de M(p) afin de maximiser cette fonction en améliorant successivement les valeurs de p jusqu'à ce que $\nabla M(p)$ tende vers zero;
l'optimum du vecteur p issu du 2eme niveau est utilisé,au 3eme niveau,pour améliorer itérativement $\varphi(\lambda)$ afin de la maximiser,l'optimum global sera atteint au moment où $\nabla\varphi(\lambda)$ et $\nabla M(p)$ seront nulles simultanemment;la Fig.3 montre cette structure d'optimisation.

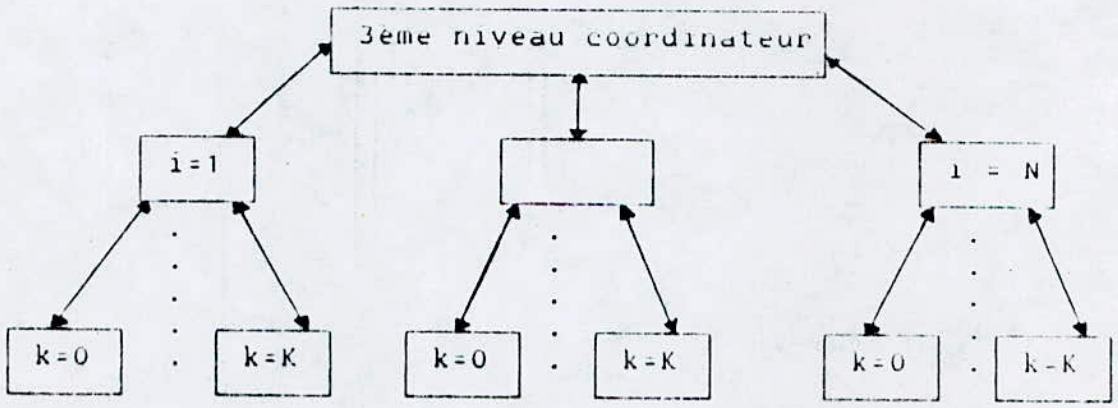


Fig.3 méthode de TAMURA à 3 niveaux

Remarque :

*cette méthode simplifie enormemment les calculs du moment qu'elle permet d'obtenir une solution au niveau bas sans avoir recours à la résolution d'equations compliquees.

* on constate egalement l'existance du terme quadratique $||Z_i||^2 s_i$ dans la fonction critère qui permet, tout comme dans la méthode de coordination du but ,d'éviter les solutions singulieres [5].

IV-3-RESOLUTION INFORMATIQUE

Le programme de cette méthode est donné en annexe H4, pour son élaboration il a fallu avoir recours à plusieurs méthodes numériques entre autres quelques methodes du calcul matriçiel [Annexe F], la méthode du gradient ;qu'on a d'ailleurs développé en annexe[D];
en particulier la méthode du gradient conjuguée dont l'utilisation accèlerera le processus de convergence de l'algorithme de calcul du nouveau vecteur p .

Elle consiste à déterminer p^{i+1} (iteration i+1) comme suit:

$$p^{i+1} = p^{i} + \alpha \, D^{i} \quad ; \ \alpha \text{ etant le pas}$$

$$\text{où} \quad D^{i} = E(p^{i}) + \beta \, D^{i-1} \quad ; \ D^{1} = E(p^{1});$$

$E(p^{i})$: gradient à l'iteration i

avec

$$\beta^{i-1} = \left[\sum_{k=0}^{K-1} E(p^{i}(k)).\, E(p^{i}(k)) \right] / \left[\sum_{k=0}^{K-1} E(p^{i}(k)).E(p^{i}(k)) \right]^{t}$$

on a également utilisé le test de Cholesky, qui permet de voir si une matrice est définie positive[Annexe E], ainsi que le calcul des normes .

CHAPITRE V :

ALGORITHME DE TAMURA DU RETARD

Comme on a souligné auparavant la methode de Tamura ne résout pas les systèmes soumis à des contraintes de type inégalité, or il se trouve que la plupart des systemes pratiques comportent des retards et sont souvent soumis à des contraintes de type inégalité vue la nature physique de ces systèmes.

Tamura a proposé un algorithme dit de retard qui permet de traiter ce genre de problèmes.

Tout système dynamique peut etre représentés par des équations aux différences d'ordre éleve :

$$X(k+1)=\sum_{i=0}^{\theta} Ai\ X(k-i)+\sum_{i=0}^{\theta} Bi\ U(k-i) \qquad (V-1)$$

où Ai (i=0,1,...,θ) est une matrice (nxn)
X est un n-vecteur
U est un r-vecteur
Bi (i=0,1,...,θ) est une matrice (nxr)

avec X(0)=X0 ;(V-2)

nous supposons que X(k)=U(k)=0 ; pour k<0.

Les états et les commandes etant soumis aux contraintes suivantes:

$$Xmin \le X(k) \le Xmax\ ;\ k=0,1,...K$$
$$Umin \le U(k) \le Umax\ ;\ k=0,1,...K-1 \qquad (V-3)$$

probleme:

$$MIN[J=(1/2)||X(K||^2 f+\sum_{k=0}^{K-1}(1/2)(||X(k)||^2 q(k)+||U(k)||^2 r(k))] \qquad (V-4)$$

où f,r et q sont des matrices diagonales definies positives

Un examen préalable de ce problème montre qu'il est difficile à résoudre et ce pour 2 raisons.

1- les contraintes de type inégalites que nous ne pouvons résoudre avec les méthodes vues précedemment.
2- les retards peuvent entrainer une augmentation de l'espace d'état ce qui accroitera l'ordre de nôtre système impliquant ainsi une augmentation des calculs .

Illustrons ceci par un exemple:
soit un système décrit par l'équation d'état:

$$X(k+1) = X(k) + X(k-1)$$

pour étudier ce probleme on introduit une variable additionnelle :

$$X1(k) = X(k-1)$$

nôtre système devient alors:

$$\begin{bmatrix} X(k+1) \\ X1(k+1) \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} X(k) \\ X1(k) \end{bmatrix}$$

ainsi un seul retard augmente l'ordre des matrices d'évolution et des vecteurs d'état,cet accroissement sera encore plus important si le dit système comportait plusieurs retards.
La présente méthode de Tamura permet de surmonter aussi bien les difficultés des contraintes d'inégalités que celle de l'accroissement de l'espace d'état.
Ecrivons le Hamiltonien du système global :

$$H(X(k),U(k),p(k),k) = (1/2)[||X(k)||^2 q(k) + ||U(k)||^2 r(k)] +$$

$$+ \sum_{j=0}^{\theta} [p(k+j)]^t [Aj\ X(k) + Bj\ U(k)];\ k=0,1,\ldots,K-1 \quad (V-5)$$

où $p(k) = 0$ pour $k > K$

pour $p = p^*$ (donné) $= [p^*(0),p^*(1),\ldots,p^*(K-1)]$

on a :

$$L(X,U,p,k) = (1/2)||X(K)||^2 f(k) - p^{*t}(K-1)\ X(k) +$$

$$+ \sum_{k=0}^{K-1} [H(X(k),U(k),p^*(k),k) - p^{*t}(k-1)\ X(k)];\ (V-6)$$

sous les contraintes (V-3)

Pour obtenir le contrôle optimale on calculera p qui maximise le minimum de L(X,U,p,k) (par rapport aux vecteurs X et U);
Comme dans l'algorithme à 3 niveaux,on décompose ,pour p donné, le Lagrangien L (V-6) ,suivant les instants k, en K+1 problemes de minimisation indépendents:

* Pour k=0

probleme:

$$MIN[H(X(0),U(0),p(0))]=MIN[(1/2)(||(X(0)||^2 q(0)+$$

$$+||U(0)||^2 r(0))+\sum_{j=0}^{\theta} p^*(j)^t (Aj\ X(0)+Bj\ U(0))]$$

avec

$$X(0)=X0$$

et

$$Umin \leq U(0) \leq Umax$$

r(0) étant une matrice diagonale, le probleme de minimisation se réduit donc à r problemes de minimisation indépendants à une seule variable,ce qui facilitera la prise en compte des contraintes .

La solution sera:

$$U(0)=SAT2[-r(0)^{-1} \sum_{j=0}^{\theta} Bj^t p^*(j)] \qquad (V-7)$$

où pour i=1,...r, le ième élement de SAT2 est:

$$SAT2(\mu i) = \begin{cases} Umax & si\ \mu i > Umax,i \\ \mu i & si\ Umin,i \leq \mu i \leq Umax,i \\ Umin & si\ \mu i < Umin,i \end{cases} \qquad (V-8)$$

* Pour k=1,...K-1

problème :

$$MIN[H(X(k),U(k),p^*(k),k) - p^*(k-1)^t X(k)]$$

sous les contraintes: $Xmin \leq X(k) \leq Xmax$

$Umin \leq U(k) \leq Umax$

comme precedemment en raison des matrices diagonales q et r le problème original devient (n+r)problèmes de minimisation ,indépendents,à une variable;

ce qui donne comme solution:

$$X^*(k)=SAT1\ [-q(k)\ [-p^*(k-1)+\ \Theta_{j=0}\Sigma\ Aj\ p^*(k+j)]]$$

$$U^*(k)=SAT2\ [-r(k)\ \Theta_{j=0}\Sigma\ Bj\ p^*(k+j)] \qquad (V-9)$$

où pour i=1,...,n ;le ième élement de SAT1 est:

$$SAT1(\tau i) = \begin{cases} Xmax,i & si\ \tau i > Xmax,i \\ \tau i & si\ Xmin,i \leq \tau i \leq Xmax,i \\ Xmin,i & si\ \tau i < Xmin \end{cases} \qquad (V-10)$$

* Pour k=K

problème :

$$MIN[(1/2)||X(K)||^2 f(K)-p^*(K-1)\ X(K)]$$

sous les contraintes: $Xmin \leq X(K) \leq Xmax$

la solution est donnée par:

$$X^*(K)=SAT1[f(K)\ p^*(K-1)] \qquad (V-11)$$

Les solutions données par les relations (V-7) à (V-11) permettent d'obtenir le minimum du Lagrangien pour un p donné ;
au second niveau on améliore p en vue de maximiser la fonction duale en calculant le gradient:

$$X(k+1) - \Theta_{j=0}\Sigma\ [Aj\ X(k-j) + Bj\ U(k-j)]$$

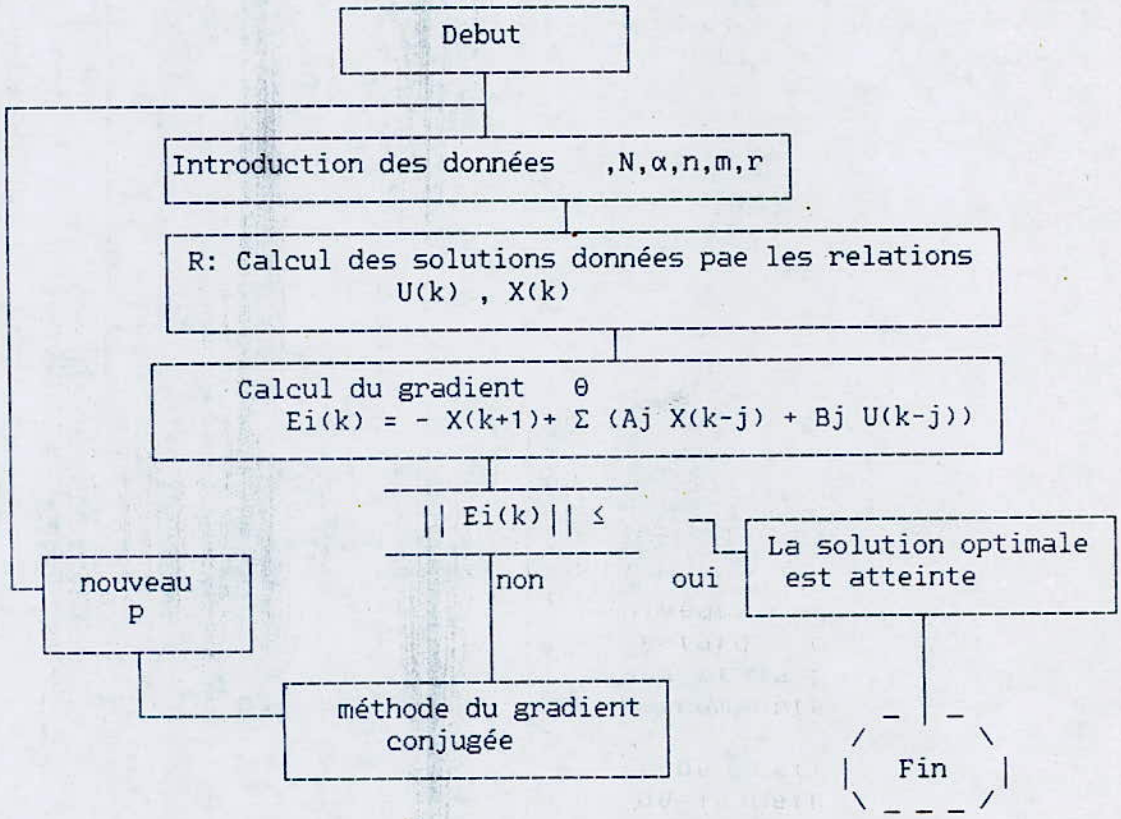
Commentaires:
L'algorithme du retard est une méthode de type coordination du but, largement appliqué dans la pratique pour les raisons suivantes:

