MONTPELLIER
IUT
INFORMATIQUE

Année 85/86

RAPPORT DE STAGE

AJUSTEMENT ET TRACE DE FONCTIONS

SUR IBM PC-AT

ET HP 7475A

Par Jean-Baptiste PUEL

Tuteurs : Patricio SOTO
Jean-Louis SALAGER

CNRS : Centre Louis EMBERGER
MONTPELLIER

REMERCIEMENTS

Je tiens à exprimer toute ma gratitude à mes tuteurs, Patricio SOTO et Jean-Louis SALAGER pour l'aide qu'ils m'ont apportée tout au long de mon stage, ainsi que pour leurs précieux conseils lors de la réalisation de ce rapport.

De plus, ma reconnaissance s'adresse à monsieur Di Castri et à l'ensemble du personnel du centre Louis EMBERGER qu'il dirige, pour la bonne ambiance et l'accueil dont j'ai bénéficié et qui ont rendu mon stage particulièrement agréable.

Je souhaite également remercier monsieur Gérard BANCO, qui a mis toute sa compétence à ma disposition lors des phases de programmation les plus délicates; ainsi que monsieur Marcel BOUCHE qui m'a permis d'utiliser son matériel.
Je remercie toute la section "Dessin" du centre Louis EMBERGER et plus particulièrement monsieur André CARRIERE, pour l'aide apportée lors de la réalisation des publications et de ce rapport.

Je tiens enfin à exprimer tout mes remerciements et toute mon amitié à Jean-Marc BERENGUIER dont les compétences m'ont si souvent été utiles.

SOMMAIRE

INTRODUCTION

Les chercheurs du centre Louis EMBERGER manipulent un grand nombre de données provenant des résultats de toutes sortes d'expérimentations.
Ces résultats, généralement contenus dans des fichiers ASCII, représentent un forme "brute" difficilement exploitable.
Aussi, Patricio SOTO et Jean-Louis SALAGER ont-ils réalisé un programme d'ajustement utilisant l'algorithme de NELDER & MEAD, et qui génère une fonction représentant fidèlement les point expérimentaux.
L'intéret de cette méthode est évident, car si une équation ne représente pas exactement un échantillon de points, elle est par contre nettement plus souple à utiliser, représenter, et commenter.
Tout naturellement, une représentation graphique des fonctions ainsi ajustées s'imposait, d'autant que le Centre EMBERGER dispose d'un traceur de courbes Hewlett Packard conçu pour ce type d'applications.
C'est donc la réalisation de cette application qui m'a été confiée.

Mon travail s'est déroulé en trois phases :

Dans un premier temps, j'ai été amené à modifier le programme d'ajustement NELDER-MEAD de façon à organiser le fichier de sortie d'une manière plus pratique pour un tracé ultérieur.

Ensuite, après un certain temps consacré à résoudre les problèmes de liaison entre le traceur et le PC-AT, j'ai pu me consacrer à l'élaboration du module de tracé de fonctions et de pointé d'un fichier. Il était dès lors possible de regrouper tous ces modules en seul programme : GRAF.

Mais pourquoi se limiter au seul tracé de fonctions en coordonnées Cartésiennes Implicites ? Il peut parfois être intéressant de tracer des courbes Polaires, ou bien en équations paramétriques, cette idée a donné lieu à un dernier programme, complet, GRAPHIX.

Ce travail a donné lieu à 2 publications, qui constituent le troisiemme chapitre.

PRESENTATION DU CNRS

Crée en 1939, le Centre National de la Recherche Scientifique est un établissement public national à caractère administratif, doté de la personnalité civile.

Placé sous la tutelle du ministère de la recherche, le CNRS est le principal organisme de recherche fondamentale en France.

Sa mission est "d'effectuer ou de faire effectuer, d'orienter, susciter, coordonner, évaluer, et développer des recherches présentant un intêret pour l'avancement de la science et le progrès économique et social, sur le plan national et international et de favoriser leurs applications".

En 1986, le CNRS regroupe 25.000 personnes dont 9.900 chercheurs, 15.100 ingénieurs, techniciens, et administratifs; il dispose de près de 1.300 unités de recherche propres ou associées pour un budget de 6 milliards de Francs.

Cet établissement comporte des laboratoires propres, mais il a également noué avec des formetions universitaires des liens tres étroits (aides individuelles, contrats, formation,...).

Ces relations entre le monde universitaire et celui de la recherche constituent un fait important dans le développement de la recherche et de l'éducation.

Présentation du Centre Louis EMBERGER :

Crée en 1961, le centre Louis EMBERGER a pour objectifs principaux de :

- comprendre les lois d'adaptation des communautés végétales à leur environnement.
- Voir comment l'homme peut modifier ces structures pour en tirer le meilleur profit.

Ce centre , qui constituait un unique laboratoire propre au CNRS s'est vu récemment divisé en trois unités.

L'informatique au CNRS est constituée d'une grappe d'ordinateurs "CORAIL" et de nombreux micro-ordinateurs, APPLE et IBM PC.
De plus, des terminaux reliés sur le CNUSC permettent de travailler sur le système IBM 3081, sous TSO, et permettent d'utiliser des logiciels comme S.A.S.

Présentation du matériel utilisé :

Mon stage à consisté principalement en la réalisation d'un logiciel qui devait assurer la continuité avec d'autres programmes existants. Aussi, j'ai du l'écire sur la même machine et sous le même système d'exploitation, afin d'éviter des problèmes de transfert de fichiers.
Ce logiciel est prévu pour fonctionner sur un IBM PC, ou compatible, mais a été développé sur un PC AT dont la vitesse d'exécution permet une utilisation plus souple.
Par conséquent, il est implanté sous le PC-DOS version 3.10, et est écrit sous BASICA version 3.10.
Cependant, il peut passer sans problème sur n'importe quelle machine de la série PC.
Le traceur, HP 7475A est un plotter tres classique, à 6 couleurs de feutres, et qui accepte les formats de papier A4 et A3. Il dispose d'un jeu d'instructions important qui permettent de simplifier la plupart des tracés. Les instructions sont de la forme : Mnémonique, Paramètre, Terminateur.

Présentation du langage utilisé :

Parmi tous les langages à ma disposition, je n'ai retenu que Pascal et Basic pour les facilites de mise au point qu'ils offraient.
Cependant, le Pascal (Microsoft ou Turbo) ne s'est pas avéré satisfaisant pour l'interfaçage avec le traceur, et j'ai du me tourner vers Basic. Malgré tout les defauts connus du Basic, comme sa lenteur ou son manque de structure, il est apparu comme étant le seul langage apte à piloter facilement un périphérique.
La syntaxe est simple : OPEN COM1: ... et arguments.
Pour une émission vers le traceur : PRINT #3, instruction traceur
Pour la réception de données : INPUT A$,n,#3

AJUSTEMENTS ET TRACES DE COURBES SUR PC-AT IBM ET TRACEUR HP 7475A

Par: J-B PUEL, P SOTO & J-L SALAGER

CNRS Centre Louis EMBERGER
Route de Mende
BP 5051
34033 MONTPELLIER-Cedex

CHAPEAU

Modélisateur et analyste rencontrent des problèmes d'ajustement de courbes ou "fittage" dans des domaines aussi variés que la physique, l'économie, la physiologie, la démographie, etc . Ces spécialistes demanderont ensuite, tout naturellement à visualiser le résultat de leurs calculs et à en conserver la trace.

Les procédures réunies dans le logiciel décrit ci-dessous répondent à leurs besoins.

--- figure 1 ---

INTRODUCTION

Rappelons tout d'abord le problème posé par un ajustement. Il s'agit, étant donné un ensemble de points dans R^n, de déterminer les paramètres P d'une fonction de n-1 variables tels que l'écart entre les points et la fonction soit minimum. Cette fonction peut être imposée par une loi théorique, issue d'une hypothèse, ou bien encore choisie a priori.

Différents algorithmes permettent d'obtenir ce résultat. Celui proposé par NELDER & MEAD (1965) offre des avantages développés dans le chapitre 2. Le lecteur trouvera dans le chapitre 3 comment nous avons résolu les problèmes particuliers rencontrés tant au niveau du HARD comme les "standards" RS232, qu'au niveau du SOFT. Nous terminons enfin en donnant un exemple d'application. Mais commençons tout de suite par une présentation d'ensemble du logiciel.

Présentation de l'ensemble des menus :

MENU PRINCIPAL :

GRAPHIX :

Pointé d'un fichier	---> 1
Réalisation d'un ajustement	---> 2
Tracé d'une fonction	---> 3
Fin du travail	---> 4

La première option de ce menu, "Pointé d'un fichier" permet de charger un fichier de points et de les tracer.Ces fichiers de points peuvent être sous 2 formats différents :

- fichiers standards : ce sont des fichiers de données ne contenant que des coordonnées de points.

- fichiers NELDER-MEAD : ce sont les fichiers de sortie du programme d'ajustement. On peut les utiliser pour le pointé, bien qu'ils contiennent un grand nombre d'informations inutiles. (Valeur des paramètres, moyenne des carrés des résidus,...)

La seconde option permet de réaliser un ajustement selon l'algorithme "NELDER-MEAD". Cette option utilise un fichier de points expérimentaux. Il faudra ensuite saisir la fonction de départ ainsi que les valeurs intiales des paramètres.

La troisiemme option de ce menu est consacrée au tracé :
Elle permet de saisir une fonction, et éventuellement de charger d'un fichier sur disque les paramètres calcules lors de l'ajustement.

La dernière option de ce menu principal permet de quitter le programme, éventuellement en retournant au système.

Principe de la méthode d'ajustement

Rappelons tout d'abord les nombreux avantages qu'offre l'algorithme de NELDER & MEAD par rapport aux autres algorithmes disponibles.
Tout d'abord, cette méthode d'ajustement ne nécéssite pas la connaissance des dérivées partielles de la fonction, par conséquent elle évite à l'utilisateur des calculs fastidieux que d' autres algorithmes imposent avant chaque ajustement.
De plus, la qualité des paramètres calculés par cette méthode est excellente, et supporte parfaitement la comparaison avec les logiciels fonctionnant sur de tres gros systèmes, comme S.A.S, le standard en la matière.
Enfin, si on peut reprocher à NELDER-MEAD sa relative lenteur, elle est compensée par la sécurité qu'offre la bonne qualité de ses calculs.

Comme nous l'avons vu plus haut, le problème est de trouver les valeurs de p paramètres Pi, d'un modèle fPi (x) tel que la courbe représentant le modèle soit la plus proche possible de la courbe expérimentale Yj = f(xj).
Le critère de "proximité" entre les 2 courbes peut être apprécié de plusieurs façons. La méthode que nous utilisons consiste à minimiser les résidus, ou erreur d'évaluation de la fonction.
Dans le cas d'un ajustement de m points et 1 variable, la valeur du résidu est donnée par :

$$F(Pi) = \sum_{j} (fPi\ (xj - yj)^2)$$

Cette fonction F est progressivement minimisée à l'aide d'une procédure de recherche séquentielle.
L'intérêt de cette méthode réside dans le fait suivant : les variations de F sont suivies au fur et à mesure que la fonction approche de son minimum. 7 étapes sont nécessaires à la réalisation de l'ajustement, elles sont clairement indiquées en REM dans le programme, et illustrées dans l'algorithme de la figure 2.

--- figure 2 ---

Choix du matériel utilisé :

Ce logiciel est prévu pour fonctionner sur un IBM PC, ou compatible, mais a été développé sur un PC AT dont la vitesse d'exécution permet une utilisation plus souple.
Par conséquent, il est implanté sous le PC-DOS version 3.10, et est écrit sous BASICA version 3.10.
Cependant, il peut passer sans problème sur n'importe quelle machine de la série PC.
Le traceur, HP 7475A est un plotter tres classique, à 6 couleurs de feutres, et qui accepte les formats de papier A4 et A3. Il dispose d'un jeu d'instructions important qui permettent de simplifier la plupart des tracés. Les instructions sont de la forme : Mnémonique, Paramètre, Terminateur.

Choix du langage utilisé :

Parmi tous les langages à notre disposition, nous n'avons retenu que Pascal et Basic pour les facilités de mise au point qu'ils offraient.
Cependant, le Pascal (Microsoft ou Turbo) ne s'est pas avéré satisfaisant pour l'interfaçage avec le traceur, et nous avons du nous tourner vers Basic.
Malgré tout les défauts connus du Basic, comme sa lenteur ou son manque de structure, il nous est apparu comme étant le seul langage apte à piloter facilement un périphérique.
La syntaxe est simple : OPEN COM1: ... et arguments.
Pour une émission vers le traceur : PRINT #3, instruction traceur
Pour la réception de données : INPUT A$,n,#3

Problèmes rencontrés :

En premier lieu, la liaison avec le traceur s'est avérée épineuse dans la mesure ou le port RS 232 du PC AT ne répond pas exactement à la norme. Emission et réception se déroulent à peu près normalement, mais il est impossible de se fier au Handshaking matériel car le brochage sur le AT n'est pas standard.
Par conséquent, les messages émis par le traceur, comme "Données mal reçues" ou "Buffer plein", ne sont pas reçus correctement, et il s'en suit une perte de données.
Cet inconvenient est particulièrement grave, car le buffer du traceur ne contient que 1024 Octets, alors que l'ordinateur le remplit en moins d'une seconde. Bien entendu, le traceur ne peut dessiner à l'allure ou il reçoit les données, et il faut programmer un driver artificiel de temporisation, qui teste l'état de remplissage du buffer avant toute émission. Cette méthode permet de remplacer les signaux de ligne, qui seraient sensés remplir le même office, mais que le AT ne sait gérer correctement.

Un second problème délicat est celui du format des données.
Le AT émet bien entendu les chiffres sous le format réel, avec mantisse et exposant, alors que le traceur ne reconnait que le format décimal, avec 4 chiffres significatifs.
Il faut donc utiliser les PRINT USING avec un format identique à celui du traceur. Prévoir également un agrandissement dans le cas ou les données traitées nécessitent une précision supérieure à 4 chiffres significatifs, où une diminution lorsque les données dépassent les limites du traceur (32767).

EXEMPLE D'APPLICATION

L'exemple d'application suivant fait partie d'une série d'expériences menées, dans un verger prés de Montpellier, par G.SELLES sous la direction de A. BERGER, directeur de recherche au CNRS. Il correspond plus particulièrement à la récupération hydrique d'un plan de pêcher, cultivé en pot, aprés irrigation. L'instant t=0 correspond à l'instant d'irrigation. La durée de l'irrigation est d'une mn seulement (G SELLES (1985)).

Il s'agit d'introduire dans un modèle du fonctionnement hydrique du pêcher la cinétique de récupération du diamètre du tronc $\phi=f(t)$ ainsi que la vitesse de variation $d\phi/dt$ de celui-ci. Un capteur de déplacement disposé sur le tronc de l'arbre fruitier effectue cette mesure et une centrale d'acquisition de données enregistre une centaine de valeurs par jour. Nous nous trouvons donc en présence d'un ensemble de couples (ϕexpérimental,t), représentés en coordonnées cartésiennes sur le graphique 1; le temps t étant exprimé en heures décimales et ϕ en microns.

-- GRAPHIQUE 1 --

Nous voyons apparaitre la forme d'une sigmoide. Cette forme en "sigma" est assez fréquente dans les phénomènes de croissance. Elle traduit une croissance lente au début, suivie d'une croissance rapide,(phase réversible liée à la récupération hydrique) pour terminer à nouveau par une phase de croissance lente. Après quelques essais d'ajustements de sigmoides: modèle logistique et modèle de Gompertz en particulier, c'est la logistique généralisée passant par l'origine qui a donné le meilleur résultat, le point d'inflexion se situant dans notre cas près de l'origine (LEBRETON J-D & MILLIER C (1982)). Nous complétons la fonction en ajoutant, à la logistique

généralisée, de la cinétique de récupération, une composante "croissance" de forme linéaire. On obtient ainsi la fonction définitive suivante:

$$\phi = f(t) = \left(\frac{P_1}{P_2} \left(1 - \mathrm{Exp}((P_3 - 1) P_2 . t)\right)\right)^{\frac{1}{1-P_3}} + P_4 . t \qquad (1)$$

Le graphique 2 montre le résultat de cet ajustement.

-- GRAPHIQUE 2 --

La vitesse de croissance, dérivée de cette fonction, a pour expression:

$$\frac{d\phi}{dt} = P_2 \left(\frac{P_1}{P_2}\right)^{\frac{1}{1-P_3}} . (1 - \mathrm{EXP}((P_3 - 1).P_2\ t)) \quad . \ \mathrm{EXP}((P_3 - 1).P_2\ t) + P_4 \qquad (2)$$

Remarquons que le calcul de cette dérivée à partir des accroissements finis $\frac{\Delta \phi \text{ expérimentaux}}{\Delta t}$ (sur le graphique 1) amplifierait considérablement les erreurs. On trouverait alors une trés grande variation entre deux valeurs successives de la dérivée, variation pouvant même aller jusqu'à une inversion de signe dans quelques cas, par exemple pour les 2 points repérés. Il faudrait donc, si ce calcul était retenu, faire appel à d'autre procédés mathématiques de type moyenne mobile. La méthode que nous avons utilisée: ajuster la fonction, puis calculer sa dérivée supprime cet inconvenient.

Le graphique 3 montre la fonction ϕ exprimée en μm et sa dérivée ϕ' en μmh^{-1}

-- GRAPHIQUE 3 --

L'algorithme proposé par NELDER & MEAD demande une estimation des paramètres au départ. Dans le cas de la logistique généralisée, P_3 est fonction de la position du point d'inflexion, $\left(\frac{P_1}{P_2}\right)^{\frac{1}{1-}}$ est l'assymptote horizontale, P_4 pente de la croissance est le plus facile à estimer. Pour ajuster cette fonction aux 76 points, nous avons commencé avec l'estimation du quadruplet (P_1 ,P_2 ,P_3 ,P_4) suivante: (20,2,0.7,6). La convergence a été obtenue au bout de 230 itérations en 15 mn en BASIC interprété et a donné le résultat suivant (18.44,10.87,0.906,9.1) . Le temps d'exécution du tracé du graphique 3 a été de 2,5mn avec un pas de 0.3mm.

Dans cet exemple d'application, aucune preuve particulière, ne nous permet d'affirmer que la loi de variation du diamètre du tronc au cours d'une journée suit la fonction (1). Mais nous remarquons que cette fonction décrit trés bien le phénomène. Le modélisateur pourra donc avantageusement introduire dans son modéle mathématique cette fonction continue de lissage à la place de ses points expérimentaux. Ceci ne diminuera pas la précision des résultats, puisque les écarts entre les valeurs observées et celles données par la fonction sont le plus souvent inférieurs aux erreurs de mesure. Les avantages sont par contre importants: facilité de mise en équation et de programmation, continuité dans le calcul qui donne une plus grande souplesse dans le fonctionnement du modèle et diminue les risques de "pompage".

CONCLUSION :

Ce logiciel est tres utilisé et nous pourrions donner de nombreux exemples d'applications. Il a servi tout récemment pour les recherches conduisant à une thèse d'état, soutenue par Gérard FERRIERE, qui a nécessité une version mise au point par Patricio SOTO, version fonctionnant sur APPLE II, et comportant une sortie sur imprimante graphique.

REMERCIEMENTS :

Nous tenons à exprimer ici nos remerciements à monsieur Gérard BANCO pour sa contribution à la résolution des problèmes Hardware; à messieurs A. BERGER et J. ROY pour avoir bien voulu lire les parties de cet article les concernant et avoir apporté leur corrections, à monsieur A. CARRIERE de la section dessin du centre EMBERGER, et à monsieur F. SORRANTINO pour nous avoir fourni une première version (TRS 80) de l'algorithme NELDER-MEAD.

BIBLIOGRAPHIE :

FERRIERE G. , 1986 : Mouvements naturels des éléments dans une prairie : Quantification des échanges d'azote entre lombriciens, sol et plantes. Thèse d'Etat, Université Claude BERNARD, LYON I.