1)il y'a moins de calculs car, il n'est pas necessaire d'augmenter l'espace d'état pour prendre en compte les différents retards.
2)les contraintes d'inégalités sont traitées de manière simple.
3)la fonction objectif possède un sens physique ,car aucun terme additionnel n'est ajouté pour éviter les solutions singulières qui se présentent souvent dans l'approche de coordination du but.

Organigramme:

Il ressemble fortement à celui à 3 niveaux excepte que, contrairement à la méthode sans retards de Tamura, l'algorithme du retard comprend 2 niveaux et la phase de calcul des solutions dans cette approche comporte des tests d'inégalités.

En raison de la longueur du programme de cette methode nous ne donnons içi que l'organigramme de la méthode.



CONCLUSION GENERALE

Dans ce travail,nous avons presente les differentes methodes de contrôle et d'optimisation des systemes lineaires.

Ainsi pour les systemes statiques,on a developpe quelques algorithmes de resoluton issus de la theorie du Simplexe,et nous avons montré que pour les systemes d'ordre eleve la methode de décomposition de Dantzing-Wolfe est la plus adequate aussi bien du point de vue capacite memoire que de celui du temps de calcul necessaires à ce type de programme.

Pour les systemes dynamiques,on a vu que les methodes variationnelles s'averent inaptes à resoudre d'innombrables problemes poses par les techniques actuelles particulierement le problème de contrôle des grands systemes complexes; aussi on a opté à l'utilisation des methodes d'optimisation hierarchiques longuement exposées dans la 3eme partie .

Nous avons également montré que la methode de Tamura apporte beaucoup de simplifications quant a la resolution de ce type de problèmes .

En annexe,nous donnons quelques programmes de controle des systemes lineaires notamment ceux du simplexe,simplexe revisé,Dantzing-Wolfe et celui de Tamura.

Nous esperons enfin que le travail qui a ete developpe dans cette presente these ainsi que celles precedentes traitant le probleme de commande et de contrôle des systemes serviront de guide aux ingenieurs qui doivent tout simplement choisir une methode ou une combinaison de methodes en fonction des problemes poses et,évidemment, des moyens de calcul dont ils disposent .

ANNEXES

ANNEXE A

RAPPEL SUR LES CONVEXES ET LES CONTRAINTES SATUREES

1-LES CONVEXES

définition 1 : soit A une partie d'un espace affine E; A est dit convexe si toute combinaison linéaire convexe

$$V = \sum_{i=1}^{n} \alpha_i V_i \; ; \; \alpha_i \geq 0, \; \sum_{i=1}^{n} \alpha_i = 1$$

des points de A est un point de A.

proposition 1 : L'intersection d'une famille de convexe est convexe.

définition 2 : un point V de A est appelé point extrémal s'il ne peut être exprimé comme une combinaison linéaire convexe des points de A(exemple: sommets d'un triangle,frontière d'un disque)

définition 3 :on appelle face d'un convexe une partie F de A telle que pour toute droite D,]DnA[∩ A = ∅ ⟶ (D ∩ A) ⊂ F. un convexe dont le nombre de faces est fini est appelé polyèdre convexe

définition 4 : une variété linéaire de l'espace E est une partie A de E qui contient toutes les droites passannt par 2 quelconques de ses points.

Un hyperplan de E est une variété linéaire de dimension (dim E - 1)
en particulier l'ensemble des points M de E tels que f(M)+a=0;où f est une forme linéaire sur E et a un scalaire ;est une variété linéaire.

proposition 2 : l'intersection de 2 polyèdres convexes est un polyèdre convexe.

proposition 3 : Une variete linéaire est un polyèdre convexe.

proposition 4 :Le demi espace fermé contenant les points M de E tels que f(M) + a ≤ 0;où f est une forme linéaire sur E et a un scalaire;est un polyèdre convexe.

2-SYSTEME SATURE

On appelle système de contraintes saturées(ou système saturé) un sous système de l'ensemble des contraintes du problème,de la forme :

$$\begin{cases} \alpha_i \; x = d_i \; ; \; i \in I1 & (I1 \subset M) \\ x_j = 0 \; ; \; j \in (N-J1) & (J1 \subset N) \end{cases}$$

où M et N sont respectivement les ensembles dse indices des contraintes et ceux des variables.

avec Card(I1)=Card(J1),c'est à dire obtenu en remplaçant certaines inéquations par des équations.

Un système de contraintes saturées est dit régulier lorsque la matrice A1=(aij);i∈I1 et j∈J1;est réguliere.

Le système $\alpha_i x_i = d_i, i \in I1$, constiue alors par definition un système de Cramer par rapport aux variables $x_j, j \in J1$, lorsqu'on y est fait $x_j = 0, j \in (N-J1)$.

ANNEXE B

RAPPELS D'ALGEBRE LINEAIRE

THEOREME 1:

Des vecteurs X1,X2,...,Xk sont dits lineairement independants si et seulement si :

$$\sum_{i=1}^{k} \pi i \, Xi = 0 \dashrightarrow \pi i = 0 \quad (i=1,2,...k)$$

dans le cas contraire ils sont dits dépendants.

définition 1

Un vecteur X est une combinaison lineaire de vecteurs Xj (j=1,..k) si:

$$X = \sum_{j=1}^{k} \pi j \, Xj$$

définition 2

On appelle base d'un espace vectoriel V^n à n dimensions tout ensemble de n vecteurs linéairement indépendants de V^n.

THEOREME

Etant donné une base Xi (i=1,2,..n) d'un ensemble S^n de vecteurs et l'expression linéaire d'un vecteur X quelconque (autre que le 0)de cet ensemble en fonction des vecteurs de la base ,

$$X = \sum_{i=1}^{n} \pi i \, Xi \quad \text{(au moins un } \pi i \text{ non nul)}$$

on obtient encore une base en substituant X à Xi dans la base actuelle si et seulement si πi est different de 0.

ANNEXE C

1- INVERSION PAR PARTITIONNEMENT

Soit A une matrice carrée réguliere partitionnee de la façon suivante:

$$A=\begin{bmatrix} M & R \\ L & N \end{bmatrix}$$

M:matrice mxm
L:matrice nxm
R:matrice mxn
N:matrice nxn

et telle que N soit réguliere.

Partitionnons A^{-1} de la meme façon que A:

$$A^{-1} = \begin{bmatrix} \mu & \lceil \\ \pi & \Omega \end{bmatrix}$$

en effectuant le produit de $A.A^{-1}$ et en l'identifiant à $I_{(m+n)}$, il est facile de voir que $\mu, \pi, \lceil, \Omega$ sont données par les relations suivantes:

$$\begin{cases} \mu = [M - R\,N^{-1}\,L]^{-1} \\ \pi = -N^{-1}\,L\,\mu \\ \lceil = \mu\,R\,N^{-1} \\ \Omega = N^{-1} - N^{-1}\,L\,\lceil \end{cases}$$

2-FORME PRODUIT DE L'INVERSE

a)resultats préliminaires:

Soit $A = (a1, a2, \ldots, ap, \ldots, am)$ une matrice réguliere mxm, dont l'inverse A^{-1} est connue.
Nous nous proposons de trouver l'inverse de :

$$A' = (a1, a2, \ldots, p, \ldots, am)$$

déduite de A par substitution du vecteur ap ;
puisque A est réguliere ,par hypothèse ,les vecteurs aj (j=1,..,m) sont linéairement indépendents et forment donc une base .
Soit y le vecteur exprimant p en fonction des vecteurs aj:

$$p = Ay = \sum_{j=1}^{m} y_j\,a_j$$

une condition necessaire et suffisante pour que $(A')^{-1}$ existe est que les vecteurs (a1,a2,...,p,...am) soient linéairement indepndents

Nous supposons qu'il en est bien ainsi on peut alors écrire :

$$ap = -(1/yp) \sum_{j=p} yj\ aj + (p/yp)$$

soit $ap = A'\ vp$ [C-1]

en posant

$$vp = [(-y1/yp),(-y2/yp),..(-yp-1/yp),(-1/yp),(-yp+1/yp),..(-ym/yp)]^*$$

et en notant que le produit d'une matrice quelconque par le ième vecteur unité est la ième colonne de la matrice ,il resulte alors de la definition de A' et de la formule [C-1] que:

$A = A'(e1,e2,...,ep-1,vp,ep+1,...,em) = A'\ Jp$ [C-2]

ou ei désigne comme d'habitude le ième vecteur unité,Jp est une matrice obtenue à partir de la matrice unité I(m) par la substitution de vp à la peme colonne ep;

multiplions les deux membres extrêmes de [C-2] à gauche par $(A')^{-1}$, on obtient : $(A')^{-1} = Jp\ A^{-1}$

avec

$$Jp = (e1,e2,...ep-1,vp,.....em)$$

$$vp = \left(-\frac{y1}{yp},-\frac{y2}{yp},...,-\frac{yp+1}{yp},\frac{1}{yp},-\frac{yp+1}{yp},...,-\frac{ym}{yp}\right)^*$$

$$(y1,y2,...,yp-1,yp,yp+1,...,ym) = y^*$$

$$y = A^{-1}\ p$$

b/ Calcul de l'inverse de A

Le resultat précedent peut servir à calculer de proche en proche l'inverse de A^{-1} d'une matrice régulière, m x m : $A = (a1,...,am)$

on part de la matrice unité $I^{(m)} = (e1,e2,...,em)$

et on remplace successivement e1,e2,...,em par a1,a2,...,am formant ainsi les matrices A1,A2,A3,...,Am

on a : $A1 = J1\ I^{(m)} = J1$

$A2 = J2\ A1^{-1} = J2\ J1$

etc.... $A = Am = Jm\ A(m-1)^{-1} = ...= Jm\ J(m-1)...J1$

avec

$$J1 = (e1,e2,...,e(i-1),vi,e(i+1),...,em)$$

$$vi = \left(-\frac{y11}{yii},-\frac{yi2}{yii},...,-\frac{yi,i-1}{yii},\frac{1}{yii}^*,-\frac{yi,i+1}{yii},...,-\frac{yim}{yii}\right)$$

$$(yi1,yi2,...,yim) = yi$$

$$yi = J(i-1)\ J(i-2)..... J1\ a1$$

ANNEXE D

METHODE DU GRADIENT

Soit une fonction dont on cherche le minimum ,si f(x) est continue et differentiable,alors dans le voisinage proche de x on peut developper en serie de Taylor limite au 1er terme:

$$f(x+\delta x) = f(x) + f'(x)\delta x$$

si $f'(x) < 0$ alors : $f(x+\delta x) < f(x)$

δx est dit direction de descente.

Le choix de la directionδx minimisant $f(x+\delta x)$ est celui qui rend $\delta x\, f'(x)$ algebriquement minimum,et ce sera bien sûr la direction colineaire à $f'(x)$ et de sens contraire $-f'(x)$.
On dit aussi que la direction $-f'(x)$ est la direction de la plus grande pente de f(x) au point x.

a)<u>Methode du gradient (ou de la plus grande pente)</u>

On passe d'un estimé $x^{(k)}$ de la solution x^* par l'algorithme:

$$x^{(k+1)} = x^{(k)} - t^{(k)} f'(x^{(k)})$$

jusqu'à ce que $\| f(x^{(k+1)}) \| < \pi$

π: nombre tres petit

deux techniques existent pour determiner le scalaire $t^{(k)}$ qu'on appelle aussi le pas:

1-pas optimal

On determine $t^{(k)}$ minimisant $f(x^{(k+1)})$

2-pas de descente

On choisit $t^{(k)}$ tel que :$f(x^{(k+1)}) < f(x^{(k)})$

b)<u>Methode du gradient conjuguée</u>

Generalement le choix de la direction de recherche $p^{(k)} = - f'(x^{(k)})$ n'est pas globalement le meilleur en pratique.

Dans la presente methode,on genere des directions $p^{(k)}$ definies ci-dessous:

L'algorithme est:

$$x^{(k+1)} = x^{(k)} + t^{(k)} p^{(k)}$$

où $p^{(k)} = - f'^{(k)} + \alpha^{(k)} S^{(k-1)}$; $\alpha^{(0)} = 0$

$\alpha^{(k)} = \| f(x^{(k)}) \|^2 / \| f(x^{(k-1)}) \|^2$

ANNEXE E

TEST DE CHOLESKY

a-1-forme quadratique

la forme quadratique est un scalaire defini par :

$$q = x^t Q x = \sum_{i,j=1}^{n} qij\ xi\ xj$$

Q est une matrice carree (nxn) et symetrique,c'est à dire ses elements doivent respecter la condition qij=qji ;le vecteur x posséde alors la dimension n.

a-2-matrices définies positives

La matrice carrée est définie positive lorsqu'il existe pour sa forme quadratique la condition :

$$q = x^t\ Q\ x > 0 \ ; x \text{ non nul}$$

le scalaire x doit donc être positif pour toutes lés valeurs du vecteur x exception faite pour x=0 ,dans ce cas on a evidemment q=0.

La matrice Q est semi définie positive lorsque pour certaines valeurs de x=0 le scalaire q s'annule et, mis a part ces points particuliers,le scalaire est toujours positif.

On a donc la condition :

$$q = x^t\ Q\ x \geq 0 \ ; \ x \text{ non nul}$$

pour tester si une matrice est définie positive ou non ,on peut appliquer le test de Sylvester [16],ou bien le test de Cholesky decrit ci-dessous.

b-Test de Cholesky

b-1-Theoreme de Cholesky

Si une matrice est symetrique et définie positive,alors elle peut être décomposée en :

$$A = L\ .\ L^t$$

où L est une matrice reelle triangulaire inferieure.

b-2-Decomposition de la matrice A

Algorithme de Cholesky

$$Lrr = \sqrt{[arr - \sum_{k=1}^{r-1} L^2rk]} \qquad r=1,\ldots n$$

$$Ljr = [arj - \sum_{k=1}^{r-1} Lrk\ .\ Lkj\ ]/\ Lrr \qquad ;j=r+1,..n$$

b-3-Test

si un terme $\sqrt{[arr - \sum_{k=1}^{r-1} L^2rk]} < 0$

alors la matrice n'est pas définie positive.

Ce test constitue un moyen simple pour verifier q'une matrice A est bien définie positive.

ANNEXE E

INVERSION D'UNE MATRICE

Calcul de l'inverse d'une matrice A par la methode de Jordan

L'algorithme de Jordan[13] opere le passage de la matrice C=[A,b] à la matrice D=[I,X] où X est solution du systeme lineaire A X=b

soit $X = A^{-1} b$

Donc ,fonctionnellement,l'algorithme de Jordan fait la transformation de $C = [A,b]$ à $b = [I,A^{-1},b]$

l'interet de cette remarque est que si l'on augmente à nouveau C on lui adjoinant la matrice identité In,on aura $C = [A,I]$

après les operations de l'algorithme de Jordan on obtiendera :

$$D = A^{-1} C = A^{-1} [A,I] = [I,A^{-1}] .$$

La methode se divise en deux phases :

1ere phase : recherche du pivot:

a/ Si le pivot $a_{kk}^{(k-1)}$ est nul,on cherche alors un autre pivot $a_{kk}^{(k-1)}$ telque : $a_{kk}^{(k-1)} = \max_{i \in [k,n]} | a_{ik}^{(k-1)} |$

b/ Si le pivot est trés petit ce qui peut causer des erreurs d'arrondis importantes [13] on choisira alors un pivot:

$$a_{lm}^{(k-1)} = \max_{i,j \in [k,n]} | a_{ij}^{(k-1)} |$$

c/ Si tous les pivots sont nuls alors la matrice A est singuliere donc non inversible.

2eme phase : algorithme de Jordan

$$\left.\begin{array}{l} a_{kj} = a_{kj}/a_{kk} \; ; \; j = 2\,n\,,\ldots,k \\ a_{ij} = a_{ij} - a_{ik}\,a_{kj} \; ; \; j = 2\,n\,,\ldots,k \\ \qquad\qquad\qquad\quad i = 1,2,\ldots,n \\ \qquad\qquad\qquad\quad i = j \end{array}\right| \quad k = 1,\ldots,n$$

cette methode permet de calculer l'inverse d'une matrice avec un nombre d'operations netement inferieur à celui de la methode de Cramer [13] .