LEBRETON, J. D., MILLIER, C. 1982 : Modèles dynamiques déterministes en biologie

NELDER, J. A. , MEAD, R. 1965 : A Simplex method for function minimisation. In : The Computer Journal 7

SELLES G. , 1985 : Réponse de quelques paramètres physiologiques à la contrainte hydrique. DAA - ENSAM Montpellier.

```
10 REM
20 REM ------------------------------------------------------------------------------------------------------------
30 REM                                  ----- GRAPHIX -----
40 REM Par JB PUEL, P SOTO, JL SALAGER CNRS Louis EMBERGER    Montpellier
50 REM Copyright (C) 1986
60 REM
70 REM ------------------------------------------------------------------------------------------------------------
80 REM
90 REM
100 ON ERROR GOTO 6300
110 REM Menu
120 KEY OFF : KEY 1,"GOTO 500"+CHR$(13) : KEY 3,"GOTO 4220"+CHR$(13)
130 CLS
140 ST1$=CHR$(201)+STRING$(76,205)+CHR$(187)
150 ST2$=CHR$(200)+STRING$(76,205)+CHR$(188)
160 LL$=CHR$(186)+STRING$(76,32)+CHR$(186)
170 LOCATE 2,2 : PRINT ST1$
180 FOR I=2 TO 21
190 LOCATE I+1,2 : PRINT LL$
200 NEXT
210 LOCATE 22,2 : PRINT ST2$;
220 REM
230 REM
240 LOCATE 6,34 : COLOR 1 : PRINT "GRAPHIX :" : COLOR 4
250 LOCATE 8,19 : PRINT "CNRS  Centre Louis EMBERGER  Montpellier"
260 LOCATE 11,21 : PRINT "Tracé d'une fonction            ---> 1"
270 LOCATE 13,21 : PRINT "Pointé d'un fichier             ---> 2"
280 LOCATE 15,21 : PRINT "Réalisation d'un ajustement     ---> 3"
290 LOCATE 17,21 : PRINT "Fin du travail                  ---> 4"
300 LOCATE 20,40 : PRINT "Votre choix :"
310 X$="" : WHILE X$="" : X$=INKEY$ : WEND
320 LOCATE 20,65 : PRINT X$
330 IF VAL(X$)=0 THEN 310
340 ON VAL(X$) GOTO 360,770,4040,3940
350 REM
360 CLS
370 LOCATE 5,10 : COLOR 1 : PRINT"Tracé d'une fonction ajustée :" : COLOR 4
380 LOCATE 8,2 : PRINT "Saisie de la fonction :"
390 LOCATE 10,2 : PRINT "Saisir la fonction entre les lignes 500 et 550"
400 LOCATE 11,2 : PRINT "Puis presser la touche ";: COLOR 11 : PRINT "F1" : COLO
R 4
410 LOCATE 13,2 : PRINT "Fonction actuelle :"
420 PRINT : PRINT
430 LIST 500-550
440 REM
450 REM
460 REM
470 REM
480 REM
490 REM
500 :
510 :
520 :
530 :
540 :
550 DEF FNY(X)=((H(1)/H(2)-(H(1)/H(2)*EXP((-1+H(3))*H(2)*X)))^((1/(1-H(3))))+H(4
)*X)
560 CLS
570 LOCATE 3,5 : PRINT "Utilisez-vous des paramètres d'ajustement ? (O/N) "
580 X$="" : WHILE X$="" : X$=INKEY$ : WEND
590 LOCATE 3,56 : PRINT X$
```

```
620 GOTO 580
630 LOCATE 5,2 : INPUT "Nom du fichier contenant les paramètres ? ",FP$
640 OPEN "I",#1,FP$
650 FOR I=1 TO 11 : LINE INPUT #1,A$ : NEXT
660 INPUT #1,A$
670 N=VAL(RIGHT$(A$,1))
680 FOR I=1 TO N
690 INPUT #1,A$
700 H(I)=VAL(RIGHT$(A$,LEN(A$)-17))
710 NEXT
720 TYPFN=1
730 GOTO 1160 : REM Tracé
740 REM
750 REM Tracé des points issus d'un ajustement Nelder-Mead
760 REM
770 CLS
780 TYPFN=2
790 DIM X(300),Y(300) : REM Dim arbitraire, 300 maximum présumé
800 LOCATE 5,23 : COLOR 1 : PRINT "Pointé d'un fichier" : COLOR 4
810 LOCATE 7,2 : INPUT"Nom du fichier à charger ?  ",FCH$
820 IF FCH$="" THEN 130
830 LOCATE 9,2 : PRINT "Fichier Nelder-Mead ? (O/N) "
840 X$="" : WHILE X$="" : X$=INKEY$ : WEND
850 LOCATE 9,31 : PRINT X$
860 IF X$="O" THEN ND=1 : GOTO 890
870 IF X$="N" THEN ND=0 : GOTO 890
880 GOTO 840
890 OPEN "I",#1,FCH$
900 IF ND=0 THEN 1060
910 WHILE A$<>"VALEURS CALCULEES ET ECARTS :"
920 LINE INPUT #1,A$
930 WEND
940 LINE INPUT #1,A$
950 LINE INPUT #1,A$
960 A$=""
970 P=0
980 WHILE NOT EOF(1)
990 P=P+1
1000 INPUT #1,A
1010 INPUT #1,X(P),Y(P)
1020 INPUT #1,A : INPUT #1,A
1030 WEND
1040 GOTO 1160 : REM Tracé
1050 REM
1060 A$="" : P=0
1070 WHILE NOT EOF(1)
1080 P=P+1
1090 INPUT #1,X(P),Y(P)
1100 WEND
1110 REM
1120 REM
1130 REM
1140 REM
1150 REM
1160 CLS
1170 LOCATE 1,23 : PRINT " Tracé de courbes sur Plotter HP "
1180 LOCATE 3,2 : PRINT " Couleur de la plume ?     " : LOCATE 3,26 : COLOR 11 :
PRINT "2" : COLOR 4
1190 X$="" : WHILE X$="" : X$=INKEY$ : WEND
1200 LOCATE 3,26 : COLOR 11 : PRINT X$ : COLOR 4
1210 PN=VAL(X$)-(2*(X$=CHR$(13))) : REM Si X$=return, PN=2
1220 LOCATE 5,2 : PRINT " Agrandissement sur X ? "
1230 LOCATE 5,27 : COLOR 11 : PRINT "1" : COLOR 4
1240 LOCATE 5,27 : COLOR 11 : INPUT "",AGX$ : COLOR 4
```

```
1260 LOCATE 5,67 : COLOR 11 : PRINT "1" : COLOR 4
1270 LOCATE 5,67 : COLOR 11 : INPUT "",AGY$ : COLOR 4
1280 AGX=VAL(AGX$) : AGY=VAL(AGY$)
1290 IF AGX$="" THEN AGX=1 : IF AGY$="" THEN AGY=1
1300 IF TYPFN<>2 THEN 1360
1310 REM
1320 REM Si Nelder-Mead, le paramètre est l'indice du tableau
1330 P1$="1" : P2$=STR$(P) : PAS=1
1340 GOTO 1470
1350 REM
1360 LOCATE 7,2 : PRINT " Valeur inférieure du paramètre ?            "
1370 LOCATE 7,36 : COLOR 11 : INPUT" ",P1$ : COLOR 4
1380 IF P1$="" THEN 1360
1390 LOCATE 8,2 : PRINT " Valeur supérieure du paramètre ?            "
1400 LOCATE 8,36 : COLOR 11 : INPUT" ",P2$ : COLOR 4
1410 IF P2$="" THEN 1390
1420 IF VAL(P2$)<=VAL(P1$) THEN 1360
1430 IF TYPFN<> 2 THEN PAS=((VAL(P2$)-VAL(P1$))/100)
1440 REM
1450 INST$="PD" : SM$=""
1460 REM Le tracé se fait plume basse, SM$ est le marqueur de point.
1470 IF TYPFN=2 THEN INST$="PU" : SM$="x"
1480 REM Ici, tracé plume haute, le marqueur est "x" (modifiable)
1490 REM Pour un tracé plume basse, INST$="PD"
1500 REM
1510 REM Calcul des bornes XMI, XMA, YMI et YMA
1520 REM
1530 GOSUB 3620
1540 REM
1550 REM
1560 IF TYPFN=2 THEN PAS=1 : GOTO 1610
1570 LOCATE 15,2 : PRINT " Pas du tracé ? " : LOCATE 15,19 : COLOR 11 : PRINT "0
.01" : COLOR 4
1580 LOCATE 15,18 : COLOR 11 : INPUT " ",PAS$ : COLOR 4
1590 PAS=VAL(PAS$)
1600 IF PAS$="" THEN PAS=.01
1610 LOCATE 15,40 : PRINT " Tracé des axes ? (O/N)        " : AX$="" : WHILE AX$=
"" : AX$=INKEY$ : WEND
1620 LOCATE 15,68 :COLOR 11 : PRINT AX$ : COLOR 4 : IF AX$<>"O" AND AX$<>"N" THE
N 1610
1630 LOCATE 17,2 : PRINT " Titre du tracé ? "
1640 LOCATE 17,21 : COLOR 11 : INPUT "",TITRE$ : COLOR 4
1650 LOCATE 19,2 : PRINT " Légende, axe des X ? "
1660 LOCATE 19,25 : COLOR 11 : INPUT "",LEGX$ : COLOR 4
1670 LOCATE 20,2 : PRINT " Légende, axe des Y ? "
1680 LOCATE 20,25 : COLOR 11 : INPUT "",LEGY$ : COLOR 4
1690 IF AX$="N" THEN 1740
1700 LOCATE 22,2 : PRINT " Unité de graduation - axe des X ? "
1710 LOCATE 22,38 : COLOR 11 : INPUT "",UX : COLOR 4
1720 LOCATE 23,2 : PRINT " Unité de graduation - axe des Y ? "
1730 LOCATE 23,38 : COLOR 11 : INPUT "",UY : COLOR 4
1740 LOCATE 24,2 : PRINT " OK ? (O/N) "
1750 X$="" : WHILE X$="" : X$=INKEY$ : WEND
1760 LOCATE 23,15 : COLOR 11 : PRINT X$ : COLOR 4
1770 IF X$="O" THEN 1800
1780 IF X$="N" THEN 1160
1790 GOTO 1750
1800 REM
1810 BORNE$=XMN$+","+XMM$+","+YMN$+","+YMM$+";"
1820 REM
1830 ET$=CHR$(3)
1840 OPEN "COM1:9600,N,8,1,CS0,DS0,CD0"AS #3
1850 PRINT #3,"IN;IP;";ET$;
1860 GOSUB 2500
1870 REM
```

```
1890 REM
1900 IF LEGX$="" AND LEGY$="" AND TITRE$="" THEN 1930
1910 GOSUB 2760
1920 REM
1930 GOSUB 2920 : REM Fixe les limites matérielles
1940 GOSUB 2500
1950 REM
1960 REM Définition du cadre
1970 PRINT #3,"SC";BORNE$;ET$;
1980 REM
1990 REM Tracé des axes
2000 REM
2010 IF AX$="O" THEN GOSUB 2960
2020 GOSUB 2500
2030 REM
2040 REM Positionnement sur le premier point
2050 REM
2060 X=VAL(P1$)
2070 GOSUB 2600
2080 PRINT #3,"PU;PA"; : PRINT #3,USING"#####.####";XX; : PRINT #3,","; : PRINT
#3,USING"#####.####";YY; : PRINT #3,";";ET$;
2090 REM
2100 REM Dernieres initialisations
2110 REM
2120 PRINT #3,"SP";PN;";";ET$;
2130 REM
2140 REM Boucle principale
2150 REM
2160 PRINT #3,INST$;ET$;
2170 FOR X=VAL(P1$) TO VAL(P2$) STEP PAS
2180 REM Premier test de tempo
2190 PRINT #3,CHR$(27);".B";ET$;
2200 A$="" : B$=""
2210 WHILE B$<>CHR$(13) : B$=INPUT$(1,#3) : A$=A$+B$ : WEND
2220 IF VAL(A$)<512 THEN GOSUB 2520
2230 REM Test erreurs traceur
2240 PRINT #3,"OE;";ET$
2250 A$="" : B$=""
2260 WHILE B$<>CHR$(13) : B$=INPUT$(1,#3) : A$=A$+B$ : WEND
2270 IF VAL(A$)<>0 THEN ERROR 200
2280 GOSUB 2570
2290 PRINT #3,"PA"; : PRINT #3,USING"#####.####";XX; : PRINT #3,","; : PRINT #3,
USING"#####.####";YY; : PRINT #3,";SM";SM$;";";ET$;
2300 NEXT X
2310 PRINT #3,"SPO;";ET$;
2320 CLOSE
2330 ERASE X,Y
2340 GOTO 130 : REM Retour au menu
2350 REM
2360 REM
2370 REM Pointage Nelder-Mead
2380 REM
2390 REM Tri du tableau
2400 REM
2410 XMI$=STR$(X(1)) : XMA$=XMI$ : YMI$=STR$(Y(1)) : YMA$=YMI$
2420 FOR I=1 TO P
2430 IF X(I)<VAL(XMI$) THEN XMI$=STR$(X(I))
2440 IF X(I)>VAL(XMA$) THEN XMA$=STR$(X(I))
2450 IF Y(I)<VAL(YMI$) THEN YMI$=STR$(Y(I))
2460 IF Y(I)>VAL(YMA$) THEN YMA$=STR$(Y(I))
2470 NEXT
2480 GOTO 2140 : REM Les tableaux sont initialisés, début du tracé
2490 REM
2500 REM Boucle de tempo pour le tracé
2510 REM
2520 A$="" : B$=""
```

```
2530 PRINT #3,CHR$(27);".B";ET$;
2540 WHILE B$<>CHR$(13) : B$=INPUT$(1,#3) : A$=A$+B$ : WEND
2550 IF VAL(A$)>512 THEN RETURN ELSE GOTO 2520
2560 RETURN
2570 REM
2580 REM Calcul des valeurs à tracer
2590 REM
2600 ON TYPFN GOSUB 2630,2710
2610 XX=XX*AGX : YY=YY*AGY
2620 RETURN
2630 REM
2640 REM Cartésiennes implicites
2650 REM
2660 XX=X
2670 YY=FNY(X)
2680 RETURN
2690 REM
2700 REM Récupération des points Nelder-Mead
2710 REM
2720 XX=X(X)
2730 YY=Y(X)
2740 RETURN
2750 REM
2760 REM
2770 REM Ecriture des labels
2780 REM
2790 PRINT #3,"SP";PN;";";ET$;
2800 IF LEGX$="" THEN 2830
2810 NI=INT(47-(LEN(LEGX$)/2))
2820 PRINT #3,"PA552,386;CP";NI;",-.95;LB";LEGX$;ET$;
2830 IF LEGY$="" THEN 2860
2840 NI=INT(31-(LEN(LEGY$)/2))
2850 PRINT #3,"PA400,386;DIO,1;CP";NI;",.95;LB";LEGY$;ET$;
2860 IF TITRE$="" THEN 2920
2870 NI=INT(25-(LEN(TITRE$)/2))
2880 PRINT #3,"PA552,7300;DI1,0;SR1.3,2.6;CP";NI;",.5;LB";TITRE$;ET$;
2890 PRINT #3,"SR.75,1.5;";ET$;
2900 RETURN
2910 REM
2920 PRINT #3,"IP1000,1000,10000,7000;";ET$;
2930 REM
2940 RETURN
2950 REM
2960 REM
2970 REM Routine de tracé des axes
2980 REM
2990 L=VAL(XMA$)-VAL(XMI$)
3000 H=VAL(YMA$)-VAL(YMI$)
3010 PRINT #3,"SP1;PUO,O;";ET$;
3020 X=0
3030 PRINT #3,"PD;";ET$;
3040 WHILE X<VAL(XMA$)
3050 PRINT #3,"PA";X;",O;XT;";ET$;
3060 PRINT #3,"PU;CP-1,-1;";ET$;
3070 PRINT #3,"LB";STR$(X);ET$;
3080 PRINT #3,"PU;PA";X;",O;PD;";ET$;
3090 X=X+UX
3100 WEND
3110 GOSUB 2500
3120 PRINT #3,"PD";XMA$;",O;";ET$;
3130 PRINT #3,"PU;PR"; : PRINT #3,USING"###.##";(L/-100); : PRINT #3,","; : PRIN
T #3,USING"###.##";(H/-120); : PRINT #3,"PD;PA";XMA$;",O;";
3140 PRINT #3,"PD";XMA$;",O;";ET$;
3150 PRINT #3,"PU;PR"; : PRINT #3,USING"###.##";(L/-100); : PRINT #3,","; : PRIN
T #3,USING"###.##";(H/-120); : PRINT #3,"PD;PA";XMA$;",O;";
```

```
T #3,USING"###.##";(H/120); : PRINT #3,"PD;P/";XMA$;",0;";ET$;
3170 PRINT #3,"PU0,0";ET$;
3180 GOSUB 2500
3190 X=0
3200 PRINT #3,"PD";ET$;
3210 WHILE X>VAL(XMI$)
3220 PRINT #3,"PA";X;",0;XT;";ET$;
3230 IF X=0 THEN 3270
3240 PRINT #3,"PU;CP-1,-1;";ET$;
3250 PRINT #3,"LB";STR$(X);ET$;
3260 PRINT #3,"PU;PA";X;",0;PD;";ET$;
3270 X=X-UX
3280 WEND
3290 GOSUB 2500
3300 PRINT #3,"PD";XMI$;",0;";ET$;
3310 PRINT #3,"PU0,0;";ET$;
3320 Y=0
3330 PRINT #3,"PD;";ET$;
3340 WHILE Y<VAL(YMA$)
3350 PRINT #3,"PA0,";Y;";YT;";ET$;
3360 PRINT #3,"PU;CP-6,0;";ET$;
3370 PRINT #3,"LB";STR$(Y);ET$;
3380 PRINT #3,"PU;PA0,";Y;";PD;";ET$;
3390 Y=Y+UY
3400 WEND
3410 GOSUB 2500
3420 PRINT #3,"PD0,";YMA$;";";ET$;
3430 PRINT #3,"PU;PR"; : PRINT #3,USING"###.##";(L/-130); : PRINT #3,","; : PRIN
T #3,USING"###.##";(H/-75); : PRINT #3,"PD;PA0,";YMA$;";";
3440 PRINT #3,"PU;PR"; : PRINT #3,USING"###.##";(L/130); : PRINT #3,","; : PRINT
 #3,USING"###.##";(H/-75); : PRINT #3,"PD;PA0,";YMA$;ET$;
3450 PRINT #3,"PU0,0;";ET$;
3460 Y=0
3470 PRINT #3,"PD;";ET$;
3480 WHILE Y>VAL(YMI$)
3490 PRINT #3,"PA0,";Y;";YT;";ET$;
3500 IF Y=0 THEN 3540
3510 PRINT #3,"PU;CP-6,0;";ET$;
3520 PRINT #3,"LB";STR$(Y);ET$;
3530 PRINT #3,"PU;PA0,";Y;";PD;";ET$;
3540 Y=Y-UY
3550 WEND
3560 GOSUB 2500
3570 PRINT #3,"PD0,";YMI;";";ET$;
3580 RETURN
3590 REM
3600 REM Calcul de XMI$, XMA$, YMI$ et YMA$
3610 REM
3620 X=VAL(P1$) : GOSUB 2600 : XMI$=STR$(XX) : XMA$=XMI$
3630 YMI$=STR$(YY) : YMA$=YMI$
3640 FOR X=VAL(P1$) TO VAL(P2$) STEP PAS
3650 GOSUB 2600
3660 IF XX<VAL(XMI$) THEN XMI$=STR$(XX)
3670 IF XX>VAL(XMA$) THEN XMA$=STR$(XX)
3680 IF YY<VAL(YMI$) THEN YMI$=STR$(YY)
3690 IF YY>VAL(YMA$) THEN YMA$=STR$(YY)
3700 NEXT
3710 IF TYPFN=1 THEN XMI$=P1$ : XMA$=P2$
3720 LOCATE 10,2 : PRINT " Valeur inférieure - axe des X ? "; : COLOR 11 : PRINT
 XMI$ : COLOR 4
3730 LOCATE 10,35 : COLOR 11 : INPUT"",X1$ : COLOR 4
3740 IF X1$<>"" THEN XMI$=X1$
3750 LOCATE 11,2 : PRINT " Valeur supérieure - axe des X ? "; : COLOR 11 : PRINT
 XMA$ : COLOR 4
3760 LOCATE 11,35 : COLOR 11 : INPUT"",X2$ : COLOR 4
```

```
3780 LOCATE 12,2 : PRINT " Valeur inférieure - axe des Y ? "; : COLOR 11 : PRINT
 YMI$ : COLOR 4
3790 LOCATE 12,35 : COLOR 11 : INPUT"",Y1$ : COLOR 4
3800 IF Y1$<>"" THEN YMI$=Y1$
3810 LOCATE 13,2 : PRINT " Valeur supérieure - axe des Y ? "; : COLOR 11 : PRINT
 YMA$ : COLOR 4
3820 LOCATE 13,35 : COLOR 11 : INPUT"",Y2$ : COLOR 4
3830 IF Y2$<>"" THEN YMA$=Y2$
3840 LX=(VAL(XMA$)-VAL(XMI$))/30 : LY=(VAL(YMA$)-VAL(YMI$))/30
3850 XMN$=STR$((INT(VAL(XMI$)*100))/100)
3860 XMM$=STR$((INT(VAL(XMA$)*100))/100)
3870 YMN$=STR$((INT(VAL(YMI$)*100))/100)
3880 YMM$=STR$((INT(VAL(YMA$)*100))/100)
3890 REM
3900 RETURN
3910 REM
3920 REM Sortie
3930 REM
3940 CLS
3950 LOCATE 10,35 : COLOR 1 : PRINT "Au revoir" : COLOR 4
3960 LOCATE 22,2 : PRINT "Retour au système ?  (O/N) "
3970 X$="" : WHILE X$="" : X$=INKEY$ : WEND
3980 LOCATE 22,31 : PRINT X$
3990 IF X$="N" THEN CLS : END
4000 IF X$="O" THEN CLS : SYSTEM
4010 GOTO 3970
4020 REM
4030 END
4040 REM
4050 REM     PROGRAMME  *** NELDER-MEAD ***
4060 REM     PROGRAMME D'AJUSTEMENT MULTIPARAMETRIQUE ET MULTIVARIABLE
4070 :
4080  CLS
4090  PRINT : PRINT "ENTRER LA FONCTION DE LA LIGNE 5500 A LA LIGNE 5550 SOUS  L
A FORME SUIVANTE,EX.:": PRINT
4100  PRINT "5500 Y=LOG(H(1)*X(1)+H(2))+H(3)*X(2) ..."
4110  PRINT : PRINT "H(1),H(2)....DIFFERENTS PARAMETRES A  AJUSTER"
4120  PRINT : PRINT "X(1),X(2)....DIFFERENTES VARIABLES": PRINT
4130  PRINT "PUIS PRESSER LA TOUCHE  ";: COLOR 11 : PRINT "F3" : COLOR 4
4140 PRINT TAB(14);"----------"
4150 PRINT"":PRINT""
4160 PRINT"Fonction déjà programmée":PRINT"":PRINT""
4170 LIST 5710-5760
4180 :
4190 :
4200 :
4210 :
4220 REM
4230 DIM X0(1,300),Y(300)
4240 DIM J$(20)
4250 REM     ENTREE DES VALEURS INITIALES DES PARAMETRES
4260  CLS
4270  INPUT "NOMBRE DE PARAMETRES = ";N
4280  DIM P(N,N + 3),F(N + 3),C(N),H(N)
4290  CLS
4300  PRINT  TAB( 5);"VALEURS INITIALES DES PARAMETRES": PRINT
4310  FOR I = 1 TO N
4320  PRINT "PARAMETRE ";I;"= ";: INPUT "";C(I)
4330  NEXT I
4340  REM  CREATION DE LA MATRICE DE PARAMETRES
4350 :
4360  FOR I = 1 TO N + 1
4370  FOR J = 1 TO N
4380  P(J,I) = C(J)
4390  NEXT J
```

```
4410  NEXT I
4420 PRINT : PRINT
4430  INPUT "NOM DU FICHIER DES DONNEES ENTREES : ";R$: PRINT
4440  INPUT "NOM DU FICHIER DE SORTIE : ";J$:PRINT
4450  OPEN "O", #1, J$
4460  OPEN "I", #2, R$
4470  GOSUB 6190
4480  REM   E T A P E  1
4490  REM   ------------------
4500  GOSUB 5880
4510  REM   E T A P E  2
4520  REM   ------------------
4530  REM   CALCUL DE M :NUMERO DE L'ENSEMBLE A ERREUR MAXIMALE
4540  REM              MO:NUMERO DE L'ENSEMBLE A ERREUR MINIMALE
4550 M = 1:MO = 1
4560  FOR I = 2 TO N + 1
4570  IF F(I) > F(M) THEN M = I
4580  IF F(I) < F(MO) THEN MO = I
4590  NEXT I
4600  GOSUB 5780
4610  REM  CALCUL DU POINT REFLECHI / CENTROIDE
4620 :
4630  FOR I = 1 TO N
4640 P(I,N + 2) = C(I) + (C(I) - P(I,M))
4650  NEXT I
4660 L = N + 2
4670  GOSUB 5940
4680  REM   E T A P E  3
4690  REM   ------------------
4700  IF F(N + 2) < F(MO) THEN  GOTO 4850
4710  IF F(N + 2) > F(M) THEN  GOTO 4980
4720 H = 1
4730  FOR I = 1 TO N + 2
4740  IF F(H) < F(I) AND I <  > M THEN H = I
4750  NEXT I
4760  FOR I = 1 TO N
4770 P(I,M) = P(I,N + 2)
4780  NEXT I
4790 F(M) = F(N + 2)
4800  IF H = 0 THEN 5110
4810  IF H <  > N + 2 THEN  GOTO 5110
4820  GOTO 4980
4830  REM  E T A P E  4
4840  REM  ------------------
4850  FOR I = 1 TO N
4860 P(I,N + 3) = C(I) + 2 * (P(I,N + 2) - C(I))
4870  NEXT I
4880 L = N + 3
4890  GOSUB 5940
4900  IF F(N + 3) > F(N + 2) THEN H = 0: GOTO 4760
4910  FOR I = 1 TO N
4920 P(I,M) = P(I,N + 3)
4930  NEXT I
4940 F(M) = F(N + 3)
4950  GOTO 5110
4960  REM   E T A P E  5
4970  REM   ------------------
4980  FOR I = 1 TO N
4990 P(I,N + 3) = C(I) - .5 * (C(I) - P(I,M))
5000  NEXT I
5010 L = N + 3: GOSUB 5940
5020  IF F(N + 3) < F(M) THEN 4910
5030  FOR I = 1 TO N + 1
5040  FOR J = 1 TO N
5050 P(J,I) = (P(J,I) + P(J,MO)) / 2
```

```
5070  NEXT I
5080  GOSUB 5880
5090  REM   E T A P E  6
5100  REM   ------------------
5110  GOSUB 6040
5120  REM  CRITERE D'ARRET
5130 :
5140  IF SD = SS THEN QQ = QQ + 1: GOTO 5180
5150 SS = SD
5160  IF QQ > 1 THEN QQ = 1
5170  GOTO 4550
5180  IF QQ < 10 THEN 4550
5190  REM   E T A P E  7
5200  REM   ------------------
5210  REM    IMPRESSION DES RESULTATS FINAUX
5220  ST$="*********************************"
5230  PRINT #1,: PRINT #1,: PRINT #1,: PRINT #1,: PRINT #1,
5240  PRINT #1,"FICHIER D'ENTREE : ";R$
5250  PRINT #1,"FICHIER DE SORTIE: ";J$
5260  PRINT #1, TAB( 20);"PARAMETRES OPTIMAUX"
5270  PRINT #1, ST$: PRINT #1,
5280  PRINT #1,"NOMBRE DE PARAMETRES :";N
5290  FOR I = 1 TO N
5300  PRINT #1, "PARAMETRE ";I;" : ";C(I)
5310  NEXT I
5320  PRINT #1,: PRINT #1, ST$
5330  PRINT #1, : PRINT #1,"MOYENNE DES CARRES D'ECART DES RESIDUS : ";EM/P
5340  PRINT #1,: PRINT #1, ST$ : PRINT #1,
5350  PRINT #1,"VALEURS CALCULEES ET ECARTS :": PRINT #1,
5360  PRINT #1, "NO PT";
5370  FOR K = 1 TO NX: PRINT #1, "      X(";K;")";: NEXT K
5380  PRINT #1,"       YEXP          YCAL        ERR"
5390  FOR I = 1 TO P
5400  FOR K=1 TO NX : X(K)=XO(K,I) : NEXT K
5410  GOSUB 5700
5420  PRINT #1, USING "###"; I;
5430  FOR K = 1 TO NX
5440  PRINT #1, USING " ####.### "; X(K);: NEXT K
5450  PRINT #1, USING " #####.#### #.#######^^^^ #.#######^^^^";Y(I), Y, Y-Y(I
)
5460  NEXT I
5470  REM   E T A P E  8
5480  REM   ------------------
5490  CLS
5500  CLOSE #1
5510  CLOSE #2
5520 PRINT TAB(20);"Paramètres optimaux "
5530 PRINT ST$:PRINT
5540 FOR I= 1 TO N
5550 PRINT"Paramètre ";I;" : ";C(I)
5560 NEXT I
5570 PRINT"MOYENNE DES CARRES DES RESIDUS= ";EM/P
5580 LOCATE 23,60 : PRINT "--->"
5590 X$="" : WHILE X$="" : X$=INKEY$ : WEND
5600 REM
5610 REM Retour au menu
5620 REM
5630 GOTO 100
5640 REM
5650  REM   CALCUL DE LA FONCTION POUR LE L
5660 IEME ENSEMBLE DE PARAMETRES AU POINT X
5670  REM  ***** DESCRIPTION DE LA FONCTION *****
5680 REM H(n) est le n iéme paramètre
5690 REM X(n) est la n iéme variable
5700 REM Ecrire la fonction de la ligne 1600 à 1650
```

```
5720 AB=H(1)/H(2)
5730 Y=(AB-(AB)*EXP(-MM*H(2)*X(1)))^(1/MM)+H(4)*X(1)
5740 :
5750 :
5760 :
5770 RETURN
5780 REM COORDONNEES DU CENTROIDE
5790 :
5800  FOR I = 1 TO N
5810 C(I) = 0
5820  FOR J = 1 TO N + 1
5830 C(I) = C(I) + P(I,J)
5840  NEXT J
5850 C(I) = (C(I) - P(I,M)) / N
5860  NEXT I
5870  RETURN
5880  REM   CALCUL DE L'ERREUR POUR TOUS LES ENSEMBLES DE PARAMETRES
5890 :
5900  FOR L = 1 TO N + 1
5910  GOSUB 5940
5920  NEXT L
5930  RETURN
5940  REM   CALCUL DE L'ERREUR POUR LE L IEME ENSEMBLE DE PARAMETRES
5950 :
5960 F(L) = 0
5970  FOR I = 1 TO N:H(I) = P(I,L): PRINT H(I): NEXT : PRINT " "
5980  FOR G = 1 TO P
5990  FOR K = 1 TO NX:X(K) = XO(K,G): NEXT K
6000  GOSUB 5700
6010 F(L) = F(L) + (Y - Y(G)) ^ 2
6020  NEXT G
6030  RETURN
6040  REM  CALCUL DE L'ECART TYPE DE L'ERREUR
6050 :
6060  REM  ERREUR MOYENNE
6070 EM = 0
6080  FOR I = 1 TO N + 1
6090 EM = EM + F(I)
6100  NEXT I
6110 EM = EM / (N + 1)
6120  PRINT : PRINT "MOY. DES CARRES DES RESIDUS ";EM / P
6130 SD = 0
6140  FOR I = 1 TO N + 1
6150 SD = SD + (F(I) - EM) ^ 2
6160  NEXT I
6170 SD = SD / (N + 1)
6180  RETURN
6190  REM   LECTURE DES POINTS EXPERIMENTAUX SUR DISQUETTE
6200 :
6210 INPUT #2,ID$
6220 PRINT #1,ID$
6230 P=1
6240  K=1 : NX=1
6250 INPUT #2,XO(K,P),Y(P)
6260 IF EOF(2) THEN 6290
6270 P=P+1
6280 GOTO 6250
6290  RETURN
6300 REM
6310 REM Traitement des erreurs
6320 REM
6330 IF (ERL=2330) THEN RESUME NEXT
6340 CLS
6350 IF (ERR=53) THEN MES$= " Je ne trouve pas ce fichier ... "  : GOTO 6420
6360 IF (ERR=71) THEN MES$= " Le disque n'est pas pret ... " : GOTO 6420
```

```
6380 IF (ERR=75) THEN MES$= " Erreur de chemin d'acces ... " : GOTO 6420
6390 IF (ERR=76) THEN MES$= " Chemin non trouvé ... " : GOTO 6420
6400 IF (ERR=200) THEN MES$= " Erreur traceur N° : "+A$+" en ligne : "+STR$(ERL)
6410 IF (ERR=200) AND (A$="3") THEN X=X+PAS : RESUME
6420 LOCATE 12,25 : PRINT MES$
6430 LOCATE 23,60 : PRINT "--->" : X$="" : WHILE X$="" : X$=INKEY$ : WEND
6440 IF (ERR=53) OR (ERR=71) OR (ERR=72) OR (ERR=75) OR (ERR=76) THEN RESUME 110
6450 RESUME NEXT
6460 END
```