ANNEXE II

PROGRAMMES

```
1 KEY OFF:CLS
3 FOR I=3 TO 70 STEP 2:LOCATE 2,I:PRINT "*":LOCATE 8,I:PRINT "*":LOCATE 20,I:P
NT "*":NEXT I
5 FOR I=2 TO 20:LOCATE 1,3:PRINT "*":LOCATE I,14:PRINT "*":LOCATE I,70:PRINT "
:NEXT I
7 LOCATE 5,20:PRINT " TAPER SUR UN DES CHIFFRES "
9 FOR I=1 TO 2:LOCATE  4+6*I,8:PRINT I:NEXT I
11 LOCATE 10,20:PRINT " METHODE DU SIMPLEXE GENERAL "
13 LOCATE 16,20:PRINT " METHODE DU SIMPLEXE PRIMAL-DUAL "
15 LOCATE 23,70:INPUT Y:IF Y=1 THEN 17
16 IF Y=2 THEN 17 ELSE 15
17 LOCATE 23,20:PRINT " VOULEZ VOUS CORRIGER  (O/N) "
18 CC$=INKEY$:IF CC$="" THEN 18
21 IF ASC(CC$)=79 THEN 15
24 IF ASC(CC$)=78 THEN 28 ELSE 17
28 SCREEN 0,0:CLS
29 IF Y=2 THEN 1590
30 '**************** ALGORITHME GENERAL DU SIMPLEXE ****************
35 CLS
40 '********************* ENTREE DES DONNEES *************************
45 FOR I=3 TO 70 STEP 2:LOCATE 1,I:PRINT "*":LOCATE 22,I:PRINT "*":NEXT I
50 FOR I=1 TO 22:LOCATE I,3:PRINT "*":LOCATE I,70:PRINT "*":NEXT I
57 LOCATE 4,20:PRINT" ALGORITHME GENERAL DU SIMPLEXE "
60 LOCATE 9,10:PRINT "NOMBRE D'INCONNUES =";:INPUT N
70 LOCATE 12,10:PRINT "NOMBRE DE CONTRAINTES DU TYPE =< ";:INPUT T1
80 LOCATE 15,10:PRINT "NOMBRE DE CONTRAINTES DU TYPE >= ";:INPUT T2
90 LOCATE 19,10:PRINT "NOMBRE DE CONTRAINTES DU TYPE =";:INPUT T3
95 CLS
100 D1=T1+T2+T3-1:D2=N+D1+1+T2:DIM PGM(D1,D2)
110 DIM BAS(D1)
120 FOR I=0 TO D1
130 FOR J=0 TO D2
140 PGM(I,J)=0
150 NEXT J
160 NEXT I
170 IF T1=0 THEN 250
180 FOR K=1 TO T1
190 LOCATE 6,20:PRINT K;"ie CONTRAINTE DU TYPE =<:"
200 FOR J=0 TO N-1
210 LOCATE 10,20:PRINT "COEF DE X";J+1;:INPUT " = ";PGM(K-1,J)
220 NEXT J
230 LOCATE 14,20:INPUT "VALEUR DU SECOND MEMBRE =";PGM(K-1,D2)
240 NEXT K
250 IF T2=0 THEN 340
260 L1=T1+1:L2=T2+T1:CLS
270 FOR K=L1 TO L2
280 LOCATE 6,20:PRINT K-L1+1;"ie CONTRAINTE DU TYPE >="
290 FOR J=0 TO N-1
300 LOCATE 10,20:PRINT "COEF DE X";J+1;:INPUT " = ";PGMP(K-1,J)
310 NEXT J
320 LOCATE 14,20:INPUT "VALEUR DU SECOND MEMBRE =";PGM(K-1,D2)
330 NEXT K
340 IF T3=0 THEN 430
350 L1=T1+T2+1:L2=T1+T2+T3:CLS
360 FOR K=L1 TO L2
370 LOCATE 6,20:PRINT K-L1+1;"ie CONTRAINTE DU TYPE ="
380 FOR J=0 TO N-1
390 LOCATE 10,20:PRINT "COEF DE X";J+1;:INPUT " =";PGM(K-1,J)
400 NEXT J
410 LOCATE 14,20:INPUT "VALEUR DU SECOND MEMBRE =";PGM(K-1,D2)
420 NEXT K
425 CLS
430 A1=N+T1-1:A2=A1+T2:A3=A2+T2:DIM ECOM(2,D2):A4=A3+T3
460 FOR J=0 TO D2
```

```
980 NEXT I
990 ECOM(0,J)=0:ECOM(1,J)=0
1000 NEXT J
1010 EXO=1:COEF=MODE:GOSUB 1270
1020 IF RET=2 THEN 1180
1050 REM**************** AFFICHAGE DU PROGRAMME OPTIMAL ******************
```

```
1060 REM
1070 CLS:LOCATE 3,30:"SOLUTION "
1080 FOR I=1 TO N
1090 FOR J=0 TO D1
1100 IF BASE(J)=I-1 THEN  LOCATE 5+I,10:PRINT "X";I;"=";PGM(J,D2):GOTO 1130
1110 NEXT J
1120 LOCATE LOCATE 5+I,10:PRINT "X";I;"=0"
1130 NEXT I
1140 LOCATE 7+N:PRINT "VALEUR DE LA FONCTION ECONOMIQUE=";ECOM(1,D2)
1160 END
1170 CLS:LOCATE 13,10:PRINT "CE PROBLEME N'A PAS DE SOLUTION OPTIMALE":PRINT :G
TO 1150
1190 FOR JJ=0 TO A4
1200 IF ECOM(0,JJ)>-ZZ THEN 1250
1210 FOR I=0 TO D1
1220 PGM(I,JJ)=0
1230 NEXT I
1240 ECOM(0,JJ)=0:ECOM(1,JJ)=0
1250 NEXT JJ
1260 GOTO 1010
1270 REM
1280 REM BOUCLE ALGORITHME DU SIMPLEXE
1290 REM
1300 MAX=ZZ
1310 FOR J=0 TO D2-1
1320 IF ECOM(EXO,J)*COEF>MAX THEN MAX=ECOM(EXO,J)*COEF:J1=J
1330 NEXT J
1340 IF MAX=ZZ THEN RET=1:RETURN
1350 MI=1E+20
1360 FOR I=0 TO D1
1370 A=PGM(I,J1)
1380 IF A<ZZ THEN 1410
1390 R=PGM(I,D2)/A
1400 IF R<MI THEN MI=R:I1=I
1410 NEXT I
1420 IF MI=1E+20 THEN RET=2:RETURN
1430 BASE(I1)=J1:P=PGM(I1,J1)
1440 FOR J=0 TO D2

1450 PGM(I1,J)=PGM(I1,J)/1450 PGM(I1,J)=PGM(I1,J)/P
1460 NEXT J
1470 FOR I=0 TO D1
1480 IF I1=I THEN 1530
1490 B1=PGM(I,J1)
1500 FOR J=0 TO D2
1510 PGM(I,J)=PGM(I,J)-B1*PGM(I1,J)
1520 NEXT J
1530 NEXT I
1540 B1=ECOM(EXO,J1):B2=ECOM(EXO+1,J1)
1550 FOR J=0 TO D2
1560 ECOM(EXO,J)=ECOM(EXO,J)-B1*PGM(I1,J):ECOM(EXO+1,J)=ECOM(EXO+1,J)-B2*PGM(I1
J)
1570 NEXT J
1580 GOTO 1300
1581 END
1590 LET C=1
1597 CLS
1600 DIM A(18,30)
1610 FOR I=1 TO 71 STEP 2:LOCATE 2,I:PRINT"*":LOCATE 6,I:PRINT"*":LOCATE 22,I:P
INT"*":NEXT I
1612 LOCATE 4,10:PRINT" CHOISISSER LE TYPE DE REPONS QUE VOUS VOULEZ "
1615 FOR I=1 TO 21 :LOCATE 1+I,1:PRINT"*":LOCATE 1+I,8:PRINT"*":LOCATE 1+I,73:P
INT"*":NEXT I
1620 LOCATE 3,10:PRINT"1POUR SORTIR LES VALEURS AUX ETNEXTBASE A CHAQUE ITERATION
```

```
1630 LOCATE 14,10:PRINT " POUR LA BASE SEULEMENT,"
1635 LOCATE 18,10:PRINT " POUR LA SOLUTION SEULEMENT."
1638 LOCATE 24,75:INPUT P5
1640 LOCATE 23,40:PRINT" VOULEZ VOUS CORRIGER (O/N) "
1643 CC$=INKEY$:IF CC$="" THEN 1643
1645 IF ASC(CC$)=79 THEN 1625
1647 IF ASC(CC$)=78 THEN 1660 ELSE 1640
1660 CLS
1664 LOCATE 13,20:PRINT "DONNER LE NOMBRE DE CONTRAINTES";:INPUT M
1666 CLS
1670 LOCATE 13,20:PRINT "DONNER LE NOMBRE DE VARIABLES";:INPUT N
1675 CLS
1677 LOCATE 8,20:INPUT "NOMBRE D'EQUATIONS DE TYPE  =<  ";L
1678 PRINT
1679 LOCATE 13,20:INPUT " NOMBRE D'EQUATIONS DE TYPE  =  ";E
1680 PRINT
1681 LOCATE 18,20:INPUT " NOMBRE D'EQUATIONS DE TYPE >= ";G
1682 CLS
1710 IF L+E+G-M THEN 1740
1720 PRINT "LES DONNEES NE SONT PAS PRATICABLES."
1730 STOP
1740 B=M+N+G+1
1750 W=M
1760 IF B*(W+1)<540 THEN 1790
1770 PRINT "LE PROBLEME EST TROP GRAND"
1780 STOP
1790 IF B>30 THEN 1810
1800 IF W+1<30 THEN 1910
1810 PRINT "CHANGER LA DIMENSION D'ENTREE(NO 102) POUR A";W+1;"BY";B;"MATRICE"
1820 PRINT
1830 PRINT "SOIT L'INSTRUCTION 164 OU 168 OU LES DEUX DOIVENT ETRE CHANGEES"
1840 PRINT "COMME SUITE:"
1850 PRINT
1860 IF B<31 THEN 1880
1870 PRINT " 164 IF B>";B;"THEN 169"
1880 IF W+1<18 THEN 1900
1890 PRINT "168 IF W+1<";"THEN 180"
1900 REM INITIALISATION DU TABLEAU SIMPLEXE
1910 M=M-1
1920 H=1
1930 FOR I=0 TO W+1
1940 FOR J=1 TO B
1950 A(I,J)=0
1960 NEXT J
1970 NEXT I
1975 REM ********************  INTRODUCTION DES DONNEES  ********************
1980 FOR I=0 TO M
1985 PRINT "DONNER LES COEFS DE L'EQUATION";I+1
1986 PRINT
1990 FOR J=1 TO N
2000 PRINT "LE COEF DE X";J;:INPUT A(I,J)
2005 PRINT
2010 PRINT
2015 PRINT
2020 NEXT J
2030 NEXT I
2040 FOR I=0 TO M
2050 PRINT "DONNER LE SECOND MEMBRE DE L'EQUATION";I+1;:INPUT A(I,B)
2055 PRINT
2060 PRINT
2070 NEXT I
2072 CLS:PRINT :PRINT "************************************************":PRINT
2074 PRINT " TAPER  1  POUR UNE MINIMISATION  ":PRINT :PRINT
2076 INPUT Z        2  POUR UNE MAXIMISATION  "
```

```
2085 PRINT
2090 PRINT "LA FONCTION ECONOMMIQUE"
2095 PRINT
2097 FOR J=1 TO N
2100 PRINT "LE COEF DE X";J;:INPUT A(W,J)
2105 PRINT
2110 PRINT
2118 IF Z=1 THEN 2125
2120 A(W,J)=-A(W,J):GOTO 2130
2125 A(W,J)=A(W,J)
2130 NEXT J
2140 FOR  K=1 TO M+1
2150 A(K-1,N+G+K)=1
2160 A(K-1,0)=K+N+G
2170 NEXT K
2180 IF E<>0 THEN 2200
2190 IF G=0 THEN 2350
2200 FOR K=L+E+1 TO M+1
2210 A(K-1,K+N-L-E)=-1
2220 NEXT K
2230 W=W+1
2240 Q=0
2250 FOR  J=1 TO N+G
2260 S=0
2270 FOR I=M-G-E+1 TO M
2280 S=S+A(I,J)
2290 NEXT I
2300 A(W,J)=-S
2310 IF Q<A(I,J) THEN 2340
2320 Q=A(W,J)
2330 C=J
2340 NEXT J
2350 PRINT
2360 PRINT "VOS VARIABLES";H;"A";N
2370 IF G=0 THEN 2390
2380 PRINT "VOS VARIABLES DE SURPLUS";N+1,"A";N+G
2390 IF L=0 THEN 2410
2400 PRINT "VOS VARIABLES D'ECART";N+G+1;"A";N+G+L
2410 IF G+E=0 THEN 2430
2420 PRINT "VARIABLES ARTIFICIELLES";N+G+L+1;"A";B-1
2430 IF P5=0 THEN 2450
2440 GOSUB 3190
2450 IF G+E=0 THEN 2790
2460 IF Q=0 THEN 2860
2470 IF P5>0 THEN 2980
2480 H=H+1
2490 Q=1E+30
2500 R=-1
2510 FOR I=0 TO M
2520 IF A(I,C)<=0 THEN 2560
2530 IF A(I,B)/A(I,C)>Q THEN 2560
2540 Q=A(I,B)/A(I,C)
2550 R=I
2560 NEXT I
2570 IF R>=-.5 THEN 2610
2580 PRINT "SOLUTION NON DEFINIE"
2590 GOSUB 3190
2600 STOP
2610 P=A(R,C)
2620 A(R,0)=C
2630 FOR J=1 TO B
2640 A(R,J)=A(R,J)/P
2650 NEXT J
2660 FOR I=0 TO W
2670 IF I=R THEN 2740
```

```
2680 FOR J=1 TO B
2690 IF J=C THEN 2730
2700 A(I,J)=A(I,J)-A(R,J)*A(I,C)
2710 IF ABS(A(I,J))>1E-08 THEN 2730
2720 A(I,J)=0
2730 NEXT J
2740 NEXT I
2750 FOR I=0 TO W
2760 A(I,C)=0
2770 NEXT I
2780 A(R,C)=1
2790 Q=0
2800 FOR J=1 TO N+G+L
2810 IF A(W,J)>Q THEN 2840
2820 Q=A(W,J)
2830 C=J
2840 NEXT J
2850 GOTO 2460
2860 IF W=M+1 THEN 2890
2870 W=W-1
2880 GOTO 2790
2890 PRINT
2900 FOR I=0 TO M
2910 IF A(I,0)<N+L+G+1 THEN 2930
2920 IF A(I,B)<>0 THEN 2950
2930 NEXT I
2940 GOTO 2970
2950 PRINT "PAS DE SOLUTION FAISABLE TROUVEE"
2960 GOTO 3320
2970 PRINT "REPONSES"
2980 PRINT
2990 IF Q=0 THEN 3010
3000 PRINT "BASE AVANT ITERATION";H
3010 PRINT "VARIABLE","VALEUR"
3020 FOR I=0 TO M
3030 IF A(I,0)=0 THEN 3050
3032 IF A(I,0)>N THEN  3050
3040 PRINT" X ";A(I,0);"=",A(I,B)
3050 NEXT I
3055 IF Z=1 THEN  3075
3060 PRINT
3070 PRINT "VALEUR DE LA FONCTION OBJECTIVE",A(W,B):GOTO 3080
3075 PRINT "VALEUR DE LA FONCTION OBJECTIVE",-A(W,B)
3080 IF Q<>0 THEN 2480
3090 PRINT
3100 PRINT
33100 PRINT
3110 PRINT "VARIABLES DUALES"
3120 PRINT "COLONNE","VALEUR"
3130 FOR J=N+1 TO B-G-1
3140 PRINT"Y"J-N,"="A(W,J)
3150 NEXT J
3160 IF P5=0 THEN 3180
3170 GOSUB 3190
3180 GOTO 3320
3190 PRINT
3200 IF P5=1 THEN 3300
3210 PRINT
3220 PRINT "TABLEAU APRES";H-1;"ITERATIONS"
3230 FOR I=0 TO W
3240 FOR J=1 TO B
3250 PRINT A(I,J),
3260 NEXT J
3280 PRINT
```

```
3290 NEXT I
3300 RETURN
3310 GOTO 1590
3320 END
```

```
2 KEY OFF:CLS
3 REM ********************  PROGRAMMATION DE LA M.R.S  ****************
5 COLOR 5,3:LOCATE 13,13:PRINT "*************  PROGRAMMATION DE LA M.R.S ******
******":COLOR 3,0:FOR X=1 TO 1500:X=X+1:NEXT X:CLS
6 FOR I=5 TO 65 STEP 2 LOCATE 2,I:PRINT"*":LOCATE 22,I:PRINT "*":NEXT I
7 FOR I=5 TO 65 STEP 2:LOCATE 3,I:PRINT "*":LOCATE 23,I:PRINT "*":NEXT I
8 FOR I=1 TO 21:LOCATE 2+I,5:PRINT "* *":LOCATE 2+I,63:PRINT "* *":NEXT I
10 LOCATE 4,15:PRINT "NOMBRE DE VARIABLES ";:INPUT N
15 LOCATE 8,15:RRINT "NOMBRE DE CONTRAINTES";:INPUT M
20 LOCATE 12,15:PRINT "NOMBRE DE CONTRAINTES DE TYPE =<";:INPUT L
25 LOCATE 16,15:PRINT "NOMBRE DE CONTRAINTES DE TYPE >=";:INPUT G
30 COLOR 3,5
31 LOCATE 20,15:PRINT" VOULEZ VOUS CORRIGER VOS DONNEES (O/N) ";:INPUT CC$
32 COLOR 3,0
33 IF ASC(CC$)=79 THEN  10
34 IF ASC(CC$)=78 THEN 37
35 IF ASC(CC$)<>78 AND 79 THEN 32
37 CLS
40 REM ************** INITIALISATION DU TABLEAU DE LA M.R.S *****************
45 DIM A(100,100)
50 W=N+L+G:H=0
52 B=L+G+1
55 FOR I=1 TO B
60 FOR J=0 TO W+1
70 A(I,J)=0
80 NEXT J
85 NEXT I
90 REM ******************** INTRODUCTIO DES DONNEES ************************
100 CLS:LOCATE 4,20:PRINT" LES COEFS DE LA MATRICE CONTRAINTE"
105 FOR I=1 TO M
107 LOCATE 8,10:PRINT "CONTRAINTE No:";I
110 FOR J=1 TO N
120 LOCATE 10+J,10:PRINT"LE COEF DE X";J;:INPUT A(I,J)
130 NEXT J
135 NEXT I
140 CLS:LOCATE 10,20:PRINT " LES SECONDS MEMBRES "
150 FOR I=1 TO M
155 LOCATE 10+2*I,10:INPUT A(I,W+1)
160 NEXT I
161 CLS:COLOR 3,5
162 LOCATE 10,20:PRINT " TAPER   1   POUR UNE MINIMISATION  "
163 LOCATE 15,20:PRINT " TAPER   2   POUR UNE MAXIMISATION  ";INPUT Y
164 COLOR 0,3:ON Y GOTO 165,165:GOTO 162
165 CLS:LOCATE 8,10:PRINT" LES COEFS DE LA FONCTION OBJECTIVE"
167 FOR J=1 TO N
169 LOCATE 8+2*J,8:INPUT A(B,J)
170 IF Y=1 THEN 174
172 A(B,J)=-A(B,J)
174 NEXT J
178 CLS:COLOR 3,5
180 LOCATE 13,20:PRINT" VOULEZ VOUS CORRIGER VOS DONNEES (O/N) ";:INPUT CC$
181 COLOR 3,0
182 IF ASC(CC$)=79 THEN 100
185 IF ASC(CC$)=78 THEN 195
186 IF ASC(CC$)<>78 AND 79 THEN 180
195 IF L=0 THEN 251
200 FOR I=1 TO L
210 A(I,M+I)=1
240 A(I,0)=N+I
250 NEXT I
251 IF G=0 THEN 260
252 FOR I=1 TO G
253 A(I+L,M+L+I)=1
254 A(I+L,0)=N+L+I
```

```
255 NEXT I
260 REM ******************** APPLICATION DE LA M.R.S ***************************
270 Q=0
280 FOR J=1 TO W
290 IF A(B,J)>=Q THEN 320
300 Q=A(B,J)
310 C=J
320 NEXT J
330 IF Q=0 THEN 800
340 U=1E+30
345 R=0
346 FOR I=1 TO L+G
350 IF A(I,C)=<0 THEN 370
360 IF A(I,W+1)/A(I,C)>U THEN 370
365 U=A(I,W+1)/A(I,C)
367 R=I
370 NEXT I
378 IF R=0 THEN 800:P=A(R,C)
380 FOR J=1 TO W+1
385 A(R,J)=A(R,J)/P
390 NEXT J
395 FOR I=1 TO B
400 IF I=R THEN 440
410 FOR J=1 TO W+1
415 IF J=C THEN 435
420 A(I,J)=A(I,J)-A(R,J)*A(I,C)
425 IF ABS(A(I,J))>1E-08 THEN 435
430 A(I,J)=0
435 NEXT J
440 NEXT I
450 FOR I=1 TO B
455 A(I,C)=0
460 NEXT I
470 A(R,C)=1
475 A(R,0)=C
500 H=H+1
530 REM ******************** IMPRESSION DES RESULTATS ***************************
540 PRINT " ********************  REPONSES  ****************************"
550 PRINT :PRINT " ITERATION NO ";H:PRINT :PRINT
600 PRINT "  VARIABLE  "      ,       "  VALEUR    " :PRINT
610 FOR I=1 TO B
620 IF A(I,0)=0 THEN 640
630 PRINT "     X ";A(254 A(I+L,0)=N+L+I
255 NEXT I
260 REM ******************** APPLICATION DE LA M.R.S ***************************
270 Q=0
280 FOR J=1 TO W
290 IF A(B,J)>=Q THEN 320
300 Q=A(B,J)
310 C=J
320 NEXT J
330 IF Q=0 THEN 800
340 U=1E+30
345 R=0
346 FOR I=1 TO L+G
350 IF A(I,C)=<0 THEN 370
360 IF A(I,W+1)/A(I,C)>U THEN 370
365 U=A(I,W+1)/A(I,C)
367 R=I
370 NEXT I
378 IF R=0 THEN 800:P=A(R,C)
380 FOR J=1 TO W+1
385 A(R,J)=A(R,J)/P
390 NEXT J
395 FOR I=1 TO B
```

```
280 REM ************** INTRODUCTION DES MATRICES TECHNOLOGIQUES
285 FOR K=1 TO P0:IF H1>=1 THEN 350
288 COLOR 3,0:N(0)=0:CLS:LOCATE 2,27:PRINT " MATRICES TECHNOLOGIQUES"
290 FOR X=1 TO 500:X=X+1:NEXT X
295 IF H1<>0 THEN 350
300 LOCATE 5,10:PRINT " MATRICE NO ";K
310 LOCATE 7,10:PRINT " LE NOMBRE DE VARIABLES ";:INPUT N(K)
```

```
315 LOCATE 10,10:PRINT " INTRODUIRE LES ELEMENTS DE LA MATRICE  "
320 FOR I=1 TO RO
330 FOR J=1 TO N(K)
335 LOCATE 12,10*I-7:PRINT"ligne No=";I
340 LOCATE 13+J,10*I-8:PRINT "E(";K;",";I;",";J;") =";:INPUT E(K,I,J)
342 NEXT J
347 NEXT I
350 FOR J=1 TO RO
355 FOR I=1 TO N(K)
360 C(I)=C(I)+B(J)*E(K,J,I):PRINT "C";C(I):INPUT T
365 NEXT I
380 NEXT J
382 GOTO 420
385 REM ****************          CALCUL DE CVJ          ********************
390 FOR J=U-RO TO U
392 V(U+1,J)=0
396 FOR I=1 TO U
400 V(U+1,J)=V(U+1,J)+B(I,U+J+1)*B(U+1,I)
410 NEXT I
412 NEXT J
415 GOTO 280
420 GOSUB 1500
425 NEXT K
435 IF Z(R2)>=0 THEN 4000
437 REM ********************       CALCUL DE Cb       ************************
438 C2(J)=0
440 FOR J=1 TO U
443 IF J<>R2 THEN 456
446 FOR K3=1 TO N(R2)
450 C2(J)=C2(J)+A1(W1(R2),K3)*B(U+K3+1,H1+1):PRINT "a1";A1(W1(R2),K3)
453 NEXT K3
454 PRINT "C2(";J;")=";C2(J):INPUT "TTT";T
456 NEXT J
460 REM ******************   CALCUL DE PRODUIT  Aj x Xj  ******************
464 FOR I460 REM ******************   CALCUL DE PRODUIT  Aj x Xj  ************
464 FOR I=1 TO U+1
466 B(I,0)=0
467 NEXT I
468 FOR J=1 TO RO
470 P2(I)=0
475 FOR K4=1 TO N(R2)
480 P2(J)=P2(J)+E(R2,J,K4)*B(U+1+K4,H1+1)
482 NEXT K4
483 B(J,0)=P2(J)
484 NEXT J
490 B(R2+1,0)=1
492 B(U+1,0)=Z(R2)
493 'FOR I=1 TO U+1
495 'NEXT I
510 REM ******************   CALCUL DE B(inverse)next   ******************
515 Q=1E+20
516 IF R1=0 THEN 520
518 IF R1=<RO THEN 1000
520 FOR I=1 TO U
525 PRINT "B(";I;",0)=";B(I,0):IF B(I,0)=<0 THEN 545
530 IF B(I,U+1)/B(I,0)>Q THEN 545
535 Q=B(I,U+1)/B(I,0)
540 R1=I
545 NEXT I
546 B(U+1,R1)=C2(R2)
550 IF R1>=-.5 THEN 573
555 PRINT " SOLUTION NO DEFENIE"
560 STOP
573 P1=B(R1,0)
580 B(R1,J)=B(R1,J)/P1:PRINT "B";B(R1,J)
```

```
585 NEXT J
590 FOR I=1 TO U
593 IF I=R1 THEN 620
597 IF B(I,0)=0 THEN 620
600 FOR J=1 TO U+1
610 B(I,J)=B(I,J)-B(R1,J)*B(I,0):PRINT "B(I,J)=";B(I,J):
615 NEXT J
620 NEXT I
630 REM ********  CALCUL DU PRODUIT  B(inverse)start x B(inverse)next  ******
635 FOR I=1 TO U
640 FOR J=2 TO U+1
645 C1(I,U+J)=0
650 FOR K4=1 TO U
655 C1(I,U+J)=C1(I,U+J)+B(I,K4)*B(K4,U+J)
660 NEXT K4
670 NEXT J
675 NEXT I
680 FOR I=1 TO U
685 FOR J=0 TO U
690 IF I<>J THEN 700
695 B(I,J)=1:GOTO 705
700 B(I,J)=0
705 NEXT J
708 NEXT I
710 B(U+1,0)=0
720 H1=H1+1
730 FOR I=1 TO U
740 FOR J=2 TO U+1
750 B(I,J+U)=C1(I,U+J)
760 PRINT "B(";I;",";J+U;")=";B(I,U+J)
770 NEXT J
780 NEXT I
900 INPUT "T=";T:GOTO 227
1000 REM *************  CALCUL DE Cb x B(inv) x Psj  *************
1005 PRINT
1010 REM ***************  CALCUL DE Cb x B(inv)  ****************
1020 C(J)=0
1030 FOR J=1 TO U
1040 FOR I=1 TO U
1050 C(J)=C(J)+B(I,U+J+1)*B(U+1,J)
1060 NEXT I
1070 FOR I=1 TO U
1080 PS(I)=0
1090 NEXT I
1000 NEXR1I=1
1100 PS(R1)=1
1110 Z(J)=Z(J)+PS(J)*C(J)
1120 NEXT J
1130 IF Z(U)>=Z(R2) THEN 520
1140 R1=R2
1150 B(0,R2)=R1:GOTO 680
1160 REM ******************  FIN DU CALCUL  ******************
1500 KEY OFF:CLS
1510 IF H1>=1 THEN 1900
1515 COLOR 2,0:LOCATE 2,2:PRINT"* * * * * * * * * * * * * * * * * * * *
* * * * * * * * * * * * * *"
1516 LOCATE 3,2:PRINT"* * * * * * * * * * * * * * * * * * * * * *
* * * * * * * * ":FOR I=1 TO 20:LOCATE 3+I,2:PRINT"* *":LOCATE 3+I,70:PRINT"* *"
EXT I
1520 LOCATE 22,2:PRINT"* * * * * * * * * * * * * * * * * * * * * *
 * * * * * * "
1521 LOCATE 23,2:PRINT"* * * * * * * * * * * * * * * * * * * * * *
 * * * * * * * *"
1525 LOCATE 10,15:PRINT"APPLICATION DE LA [illegible]
1530 LOCATE 15,10:PRINT "INTRODUIRE LES DONNEES DE SOUS SYSTEME NO
```

```
1540 X=0
1550 FOR I=1 TO 50
1560 X=X+1
1570 NEXT I
1590 LET C=1
1640 P5=0
1645 CLS
1900 REM ***********  INITIALISATION DU TABLEAU SIMPLEXE  ************
1910 PRINT "MAM";M(K):INPUT "Oh";T:M(K)=M(K)-1
1920 H=1
1930 FOR I=0 TO 20
1940 FOR J=1 TO 20
1950 A(I,J)=0
1960 NEXT J
1970 NEXT I
1975 GOSUB 5300
1980 '************************  FIN D'INITIALISATION  ***************************
1990 M(K)=M(K)-1
2164 PRINT "W(k)=";W(K):INPUT T:FOR  K1=1 TO M(K)+1
2167 A(K1-1,N(K)+G(K)+K1)=1
2170 A(K1-1,0)=K1+N(K)+G(K)
2175 NEXT K1
2180 IF E1(K)<>0 THEN 2200
2190 IF G(K)=0 THEN 2350
2200 FOR K1=L(K)+E1(K)+1 TO M(K)+1
2210 A(K1-1,K1+N(K)-L(K)-E1(K))=-1
2220 NEXT K1
2230 W(K)=W(K)+1
2240 Q=0
2250 FOR  J=1 TO N(K)+G(K)
2260 S=0
2270 FOR I=M(K)-G(K)-E1(K)+1 TO M(K)
2280 S=S+A(I,J)
2290 NEXT I
2300 A(W(K),J)=-S
2310 IF Q<A(I,J) THEN 2340
2320 Q=A(W(K),J)
2330 C=J
2340 NEXT J
2350 PRINT
2450 IF G(K)+E1(K)=0 THEN 2790
2460 IF Q=0 THEN 2860