MICRO
SYSTEME

AUTOMODIFICATION DE PROGRAMMES BASIC SUR IBM PC AT

par: J-B PUEL, J-L SALAGER & P SOTO
CNRS Centre Louis EMBERGER
route de Mende
BP 5051
34033 MONTPELLIER-Cedex

L'automodification d'un programme est un sujet souvent rencontré aujourd'hui en intelligence artificielle, en autodépannage et en d'autres domaines porteurs.

L'aspect que nous traitons ici concerne le langage BASIC sur un IBM PC AT. Lors du déroulement de sa séquence le programme écrit lui même de nouvelles lignes d'instruction. L'exemple d'application que nous proposons, concerne la saisie d'une instruction en INPUT, son introduction au bon endroit de la séquence et son exécution.

Nous donnons en annexe le programme de DUMP que nous avons réalisé, outil indispensable pour la mise au point de programmes nécéssitant de fouiller dans le niveau machine.

Nous nous proposons donc de saisir une fonction mathématique dans une chaine de caractères, que nous intègrerons au programme en tant que ligne d'instructions.
Il faut tout d'abord déterminer à quel endroit du programme la ligne doit être insérée car les pointeurs Basic du PC sont trés mobiles et dépendent de l'occupation de la mémoire.
C'est là le rôle du premier sous programme : il "balaie" la mémoire entre les deux adresses qui, par expérience, encadrent le mieux la zone recherchée. Ce sous programme recherche tout simplement les caractères correspondant à "FNY(", contenus dans l'expression DEF FN, qui réside à cet endroit.

La seconde phase de la modification concerne le codage, pour cela nous utiliserons deux tableaux de réels TFN et TFX.
Dans un premier temps, nous conserverons les parenthèses et les espaces ainsi que le caractère X qui représente la variable de la fonction. Ces caractères ne sont pas codés par l'interpréteur Basic, aussi, ils pourront être "Pokés" sans modifications.
Dans un second temps, vient le codage des fonctions mathématiques SIN, COS, TAN, ATN, LOG, EXP, SQR et ABS. Nous négligerons la fonction RND qui, bien qu'étant assimilée aux précédentes ne présente qu'un intérêt ludique. Le codage de ces fonctions est respectivement : 137, 140, 141, 142, 138, 139, 135 et 134 précédés par 255.
Ensuite, vient le codage des opérateurs mathématiques: +,-,*,/ et ^. Notre programme de Dump nous a permis de trouver leur codage: il s'agit de 233,234,235,236 et 237. Nous placerons donc ces valeurs dans le tableau TFN, à leur place respective.
Enfin, nous allons coder les numériques: pour cela nous utiliserons le tableau MARK qui servira à repérer la position des numériques dans le tableau TFN. Nous allons ensuite compacter le tableau TFN de sorte que les nombres soient contenus dans une seule case du tableau et non plus chiffre par chiffre, dans plusieurs cases.

Quant au codage des entiers, il faut examiner plusieurs cas:
Tout d'abord, le cas des entiers compris entre 0 et 9: on leur ajoute 17 et la somme ainsi obtenue est placée en mémoire.
Le second cas est celui des entiers compris entre 10 et 256: on conserve leur valeur sans modification, mais on les place aprés le chiffre 15 (ex. 15 131 représente 131).
Pour les entiers, positifs ou négatifs, compris entre -32768 et -256 et entre 256 et 32767, ils suivent un codage différent. Ces nombres sont décomposés en deux autres entiers, qui sont le DIV et le MODULO 256 de l'entier à coder (ex. 4 2 représente 1026). Là encore, nous allons mettre les nombres ainsi obtenus dans le tableau TFN. Le cas des entiers négatifs ne pose aucun problème, en effet, le signe - qui les précède est codé comme l'opérateur arithmétique de la soustraction (234).

La dernière phase de la modification est plus rapide : elle comprend tout d'abord la "mise en ordre" du tableau TFN (décalage

à gauche et suppression des 0), qui utilise un tableau intermédiaire : TFX, puis le "pokage" en mémoire de tout le contenu du tableau.

Nous prendrons une dernière précaution : placer après la fonction un bon nombre de '''', c'est à dire de REMs dans le but d'"écraser" tout reste d'une fonction précédente.

Remarquons, bien qu'il ne soit pas envisagé dans ce programme, le problème posé par le codage des réels. Le basic Microsoft code les réels sur 4 octets, respectivement 3 de mantisse et 1 d'exposant. Cette méthode de codage est nettement plus complexe puisqu'elle utilise la notion de développement en série.

Par exemple, pour coder 0.8125, nous l'écrirons:

1/2 + (1*1/4 + 0*1/8 + 1*1/16 + 0*1/32 + ...).

Ce développement s'arrête au niveau 23, c'est à dire pour 1/2^24, et il est clair que l'on peut le résumer par une série de 0 et 1 représentant un masque de bits. Le signe représente un 24° bit (0 = positif, et 1 = négatif). Ces 24 bits forment les 3 octets de la mantisse.

L'exposant est formé d'une manière plus simple, il représente la position du premier bit à 1 dans le développement de la mantisse, auquel on ajoute &H7F soit 127. Dans l'exemple ci-dessus, le premier bit à 1 est celui de 1/4, c'est à dire le bit 1. L'exposant sera donc 128.

Ce codage est trés délicat sur IBM PC, car l'interpréteur basic ne respecte pas l'ordre conventionnel pour placer les octets entre eux, et pour placer les bits à l'intérieur de chaque octet (poids fort en tête), ni pour placer les trois octets de mantisse entre eux (la aussi, poids fort en tête).

```
10 REM ----------------------------------------------------------------------
20 REM                    Programme de saisie d'une fonction
30 REM                              sur IBM PC
40 REM                    Par J-B PUEL, J-L SALAGER & P SOTO
50 REM                  CNRS Centre Louis EMBERGER, Montpellier
60 REM                        Copyright (C) Juin 1986
70 REM ----------------------------------------------------------------------
80 KEY OFF
90 REM
100 REM Recherche du futur emplacement de la fonction
110 REM
120 DEF SEG
130 FOR I=7000 TO 8000
140 IF PEEK(I)=151 THEN IF PEEK(I+1)=32 AND PEEK(I+2)=209 THEN 160
150 NEXT
160 DEB = I : SV = I
170 CLS
180 COLOR 1 : LOCATE 5,5 : PRINT "Automodification d'un programme BASIC " : CO
OR 4
190 LOCATE 7,2 : PRINT "Saisie d'une fonction :"
200 LOCATE 9,2 : PRINT "Voulez-vous charger une fonction du disque ?"
210 X$="" : WHILE X$="" : X$=INKEY$ : WEND
220 LOCATE 9,47 : PRINT X$
230 CHG=0
240 IF X$="N" THEN 320
250 IF X$<>"O" THEN 200
260 CHG=1
270 LOCATE 11,5 : INPUT "Nom de la fonction ? ",NOM$
280 IF NOM$="" THEN 170
290 NOM$="F-"+NOM$
300 BLOAD NOM$,DEB
310 GOTO 1220
320 LOCATE 11,5 : PRINT "Entrez la fonction sous la forme f(X) = 4 * X"
330 LOCATE 13,5 : PRINT "ou bien utilisez des parentheses : f(X) = (X*4)"
340 LOCATE 16,6 : PRINT" Y = F(X) = ";
350 LOCATE 16,17 : INPUT" ",FONCT$
360 REM Traitement de la fonction
370 LN=LEN(FONCT$)
380 DIM TFN(LN) : DIM MARK(LN)
390 REM On conserve les caracteres et les parentheses
400 FOR I=1 TO LN
410 C$=MID$(FONCT$,I,1)
420 IF C$="(" OR C$=")" OR C$="X" OR C$=" " THEN TFN(I)=ASC(C$)
430 NEXT
440 REM Codage des fonctions mathématiques
450 FOR I=1 TO LN
460 F$=MID$(FONCT$,I,3)
470 IF F$="ABS" THEN TFN(I)=255 : TFN(I+1)=134
480 IF F$="SQR" THEN TFN(I)=255 : TFN(I+1)=135
490 IF F$="SIN" THEN TFN(I)=255 : TFN(I+1)=137
500 IF F$="LOG" THEN TFN(I)=255 : TFN(I+1)=138
510 IF F$="EXP" THEN TFN(I)=255 : TFN(I+1)=139
520 IF F$="COS" THEN TFN(I)=255 : TFN(I+1)=140
530 IF F$="TAN" THEN TFN(I)=255 : TFN(I+1)=141
540 IF F$="ATN" THEN TFN(I)=255 : TFN(I+1)=142
550 NEXT
560 REM Codage des opérateurs
570 FOR I=1 TO LN
580 F$=MID$(FONCT$,I,1)
590 IF F$="+" THEN TFN(I)=233
600 IF F$="-" THEN TFN(I)=234
610 IF F$="*" THEN TFN(I)=235
```

```
 630 IF F$="^" THEN TFN(I)=237
 640 NEXT
 650 REM Codage des numériques
 660 REM Il faut d'abord isoler les nombres
 670 FOR I=1 TO LN
 680 IF VAL(MID$(FONCT$,I,1))<>0 THEN II=I : GOSUB 710
 690 NEXT
 700 GOTO 780
 710 FOR J=II TO LN
 720 IF VAL(MID$(FONCT$,J,1))=0 OR J=LN THEN GOSUB 750
 730 NEXT
 740 RETURN
 750 F$=MID$(FONCT$,II,(J-II)) : F=VAL(F$)
 760 TFN(II)=F
 770 RETURN
 780 I=I-1
 790 IF VAL(MID$(FONCT$,I,1))<>0 THEN TFN(I)=VAL(MID$(FONCT$,I,1))
 800 REM Compactage du tableau TFN
 810 FOR I=LN TO 1 STEP -1
 820 IF TFN(I)= ( TFN(I-1) MOD (10^(LEN(STR$(TFN(I-1)))-2))) THEN TFN(I)=0
 830 NEXT
 840 REM Recherche de la position des numériques
 850 FOR I=1 TO LN
 860 MARK(I)=0
 870 IF VAL(MID$(FONCT$,I,1))<>0 THEN MARK(I)=1
 880 NEXT
 890 FOR I=LN TO 1 STEP -1
 900 IF MARK(I)=1 AND MARK(I-1)=1 THEN MARK(I)=0
 910 NEXT
 920 REM Codage des numériques
 930 FOR I=1 TO LN
 940 IF MARK(I)=1 THEN GOSUB 970
 950 NEXT
 960 GOTO 1010
 970 IF (TFN(I)>0 AND TFN(I)<=9) THEN TFN(I)=17+TFN(I) : GOTO 1000
 980 IF (TFN(I)>9 AND TFN(I)<256) THEN TFN(I+1)=TFN(I) : TFN(I)=15
 990 IF TFN(I)>255 THEN TFN(I+1)=TFN(I) MOD 256 : TFN(I+2)=TFN(I) \ 256 : TFN(I
=28
 1000 RETURN
 1010 REM
 1020 REM Supression des 0
 1030 CT=0
 1040 FOR I=1 TO LN
 1050 IF TFN(I)=0 THEN CT=CT+1
 1060 NEXT
 1070 DIM TFX(LN-CT)
 1080 J=1
 1090 FOR I=1 TO LN
 1100 IF TFN(I)<>0 THEN TFX(J)=TFN(I) : J=J+1
 1110 NEXT
 1120 REM Pokage de la fonction
 1130 FOR I=1 TO LN-CT
 1140 POKE DEB+7+I,TFX(I)
 1150 NEXT
 1160 DEB=DEB+8+LN-CT
 1170 POKE DEB,32 : POKE DEB+1,58 : POKE DEB+2,143 : POKE DEB+3,217 : POKE DEB+
,39
 1180 FOR I=1 TO 50 : POKE DEB+4+I,39 : NEXT
 1190 PRINT
 1200 FIN=DEB+3+I
 1210 GOTO 1220 : REM Nécessaire pour forcer l'interpréteur à reconnaitre la fc
ction
 1220 DEF FNY(X)=SQR(ABS(X)) ??????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????????????
```

```
''''''''''''''''
1230 REM
1240 REM Essai de la fonction
1250 REM
1260 CLS
1270 KEY 1,"GOTO 1300"+CHR$(13)
1280 LOCATE 25,5 : PRINT "Pressez la touche " : COLOR 11 : LOCATE 25,23 : PRI
"F1" : COLOR 4
1290 PRINT : PRINT : LIST 1220
1300 CLS : PRINT " " : PRINT "          Essai de la fonction"
1310 PRINT : PRINT
1320 INPUT "Borne inférieure : ",B1
1330 INPUT "Borne supérieure : ",B2
1340 PRINT : PRINT "Valeur de la fonction entre ";B1;" et";B2
1350 PRINT
1360 FOR I=B1 TO B2
1370 PRINT " F(";I;") = ";FNY(I)
1380 NEXT
1390 X$="" : WHILE X$="" : X$=INKEY$ : WEND
1400 CLS
1410 IF CHG=1 THEN END
1420 LOCATE 5,5 : PRINT "Désirez-vous sauver votre fonction ?"
1430 X$="" : WHILE X$="" : X$=INKEY$ : WEND
1440 IF X$<>"N" AND X$<>"O" THEN 1420
1450 IF X$="N" THEN CLS : END
1460 LOCATE 7,2 : PRINT "Quel nom donnez-vous à la fonction ?"
1470 LOCATE 8,25 : PRINT SPACE$(30)
1480 LOCATE 8,5 : INPUT " (6 caracteres maxi) ",NOM$
1490 IF NOM$="" THEN 1400
1500 NOM$="F-"+NOM$
1510 IF LEN(NOM$)>8 THEN PRINT "Le nom est trop long ..." : GOTO 1470
1520 BSAVE NOM$,SV,(FIN-SV)
```

```
10000 REM DUMP, ce module est destiné à etre "mergé" en fin de programmes.
10010 KEY OFF : CLS
10020 LOCATE 5,1 : COLOR 1 : PRINT " A partir de quelle adresse (décimale) désir
ez-vous commencer le DUMP  "; : COLOR 4 : INPUT ADR
10030 LOCATE 6,1 : COLOR 1 : PRINT " Et sur combien de pages (276 octets)  "; :
COLOR 4 : INPUT LG : LG=(LG-1)*276
10040 LOCATE 8,5 : PRINT " Désirez-vous un écho ASCII ? "
10050 REP$="" : WHILE REP$="" : REP$=INKEY$ : WEND
10060 IF LEFT$(REP$,1)<>"O" AND LEFT$(REP$,1)<>"o" AND LEFT$(REP$,1)<>"N" AND LE
FT$(REP$,1)<>"n" THEN 10040
10070 IF LEFT$(REP$,1)="O" OR LEFT$(REP$,1)="o" THEN 10230
10080 CLS : LOCATE 25,1 : PRINT " DUMP mémoire à partir de :";ADR
10090 PAGE = 12*23 : NP=INT(LG/PAGE)+1
10100 FOR I = 1 TO NP
10110 CLS
10120 FOR J = 1 TO 23
10130 PRINT USING"##### ";ADR+((I-1)*23*12)+((J-1)*12); : PRINT ": ";
10140 FOR K = 1 TO 12
10150 AC=ADR+((I-1)*23*12)+((J-1)*12)+K-1
10160 PRINT USING"###   ";PEEK(AC);
10170 NEXT K
10180 NEXT J
10190 LOCATE 25,1 :COLOR 10 : PRINT "              RETURN Pour la page suivante.
" : COLOR 4
10200 X$="" : WHILE X$="" : X$=INKEY$ : WEND
10210 NEXT I
10220 GOTO 10440
10230 REM ECHO ASCII
10240 CLS
10250 LOCATE 25,1 : COLOR 1 : PRINT "  L'affichage se fait par moitiés de page.
" : COLOR 4
10260 PAGE = 6*23 : NP=INT(LG/PAGE)+2
10270 FOR I = 1 TO NP
10280 CLS
10290 FOR J = 1 TO 23
10300 PRINT USING"##### ";ADR+((I-1)*23*6)+((J-1)*6); : PRINT ": ";
10310 FOR K = 1 TO 6
10320 AC = ADR+((I-1)*23*6)+((J-1)*6)+K-1
10330 PRINT USING"###   ";PEEK(AC);
10340 NEXT K
10350 FOR K=1 TO 6
10360 AC = ADR+((I-1)*23*6)+((J-1)*6)+K-1
10370 GOSUB 10500 : REM TESTE LA VALIDITE DU CHR
10380 COLOR 10 : PRINT "  ";D$;"   "; : COLOR 4
10390 NEXT K
10400 NEXT J
10410 LOCATE 25,1 : COLOR 10 : PRINT "              RETURN Pour la page suivante
" : COLOR 4
10420 X$="" : WHILE X$="" : X$=INKEY$ : WEND
10430 NEXT I
10440 CLS
10450 LOCATE 5,5 : PRINT " Un autre DUMP ? "
10460 REP$="" : WHILE REP$="" : REP$=INKEY$ : WEND
10470 IF LEFT$(REP$,1)<>"O" AND LEFT$(REP$,1)<>"o" AND LEFT$(REP$,1)<>"N" AND LE
FT$(REP$,1)<>"n" THEN 10450
10480 IF LEFT$(REP$,1)="O" OR LEFT$(REP$,1)="o" THEN 10010
10490 END
10500 REM        TESTE SI CHR$(AC) EST AFFICHABLE
10510 D=PEEK(AC)
10520 IF D=0 OR D=7 OR D=9 THEN D$=" " ELSE IF D>=10 AND D<=13 THEN D$=" " ELSE
IF D>=28 AND D<=32 THEN D$=" " ELSE D$=CHR$(D)
10530 RETURN
```

CONCLUSION :

Ce stage au Centre Louis EMBERGER m'a permis de découvrir et d'apprécier le rôle et l'importance de l'informatique dans un organisme scientifique.

De plus, j'ai pu me familiariser avec l'IBM PC-AT, machine que je ne connaissais pas, et réaliser une application entière en BASIC, langage particulièrement performant mais descrédité actuellement.

Sur le plan professionnel, j'estime que ce stage est une expérience tres profitable dans la mesure où il m'a mis devant mes responsabilités. J'ai eu des délais à respecter, des contraintes auquelles il m'a fallu m'adapter; ce qui change beaucoup de l'optique que je pouvais avoir durant ma scolarité à l'IUT.

Enfin, j'ai beaucoup apprécié l'Informatique Scientifique, discipline que je connaissais peu et qui m'a changé de l'habituelle informatique de gestion qui constitue le fond des études à l'IUT.

Notice d'utilisation de GRAPHIX :

MENU PRINCIPAL :

GRAPHIX :

Cartésiennes implicites	--> 1
Cartésiennes paramétriques	--> 2
Polaires implicites	--> 3
Polaires paramétriques	--> 4
Pointé d'un fichier	--> 5
Réalisation d'un ajustement	--> 6
Opérations Disque	--> 7
Fin du travail	--> 8

Détail de ce menu :

Les 4 premières options, (options de tracé) débutent par la saisie de la fonction.
Si la fonction a déja été sauvegardée sur disque, il est possible de la rappeler; dans le cas contraire, il faut la saisir.
Cette fonction utilise peut-être des paramètres ou constantes réelles, auquel cas on doit les saisir au préalable sous le nom de H(1) à H(n). Lors de la saisie de la fonction, on rappelera ces constantes par leur nom.
Enfin, ces paramètres réels peuvent avoir été calculés par le programme d'ajustement Nelder-Mead, dans ce cas, ils ont été sauvegardés sur le fichier de sortie de l'ajustement. Il est possible de les lire directement sur le disque; après quoi on utilisera le nom habituel H(n) pour les désigner dans la fonction.
La dernière question posée par le programme permet de travailler avec la fonction courante. Cette option est particulièrement pratique lorsque l'on utilise plusieurs fois de suite la m@me fonction.

La seconde partie de ce menu principal est consacrée aux ajustements. On peut lancer le programme d'ajustement ou bien utiliser un fichier de données pour le pointer.
Les fichiers de données peuvent avoir 2 formats différents :
- fichiers Nelder-Mead : ce sont les fichiers issus d'un ajustement, ces fichiers contiennent les paramètres d'ajustement, les valeurs des point expérimentaux et calculés, ainsi que l'erreur commise lors de l'ajustement.

- fichiers standard : ce sont des fichiers de données ne contenant que des coordonnées de points.

L'option "Opérations Disque" permet de cataloguer une disquette, de détruire et de renommer une fonction sauvée sous GRAPHIX, et enfin de formatter une disquette (A ou B, aucun risque pour le disque dur !).

La dernière option de ce menu principal permet de quitter le programme, éventuellement en retournant au système.

VERSION SIMPLIFIEE : "GRAF"

Il est possible d'utiliser une version simplifiée de GRAPHIX, plus particulièrement consacrée aux ajustements NELDER-MEAD.
Cette version ne comporte qu'un type de tracé : en coordonnées Cartésiennes Implicites , qui permet le tracé d'une fonction ajustée et de sa dérivée.
Comme dans GRAPHIX, il est possible de récupérer les paramètres issus d'un ajustement.
Remarque : le module de pointé permet, bien entendu de pointer un fichier NELDER-MEAD, mais aussi de pointer un fichier quelconque saisi, par exemple, sous Professionnal Editor IBM.
Enfin, le programme d'ajustement est inclus à cette version de GRAPHIX, il est donc possible de chainer parfaitement une session de travail du type :

Pointé --> Ajustement --> Tracé de la fonction ajustée.

Description du Menu de Tracé

Ce menu est parfaitement identique pour GRAPHIX et GRAF, sa version simplifiée.
En premier lieu, le menu de tracé permet de sélectionner le couleur de la plume. Les codes sont les suivants :

1- Noir épais.
2- Noir fin (valeur par défaut).
3- Rouge.
4- Vert.
5- Bleu.
6- Mauve.

Ensuite, il faut choisir le coefficient d'agrandissement sur les axes X et Y. La valeur par défaut, 1, correspond à la dimmension normale.

Dans le cas d'un tracé de fonction, le programme demande ensuite les valeurs minimales et maximales du paramètre de la fonction, c'est à dire X pour une fonction cartésienne implicite, t pour une fonction paramétrique, et Téta (angle) pour une équation polaire.
Lors d'un pointé de fichier, les bornes sont recherchées automatiquement par le programme.

En fonction de ces paramètres, le programme optimise la fonction,

c'est à dire qu'il détermine quelles sont les valeurs extrêmes sur les axes X et Y. Ces valeurs sont admises par défaut, mais l'utilisateur peut les modifier à son gré. Attention : si l'on sélectionne un cadre de dimmensions inférieures aux extrema de la fonction, le traceur risque de se mettre en erreur, et de toute façons, le tracé ne se fera pas dans son intégralité.

Toujours dans le cas d'un tracé de fonctions, le programme demande ensuite quel sera le pas de calcul de la fonction. La valeur par défaut est de 1 centiemme d'unité, mais peut etre redéfinie.
Pour un tracé de bonne qualité, il faudra calculer environ 2000 points.

Ensuite, le programme propose le tracé des axes. Il n'y a pas de réponse par défaut.

L'utilisateur peut maintenant choisir le titre de son tracé ainsi que les légendes qui seront portées sur les axes X et Y. Ces labels sont tous trois facultatifs.

Le programme demande enfin quelles seront les unités portées sur les axes, si l'option "tracé des axes" a bien entendu été choisie.

Quelques remarques à propos du traceur :

- Si le voyant orange "ERROR" clignote en fin de tracé, il est nécessaire de réinitialiser le traceur, pour cela, presser simultanément les touches de traceur "ENTER" et "VIEW".

- Si une plume reste engagée dans le porte-plume, il est possible de la ramener dans le charriot : presser simultanement "ENTER" et le numéro de la plume.

- A cause de certains problèmes de liaison, il peut arriver que l'IBM provoque une erreur 57 ou "DEVICE I/O ERROR". On ne peut rien faire contre cela, il faut relancer le programme.

MODIFICATION POSSIBLES DE GRAPHIX :

Il est possible de changer quelques paramètres du programme, en particulier la vitesse de transmission, la vitesse de tracé, les valeurs par défaut, le symbole de pointé.

- Vitesse de transmission : fixée à 9600 bauds, mais on peut la réduire à volonté.
Modification de GRAPHIX, ligne : 4720
GRAF, ligne : 1840

- Vitesse de tracé : le programme utilise la vitesse de tracé standard, mais pour un tracé sur papier fin ou transparents, il faudra la réduire ajouter la ligne suivante au programme :
PRINT #3,"VS+paramètre;";ET$; en ligne 4735 sur graphix, ou en ligne 1855 sur GRAF. Le paramètre doit être un nombre compris entre 0 et 127. 0 correspond à la vitesse minimale : 0.38 cm/s
127 est la vitesse maximale : 38.1 cm/s (valeur par défaut).

- Symbole de tracé : fixé par défaut à "x". On peut le remplacer par n'importe quel caractère ASCII, comme "X" ou "*" par exemple. Pour cela, modifier l'affectation de la variable SM$ en ligne 4360 sur GRAPHIX ou 1470 sur GRAF.