2470 IF P5>0 THEN 2980
2480 H=H+1
2490 Q=1E+30
2501500 KEY OFF:CLS
1510 IF H1>=1 THEN 1900
1515 COLOR 2,0:LOCATE 2,2:PRINT"* * * * * * * * * * * * * * * * * * * * * * * *
* * * * * * * * * * * * *"
1516 LOCATE 3,2:PRINT"* * * * * * * * * * * * * * * * * * * * * * * * * * * * *
* * * * * * * ":FOR I=1 TO 20:LOCATE 3+I,2:PRINT"* *":LOCATE 3+I,70:PRINT"* *"
EXT I
1520 LOCATE 22,2:PRINT"* * * * * * * * * * * * * * * * * * * * * * * * * * * *
 * * * * * "
1521 LOCATE 23,2:PRINT"* * * * * * * * * * * * * * * * * * * * * * * * * * * *
 * * * * * * *"
1525 LOCATE 10,15:PRINT"APPLICATION DE LA METHODE DU SIMPLEXE GENERAL"
1530 LOCATE 15,10:PRINT "INTRODUIRE LES DONNEES DE SOUS-SYSTEME NO=";K
1540 X=0
1550 FOR I=1 TO 50
1560 X=X+1
1570 NEXT I
1590 LET C=1
1640 BEEP
```

```
1900 REM ***********  INITIALISATION DU TABLEAU SIMPLEXE  ***********
1910 PRINT "MAM";M(K):INPUT "Oh";T:M(K)=M(K)-1
1920 H=1
1930 FOR I=0 TO 20
1940 FOR J=1 TO 20
1950 A(I,J)=0
1960 NEXT J
1970 NEXT I
1975 GOSUB 5300
1980 '*********************  FIN D'INITIALISATIOM  *************************
1990 M(K)=M(K)-1
2164 PRINT "W(k)=";W(K):INPUT T:FOR  K1=1 TO M(K)+1
2167 A(K1-1,N(K)+G(K)+K1)=1
2170 A(K1-1,0)=K1+N(K)+G(K)
2175 NEXT K1
2180 IF E1(K)<>0 THEN 2200
2190 IF G(K)=0 THEN 2350
2200 FOR K1=L(K)+E1(K)+1 TO M(K)+1
2210 A(K1-1,K1+N(K)-L(K)-E1(K))=-1
2220 NEXT K1
2230 W(K)=W(K)+1
2240 Q=0
2250 FOR  J=1 TO N(K)+G(K)
2260 S=0
2270 FOR I=M(K)-G(K)-E1(K)+1 TO M(K)
2280 S=S+A(I,J)
2290 NEXT I
2300 A(W(K),J)=-S
2310 IF Q<A(I,J) THEN 2340
2320 Q=A(W(K),J)
2330 C=J
2340 NEXT J
2350 PRINT
2450 IF G(K)+E1(K)=0 THEN 2790
2460 IF Q=0 THEN 2860
2470 IF P5>0 THEN 2980
2480 H=H+1
2490 Q=1E+30
2500 R=-1
2510 FOR I=0 TO M(K)
2520 IF A(I,C)<=0 THEN 2560
2530 IF A(I,B1(K))/A(I,C)>Q THE2520 IF A(I,C)<=0 THEN 2560
2530 IF A(I,B1(K))/A(I,C)>Q THEN 2560
2540 Q=A(I,B1(K))/A(I,C)
2550 R=I
2560 NEXT I
2570 IF R>=-.5 THEN 2610
2580 PRINT "SOLUTION NON DEFINIE POUR CE SOUS-SYSTEME NO=";K
2600 GOTO 4000
2610 P=A(R,C)
2620 A(R,0)=C
2630 FOR J=1 TO B1(K)
2640 A(R,J)=A(R,J)/P
2650 NEXT J
2660 FOR I=0 TO W(K)
2670 IF I=R THEN 2740
2680 FOR J=1 TO B1(K)
2690 IF J=C THEN 2730
2700 A(I,J)=A(I,J)-A(R,J)*A(I,C)
2710 IF ABS(A(I,J))>1E-08 THEN 2730
2720 A(I,J)=0
2730 NEXT J
2740 NEXT I
2750 FOR I=0 TO W(K)
2760 A(I,C)=0
```

```
2770 NEXT I
2780 A(R,C)=1
2790 Q=0
2800 FOR J=1 TO N(K)+G(K)+L(K)
2810 IF A(W(K),J)>Q THEN 2840
2820 Q=A(W(K),J)
2830 C=J
2840 NEXT J
2850 GOTO 2460
2860 IF W(K)=M(K)+1 THEN 2890
2870 W(K)=W(K)-1
2880 GOTO 2790
2890 PRINT
2900 FOR I=0 TO M(K)
2910 IF A(I,0)<N(K)+L(K)+G(K)+1 THEN 2930
2920 IF A(I,B1(K))<>0 THEN 2940
2930 NEXT I
2940 GOTO 2970
2950 PRINT "PAS DE SOLUTION FAISABLE TROUVEE"
2960 GOTO 4110
2970 PRINT "REPONSES"
2980 PRINT
2990 IF Q=0 THEN 3010
3000 PRINT "BASE AVANT ITERATION";H
3010 PRINT "VARIABLE","VALEUR"
3020 FOR I=0 TO M(K)+1
3030 IF A(I,0)=0 THEN 3050
3032 IF A(I,0)>N(K) THEN  3050
3040 PRINT" X ";A(I,0);"=",A(I,B1(K))
3050 NEXT I
3055 IF Y=2 THEN  3075
3060 PRINT
3070 PRINT "VALEUR DE LA FONCTION OBJECTIVE",A(W(K),B1(K)):GOTO 3080
3075 PRINT "VALEUR DE LA FONCTION OBJECTIVE",-A(W(K),B1(K))
3080 Z(K)=-A(W(K),B1(K))+V(U+1,K+R0)
3082 PRINT "Z(";K;")=";Z(K):INPUT "t1",T
3085 IF Q<>0 THEN 2480
3100 IF Z(K)>=U1 THEN 3260
3110 U1=Z(K)
3120 R2=K
3130 K1=1
3130 K1=1
3140 FOR I=0 TO M(R2)+1
3150 IF A(I,0)>N(R2) THEN 3190
3160 IF A(I,0)<>K1 THEN 3190
3170 B(U+K1+1,H1+1)=A(I,B1(K)):B(0,H1+1)=R2
3180 K1=K1+1
3190 NEXT I
3210 IF K1<=N(R2) THEN 3240
3212 ALPHA=K1-1:FOR K1=1 TO ALPHA
3214 PRINT " B(";U+K1+1;",";H1+1;")=";B(U+K1+1,H1+1)
3216 NEXT K1
3218 INPUT "t2";T
3220 GOTO 3260
3240 B(U+K1+1,H1+1)=0
3250 K1=K1+1:GOTO 3140
3260 RETURN
4000 CLS:REM ***************** IMPRESSION DES RESULTATS  ************************
4010 R=H1:
4020 I=1
4130 FOR J=1 TO P
4150 FOR K=1 TO N(J)
4160 B1(U+1+K,I)=B(U+1+K,I)*B(J,U+1)
4175 LOCATE 5+K*2,15:PRINT "B1(";U+K+1;",";I;")=";B1(U+K+1,I)
4180 NEXT K
```

```
4190 I=I+1:INPUT "t=";T:IF I<=R THEN 4025
4200 NEXT J
4210 END
5300 IF H1>=1 THEN 7400
5664 LOCATE 13,30:PRINT "DONNER LE NOMBRE DE CONTRAINTES";:INPUT M1(K)
5666 COLOR 2,4:LOCATE 20,12:PRINT "VOULEZ VOUS CORRIGER  (O/N)":COLOR 2,0
5667 CC$=INKEY$:IF CC$="" THEN 5667
5668 IF ASC(CC$)=79 THEN 5664
5669 IF ASC(CC$)=78 THEN 5670 ELSE 5667
5670 CLS:LOCATE 13,30:PRINT "DONNER LE NOMBRE DE VARIABLES";:INPUT N1(K)
5671 COLOR 2,4:LOCATE 20,12:PRINT "VOULEZ VOUS CORRIGER  (O/N)":COLOR 2,0
5672 CC$=INKEY$:IF CC$="" THEN 5672
5673 IF ASC(CC$)=79 THEN 5670
5674 IF ASC(CC$)=78 THEN 5675 ELSE 5672
5675 CLS
5678 LOCATE 5,5:PRINT " NOMBRE D'EQUATIONS DE TYPE  =<  ";:INPUT L1(K)
5680 LOCATE 10,5:PRINT " NOMBRE D'EQUATIONS DE TYPE  =  ";:INPUT E2(K)
5685 LOCATE 15,5:PRINT " NOMBRE D'EQUATIONS DE TYPE >=  ";:INPUT G1(K)
5690 COLOR 2,4:LOCATE 20,12:PRINT "VOULEZ VOUS CORRIGER  (O/N)":COLOR 2,0
5692 CC$=INKEY$:IF CC$="" THEN 5690
5695 IF ASC(CC$)=79 THEN 5675
5700 IF ASC(CC$)=78 THEN 5710 ELSE 5690
5710 IF 5700 IF ASC(CC$)=78 THEN 5710 ELSE 5690
5710 IF L1(K)+E2(K)+G1(K)=M1(K) THEN 5725
5715 PRINT "LES DONNEES NE SONT PAS PRATICABLES."
5720 GOTO 5666
5725 B2(K)=M1(K)+N1(K)+G1(K)+1:W1(K)=M1(K)
5732 '*****************  INITIALISATION DU TABLEAU SIMPLEXE  ****************
5740 FOR I=0 TO W1(K)+1
5743 FOR J=1 TO B2(K)
5744 FOR K4=1 TO P0
5746 A2(K4,I,J)=0
5747 NEXT K4
5748 NEXT J
5750 NEXT I
5760 IF B2(K)*(W1(K)+1)<540 THEN 5790
5770 PRINT "LE PROBLEME EST TROP GRAND"
5790 IF B2(K)>30 THEN 5810
5800 IF W1(K)+1<30 THEN 6975
5810 PRINT "CHANGER LA DIMENSION D'ENTREE(NO 102) POUR A";W+1;"BY";B;"MATRICE"
5820 PRINT
5830 PRINT "SOIT L'INSTRUCTION 164 OU 168 OU LES DEUX DOIVENT ETRE CHANGEES"
5840 PRINT "COMME SUITE:"
5850 PRINT
5860 IF B2(K)<31 THEN 5880
5870 PRINT " 164 IF B>";B(K);"THEN 169"
5880 IF W1(K)+1<18 THEN 6975
5890 PRINT "168 IF W+1<";"THEN 180"
6975 CLS:REM ******************  INTRODUCTION DES DONNEES  ******************
6980 PRINT "M(K)=";M(K)
6982 FOR I=0 TO M1(K)-1
6985 LOCATE 8,20:PRINT "DONNER LES COEFS DE L'EQUATION";I+1
6986 PRINT
6990 FOR J=1 TO N1(K)
7000 LOCATE 10+2*J,6:PRINT "LE COEF DE X";J;:INPUT A2(K,I,J)
7005 NEXT J
7050 LOCATE 12+2*N1(K),6:PRINT "DONNER LE SECOND MEMBRE DE LA CONTRAINTE";I+1:
PUT A2(K,I,B2(K))
7055 PRINT
7060 NEXT I
7064 CLS:FOR I=1 TO 45 STEP 2:LOCATE 2,I+14:PRINT "*":LOCATE 8,I+14:PRINT "*":
XT I
7068 FOR I=1 TO 6:LOCATE 2+I,15:PRINT"*":LOCATE 2+I,59:PRINT"*":NEXT I
7070 COLOR 1,3:LOCATE 5,26:PRINT "LA FONCTION ECONOMMIQUE":COLOR 2,0
7095 PRINT
```

```
7097 FOR J=1 TO N1(K)
7100 LOCATE 9+J,10:INPUT A2(K,W1(K),J):A1(W1(K),J)=A2(K,W1(K),J)
7103 PRINT
7107 A2(K,W1(K),J)=C(J)-A2(K,W1(K),J)
7109 IF Y=2 THEN 7117
7112 A2(K,W1(K),J)=-A2(K,W1(K),J):GOTO 2120
7117 A2(K,W1(K),J)=A2(K,W1(K),J)
7120 NEXT J
7123 LOCATE 23,60:COLOR 2,4:PRINT" VOULEZ VOUS CORRIGER  (O/N)":COLOR 2,0
7125 CC$=INKEY$:IF CC$="" THEN 7125
7128 IF ASC(CC$)=79 THEN 6975
7130 IF ASC(CC$)=78 THEN 7500 ELSE 7123
7400 'FOR I=1 TO M1(K)+1
7420 'FOR J=1 TO N1(K)+1
7440 'A(I,J)=0:A(I,B2(K))=0:A(W1(K),J)=0
7460 'NEXT J
7480 'NEXT I
7500 REM ********************  AFFECTATION DES VARIABLES  ********************
7510 M(K)=M1(K):N(K)=N1(K):W(K)=W1(K):PRINT "CONT";M(K),"CONT1";W(K)
7520 B1(K)=B2(K):L(K)=L1(K):G(K)=G1(K):E1(K)=E2(K)
7530 REM ************************  FIN D'AFFECTATION  ************************
7540 FOR I=0 TO M(K)-1
7550 FOR J=1 TO N(K)
7560 A(I,J)=A2(K,I,J)
7580 A(W(K),J)=A2(K,W(K),J):PRINT "COEF";A(W(K),J)
7590 NEXT J

7600 A(I,B1(K))=A2(K,I,B1(K)):PRINT "COEF1"7590 NEXT J
7600 A(I,B1(K))=A2(K,I,B1(K)):PRINT "COEF1";A(I,B1(K)):INPUT T
7610 NEXT I
7620 RETURN
8000 DIM B(F1,X2+1):DIM E(10,10,10)
8100 DIM A1(20,30),B1(20)
8200 DIM C(20):DIM C1(20,20)
8300 DIM A2(20,20,20),A(20,20)
8600 RETURN
```

```
5 KEY OFF:CLS
10 '** ETUDE DES SYSTEMES LINEAIRES AVEC FONCTION QUADRATIQUE **
20 '*************** A L'AIDE DE LA METHODE DE TAMURA ***************
23 PRINT
27 PRINT
30 '***************  ENTREE DES DONNEES *************
40 PRINT "EPS=";:INPUT EPS
41 PRINT "ALPHA=";:INPUT ALPHA
42 PRINT "N=";:INPUT N
43 PRINT "K=";:INPUT K
44 PRINT "N1=";:INPUT N1
45 PRINT "M=";:INPUT M
46 PRINT "R=";:INPUT R
47 PRINT "NX=";:INPUT NX
50 FOR I=1 TO N
53 '*****  ENTREE DES MATRICES A1,B1,Ci  *******
60 GOTO 20000
78 PRINT " DONNER LA MATRICE D'EVOLUTION   A";I
80 FOR I1=1 TO N1
90 FOR I2=1 TO N1
100 INPUT A(I,I1.I2)
110 NEXT I2
120 NEXT I1
125 PRINT " DONNER LA MATRICE DE COMMANDE   B";I
130 FOR I1=1 TO N1
140 FOR I2 =1 TO M
150 INPUT B(I,I1.I2)
160 NEXT I2
170 NEXT I1
180 PRINT " DONNER LA MATRICE D'INTERACTION  C";I
190 FOR I1=1 TO N1
200 FOR I2=1 TO R
210 INPUT C(I,I1.I2)
220 NEXT I2
230 NEXT I1
240 FOR J=1 TO N
250 PRINT " DONNER LA MATRICE  L(";I;",";J;")"
260 FOR I1=1 TO R
270 FOR I2=1 TO N1