- Pointé "Plume haute" ou Plume basse" : modifier l'affectation de INST$ en ligne 4360 sur GRAPHIX ou 1470 sur GRAF. PU signifie Plume haute et PD signifie Plume basse.

```
10 REM
20 REM ----------------------------------------------------------------------------------------------------
30 REM Programme de tracé de fonctions sur HP
40 REM Par JB PUEL
50 REM Copyright (C) 1986
60 REM
70 REM ----------------------------------------------------------------------------------------------------
80 REM
90 REM
100 CLOSE
110 ON ERROR GOTO 3660
120 GOTO 1790 : REM Le programme de tracé est après la saisie de la fonction
130 REM
140 REM Recherche du futur emplacement de la fonction
150 REM
160 DEF SEG
170 FOR I= 8000 TO 9000
180 IF PEEK(I)=209 THEN IF PEEK(I+1)=ASC(VAR$) THEN IF PEEK(I+2)=40 THEN 200
190 NEXT
200 DEB = I : SV = I
210 RETURN
220 REM
230 REM Saisie de la fonction
240 REM
250 CLS : LOCATE 5,5 : COLOR 1 : PRINT "Désirez-vous charger une fonction du dis
que ? "  : COLOR 4
260 CHG=0
270 X$="" : WHILE X$="" : X$=INKEY$ : WEND
280 IF X$<>"O" AND X$<>"N" THEN 250
290 IF X$="N" THEN 370
300 CHG=1 : REM Flag, permet de sauter la sauvegarde
310 LOCATE 8,2 : INPUT " Nom de la fonction à charger ",NOM$
320 IF NOM$="" THEN 250
330 NOM$="F-"+NOM$
340 BLOAD NOM$,DEB
350 GOTO 1590
360 REM
370 CLS : LOCATE 4,2 : PRINT "Saisie de la fonction :"
380 LOCATE 6,5 : PRINT "Entrez la fonction sous la forme f(X) = 4 * X"
390 LOCATE 7,5 : PRINT "ou bien utilisez des parentheses : f(X) = (X*4)"
400 LOCATE 9,2 : PRINT "Utilisez-vous des constantes réelles ?  (O/N)"
410 X$="" : WHILE X$="" : X$=INKEY$ : WEND
420 LOCATE 9,48 : PRINT X$
430 IF X$="N" THEN I=0 : GOTO 710
440 IF X$<>"O" THEN 410
450 LOCATE 10,2 : PRINT "Paramètres de Nelder-Mead ?  (O/N) "
460 X$="" : WHILE X$="" : X$=INKEY$ : WEND
470 LOCATE 10,38 : PRINT X$
480 IF X$="N" THEN 600
490 IF X$<>"O" THEN 460
500 LOCATE 11,2 : INPUT "Nom du fichier contenant ces paramètres ? ",FP$
510 OPEN "I",#1,FP$
520 FOR I=1 TO 11 : LINE INPUT #1,A$ : NEXT
530 INPUT #1,A$
540 N=VAL(RIGHT$(A$,1))
550 FOR I=1 TO N
560 INPUT #1,A$
570 H(I)=VAL(RIGHT$(A$,LEN(A$)-17))
580 NEXT
590 GOTO 680
600 LOCATE 13,2 : PRINT "Combien de constantes ? "
```

```
620 LOCATE 13,27 : PRINT X$
630 CT=VAL(X$)
640 FOR I=1 TO CT
650 LOCATE 14+I,5 : PRINT "Valeur de la constante H(";I;")  ? "
660 LOCATE 14+I,39 : INPUT"",H(I)
670 NEXT
680 LOCATE 14+I+2 : PRINT "Désirez-vous utiliser la fonction courante ?"
690 X$="" : WHILE X$="" : X$=INKEY$ : WEND
700 IF X$="O" THEN CHG=1 : GOTO 1600
710 LOCATE 14+I+3,6 : PRINT MODELE$;
720 LOCATE 14+I+3,18 : INPUT" ",FONCT$
730 IF FONCT$="" THEN 1600
740 REM Traitement de la fonction
750 LN=LEN(FONCT$)
760 DIM TFN(LN) : DIM MARK(LN)
770 REM On conserve les caractères et les parenthèses
780 FOR I=1 TO LN
790 C$=MID$(FONCT$,I,1)
800 IF C$="(" OR C$=")" OR C$="X" OR C$="H" OR C$=" " THEN TFN(I)=ASC(C$)
810 NEXT
820 REM Codage des fonctions mathématiques
830 FOR I=1 TO LN
840 F$=MID$(FONCT$,I,3)
850 IF F$="SIN" THEN TFN(I)=255 : TFN(I+1)=137
860 IF F$="COS" THEN TFN(I)=255 : TFN(I+1)=140
870 IF F$="ATN" THEN TFN(I)=255 : TFN(I+1)=142
880 IF F$="LOG" THEN TFN(I)=255 : TFN(I+1)=138
890 IF F$="EXP" THEN TFN(I)=255 : TFN(I+1)=139
900 IF F$="SQR" THEN TFN(I)=255 : TFN(I+1)=135
910 IF F$="TAN" THEN TFN(I)=255 : TFN(I+1)=141
920 NEXT
930 REM Codage des opérateurs
940 FOR I=1 TO LN
950 F$=MID$(FONCT$,I,1)
960 IF F$="+" THEN TFN(I)=233
970 IF F$="-" THEN TFN(I)=234
980 IF F$="*" THEN TFN(I)=235
990 IF F$="/" THEN TFN(I)=236
1000 IF F$="^" THEN TFN(I)=237
1010 NEXT
1020 REM Codage des numériques
1030 REM Il faut d'abord isoler les nombres
1040 FOR I=1 TO LN
1050 IF VAL(MID$(FONCT$,I,1))<>0 THEN II=I : GOSUB 1080
1060 NEXT
1070 GOTO 1150
1080 FOR J=II TO LN
1090 IF VAL(MID$(FONCT$,J,1))=0 OR J=LN THEN GOSUB 1120
1100 NEXT
1110 RETURN
1120 F$=MID$(FONCT$,II,(J-II)) : F=VAL(F$)
1130 TFN(II)=F
1140 RETURN
1150 I=I-1
1160 IF VAL(MID$(FONCT$,I,1))<>0 THEN TFN(I)=VAL(MID$(FONCT$,I,1))
1170 REM Compactage du tableau TFN
1180 FOR I=LN TO 1 STEP -1
1190 IF TFN(I)= ( TFN(I-1) MOD (10^(LEN(STR$(TFN(I-1)))-2))) THEN TFN(I)=0
1200 NEXT
1210 REM Recherche de la position des numériques
1220 FOR I=1 TO LN
1230 MARK(I)=0
1240 IF VAL(MID$(FONCT$,I,1))<>0 THEN MARK(I)=1
1250 NEXT
1260 FOR I=LN TO 1 STEP -1
```

```
1280 NEXT
1290 REM Codage des numériques
1300 FOR I=1 TO LN
1310 IF MARK(I)=1 THEN GOSUB 1340
1320 NEXT
1330 GOTO 1390
1340 IF (TFN(I)>0 AND TFN(I)<=9) THEN TFN(I)=17+TFN(I) : GOTO 1380
1350 IF TFN(I)>32767 THEN TFN(I)=32767
1360 IF (TFN(I)>9 AND TFN(I)<256) THEN TFN(I+1)=TFN(I) : TFN(I)=15
1370 IF (TFN(I)>=256) THEN TFN(I+1)=(TFN(I) MOD 256) : TFN(I+2)=(TFN(I) \ 256) :
 TFN(I)=28
1380 RETURN
1390 REM
1400 REM Supression des 0
1410 CT=0
1420 FOR I=1 TO LN
1430 IF TFN(I)=0 THEN CT=CT+1
1440 NEXT
1450 DIM TFX(LN-CT)
1460 J=1
1470 FOR I=1 TO LN
1480 IF TFN(I)<>0 THEN TFX(J)=TFN(I) : J=J+1
1490 NEXT
1500 REM Pokage de la fonction
1510 FOR I=1 TO LN-CT
1520 POKE DEB+5+I,TFX(I)
1530 NEXT
1540 DEB=DEB+6+LN-CT
1550 POKE DEB,32 : POKE DEB+1,58 : POKE DEB+2,143 : POKE DEB+3,217 : POKE DEB+4,
39
1560 FOR I=1 TO 25 : POKE DEB+4+I,39 : NEXT
1570 PRINT
1580 FIN=DEB+3+I
1590 GOTO 1600
1600 DEF FNY(X)=((H(1)/H(2)-(H(1)/H(2)*EXP((-1+H(3))*H(2)*X)))^((1/(1-H(3))))+H(
4)*X) ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''''''''''''''''
1610 GOTO 3980
1620 REM La fonction ci-dessous est utilisée en paramétrique
1630 DEF FNZ(X)=COS(X) '''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
1640 IF CHG=1 THEN RETURN
1650 CLS
1660 LOCATE 5,5 : PRINT "Désirez-vous sauver votre fonction ?"
1670 X$="" : WHILE X$="" : X$=INKEY$ : WEND
1680 IF X$<>"N" AND X$<>"O" THEN 1660
1690 IF X$="N" THEN RETURN
1700 LOCATE 7,2 : PRINT "Quel nom donnez-vous à la fonction ?"
1710 LOCATE 8,25 : PRINT SPACE$(30)
1720 LOCATE 8,5 : INPUT " (6 caractères maxi) ",NOM$
1730 IF NOM$="" THEN 1650
1740 NOM$="F-"+NOM$
1750 IF LEN(NOM$)> 8-((LEFT$(NOM$,2)="B:")*2)  THEN PRINT "Le nom est trop long
..." : GOTO 1700
1760 BSAVE NOM$,SV,(FIN-SV)
1770 RETURN
1780 REM
1790 REM Menu
1800 KEY OFF
1810 CLS
1820 ST1$=CHR$(201)+STRING$(76,205)+CHR$,(187)
1830 ST2$=CHR$(200)+STRING$(76,205)+CHR$(188)
1840 LL$=CHR$(186)+STRING$(76,32)+CHR$(186)
1850 LOCATE 2,2 : PRINT ST1$
```

```
1870 LOCATE I+1,2 : PRINT LL$
1880 NEXT
1890 LOCATE 22,2 : PRINT ST2$;
1900 REM
1910 REM IMPORTANT = Quel que soit le nom des variables apparentes
1920 REM   pour l'utilisateur, toutes les fonctions seront traitées
1930 REM   en tant que FNY(X) et FNZ(X)
1940 REM
1950 LOCATE 4,34 : COLOR 1 : PRINT "GRAPHIX :" : COLOR 4
1960 LOCATE 6,21 : PRINT "Stage IUT 1986 ---- Jean-Baptiste PUEL"
1970 LOCATE 9,21 : PRINT "Cartésiennes Implicites        --> 1"
1980 LOCATE 10,21 : PRINT "Cartésiennes Paramétriques     --> 2"
1990 LOCATE 11,21 : PRINT "Polaires Implicites            --> 3"
2000 LOCATE 12,21 : PRINT "Polaires Paramétriques         --> 4"
2010 LOCATE 14,21 : PRINT "Pointé d'un fichier            --> 5"
2020 LOCATE 15,21 : PRINT "Réalisation d'un ajustement    --> 6"
2030 LOCATE 17,21 : PRINT "Opérations disque              --> 7"
2040 LOCATE 18,21 : PRINT "Fin du travail                 --> 8"
2050 LOCATE 20,40 : PRINT "Votre choix :"
2060 BEEP
2070 X$="" : WHILE X$="" : X$=INKEY$ : WEND
2080 LOCATE 20,65 : PRINT X$
2090 IF VAL(X$)=0 THEN 2070
2100 TYPFN=VAL(X$)
2110 ON VAL(X$) GOTO 2150,2240,2390,2490,2720,2650,3090,6970
2120 REM
2130 REM Cartésiennes Implicites
2140 REM
2150 CLS : LOCATE 7,5 : COLOR 1 : PRINT "Tracé de fonctions en Cartésiennes Impl
icites." : COLOR 4
2160 LOCATE 9,3 : PRINT "Ces fonctions sont de la forme :"
2170 LOCATE 11,9 : PRINT " Y=f(X)
2180 LOCATE 23,60 : PRINT"==>" : X$="" : WHILE X$="" : X$=INKEY$ : WEND
2190 VAR$="Y" : MODELE$=" Y = f(X) = " : CLS : GOSUB 160 : GOSUB 250
2200 GOTO 3950 : REM Tracé
2210 REM
2220 REM Cartésiennes Paramétriques
2230 REM
2240 CLS : LOCATE 7,5 : COLOR 1 : PRINT "Tracé de fonctions en Cartésiennes Para
métriques." : COLOR 4
2250 LOCATE 9,3 : PRINT "Ces fonctions sont de la forme :"
2260 LOCATE 11,9 : PRINT " X=f(e) et Y=f(e)"
2270 LOCATE 15,5 : PRINT "Il faut donc entrer les deux fonctions séparément."
2280 LOCATE 23,60 : PRINT"==>" : X$="" : WHILE X$="" : X$=INKEY$ : WEND
2290 REM
2300 REM Saisie des deux fonctions séparément.
2310 REM
2320 VAR$="Y" : MODELE$=" X1 = f(X) = " : CLS : LOCATE 2,3 : PRINT " Première fo
nction :" : GOSUB 160 : GOSUB 250
2330 ERASE TFN,TFX,MARK
2340 VAR$="Z" : MODELE$=" Y1 = f(X) = " : CLS : LOCATE 2,3 : PRINT  " Seconde fo
nction :" : GOSUB 160 : GOSUB 250
2350 GOTO 3950 : REM Tracé
2360 REM
2370 REM Polaires implicites.
2380 REM
2390 CLS : COLOR 1 : LOCATE 7,5 : PRINT "Tracé de fonctions en Polaire Implicite
." : COLOR 4
2400 LOCATE 9,3 : PRINT "Ces fonctions sont de la forme :"
2410 LOCATE 11,9 : PRINT " R=f(t)  avec R,rayon et t,angle."
2420 LOCATE 23,60 : PRINT "==>" : X$="" : WHILE X$="" : X$=INKEY$ : WEND
2430 REM
2440 REM On boucle sur t , le pas est défini par l'utilisateur ou fixé
2450 REM
2460 VAR$="Y" : MODELE$=" R = f(X) = " : CLS : GOSUB 160 : GOSUB 250
```

```
2480 REM
2490 REM Polaires Paramétriques
2500 REM
2510 CLS : LOCATE 7,5 : COLOR 1 : PRINT "Tracé de fonctions en Polaire Paramétri
que." : COLOR 4
2520 LOCATE 9,3 : PRINT "Ces fonctions sont de la forme :"
2530 LOCATE 11,9 : PRINT " R=f(e) et t=f(e) "
2540 LOCATE 15,5 : PRINT "Il faut donc entrer les deux fonctions séparément."
2550 LOCATE 23,60 : PRINT"==>" : X$="" : WHILE X$="" : X$=INKEY$ : WEND
2560 REM
2570 REM Saisie des deux fonctions séparément
2580 REM
2590 VAR$="Y" : MODELE$=" R = f(X) = " : CLS : LOCATE 2,3 : PRINT "Première fonc
tion : " : GOSUB 160 : GOSUB 250
2600 ERASE TFN,TFX,MARK
2610 VAR$="Z" : MODELE$=" T = f(X) = " : CLS : LOCATE 2,3 : PRINT "Seconde fonct
ion : " : GOSUB 160 : GOSUB 250
2620 REM
2630 GOTO 3950
2640 REM
2650 REM
2660 REM Réalisation d'un ajustement Nelder-Mead
2670 REM
2680 RUN"NELDER"
2690 REM
2700 REM Tracé des points issus d'un ajustement Nelder-Mead
2710 REM
2720 CLS
2730 DIM X(300),Y(300) : REM Dim arbitraire, 300 maximum présumé
2740 LOCATE 5,23 : COLOR 1 : PRINT "Pointé d'un fichier" : COLOR 4
2750 LOCATE 7,2 : INPUT"Nom du fichier à charger ?  ",FCH$
2760 IF FCH$="" THEN 1810
2770 LOCATE 9,2 : PRINT "Fichier Nelder-Mead ? (O/N) "
2780 X$="" : WHILE X$="" : X$=INKEY$ : WEND
2790 LOCATE 9,31 : PRINT X$
2800 IF X$="O" THEN ND=1 : GOTO 2830
2810 IF X$="N" THEN ND=0 : GOTO 2830
2820 GOTO 2780
2830 OPEN "I",#1,FCH$
2840 IF ND=0 THEN 3000
2850 WHILE A$<>"VALEURS CALCULEES ET ECARTS :"
2860 LINE INPUT #1,A$
2870 WEND
2880 LINE INPUT #1,A$
2890 LINE INPUT #1,A$
2900 A$=""
2910 P=0
2920 WHILE NOT EOF(1)
2930 P=P+1
2940 INPUT #1,A
2950 INPUT #1,X(P),Y(P)
2960 INPUT #1,A : INPUT #1,A
2970 WEND
2980 GOTO 5220 : REM Tracé
2990 REM
3000 A$="" : P=0
3010 WHILE NOT EOF(1)
3020 P=P+1
3030 INPUT #1,X(P),Y(P)
3040 WEND
3050 GOTO 5220 : REM Tracé
3060 REM
3070 REM
3080 REM Opérations disque
3090 CLS
```

```
3110 LOCATE 10,24 : PRINT "Détruire une fonction        ---> 1"
3120 LOCATE 11,24 : PRINT "Renommer une fonction        ---> 2"
3130 LOCATE 12,24 : PRINT "Initialiser une disquette    ---> 3"
3140 LOCATE 13,24 : PRINT "Catalogue d'une disquette    ---> 4"
3150 LOCATE 15,24 : PRINT "Retour au menu               ---> 5"
3160 X$="" : WHILE X$="" : X$=INKEY$ : WEND
3170 IF VAL(X$)=0 THEN 3160
3180 ON VAL(X$) GOTO 3190,3310,3380,3520,1790
3190 REM Détruire une fonction
3200 LOCATE 20,6 : INPUT "Nom de la fonction à détruire ? ",NOM$
3210 IF NOM$="" THEN 3090
3220 LOCATE 22,3 : PRINT"Détruire ";NOM$;"...  Confirmation ? (O/N)"
3230 X$="" : WHILE X$="" : X$=INKEY$ : WEND
3240 IF X$<>"O" AND X$<>"N" THEN 3220
3250 IF X$="N" THEN 3090
3260 NOM$="F-"+NOM$+".BAS"
3270 KILL NOM$
3280 GOTO 3090
3290 REM
3300 REM Renommer une fonction
3310 LOCATE 20,6 : INPUT "Nom de la fonction à renommer ? ",NOM$
3320 IF NOM$="" THEN 3090
3330 LOCATE 21,6 : INPUT "Nouveau nom de la fonction ? ",NNOM$
3340 IF NNOM$="" THEN 3090
3350 NOM$="F-"+NOM$+".BAS" : NNOM$="F-"+NNOM$+".BAS"
3360 NAME NOM$ AS NNOM$
3370 GOTO 3090
3380 REM
3390 REM Formattage d'une disquette
3400 LOCATE 18,4 : PRINT "Quel disque voulez-vous formatter ? (A/B) "
3410 X$="" : WHILE X$="" : X$=INKEY$ : WEND
3420 IF X$<>"A" AND X$<>"B" THEN 3400
3430 D$=X$
3440 LOCATE 20,10 : PRINT "Formattage du disque ";X$
3450 LOCATE 21,1 : PRINT "Insérez une disquette vierge dans le lecteur ";X$
3460 LOCATE 22,1 : PRINT "Tapez F pour formatter, n'importe quelle touche pour a
rreter"
3470 X$="" : WHILE X$="" : X$=INKEY$ : WEND
3480 IF X$<>"F" THEN 3090
3490 IF D$="A" THEN SHELL"FORMAT A:" : GOTO 3090
3500 IF D$="B" THEN SHELL"FORMAT B:" : GOTO 3090
3510 REM
3520 REM Catalogue
3530 LOCATE 20,5 : PRINT "De quel disque voulez-vous le catalogue ? (A/B/C) "
3540 X$="" : WHILE X$="" : X$=INKEY$ : WEND
3550 CLS
3560 ON (ASC(X$)-64) GOSUB 3580,3590,3600
3570 GOTO 3610
3580 FILES"A:" : RETURN
3590 FILES"B:" : RETURN
3600 FILES"C:" : RETURN
3610 LOCATE 22,60 : PRINT"--->" : X$="" : WHILE X$="" : X$=INKEY$ : WEND
3620 GOTO 3090
3630 REM
3640 REM
3650 REM Traitement des erreurs
3660 REM
3670 IF (ERR=18) THEN RESUME : GOTO 1600
3680 IF (ERL=2330 OR ERL=3950 OR ERL=5160) THEN RESUME NEXT : REM Evite de plant
er en détruisant les 3  tableaux pas encore crées.
3690 IF (ERR=11) OR (ERR=6) THEN X=X+PAS : RESUME
3700 CLS
3710 IF (ERR=61) THEN MES$= " Le disque est plein ... " : GOTO 3810
3720 IF (ERR=67) THEN MES$= " Il y a trop de fichiers ... "  : GOTO 3810
3730 IF (ERR=53) THEN MES$= " Je ne trouve pas ce fichier ... "  : GOTO 3810
```

```
3750 IF (ERR=71) THEN MES$= " Le disque n'est pas pret ... " : GOTO 3810
3760 IF (ERR=72) THEN MES$= " Erreur d'E/S ... " : GOTO 3810
3770 IF (ERR=75) THEN MES$= " Erreur de chemin d'acces ... " : GOTO 3810
3780 IF (ERR=76) THEN MES$= " Chemin non trouvé ... " : GOTO 3810
3790 IF (ERR=200) THEN MES$= " Erreur traceur N° : "+A$+" en ligne : "+STR$(ERL)
3800 IF (ERR=200) AND (A$="3") THEN X=X+PAS : RESUME
3810 LOCATE 12,25 : PRINT MES$
3820 LOCATE 23,60 : PRINT "-->" : X$="" : WHILE X$="" : X$=INKEY$ : WEND
3830 IF (ERL=340) THEN RESUME 250
3840 IF (ERL=1760) THEN RESUME 1650
3850 IF (ERL=3270) OR (ERL=3360) OR (ERL=3490) OR (ERL=3500) THEN RESUME 3090
3860 IF (ERL=3580) OR (ERL=3590) OR (ERL=3600) THEN RESUME 3090
3870 IF (ERR=200) THEN RESUME
3880 IF (ERL=2830) THEN RESUME 1810
3890 END
3900 REM
3910 REM
3920 REM
3930 REM     ---- Enfin le tracé ... ----
3940 REM
3950 ERASE TFN,TFX,MARK
3960 REM
3970 REM
3980 CLS
3990 LOCATE 1,23 : PRINT " Tracé de courbes sur Plotter HP "
4000 LOCATE 3,2 : PRINT " Couleur de la plume ?     " : LOCATE 3,26 : COLOR 11 :
PRINT "2" : COLOR 4
4010 X$="" : WHILE X$="" : X$=INKEY$ : WEND
4020 LOCATE 3,26 : COLOR 11 : PRINT X$ : COLOR 4
4030 PN=VAL(X$)-(2*(X$=CHR$(13))) : REM Si X$=return, PN=2
4040 LOCATE 5,2 : PRINT " Agrandissement sur X ? "
4050 LOCATE 5,27 : COLOR 11 : PRINT "1" : COLOR 4
4060 LOCATE 5,27 : COLOR 11 : INPUT "",AGX$ : COLOR 4
4070 LOCATE 5,42 : PRINT " Agrandissement sur Y ? "
4080 LOCATE 5,67 : COLOR 11 : PRINT "1" : COLOR 4
4090 LOCATE 5,67 : COLOR 11 : INPUT "",AGY$ : COLOR 4
4100 AGX=VAL(AGX$) : AGY=VAL(AGY$)
4110 IF AGX$="" THEN AGX=1 : IF AGY$="" THEN AGY=1
4120 IF TYPFN<>5 THEN 4180
4130 REM
4140 REM Si Nelder-Mead, le paramètre est l'indice du tableau
4150 P1$="1" : P2$=STR$(P) : PAS=1
4160 GOTO 4290
4170 REM
4180 LOCATE 7,2 : PRINT " Valeur inférieure du paramètre ?          "
4190 LOCATE 7,36 : COLOR 11 : INPUT" ",P1$ : COLOR 4
4200 IF P1$="" THEN 4180
4210 LOCATE 8,2 : PRINT " Valeur supérieure du paramètre ?          "
4220 LOCATE 8,36 : COLOR 11 : INPUT" ",P2$ : COLOR 4
4230 IF P2$="" THEN 4210
4240 IF VAL(P2$)<=VAL(P1$) THEN 4180
4250 IF TYPFN<> 5 THEN PAS=((VAL(P2$)-VAL(P1$))/100)
4260 REM
4270 INST$="PD" : SM$=""
4280 REM Le tracé se fait plume basse, SM$ est le marqueur de point.
4290 IF TYPFN=5 THEN INST$="PU" : SM$="x"
4300 REM Ici, tracé plume haute, le marqueur est "x" (modifiable)
4310 REM
4320 REM Calcul des bornes XMI, XMA, YMI et YMA
4330 REM
4340 GOSUB 6660
4350 REM
4360 REM
4370 IF TYPFN=5 THEN PAS=1 : GOTO 4420
4380 LOCATE 15,2 : PRINT " Pas du tracé ?" : LOCATE 15,19 : COLOR 11 : PRINT "0
```

```
4390 LOCATE 15,18 : COLOR 11 : INPUT " ",PAS$ : COLOR 4
4400 PAS=VAL(PAS$)
4410 IF PAS$="" THEN PAS=.01
4420 LOCATE 15,40 : PRINT " Tracé des axes ? (O/N)        " : AX$="" : WHILE AX$=
"" : AX$=INKEY$ : WEND
4430 LOCATE 15,68 :COLOR 11 : PRINT AX$ : COLOR 4 : IF AX$<>"O" AND AX$<>"N" THE
N 4420
4440 LOCATE 17,2 : PRINT " Titre du tracé ? "
4450 LOCATE 17,21 : COLOR 11 : INPUT "",TITRE$ : COLOR 4
4460 LOCATE 19,2 : PRINT " Légende, axe des X ? "
4470 LOCATE 19,25 : COLOR 11 : INPUT "",LEGX$ : COLOR 4
4480 LOCATE 20,2 : PRINT " Légende, axe des Y ? "
4490 LOCATE 20,25 : COLOR 11 : INPUT "",LEGY$ : COLOR 4
4500 IF AX$="N" THEN 4550
4510 LOCATE 22,2 : PRINT " Unité de graduation - axe des X ? "
4520 LOCATE 22,38 : COLOR 11 : INPUT "",UX : COLOR 4
4530 LOCATE 23,2 : PRINT " Unité de graduation - axe des Y ? "
4540 LOCATE 23,38 : COLOR 11 : INPUT "",UY : COLOR 4
4550 LOCATE 24,2 : PRINT " OK ? (O/N) "
4560 X$="" : WHILE X$="" : X$=INKEY$ : WEND
4570 LOCATE 23,15 : COLOR 11 : PRINT X$ : COLOR 4
4580 IF X$="O" THEN 4610
4590 IF X$="N" THEN 3980
4600 GOTO 4560
4610 REM
4620 BORNE$=XMN$+","+XMM$+","+YMN$+","+YMM$+";"
4630 REM
4640 ET$=CHR$(3)
4650 OPEN "COM1:4800,N,8,1,CS0,DS0,CD0"AS #3
4660 PRINT #3,"IN;IP;";ET$;
4670 GOSUB 5340 : REM Tempo
4680 REM
4690 REM Ecriture des Labels
4700 REM
4710 IF LEGX$="" AND LEGY$="" AND TITRE$="" THEN 4750
4720 GOSUB 5800
4730 GOSUB 5340
4740 REM
4750 GOSUB 5340
4760 GOSUB 5960 : REM Fixe les limites matérielles
4770 REM
4780 REM Définition du cadre
4790 PRINT #3,"SC";BORNE$;ET$;
4800 REM
4810 REM Tracé des axes
4820 REM
4830 IF AX$="O" THEN GOSUB 6000
4840 GOSUB 5340
4850 REM
4860 REM Positionnement sur le premier point
4870 REM
4880 X=VAL(P1$)
4890 GOSUB 5430
4900 PRINT #3,"PU;PA"; : PRINT #3,USING"#####.####";XX; : PRINT #3,","; : PRINT
#3,USING"#####.####";YY; : PRINT #3,";";ET$;
4910 REM
4920 REM Dernieres initialisations
4930 REM
4940 PRINT #3,"SP";PN;";";ET$;
4950 GOSUB 5340
4960 REM
4970 REM Boucle principale
4980 REM
4990 PRINT #3,INST$;ET$;
5000 FOR X=VAL(P1$) TO VAL(P2$) STEP PAS
```

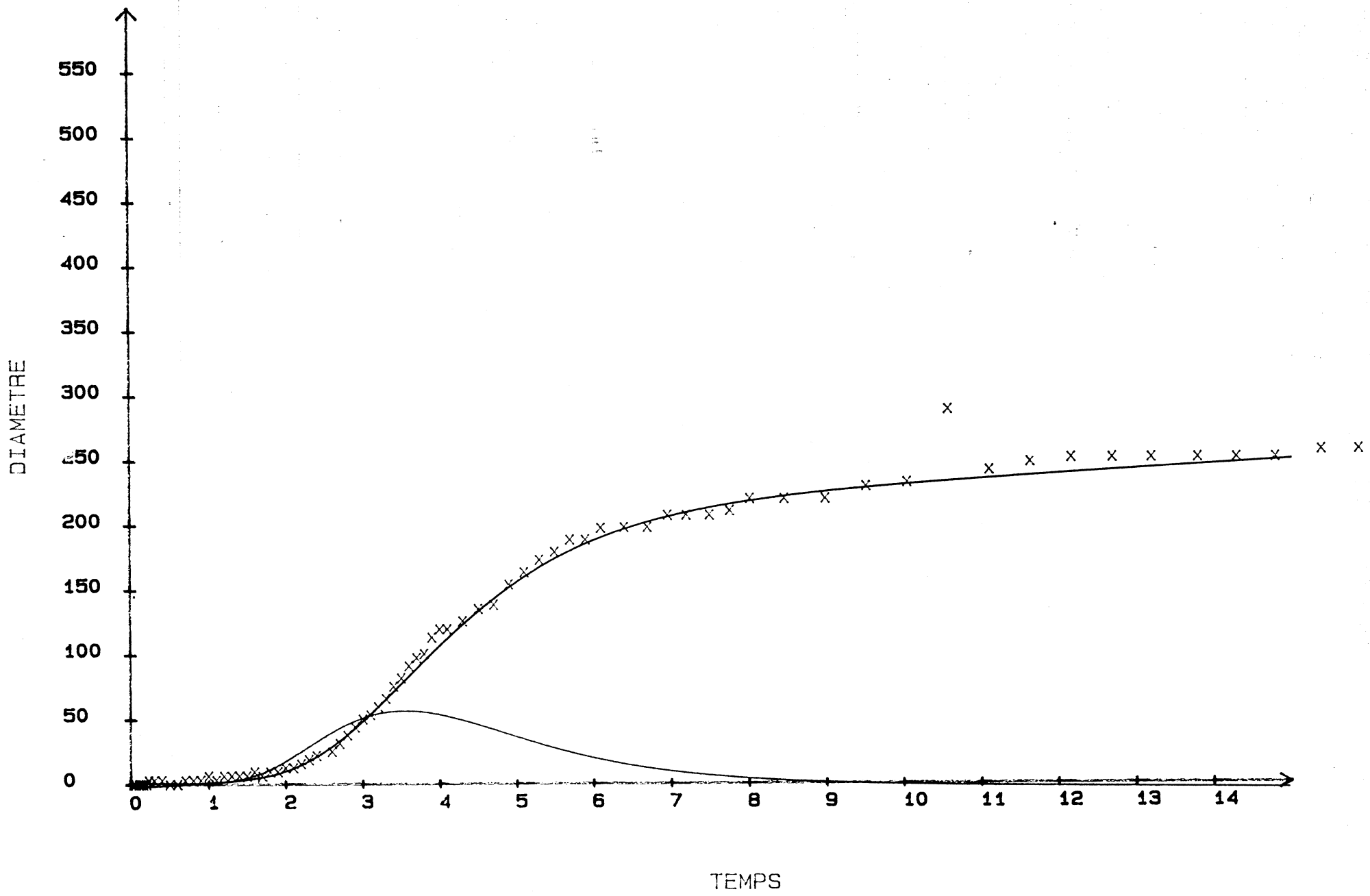
```
5020 PRINT #3,CHR$(27);".B";ET$;
5030 A$="" : B$=""
5040 WHILE B$<>CHR$(13) : B$=INPUT$(1,#3) : A$=A$+B$ : WEND
5050 IF VAL(A$)<512 THEN GOSUB 5350
5060 REM Test erreurs traceur
5070 PRINT #3,"OE;";ET$
5080 A$="" : B$=""
5090 WHILE B$<>CHR$(13) : B$=INPUT$(1,#3) : A$=A$+B$ : WEND
5100 IF VAL(A$)<>0 THEN ERROR 200
5110 GOSUB 5400
5120 PRINT #3,"PA"; : PRINT #3,USING"#####.####";XX; : PRINT #3,","; : PRINT #3,
USING"#####.####";YY; : PRINT #3,";SM";SM$;";";ET$;
5130 NEXT X
5140 PRINT #3,"SPO;";ET$;
5150 CLOSE
5160 ERASE TFN,TFX,MARK,X,Y
5170 GOTO 1810 : REM Retour au menu
5180 REM
5190 REM
5200 REM Pointage Nelder-Mead
5210 REM
5220 REM Tri du tableau
5230 REM
5240 XMI$=STR$(X(1)) : XMA$=XMI$ : YMI$=STR$(Y(1)) : YMA$=YMI$
5250 FOR I=1 TO P
5260 IF X(I)<VAL(XMI$) THEN XMI$=STR$(X(I))
5270 IF X(I)>VAL(XMA$) THEN XMA$=STR$(X(I))
5280 IF Y(I)<VAL(YMI$) THEN YMI$=STR$(Y(I))
5290 IF Y(I)>VAL(YMA$) THEN YMA$=STR$(Y(I))
5300 NEXT
5310 GOTO 3930 : REM Les tableaux sont initialisés, début du tracé
5320 REM
5330 REM Boucle de tempo pour le tracé
5340 REM
5350 A$="" : B$=""
5360 PRINT #3,CHR$(27);".B";ET$;
5370 WHILE B$<>CHR$(13) : B$=INPUT$(1,#3) : A$=A$+B$ : WEND
5380 IF VAL(A$)>512 THEN RETURN ELSE GOTO 5350
5390 RETURN
5400 REM
5410 REM Calcul des valeurs à tracer
5420 REM
5430 ON TYPFN GOSUB 5460,5520,5580,5650,5750
5440 XX=XX*AGX : YY=YY*AGY
5450 RETURN
5460 REM
5470 REM Cartésiennes implicites
5480 REM
5490 XX=X
5500 YY=FNY(X)
5510 RETURN
5520 REM
5530 REM Cartésiennes paramétriques
5540 REM
5550 XX=FNY(X)
5560 YY=FNZ(X)
5570 RETURN
5580 REM
5590 REM Polaires implicites
5600 REM
5610 R=FNY(X) : REM La variable X correspond à T (angle)
5620 XX=R*COS(X)
5630 YY=R*SIN(X)
5640 RETURN
5650 REM
```

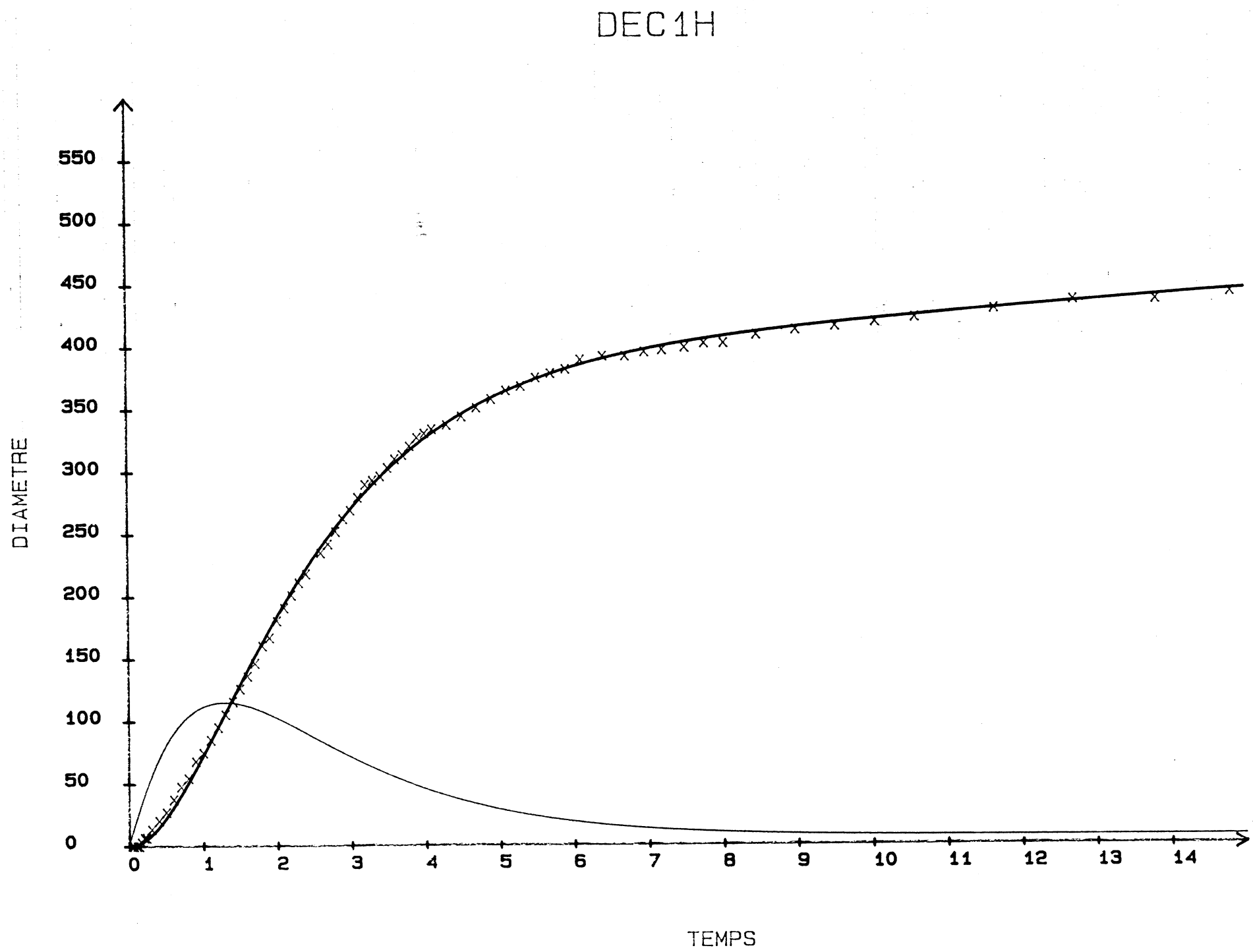
```
5670 REM
5680 R=FNY(X) : REM Le rayon et l'angle sont fonction de X
5690 T=FNZ(X)
5700 XX=R*COS(T)
5710 YY=R*SIN(T)
5720 RETURN
5730 REM
5740 REM Récupération des points Nelder-Mead
5750 REM
5760 XX=X(X)
5770 YY=Y(X)
5780 RETURN
5790 REM
5800 REM
5810 REM Ecriture des labels
5820 REM
5830 PRINT #3,"SP";PN;";";ET$;
5840 IF LEGX$="" THEN 5870
5850 NI=INT(47-(LEN(LEGX$)/2))
5860 PRINT #3,"PA552,386;CP";NI;",-.95;LB";LEGX$;ET$;
5870 IF LEGY$="" THEN 5900
5880 NI=INT(31-(LEN(LEGY$)/2))
5890 PRINT #3,"PA400,386;DI0,1;CP";NI;",.95;LB";LEGY$;ET$;
5900 IF TITRE$="" THEN 5960
5910 NI=INT(25-(LEN(TITRE$)/2))
5920 PRINT #3,"PA552,7300;DI1,0;SR1.3,2.6;CP";NI;",.5;LB";TITRE$;ET$;
5930 PRINT #3,"SR.75,1.5;";ET$;
5940 RETURN
5950 REM
5960 PRINT #3,"IP800,600,10000,7000;";ET$;
5970 REM
5980 RETURN
5990 REM
6000 REM
6010 REM Routine de tracé des axes
6020 REM
6030 L=VAL(XMA$)-VAL(XMI$)
6040 H=VAL(YMA$)-VAL(YMI$)
6050 PRINT #3,"SP1;PU0,0;";ET$;
6060 X=0
6070 PRINT #3,"PD;";ET$;
6080 WHILE X<VAL(XMA$)
6090 PRINT #3,"PA";X;",0;XT;";ET$;
6100 PRINT #3,"PU;CP-1,-1;";ET$;
6110 PRINT #3,"LB";STR$(X);ET$;
6120 PRINT #3,"PU;PA";X;",0;PD;";ET$;
6130 X=X+UX
6140 WEND
6150 GOSUB 5340
6160 PRINT #3,"PD";XMA$;",0;";ET$;
6170 PRINT #3,"PU;PR"; : PRINT #3,USING"###.##";(L/-100); : PRINT #3,","; : PRIN
T #3,USING"###.##";(H/-120); : PRINT #3,"PD;PA";XMA$;",0;";
6180 PRINT #3,"PD";XMA$;",0;";ET$;
6190 PRINT #3,"PU;PR"; : PRINT #3,USING"###.##";(L/-100); : PRINT #3,","; : PRIN
T #3,USING"###.##";(H/-120); : PRINT #3,"PD;PA";XMA$;",0;";
6200 PRINT #3,"PU;PR"; : PRINT #3,USING"###.##";(L/-100); : PRINT #3,","; : PRIN
T #3,USING"###.##";(H/120); : PRINT #3,"PD;PA";XMA$;",0;";ET$;
6210 PRINT #3,"PU0,0";ET$;
6220 GOSUB 5340
6230 X=0
6240 PRINT #3,"PD";ET$;
6250 WHILE X>VAL(XMI$)
6260 PRINT #3,"PA";X;",0;XT;";ET$;
6270 IF X=0 THEN 6310
6280 PRINT #3,"PU;CP-1,-1;";ET$;
```

```
6300 PRINT #3,"PU;PA";X;",0;PD;";ET$;
6310 X=X-UX
6320 WEND
6330 GOSUB 5340
6340 PRINT #3,"PD";XMI$;",0;";ET$;
6350 PRINT #3,"PU0,0;";ET$;
6360 Y=0
6370 PRINT #3,"PD;";ET$;
6380 WHILE Y<VAL(YMA$)
6390 PRINT #3,"PA0,";Y;";YT;";ET$;
6400 PRINT #3,"PU;CP-6,0;";ET$;
6410 PRINT #3,"LB";STR$(Y);ET$;
6420 PRINT #3,"PU;PA0,";Y;";PD;";ET$;
6430 Y=Y+UY
6440 WEND
6450 GOSUB 5340
6460 PRINT #3,"PD0,";YMA$;";";ET$;
6470 PRINT #3,"PU;PR"; : PRINT #3,USING"###.##";(L/-130); : PRINT #3,","; : PRIN
T #3,USING"###.##";(H/-75); : PRINT #3,"PD;PA0,";YMA$;";";
6480 PRINT #3,"PU;PR"; : PRINT #3,USING"###.##";(L/130); : PRINT #3,","; : PRINT
 #3,USING"###.##";(H/-75); : PRINT #3,"PD;PA0,";YMA$;ET$;
6490 PRINT #3,"PU0,0;";ET$;
6500 Y=0
6510 PRINT #3,"PD;";ET$;
6520 WHILE Y>VAL(YMI$)
6530 PRINT #3,"PA0,";Y;";YT;";ET$;
6540 IF Y=0 THEN 6580
6550 PRINT #3,"PU;CP-6,0;";ET$;
6560 PRINT #3,"LB";STR$(Y);ET$;
6570 PRINT #3,"PU;PA0,";Y;";PD;";ET$;
6580 Y=Y-UY
6590 WEND
6600 GOSUB 5340
6610 PRINT #3,"PD0,";YMI;";";ET$;
6620 RETURN
6630 REM
6640 REM Calcul de XMI$, XMA$, YMI$ et YMA$
6650 REM
6660 X=VAL(P1$) : GOSUB 5430 : XMI$=STR$(XX) : XMA$=XMI$
6670 YMI$=STR$(YY) : YMA$=YMI$
6680 FOR X=VAL(P1$) TO VAL(P2$) STEP PAS
6690 GOSUB 5430
6700 IF XX<VAL(XMI$) THEN XMI$=STR$(XX)
6710 IF XX>VAL(XMA$) THEN XMA$=STR$(XX)
6720 IF YY<VAL(YMI$) THEN YMI$=STR$(YY)
6730 IF YY>VAL(YMA$) THEN YMA$=STR$(YY)
6740 NEXT
6750 IF TYPFN=1 THEN XMI$=P1$ : XMA$=P2$
6760 LOCATE 10,2 : PRINT " Valeur inférieure - axe des X ? "; : COLOR 11 : PRINT
 XMI$ : COLOR 4
6770 LOCATE 10,35 : COLOR 11 : INPUT"",X1$ : COLOR 4
6780 IF X1$<>"" THEN XMI$=X1$
6790 LOCATE 11,2 : PRINT " Valeur supérieure - axe des X ? "; : COLOR 11 : PRINT
 XMA$ : COLOR 4
6800 LOCATE 11,35 : COLOR 11 : INPUT"",X2$ : COLOR 4
6810 IF X2$<>"" THEN XMA$=X2$
6820 LOCATE 12,2 : PRINT " Valeur inférieure - axe des Y ? "; : COLOR 11 : PRINT
 YMI$ : COLOR 4
6830 LOCATE 12,35 : COLOR 11 : INPUT"",Y1$ : COLOR 4
6840 IF Y1$<>"" THEN YMI$=Y1$
6850 LOCATE 13,2 : PRINT " Valeur supérieure - axe des Y ? "; : COLOR 11 : PRINT
 YMA$ : COLOR 4
6860 LOCATE 13,35 : COLOR 11 : INPUT"",Y2$ : COLOR 4
6870 IF Y2$<>"" THEN YMA$=Y2$
6880 LX=(VAL(XMA$)-VAL(XMI$))/30 : LY=(VAL(YMA$)-VAL(YMI$))/30
6890 XMN$=STR$((INT(VAL(XMI$)*100))/100)
```