280 PRINT"L(";I;",";J;",";I1;",";I2;")=";:INPUT L(I,J,I1,I2)
290 NEXT I2
300 NEXT I1
310 NEXT J
315 DIM LANDA(5,5,5)
320 DIM ET(5,5,5)
328 DIM SL(5,5,5)
335 PRINT " DONNER LES MATRICES DE PONDERATION "
344 FOR K1=0 TO K-1
348 '**********************  MATRICE  Q  ************************
350 FOR I1=1 TO N1
360 FOR I2=1 TO N1
370 INPUT Q(I,K1,I1,I2)
380 NEXT I2
384 '*************** MATRICE Q ***********************
390 NEXT I1
391 'DIM V(N1,M)
392 FOR I1=1 TO N1
393 FOR I2=1 TO N1
394 V(I1,I2)=Q(I,K1,I1,I2)
395 NEXT I2
396 NEXT I1
397 GOSUB 10000
400 '**********************  MATRICE  R  ***************************
```

```
400 '********************** MATRICE R **************************
410 FOR I1=1 TO M
420 FOR I2=1 TO M
430 INPUT R(I.K1,I1,I2)
440 NEXT I2
450 NEXT I1
451 FOR I1=1 TO M
452 FOR I2=1 TO M
453 V(I1,I2)=R(I,K1,I1,I2)
454 NEXT I2
455 NEXT I1
456 GOSUB 10000
460 '*************** MATRICE S ***********************
470 FOR I1=1 TO R
480 FOR I2=1 TO R
490 INPUT S(I,K1,I1,I2)
500 NEXT I2
510 NEXT I1
511 FOR I1=1 TO R
512 FOR I2=1 TO R
513 V(I1.I2)=S(I,K1,I1,I2)
514 NEXT I2
515 NEXT I1
520 GOSUB 10000
530 NEXT K1
540 '**** LECTURE DES MATRICES P(I,K)=F1 ***********
550 FOR I1=1 TO N1
560 FOR I2=1 TO N1
570 INPUT F(I,I1,I2)
580 NEXT I2
590 NEXT I1
591 FOR I1=1 TO N1
592 FOR I2=1 TO N1
593 V(I1,I2)=F(I,I1,I2)
594 NEXT I2
595 NEXT I1
600 GOSUB 10000
610 '**** INITIALISATION DU VECTEUR LANDA ET P *****
620 FOR K1=0 TO K-1
630 FOR I1=1 TO R
640 INPUT LAN(I,K1,I1)
650 NEXT I1
660 FOR I1=1 TO N1
670 INPUT P(I,K1,I1)
680 NEXT I1
690 NEXT K1
700 FOR I2=1 TO N1
710 P(I,-1,I2)=0
720 NEXT I2
722 '******************** INTRODUCTION DE XO(I) ******************
724 FOR H1=1 TO N1
726 INPUT XO(I,H1)
728 NEXT H1
730 NEXT I
740 FOR I=1 TO N
750 '****************** OPTIMISATION INTERMEDIAIRE ******************
760 '*************** TRANSPOSITION DES MATRICES A,B et C ***************
770 '******************** TRANSPOSEE DE A ************************
780 FOR J1=1 TO W1
790 'DIM D(W1,W2)
800 FOR J1=1 TO W1
```

```
800 FOR J1=1 TO W1
810 FOR J2=1 TO W2
820 D(J1.J2)=A(I,J1,J2)
830 NEXT J2
840 NEXT J1
850 GOSUB 11000
860 FOR J1=1 TO W1
870 FOR J2=1 TO W2
880 AT(I.J1,J2)=D(J1,J2)
890 NEXT J2
900 NEXT J1
910 '********************  TRANSPOSEE DE  B  ***************************
920 W2=M
930 FOR J1=1 TO W1
940 FOR J2=1 TO W2
950 D(J1,J2)=A(I,J1,J2)
960 NEXT J2
970 NEXT J1
980 GOSUB 11000
990 FOR J1=1 TO W1
1000 FOR J2=1 TO W2
1010 BT(I.J1,J2)=D(J1,J2)
1020 NEXT J2
1030 NEXT J1
1040 '******************  TRANSPOSEE DE  C  ***************************
1050 W2=R
1060 FOR J1=1 TO W1
1070 FOR J2=1 TO W2
1080 D(J1.J2)=A(I,J1,J2)
1090 NEXT J2
1100 NEXT J1
1110 GOSUB 11000
1120 FOR J1=1 TO W1
1130 FOR J2=1 TO W2
1140 CT(J1,J2)=D(J1,J2)
1150 NEXT J2
1160 '*** INVERSION DES MATRICES DE PONDERATION ****
1170 FOR K1=0 TO K-1
1175 'DIM A(N1.N1),B(N1,N1)
1180 W=M
1190 FOR H1=0 TO W
1200 FOR H2=0 TO W
1200 FOR H2=0 TO W
1210 A(H1,H2)=R(I,K1,H1.H2)
1220 NEXT H2
1230 NEXT H1
1240 GOSUB 12000
1250 FOR H1=1 TO W
1260 FOR H2=1 TO W
1270 R(I,K1,H1,H2)=A(H1,H2)
1280 NEXT H2
1290 NEXT H1
1300 W=R
1310 FOR H1=1 TO W
1320 FOR H2=1 TO W
1330 A(H1,H2)=S(I,K1,H1.H2)
1340 NEXT H2
1350 NEXT H1
1360 GOSUB 12000
1370 FOR H1=1 TO W
1380 FOR H2=1 TO W
1390 S(I,K1,H1,H2)=A(H1.H2)
1400 NEXT H2
```

```
1410 NEXT H1
1412 B=K1+1
1415 IF B>K-1 THEN 1540
1420 W=N
1430 FOR H1=1 TO W
1440 FOR H2=1 TO W
1450 A(H1,H2)=Q(I,K1,H1.H2)
1460 NEXT H2
1470 NEXT H1
1480 GOSUB 12000
1490 FOR H1=1 TO W
1500 FOR H2=1 TO W
1510 Q(I,K1,H1,H2)=A(H1,H2)
1520 NEXT H2
1530 NEXT H1
1540 '*** CALCUL DU PRODUIT R(invers)xB(transposée)***
1550 FOR I1=1 TO M
1560 FOR I2=1 TO M
1570 A(I1,I2)=R(I,K1,I1,I2)
1580 NEXT I2
1590 NEXT I1
1600 FOR I1=1 TO M
1610 FOR I2=1 TO N1
1620 B(I1,I2)=BT(I,I1,I2)
1630 NEXT I2
1640 NEXT I1
1650 W3=M:W5=M:W4=N1
1660 GOSUB 13000
1670 FOR I1=1 TO M
1680 FOR I2=1 TO N1
1690 GC(I,K1,I1,I2)=H(I1,I2)
1700 NEXT I2
1710 NEXT I1
1720 '*** CALCUL DU PRODUIT S1(invers)xC1(transposée)***
1730 FOR I1=1 TO R
1740 FOR I2=1 TO R
1750 A(I1,I2)=S(I,K1,I1,I2)
1760 NEXT I2
1770 NEXT I1
1780 FOR I1=1 TO R
1790 FOR I2=1 TO N1
1800 B(I1,I2)=CT(I,I1,I2)
1810 NEXT I2
1820 NEXT I1
1830 W3=R
1840 W4=N1
1850 W5=R
1860 GOSUB 13000
1870 FOR I1=1 TO R
1880 FOR I2=1 TO N1
1890 GD(I,K1,I1,I2)=H(I1,I2)
1900 NEXT I2
1910 NEXT I1
1915 V=1
1918 V1=1
1920 GOSUB 14000
1930 NEXT K1
1940 '*********************  CALCUL DE X1(K)  **********************
1950 W3=N1
1960 W4=1
1970 W5=N1
1980 FOR I1=1 TO N1
1990 FOR I2=1 TO N1
```

```
2000 A(I1.I2)=F(I,I1,I2)
2010 NEXT I2
2020 NEXT I1
2030 FOR I1=1 TO N1
2040 FOR I2=1 TO W4
2050 B(I1,I2)=P(I,K-1,I1)
2060 NEXT I2
2070 NEXT I1
2080 GOSUB 13000
2090 FOR I1=1 TO N1
2100 FOR I2=1 TO W4
2110 X(I,K,I1)=H(I1,I2)
2120 NEXT I2
2130 NEXT I1
2140 '**************  CALCUL DU GRADIENT  DE M(p)  ****************
2145 DIM E(N1,N1,N1)
2150 FOR K1=0 TO K-1
2160 FOR H1=1 TO N1
2170 E(I,K1,H1)=0
2180 NEXT H1
2190 W3=N1
2200 W4=1
2210 W5=N1
2220 FOR H1=1 TO N1
2230 FOR H2=1 TO N2
2240 A(H1,H2)=A(I,H1,H2)
2250 NEXT H2
2260 FOR H3=1 TO W4
2270 B(H1,H2)=X(I,K1,H1)
2280 NEXT H3
2290 NEXT H1
2300 GOSUB 13000
2310 FOR H1=1 TO N1
2320 FOR H2=1 TO W4
2330 E(I,K1,H1)=E(I,K1,H1)+H(H1,H3)
2340 NEXT H3
2350 NEXT H1
2360 W5=M
2370 FOR H1=1 TO W4
2380 FOR H2=1 TO W5
2390 A((H1,H2)=B(I,K1,H2)
2410 FOR H3=1 TO W4
2420 B(H2,H1)=U(I,K1,H2)
2430 NEXT H3
2440 NEXT H2
2450 NEXT H1
2460 GOSUB 13000
2470 FOR H1=1 TO N1
2480 FOR H2=1 TO W4
2490 E(I,K1,H1)=E(I,K1,H1)+H(H1,H2)
2500 NEXT H2
2510 NEXT H1
2520 W5=R
2530 FOR H1=1 TO W3
2540 FOR H2=1 TO W5
2550 A(H1,H2)=C(I,H1.H2)
2560 FOR H3=1 TO W4
2570 B(H2,H3)=Z(I,K1.H2)
2580 NEXT H3
2590 NEXT H2
```

```
2600 NEXT H1
2610 GOSUB 13000
2620 FOR H1=1 TO N1
2630 FOR H2=1 TO W4
2640 E(I,K1,H1)=E(I,K1,H1)+H(H1,H2)
2650 NEXT H2
2670 GL(I,K1,H1)=-X(I,K1,H1)
2680 E(I,K1,H1)=E(I,K1,H1)+GL(I,K1,H1)
2690 NEXT H1
2700 NEXT K1
2710 W=N*K
2720 DIM Y(W)
2730 FOR K1=1 TO N1
2740 FOR K1=1 TO N1
2750 M(I,K1,H1)=E(I,K1,H1)
2760 NEXT H1
2770 NEXT K1
2780 T=0!
2790 FOR K1=0 TO K-1
2795 I1=I2+T
2800 FOR I2=1 TO N1
2810 Y(I1)=M(I,K1,I2)
2820 NEXT I2
2830 T=N1
2840 NEXT K1
2850 DF=0!
2860 FOR I1=1 TO W
2870 DG(I1)=Y(I1)*Y(I1)
2880 DF=DF+DG(I1)
2890 NEXT I1
2900 DR=SQR(DF)
2910 IF DR<=EPS THEN 3085
2920 IF V <> 1 THEN 2972
2930 FOR K1=0 TO K-1
2940 FOR H1=1 TO N1
2950 D(K1,H1)=E(I,K1,H1)
2960 NEXT H1
2965 NEXT K1
2970 GOTO 2985
2972 W8=0:W9=K-1:W10=N1:W11=1
2980 GOSUB 16000
2985 FOR K1=0 TO K-1
2990 FOR H1=1 TO N1
3000 O(K1,H1)=ALPHA*D(K1,H1)
3010 P(I,K1,H1)=P(I,K1,H1)+O(K1,H1)
3020 D(K1,H1)=E(I,K1,H1)
3030 NEXT H1
3040 NEXT K1
3060 IF V>NX THEN 3090
3070 V=V+1
3075  FOR K1=0 TO K-1
3080 GOTO 1920
3085 NEXT I
3086 GOTO 3110
3090 PRINT" IL Y A TROP DE CALCULS LE GRADIENT NE CONVERGE PAS AU BOUT DE";NX
ITERATION"
3100 GOTO 3920
3110 '************ CALCUL DU GRADIENT DE FI(LANDA)  ***************
3120 FOR K1=0 TO K-1
3130 FOR I=1 TO N
3140 '****************  CALCUL DE SIGMA L1i  X1   ******************
3150 FOR H1=1 TO R
3160 D(H1)=0
```

```
3170 NEXT H1
3180 FOR J=1 TO N
3190 W3=R
3200 W4=1
3210 W5=R
3220 FOR H1=1 TO W3
3230 FOR H2=1 TO W5
3240 A(H1,H2)=L(J,I,H1,H2)
3250 NEXT H2
3260 FOR H3=1 TO W4
3270 B(H1,H3)=X(J,H3)
3280 NEXT H3
3290 NEXT H1
3300 GOSUB 13000
3310 FOR H1=1 TO W3
3320 FOR H2=1 TO W4
3330 D(H1)=-(D(H1)+H(H1,H2))
3340 NEXT H2
3350 NEXT J
3360 FOR H1=1 TO R
3370 E(I,K1,H1)=Z(I,K1,H1)+D(H1)
3380 NEXT H1
3390 NEXT I
3400 W=N*R
3410 DIM Y(W)
3420 FOR I=1 TO N
3430 FOR H1=1 TO R
3440 M(I,H1)=E(I,K1,H1)
3450 NEXT H1
3460 NEXT I
3470 T=0
3480 FOR I1=1 TO N1
3490 I1=I2+T
3500 Y(I1)=M(I,I2)
3510 NEXT I2
3520 T=R
3530 NEXT I
3540 DF=0
3550 FOR I1=1 TO W
3560 DG(I1)=Y(I1)^2
3570 DF=DF+DG(I1)
3580 NEXT I1
3590 DR(K1)=SQR(DF)
3595 NEXT K1
3597 TEST=0
3598 FOR K1=0 TO K-1
3600 IF DR(K1)<=EPS THEN 3770
3610 IF V1<>1 THEN 3661
3620 FOR I=1 TO N
3630 FOR H1=1 TO R
3640 D(I,H1)=E(I,K1,H1)
3650 NEXT H1
3655 NEXT I
3660 GOTO 3675
3661 W8=1:W9=N:W10=R:W11=1
3670 GOSUB 16000
3675 FOR I=1 TO N
3680 FOR H1=1 TO R
3690 O(I,H1)=ALPHA*D(I,H1)
3700 LANDA(I,K1,H1)=LANDA(I,K1,H1)+O(I,H1)
3710 D(I,H1)=E(I,K1,H1)
3720 NEXT H1
3730 NEXT I
3750 V1=V1+NX THE     90
```

```
3755 FOR I=1 TO N
3760 GOSUB 14000
3765 NEXT I
3770 TEST=TEST+1
3780 NEXT K1
3781 IF TEST=K THEN 3800
3782 FOR I=1 TO N
3784 GOTO 1940
3790 PRINT "LA METHODE NE CONVERGE PAS AU BOUT DE";NX;"iterations"
3795 GOTO 3920
3800 PRINT " SOLUTION OPTIMALE ATTEINTE AU BOUT DE";V1;"ITERATION"
3810 LOCATE 2,10
3820 '********** TRACE DES TRAJECTOIRES DE COMMANDE **********
3830 FOR I=1 TO N
3840 PRINT " TRACE DES TRAJECTOIRES DE COMMANDE U(";I;") "
3850 FOR H1=1 TO M
3860 FOR K1=0 TO K-1
3870 PSET(K1,U(I,K1,H1))
3880 NEXT K1
3890 CLS
3900 NEXT H1
3910 NEXT I
3920 END
```