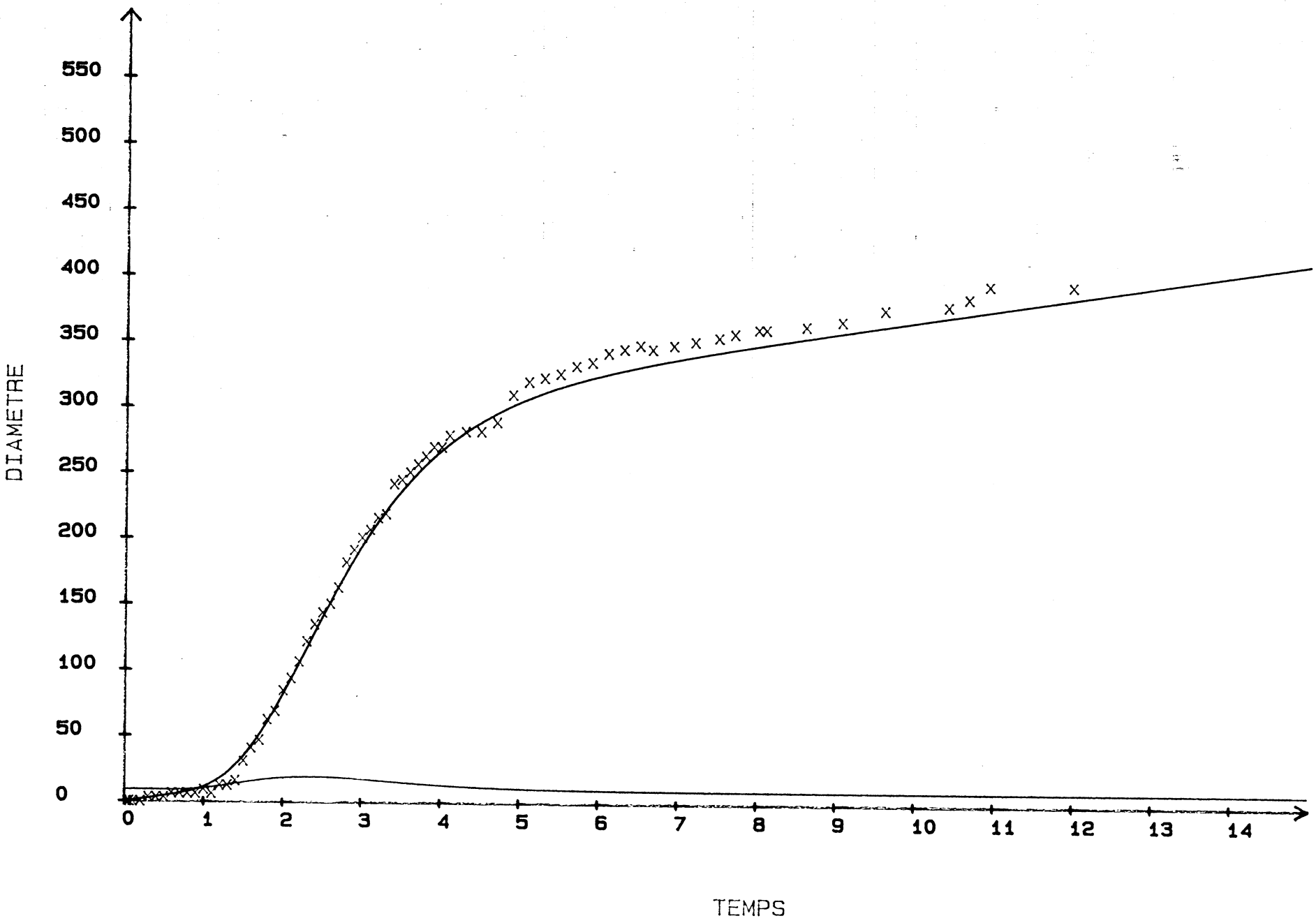
```
6900 XMM$=STR$((INT(VAL(XMA$)*100))/100)
6910 YMN$=STR$((INT(VAL(YMI$)*100))/100)
6920 YMM$=STR$((INT(VAL(YMA$)*100))/100)
6930 REM
6940 RETURN
6950 REM
6960 REM Sortie
6970 REM
6980 CLS
6990 LOCATE 10,35 : COLOR 1 : PRINT "Au revoir" : COLOR 4
7000 LOCATE 22,2 : PRINT "Retour au système ?  (O/N) "
7010 X$="" : WHILE X$="" : X$=INKEY$ : WEND
7020 LOCATE 22,31 : PRINT X$
7030 IF X$="N" THEN CLS : END
7040 IF X$="O" THEN CLS : SYSTEM
7050 GOTO 7010
```

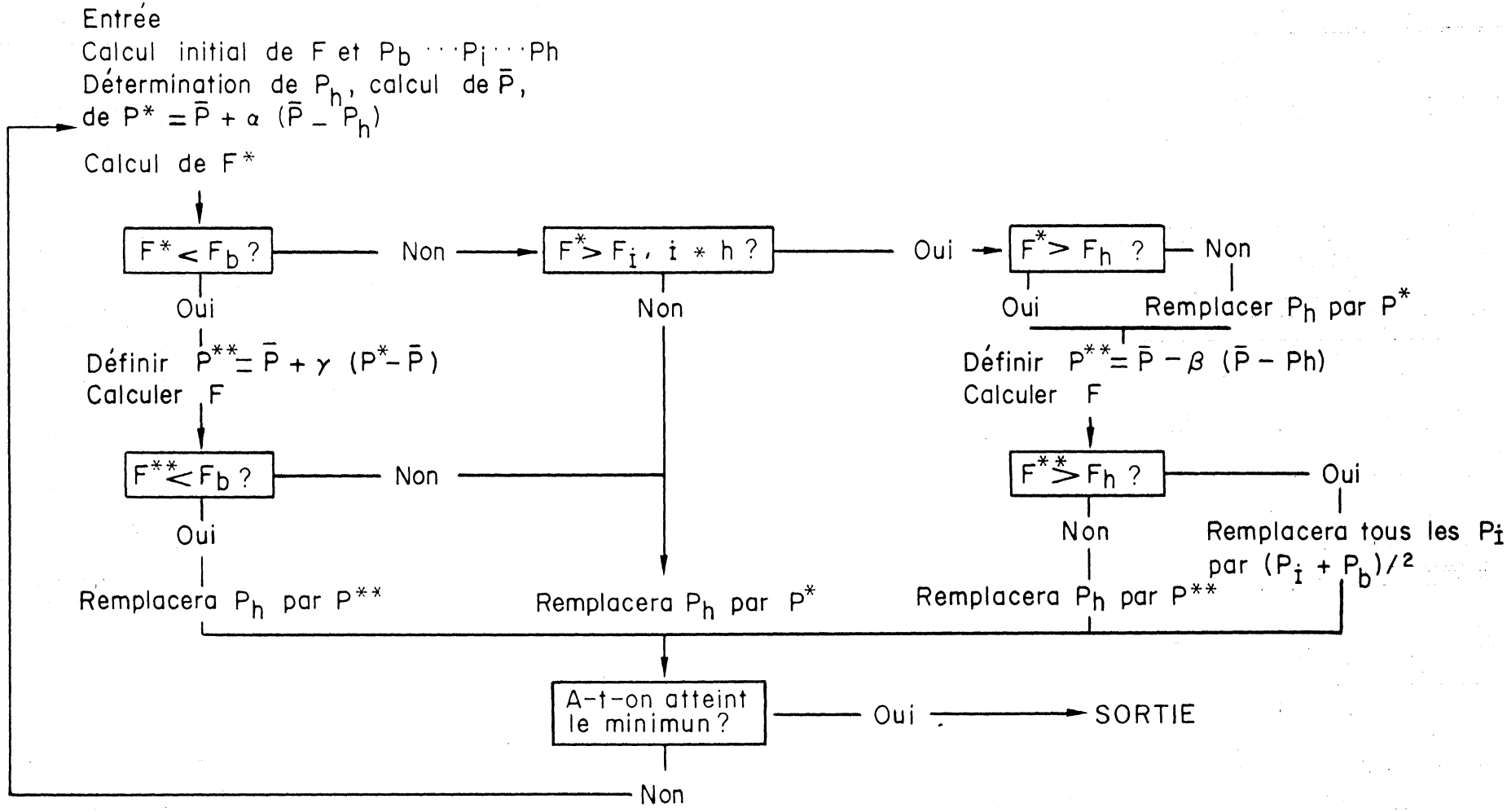
DEC1B


DIAMETRE
550
500
450
400
350
300
250
200
150
100
50
0
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
TEMPS


DEC1H
DIAMETRE
550
500
450
400
350
300
250
200
150
100
50
0
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
TEMPS

DEC2B


DIAMETRE
550
500
450
400
350
300
250
200
150
100
50
0
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
TEMPS



Organigramme du programme Nelder-Mead (1965)

The Computer Journal, 7, 308 ... 1965

Labo Zoocecologie du Sol - CEPE Montpellier -

A simplex method for function minimization

By J. A. Nelder and R. Mead†

A method is described for the minimization of a function of n variables, which depends on the comparison of function values at the $(n + 1)$ vertices of a general simplex, followed by the replacement of the vertex with the highest value by another point. The simplex adapts itself to the local landscape, and contracts on to the final minimum. The method is shown to be effective and computationally compact. A procedure is given for the estimation of the Hessian matrix in the neighbourhood of the minimum, needed in statistical estimation problems.

Spendley *et al.* (1962) introduced an ingenious idea for tracking optimum operating conditions by evaluating the output from a system at a set of points forming a simplex in the factor-space, and continually forming new simplices by reflecting one point in the hyperplane of the remaining points. This idea is clearly applicable to the problem of minimizing a mathematical function of several variables, as was recognized by these authors. However, they assumed that the relative steps to be made in varying the factors were known, and this makes their strategy rather rigid for general use. In the method to be described the simplex adapts itself to the local landscape, elongating down long inclined planes, changing direction on encountering a valley at an angle, and contracting in the neighbourhood of a minimum. The criterion for stopping the process has been chosen with an eye to its use for statistical problems involving the maximization of a likelihood function, in which the unknown parameters enter non-linearly.

The method

We consider, initially, the minimization of a function of n variables, without constraints. $P_0, P_1, \ldots P_n$ are the $(n + 1)$ points in n-dimensional space defining the current "simplex." [The simplex will not, of course, be regular in general.] We write y_i for the function value at P_i, and define

h as the suffix such that $y_h = \max_i (y_i)$ [h for "high"]

and

l as the suffix such that $y_l = \min_i (y_i)$ [l for "low"].

Further we define $\bar{P}$ as the centroid of the points with $i \neq h$, and write $[P_iP_j]$ for the distance from P_i to P_j. At each stage in the process P_h is replaced by a new point; three operations are used—*reflection*, *contraction*, and *expansion*. These are defined as follows: the reflection of P_h is denoted by P^*, and its co-ordinates are defined by the relation

$$P^* = (1 + \alpha)\bar{P} - \alpha P_h$$

where α is a positive constant, the *reflection coefficient*. Thus P^* is on the line joining P_h and $\bar{P}$, on the far side of $\bar{P}$ from P_h with $[P^*\bar{P}] = \alpha[P_h\bar{P}]$. If y^* lies between y_h and y_l, then P_h is replaced by P^* and we start again with the new simplex.

If $y^* < y_l$, i.e. if reflection has produced a new minimum, then we expand P^* to P^{**} by the relation

$$P^{**} = \gamma P^* + (1 - \gamma)\bar{P}.$$

The *expansion coefficient* γ, which is greater than unity, is the ratio of the distance $[P^{**}\bar{P}]$ to $[P^*\bar{P}]$. If $y^{**} < y_l$ we replace P_h by P^{**} and restart the process; but if $y^{**} > y_l$ then we have a failed expansion, and we replace P_h by P^* before restarting.

If on reflecting P to P^* we find that $y^* > y_i$ for all $i \neq h$, i.e. that replacing P by P^* leaves y^* the maximum, then we define a new P_h to be either the old P_h or P^*, whichever has the lower y value, and form

$$P^{**} = \beta P_h + (1 - \beta)\bar{P}.$$

The *contraction coefficient* β lies between 0 and 1 and is the ratio of the distance $[P^{**}\bar{P}]$ to $[P\bar{P}]$. We then accept P^{**} for P_h and restart, unless $y^{**} > \min(y_h, y^*)$, i.e. the contracted point is worse than the better of P_h and P^*. For such a failed contraction we replace all the P_i's by $(P_i + P_l)/2$ and restart the process.

A failed expansion may be thought of as resulting from a lucky foray into a valley (P^*) but at an angle to the valley so that P^{**} is well up on the opposite slope. A failed contraction is much rarer, but can occur when a valley is curved and one point of the simplex is much farther from the valley bottom than the others; contraction may then cause the reflected point to move away from the valley bottom instead of towards it. Further contractions are then useless. The action proposed contracts the simplex towards the lowest point, and will eventually bring all points into the valley. The coefficients α, β, γ give the factor by which the volume of the simplex is changed by the operations of reflection, contraction or expansion respectively. The complete method is given as a flow diagram in Fig. 1.

A final point concerns the criterion used for halting the procedure. The criterion adopted is somewhat different from that used by Powell (1964) in that it is concerned with the variation in the y values over the

† *National Vegetable Research Station, Wellesbourne, Warwick.*

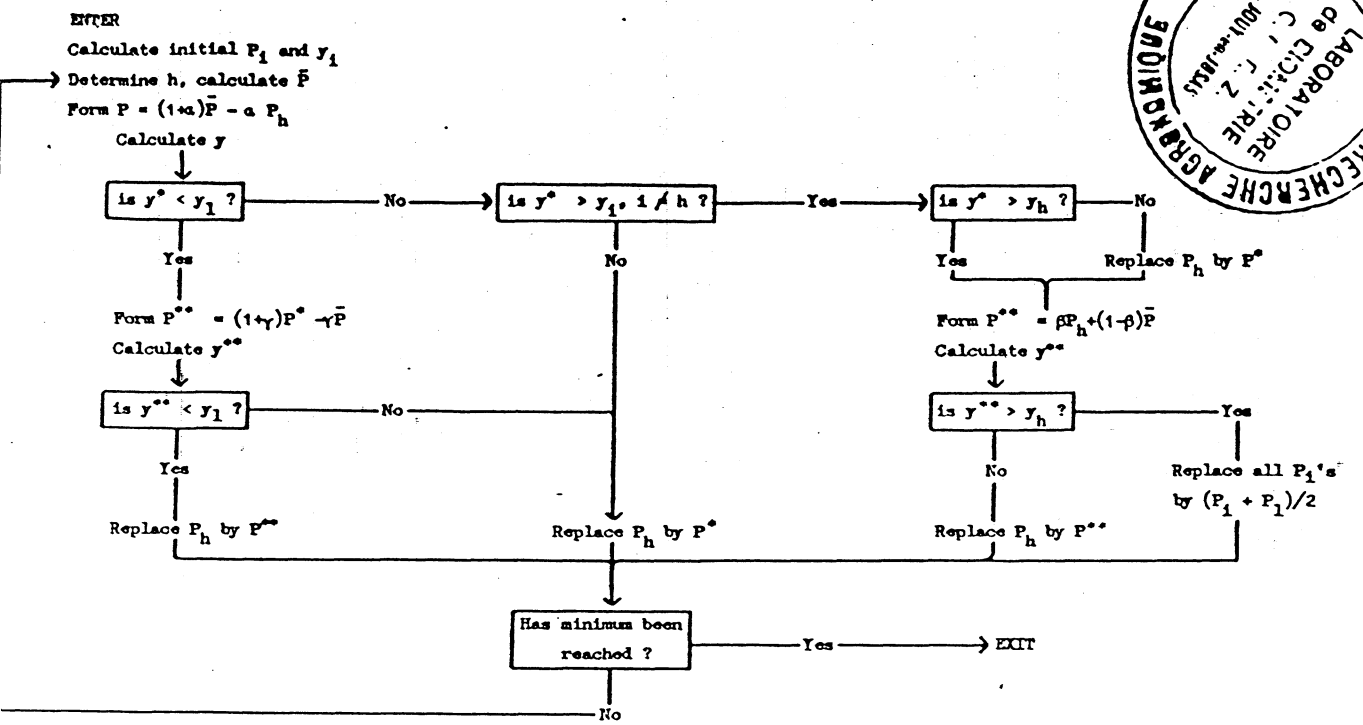


Fig. 1.—Flow diagram

mplex rather than with changes in the x's. The form hosen is to compare the "standard error" of the y's ı the form $\sqrt{\{\Sigma(y_i - \bar{y})^2/n\}}$ with a pre-set value, and stop when it falls below this value. The success of ıe criterion depends on the simplex not becoming too nall in relation to the curvature of the surface until the nal minimum is reached. The reasoning behind the iterion is that in statistical problems where one is oncerned with finding the minimum of a negative kelihood surface (or of a sum-of-squares surface) the urvature near the minimum gives the information vailable on the unknown parameters. If the curvature slight the sampling variance of the estimates will be rge so there is no sense in finding the co-ordinates of ıe minimum very accurately, while if the curvature is ıarked there is justification for pinning down the inimum more exactly.

onstraints on the volume to be searched

If, for example, one of the x_i must be non-negative in minimization problem, then our method may be lapted in one of two ways. The scale of the x conrned can be transformed, e.g., by using the logarithm, that negative values are excluded, or the function can modified to take a large positive value for all negative In the latter case any trespassing by the simplex over e border will be followed automatically by contraction oves which will eventually keep it inside. In either se an actual minimum with $x = 0$ would be inaccessle in general, though arbitrarily close approaches could be made to it. Clearly either technique can deal with individual limitations on the range of any number of x's. Constraints involving more than one x can be included using the second technique provided that an initial simplex can be found inside the permitted region, from which to start the process. Linear constraints that reduce the dimensionality of the field of search can be included by choosing the initial simplex to satisfy the constraints and reducing the dimensions accordingly. Thus to minimize $y = f(x_1, x_2, x_3)$ subject to $x_1 + x_2 + x_3 = X$, we could choose an initial simplex with vertices $(X, 0, 0)$, $(0, X, 0)$, and $(0, 0, X)$, treating the search as being in two dimensions. In particular, any x_i may be held constant by setting its value to that constant for all vertices of the initial simplex.