BIBLIOGRAPHIE

[1] - M . SIMMONARD :
" PROGRAMMATION LINEAIRE " - Edition Dunod - 1962 -

[2] - F . DROESBEKE ; M . HALLIN ; CL . LEFEVRE
" PROGRAMMATION LINEAIRE PAR L'EXEMPLE " - Ellipse -1986-

[3] - P . CHRETHIENNE ; Y . PESQUEUX
" ALGORITHMES ET PRATIQUE DE LA PROGRAMMATION LINEAIRE "
- Edition Technique - 1980 -

[4] - MARYSE . QUERE
" PROGRAMMATION LINEAIRE " - Universite De Nancy II -
- I.U.T Informatique - 1979 -

[5] - A . TITLI ; MG . SINGH
" SYSTEMS : DECOMPOSITION , OPTIMISATION AND CONTROL "
- Pergamon .Press -

[6] - CH.V . FOURRIER
" LA SIMULATION DES SYSTEMES " - Edition Dunod - 1971 -

[7] - D . KIRK
" OPTIMAL CONTROL THEORY ; AND INTRODUCTION "

[8] - LUCAS . PUN
" INTRODUCTION A LA PRATIQUE DE L'OPTIMISATION " - Edition
Dunod -

[9] - M . BOUMAHRAT ; A . GOURDIN
" METHODES NUMERIQUES APPLIQUEES " - O.P.U - 1983 -

[10] - H . BUHLER
" REGLAGES ECHANTILLONNES "
Volume 2 : Traitement Dans L'espace D'etat
- Presse Plythecniques Romandes -

[11] - J.C . NEDELEC
" OPTIMISATION DANS R^n: THEORIES ET ALGORITHMES "
- Ecole National Des Points Et Chaussees - 1985 -

[12] - BOUDAREL
" COMMANDE OPTIMALE DES PROCESSUS " - Edition Dunod -

[13] - NASLIN
" CONDUITE OPTIMALE DES PROCESSUS " - Edition Dunod -

[14] TECHNIQUES DE L'INGENIEUR
" R 7000 - 7210 "

[15] - " METHODES NUMERIQUES D'ANALYSE DES SYSTEMES "
Tome 2 : Méthodes De Décomposition
- Revue IRIA chaier N° 11 . 06/ 1972 -

[16] - F . CHIGARA
" SIMULATION SUR ORDINATEUR D'UNE UNITE DE MONTAGE DU POINT DE VUE CONTROLE " - These De Magister -
Dept Electronique ENP 1987 -

[17] - Y . LAS DES PROCESSUS " - Edition Dunod -

[14] - TECHNIQUES DE L'INGENIEUR
" R 7000 - 7210 "

[15] - " METHODES NUMERIQUES D'ANALYSE DES SYSTEMES "
Tome 2 : Méthodes De Décomposition
- Revue IRIA chaier N° 11 . 06/ 1972 -

[16] - F . CHIGARA
" SIMULATION SUR ORDINATEUR D'UNE UNITE DE MONTAGE DU POINT DE VUE CONTROLE " - These De Magister -
Dept Electronique ENP 1987 -

[17] Y . LAS D'etudes . 06/1986 - Dept D'Electronique ENP