Results

Three functions, all of which have been used before for testing minimization procedures, were used to test the method. The functions, all of which have a minimum of zero, were:

(1) Rosenbrock's parabolic valley (Rosenbrock (1960))
$y = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$, starting point $(-1{\cdot}2, 1)$.

(2) Powell's quartic function (Powell (1962))
$y = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4$,
starting point $(3, -1, 0, 1)$.

$F_b = y_l$ $F_l = y_l$ P_r

(3) Fletcher and Powell's helical valley (Fletcher and Powell (1963))

$$y = 100[x_3 - 10\theta(x_1, x_2)]^2 + [\sqrt{(x_1^2 + x_2^2)} - 1]^2 + x_3^2$$

where $2\pi\theta(x_1, x_2) = \arctan(x_2/x_1),\ x_1 > 0$
$= \pi + \arctan(x_2/x_1),\ x_1 < 0$

starting point $(-1, 0, 0)$.

The stopping criterion used was $\sqrt{\{\Sigma(y_i - \bar{y})^2/n\}} < 10^{-8}$. The function value at the centroid of the final simplex usually deviated from the true minimum by less than 10^{-8}; a sample of runs gave $2{\cdot}5 \times 10^{-9}$ as the geometric mean of this deviation. A difficulty encountered in testing the procedure was that the size and orientation of the initial simplex had an effect on the speed of convergence and consequently several initial step-lengths and several arrangements of the initial simplex were used for each trial (the arrangements consisted of two forms of the initial simplex, regular, as in Spendley *et al.*'s original method, and axial from the starting point, combined with several orientations). The first set of trials investigated the different strategies (values of α, β and γ) of the simplex method of minimization, and the second compared the results for the best strategy with those of Powell (1964), which are among the best previously obtained.

An initial trial with function (1) used all combinations of $\alpha = \frac{1}{2}, \frac{2}{3}, 1$; $\beta = \frac{1}{4}, \frac{1}{3}, \frac{1}{2}$; $\gamma = 2, 3, 4$; and initial step-lengths $\frac{1}{2}$, 1, and 2. The main result was that the lower values of α and β gave generally slower convergence. In a second trial with function (1) six main strategies $\alpha = 1$, $\beta = \frac{1}{2}$ or $\frac{1}{3}$, and $\gamma = 2, 3, 4$ with two additional strategies $(\frac{2}{3}, \frac{1}{4}, 3)$ and $(\frac{2}{3}, \frac{1}{3}, 4)$ were all tried for three initial step-lengths $\frac{1}{2}$, 1, 2, and eight arrangements of the initial simplex. This trial showed that the additional strategies were more variable in performance and, on the average, slower to converge than the six main strategies. These six strategies gave very similar results, as was shown by an analysis of variance of the number of function evaluations, the mean square for strategies being 197 compared with a residual mean square (after removing size and orientation effects) of 463.

Using function (2) five strategies were tried: $\alpha = 1$, $\beta = \frac{1}{2}$, $\gamma = 2$, 3 or 4, and $\alpha = 1$, $\beta = \frac{1}{3}$ or $\frac{1}{4}$, $\gamma = 3$. The latter two strategies gave very variable results, as is indicated in Table 1 which shows the mean and minimum numbers of evaluations required for convergence over eight arrangements.

Analyzing the results for the best three strategies gave a standard error of difference between the strategy mean numbers of $\pm 7{\cdot}0$ (based on 70 degrees of freedom), showing that the results for the three strategies are all significantly different at the 95% level.

Using function (3) for the same trial confirmed these conclusions, the last two strategies converging to false minima on several occasions.

It was clear, therefore, that the simple strategy $\alpha = 1$, $\beta = \frac{1}{2}$, $\gamma = 2$ was the best, and these values were used in comparing the method with Powell's (1964) technique.

Table 1

Mean and minimum numbers of evaluations for function 2

MEAN NUMBER

STEP-LENGTH	STRATEGY (1, ½, 2)	(1, ½, 3)	(1, ½, 4)	(1, ⅓, 2)	(1, ¼, 2)
0·25	225	234	282	255	344
0·5	210	234	254	300	335
1	216	229	260	283	343
2	216	239	250	264	253
4	226	241	251	249	347
Mean	219	235	259	270	322

MINIMUM NUMBER

0·25	191	205	241	172	189
0·5	181	181	190	190	233
1	180	191	210	150	216
2	180	194	201	174	117
4	184	170	217	152	217
Mean	183	188	212	168	194

Comparison with Powell's results

For this comparison we used the same convergence criterion; smaller values of the criterion would not be justifiable when it is remembered that most functions met with in practice are likely to have a rounding-off error of this order. We used all three functions with a range of initial step-lengths 0·1(0·1)1(0·2)3 and eight initial simplex arrangements. Table 2 shows the results of these trials.

Apart from the smaller step-lengths the effect of step-length on number of evaluations required is not very large and it is reasonable to use the mean number of evaluations over all the unasterisked step-lengths for comparison with other methods. The mean values obtained are 144 evaluations for function (1), 216 for function (2), and 228 for function (3). Powell's results for functions (1) and (2) give the evaluations required to reach our mean final value of $2{\cdot}5 \times 10^{-9}$ as 150 and 235 respectively.

An EMA version of Powell's method was written and for function (3) it was found that the initial step-length had a considerable effect on the speed of convergence. Furthermore the optimum initial step-lengths for the two methods were different. For initial step-lengths in the range 0·1 to 0·95, Powell's method gave numbers of function values between 177 and 375 (sample of six).

Table 2

ean number of evaluations required for convergence for different step-lengths for functions (1), (2) and (3)

STEP-LENGTH	FUNCTION (1)	(2)	(3)
0·1	165*	262*	310*
0·2	161*	220	256
0·3	159*	214	276
0·4	162*	215	256
0·5	143	210	230
0·6	142	206	268
0·7	151	210	264
0·8	150	231	231
0·9	157	223	236
1·0	147	216	213
1·2	145	217	183
1·4	145	208	194
1·6	154	217	188
1·8	138	225	213
2·0	144	216	207
2·2	130†	197	206
2·4	128	224	224
2·6	134	223	209
2·8	135	241	233
3·0	148	191	234

See text.
Omitting arrangements where the true minimum was a vertex of the initial simplex.

ere appears, therefore, to be little difference between two methods to this degree of accuracy as far as r evidence goes. However, there are indications that, h less severe convergence criteria, greater differences uld have been found. This is suggested by the ogress of the two methods on three typical runs, as own in Table 3; for all three functions our method ablishes an initial advantage over Powell's. For ction (2) this advantage is maintained approximately the same level for the remainder of the run; for ctions (1) and (3) the advantage is further increased to the final stage when Powell's method rapidly ses the gap.

ect of number of variables

o investigate the effect of increasing the number of iables, the sum of fourth powers was used as a test ction, starting at (1, 1, 1, . . ., 1). For functions with o 10 variables it was found that the relationship ween the number of variables k and the mean number valuations N for convergence (using a final value of roximately $2 \cdot 5 \times 10^{-9}$) was well described by

$$N = 3 \cdot 16(k + 1)^{2 \cdot 11}.$$

Discussion

The method presented in this paper differs from most previously put forward in being based neither on gradients (first-order derivatives) nor on quadratic forms (second-order derivatives). As can be seen in Fig. 1, the algebraic operations on the x_i are all linear (e.g. as in determining P from P_h and $\bar{P}$), while those on the y_i are concerned with finding the maximum or minimum of a finite set of quantities. This latter type of operation is non-linear and accounts for the ability of the method to locate a minimum with arbitrary accuracy. The remark of Spendley *et al.* (1962) that "Continued application of the simplex procedure, with progressively reduced step size, is inherently as self-defeating as any other linear technique," is thus somewhat misleading, since the use made of the simplex at any stage is not to estimate parameters in a regression equation (in which case only a linear model could be fitted) but to guide the direction of the next move.

Out method is highly opportunist, in that the least possible information is used at each stage and no account is kept of past positions. No assumptions are made about the surface except that it is continuous and has a unique minimum in the area of the search. It might thus be expected that when the curvature of the landscape (as measured by the Hessian matrix of second-order partial derivatives) is changing rapidly, the present method will do well when compared with methods which depend on arguments applicable to quadratic forms. Conversely in the neighbourhood of a minimum, when the Hessian matrix is relatively stable, it may do worse. This expectation is borne out by the results obtained on the test functions. However, the positions of minima are often needed with only limited accuracy, so that final rapid convergence is not essential. This is especially true in statistical problems where the surface may be, e.g., a sum of squares of residuals, and the position of the minimum is a function of the random errors in the observations. An important property of our method is that it will converge even when the initial simplex straddles two or more valleys, a property which is not shared by, e.g., Powell's method.

A general problem encountered by all minimization methods is that of false convergence at a point other than the minimum. This difficulty has been found in using the simplex method on a four-dimensional surface having a long, curved, valley with extremely steep sides; along the valley bottom the function varies considerably compared with the accuracy to which the minimum function value is required. On meeting the valley, the simplex may need to undergo a drastic change in both size and shape, in the course of which the variation between the function values at the simplex vertices may become small even when compared with a convergence criterion of one-tenth of the required accuracy. Merely refining the convergence criterion would often involve needless evaluations, and in other more extreme cases could still be ineffective.

Table 3

Values of the three functions on 3 typical runs, for simplex method and Powell's (1964) method

NUMBER OF EVALUATION	FUNCTION (1)		(2)		(3)	
	Simplex	Powell	Simplex	Powell	Simplex	Powell
1	$2{\cdot}4 \times 10^{1}$	$2{\cdot}4 \times 10^{1}$	$2{\cdot}2 \times 10^{2}$	$2{\cdot}2 \times 10^{2}$	$2{\cdot}5 \times 10^{3}$	$2{\cdot}5 \times 10^{3}$
20	$2{\cdot}6 \times 10^{0}$	$3{\cdot}6 \times 10^{0}$	$1{\cdot}2 \times 10^{1}$	$3{\cdot}1 \times 10^{1}$	$1{\cdot}9 \times 10^{0}$	$1{\cdot}9 \times 10^{2}$
40	$9{\cdot}7 \times 10^{-1}$	$2{\cdot}2 \times 10^{0}$	$3{\cdot}0 \times 10^{-1}$	$4{\cdot}2 \times 10^{0}$	$1{\cdot}2 \times 10^{0}$	$3{\cdot}2 \times 10^{1}$
60	$3{\cdot}8 \times 10^{-1}$	$1{\cdot}1 \times 10^{0}$	$8{\cdot}7 \times 10^{-2}$	$1{\cdot}5 \times 10^{0}$	$7{\cdot}6 \times 10^{-2}$	$7{\cdot}7 \times 10^{0}$
80	$8{\cdot}1 \times 10^{-2}$	$4{\cdot}6 \times 10^{-1}$	$1{\cdot}7 \times 10^{-2}$	$2{\cdot}3 \times 10^{-1}$	$3{\cdot}0 \times 10^{-2}$	$2{\cdot}0 \times 10^{0}$
100	$1{\cdot}3 \times 10^{-3}$	$1{\cdot}0 \times 10^{-1}$	$1{\cdot}2 \times 10^{-3}$	$6{\cdot}4 \times 10^{-3}$	$2{\cdot}0 \times 10^{-2}$	$7{\cdot}6 \times 10^{-1}$
120	$8{\cdot}4 \times 10^{-6}$	$1{\cdot}7 \times 10^{-2}$	$6{\cdot}7 \times 10^{-5}$	$1{\cdot}1 \times 10^{-3}$	$4{\cdot}4 \times 10^{-3}$	$3{\cdot}1 \times 10^{-1}$
140	$7{\cdot}7 \times 10^{-9}$	$1{\cdot}0 \times 10^{-4}$	$3{\cdot}9 \times 10^{-6}$	$1{\cdot}3 \times 10^{-4}$	$1{\cdot}3 \times 10^{-4}$	$2{\cdot}3 \times 10^{-1}$
150	$3{\cdot}3 \times 10^{-10}$	$1{\cdot}2 \times 10^{-9}$				
160			$1{\cdot}2 \times 10^{-6}$	$2{\cdot}1 \times 10^{-5}$	$1{\cdot}0 \times 10^{-6}$	$6{\cdot}6 \times 10^{-2}$
180			$5{\cdot}9 \times 10^{-7}$	$2{\cdot}1 \times 10^{-6}$	$1{\cdot}7 \times 10^{-7}$	$3{\cdot}6 \times 10^{-3}$
200			$2{\cdot}6 \times 10^{-8}$	$1{\cdot}3 \times 10^{-7}$	$3{\cdot}7 \times 10^{-8}$	$1{\cdot}1 \times 10^{-4}$
220			$9{\cdot}3 \times 10^{-10}$	$1{\cdot}6 \times 10^{-8}$	$2{\cdot}7 \times 10^{-10}$	$7{\cdot}2 \times 10^{-9}$

The values for Powell's method were obtained by logarithmic interpolation of the function values at the end of each iteration. Data for functions (1) and (2) from Powell (1964), data for function (3) from our EMA program of his method.

Powell (1964) suggested a more complex convergence criterion, for this general problem, based on perturbing the first minimum found and repeating the method to find a second minimum, followed by exploration along the line joining the two. An alternative technique, more suited to our convergence criterion in terms of function variation, is to continue after the first convergence for a prescribed number of evaluations, to test for convergence again and, if the second test proves successful, to compare the two "converged" function values. Only if these values are sufficiently close is convergence allowed.

The simplex method is computationally compact; on the Orion computer the basic routine (without final printing) contains less than 350 instructions, and the great majority of orders are additions and subtractions or simple logical orders. There are few multiplications, and no divisions at all except on entering and leaving the routine.

Copies of the routine, written in Extended Mercury Autocode, are available from the authors.

Appendix

The Hessian matrix at the minimum

The minimization method proposed, being independent of the properties of quadratic forms, does not yield any estimate of the Hessian matrix of second derivatives at the minimum. This matrix is, of course, the information matrix in statistical problems when the function being minimized is minus the log. likelihood, and its inverse is the sample variance–covariance matrix of the estimates. A convenient way of utilizing a quadratic surface to estimate the minimum when the simplex is close to that minimum was given by Spendley *et al.* (1962) and their method can be readily extended to give the required variance–covariance matrix of the estimates.

If the $(n + 1)$ points of the simplex in n dimensions are given by $P_0, P_1, \ldots P_n$, then Spendley *et al.* form the "half-way points" $P_{ij} = (P_i + P_j)/2$, $i \neq j$ and fit a quadratic surface to the combined set of $(n + 1)(n + 2)/2$ points. If the original points of the simplex are used to define a set of oblique axes with co-ordinates x_i, then the points may be taken as

$$(0, 0, 0, \ldots 0)$$
$$(1, 0, 0, \ldots 0)$$
$$(0, 1, 0, \ldots 0)$$
$$\ldots \quad \ldots \quad \ldots$$

and

$$(0, 0, 0, \ldots 1).$$

If the quadratic approximation to the function in the neighbourhood of the minimum is written as

$$y = a_0 + 2\Sigma a_i x_i + \Sigma b_{ij} x_i x_j,$$

or in vector form as

$$y = a_0 + 2a'x + x'Bx,$$

then the coefficients are estimated as

$$\begin{aligned} a_0 &= y_0 \\ a_i &= 2y_{0i} - (y_i + 3y_0)/2, & i &= 1, \ldots, n \\ b_{ii} &= 2(y_i + y_0 - 2y_{0i}), & i &= 1, \ldots, n \\ b_{ij} &= 2(y_{ij} + y_0 - y_{0i} - y_{0j}), & i &\neq j, \end{aligned}$$

where y_i is the function value at P_i and y_{ij} that at P_{ij}. The estimated minimum is then given by

$$x_{min} = -B^{-1}a$$

and the information matrix is just B.

If p_i denotes the co-ordinates of P_i in the original system, and if Q is the $n \times n$ matrix whose ith column is $p_i - p_0$, then the minimum is estimated to be at

$$\begin{aligned} p_{min} &= p_0 + Qx_{min} \\ &= p_0 - QB^{-1}a. \end{aligned}$$

The minimum value of the function is estimated to be

$$y_{min} = a_0 - a'B^{-1}a.$$

The information matrix in the original co-ordinate system is given by

$$(Q^{-1})'BQ^{-1}$$

so that the variance–covariance matrix is given by

$$QB^{-1}Q'.$$

If normal equal-variance independent errors are involved and the sum of squares of residuals is minimized, then this matrix must be multiplied by $2\sigma^2$, where as usual σ^2 would be estimated by $y_{min}/(N-n)$, N being the total number of observations, and n the number of parameters fitted.

In estimating B numerically it is necessary to steer a course between two hazards. In one the simplex is so small that $(y_{ij} + y_0 - y_{0i} - y_{0j})$ consists largely of rounding-off errors incurred in calculating the y's. In the other the simplex is so large that the quadratic approximation is poor, and the b's are correspondingly biased. If the method given in this paper is used, the former hazard will usually be the important one, and it may be necessary to enlarge the final simplex before adding the extra points. A possible way of doing this is to double the distance of each point P_i from the centroid until the corresponding function value exceeds that at the centroid by more than a given constant. The choice of this would depend on the rounding-off error attaching to the evaluation of the function, and would need to be at least 10^3 times that rounding error, if acceptable estim es of the b's were to be obtained.

References

FLETCHER, R., and POWELL, M. J. D. (1963). "A rapidly convergent descent method for minimization," *The Computer Journal*, Vol. 6, p. 163.

POWELL, M. J. D. (1962). "An iterative method for finding stationary value of a function of several variables," *The Computer Journal*, Vol. 5, p. 147.

POWELL, M. J. D. (1964). "An efficient method for finding the minimum of a function of several variables without calculating derivatives," *The Computer Journal*, Vol. 7, p. 155.

ROSENBROCK, H. (1960). "An automatic method for finding the greatest or least value of a function," *The Computer Journal*, Vol. 3, p. 175.

SPENDLEY, W., HEXT, G. R., and HIMSWORTH, F. R. (1962). "Sequential Application of Simplex Designs in Optimisation and Evolutionary Operation," *Technometrics*, Vol. 4, p. 441.

Correspondence

To the Editor,
The Computer Journal.

An impossible program

Sir,

A well-known piece of folk-lore among programmers holds that it is impossible to write a program which can examine any other program and tell, in every case, if it will terminate or get into a closed loop when it is run. I have never actually seen a proof of this in print, and though Alan Turing once gave me a verbal proof (in a railway carriage on the way to a Conference at the NPL in 1953), I unfortunately and promptly forgot the details. This left me with an uneasy feeling that the proof must be long or complicated, but in fact it is so short and simple that it may be of interest to casual readers. The version below uses CPL, but not in any essential way.

Suppose $T[R]$ is a Boolean function taking a routine (or program) R with no formal or free variables as its argument and that for all R, $T[R]$ = **True** if R terminates if run and that $T[R]$ = **False** if R does not terminate. Consider the routine P defined as follows

rec routine P

§L : **if** $T[P]$ **go to** L

Return §

If $T[P]$ = **True** the routine P will loop, and it will only terminate if $T[P]$ = **False**. In each case $T[P]$ has exactly the wrong value, and this contradiction shows that the function T cannot exist.

Yours faithfully,

C. STRACHEY.

Churchill College,
Cambridge.